

**SISTEM MONITORING METERAN AIR UNTUK PERUMAHAN
MENGGUNAKAN *MESH NETWORK* DENGAN PROTOKOL MQTT PADA
PLATFORM *NODE-RED***

SKRIPSI

MUHAMMAD RIDHO ABIDIN DAMANIK

201402058



**PROGRAM STUDI S1 TEKNOLOGI INFORMASI
FAKULTAS ILMU KOMPUTER DAN TEKNOLOGI INFORMASI
UNIVERSITAS SUMATERA UTARA
MEDAN
2025**

**SISTEM MONITORING METERAN AIR UNTUK PERUMAHAN
MENGGUNAKAN *MESH NETWORK* DENGAN PROTOKOL MQTT PADA
PLATFORM *NODE-RED***

SKRIPSI

**Diajukan untuk melengkapi tugas dan memenuhi syarat memperoleh ijazah
Sarjana Teknologi Informasi**

**MUHAMMAD RIDHO ABIDIN DAMANIK
201402058**



**PROGRAM STUDI S1 TEKNOLOGI INFORMASI
FAKULTAS ILMU KOMPUTER DAN TEKNOLOGI INFORMASI
UNIVERSITAS SUMATERA UTARA
MEDAN
2025**

PERSETUJUAN

Judul : SISTEM MONITORING METERAN AIR UNTUK PERUMAHAN MENGGUNAKAN *MESH NETWORK* DENGAN PROTOKOL MQTT PADA PLATFORM *NODE-RED*

Kategori : SKRIPSI

Nama Mahasiswa : MUHAMMAD RIDHO ABIDIN DAMANIK

Nomor Induk Mahasiswa : 201402058

Program Studi : SARJANA (S1) TEKNOLOGI INFORMASI

Fakultas : ILMU KOMPUTER DAN TEKNOLOGI INFORMASI UNIVERSITAS SUMATERA UTARA

Medan, 6 Januari 2025

Komisi Pembimbing:

Pembimbing 1,

Fahrurrozi Lubis, B.IT., M.Sc.IT
NIP. 198610122018052001

Pembimbing 2,

Dr. Romi Fadillah Rahmat B.Comp.Sc., M.Sc.
NIP. 198603032010121004

Diketahui/disetujui oleh

Program Studi S1 Teknologi Informasi



Dr. Dedy Ansandi, S.T., M.Kom
NIP. 197908312009121002

PERNYATAAN

SISTEM MONITORING METERAN AIR UNTUK PERUMAHAN
MENGGUNAKAN *MESH NETWORK* DENGAN PROTOKOL MQTT PADA
PLATFORM *NODE-RED*

SKRIPSI

Saya mengakui bahwa skripsi ini adalah hasil karya saya sendiri, kecuali beberapa kutipan dan ringkasan yang masing-masing telah disebutkan sumbernya.

Medan, 6 Januari 2025


Muhammad Ridho Abidin Damanik
201402058



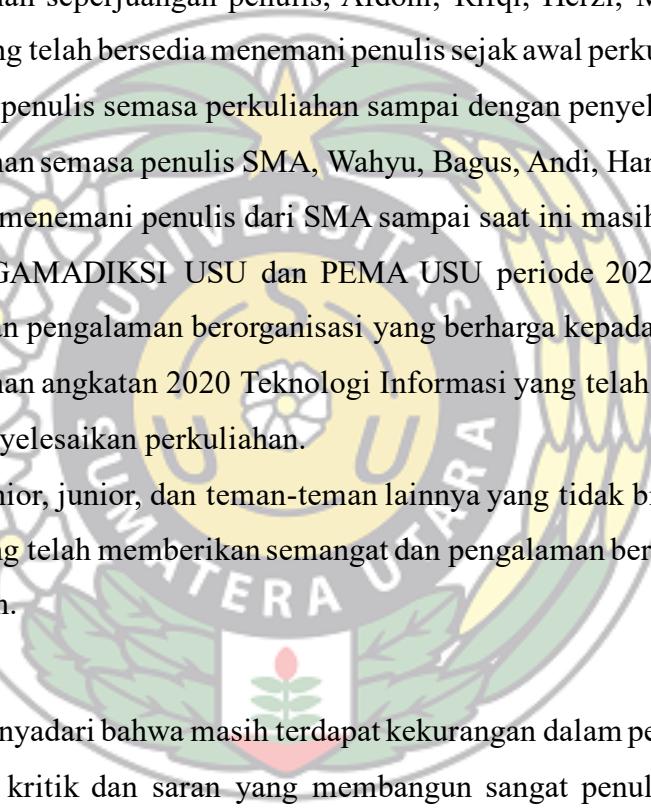
UCAPAN TERIMA KASIH

Puji dan Syukur yang tak terhingga penulis ucapkan atas kehadiran Allah Ta'ala Tuhan Yang Maha Esa yang senantiasa memberikan berkah dan petunjuk-Nya sehingga penulis dapat menyelesaikan skripsi ini sebagai salah satu syarat untuk menyelesaikan studi S1 Teknologi Informasi Fakultas Ilmu Komputer dan Teknologi Informasi di Universitas Sumatera Utara.

Penulis telah banyak mendapatkan bimbingan, doa dan dukungan dari berbagai pihak dalam menyelesaikan penulisan skripsi ini. Sehingga pada kesempatan ini penulis ingin mengucapkan terima kasih kepada:

1. Keluarga penulis, Ayahanda Zainal Abidin Damanik dan Ibunda Dra. Yusliana Hasibuan, selanjutnya kepada saudara kandung, yaitu Putri Rizki Fajar Abidin Damanik, Mahmud Akbar Abidin Damanik dan Siti Nasuha Abidin Damanik yang selalu memberikan semangat dan dukungan, serta doa terbaik untuk penulis.
2. Bapak Fahrurrozi Lubis B.IT., M.Sc.IT selaku Dosen Pembimbing 1 dan Bapak Romi Fadillah Rahmat B.Comp.Sc., M.Sc. selaku Dosen Pembimbing 2 yang telah banyak membimbing dan mengarahkan, serta memberikan kritik dan saran kepada penulis dalam proses penggerjaan skripsi ini.
3. Bapak Prof. Dr. Muryanto Amin, S.Sos., M.Si., selaku Rektor Universitas Sumatera Utara.
4. Ibu Dr. Maya Silvi Lidya B.Sc., M.Sc. dan Bapak Dr. Mohammad Andri Budiman S.T., M.Comp.Sc., M.E.M. selaku Dekan dan Wakil Dekan I Fasilkom-TI Universitas Sumatera Utara.
5. Bapak Dedy Arisandi ST., M.Kom. dan Ivan Jaya S.Si., M.Kom. selaku Ketua dan Sekretaris Program Studi S1 Teknologi Informasi Universitas Sumatera Utara.
6. Bapak Seniman, S.Kom., M.Kom. selaku Kepala Lab. Penelitian bidang Computer Systems, Network, and Security Fasilkom-TI Universitas Sumatera Utara.

7. Seluruh Dosen Program Studi S1 Teknologi Informasi Universitas Sumatera Utara yang telah memberikan ilmu yang bermanfaat selama masa perkuliahan penulis.
8. Seluruh Staff dan Pegawai Fasilkom-TI Universitas Sumatera Utara yang telah membantu segala urusan administrasi selama masa perkuliahan dan menyelesaikan skripsi penulis.
9. Sahabat-sahabat penulis, Reki, Taufik, Yusril, Dahrin dan Naufal yang telah setia menemani penulis sejak dibangku sekolah dan bersedia menjadi tempat berkeluh kesah dan orang yang tepat diajak *healing* bagi penulis.
10. Teman-teman seperjuangan penulis, Afdoni, Rifqi, Herzi, Mustafa, Bima dan Farhan, yang telah bersedia menemani penulis sejak awal perkuliahan dan banyak membantu penulis semasa perkuliahan sampai dengan penyelesaian skripsi.
11. Teman-teman semasa penulis SMA, Wahyu, Bagus, Andi, Hanum, Irma dan Siti, yang telah menemani penulis dari SMA sampai saat ini masih berteman dekat.
12. Pengurus GAMADIKSI USU dan PEMA USU periode 2022/2023 yang telah memberikan pengalaman berorganisasi yang berharga kepada penulis.
13. Teman-teman angkatan 2020 Teknologi Informasi yang telah menemani penulis dalam menyelesaikan perkuliahan.
14. Kepada senior, junior, dan teman-teman lainnya yang tidak bisa disebutkan satu persatu yang telah memberikan semangat dan pengalaman berharga selama masa perkuliahan.



Penulis menyadari bahwa masih terdapat kekurangan dalam penulisan skripsi ini, oleh karena itu, kritik dan saran yang membangun sangat penulis harapkan untuk penyempurnaan skripsi ini.

Medan, 6 Januari 2025

Penulis,



Muhammad Ridho Abidin Damanik
201402058

ABSTRAK

Dalam kehidupan manusia, air sebagai salah satu unsur yang paling penting karena sering dimanfaatkan baik untuk dikonsumsi maupun sebagai sarana kebersihan dan kebutuhan penunjang hidup lainnya. Pada umumnya di Indonesia terbagi dua jenis sumber air yang digunakan, yaitu sumber air yang didistribusikan melalui perpipaan dan selain dari perpipaan. Air yang didistribusikan melalui perpipaan dikelola oleh perusahaan penyedia air bersih. Perusahaan mendistribusikan air ke pelanggan melalui sambungan pipa yang dipasang meteran air sebagai alat penghitung penggunaan air pelanggan. Dengan penelitian ini, sistem yang dibuat dapat membantu perusahaan penyedia air melakukan *monitoring* penggunaan air pelanggan secara *realtime* dari jarak jauh dengan memanfaatkan teknologi *Internet of Things* (IoT) tanpa membebani pelanggan harus memiliki Wi-Fi di setiap rumah mereka. Penelitian yang dilakukan mendapatkan data aliran air dari sensor *waterflow* YF-S201 yang dihubungkan ke mikrokontroler ESP32. Memanfaatkan teknologi ESP-MESH untuk menerapkan *Wireless Mesh Network* sebagai protokol komunikasi sensor secara lokal tanpa harus terhubung ke jaringan internet dan memanfaatkan karakteristiknya berupa *self-configure* dan *self-healing* sehingga dapat mempermudah dalam membangun jaringan tiap *node*. Setelah data dikumpulkan secara lokal, kemudian data dikirim broker MQTT melalui *gateway* menggunakan modul *long range* (LoRa) SX1276 data ditransmit ke *receiver LoRa* untuk mencapai cakupan wilayah yang lebih luas. Data yang di-*publish* ke broker MQTT selanjutnya di-*subscribe* dan ditampilkan ke sebuah *dashboard* yang dibangun menggunakan platform *Node-RED*. Hasil penelitian menunjukkan tingkat *error* penghitungan volume air rata-rata sebesar 6.73%, *latency* pengiriman data dari *node* sensor ke *gateway* rata-rata sebesar 3029.8 ms dengan jarak hingga 50 meter.

Kata Kunci: Perusahaan Penyedia Air, *ESP32*, *Mesh Network*, *MQTT*, *LoRa*, *Node-RED*

WATER METER MONITORING SYSTEM FOR RESIDENTIAL USING MESH NETWORK WITH MQTT PROTOCOL ON NODE-RED PLATFORM

ABSTRACT

In human life, water is one of the most important elements because it is often used both for consumption and as a means of hygiene and other life-supporting needs. In general, in Indonesia there are two types of water sources used, namely water sources distributed through piping and other than piping. Water distributed through piping is managed by water supply companies. The company distributes water to customers through pipe connections that are installed with water meters as a means of calculating customer water usage. With this research, the system created can help water supply companies monitor customer water usage in real time remotely by utilizing Internet of Things (IoT) technology without burdening customers to have Wi-Fi in each of their homes. The research conducted obtained water flow data from the YF-S201 waterflow sensor connected to the ESP32 microcontroller. Utilizing ESP-MESH technology to implement Wireless Mesh Network as a sensor communication protocol locally without having to connect to the internet network and utilize its characteristics of self-configure and self-healing so that it can make it easier to build a network of each node. After the data is collected locally, then the data is sent to the MQTT broker through a gateway using the SX1276 long range (LoRa) module the data is transmitted to the LoRa receiver to achieve wider area coverage. Data published to the MQTT broker is then subscribed and displayed on a dashboard built using the Node-RED platform. The results showed an average water volume calculation error rate of 6.73%, data transmission latency from sensor nodes to the gateway averaged 3029.8 ms with a distance of up to 50 meters.

Keywords: Water Supply Company, ESP32, Mesh Network, MQTT, LoRa, Node-RED

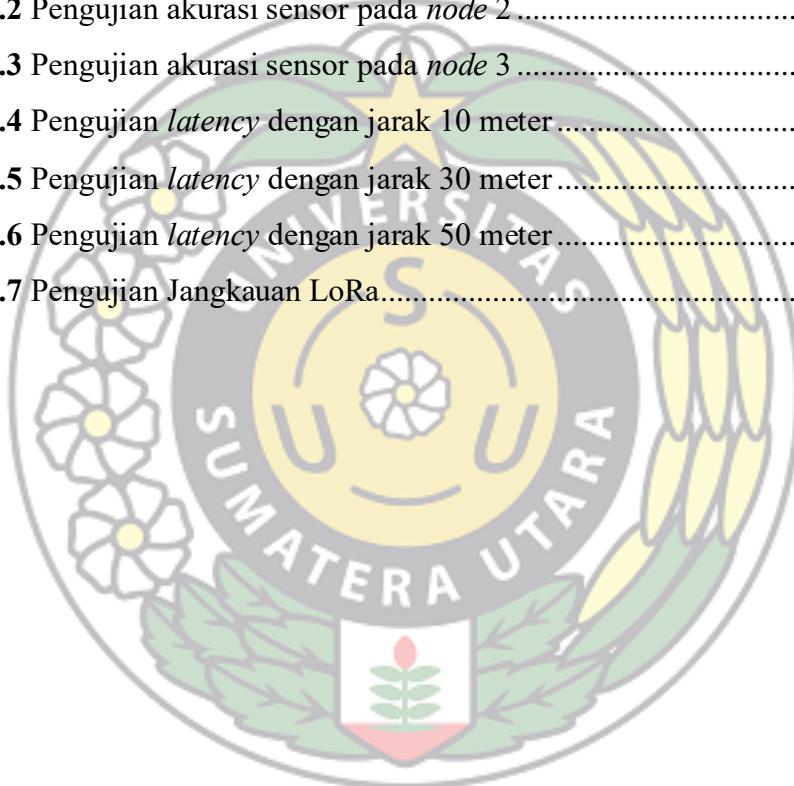
DAFTAR ISI

PERSETUJUAN	i
PERNYATAAN	ii
UCAPAN TERIMA KASIH	iii
ABSTRAK	v
ABSTRACT	vi
DAFTAR ISI.....	vii
DAFTAR TABEL.....	ix
DAFTAR GAMBAR.....	x
BAB 1 PENDAHULUAN	1
1.1. Latar Belakang	1
1.2. Rumusan Masalah.....	3
1.3. Tujuan Penelitian	4
1.4. Batasan Masalah	4
1.5. Manfaat Penelitian	4
1.6. Metodologi Penelitian	5
1.7. Sistematika Penulisan	6
BAB 2 LANDASAN TEORI	7
2.1 Meteran Air.....	7
2.2 <i>Mesh Network</i>	8
2.3 Sensor <i>Water Flow YF-S201</i>	9
2.4 ESP32 DevKit.....	9
2.5 Long Range (LoRa)	11
2.6 <i>Message Queuing Telemetry Transport (MQTT)</i>	12
2.7 <i>Node-RED</i>	14
2.8 Penelitian Terdahulu.....	16
2.9 Perbedaan Penelitian.....	21
BAB 3 ANALISIS DAN PERANCANGAN.....	22
3.1 Data yang Digunakan.....	22
3.2 Analisis Kebutuhan Sistem.....	22
3.2.1 Perangkat input.....	23

3.2.2 Pemrosesan	24
3.2.3 Output	25
3.3 Analisis Pemodelan Sistem.....	26
3.4 Perancangan Tampilan Antar Muka.....	27
3.4.1 Flow Sistem pada Platform Node-RED.....	27
3.4.2 Fitur UI Dashboard Node-RED.....	28
3.4.3 Design System Tampilan Dashboard.....	28
3.5 Perancangan Perangkat Keras (<i>Hardware</i>)	30
3.5.1 Perancangan Node Sensor	30
3.5.2 Perancangan ESP32 Gateway.....	32
BAB 4 IMPLEMENTASI DAN PENGUJIAN.....	34
4.1 Implementasi Sistem.....	34
4.2 Implementasi Perangkat Keras (<i>Hardware</i>).....	34
4.2.1 Rangkaian Protoype Sistem	34
4.2.2 Rangkaian Perangkat Node Sensor.....	36
4.2.3 Rangkaian Perangkat ESP32 Gateway	37
4.3 Implementasi Perangkat Lunak (<i>Software</i>).....	37
4.3.1 Program ESP32 dengan Arduino IDE	37
4.3.2 Konfigurasi Koneksi ESP-MESH pada ESP32 di Arduino IDE	41
4.3.3 Konfigurasi Koneksi LoRa SX1276 pada ESP32 di Arduino IDE	42
4.3.4 Konfigurasi Koneksi WiFi pada ESP32 di Arduino IDE	44
4.3.5 Konfigurasi MQTT pada ESP32 di Arduino IDE	44
4.3.6 Konfigurasi MQTT pada platform Node-RED	45
4.3.7 Format Data Komunikasi antar Node.....	46
4.4 Implementasi Tampilan Antar Muka.....	48
4.5 Pengujian Sistem	48
4.5.1 Pengujian Ketelitian Sensor Waterflow	49
4.5.2 Pengujian Koneksi Mesh.....	50
4.5.3 Pengujian Koneksi LoRa.....	54
BAB 5 KESIMPULAN DAN SARAN.....	55
5.1 Kesimpulan.....	55
5.2 Saran.....	55
DAFTAR PUSTAKA	56

DAFTAR TABEL

Tabel 2.1 Penelitian Terdahulu	18
Tabel 3.1 Rincian Perangkat Keras.....	30
Tabel 3.2 Konfigurasi Pin Sensor <i>water flow</i> ke <i>NodeMCU ESP32</i>	32
Tabel 3.3 Konfigurasi Pin modul LoRa SX1276 ke <i>NodeMCU ESP32</i>	33
Tabel 4.1 Pengujian akurasi sensor pada <i>node 1</i>	49
Tabel 4.2 Pengujian akurasi sensor pada <i>node 2</i>	49
Tabel 4.3 Pengujian akurasi sensor pada <i>node 3</i>	49
Tabel 4.4 Pengujian <i>latency</i> dengan jarak 10 meter	50
Tabel 4.5 Pengujian <i>latency</i> dengan jarak 30 meter	51
Tabel 4.6 Pengujian <i>latency</i> dengan jarak 50 meter	51
Tabel 4.7 Pengujian Jangkauan LoRa.....	54



DAFTAR GAMBAR

Gambar 2.1 Meteran Air PDAM	7
Gambar 2.2 Sensor <i>Waterflow</i> YF-S201	9
Gambar 2.3 Skema Pin <i>NodeMCU</i> ESP32.....	10
Gambar 2.4 Modul LoRa SX1276.....	11
Gambar 2.5 Protokol MQTT	14
Gambar 2.6 Tampilan Halaman Kerja Node-RED	15
Gambar 3.1 Arsitektur Umum	23
Gambar 3.2 Diagram <i>Use-case System</i>	26
Gambar 3.3 Diagram <i>Use-case User</i>	27
Gambar 3.4 Flow Node-RED untuk sistem monitoring	27
Gambar 3.5 Lo-Fi Desain Interface <i>Dashboard Platform Node-RED</i>	28
Gambar 3.6 <i>Typography / Text Style UI Dashboard</i>	29
Gambar 3.7 <i>Color Palette UI Dashboard</i>	29
Gambar 3.8 <i>Shape Style UI Dashboard</i>	30
Gambar 3.9 Skema rangkaian <i>node sensor</i>	32
Gambar 3.10 Skema rangkaian <i>gateway</i>	33
Gambar 4.1 Desain Rencana Implementasi Sistem Untuk Perumahan	35
Gambar 4.2 Prototipe Sistem Pengukuran Debit Air.....	35
Gambar 4.3 Perangkat <i>Node Sensor</i>	36
Gambar 4.4 <i>Pseudo-code</i> pembacaan sensor YF-S201	36
Gambar 4.5 Perangkat ESP32 <i>Gateway</i>	37
Gambar 4.6 File> Preference.....	38
Gambar 4.7 Preferences.....	38
Gambar 4.8 <i>Board Manager</i>	39
Gambar 4.9 <i>Install board ESP32 di Board Manager</i>	39
Gambar 4.10 <i>Manage Libraries</i>	40
Gambar 4.11 Inisiasi <i>library</i> pada program <i>node sensor</i>	40
Gambar 4.12 Inisiasi <i>library</i> pada program LoRa <i>transmitter</i>	41
Gambar 4.13 Inisiasi <i>library</i> pada program LoRa <i>receiver</i>	41
Gambar 4.14 Konfigurasi ESP-MESH program <i>node sensor</i>	41

Gambar 4.15 Inisiasi ESP-MESH program <i>node sensor</i>	42
Gambar 4.16 Konfigurasi ESP32 dengan LoRa SX1276	42
Gambar 4.17 Inisiasi LoRa SX1276.....	42
Gambar 4.18 Parameter <i>default</i> dari <i>RadioLib.h</i>	43
Gambar 4.19 Fungsi pada LoRa <i>transmitter</i>	43
Gambar 4.20 Fungsi pada LoRa <i>receiver</i>	43
Gambar 4.21 Kredensial WiFi	44
Gambar 4.22 Inisiasi koneksi ke WiFi.....	44
Gambar 4.23 Kredensial Server Broker MQTT	44
Gambar 4.24 Fungsi mengirim data ke server broker MQTT	45
Gambar 4.25 Node <i>mqtt in</i> di Node-RED sebagai <i>subscriber</i>	45
Gambar 4.26 Mengatur <i>Properties</i> node <i>mqtt in</i> di Node-RED	46
Gambar 4.27 Format Data yang dikirim dalam <i>Mesh Network</i>	47
Gambar 4.28 Tampilan antar muka <i>dashboard</i> monitoring air.....	48
Gambar 4.29 Grafik Latency pada Node 1 ke Gateway	52
Gambar 4.30 Grafik Latency pada Node 2 ke Gateway	53
Gambar 4.31 Grafik Latency pada Node 3 ke Gateway	53

PERNYATAAN

SISTEM MONITORING METERAN AIR UNTUK PERUMAHAN
 MENGGUNAKAN *MESH NETWORK* DENGAN PROTOKOL MQTT PADA
 PLATFORM *NODE-RED*

SKRIPSI

Saya mengakui bahwa skripsi ini adalah hasil karya saya sendiri, kecuali beberapa kutipan dan ringkasan yang masing-masing telah disebutkan sumbernya.

Medan, 6 Januari 2025


Muhammad Ridho Abidin Damanik
201402058

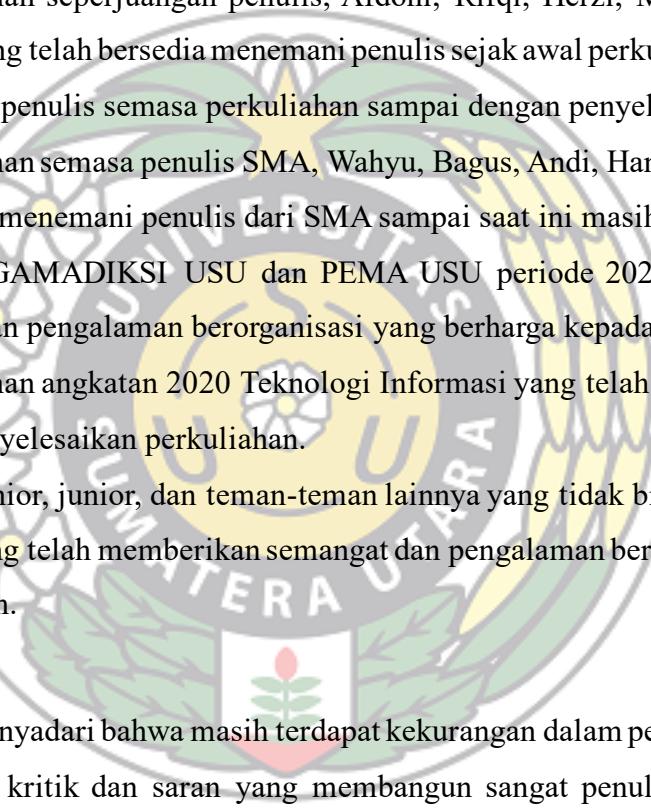
UCAPAN TERIMA KASIH

Puji dan Syukur yang tak terhingga penulis ucapkan atas kehadiran Allah Ta’ala Tuhan Yang Maha Esa yang senantiasa memberikan berkah dan petunjuk-Nya sehingga penulis dapat menyelesaikan skripsi ini sebagai salah satu syarat untuk menyelesaikan studi S1 Teknologi Informasi Fakultas Ilmu Komputer dan Teknologi Informasi di Universitas Sumatera Utara.

Penulis telah banyak mendapatkan bimbingan, doa dan dukungan dari berbagai pihak dalam menyelesaikan penulisan skripsi ini. Sehingga pada kesempatan ini penulis ingin mengucapkan terima kasih kepada:

1. Keluarga penulis, Ayahanda Zainal Abidin Damanik dan Ibunda Dra. Yusliana Hasibuan, selanjutnya kepada saudara kandung, yaitu Putri Rizki Fajar Abidin Damanik, Mahmud Akbar Abidin Damanik dan Siti Nasuha Abidin Damanik yang selalu memberikan semangat dan dukungan, serta doa terbaik untuk penulis.
2. Bapak Fahrurrozi Lubis B.IT., M.Sc.IT selaku Dosen Pembimbing 1 dan Bapak Romi Fadillah Rahmat B.Comp.Sc., M.Sc. selaku Dosen Pembimbing 2 yang telah banyak membimbing dan mengarahkan, serta memberikan kritik dan saran kepada penulis dalam proses penggerjaan skripsi ini.
3. Bapak Prof. Dr. Muryanto Amin, S.Sos., M.Si., selaku Rektor Universitas Sumatera Utara.
4. Ibu Dr. Maya Silvi Lidya B.Sc., M.Sc. dan Bapak Dr. Mohammad Andri Budiman S.T., M.Comp.Sc., M.E.M. selaku Dekan dan Wakil Dekan I Fasilkom-TI Universitas Sumatera Utara.
5. Bapak Dedy Arisandi ST., M.Kom. dan Ivan Jaya S.Si., M.Kom. selaku Ketua dan Sekretaris Program Studi S1 Teknologi Informasi Universitas Sumatera Utara.
6. Bapak Seniman, S.Kom., M.Kom. selaku Kepala Lab. Penelitian bidang Computer Systems, Network, and Security Fasilkom-TI Universitas Sumatera Utara.

7. Seluruh Dosen Program Studi S1 Teknologi Informasi Universitas Sumatera Utara yang telah memberikan ilmu yang bermanfaat selama masa perkuliahan penulis.
8. Seluruh Staff dan Pegawai Fasilkom-TI Universitas Sumatera Utara yang telah membantu segala urusan administrasi selama masa perkuliahan dan menyelesaikan skripsi penulis.
9. Sahabat-sahabat penulis, Reki, Taufik, Yusril, Dahrin dan Naufal yang telah setia menemani penulis sejak dibangku sekolah dan bersedia menjadi tempat berkeluh kesah dan orang yang tepat diajak *healing* bagi penulis.
10. Teman-teman seperjuangan penulis, Afdoni, Rifqi, Herzi, Mustafa, Bima dan Farhan, yang telah bersedia menemani penulis sejak awal perkuliahan dan banyak membantu penulis semasa perkuliahan sampai dengan penyelesaian skripsi.
11. Teman-teman semasa penulis SMA, Wahyu, Bagus, Andi, Hanum, Irma dan Siti, yang telah menemani penulis dari SMA sampai saat ini masih berteman dekat.
12. Pengurus GAMADIKSI USU dan PEMA USU periode 2022/2023 yang telah memberikan pengalaman berorganisasi yang berharga kepada penulis.
13. Teman-teman angkatan 2020 Teknologi Informasi yang telah menemani penulis dalam menyelesaikan perkuliahan.
14. Kepada senior, junior, dan teman-teman lainnya yang tidak bisa disebutkan satu persatu yang telah memberikan semangat dan pengalaman berharga selama masa perkuliahan.



Penulis menyadari bahwa masih terdapat kekurangan dalam penulisan skripsi ini, oleh karena itu, kritik dan saran yang membangun sangat penulis harapkan untuk penyempurnaan skripsi ini.

Medan, 6 Januari 2025

Penulis,



Muhammad Ridho Abidin Damanik
201402058

ABSTRAK

Dalam kehidupan manusia, air sebagai salah satu unsur yang paling penting karena sering dimanfaatkan baik untuk dikonsumsi maupun sebagai sarana kebersihan dan kebutuhan penunjang hidup lainnya. Pada umumnya di Indonesia terbagi dua jenis sumber air yang digunakan, yaitu sumber air yang didistribusikan melalui perpipaan dan selain dari perpipaan. Air yang didistribusikan melalui perpipaan dikelola oleh perusahaan penyedia air bersih. Perusahaan mendistribusikan air ke pelanggan melalui sambungan pipa yang dipasang meteran air sebagai alat penghitung penggunaan air pelanggan. Dengan penelitian ini, sistem yang dibuat dapat membantu perusahaan penyedia air melakukan *monitoring* penggunaan air pelanggan secara *realtime* dari jarak jauh dengan memanfaatkan teknologi *Internet of Things* (IoT) tanpa membebani pelanggan harus memiliki Wi-Fi di setiap rumah mereka. Penelitian yang dilakukan mendapatkan data aliran air dari sensor *waterflow* YF-S201 yang dihubungkan ke mikrokontroler ESP32. Memanfaatkan teknologi ESP-MESH untuk menerapkan *Wireless Mesh Network* sebagai protokol komunikasi sensor secara lokal tanpa harus terhubung ke jaringan internet dan memanfaatkan karakteristiknya berupa *self-configure* dan *self-healing* sehingga dapat mempermudah dalam membangun jaringan tiap *node*. Setelah data dikumpulkan secara lokal, kemudian data dikirim broker MQTT melalui *gateway* menggunakan modul *long range* (LoRa) SX1276 data ditransmit ke *receiver LoRa* untuk mencapai cakupan wilayah yang lebih luas. Data yang di-*publish* ke broker MQTT selanjutnya di-*subscribe* dan ditampilkan ke sebuah *dashboard* yang dibangun menggunakan platform *Node-RED*. Hasil penelitian menunjukkan tingkat *error* penghitungan volume air rata-rata sebesar 6.73%, *latency* pengiriman data dari *node* sensor ke *gateway* rata-rata sebesar 3029.8 ms dengan jarak hingga 50 meter.

Kata Kunci: Perusahaan Penyedia Air, *ESP32*, *Mesh Network*, *MQTT*, *LoRa*, *Node-RED*

WATER METER MONITORING SYSTEM FOR RESIDENTIAL USING MESH NETWORK WITH MQTT PROTOCOL ON NODE-RED PLATFORM

ABSTRACT

In human life, water is one of the most important elements because it is often used both for consumption and as a means of hygiene and other life-supporting needs. In general, in Indonesia there are two types of water sources used, namely water sources distributed through piping and other than piping. Water distributed through piping is managed by water supply companies. The company distributes water to customers through pipe connections that are installed with water meters as a means of calculating customer water usage. With this research, the system created can help water supply companies monitor customer water usage in real time remotely by utilizing Internet of Things (IoT) technology without burdening customers to have Wi-Fi in each of their homes. The research conducted obtained water flow data from the YF-S201 waterflow sensor connected to the ESP32 microcontroller. Utilizing ESP-MESH technology to implement Wireless Mesh Network as a sensor communication protocol locally without having to connect to the internet network and utilize its characteristics of self-configure and self-healing so that it can make it easier to build a network of each node. After the data is collected locally, then the data is sent to the MQTT broker through a gateway using the SX1276 long range (LoRa) module the data is transmitted to the LoRa receiver to achieve wider area coverage. Data published to the MQTT broker is then subscribed and displayed on a dashboard built using the Node-RED platform. The results showed an average water volume calculation error rate of 6.73%, data transmission latency from sensor nodes to the gateway averaged 3029.8 ms with a distance of up to 50 meters.

Keywords: Water Supply Company, ESP32, Mesh Network, MQTT, LoRa, Node-RED

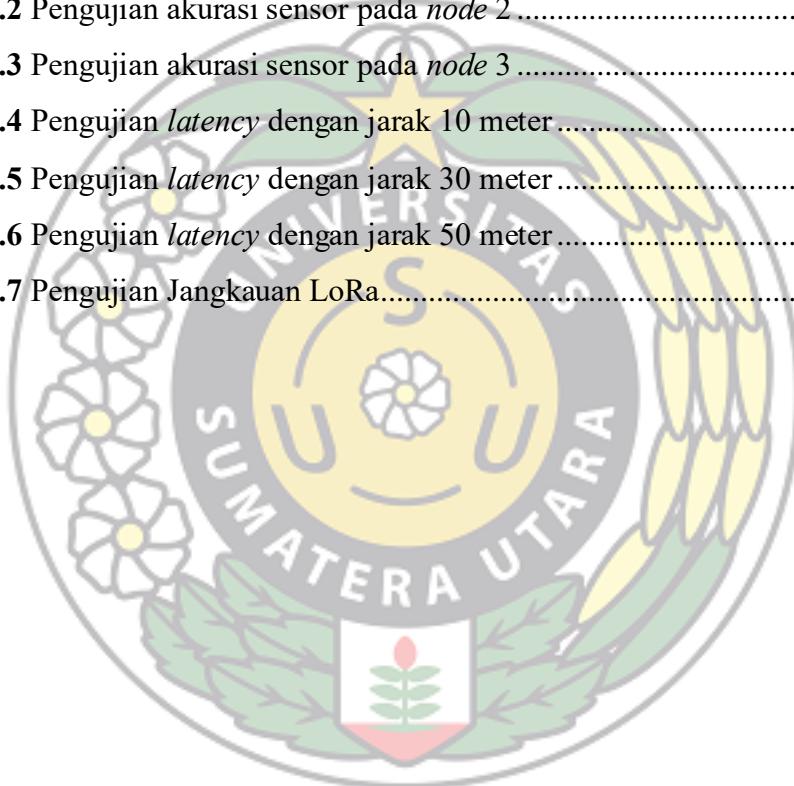
DAFTAR ISI

PERSETUJUAN	i
PERNYATAAN	ii
UCAPAN TERIMA KASIH	iii
ABSTRAK	v
ABSTRACT	vi
DAFTAR ISI.....	vii
DAFTAR TABEL.....	ix
DAFTAR GAMBAR.....	x
BAB 1 PENDAHULUAN	1
1.1. Latar Belakang	1
1.2. Rumusan Masalah.....	3
1.3. Tujuan Penelitian	4
1.4. Batasan Masalah	4
1.5. Manfaat Penelitian	4
1.6. Metodologi Penelitian	5
1.7. Sistematika Penulisan	6
BAB 2 LANDASAN TEORI	7
2.1 Meteran Air.....	7
2.2 <i>Mesh Network</i>	8
2.3 Sensor <i>Water Flow YF-S201</i>	9
2.4 ESP32 DevKit.....	9
2.5 Long Range (LoRa)	11
2.6 <i>Message Queuing Telemetry Transport (MQTT)</i>	12
2.7 <i>Node-RED</i>	14
2.8 Penelitian Terdahulu.....	16
2.9 Perbedaan Penelitian.....	21
BAB 3 ANALISIS DAN PERANCANGAN.....	22
3.1 Data yang Digunakan.....	22
3.2 Analisis Kebutuhan Sistem.....	22
3.2.1 Perangkat input.....	23

3.2.2 Pemrosesan	24
3.2.3 Output	25
3.3 Analisis Pemodelan Sistem.....	26
3.4 Perancangan Tampilan Antar Muka.....	27
3.4.1 Flow Sistem pada Platform Node-RED.....	27
3.4.2 Fitur UI Dashboard Node-RED.....	28
3.4.3 Design System Tampilan Dashboard.....	28
3.5 Perancangan Perangkat Keras (<i>Hardware</i>)	30
3.5.1 Perancangan Node Sensor	30
3.5.2 Perancangan ESP32 Gateway.....	32
BAB 4 IMPLEMENTASI DAN PENGUJIAN.....	34
4.1 Implementasi Sistem.....	34
4.2 Implementasi Perangkat Keras (<i>Hardware</i>).....	34
4.2.1 Rangkaian Protoype Sistem	34
4.2.2 Rangkaian Perangkat Node Sensor.....	36
4.2.3 Rangkaian Perangkat ESP32 Gateway	37
4.3 Implementasi Perangkat Lunak (<i>Software</i>).....	37
4.3.1 Program ESP32 dengan Arduino IDE	37
4.3.2 Konfigurasi Koneksi ESP-MESH pada ESP32 di Arduino IDE	41
4.3.3 Konfigurasi Koneksi LoRa SX1276 pada ESP32 di Arduino IDE	42
4.3.4 Konfigurasi Koneksi WiFi pada ESP32 di Arduino IDE	44
4.3.5 Konfigurasi MQTT pada ESP32 di Arduino IDE	44
4.3.6 Konfigurasi MQTT pada platform Node-RED	45
4.3.7 Format Data Komunikasi antar Node.....	46
4.4 Implementasi Tampilan Antar Muka.....	48
4.5 Pengujian Sistem	48
4.5.1 Pengujian Ketelitian Sensor Waterflow	49
4.5.2 Pengujian Koneksi Mesh.....	50
4.5.3 Pengujian Koneksi LoRa.....	54
BAB 5 KESIMPULAN DAN SARAN.....	55
5.1 Kesimpulan.....	55
5.2 Saran.....	55
DAFTAR PUSTAKA	56

DAFTAR TABEL

Tabel 2.1 Penelitian Terdahulu	18
Tabel 3.1 Rincian Perangkat Keras.....	30
Tabel 3.2 Konfigurasi Pin Sensor <i>water flow</i> ke <i>NodeMCU ESP32</i>	32
Tabel 3.3 Konfigurasi Pin modul LoRa SX1276 ke <i>NodeMCU ESP32</i>	33
Tabel 4.1 Pengujian akurasi sensor pada <i>node 1</i>	49
Tabel 4.2 Pengujian akurasi sensor pada <i>node 2</i>	49
Tabel 4.3 Pengujian akurasi sensor pada <i>node 3</i>	49
Tabel 4.4 Pengujian <i>latency</i> dengan jarak 10 meter	50
Tabel 4.5 Pengujian <i>latency</i> dengan jarak 30 meter	51
Tabel 4.6 Pengujian <i>latency</i> dengan jarak 50 meter	51
Tabel 4.7 Pengujian Jangkauan LoRa.....	54



DAFTAR GAMBAR

Gambar 2.1 Meteran Air PDAM	7
Gambar 2.2 Sensor <i>Waterflow</i> YF-S201	9
Gambar 2.3 Skema Pin <i>NodeMCU</i> ESP32.....	10
Gambar 2.4 Modul LoRa SX1276.....	11
Gambar 2.5 Protokol MQTT	14
Gambar 2.6 Tampilan Halaman Kerja Node-RED	15
Gambar 3.1 Arsitektur Umum	23
Gambar 3.2 Diagram <i>Use-case System</i>	26
Gambar 3.3 Diagram <i>Use-case User</i>	27
Gambar 3.4 Flow Node-RED untuk sistem monitoring	27
Gambar 3.5 Lo-Fi Desain Interface <i>Dashboard Platform Node-RED</i>	28
Gambar 3.6 <i>Typography / Text Style UI Dashboard</i>	29
Gambar 3.7 <i>Color Palette UI Dashboard</i>	29
Gambar 3.8 <i>Shape Style UI Dashboard</i>	30
Gambar 3.9 Skema rangkaian <i>node sensor</i>	32
Gambar 3.10 Skema rangkaian <i>gateway</i>	33
Gambar 4.1 Desain Rencana Implementasi Sistem Untuk Perumahan	35
Gambar 4.2 Prototipe Sistem Pengukuran Debit Air.....	35
Gambar 4.3 Perangkat <i>Node Sensor</i>	36
Gambar 4.4 <i>Pseudo-code</i> pembacaan sensor YF-S201	36
Gambar 4.5 Perangkat ESP32 <i>Gateway</i>	37
Gambar 4.6 File> Preference.....	38
Gambar 4.7 Preferences.....	38
Gambar 4.8 <i>Board Manager</i>	39
Gambar 4.9 <i>Install board ESP32 di Board Manager</i>	39
Gambar 4.10 <i>Manage Libraries</i>	40
Gambar 4.11 Inisiasi <i>library</i> pada program <i>node sensor</i>	40
Gambar 4.12 Inisiasi <i>library</i> pada program LoRa <i>transmitter</i>	41
Gambar 4.13 Inisiasi <i>library</i> pada program LoRa <i>receiver</i>	41
Gambar 4.14 Konfigurasi ESP-MESH program <i>node sensor</i>	41

Gambar 4.15 Inisiasi ESP-MESH program <i>node sensor</i>	42
Gambar 4.16 Konfigurasi ESP32 dengan LoRa SX1276	42
Gambar 4.17 Inisiasi LoRa SX1276.....	42
Gambar 4.18 Parameter <i>default</i> dari <i>RadioLib.h</i>	43
Gambar 4.19 Fungsi pada LoRa <i>transmitter</i>	43
Gambar 4.20 Fungsi pada LoRa <i>receiver</i>	43
Gambar 4.21 Kredensial WiFi	44
Gambar 4.22 Inisiasi koneksi ke WiFi.....	44
Gambar 4.23 Kredensial Server Broker MQTT	44
Gambar 4.24 Fungsi mengirim data ke server broker MQTT	45
Gambar 4.25 Node <i>mqtt in</i> di Node-RED sebagai <i>subscriber</i>	45
Gambar 4.26 Mengatur <i>Properties</i> node <i>mqtt in</i> di Node-RED	46
Gambar 4.27 Format Data yang dikirim dalam <i>Mesh Network</i>	47
Gambar 4.28 Tampilan antar muka <i>dashboard</i> monitoring air.....	48
Gambar 4.29 Grafik Latency pada Node 1 ke Gateway	52
Gambar 4.30 Grafik Latency pada Node 2 ke Gateway	53
Gambar 4.31 Grafik Latency pada Node 3 ke Gateway	53



BAB 1

PENDAHULUAN

1.1. Latar Belakang

Sumber utama kehidupan bagi semua organisme di bumi adalah air. Tanpa air, tidak ada makhluk hidup yang dapat bertahan hidup. Khususnya manusia membutuhkan air bersih untuk minum dan kebersihan diri. Bagi sebagian besar masyarakat Indonesia, ada dua jenis sumber air: yang disalurkan melalui pipa dan yang tidak melalui pipa. Sumber air perpipaan dikelola oleh PDAM, dan non-perpipaan dikelola oleh masyarakat, kelompok maupun individu (Dian, 2007).

Dalam proses mendistribusikan air, perusahaan penyedia air seperti PDAM biasanya menggunakan meteran air yang fungsinya untuk menghitung penggunaan air dipakai, umumnya bisa dijumpai di perumahan, perkantoran maupun industri yang menjadi pelanggan mereka (Maulidin *et al.*, 2020). Populasi yang terus bertambah tentu akan meningkatkan kebutuhan air, tetapi tidak sejalan dengan meningkatnya pemahaman masyarakat akan pentingnya menggunakan air bersih secara efisien. Terkadang, orang masih tidak tahu berapa banyak air yang mereka gunakan dalam sehari, sehingga menggunakan air dengan boros secara tidak sengaja.

Saat ini, PDAM masih melakukan pengecekan dan pemantauan penggunaan air secara manual dengan mengunjungi setiap rumah pelanggan satu per satu, mencatat setiap penggunaan, dan kemudian menghitung biaya tagihan penggunaan air setiap bulannya. Meteran air analog masih digunakan oleh PDAM saat ini. Meteran air ini memiliki sejumlah masalah yang mungkin timbul, seperti informasi yang tidak memadai pada meteran air manual, ketidakakuratan dalam proses pencatatan, dan kesalahan dalam proses input data pencatatan. Akibatnya, sulit untuk menentukan apakah ada masalah dengan peralatan yang digunakan (Paksi, *et al.*, 2021).

Dilansir dari *databoks*, Kementerian PUPR menyebutkan terdapat sebanyak 387 PDAM di seluruh wilayah Indonesia pada tahun 2020. Dibanding tahun 2019, jumlah ini meningkat 1,8% yaitu sebanyak 380 PDAM dan jumlah ini diperkirakan akan terus bertambah. Pemanfaatan suatu sistem monitoring meteran air dengan memanfaatkan teknologi Internet of Thing akan sangat diperlukan untuk menunjang efisiensi kinerja perusahaan air seperti PDAM dan bagi pelanggan dapat mengupayakan penghematan

penggunaan air karena mereka dapat mengetahui air yang digunakan secara *real-time* dan jumlah biaya yang dikenakan kepada mereka.

Dalam sistem monitoring ini, sinyal akan dikirim dari perangkat sensor ke broker MQTT melalui *gateway* kemudian ditampilkan pada sebuah *dashboard*. Teknologi *Wireless Mesh Network* digunakan untuk mengumpulkan data lokal yang kemudian akan dikirim ke *server* melalui *gateway*. *Wireless Mesh Network* (WMN) dapat digunakan sebagai solusi untuk berbagai aplikasi seperti jaringan sementara selama bencana, perangkat IoT di pabrik, smart meter, dan jaringan rumah broadband (Brajesh, 2023). WMN menyediakan jaringan sensor nirkabel yang stabil di area yang luas dan dapat membentuk diri sendiri (*self-configure*) dan menyembuhkan diri (*self-healing*). WMN menawarkan konektivitas yang fleksibel namun andal, membuatnya cocok untuk lingkungan yang tidak merata terhadap akses internet.

Penelitian tentang sistem monitoring meteran aliran air berbasis IoT sudah semakin diminati dan terus mengalami perkembangan. Penelitian oleh Ria Sood, Manjit Kaur dan Hemant Lenka (2013) dengan judul “*Design And Development Of Automatic Water Flow Meter*” telah melakukan penelitian dengan tujuan untuk mengembangkan meter aliran air otomatis berbiaya rendah yang dapat menyediakan jumlah air yang tepat untuk irigasi tanpa pemborosan. Dengan metode yang melibatkan sensor *waterflow* dengan efek Hall G1/2 dengan rotor turbin, yang kecepatan rotasinya berubah sesuai dengan laju aliran air, dan mikrokontroler AT89S52 yang diprogram dengan perangkat lunak Keil. Hasilnya menunjukkan bahwa sistem ini mampu mengukur laju aliran air dengan akurasi tinggi, memberikan manfaat dalam penghematan air dan energi, serta memiliki potensi untuk menggantikan meter aliran air berbiaya tinggi yang ada.

I Made Nova Suardiana, et. al. (2017) melakukan penelitian dengan judul “Rancang Bangun Sistem Pembacaan Jumlah Konsumsi Air Pelanggan PDAM Berbasis Mikrokontroler ATMEGA328 Dilengkapi SMS” dengan tujuan penelitian yaitu mengembangkan sistem digital untuk menggantikan sistem analog agar dapat membaca konsumsi air pelanggan PDAM dengan lebih akurat. Teknik yang digunakan adalah dengan menggunakan mikrokontroler ATmega328, *Real Time Clock* (RTC) DS1307, sensor YFS201, dan modul GSM/GPRS IComSat v1.1-SIM900 untuk pengiriman data SMS. Sistem yang dibangun dapat mengkonversi pembayaran sesuai dengan model pembayaran PDAM di Kabupaten Gianyar, dan hasil penelitian

menunjukkan bahwa sensor YF-S201 dapat mengukur penggunaan air dengan varians sebesar 0,39%.

Dengan memanfaatkan Bot Telegram, penelitian tentang inovasi seperti ini juga telah dilakukan Maulidin, *et al.*, (2020). Sebuah Arduino Uno, sensor pengukur aliran, dan Bot Telegram digunakan dalam penelitian ini. Hasil dari penelitian ini menghasilkan sebuah sistem pemantauan yang akan dipasok melalui Bot Telegram dan mengukur keluaran air, laju aliran, dan biaya.

Pada tahun 2022 lalu, Lintang *et al.* juga melakukan penelitian dengan tujuan untuk memonitor penggunaan air secara *real-time* dengan menggunakan mikrokontroler ESP32, sensor aliran air YF-S201, dan topologi *mesh network*. Metode yang diterapkan melibatkan penghitungan debit air dan komunikasi data melalui protokol MQTT. Dan penelitian ini menghasilkan kesimpulan bahwa sistem dapat memonitor penggunaan air dengan error sebesar 15.5% dan membutuhkan waktu rata-rata 10.3 detik untuk mengirim data antar *node* dengan jarak 30 meter, serta 3.95 detik dari gateway ke broker.

Berdasarkan informasi latar belakang dan penelitian terdahulu yang telah disebutkan di atas, penulis akan melakukan penelitian dengan mengimplementasikan *Wireless Mesh Network* pada prototipe sistem monitoring meteran air untuk perumahan dengan protokol komunikasi MQTT dan ditampilkan dalam *dashboard* platform *Node-RED* sehingga dapat dipantau secara *real-time*. Sehingga penelitian ini diajukan dengan judul “SISTEM MONITORING METERAN AIR UNTUK PERUMAHAN MENGGUNAKAN MESH NETWORK DENGAN PROTOKOL MQTT PADA PLATFORM NODE-RED”.

1.2. Rumusan Masalah

Banyak perusahaan penyedia air minum yang masih menggunakan meteran air analog, yang mana mengharuskan petugas dari perusahaan mengunjungi rumah pelanggan secara berkala untuk memeriksa meteran air mereka. Praktik ini dapat menyebabkan sejumlah masalah, termasuk input data yang tidak akurat dan kesalahan pencatatan. Pelanggan sering mengeluhkan hal ini karena sistem meteran air yang kurang efektif dan tidak dapat diketahui secara *real-time*. Masalah ini berdampak negatif pada kepuasan pelanggan serta efektivitas operasional bisnis. Untuk mengatasi tantangan ini,

diperlukan sistem yang dapat melacak distribusi air secara *real-time* yang dapat ditampilkan dalam sebuah *dashboard*.

1.3. Tujuan Penelitian

Penelitian ini bertujuan untuk mengembangkan sistem yang memungkinkan perusahaan penyedia air untuk memantau distribusi air pada suatu cakupan area secara *real-time* melalui tampilan sebuah *dashboard*. Dengan mengimplementasikan *Wireless Mesh Network*, data pemakaian air akan dikumpulkan secara lokal lalu dikirimkan secara langsung ke *server* melalui *gateway* dan ditampilkan dalam sebuah *dashboard*, sehingga pengguna dapat melihat informasi tentang air yang telah didistribusikan dengan mudah. Dengan demikian, diharapkan sistem ini dapat membantu mengoptimalkan pengelolaan perusahaan penyedia air dan meningkatkan efisiensi dalam mendistribusikan air tanpa membebani pelanggan harus memiliki Wi-Fi di setiap rumah.

1.4. Batasan Masalah

Adapun untuk membuat penelitian ini mempunyai pembahasan yang terlalu luas, maka terdapat batasan-batasan dalam penelitian ini yang telah penulis tentukan dalam penelitian ini :

- 1) Implementasi *mesh network* menggunakan mikrokontroler *NodeMCU ESP32 DevKit*.
- 2) Sensor untuk mengurus aliran air yang digunakan yaitu sensor *Waterflow YF-S201*.
- 3) Data hasil monitoring penggunaan air dapat dilihat melalui tampilan *dashboard*.
- 4) Alat yang dibuat dalam tugas akhir ini adalah model *prototype* dari rancangan sebenarnya.

1.5. Manfaat Penelitian

Adapun manfaat yang didapat pada penelitian ini adalah sebagai berikut :

- 1) Dengan menggunakan sistem yang terintegrasi dengan sensor dan *Wireless Mesh Network*, pengguna dapat dengan efisien dan cepat memantau penggunaan air dari banyak titik.
- 2) Mengetahui kinerja sensor YF-S201 dalam menghitung aliran air yang melewatinya.

- 3) Mengetahui efisiensi waktu pengiriman data dengan memakai Topologi *Wireless Mesh Network*.
- 4) Hasil dari penelitian ini dapat dikembangkan dengan pendekatan yang serupa untuk perusahaan penyedia air.

1.6. Metodologi Penelitian

Terdapat beberapa tahapan yang dilakukan pada penelitian ini, diantaranya ialah sebagai berikut:

1. Studi Literatur

Dalam tahap studi literatur, penulis menghimpun berbagai informasi mengenai sensor *waterflow*, penerapan *mesh network* dengan memakai *NodeMCU ESP32* dan sistem pemantauan penggunaan air secara *real-time* dengan sumber dari baik internet, termasuk jurnal, skripsi, buku, makalah ilmiah, dan sumber lainnya.

2. Analisis Permasalahan

Selanjutnya untuk memahami penggunaan *mesh network* dalam sistem pemantauan air rumah tangga secara *real-time*, penulis akan menganalisis permasalahan dari informasi dan data yang telah dikumpulkan.

3. Perancangan Sistem

Langkah berikutnya melibatkan desain sistem, yang dimulai dengan menetapkan spesifikasi alat, membuat diagram arsitektur umum, mendesain peralatan, dan akhirnya memimplementasikan sistem lalu membuat menampilkan informasi dalam bentuk tampilan *dashboard*.

4. Implementasi

Pada tahap ini, sebuah sistem dibuat dengan menggunakan rancangan sistem yang telah dibuat pada tahap sebelumnya untuk mencapai tujuan penelitian.

5. Pengujian Sistem

Tahap ini akan menguji sistem untuk memastikan sistem yang telah dibangun sebelumnya dapat digunakan dengan baik untuk memantau penggunaan air secara *real time*.

6. Penyusunan Laporan

Untuk menyajikan temuan penelitian, peneliti pada tahap ini akan membuat laporan dan membuat dokumentasi prosedur penelitian.

1.7. Sistematika Penulisan

Penulisan skripsi ini menggunakan pendekatan sistematis yang dibagi menjadi lima bagian, yang masing-masing dijelaskan secara singkat sebagai berikut:

BAB 1: PENDAHULUAN

Bab satu terdiri dari latar belakang masalah dari penelitian, rumusan masalah, tujuan penelitian, batasan masalah, manfaat penelitian, metodologi penelitian, dan sistematika penulisan.

BAB 2: LANDASAN TEORI

Bab dua memuat penjelasan landasan teori yang digunakan pada penelitian yang berkaitan dengan meterai air beserta penjelasan terkait solusi IoT yang diberikan oleh penulis.

BAB 3: ANALISIS DAN PERANCANGAN SISTEM

Bab tiga menjelaskan arsitektur rumum dari metode yang digunakan untuk penelitian dan perancangan perangkat keras.

BAB 4: IMPLEMENTASI DAN PENGUJIAN SISTEM

Bab empat berisi penjabaran proses implementasi dari sistem yang telah dibangun dan dilakukan analisis dari hasil percobaan yang dilakukan untuk mengevaluasi kinerja sistem.

BAB 5: KESIMPULAN DAN SARAN

Bab lima memaparkan hasil penelitian yang telah dilakukan dan rekomendasi untuk penelitian lanjutan.

BAB 2

LANDASAN TEORI

2.1 Meteran Air

Meteran air adalah alat yang digunakan oleh perusahaan penyedia air bersih, seperti PDAM (Perusahaan Daerah Air Minum), untuk mengukur aliran air dari jaringan pipa PDAM ke pelanggan. Hal ini memungkinkan untuk mencatat jumlah air yang digunakan pelanggannya setiap hari.



Gambar 2.1 Meteran Air PDAM

Terdapat tiga jenis meteran air yang berbeda berdasarkan bagaimana meteran air PDAM digunakan di banyak wilayah di Indonesia (Akbar. R, 2023) :

a. **Meteran Air Magnetik**

Menerapkan prinsip elektromagnetik pada sistem yang dialiri gelombang listrik. Produk akhirnya adalah data digital yang sangat akurat dan memiliki tingkat kesalahan 0,5%, yang direkam pada setiap interval tertentu. Namun, alat ini sensitif terhadap kondisi tanah dan instalasi, dan harganya relatif mahal.

b. **Meteran Air Mekanis**

Alasan mengapa pengukur air mekanis sangat populer adalah karena harganya yang murah dan perawatannya yang mudah. Aliran air menggerakkan baling-baling, yang secara mekanis melekat pada alat penghitung. Kekurangannya adalah kurasinya yang kurang konsisten.

c. **Meteran Air Ultrasonik**

Mengirimkan gelombang ultrasonik ke dalam pipa terbuka atau tertutup adalah cara teknologi ultrasonik mengukur debit air. Perusahaan penyedia air seperti

PDAM di kota-kota besar, biasa menggunakan *smart water meter*, berkat kemajuan teknologi ini. Manfaatnya adalah pemantauan yang efisien dimungkinkan oleh temuan pengukuran sensor ultrasonik yang secara otomatis dikirim ke *data center* melalui sinyal GPRS (*General Packet Radio Service*).

2.2 *Mesh Network*

Mesh Network merupakan jaringan komunikasi nirkabel yang disusun dalam bentuk topologi mesh dengan kemampuan radio untuk menghubungkan antar *node* dan menyediakan konektivitas yang fleksibel dan andal (Davoli L. & Gianluigi F., 2022). Dalam *Wireless Mesh Network* (WMN), *node* bersifat *self-configurable* (mengkonfigurasi mandiri) dan *self-healable* (penyembuhan mandiri). *Node* yang dapat mengkonfigurasi secara mandiri lebih baik sehingga bisa meningkatkan kinerja sistem, sedangkan penyembuhan mandiri bisa membuat jaringan melakukan konfigurasi ulang jika ada penambahan dan penghapusan *node* dalam jaringan. Karena jumlah *node* dan data yang sangat besar, akan ada kesalahan yang tinggi toleransi dan degradasi dalam kinerja. Integrasi jaringan yang ada mengarah ke lebih kompleks. Dengan menangani kelemahan ini, kinerja WMN dapat ditingkatkan.

Di antara fitur-fitur yang membedakan WMN dari jaringan nirkabel konvensional lainnya adalah kemampuannya untuk memenuhi tujuan menjadi jaringan yang dapat diandalkan. Berikut ini adalah beberapa karakteristik WMN (Schneider Electric dalam Rofiqoh, 2015) :

- a. Kemampuan *node* jaringan untuk mengkonfigurasi dan membangun diri sendiri dikenal sebagai "self-configuration". Ketika perangkat dihidupkan dan menemukan topologi berubah, ia akan memberi tahu *node* tetangga alamatnya sehingga mereka dapat diidentifikasi di jaringan. Selain itu, ini mencakup pencarian rute otomatis tanpa perlu diintervensi dengan mempertimbangkan kualitas link, topologi, dan konektivitas.
- b. Karakteristik jaringan untuk memperbaiki dirinya sendiri dan dapat terus berfungsi jika satu atau beberapa *node* dihancurkan atau direlokasi. Inilah yang menimbulkan frasa "self-healing" digunakan.
- c. Tidak pernah ada *downtime* dalam *mesh network* berkat komunikasi *multihop*, yang memungkinkan *node* untuk berkomunikasi dengan *node* lain yang tidak

terhubung secara langsung melalui *node* perantara. Area cakupan dapat ditingkatkan dan ditambah dengan adanya karakteristik ini.

2.3 Sensor *Water Flow* YF-S201

Sensor *Water Flow* adalah alat yang berfungsi untuk mendeteksi dan mengukur air yang mengalir melalui pipa. Perangkat ini berperan penting dalam sistem manajemen air yang efisien dan berfungsi sebagai sensor utama dalam pengukuran aliran yang memberikan informasi untuk pengambilan keputusan operasional (Gudi, *et al.*, 2019).

Sensor *Water Flow* yang digunakan dalam penelitian ini ialah YF-S201. Sensor YF-S201 adalah sensor pengukuran aliran air dengan bekerja pada prinsip efek Hall magnetik terintegrasi yang menghasilkan pulsa listrik dengan setiap putaran, sensor ini dapat dihubungkan ke saluran air karena memiliki saluran masuk dan keluar.



Gambar 2.2 Sensor *Waterflow* YF-S201

(Sumber : Components101, 2021)

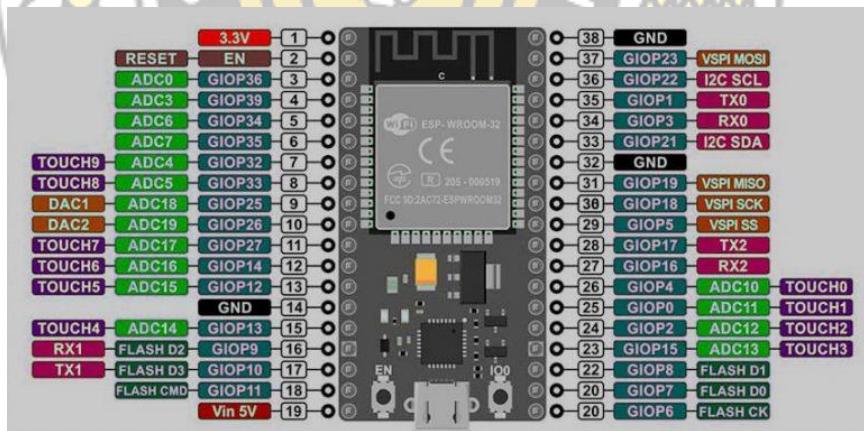
Roda rotor di dalam sensor menghitung volume cairan yang melewatkinya. Sensor YF-S201 memiliki panjang 5,6 cm dan memiliki diameter aliran masuk air $\frac{1}{2}$ inci, atau 1,25 cm. Daya, terminal *ground*, dan output analog sensor adalah tiga pin pada modul ini. Dengan tekanan kurang dari 2MPa, YF-S201 juga dapat membaca laju debit air dalam kisaran 1 ~ 30L/menit.

2.4 ESP32 DevKit

ESP32 DevKit adalah salah satu jenis mikrokontroler System On Chip (SoC) yang ampuh yang dilengkapi dengan banyak periferal, Bluetooth 4.2 mode ganda, dan Wi-Fi 802.11 b/g/n. Wi-Fi 802.11 b/g/n adalah protokol Wi-Fi standar dengan variasi

frekuensi dan kecepatan. Sebagai hasilnya, istilah “b/g/n” mengacu ke campuran berbagai standar Wi-Fi yang memungkinkan perangkat untuk berbicara satu sama lain. Dua core-nya, masing-masing dengan kecepatan terpisah hingga 240 MHz, membuatnya menjadi prosesor yang lebih unggul daripada ESP8266. Bersamaan dengan karakteristik ini, ESP32 juga memiliki memori flash sebesar 4MB dan memiliki 36 pin GPIO, bukan 17 pin GPIO pada model sebelumnya. Ia juga memiliki 16 saluran PWM per 16 pin.

Dua inti membentuk mikroprosesor ESP32 yang dikembangkan oleh *Espressif Systems*. Dimungkinkan untuk mengontrol setiap inti CPU secara terpisah. *SRAM on-chip* dengan ruang 520 KB disediakan untuk instruksi dan data. Beberapa *SoC*, seperti *ESP32-Wrover*, memiliki fitur tambahan 8 MB SPI PSRAM (*Pseudo Static RAM*) sebagai tambahan dari 4 MB flash SPI eksternal. Tergantung pada jenis *board*, ada banyak tipe seperti SPI, I2S, I2C, CAN, UART, Ethernet MAC, dan IR yang dapat digunakan. Sensor *Hall Effect*, sensor suhu, sensor sentuh, dan sensor *built-in* lainnya yang digunakan dalam Azure IoT dan Developer Kit, juga disertakan dalam perlengkapan standar. Skema pinout variasi mikrokontroler ESP32 ditampilkan pada Gambar 5 (Babiuch, 2019).



Gambar 2.3 Skema Pin *NodeMCU* ESP32

(Sumber : Electronic Clinic, 2021)

NodeMCU ESP32 tersedia dapat digunakan dalam membangun versi prototipe dari aplikasi IoT berbasis cloud, otomatisasi, aplikasi suara, aplikasi rumah pintar, dan banyak lagi. Dengan banyaknya variasi yang tersedia, seseorang dapat membuat *Embedded System* menggunakan mikrokontroler ESP32 atau memilih *Development Kit* tertentu.

2.5 Long Range (LoRa)

LoRa (Long Range) adalah modul komunikasi yang dibangun oleh *Semtech Corporation* dari Amerika Serikat dan dipatenkan oleh mereka. Modul ini menjadi banyak digunakan karena kemampuannya untuk mentransmisikan data jarak jauh dengan konsumsi daya yang rendah dibandingkan teknik modulasi lainnya. LoRa menggunakan teknologi *radio spread spectrum*, yang sangat efisien dalam penggunaan daya dalam mentransfer data, menjadikannya sangat ideal bahkan untuk membangun jaringan komunikasi di luar angkasa. Teknologi ini sangat berguna untuk sistem IoT, dengan banyak perusahaan yang telah menggunakan *gateway* LoRa untuk jaringan IoT.

LoRa beroperasi dalam pita frekuensi radio ISM (*Industrial, Scientific, and Medical*) yang tidak memerlukan lisensi, sehingga memudahkan dalam memenuhi regulasi frekuensi global yang kompleks (Firmansyah *et al.* 2020). LoRa dapat menjangkau lebih dari satu kilometer jika diatur dengan benar dan dalam lingkungan yang kondusif. Untuk *Wireless Sensor Networks* (WSN), yang membutuhkan transmisi data yang tahan terhadap gangguan, konsumsi daya yang rendah, dan jarak jauh, teknologi ini sangat membantu. Namun demikian, masih ada beberapa kekurangan pada LoRa, seperti kurangnya sistem penemuan *gateway* otonom untuk transmisi data dari *node* sensor.



Gambar 2.4 Modul LoRa SX1276

(Sumber : RSRobotica, 2024)

LoRa menggunakan tumpukan jaringan berlapis untuk mengoptimalkan kinerja (Li & Cao, 2022) :

- a. *Physical Layer (PHY)* : memanfaatkan modulasi spektrum spread chirp untuk menangani interferensi dan memastikan komunikasi jarak jauh.
- b. *Link Layer* : mengelola koneksi antara perangkat dan gateway jaringan.
- c. *Media Access Control (MAC) Layer* : LoRaWAN, lapisan MAC dari LoRa, mengadopsi topologi bintang untuk memfasilitasi komunikasi antara beberapa perangkat akhir dan *gateway*.

- d. *Application Layer* : mendukung berbagai aplikasi IoT dengan menyediakan kemampuan penanganan dan pemrosesan data yang diperlukan.

Performance metrics untuk LoRa meliputi jangkauan, *throughput*, konsumsi energi, dan keamanan. Mekanisme kecepatan data adaptif memungkinkan perangkat untuk menyesuaikan parameter secara dinamis seperti faktor penyebaran, kecepatan kode, dan frekuensi pembawa untuk mengoptimalkan kinerja dalam jaringan yang padat (Jouhari *et al.*, 2022).

LoRa dirancang untuk mendukung jaringan area luas berdaya rendah (LPWAN), sehingga sangat cocok untuk aplikasi Internet of Things (IoT). Salah satu keunggulannya adalah kemampuan untuk melakukan geolokasi tanpa biaya tambahan, dengan menggunakan teknologi berbasis jaringan yang tidak membutuhkan GPS, sehingga operasionalnya menjadi lebih efisien (Gu *et al.*, 2018). Dari segi biaya, LoRa menawarkan solusi yang ekonomis baik dari segi infrastruktur maupun operasional. Infrastruktur yang sederhana membuatnya lebih murah daripada teknologi lain seperti Wi-Fi atau seluler, sementara konsumsi daya yang rendah memungkinkan perangkat bertenaga baterai bertahan hingga 10-20 tahun.

LoRa juga memungkinkan sensor untuk berkomunikasi langsung dengan gateway tanpa memerlukan jaringan tambahan, sehingga lebih efisien. Selain itu, sebagai teknologi standar terbuka, LoRa mendukung interoperabilitas dengan berbagai produk dan sistem, sehingga memudahkan integrasi dengan aplikasi IoT yang sudah ada. Dengan jangkauan sinyal hingga 100 km dalam kondisi ideal dan kemampuan untuk beroperasi dalam kondisi *non-line-of-sight* (NLOS), LoRa sangat dapat diandalkan untuk komunikasi jarak jauh (Haritsa *et al.*, 2020). Keamanan data terjamin berkat enkripsi AES128 *end-to-end*. Tidak hanya itu, LoRa mampu menangani jutaan pesan per stasiun pangkalan, menjadikannya solusi yang kuat untuk mendukung banyak perangkat secara bersamaan di jaringan publik. Semua fitur ini membuat LoRa menjadi pilihan yang sangat baik untuk berbagai aplikasi IoT yang membutuhkan konektivitas yang andal, hemat energi, dan hemat biaya.

2.6 Message Queuing Telemetry Transport (MQTT)

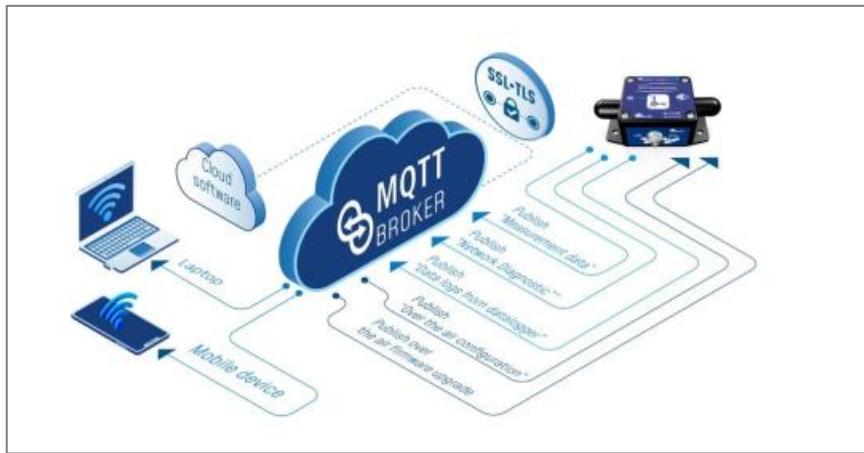
Message Queuing Telemetry Transport (MQTT) adalah sebuah protokol komunikasi yang dirancang untuk *machine to machine* (M2M) yang berjalan dengan arsitektur

TCP/IP yang sangat ringan dan efisien. Selain dapat mengirimkan data dengan sangat ringan, MQTT juga memiliki beberapa keunggulan yakni dapat mengirimkan data dengan *bandwidth* yang lebih kecil, konsumsi daya listrik yang rendah, komunikasi dan koneksi yang sangat tinggi serta variable yang banyak (Kurniati, S. et al., 2024). MQTT dikenal sebagai protokol pengiriman pesan yang terdiri dari tiga komponen mendasar, yaitu *publisher* atau *producer* (MQTT *client*), *broker* (MQTT *server*), *consumer* atau *subscriber* (MQTT *client*). *Publisher* adalah perangkat yang mengirimkan pesan ke *broker*. *Broker* berfungsi sebagai perantara yang menerima pesan dari *publisher* dan mengirimkannya kepada *subscriber* yang berlangganan topik tertentu. *Subscriber* adalah perangkat yang menerima pesan dari broker berdasarkan topik yang mereka langganan. Cara bagaimana MQTT bekerja yaitu dengan membuat koneksi dengan *broker* MQTT, klien dapat mempublikasikan pesan, berlangganan pesan tertentu, atau melakukan keduanya. Setelah broker MQTT menerima pesan, pelanggan yang tertarik akan dikirimkan pesan tersebut oleh *broker*.

MQTT menggunakan *port default* 1883 untuk komunikasi non-terenkripsi dan *port* 8883 untuk komunikasi dengan enkripsi TLS/SSL. MQTT tidak memiliki lapisan keamanan bawaan, tetapi dapat dikombinasikan dengan protokol keamanan tambahan, seperti TLS/SSL, untuk mengamankan komunikasi. Keamanan di implementasikan pada tingkat transportasi dengan menerapkan identitas, autentikasi, dan otorisasi antara klien dan broker menggunakan sertifikat SSL dan/atau kata sandi. Protokol ini mendukung tiga level Quality of Service: QoS 0 (*fire and forget*), QoS 1 (setidaknya sekali), dan QoS 2 (pastikan persis sekali). QoS (Kualitas Layanan) adalah teknologi apa pun yang mengelola lalu lintas data sehingga mengurangi latensi, *jitter*, dan *packet loss* di jaringan. Tujuan QoS menyediakan berbagai tingkat kualitas layanan berdasarkan kebutuhan layanan dalam jaringan.

- a. QoS 0 : *at most once*, di mana pesan dikirim dengan upaya terbaik dari jaringan TCP/IP. Kehilangan pesan atau terjadi duplikasi dapat terjadi.
- b. QoS 1 : *at least once*, dapat dipastikan pesan tersampaikan walaupun duplikasi dapat terjadi.
- c. QoS 2 : *exactly once*, dimana pesan dapat dipastikan tiba tepat satu kali.

Dengan adanya QoS memungkinkan pengiriman pesan yang dapat diandalkan dan menyesuaikan kebutuhan aplikasi.



Gambar 2.5 Protokol MQTT
(Sumber : Indobot.co.id, 2023)

MQTT digunakan secara luas dalam berbagai aplikasi IoT, termasuk *Smart Home*, otomasi industri, dan perawatan kesehatan. Jumlah hal yang ditambahkan ke jaringan semakin meningkat dari hari ke hari. Perangkat yang terhubung ini akan mencapai 50 miliar pada tahun 2020 (Sinde, S. *et al.*, 2016). Kemampuannya untuk mengelola komunikasi secara efisien di lingkungan dengan sumber daya terbatas membuatnya menjadi pilihan utama bagi para pengembang dan insinyur yang mengerjakan proyek IoT.

2.7 Node-RED

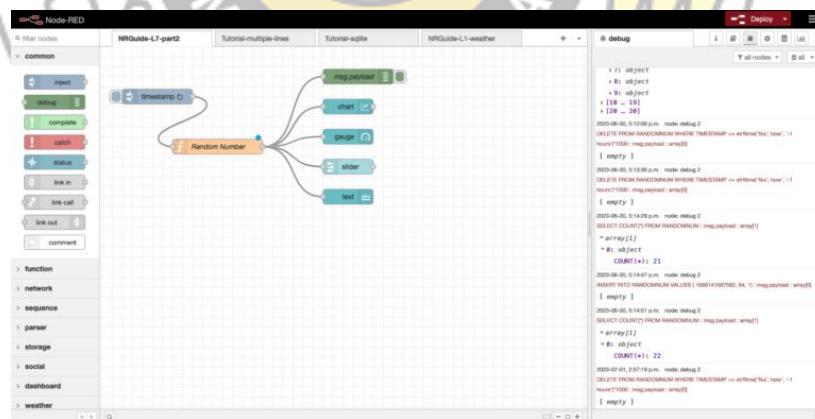
Node-RED adalah bahasa pemrograman *hybrid* yang menggabungkan pemrograman JavaScript dan blok fungsional yang dapat diatur dan dihubungkan dalam *node-flow*. Platform *Node-RED* dibuat pada tahun 2013 oleh *IBM UK's Emerging Technology Services Team*, yaitu Nick O'Leary dan Dave Conway-Jones. *OpenJS Foundation* mengakuisisi *Node-RED* pada tahun 2016 (Kodaki *et al.*, 2018). Agar *Node-RED* dapat berfungsi dengan baik, diperlukan runtime *node.js*. Pengguna dapat mengakses *Node-RED* dan *Node.js* menggunakan *browser* dan mengakses *localhost* atau alamat IP yang diterima ketika menjalankan perintah ‘*node-red*’ pada *command prompt* atau *terminal*, bersama dengan *port default* 1880 untuk melayani permintaan pengguna.

Flow-based programming (FBP) adalah inti dari pemrograman *Node-RED*. Alur kerja pada *Node-RED* dibuat dengan menyusun *Palette node* yang mewakili alur pemrograman secara keseluruhan. Dalam proyek Internet of Things, blok fungsional

disusun untuk mengelola tugas-tugas yang terlibat dalam mengolah data dan untuk menampilkannya dalam bentuk tampilan *dashboard*.

Node-RED secara tradisional beroperasi dengan cara terpusat, yang dapat membatasi dalam lingkungan sementara di mana *node* bergerak. Penelitian terbaru telah menunjukkan kelayakan migrasi alur kerja *Node-RED* ke lingkungan eksekusi yang terdesentralisasi menggunakan *Vector Symbolic Architecture* (VSA). Pendekatan ini memungkinkan penemuan dan interkoneksi layanan yang dinamis tanpa titik pusat kendali, sehingga cocok untuk jaringan *ad-hoc* seluler (Simpkin *et al.*, 2020). *Node-RED* beroperasi pada model *client-server* di mana *server* berjalan pada *Node.js*, sehingga memungkinkannya untuk menangani banyak koneksi dan aliran data secara efisien. Arsitekturnya terdiri dari:

- a. *Nodes* : Blok bangunan dasar yang mewakili input, output, dan unit pemrosesan. Setiap *node* menjalankan fungsi tertentu, seperti pengumpulan atau transformasi data.
- b. *Wires*: Koneksi antar *node* yang menentukan aliran data. Representasi visual ini menyederhanakan alur kerja yang kompleks.
- c. *Flow Editor* : Halaman antarmuka berbasis web untuk mendesain alur dengan menarik dan melepas *node* dan menghubungkannya dengan kabel.



Gambar 2.6 Tampilan Halaman Kerja Node-RED

(Sumber : noderedguide.com, 2023)

Arsitektur ini mendukung pengembangan modular dan meningkatkan penggunaan ulang, sehingga memudahkan pengembang untuk membuat solusi IoT yang dapat diskalakan (Hagino, T, 2020).

2.8 Penelitian Terdahulu

Penelitian yang berkaitan dengan meteran air dengan memanfaatkan *Internet of Thing* (IoT) pernah dilakukan sebelumnya. Ria *et al.* (2013) melakukan penelitian untuk menjelaskan desain dan pengembangan untuk meter aliran air yang dapat berfungsi secara otomatis serta membutuhkan biaya yang rendah agar bisa menggunakan air secara lebih hemat dan dengan penggunaan energi yang lebih kecil. Penelitian ini menyatakan bahwa meteran dengan menggunakan sensor Efek Hall G1/2 yang memiliki rotor turbin di dalamnya sehingga kecepatan rotasinya berubah sesuai dengan laju aliran air serta dilengkapi dengan mikrokontroler AT89S52, relay, pompa air, LCD dan beberapa komponen pasif menjadikan sistem ini lebih menarik, karena menyediakan operasi otomatis dengan akurasi tinggi dan metode yang paling terlalu murah untuk mengukur laju aliran air khususnya di bidang pertanian.

Alexandra *et al.* (2021) melakukan penelitian jaringan distribusi air yang berupa meteran air di kota Figueira da Foz, Portugal dengan memanfaatkan protokol komunikasi data *All for Everyone - Energy Aware* (AfE-EA) yang dikembangkan oleh CWJ Power Electronics dan merupakan salah satu jenis jaringan telekomunikasi nirkabel LPWAN (*Low-Power Wide-Area Network*). Hasil penelitian menunjukkan penerapan protokol AfE-EA dengan topologi mesh network telah berhasil membangun jaringan komunikasi dan memberikan pembacaan data dari semua meter air serta waktu pengembalian data yang singkat sehingga dengan protokol AfE-EA ini dapat diintegrasikan dengan utilitas publik lainnya untuk mendukung kota cerdas.

Penelitian lain dilakukan oleh Ferry *et al.* (2019) telah mengembangkan meteran air yang terkoneksi ke database, memungkinkan monitoring dan kontrol meteran air secara lebih efisien. Dengan menggunakan modul *Water flow sensor*, sistem mampu mengukur debit air dengan ketepatan 90% dan waktu rata-rata pengiriman data kurang dari 400ms. Meteran ini juga dilengkapi dengan valve yang dapat dikontrol otomatis melalui database.

Lintang *et al.* (2022) juga melakukan penelitian tentang sistem monitoring distribusi air menggunakan *mesh network* dan sensor *flowmeter* YF-S201 terintegrasi dengan mikrokontroler ESP32. Meskipun terdapat *error* pembacaan debit air sebesar 15.5%, sistem ini mampu memonitor penggunaan air secara *real-time* dengan rata-rata waktu pengiriman data dari *node* ke *gateway* adalah 10.38 detik dan dari *gateway* ke

broker adalah 3.95 detik. Penelitian ini menunjukkan potensi penggunaan topologi *mesh network network* dalam pemantauan distribusi air ledeng, namun diperlukan perbaikan pada metode pembacaan debit air untuk mengurangi tingkat *error*.

Penelitian lain juga dilakukan oleh Maulidin *et al.* (2020) membahas tentang Perancangan Sistem Monitoring Penggunaan Air PAM Berbasis IoT dengan Bot Telegram. Penelitian ini bertujuan untuk mengembangkan sistem yang memungkinkan pelanggan PDAM memonitor penggunaan air mereka secara *real-time* menggunakan Arduino Uno, Sensor *flowmeter*, dan Bot Telegram. Hasilnya diharapkan dapat meningkatkan pelayanan kepada pelanggan dengan memberikan informasi tentang debit air, *flow rate*, dan biaya penggunaan air secara langsung melalui Bot Telegram. Penelitian ini juga menunjukkan bahwa dengan meningkatnya volume air yang diamati, nilai *error* pengukuran cenderung menurun.

Pada tahun 2017, Nugra *et al.* pernah melakukan penelitian untuk mengembangkan sistem monitoring meteran air dengan memanfaatkan *Wireless Sensor Network* (WSN). Sistem ini terdiri dari *node* sensor yang mengirimkan data ke webserver dan kemudian ke platform ThingSpeak. Hasil pengujian menunjukkan bahwa *node* sensor memiliki jangkauan maksimal 34 meter dengan tingkat keberhasilan pengiriman data sebesar 99% pada jarak 20 meter. Selain itu, sistem ini memungkinkan pemantauan konsumsi air secara online yang lebih mudah dan efisien, dengan potensi untuk menghemat waktu dan biaya operasional.

Selanjutnya pada tahun 2017, dengan memanfaatkan SMS Suardiana *et al.* juga melakukan penelitian yang menghasilkan sistem pembacaan jumlah konsumsi air pelanggan PDAM yang efektif dengan menggunakan sensor YFS201, RTC DS1307 dan mikrokontroler ATmega328, dilengkapi dengan SMS untuk memudahkan akses data oleh pelanggan. Sistem ini menunjukkan penyimpangan rata-rata sebesar 0,39% dalam pembacaan konsumsi air, menawarkan kemudahan dalam memantau penggunaan air secara *real-time*, serta menyediakan konversi langsung ke satuan pembayaran yang sesuai dengan model pembayaran PDAM di Kabupaten Gianyar. Selain itu, sistem ini bekerja dengan konsumsi daya yang rendah, hanya 0.8 Watt, yang berkontribusi pada efisiensi energi.

Tabel 2.1 Penelitian Terdahulu

No.	Penulis	Tahun	Judul	Keterangan
1.	Ria Sood <i>et al.</i>	2013	<i>Design and Development Automatic Water Flow Meter</i>	Penelitian dengan tujuan untuk mengembangkan meter aliran air otomatis berbiaya rendah yang dapat menyediakan jumlah air yang tepat untuk irigasi tanpa pemborosan. Dengan metode yang melibatkan sensor <i>waterflow</i> dengan efek Hall G1/2 dengan rotor turbin, yang kecepatan rotasinya berubah sesuai dengan laju aliran air, dan mikrokontroler AT89S52 yang diprogram dengan <i>software</i> Keil.
2.	Alexandr a <i>et al</i>	2021	<i>WaterAMI - Water Automated Metering Infrastructure Based on an Energy Aware Wireless Mesh Network Communication Protocol</i>	Penelitian ini merancang sebuah Integrated Performance Management System (IMES) pada jaringan distribusi air dengan menggunakan sebuah protokol LPWAN-AfE-EA yang dikembangkan oleh CWJ Power Electronics untuk memastikan cakupan semua perangkat di infrastruktur perairan, termasuk area yang sulit dijangkau. Sistem ini telah berhasil diterapkan di berbagai WDN di Portugal, mengatasi keterbatasan sistem sebelumnya.
3.	Aryanto <i>et al</i>	2020	Rancang Bangun Kontrol Dan Monitoring Meteran Air PDAM Berbasis <i>Internet Of Things</i>	Peneliti ini berhasil mengembangkan meteran air yang terkoneksi ke Server Wemos, memungkinkan monitoring dan kontrol meteran air secara lebih efisien. Dengan modul Water flow sensor, sistem mampu mengukur debit air dengan ketepatan 90% dan waktu rata-rata pengiriman data kurang dari 400ms. Meteran ini juga dilengkapi dengan valve yang dapat dikontrol otomatis melalui database.
4.	Lintang <i>et al</i>	2022	Distribusi Air Ledeng dan Metering Menggunakan <i>Mesh</i>	Penelitian ini menggunakan mesh network dan sensor flowmeter YF-S201 terintegrasi dengan mikrokontroler ESP32 berhasil dikembangkan. Terdapat

		<i>Network Perumahan</i>	Untuk error pembacaan debit air sebesar 15.5%, sistem ini mampu memonitor penggunaan air secara real-time dengan ratarata waktu pengiriman data dari <i>node</i> ke gateway adalah 10.38 detik dan dari gateway ke broker adalah 3.95 detik.
5.	Maulidin et al 2020	Perancangan Sistem Monitoring Penggunaan Air Pam Berbasis IoT Dengan Bot Telegram	Penelitian ini bertujuan untuk mengembangkan sistem yang memungkinkan pelanggan PDAM memonitor penggunaan air mereka secara real-time menggunakan Arduino Uno, Sensor Flowmeter, dan Bot Telegram. Hasilnya diharapkan dapat meningkatkan pelayanan kepada pelanggan dengan memberikan informasi tentang debit air, flow rate, dan biaya penggunaan air secara langsung melalui Bot Telegram.
6.	Arsyistawa et al 2017	Aplikasi Wireless Sensor Network Untuk Pembacaan Meteran Air	Penelitian ini memanfaatkan Wireless Sensor Network (WSN) untuk sistem pemantauan meter air. Sistem ini terdiri dari <i>node</i> sensor yang mengirim data ke server web dan kemudian ke platform ThingSpeak. Hasil pengujian menunjukkan bahwa <i>node</i> sensor mempunyai jangkauan maksimum 34 meter dengan tingkat keberhasilan transmisi data sebesar 99% pada jarak 20 meter persegi. Selain itu, sistem ini memungkinkan pemantauan konsumsi air secara online lebih sederhana dan efisien, sehingga berpotensi menghemat waktu dan biaya pengoperasian.
7.	Suardiana et al 2017	Rancang Bangun Sistem Pembacaan Jumlah Konsumsi Air Pelanggan PDAM Berbasis Mikrokontroler ATMEGA328 Dilengkapi SMS	Penelitian ini menggunakan sensor YF-S201, RTC DS1307 dan sebuah mikrokontroler ATmega328, dilengkapi dengan SMS untuk memudahkan akses data oleh pelanggan. Sistem ini menunjukkan penyimpangan rata-rata sebesar 0,39% dalam pembacaan konsumsi air, menawarkan kemudahan dalam memantau penggunaan air secara real-time, serta menyediakan konversi

langsung ke satuan pembayaran yang disesuaikan pada cara pembayaran PDAM di Kabupaten Gianyar. Selain itu, sistem ini bekerja dengan konsumsi daya yang rendah, hanya 0.8 Watt, yang berkontribusi pada efisiensi energi.



2.9 Perbedaan Penelitian

Penelitian yang ditulis ini dibuat dengan mengambil perbedaan dari penelitian yang sudah ada sebelumnya, yang mana pada penelitian Alexandra *et al.* (2021) menggunakan protokol jaringan berupa AfE-EA yang dikembangkan oleh CWJ Power Electronics, namun untuk penelitian ini akan menggunakan jaringan Wi-Fi yang sudah banyak digunakan di rumah masyarakat Indonesia.

Dari penelitian Ferry *et al.* (2019) yang menggunakan jaringan selular 4G LTE dan mikrokontroler berupa Wemos D1 Mini ESP8266, sedangkan untuk penelitian ini memanfaatkan jaringan Wi-Fi dengan mikrokontroler ESP32 Dev Kit. Perbedaan juga terletak pada interface monitoring meteran air, yang mana pada penelitian sebelumnya dari Maulidin *et al.* (2020) memakai fitur Chat Bot Telegram dan Suardiana *et al.* (2017) memanfaatkan SMS dengan modul IComSat V1.1-SIM900, sedangkan untuk penelitian ini akan menggunakan *Node-RED* sebagai platform tampilan *dashboard*.

Dari penelitian yang dilakukan oleh Nugra *et al.* (2017) terdapat perbedaan sensor untuk mengukur arus air yang digunakan yang digunakan yaitu *Waterflow Sensor FS300A*, untuk penelitian ini sensor *waterflow* yang digunakan berupa *Waterflow Sensor YF-S201*. Sebagai otak pengontrolnya mereka menggunakan mikrokontroler Arduino Uno dan modul Xbee sebagai modul komunikasi data antar *node*, sedangkan dalam penelitian ini digunakan ESP32 dengan modul Wi-Fi yang sudah terdapat di dalam paket *NodeMCU ESP32* tersebut.

BAB 3

ANALISIS DAN PERANCANGAN

Pada bab ini membahas analisis dengan rancangan sistem terintegrasi aplikasi berbasis *dashboard* antara data *real-time* dari sensor *waterflow* menggunakan *Wireless Mesh Network*. Hasil dari pembacaan debit air dan volume air yang terdeteksi sensor akan dikalibrasi agar lebih mendekati dengan air yang mengalir lalu akan ditampilkan pada antarmuka *dashboard* menggunakan platform *Node-RED*.

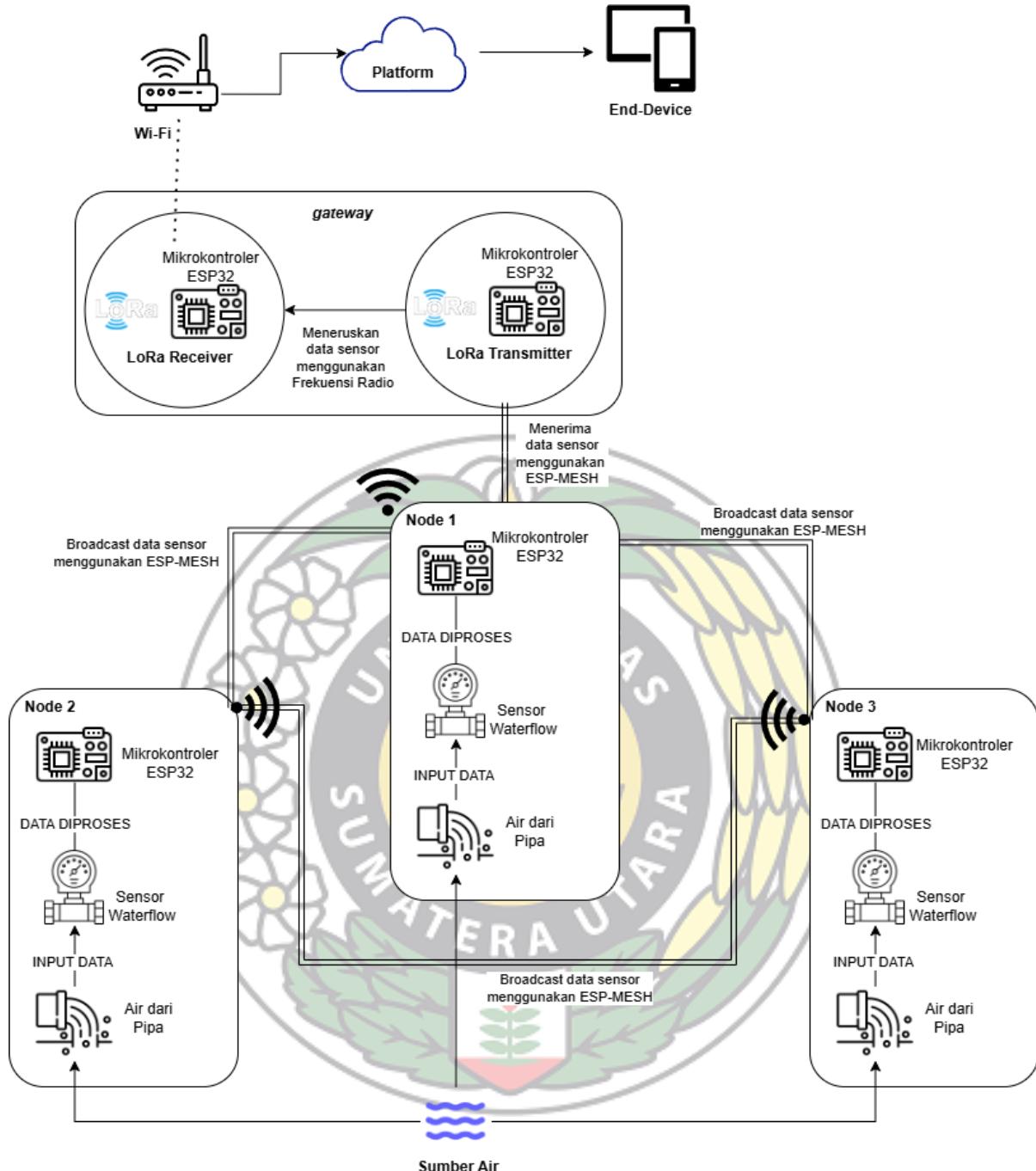
3.1 Data yang Digunakan

Data yang digunakan dalam penelitian diperoleh langsung dari sensor *Water flow*. Data *real-time* sensor *Water flow* akan dikirim ke mikrokontroler *NodeMCU ESP32* bersama dengan unit pengukuran untuk laju aliran air, yaitu Liter per detik (L/detik). Tingkat kesesuaian antara volume air yang dialirkan dan volume yang dibaca sistem akan diperiksa, bersama dengan kecepatan pengiriman data dari *node* lalu ditampilkan ke *dashboard*, menggunakan *Mesh Network*. Sensor memberikan data berupa *ID Node* sensor dan debit air dalam angka desimal.

3.2 Analisis Kebutuhan Sistem

Beberapa instrumen dan peralatan digunakan dalam penelitian ini untuk menyesuaikan dengan kebutuhan sistem. Sistem dan pengujian yang dibangun dalam bentuk *prototype*, yang mana jika sistem dapat menghitung nilai aliran dan volume air yang dan dapat ditampilkan di *dashboard*, maka sistem dapat dianggap berhasil. Pada umumnya dengan adanya digunkannya teknologi IoT pada rumah maka rumah tersebut harus dimemakai Wi-Fi. Namun pada sistem ini dengan memanfaatkan ESP-MESH maka tiap *node* di rumah-rumah tidak memerlukan Wi-Fi untuk mengirim data, ESP-MESH akan dapat berkomunikasi dengan sesama *node* yang memakai ESP-MESH..

Analisis kebutuhan sistem dilakukan agar dapat memastikan setiap bagian berfungsi sebagaimana mestinya. Sistem ini menggunakan mikrokontroler *ESP32*, yang memproses data dari input sensor, menerima data sensor, dan mengirimkan data pembacaan sensor ke server. Sensor aliran air berjenis YF-S201 adalah sensor yang digunakan untuk menentukan debit air pengguna. Data sensor akan dikirim melalui protokol komunikasi *MQTT* dan ditampilkan secara real time di *dashboard* yang dibangun menggunakan platform *Node-RED*.



Gambar 3.1 Arsitektur Umum

Berikut ini merupakan penjelasan dari arsitektur umum yang terdapat pada Gambar 3.1.

3.2.1 Perangkat input

Sistem ini menggunakan air sebagai media utama yang mengalir melalui jaringan pipa distribusi ke rumah-rumah. Setiap pipa distribusi yang masuk ke rumah pelanggan dilengkapi dengan sensor aliran air. Sensor ini dirancang untuk mendeteksi laju aliran

air di dalam pipa dan mengubahnya menjadi pulsa elektrik yang merepresentasikan jumlah air yang mengalir dalam satuan waktu tertentu.

Sensor aliran air bekerja berdasarkan prinsip mekanis, di mana aliran air menyebabkan bilah-bilah di dalam sensor berputar. Perputaran ini kemudian diubah menjadi sinyal listrik dalam bentuk pulsa, yang jumlahnya sebanding dengan volume air yang melewati sensor.

Komponen ini kemudian diintegrasikan dengan *NodeMCU ESP32*, yang berperan sebagai mikrokontroler untuk membaca dan memproses data dari sensor dan mengubah sinyal analog menjadi data digital yang dapat diolah lebih lanjut. *NodeMCU ESP32* pada setiap *node* sensor memegang peranan penting dalam sistem ini. Selain mengolah data dari sensor, perangkat ini juga bertugas untuk menghubungkan data yang telah diolah ke jaringan komunikasi berbasis ESP-MESH. Perangkat ini bekerja secara efisien karena memiliki modul Wi-Fi internal yang memungkinkan komunikasi data dengan perangkat lain dalam sistem.

3.2.2 Pemrosesan

3.2.2.1 Pemrosesan data pada node sensor

Setiap *node* sensor dirancang untuk melakukan dua fungsi utama yaitu pengukuran dan pemrosesan data. Pengukuran dilakukan dengan membaca pulsa listrik yang dihasilkan oleh sensor aliran air. *NodeMCU ESP32* kemudian memproses data ini untuk menghitung parameter penting, seperti debit air (laju aliran) yang berupa kecepatan aliran air dalam liter per menit (L/menit). Volume air yang berupa total air yang melewati sensor dalam liter (L). Proses ini melibatkan pengubahan data mentah menjadi informasi yang dapat digunakan oleh sistem, termasuk kalibrasi untuk meningkatkan akurasi pengukuran.

3.2.2.2 Jaringan Komunikasi ESP-MESH

Teknologi ESP-MESH digunakan untuk menghubungkan semua *node* sensor dalam jaringan berdasarkan topologi *mesh*. Tidak seperti topologi tradisional seperti star atau bus, topologi *mesh* memungkinkan setiap *node* untuk berkomunikasi secara langsung satu sama lain. Hal ini menghasilkan beberapa keuntungan, antara lain:

- 1) *Self-configuration* : *node* dalam jaringan dapat secara otomatis menetapkan jalur komunikasi terbaik berdasarkan kondisi jaringan.

- 2) *Self-healing*: jika sebuah *node* mengalami kegagalan, jaringan secara otomatis menemukan jalur alternatif untuk mempertahankan konektivitas data.

Dengan teknik ini, setiap *node* sensor tidak hanya bertindak sebagai pengirim data, tetapi juga bertindak sebagai perantara untuk meneruskan data dari *node* lain ke gateway. Data yang dikirim dari setiap *node* dikumpulkan oleh gateway untuk diproses lebih lanjut.

3.2.2.3 Pengumpulan Data Gateway

Gateway pada sistem ini terdiri dari dua perangkat utama dengan modul LoRa:

- 1) LoRa *transmitter* : ESP32 *NodeMCU* yang menerima data dari jaringan ESP-MESH dan bertanggung jawab untuk mengirimkan data melalui modul LoRa menggunakan frekuensi radio.
- 2) LoRa *receiver* : Perangkat ESP32 *NodeMCU* lain yang menerima data dari pemancar LoRa dan meneruskannya ke sistem berikutnya.

Komunikasi berbasis LoRa dipilih karena kemampuannya untuk mengirimkan data jarak jauh dengan konsumsi daya yang sangat rendah, menjadikannya solusi ideal untuk aplikasi *Internet of Things* (IoT) yang hemat energi. Setelah data diterima oleh penerima LoRa, modul WiFi pada *NodeMCU* ESP32 digunakan untuk mengirim data ke *broker MQTT* melalui Internet. Protokol MQTT dipilih karena ringan, dapat diandalkan, dan cocok untuk transmisi data waktu nyata di lingkungan dengan bandwidth terbatas. Data yang ditransmisikan mencakup pengukuran debit air dan volume air dari semua *node* sensor.

Setelah data tiba di *broker MQTT*, data diproses lebih lanjut untuk memastikan keakuratannya. Proses kalibrasi dilakukan untuk mengubah pulsa listrik menjadi nilai yang lebih akurat untuk debit air dan volume air. Data ini kemudian divisualisasikan dalam bentuk dasbor untuk pemantauan.

3.2.3 Output

3.2.3.1 Hasil Kalibrasi dan Analisis Data

Output yang dihasilkan dari sistem ini mencakup informasi yang telah dikalibrasi, yaitu berupa debit air dan volume air. Debit air (*flow rate*) merupakan kecepatan aliran air dalam satuan liter per menit, yang digunakan untuk mengetahui pola konsumsi air

pelanggan. Volume air total volume air yang telah terdistribusi dalam satuan liter, digunakan untuk akuntansi dan analisis distribusi air.

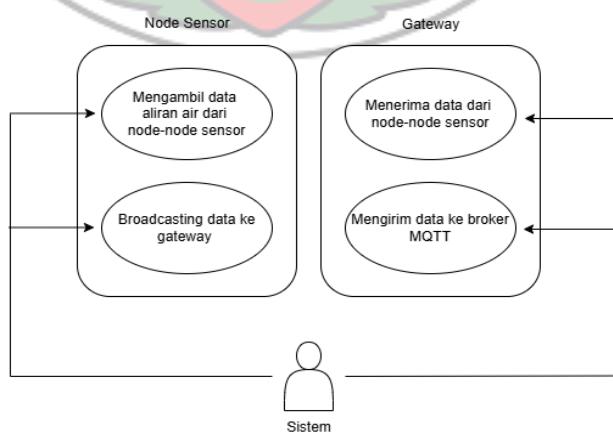
3.2.3.2 Pemantauan *real-time* melalui *dashboard*

Data ini ditampilkan dalam bentuk *dashboard* monitoring secara real-time, memungkinkan perusahaan penyedia air untuk dapat mengidentifikasi pola konsumsi air pelanggan secara langsung sehingga dapat mengambil keputusan strategis berdasarkan data yang diperoleh.

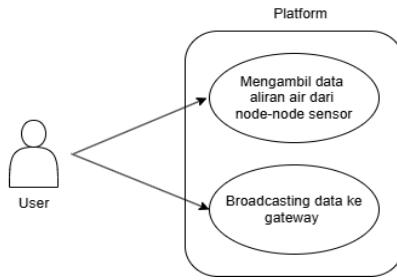
Dashboard dirancang untuk menyajikan data secara intuitif dan interaktif, sehingga mempermudah pengguna dalam menganalisis dan mengambil tindakan. Dengan sistem ini, perusahaan penyedia air dapat meningkatkan efisiensi operasional sekaligus memberikan layanan yang lebih baik kepada pelanggan.

3.3 Analisis Pemodelan Sistem

Pada tahapan ini, dilakukan sebuah analisis pemodelan menggunakan sebuah diagram dapat membantu menjelaskan secara ringkas alur penggunaan sistem dari sisi *user*. *Use-case* diagram merupakan diagram yang digunakan untuk memberikan gambaran singkat mengenai alur penggunaan sistem dari perspektif pengguna. Diagram ini tidak menjelaskan secara mendetail cara kerja *use-case*, melainkan hanya menggambarkan secara ringkas hubungan antara *use-case*, *user/aktor*, dan sistem. Alur sistem ini menggunakan dua *use-case* diagram, yaitu *use-case* diagram untuk sistem dan *use-case* diagram untuk pengguna. Ilustrasi dari *use-case* diagram sistem terdapat pada gambar 3.2, sementara *use-case* diagram pengguna terdapat pada gambar 3.3.



Gambar 3.2 Diagram *Use-case System*



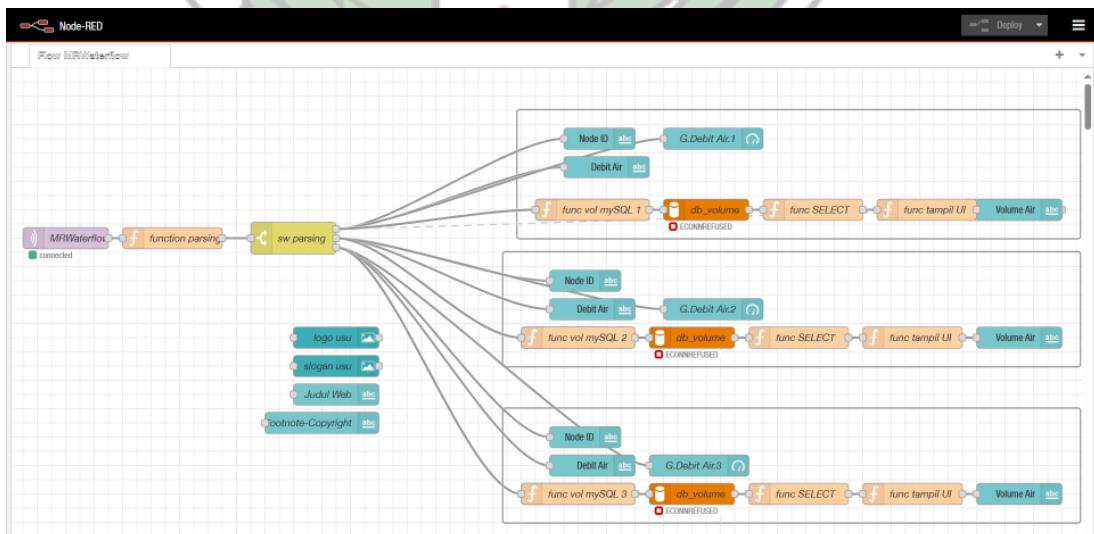
Gambar 3.3 Diagram *Use-case User*

3.4 Perancangan Tampilan Antar Muka

Tampilan antar muka pada penelitian ini dibangun menggunakan fitur UI *Dashboard* dari platform *Node-RED*. Platform *Node-RED* digunakan dengan membuat alur kerja (*flow*) dengan menambahkan beberapa *node* dan menghubungkannya berdasarkan fungsi dan tujuan yang diinginkan. *Node* ini dapat difungsikan sebagai perangkat keras, layanan web, operasi logika, manipulasi data, tampilan data dan banyak fungsi lainnya. Tampilan yang dibuat di *dashboard* selaras dengan output yang dibutuhkan untuk penelitian. Terdapat 3 variabel pada UI *dashboard* yaitu *node ID*, debit air, dan volume.

3.4.1 Flow Sistem pada Platform Node-RED

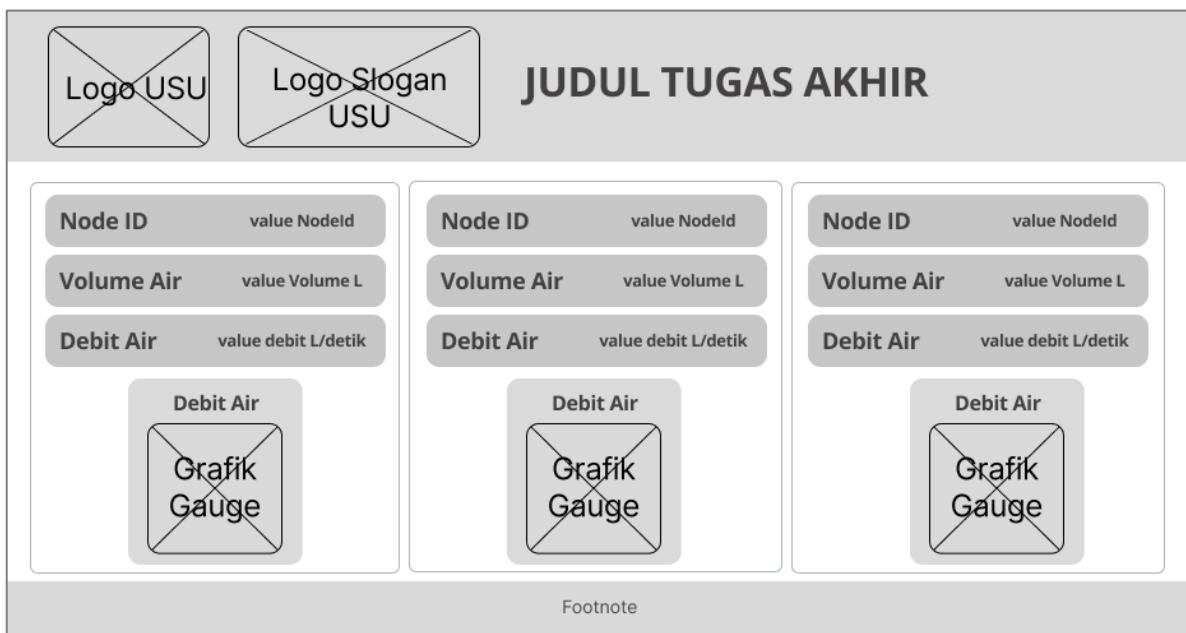
Sejumlah komponen *node* digunakan oleh sistem berbasis Platform *Node-RED* untuk membuat alur. Beberapa *node* yang dipakai seperti *node subscriber mqtt (mqtt in)*, *node function*, *node switch*, *node mySQL*, *node gauge* dan *node text* membentuk alur pemantauan yang ditunjukkan pada Gambar 3.4.



Gambar 3.4 Flow Node-RED untuk sistem monitoring

3.4.2 Fitur UI Dashboard Node-RED

Pada UI *dashboard* di *Node-RED* berfungsi sebagai tampilan yang mana pengguna dapat melakukan monitoring informasi tentang penggunaan air berupa debit air dan volume air di setiap rumah berdasarkan variable *node ID*.



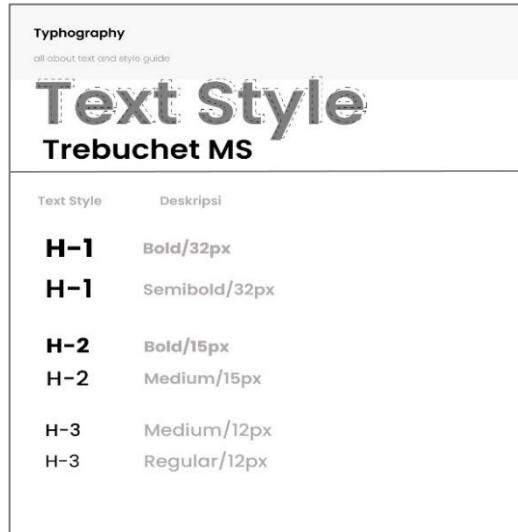
Gambar 3.5 Lo-Fi Desain Interface Dashboard Platform Node-RED

3.4.3 Design System Tampilan Dashboard

Dalam membangun tampilan *dashboard* untuk sistem ini memiliki sistematika desain yang berupa *typography*, *color Palette*, dan *shape style* dijelaskan seperti dibawah.

3.4.2.1 Typography / Text Style

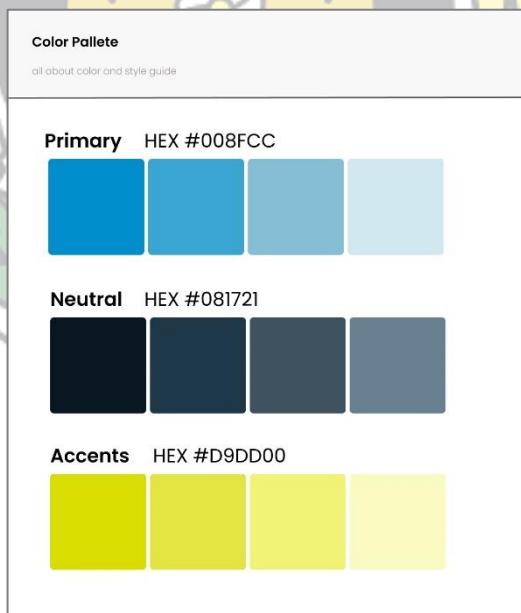
Untuk huruf yang digunakan yaitu jenis *font* “Trebuchet MS” yang tersedia di dalam platform *Node-RED*. Ukuran font pada H-1 yaitu 32px yang digunakan pada bagian *Header* halaman, H-2 yaitu 15px yang digunakan pada isi konten halaman dan H-3 yaitu 12px yang digunakan pada bagian *Footnote*.



Gambar 3.6 Typography / Text Style UI Dashboard

3.4.2.2 Color Palette

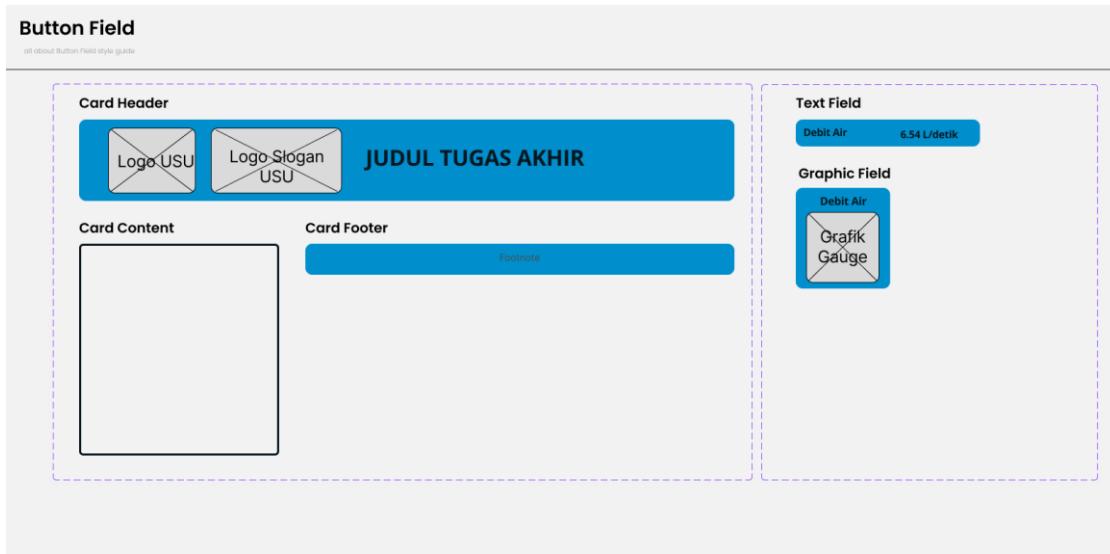
Tema warna yang digunakan pada tampilan dashboard terdiri dari 3 bagian, yaitu *Primary*, *Neutral*, dan *Accents*. Pada *primary* digunakan sebagai warna mayoritas yaitu pada *card* tampilan, *neutral* digunakan sebagai warna teks dan warna latar belakang, *accents* digunakan sebagai warna tambahan yaitu pada indicator grafik *gauge*.



Gambar 3.7 Color Palette UI Dashboard

3.4.2.3 Shape Style

Shape style berisi informasi desain bentuk-bentuk yang digunakan pada tampilan dashboard, yang berupa *card* maupun *field*.



Gambar 3.8 Shape Style UI Dashboard

3.5 Perancangan Perangkat Keras (*Hardware*)

Sistem monitoring penggunaan air untuk perumahan menggunakan *mesh network* akan dibuat dengan rancangan perangkat keras yang sesuai gambar 3.5 dan gambar 3.6. Perancangan menunjukkan skema rangkaian antara komponen yang ada pada alat untuk dapat menjalankan sistem. Sumber daya listrik dari perangkat alat ini menggunakan sumber daya listrik dari USB. Rincian *Hardware* (perangkat keras) yang dipakai untuk membangun sistem dapat dilihat pada tabel 3.1.

Tabel 3.1 Rincian Perangkat Keras

No.	Nama Barang	Banyak	Fungsi
1.	<i>NodeMCU ESP32</i>	5	Mikrokontroler
2.	Sensor <i>waterflow</i>	3	Pengukur laju aliran air
3.	Modul LoRa SX1276	2	<i>Transmitter</i> dan <i>Receiver</i> data

3.5.1 Perancangan Node Sensor

NodeMCU ESP32 digunakan dalam desain node sensor sebagai perangkat untuk mengatur data yang diperoleh dari sensor *waterflow* YF-S201. Pulsa listrik yang dihasilkan oleh sensor berbanding lurus dengan kecepatan putar rotor. Sensor akan menghasilkan pulsa listrik ketika air mengalir melewatiinya, dan mikrokontroler ESP32 akan memproses pulsa-pulsa tersebut. Pulsa listrik tersebut dikonversi ke besaran flow

rate dan dikirim atau publish ke broker MQTT melalui *gateway*, lalu di-*subscribe* dari broker MQTT untuk ditampilkan pada *dashboard* menggunakan platform Node-RED. Pada sensor *waterflow* seperti YF-S201, perhitungan *flow rate* atau debit (Q) didasarkan pada frekuensi pulsa yang dihasilkan oleh baling-baling rotor magnetik dalam sensor. Hubungan antara pulsa listrik dan debit aliran dinyatakan sebagai:

$$Q = \frac{\frac{1000}{\text{Interval (ms)}} \times \text{Pulse}}{\text{Calibration Factor}}$$

dengan:

Q : Debit aliran (liter/detik)

Pulse Count : Jumlah pulsa listrik yang dihasilkan oleh rotor

Interval : Waktu pengukuran dalam milidetik

Calibration Factor : Konstanta kalibrasi untuk sensor

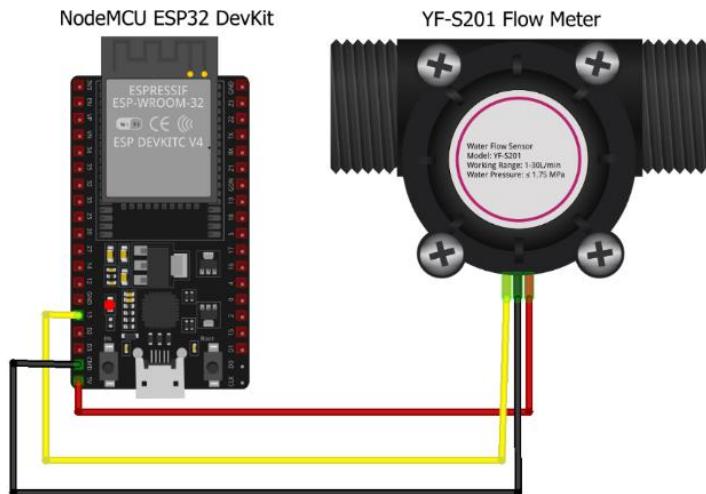
Faktor kalibrasi ini sangat penting untuk menyesuaikan perhitungan dengan karakteristik sensor agar akurat. Konstanta kalibrasi akan mengubah pulsa menjadi debit air dalam satuan liter/detik. Semakin tinggi nilai kalibrasi, debit yang dihitung menjadi lebih kecil, dan sebaliknya.

Dari perumusan tersebut, maka dapat dihitung volume air dalam waktu tertentu. Untuk mengetahui volume air (V) dalam durasi tertentu (t), digunakan rumus:

$$V = Q \times t$$

Sebagai contoh, jika *flow rate* (Q) adalah 2,5 liter/detik, maka volume air yang mengalir selama 10 detik ($t=10$) adalah $V=2,5 \times 10=25$ liter.

Penghitungan ini dilakukan secara otomatis oleh perangkat, dengan data dikirimkan melalui jaringan untuk pemantauan *real-time*. Skema rangkaian node sensor yang dipasang pada setiap rumah ditunjukkan pada Gambar 3.9.



Gambar 3.9 Skema rangkaian *node* sensor

Berikut konfigurasi pin dari sensor *waterflow* yang sama di setiap *node* mikrokontroler *NodeMCU ESP32*.

Tabel 3.2 Konfigurasi Pin Sensor *water flow* ke *NodeMCU ESP32*

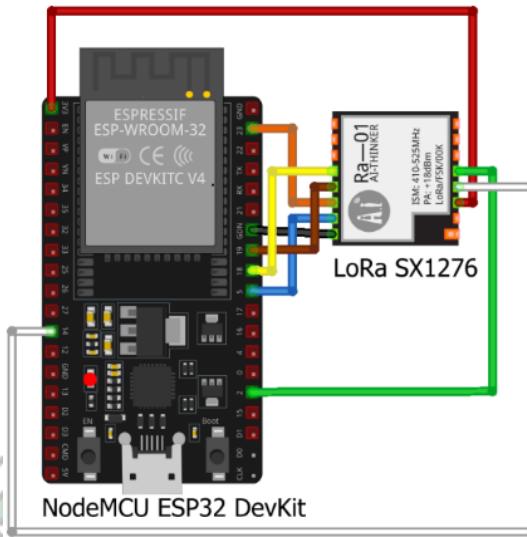
No.	Sensor <i>Water Flow</i>	ESP32
1.	VCC	5v
2.	GND	GND
3.	Output	GPIO13

Node sensor bertugas melakukan pembacaan data dari sensor dan mengirimkan data sensor kepada *gateway* melalui media komunikasi *Mesh Network*. Sensor *Node* yang membawa data sensor dan ID *node* dikirim ke *node* terdekat di dalam *Mesh Network* secara dinamis merutekan data melalui beberapa *node* jika diperlukan untuk mencapai *gateway*.

3.5.2 Perancangan *ESP32 Gateway*

Gateway yang berfungsi untuk menerima data dari *node* sensor dan mengirimkan ke broker MQTT ini dirancang terdiri dari dua perangkat, yaitu *NodeMCU ESP32* yang memiliki fungsi *ESP-MESH* untuk menerima data dari *node* sensor dan dipasang modul LoRa SX1276 sebagai *transmitter* dan perangkat lain yang memiliki fungsi WiFi untuk mengirim data ke broker MQTT dan juga dipasangi modul LoRa SX1276 sebagai

receiver. Skema rangkaian *node* sensor yang dipasang pada setiap rumah ditunjukkan oleh gambar 3.6



Gambar 3.10 Skema rangkaian *gateway*

Berikut konfigurasi pin dari modul LoRa ke mikrokontroler *NodeMCU ESP32*.

Tabel 3.3 Konfigurasi Pin modul LoRa SX1276 ke *NodeMCU ESP32*

No.	Modul LoRa SX1276	ESP32
1.	VCC	3.3v
2.	GND	GND
3.	MISO	GPIO19
4.	MOSI	GPIO23
5.	SCLK	GPIO18
6.	NSS	GPIO5
7.	GPIO0	GPIO2
8.	RESET	GPIO14

BAB 4

IMPLEMENTASI DAN PENGUJIAN

4.1 Implementasi Sistem

Monitoring meteran air untuk perumahan menggunakan teknologi *mesh network* ini dibangun dan diterapkan ke dalam sistem dengan tampilan *dashboard* dari *platform Node-RED* dengan protokol MQTT yang ringan sehingga dapat dipantau secara *realtime*.

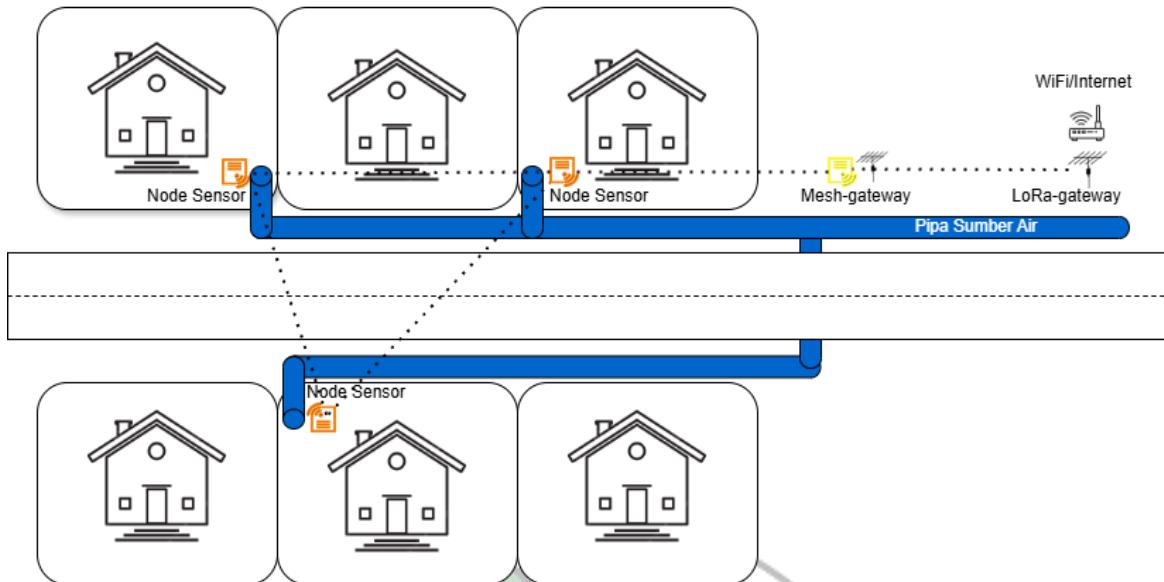
Untuk membangun sistem pada penelitian ini, digunakan perangkat keras dan perangkat lunak yang berspesifikasi sebagai berikut:

1. Laptop *Lenovo ideapad slim 3*
2. Processor *AMD Ryzen™ 3 4300U with Radeon Graphics 2.70 GHz*
3. Storage *SSD 512 GB*
4. RAM *12 GB*
5. Sistem Operasi *Windows 10 Home Single Language*
6. *Arduino IDE*

4.2 Implementasi Perangkat Keras (*Hardware*)

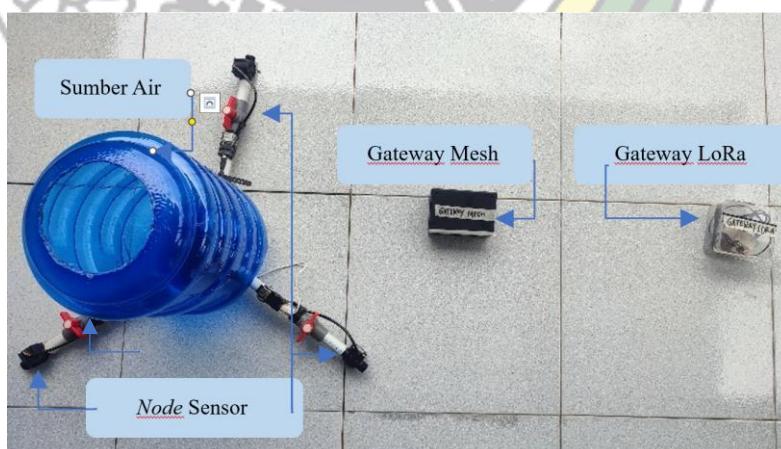
4.2.1 Rangkaian Prototype Sistem

Rangkaian perangkat keras disusun sedemikian rupa agar terlihat professional, yang terdiri dari perangkat *node* sensor dan perangkat *gateway*. Perangkat *node* sensor terdiri dari *NodeMCU ESP32* dan sensor *waterflow YF-S201*. Adapun perangkat *gateway* terdiri dari *NodeMCU ESP32* dan modul LoRa *SX1276*.



Gambar 4.1 Desain Rencana Implementasi Sistem Untuk Perumahan

Untuk simulasi *node sensor* di perumahan digunakan tabung galon yang berisi air sebagai sumber air dan disambungkan menggunakan pipa dan dipasang perangkat *node sensor* di ujung setelah keran air sebanyak 3 *node sensor* yang saling terhubung menggunakan *Wireless Mesh Network*. Lalu untuk dapat mengirim data ke broker MQTT, maka dihubungkan secara wireless ke sebuah *gateway* dengan teknologi LoRa dan modul Wi-Fi pada ESP32 yang terhubung ke jaringan internet.



Gambar 4.2 Prototipe Sistem Pengukuran Debit Air

Simulasi pengukuran air pelanggan menggunakan sebuah prototipe seperti yang ditunjukkan gambar 4.2. Simulasi ini dilakukan untuk mengetahui akurasi sensor *waterflow* dengan membandingkan jumlah volume yang terhitung sistem dengan volume air sebenarnya.

4.2.2 Rangkaian Perangkat Node Sensor

NodeMCU ESP32 yang terpasang dengan sensor *waterflow* YF-S201 dan sudah diprogram untuk membaca aliran air yang masuk dan melakukan *broadcast* paket data dengan memanfaatkan *ESP-MESH* sehingga antar *node* yang memiliki konfigurasi *ESP-MESH* yang sama dapat saling menerima dan mengirim data.



Gambar 4.3 Perangkat Node Sensor

Sensor *waterflow* YF-S201 menggunakan *output* berbasis pulsa digital yang dihasilkan oleh putaran baling-baling magnetik. ESP32 membaca pulsa ini melalui pin digital yang diatur dalam mode input dengan *pull-up resistor*. Sensor YF-S201 tidak menggunakan protokol komunikasi seperti SPI atau I2C karena hanya memerlukan sinyal pulsa sederhana untuk membaca data.

Pulsa yang dihasilkan dihitung oleh ESP32 menggunakan *interrupt*, yang memicu setiap kali terjadi perubahan sinyal dari tinggi ke rendah (*falling edge*). Pulsa ini kemudian diolah menjadi nilai *flowrate* berdasarkan waktu dan faktor kalibrasi.

```
SETUP:
1. Konfigurasikan GPIO 13 sebagai input dengan mode PULLUP.
2. Inisialisasi variabel `pulseCount` untuk menyimpan jumlah pulsa.
3. Atur interrupt untuk GPIO 13 pada sinyal falling edge untuk memanggil fungsi `pulseCounter`.
```

LOOP:

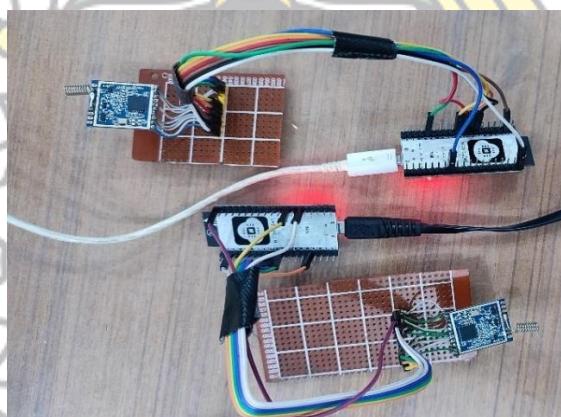
1. Detach interrupt untuk mencegah konflik selama penghitungan.
2. Simpan jumlah pulsa yang telah dihitung (`pulse1Sec = pulseCount`).
3. Reset `pulseCount` untuk siklus berikutnya.
4. Hitung flow rate menggunakan rumus:
`flowRate = (pulse1Sec / calibrationFactor) * waktuPeriode`.
5. Attach kembali interrupt untuk melanjutkan penghitungan pulsa.
6. Kirim atau tampilkan hasil `flowRate`.

Gambar 4.4 *Pseudo-code* pembacaan sensor YF-S201

Pin yang digunakan untuk dihubungkan ke ESP32 adalah GPIO 13 yang berfungsi membaca pulsa digital dari sensor. *Interrupt* digunakan untuk mendeteksi pulsa listrik dengan efisien tanpa menghalangi eksekusi kode utama. Konstanta kalibrasi (*calibration factor*) menentukan konversi dari jumlah pulsa ke debit air (*flow rate*).

4.2.3 Rangkaian Perangkat ESP32 Gateway

Rangkaian yang berfungsi sebagai *gateway* pada sistem ini terdiri dari dua perangkat, yaitu ESP32 LoRa transmitter yang diprogram menerima paket data dari semua *node* menggunakan ESP-MESH lalu diteruskan ke LoRa receiver menggunakan LoRa dalam frekuensi radio yang sama. Frekuensi yang digunakan pada sistem ini ialah pada 915 MHz. Lalu perangkat ESP32 LoRa receiver yang sudah diprogram agar terhubung ke jaringan Wi-Fi tertentu akan menerima data tersebut dan mengirimkannya ke broker MQTT.

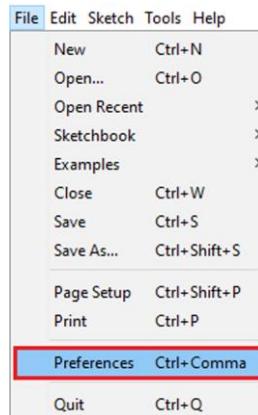


Gambar 4.5 Perangkat ESP32 Gateway

4.3 Implementasi Perangkat Lunak (*Software*)

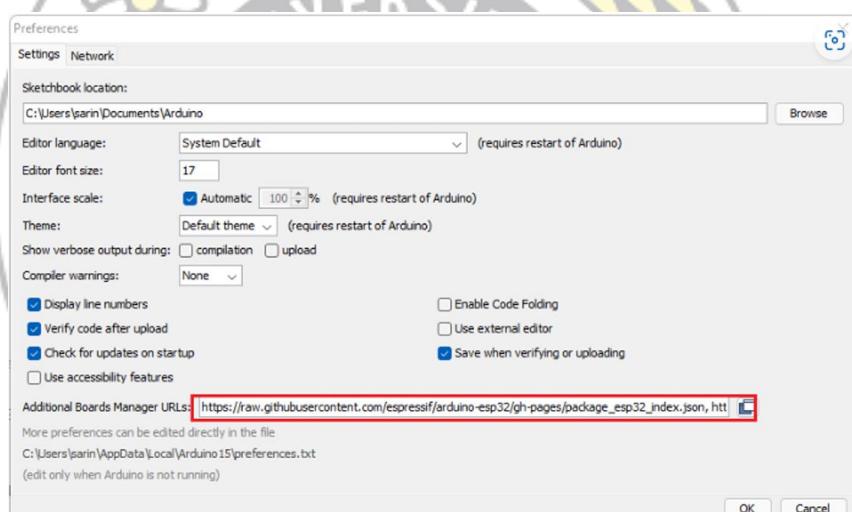
4.3.1 Program ESP32 dengan Arduino IDE

Arduino IDE adalah sebuah lingkungan pengembangan terintegrasi (*Integrated Development Environment/IDE*) yang berfungsi untuk memprogram papan sirkuit elektronik Arduino. Arduino itu sendiri adalah papan sirkuit elektronik open-source yang dilengkapi dengan mikrokontroler serta berbagai komponen tambahan. Dalam penelitian ini, terdapat beberapa langkah yang harus dilakukan, dimulai dengan membuat menu *Preference* pada Arduino IDE dengan menambahkan paket tertentu.



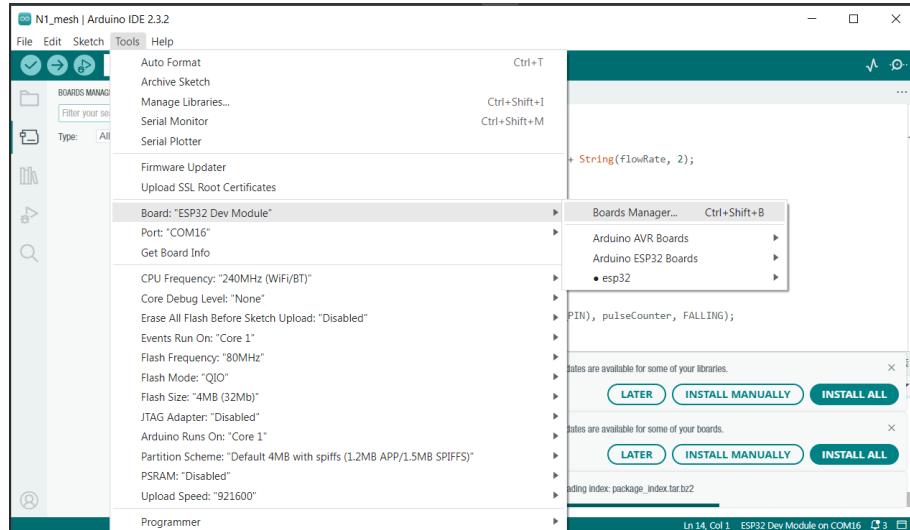
Gambar 4.6 File> Preference

Di kolom “Additional BoardManager URLs”, masukkan link berikut:
https://raw.githubusercontent.com/espressif/arduino-esp32/gh-pages/package_esp32_index.json, lalu klik “OK”.



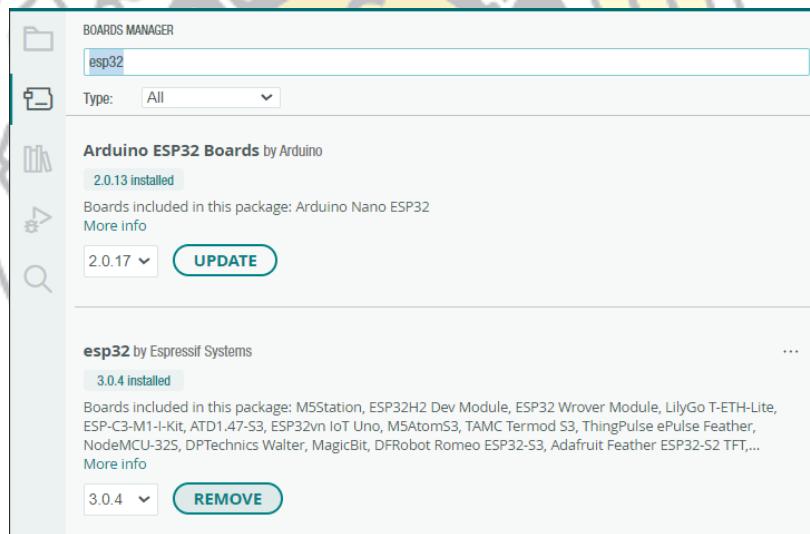
Gambar 4.7 Preferences

Selanjutnya, buka menu “Board Manager” dengan cara pergi ke bagian navbar Tools> Board> Board Manager



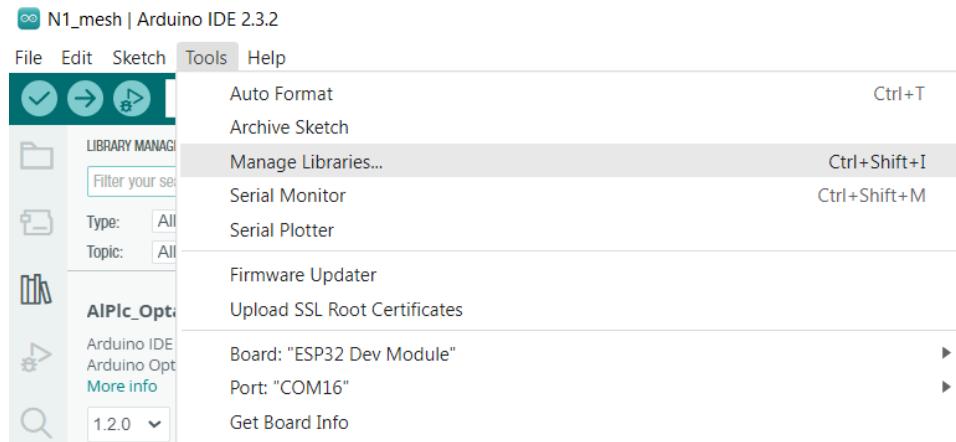
Gambar 4.8 Board Manager

Di kolom pencarian *Board Manager*, cari board “ESP32” kemudian muncul “esp32 by Espressif System” klik di tombol *install*.



Gambar 4.9 Install board ESP32 di Board Manager

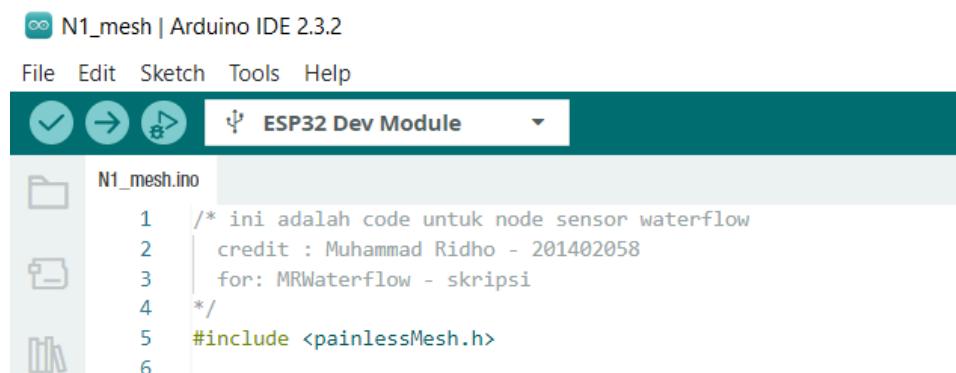
Buka menu Manage Libraries, dengan cara pergi ke *navbar Tools> Manage Libraries*



Gambar 4.10 Manage Libraries

Setelah itu akan terbuka menu *Library Manager*, di kolom pencarian *Library Manager* cari *library* yang dibutuhkan, lalu klik tombol install di bagian *library* yang sesuai.

Inisiasi *library* yang akan diperlukan untuk membuat program dengan *syntax* `#include <namaLibrary>`. Pada program untuk di *node sensor*, hanya menggunakan *library* “*painlessMesh.h*”.



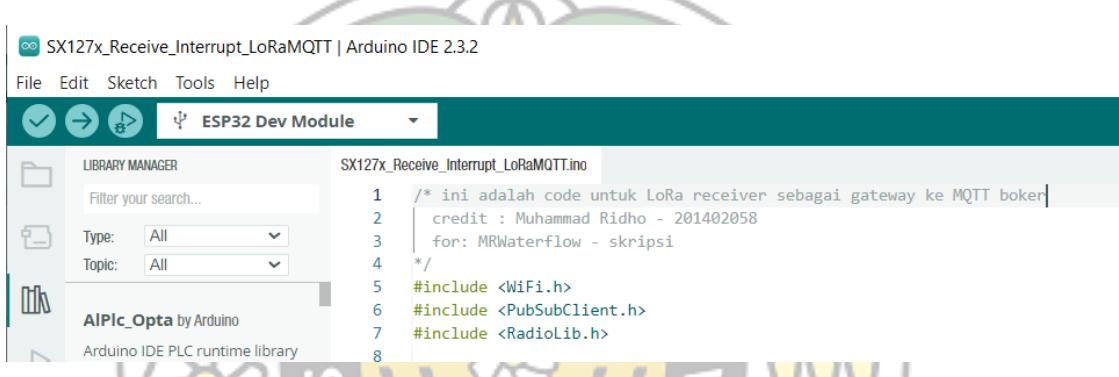
Gambar 4.11 Inisiasi library pada program node sensor

Untuk program LoRa *transmitter*, menggunakan *library* `<RadioLib.h>` dan “*painlessMesh.h*”.



Gambar 4.12 Inisiasi *library* pada program LoRa transmitter

Untuk program LoRa *receiver*, menggunakan *library* <WiFi.h>, <RadioLib.h> dan <PubSubClient.h>.



Gambar 4.13 Inisiasi *library* pada program LoRa receiver

4.3.2 Konfigurasi Koneksi ESP-MESH pada ESP32 di Arduino IDE

Pada penelitian ini, untuk *node* sensor agar bisa saling terhubung menggunakan ESP-MESH dengan *library* “painlessMesh.h”. Adapun konfigurasinya sebagai berikut.

```

 7  #define MESH_PREFIX "MRWaterflow"
 8  #define MESH_PASSWORD "201402058"
 9  #define MESH_PORT 5555

```

Gambar 4.14 Konfigurasi ESP-MESH program *node* sensor

Pada bagian fungsi *setup()*, inisiasikan ESP-MESH pada *node* sensor sebagai berikut. Yang mana dipanggil beberapa fungsi seperti *debugging*, inisiasi mesh, penerima data, penghubung koneksi dan pemberitahu jika ada perubahan koneksi.

```

44 // Inisiasi mesh network
45 mesh.setDebugMsgTypes(ERROR | STARTUP);
46 mesh.init(MESH_PREFIX, MESH_PASSWORD, MESH_PORT);
47 mesh.onReceive(&receivedCallback);
48 mesh.onNewConnection(&newConnectionCallback);
49 mesh.onChangedConnections(&changedConnectionCallback);
50 Serial.println("Mesh initialized");

```

Gambar 4.15 Inisiasi ESP-MESH program *node* sensor

4.3.3 Konfigurasi Koneksi LoRa SX1276 pada ESP32 di Arduino IDE

Pada penelitian ini, untuk koneksi di *gateway* menggunakan LoRa agar bisa saling terhubung secara *realtime* dengan *library* <RadioLib.h>. Adapun konfigurasi pin dari modul LoRa sx1276 ke *NodeMCU* ESP32 sebagai berikut.

```

10 // SX1276 has the following connections:
11 // NSS pin: 5
12 // DIO0 pin: 2
13 // RESET pin: 14
14 SX1276 radio = new Module(5, 2, 14);
15

```

Gambar 4.16 Konfigurasi ESP32 dengan LoRa SX1276

Pada bagian fungsi *setup()*, inisiasikan LoRa dengan pengaturan *default*, yang mana dideklarasikan fungsi *radio.begin()* yang menyimpan parameter *frequency*, *bandwidth*, *spreading factor*, *symbol rate*, *power*, dan sebagainya. Kemudian dilakukan pengecekan dan *debugging* jika ada terjadi *error*, serta mengatur frekuensi pada fungsi *radio.setFrequency(915E6)*; yang berarti frekuensi yang digunakan yaitu pada 915 MHz dan disamakan antara LoRa *transmitter* dengan *receiver*.

```

76 // Initialize SX1276 with default settings
77 Serial.print(F("[SX1278] Initializing ... "));
78 int state = radio.begin();
79 if (state == RADIOLIB_ERR_NONE) {
80   Serial.println(F("success!"));
81   radio.setFrequency(915E6);
82 } else {
83   Serial.print(F("failed, code "));
84   Serial.println(state);
85   while (true);
86 }
87

```

Gambar 4.17 Inisiasi LoRa SX1276

Untuk parameter lainnya seperti *bandwidth*, *spreading factor*, *symbol rate*, *power*, dan sebagainya dibiarkan secara *default* sesuai yang sudah ada di *library* ‘*RadioLib.h*’

```
int16_t begin(float freq = 434.0, float bw = 125.0, uint8_t sf = 9, uint8_t cr = 7, uint8_t syncWord = RADIOLIB_SX127X_SYNC_WORD,
int8_t power = 10, uint16_t preambleLength = 8, uint8_t gain = 0);

/*
\breif FSK modem initialization method. Must be called at least once from Arduino sketch to initialize the module.
\param freq Carrier frequency in MHz. Allowed values range from 137.0 MHz to 1020.0 MHz.
\param br Bit rate of the FSK transmission in kbps (kilobits per second). Allowed values range from 1.2 to 300.0 kbps.
\param freqDev Frequency deviation of the FSK transmission in kHz. Allowed values range from 0.6 to 200.0 kHz.
Note that the allowed range changes based on bit rate setting, so that the condition FreqDev + BitRate/2 <= 250 kHz is always met.
\param rxBw Receiver bandwidth in kHz. Allowed values are 2.6, 3.1, 3.9, 5.2, 6.3, 7.8, 10.4, 12.5, 15.6, 20.8, 25, 31.3, 41.7, 50,
62.5, 83.3, 100, 125, 166.7, 200 and 250 kHz.
\param power Transmission output power in dBm. Allowed values range from 2 to 17 dBm.
\param preambleLength Length of FSK preamble in bits.
\param enableOOK Use OOK modulation instead of FSK.
\returns \ref status_codes
```

Gambar 4.18 Parameter *default* dari *RadioLib.h*

Inisiasi LoRa sebagai *transmitter* dengan *receiver* sebenarnya sama, dan yang membedakannya yaitu pada fungsi *radio.startReceiver()*; yang berarti berfungsi sebagai LoRa *receiver*, sedangkan LoRa *transmitter* memanggil fungsi *radio.transmit(msg)*:

```
24 // Transmit the received message over LoRa
25 Serial.print(F("[LoRa] Transmitting: "));
26 Serial.println(msg);
27
28 int state = radio.transmit(msg);
29 if (state == RADIOLIB_ERR_NONE) {
30   Serial.println(F("LoRa transmission success!"));
31 } else {
32   Serial.print(F("LoRa transmission failed, code "));
33   Serial.println(state);
34 }
35 }
36 }
```

Gambar 4.19 Fungsi pada LoRa *transmitter*

```
91 // Start listening for incoming LoRa packets
92 Serial.print(F("[SX1278] Starting to listen ... "));
93 state = radio.startReceive();
94 if (state == RADIOLIB_ERR_NONE) {
95   Serial.println(F("success!"));
96 } else {
97   Serial.print(F("failed, code "));
98   Serial.println(state);
99   while (true);
100 }
101 }
```

Gambar 4.20 Fungsi pada LoRa *receiver*

4.3.4 Konfigurasi Koneksi WiFi pada ESP32 di Arduino IDE

Pada penelitian ini, untuk koneksi di *gateway* agar terhubung ke internet menggunakan WiFi yang sudah terpasang di dalam *NodeMCU ESP32* sehingga hanya perlu dilakukan konfigurasi dengan *library* <WiFi.h>. Adapun kredensial WiFi pada *NodeMCU ESP32* sebagai berikut.

```
19 const char* ssid = "Pakaiaja kali";
20 const char* password = "";
```

Gambar 4.21 Kredensial WiFi

Pada bagian fungsi *setup()*, terdapat fungsi menghubungkan ke WiFi dengan fungsi *WiFi.begin(ssid, password)*.

```
29 // Function to connect to WiFi
30 void setupWiFi() {
31     delay(10);
32     Serial.println();
33     Serial.print("Connecting to ");
34     Serial.println(ssid);
35
36     WiFi.begin(ssid, password);
37
38     while (WiFi.status() != WL_CONNECTED) {
39         delay(500);
40         Serial.print(".");
41     }
42
43     Serial.println("");
44     Serial.println("WiFi connected");
45 }
```

Gambar 4.22 Inisiasi koneksi ke WiFi

4.3.5 Konfigurasi MQTT pada ESP32 di Arduino IDE

Pada penelitian ini menggunakan protokol komunikasi MQTT, yang mana sistem perangkat sensor berfungsi sebagai *publisher*, sedangkan yang berperan sebagai *subscriber* ialah platform *dashboard* yang digunakan, yaitu *Node-RED*. Adapun Broker MQTT yang digunakan pada penelitian ini ialah broker yang bersifat public, yaitu server broker dari “mqtt.eclipseprojects.io”. Dengan memakai satu topik “*mrwaterflow/data*” dan port “1883”.

```
21 const char* mqttServer = "mqtt.eclipseprojects.io"; /
22 const int mqttPort = 1883;
23 const char* mqttTopic = "mrwaterflow/data";
24
```

Gambar 4.23 Kredensial Server Broker MQTT

Dengan menggunakan *library* <PubSubClient.h>, sehingga dapat mengirim data ke server broker dengan memanggil fungsi *client.publish()*.

```

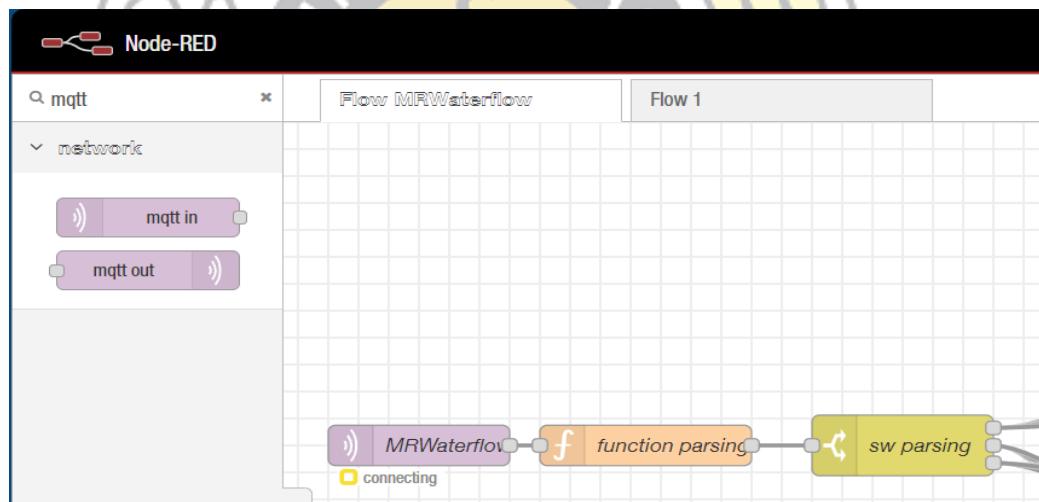
122     // Send the data to the MQTT broker
123     if (client.publish(mqttTopic, receivedData.c_str())) {
124         Serial.println(F("Data sent to MQTT broker!"));
125     } else {
126         Serial.println(F("Failed to send data to MQTT broker"));
127     }
128

```

Gambar 4.24 Fungsi mengirim data ke server broker MQTT

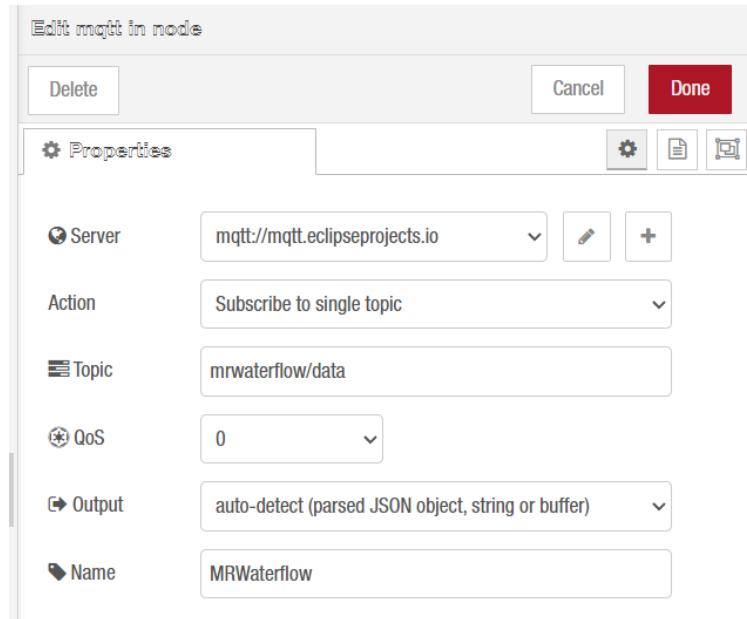
4.3.6 Konfigurasi MQTT pada platform Node-RED

Untuk menampilkan data air, digunakan Node-RED sebagai *dashboard* yang mana protokol komunikasi yang digunakan yaitu MQTT. Agar data bisa tampil, maka *dashboard* Node-RED perlu dilakukan konfigurasi sebagai *subscriber* dengan mengambil *node* “mqtt in” yang berada di kategori *network* pada IDE Node-RED lalu meletakkannya di dalam *flow*, seperti gambar berikut.



Gambar 4.25 Node *mqtt in* di Node-RED sebagai *subscriber*

Setelah itu, node *mqtt in* perlu diatur pada *properties* yang disediakan Node-RED berupa alamat server broker MQTT, topik, QoS dan sebagainya. Server broker yang diinput yaitu “<mqtt://mqtt.eclipseprojects.io>” dengan topik “mrwaterflow/data” dan opsi lain secara *default*, dapat dilihat pada gambar berikut.



Gambar 4.26 Mengatur *Properties* node *mqtt in* di Node-RED

4.3.7 Format Data Komunikasi antar Node

Pada sistem monitoring ini, data sensor yang dihasilkan perlu dikomunikasikan secara berjenjang mulai dari node sensor, melalui *mesh network*, hingga diteruskan melalui LoRa dan akhirnya dipublikasikan ke broker MQTT. Format data yang digunakan dalam setiap tahap komunikasi dirancang untuk memastikan efisiensi pengiriman dan kompatibilitas antar perangkat. Berikut adalah penjelasan rinci mengenai format data dalam sistem ini.

4.3.7.1 Format Data dari Mesh ke LoRa

Dalam sistem ini, data yang dikirimkan antar node dirumuskan dalam format *string* yang sederhana. Format ini berfungsi untuk memastikan setiap data yang dikirimkan dapat dikenali dengan jelas dan diproses oleh node penerima (*Lora Receiver*). Struktur data yang dikirimkan berupa “*NodeId*: *FlowRate*: *Timestamp*”. *NodeId* merupakan identitas unik dari setiap node di *mesh network*. Elemen ini diperoleh dari metode *mesh.getNodeId()*. Node ID digunakan untuk mengidentifikasi node pengirim data, sehingga memudahkan pelacakan dan analisis performa setiap node di dalam jaringan. *FlowRate* menunjukkan debit air yang diukur oleh sensor water flow dalam satuan liter per detik (L/detik). Hasil perhitungan ini dibulatkan hingga dua angka di belakang koma untuk meningkatkan presisi data. *Timestamp* ialah waktu

pengukuran data dalam satuan milidetik sejak node dihidupkan. *Timestamp* ini disertakan untuk mencatat kapan pengukuran dilakukan, sehingga data dapat dianalisis secara kronologis.

```
// Kirim data dengan timestamp
uint32_t currentMillis = millis();
String msg = String(mesh.getNodeId()) + ":" + String(flowRate, 2) + ":" + String(currentMillis);
mesh.sendBroadcast(msg);
```

Gambar 4.27 Format Data yang dikirim dalam *Mesh Network*

Dalam kode ini setelah menghitung debit air, menyusun data dalam format string, dan mengirimkannya menggunakan fungsi *mesh.sendBroadcast()* untuk disiarkan ke node lain dalam *Mesh network*.

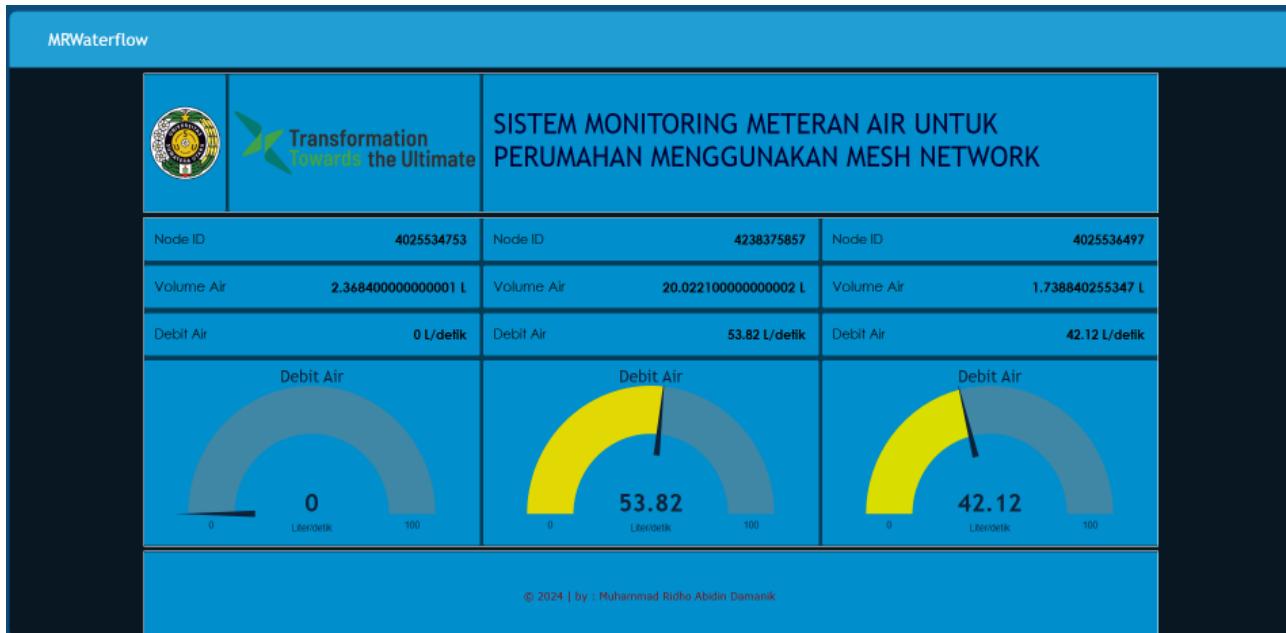
4.3.7.2 Format Data dari LoRa ke broker MQTT

Pada sistem ini, data yang diterima oleh *LoRa Receiver* dari node pengirim dikirimkan ke broker MQTT berupa *string* untuk keperluan pemantauan *real-time*. Data yang dikirimkan ke broker MQTT memiliki format yang sama seperti yang diterima dari *mesh network*, yaitu “NodeID: FlowRate: Timestamp”.

Data diterima dari node *Mesh* melalui fungsi *callback receivedCallback(uint32_t from, String &msg)*. Pesan yang diterima oleh LoRa transmitter dalam variabel *msg* kemudian diteruskan melalui modul LoRa menggunakan fungsi *radio.transmit(msg)*. Setelah data diterima oleh LoRa Receiver, proses transformasi dan pengiriman dimulai dengan membaca data yang dikirimkan oleh node LoRa menggunakan fungsi *radio.readData(receivedData)*, yang menyimpan hasilnya dalam variabel *receivedData*. Data tersebut kemudian diproses lebih lanjut dengan mengekstrak dan menganalisis informasi yang terkandung di dalamnya, di mana metode *lastIndexOf(":")* digunakan untuk memisahkan nilai debit air (flow rate) dan timestamp. Selanjutnya, latency atau waktu tunda dihitung dengan mengambil selisih antara timestamp yang diterima oleh node LoRa dan timestamp yang dikirimkan oleh node pengirim. Setelah proses ini selesai, data yang telah diterima dan diproses diteruskan ke broker MQTT melalui topik *mrwaterflow/data* menggunakan fungsi *client.publish(mqttTopic, receivedData.c_str())*. Proses ini memastikan bahwa data diterima dengan benar di broker MQTT dan siap untuk digunakan dalam pemantauan *real-time*.

4.4 Implementasi Tampilan Antar Muka

Dibutuhkan sebuah antar muka yang dapat menerima data dari broker MQTT dan menampilkan data ke sebuah *dashboard* agar mempermudah monitoring penggunaan air. Penelitian ini akan menggunakan platform *Node-RED* yang berisi beberapa *node* untuk mengatur bagaimana data yang ingin ditampilkan, Gambar berikut menunjukkan tampilan *dashboard* dari platform *Node-RED* yang telah dibuat.



Gambar 4.28 Tampilan antar muka *dashboard* monitoring air

Pada tampilan *dashboard* terdiri dari tiga bagian, yaitu *topbar*, *content* dan *footbar*. Pada bagian *topbar* terdiri dari logo dan slogan USU dan dilanjutkan judul dari tugas akhir. Di dalam *content*, informasi untuk monitoring terdiri dari *Node ID*, Volume Air dan Debit Air berupa angka dan *gauge*. Dan bagian *footbar* berisi nama penulis dan tahun pembuatan sistem.

4.5 Pengujian Sistem

Setelah seluruh tahapan implementasi perangkat keras dilakukan, selanjutnya penulis melakukan pengujian sistem dengan membagi beberapa pengujian, yaitu dengan pengujian ketelitian sensor yang tampil *dashboard*, pengujian pengiriman data dari *node* sensor ke gateway Mesh dan dari gateway mesh ke gateway LoRa yang dipantau melalui serial monitor Arduino IDE dengan dibuat kode untuk menghitung masing-masing pengujian. Perlu diketahui bahwa pengujian dilakukan di lokasi yang tidak

pada kondisi LoS (*Line of Sight*) dan terdapat *noise* seperti tembok, pepohonan dan kendaraan yang lewat.

4.5.1 Pengujian Ketelitian Sensor Waterflow

Pengujian ketelitian sensor *waterflow* dilakukan perulangan lima kali dengan volume air 5, 10 dan 20 liter. Tujuan pengujian adalah untuk mengetahui ketelitian yang ditunjukkan oleh sensor yang terhubung ke server dan penilitian dilakukan dengan memantau dari *dashboard* dengan jumlah volume air yang berbeda. Berikut adalah hasil dari pengujian.

4.5.1.1 Pengujian dengan volume 5 liter air

Tabel 4.1 Pengujian akurasi sensor pada *node 1*

#	Volume Air Diuji (Liter)	Volume Air Terbaca (Liter)	Selisih (Liter)	Error (%)
1	5	4.47	0.53	10.60
2	5	4.72	0.28	5.60
3	5	4.59	0.41	8.20
4	5	4.83	0.17	3.40
5	5	4.57	0.43	8.60
Rata-rata			0.36	7.28

Tabel 4.2 Pengujian akurasi sensor pada *node 2*

#	Volume Air Diuji (Liter)	Volume Air Terbaca (Liter)	Selisih (Liter)	Error (%)
1	5	4.57	0.43	8.60
2	5	4.55	0.45	9.00
3	5	4.83	0.17	3.40
4	5	4.74	0.26	5.20
5	5	4.65	0.35	7.00
Rata-rata			0.33	6.64

Tabel 4.3 Pengujian akurasi sensor pada *node 3*

#	Volume Air Diuji (Liter)	Volume Air Terbaca (Liter)	Selisih (Liter)	Error (%)
1	5	4.61	0.39	7.80
2	5	4.73	0.27	5.40
3	5	4.63	0.37	7.40
4	5	4.68	0.32	6.40

5	5	4.78	0.22	4.40
Rata-rata			0.32	6.28

Dari tabel data pengujian diatas, diketahui bahwa rata-rata selisih volume pada *node 1* sebesar 0.36 liter dengan rata-rata *error* 7.28%, pada *node 2* rata-rata selisih volume 0.33 liter dengan rata-rata *error* 6.64%, dan pada *node 3* rata-rata selisih volume 0.32 liter dengan rata-rata *error* 6.28%. Dari keseluruhan pengujian tersebut didapatkan rata-rata *error* pembacaan volume air sebesar 6.73%.

Hasil pengujian menunjukkan bahwa sensor *waterflow YF-S201* memiliki tingkat akurasi yang cukup baik untuk pengukuran volume air, meskipun terdapat *error* rata-rata sebesar 6.73% di antara ketiga node. *Error* ini dapat disebabkan oleh beberapa faktor, seperti kondisi jaringan internet pada *gateway*, kepekaan sensor terhadap pulsa yang dihasilkan, serta kalibrasi sensor yang mempengaruhi penghitungan debit air. Pada setiap node, rata-rata selisih volume berkisar antara 0.32 hingga 0.36 liter, dengan *node 3* menunjukkan *error* terkecil, yaitu 6.28%. Ketelitian ini cukup memadai untuk aplikasi monitoring volume air secara *real-time*, terutama jika dikombinasikan dengan kalibrasi tambahan untuk meningkatkan akurasi. Dengan hasil ini, sistem dapat dioptimalkan lebih lanjut untuk pengukuran volume yang lebih presisi pada berbagai skenario.

4.5.2 Pengujian Koneksi Mesh

Pengujian pengiriman data dengan koneksi *ESP-Mesh* dilakukan dengan menguji besaran *latency* pengiriman data dari *node* sensor ke *gateway Mesh* dengan jarak yang berbeda.

Tabel 4.4 Pengujian *latency* dengan jarak 10 meter

No	Node Sensor 1 (ms)	Node Sensor 2 (ms)	Node Sensor 3 (ms)
1	93	84	67
2	72	63	55
3	66	95	61
4	78	99	92
5	92	101	58
6	83	78	69
7	111	82	62

8	94	67	97
9	87	72	79
10	76	74	81
Rata-rata	85.2	81.5	72.1

Dari tabel 4.4, didapatkan rata-rata *latency* dengan jarak 10 meter pada *node* 1 sebesar 85.2 ms, *node* 2 sebesar 81.5 ms dan *node* 3 sebesar 72.1 ms. Dan rata-rata *latency* dari semua *node* dengan jarak 10 meter sebesar 79.6 ms.

Tabel 4.5 Pengujian *latency* dengan jarak 30 meter

No	Node Sensor 1 (ms)	Node Sensor 2 (ms)	Node Sensor 3 (ms)
1	117	214	123
2	111	201	179
3	121	178	152
4	122	125	207
5	118	229	116
6	129	153	184
7	212	192	135
8	137	136	221
9	109	185	168
10	111	117	140
Rata-rata	128.7	173	162.5

Dari tabel 4.5, didapatkan rata-rata *latency* dengan jarak 30 meter pada *node* 1 sebesar 128.7 ms, *node* 2 sebesar 173 ms dan *node* 3 sebesar 162.5 ms. Dan rata-rata *latency* dari semua *node* dengan jarak 30 meter sebesar 152.7 ms.

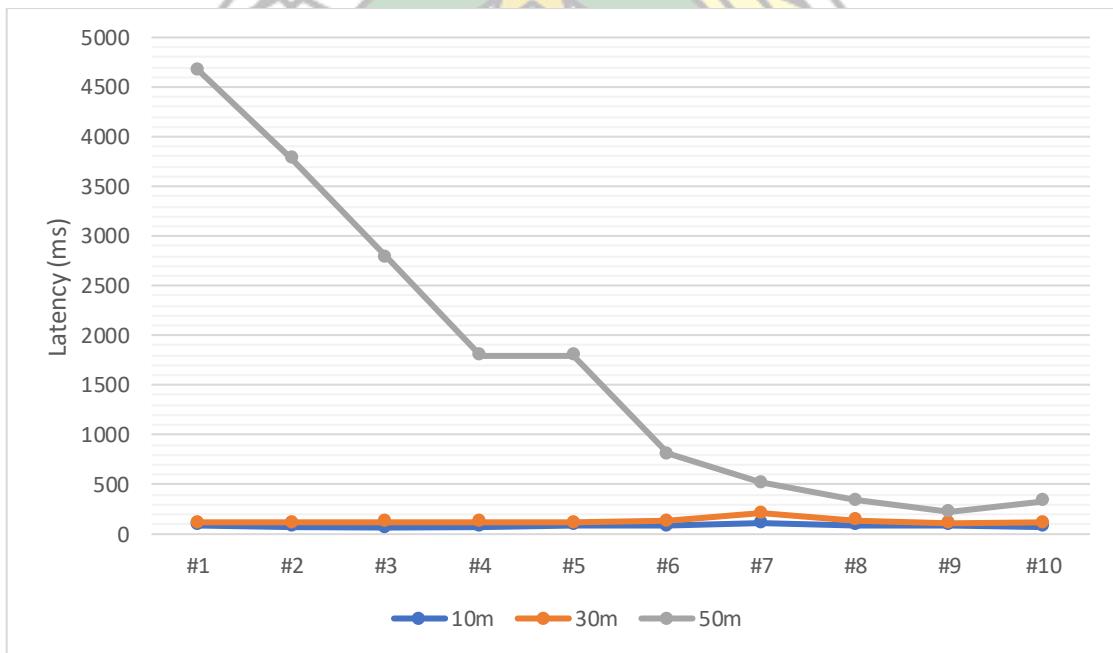
Tabel 4.6 Pengujian *latency* dengan jarak 50 meter

No	Node Sensor 1 (ms)	Node Sensor 2 (ms)	Node Sensor 3 (ms)
1	4669	5234	5872
2	3780	6481	6215
3	2789	5902	5369
4	1797	5753	5548
5	1797	5126	6037
6	805	4037	4126

7	514	2548	1753
8	341	853	648
9	224	621	702
10	332	587	434
Rata-rata	1704.8	3714.2	3670.4

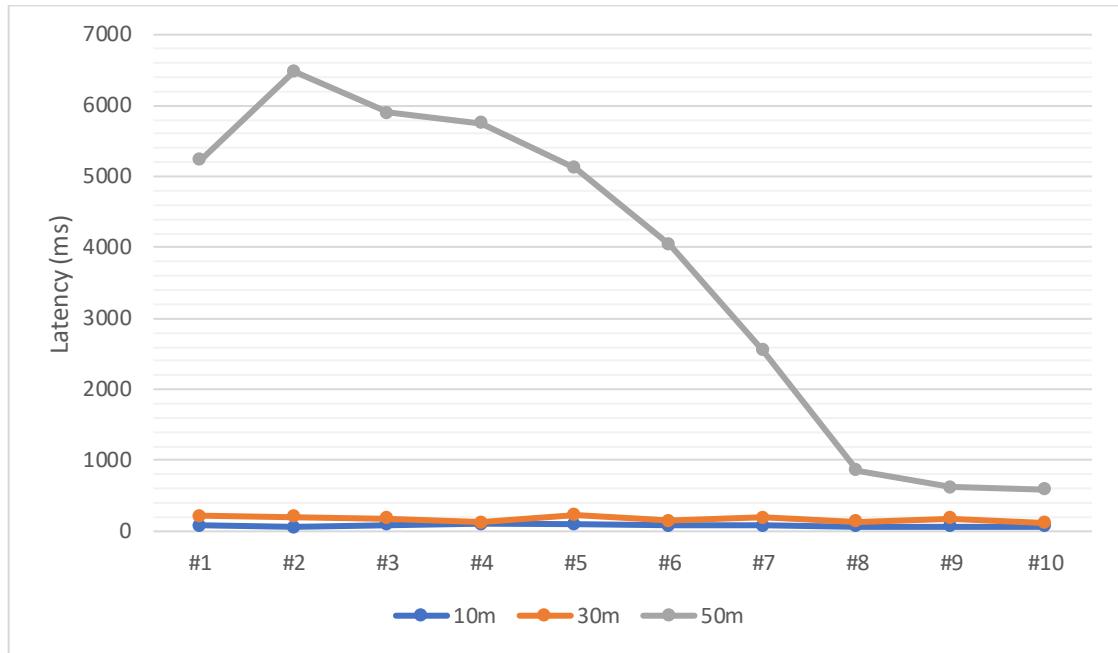
Dari tabel 4.6, didapatkan rata-rata *latency* dengan jarak 50 meter pada *node* 1 sebesar 1704.8 ms, *node* 2 sebesar 3714.2 ms dan *node* 3 sebesar 3670.4 ms. Dan rata-rata *latency* dari semua *node* dengan jarak 50 meter sebesar 3029.8 ms.

Berdasarkan data dari hasil pengujian diatas, didapatkan bahwa semakin jauh jarak antar *node* maka *latency* akan semakin meningkat, dan pada jarak 50 meter pengiriman data sudah tidak stabil.



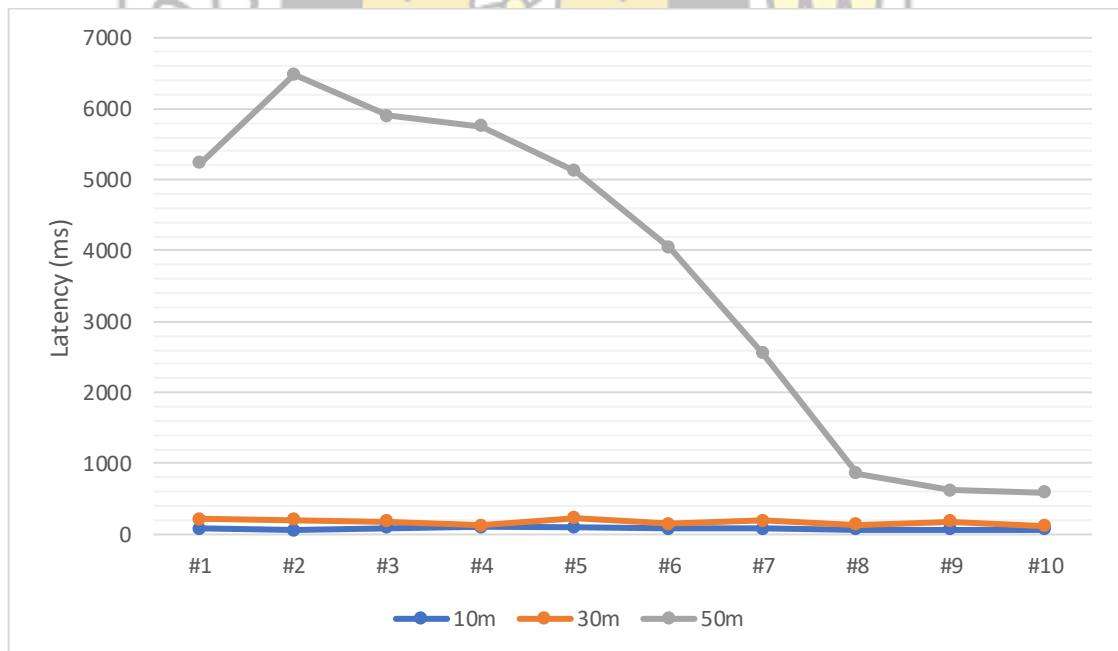
Gambar 4.29 Grafik Latency pada Node 1 ke Gateway

Gambar 4.26 menunjukkan grafik perubahan latency dalam mengirim data ke gateway pada Node 1 dengan perbandingan 3 jarak yang berbeda.



Gambar 4.30 Grafik Latency pada Node 2 ke Gateway

Gambar 4.27 menunjukkan grafik perubahan latency dalam mengirim data ke gateway pada Node 2 dengan perbandingan 3 jarak yang berbeda.



Gambar 4.31 Grafik Latency pada Node 3 ke Gateway

Gambar 4.28 menunjukkan grafik perubahan latency dalam mengirim data ke gateway pada Node 3 dengan perbandingan 3 jarak yang berbeda.

Gambar 4.29 – Gambar 4.31 menunjukkan grafik latency dalam mengirim data pada tiap Node ke gateway, dari grafik dapat dilihat ketidakstabilan latency yang terdata dengan pengujian pada jarak yang berbeda. Namun, dapat dilihat semakin lama pengujian dilakukan maka latency akan cenderung mengecil.

4.5.3 Pengujian Koneksi LoRa

Long Range (LoRa) sendiri menggunakan komunikasi frekuensi radio, karena itu pengujian koneksi antara *transmitter* dan *receiver* dilakukan dengan membandingkan RSSI (*Received Signal Strength Indicator*) ataupun kekuatan sinyal dan SNR (*Signal-to-Noise Ratio*) ataupun kualitas sinyal dengan jarak yang berbeda.

Tabel 4.7 Pengujian Jangkauan LoRa

Jarak (meter)	Kekuatan Sinyal/RSSI (dBm)	Kualitas Sinyal/SNR (dB)
5	--44	11.25
10	-44	11.45
20	-56	10.35
50	-64	10.25
100	-98	9.45
150	-118	-3.25
200	-122	-5.45

Dari tabel 4.7, dapat disimpulkan bahwa menggunakan komunikasi LoRa dapat menjangkau jarak yang relatif jauh. Dengan jarak 100 meter masih dapat terhubung dengan SNR bernilai positif yaitu sebesar 9.45 dB dan RSSI masih dibawah -100 yaitu sebesar -98 dBm. Pada jarak 150 meter, didapatkan nilai RSSI sebesar -118 dBm dan SNR bernilai negatif, yaitu sebesar -3.25 dB. Dan pada jarak 200 meter didapatkan sinyal yang sangat lemah dengan nilai RSSI sebesar -122 dBm dan SNR sebesar -5.45 dB, bahkan tidak terhubung.

BAB 5

KESIMPULAN DAN SARAN

5.1 Kesimpulan

Penelitian yang telah mengimplementasikan *Mesn Network* pada sistem monitoring penggunaan air dengan sensor *waterflow* YF-S201 yang terhubung dengan mikrokontroler *NodeMCU ESP32* dan ditampilkan pada *dashboard* platform *Node-RED* menggunakan protokol komunikasi MQTT. Adapun kesimpulan hasil uji yang didapat sebagai berikut.

1. Perangkat *NodeMCU ESP32* dapat menerapkan *mesh network* untuk menghubungkan *node* sensor berkomunikasi secara lokal.
2. Kinerja perangkat sensor *waterflow* YF-S201 yang sudah di-kalibrasi dapat menghitung debit air dan volume air secara *realtime* yang dipantau dari *dashboard* didapatkan tingkat *error* penghitungan volume air sebesar 6.73%.
3. Pengiriman data dari *node* sensor ke *gateway* dengan *mesh network* dapat dijangkau pada jarak 50 meter dengan *latency* rata-rata sebesar 3029.8 ms.
4. Untuk kinerja transfer data yang *realtime* dan efisien, *mesh network* dapat dihubungkan dengan komunikasi LoRa sehingga mikrokontroler ESP32 dapat mengirim data ke *server*.
5. Komunikasi LoRa menggunakan modul SX1276 dapat menjangkau jarak yang relatif jauh hingga 200 meter dengan tingkat RSSI sebesar -122 dBm dan SNR sebesar -5.45 dB.

5.2 Saran

Dari penelitian yang sudah dilakukan, penulis memberikan saran agar dapat meningkatkan sistem ini pada penelitian lebih lanjut sebagai berikut.

1. Meningkatkan akurasi penghitungan debit air dan volume air dengan melakukan efisiensi sistem dalam mentransfer data hingga sampai ke *server*.
2. Untuk skala produksi, perlu dilakukan penelitian terkait konsumsi daya listrik yang diperlukan sistem dan strategi ketika sumber listrik padam.
3. Dalam menggunakan komunikasi LoRa dapat ditingkatkan jarak jangkauan dan kekuatan sinyal dengan menggunakan modul LoRa yang terbaru.
4. Untuk diterapkan pada daerah dengan lanskap yang terdapat lebih banyak *noise*, dapat menggunakan modul LoRa di setiap *node* sensor.

DAFTAR PUSTAKA

- Akyildiz, I. F., & Wang, X. (2005). A survey on wireless mesh networks. *IEEE Communications magazine*, 43(9), S23-S30.
- Alexandra, Vieira., Carlos, Patrão., Tiago, Gonçalves., Paulo, Monteiro., Sylvain, Marcelino., Edmundo, Faria, Filipe., João, Damasceno., Hélio, Pereira., Catarina, Sousa., Paulo, J., Oliveira., João, Carvalho. (2019). WaterAMI - Water Automated Metering Infrastructure Based on an Energy Aware Wireless Mesh Network Communication Protocol. 209-220. doi: 10.1007/978-3-030-45694-8_16.
- Babiuch, M., Foltýnek, P., & Smutný, P. (2019). Using the ESP32 microcontroller for data processing. In 2019 20th International Carpathian Control Conference (ICCC) (pp. 1-6). IEEE.
- Brajesh, Panth. (2023). Several Routing Protocols, Features and Limitations for Wireless Mesh Network (WMN): A Review. doi: 10.1007/978-981-19-5936-3_18.
- Components101, 2021. YF-S201-Water Flow Measurement Sensor [Online] Available at: <https://components101.com/sensors/yf-s201-water-flow-measurement-sensor> [Diakses 26 Februari 2024].
- Davoli, Luca, Gianluigi Ferrari. (2022). Wireless Mesh Networks for IoT and Smart Cities: Technologies and applications. In *Institution of Engineering and Technology eBooks*. doi: 10.1049/pbte101e.
- Dian Vitta Agustina. (2007). Analisa Kinerja Sistem Distribusi Air Bersih Pdam Kecamatan Banyumanik Di Perumnas Banyumanik (Studi Kasus Perumnas Banyumanik Kel. Srondol Wetan). Tesis. Semarang: Universitas Diponegoro.
- Electronic Clinic, 2021. ESP32 Vs ESP8266 Nodemcu, ESP32 Vs Nodemcu, ESP8266 Vs ESP32. [Online] Available at: <https://www.electronicclinic.com/esp32-vs-esp8266-nodemcu-esp32-vs-nodemcu-eps8266-vs-esp32/> [Diakses 26 Februari 2024].
- Firmansyah, D. W., Ichsan, M. H. H., & Bhawiyuga, A. (2020). Pengembangan gateway LoRA-MQTT untuk transmisi data dua arah antara wireless sensor

- network dan cloud server. *Jurnal Pengembangan Teknologi Informasi dan Ilmu Komputer*, 4(1), 397-405.
- Gu, C., Jiang, L., & Tan, R. (2018). LoRa-Based Localization: Opportunities and Challenges. *ArXiv*, abs/1812.11481.
- Gudi, G., Tushar G., Akshay K., Raj K., Vyankatesh K. (2019). Design and Development of Water Flow Sensor. *IOSR Journal of Engineering*. 58-62.
- Haritsa, T., Yashu, B., Kumar, U., & Suma, M. (2020). Mathematical Characterization and Simulation of LoRa. *Wireless Personal Communications*, 115, 1481 - 1506. <https://doi.org/10.1007/s11277-020-07638-y>.
- Kodali, R. K., & Anjum, A. (2018). IoT based home automation using *Node-RED*. In *2018 Second International Conference on Green Computing and Internet of Things (ICGCIoT)* (pp. 386-390). IEEE.
- Kurniati, S., Kom, M., Saptadi, I. N. T. S., Kom, S., Pardosi, V. B. A., Kom, S., ... & Ilham, S. T. (2024). *INTERNET OF THING*. CV Rey Media Grafika.
- Li, C., & Cao, Z. (2022). LoRa Networking Techniques for Large-scale and Long-term IoT: A Down-to-top Survey. *ACM Computing Surveys (CSUR)*, 55, 1 - 36.
- Maulidin, M. A. R., Ali, T. N., & Mustofa, M. I. (2020). Perancangan Sistem Monitoring Penggunaan Air Pam Berbasis IoT Dengan Bot Telegram. *Indonesian Journal Of Technology, Informatics And Science (Ijtis)*, 2(1), 46-50.
- Mishra, B., Mishra, B., & Kertesz, A. (2021). Stress-testing MQTT brokers: A comparative analysis of performance measurements. *Energies*, 14(18), 5817.
- Paksi, Y. E. E., Prihartono, E., & Vitianingsih, A. V. (2021). Sistem Monitoring Pemakaian Air PDAM Tirta Kencana Kota Samarinda Berbasis Arduino. *JIMP (Jurnal Informatika Merdeka Pasuruan)*, 5(3).
- Rizqi Akbar. 2023. Flip.id: Ini Dia Cara Kerja Meteran Air PDAM untuk Tahu Penggunaan Air. [Online] Available at: <https://flip.id/blog/cara-kerja-meteran-air-pdam> [Diakses 26 Februari 2024]
- Rizqulloh, M. A., Wahyudin, D., & Pramudita, R. (2022). Distribusi Air Ledeng Dan Metering Menggunakan Mesh Network Untuk Perumahan. *Jurnal Ilmiah Teknologi Infomasi Terapan*, 8(2).
- Rofiqoh Permata Sari. (2015). Analisis Self-Configuration Dan Self-Healing Di Lingkungan Wireless Mesh Network. Skripsi. Malang: Universitas Brawijaya.

- Shinde, S., Nimkar, P., Singh, S., Salpe, V., & Jadhav, Y. (2016). MQTT-Message Queuing Telemetry Transport protocol. *International Journal of Research*, 3, 240-244.
- Simpkin, C., Taylor, I., Harborne, D., Bent, G., Preece, A., & Ganti, R. (2018). Dynamic Distributed Orchestration of *Node-RED* IoT Workflows Using a Vector Symbolic Architecture. *2018 IEEE/ACM Workflows in Support of Large-Scale Science (WORKS)*, 52-63.
- Sood, R., Kaur, M., & Lenka, H. (2013). Design and development of automatic water flow meter. *International journal of computer science, engineering and applications*, 3(3), 49.
- Vebby Aprilyan Alhadi. (2008). Implementasi Dan Unjuk Kerja Hybrid Wireless Mesh Network Dengan Menggunakan Protokol Routing AODV-UU Dan UOBWINAODV. Skripsi. Depok: Universitas Indonesia.
- WatElectronics, 2023. DS3231 RTC Module : Pin Configuration, Specifications, Interfacing with Microcontroller & Its Applications [Online] Available at: <https://www.watelectronics.com/ds3231-rtc-module/> [Diakses 26 Februari 2024].
- Wicaksono, R. (2017). Pengaruh Pemisah Udara Pada Pembacaan Hasil Pengukuran Meter Air Digital Skala Rumah Tangga. *Autocracy: Jurnal Otomasi, Kendali, dan Aplikasi Industri*, 4(02), 63-38.