

**APLIKASI *MOBILE* ANDROID BERBASIS MQTT UNTUK
MONITORING DAN KONTROL SISTEM AUDIO
MULTI-ZONA MASJID**

SKRIPSI

**Diajukan untuk memenuhi sebagian persyaratan mendapatkan gelar Strata Satu
Program Studi Informatika**



IMAM WAHYU TRI SUJANTO

22EO10047

**PROGRAM STUDI INFORMATIKA
FAKULTAS MATEMATIKA DAN ILMU KOMPUTER
UNIVERSITAS NAHDLATUL ULAMA AL GHAZALI
CILACAP
2026**

**APLIKASI *MOBILE* ANDROID BERBASIS MQTT UNTUK
MONITORING DAN KONTROL SISTEM AUDIO
MULTI-ZONA MASJID**

SKRIPSI

**Diajukan untuk memenuhi sebagian persyaratan mendapatkan gelar Strata Satu
Program Studi Informatika**



IMAM WAHYU TRI SUJANTO

22EO10047

**PROGRAM STUDI INFORMATIKA
FAKULTAS MATEMATIKA DAN ILMU KOMPUTER
UNIVERSITAS NAHDLATUL ULAMA AL GHAZALI
CILACAP
2026**

HALAMAN PERNYATAAN KEASLIAN SKRIPSI

Dengan ini saya menyatakan bahwa penulisan skripsi dengan judul “Aplikasi *Mobile* Android Berbasis MQTT untuk *Monitoring* dan Kontrol Sistem Audio Multi-Zona Masjid” adalah hasil karya saya dengan arahan dari pembimbing dan belum diajukan kepada pihak manapun. Sumber informasi yang dikutip dalam skripsi ini telah dicantumkan dalam Daftar Pustaka.

Demikian pernyataan ini dibuat dengan sebenarnya. Apabila di kemudian hari terdapat ketidaksesuaian dalam pernyataan ini, maka saya bersedia menerima sanksi sesuai dengan peraturan yang berlaku.

Cilacap, 17 Juli 2026

A handwritten signature in black ink is written over a red circular official stamp. The stamp contains the text "KEMENTERIAN" at the top, "PENDAGRI" in the middle, and "CILACAP" at the bottom. To the left of the stamp is a vertical rectangular stamp with the text "KEMENTERIAN" and "PENDAGRI" visible. Below the signature, the text "22EO10047" is printed.

Imam Wahyu Tri Sujianto
22EO10047

LEMBAR PENGESAHAN

Skripsi Saudara

Nama : Imam Wahyu Tri Sujianto
NIM : 22EO10047
Fakultas/Prodi : Fakultas MIKOM / Informatika
Judul : Aplikasi *Mobile* Android Berbasis MQTT untuk *Monitoring* dan Kontrol Sistem Audio Multi-Zona Masjid

Telah disidangkan oleh Dewan Penguji Fakultas Matematika dan Ilmu Komputer Universitas Nahdlatul Ulama Al Ghazali (UNUGHA) Cilacap pada hari / tanggal :

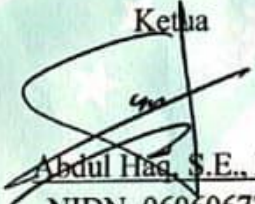
Jumat, 03 Juli 2026

Dan dapat diterima sebagai pemenuhan tugas akhir mahasiswa Program Strata Satu (S.1) Program Studi Informatika (S.Kom.) Fakultas Matematika dan Ilmu Komputer (FMIKOM) pada Universitas Nahdlatul Ulama Al Ghazali (UNUGHA) Cilacap.

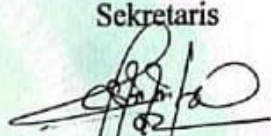
Cilacap, 17 Juli 2026

Dewan Sidang

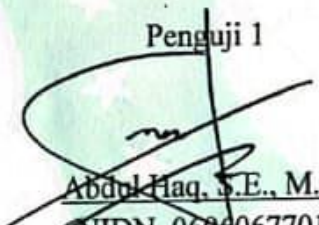
Ketua


Abdul Haq, S.E., M.Cs.
NIDN. 0606067701

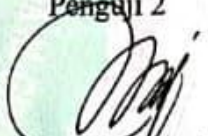
Sekretaris


Safiq Rosad, S.T., M.Kom.
NIDN. 0609018101

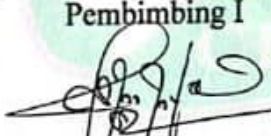
Penguji 1


Abdul Haq, S.E., M.Cs.
NIDN. 0606067701

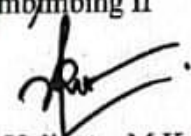
Penguji 2


Dr. M. Khanif, M.Pd.B.I.
NIDN. 0617058703

Pembimbing I


Safiq Rosad, S.T., M.Kom.
NIDN. 0609018101

Pembimbing II


Koko Edy Yulianto, M.Kom.
NIP. 230714144

Mengetahui,

Dekan Fakultas Matematika dan Ilmu Komputer


Mochamad T. A. Aziz Zein, S.Si., M.Kom.
NIDN. 2125098601

HALAMAN NOTA KONSULTAN

Abdul Haq, S.E., M.Cs.

Dosen Fakultas Matematika dan Ilmu Komputer
Universitas Nahdlatul Ulama Al Ghazali Cilacap

NOTA KONSULTAN

Hal : Skripsi Saudara Imam Wahyu Tri Sujianto
Lampiran : -

Kepada
Yth. Dekan Fakultas Matematika dan Ilmu Komputer
Universitas Nahdlatul Ulama Al Ghazali Cilacap
di Cilacap

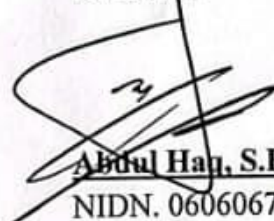
Assalamu'alaikum Wr. Wb.

Setelah membaca, memeriksa dan melakukan perbaikan seperlunya maka skripsi saudara :

Nama : Imam Wahyu Tri Sujianto
NIM : 22EO10047
Prodi : Informatika
Judul : Aplikasi *Mobile* Android Berbasis MQTT untuk *Monitoring* dan Kontrol
Sistem Audio Multi-Zona Masjid

Dapat diajukan ke Fakultas Matematika dan Ilmu Komputer, Universitas Nahdlatul Ulama Al Ghazali Cilacap untuk memenuhi syarat memperoleh gelar Strata Satu (S1).
Wassalamu'alaikum Wr. Wb.

Cilacap, 17 Juli 2026
Konsultan


Abdul Haq, S.E., M.Cs.
NIDN. 0606067701

NOTA PEMBIMBING

Cilacap, 25 Juni 2026

Kepada Yth :
Fakultas Matematika dan Komputer (FMIKOM)
UNUGHA Cilacap
di Cilacap

Assalamu'alaikum Wr. Wb.

Setelah melakukan bimbingan, telaah, arahan dan koreksi tahap penulisan skripsi saudara:

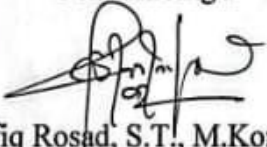
Nama : Imam Wahyu Tri Sujianto
NIM : 22EO10047
Fakultas : Matematika dan Ilmu Komputer
Prodi : Informatika
Judul : Aplikasi *Mobile* Android Berbasis MQTT untuk *Monitoring* dan Kontrol Sistem Audio Multi-Zona Masjid

Kami berpendapat bahwa skripsi tersebut sudah dapat diajukan ke sidang skripsi. Bersamaan ini kami kirimkan skripsi tersebut, semoga dapat segera disidangkan.

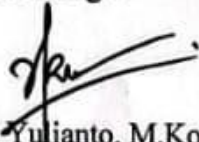
Atas perhatiannya kami ucapkan terima kasih.
Wassalamu'alaikum Wr. Wb.

Mengetahui,

Pembimbing I


Safiq Rosad, S.T., M.Kom.
NIDN. 0609018101

Pembimbing II


Koko Edy Yulianto, M.Kom.
NIP. 230714144

HALAMAN MOTO

“Allah tidak akan mengubah nasib suatu kaum hingga mereka mengubah apa yang ada pada diri mereka sendiri.”

(QS. Ar-Ra’d : 11)

"Kurang cerdas dapat diperbaiki dengan belajar. Kurang cakap dapat dihilangkan dengan pengalaman. Namun tidak jujur sukar diperbaiki."

(Mohammad Hatta)

Don't stop when you tired, stop when you are done.

Jangan berhenti ketika lelah, berhentilah ketika selesai.

(Imam Wahyu Tri Sujianto)

HALAMAN PERSEMBAHAN

Segala puji bagi Allah SWT, Rabb semesta alam, atas segala limpahan rahmat, kasih sayang, kesehatan, kekuatan, serta kemudahan yang telah diberikan sehingga penulis dapat menyelesaikan skripsi ini. Dengan penuh rasa syukur dan hormat, karya sederhana ini penulis persembahkan kepada :

1. Kedua orang tua tercinta, Bapak Sugiyo, A.Md.Pt. dan Ibu Murtofi'ah, yang telah menjadi alasan terbesar penulis untuk terus berjuang. Terima kasih atas setiap doa yang tak pernah putus, kasih sayang yang tulus, pengorbanan yang tak terhingga, nasihat yang menenangkan, serta dukungan yang selalu menguatkan di setiap langkah kehidupan. Semoga Allah SWT senantiasa melimpahkan kesehatan, kebahagiaan, dan keberkahan kepada Bapak dan Ibu.
2. Kakak tercinta Shofia Nuraini Umami, S.Hum. serta adik dan saudara tersayang, Nafa Rizki Catur S. dan Akmal Zaki Fitriansyah, yang selalu menghadirkan semangat, doa, perhatian, dan keceriaan. Terima kasih telah menjadi tempat pulang, berbagi cerita, dan sumber kekuatan ketika lelah menyapa.
3. Seseorang yang istimewa dengan inisial NIM 22BH10062, terima kasih telah hadir sebagai sosok yang senantiasa memberikan dukungan, motivasi, doa, perhatian, serta keyakinan agar penulis tidak mudah menyerah. Terima kasih karena telah menjadi pendengar yang baik di tengah berbagai keluh kesah, mengingatkan untuk tetap kuat ketika keadaan terasa berat, dan turut mewarnai perjalanan panjang dalam menyelesaikan skripsi ini. Semoga segala kebaikan yang telah diberikan mendapat balasan terbaik dari Allah SWT.
4. Keluarga besar FMIKOM Angkatan 2022, terima kasih atas kebersamaan, persahabatan, tawa, kerja sama, serta kenangan indah yang telah terukir selama masa perkuliahan. Semoga tali silaturahmi dan kekeluargaan ini tetap terjaga.
5. Teman-teman HIMAFORTIC (Himpunan Mahasiswa Informatika) UNUGHA, terima kasih atas pengalaman berorganisasi, kebersamaan, solidaritas, serta semangat yang telah diberikan. Terima kasih telah menjadi ruang untuk belajar, bertumbuh, bertukar pikiran, dan saling menguatkan.

Semoga kekompakan, persaudaraan, dan semangat untuk terus berkarya demi kemajuan organisasi senantiasa terjaga.

6. Seluruh teman-teman Pimpinan Cabang IPNU Cilacap, terima kasih atas kebersamaan, persaudaraan, pengalaman berorganisasi, semangat pengabdian, serta dukungan dan doa yang telah diberikan kepada penulis. Terima kasih telah menjadi keluarga dalam perjuangan, tempat belajar, berbagi pengalaman, serta saling menguatkan dalam setiap proses. Semoga ukhuwah, semangat juang, dan dedikasi dalam berkhidmat kepada organisasi, agama, bangsa, dan masyarakat senantiasa terjaga serta mendapat keberkahan dari Allah SWT.
7. Seluruh teman-teman Universitas Nahdlatul Ulama Al Ghazali (UNUGHA) yang tidak dapat penulis sebutkan satu per satu. Terima kasih atas doa, dukungan, bantuan, dan semangat yang telah diberikan selama proses penyusunan skripsi ini.

Akhir kata, karya ini bukanlah akhir dari sebuah perjalanan, melainkan awal untuk melangkah menuju pengabdian yang lebih luas. Semoga segala ilmu yang diperoleh dapat menjadi manfaat bagi diri sendiri, keluarga, masyarakat, serta bernilai ibadah di sisi Allah SWT.

KATA PENGANTAR

Puji syukur kepada Allah SWT atas segala limpahan nikmat dan karuniaNya, sehingga penulis dapat menyelesaikan penulisan skripsi. Sholawat dan salam senantiasa penulis haturkan kepada Nabi Muhammad SAW sebagai pembimbing seluruh umat manusia.

Skripsi ini tidak akan selesai tanpa adanya bantuan dari banyak pihak, karena itu penulis menyampaikan terima kasih kepada :

1. Bapak Mochamad T. A. Aziz Zein, S.Si., M.Kom., Dekan FMIKOM UNUGHA.
2. Bapak Safiq Rosad, S.T., M.Kom., pembimbing I yang telah dengan sabar memberikan bimbingan dan petunjuk dalam menyelesaikan skripsi ini.
3. Bapak Koko Edy Yulianto, M.Kom., pembimbing II yang telah dengan sabar memberikan bimbingan dan petunjuk dalam menyelesaikan skripsi ini.
4. Bapak Verry, S.T., M.Kom., sebagai Pembimbing Akademis.
5. Pihak Instansi dan teman-teman dari SHATOMEDIA.
6. Bapak-Ibu dosen Program Studi Informatika FMIKOM UNUGHA.
7. Semua pihak yang tidak dapat penulis sebutkan satu persatu.

Semoga Allah SWT membalas jerih payah dan pengorbanan yang telah diberikan dengan balasan yang lebih baik. Aamiin. Penulis berharap semoga karya kecil ini bermanfaat bagi pembaca.

Cilacap, 12 Juni 2026

Imam Wahyu Tri Sujianto

ABSTRAK

IMAM WAHYU TRI SUJANTO. Aplikasi *Mobile* Android Berbasis MQTT untuk *Monitoring* dan Kontrol Sistem Audio Multi-Zona Masjid. Dibimbing oleh SAFIQ ROSAD, S.T.,M.Kom. dan KOKO EDY YULIANTO, M.Kom.

Sistem distribusi audio multi-zona masjid yang dikembangkan dengan arsitektur *Cyber-Physical System* (CPS) memerlukan antarmuka pengguna yang mampu mendukung proses *monitoring* dan kontrol sistem secara *real-time*. Penelitian ini bertujuan untuk mengembangkan aplikasi *mobile* Android berbasis MQTT yang dapat digunakan untuk memantau kondisi sistem audio multi-zona serta melakukan kontrol volume dan *mute* pada setiap zona.

Metode penelitian yang digunakan adalah *Research and Development* (R&D) yang meliputi tahapan analisis kebutuhan sistem, perancangan antarmuka pengguna (UI/UX), implementasi aplikasi *mobile* Android menggunakan *framework* Flutter dengan pola arsitektur *Model-View-ViewModel* (MVVM), integrasi komunikasi data menggunakan protokol MQTT, serta pengujian sistem melalui *Black-box Testing* dan pengukuran *response time* MQTT. Aplikasi dikembangkan untuk menampilkan data *monitoring* berupa status keaktifan *node* (*online/offline*) dan nilai *Sound Pressure Level* (SPL) secara *real-time* serta menyediakan fitur kontrol volume dan *mute* yang terintegrasi dengan *backend system* melalui MQTT broker.

Hasil penelitian menunjukkan bahwa aplikasi berhasil menampilkan data *monitoring* sistem secara *real-time* dan mampu mengirimkan instruksi kontrol volume maupun *mute* untuk diproses oleh *backend system* melalui mekanisme *publish-subscribe* MQTT. Hasil pengujian *Black-box Testing* menunjukkan bahwa seluruh fitur utama aplikasi berjalan sesuai dengan kebutuhan fungsional yang telah ditetapkan. Selain itu, pengujian *response time* MQTT menunjukkan bahwa komunikasi data antara aplikasi Android dan *backend system* dapat berlangsung secara responsif sehingga mendukung kebutuhan *monitoring* dan kontrol *real-time*.

Kontribusi penelitian ini meliputi pengembangan aplikasi *mobile* Android berbasis Flutter untuk mendukung *monitoring* dan kontrol sistem audio multi-zona masjid, penerapan arsitektur MVVM untuk mendukung pengelolaan komunikasi data berbasis MQTT pada aplikasi Android, serta evaluasi performa komunikasi menggunakan parameter *response time* MQTT pada lingkungan sistem audio berbasis CPS.

Kata kunci: Android, *Cyber-Physical System*, Flutter, *Monitoring* dan Kontrol, MQTT, Multi-Zona Audio, *Smart Mosque*.

ABSTRACT

IMAM WAHYU TRI SUJANTO. *MQTT-Based Android Mobile Application for Monitoring and Controlling a Mosque's Multi-Zone Audio System. Supervised by SAFIQ ROSAD, S.T., M.Kom. and KOKO EDY YULIANTO, M.Kom.*

A mosque's multi-zone audio distribution system, developed using a Cyber-Physical System (CPS) architecture, requires a user interface capable of supporting real-time system monitoring and control. This research aims to develop an MQTT-based Android mobile application that can be used to monitor the condition of the multi-zone audio system and control the volume and mute in each zone.

The research method used is Research and Development (R&D), which includes system requirements analysis, user interface (UI/UX) design, Android mobile application implementation using the Flutter framework with a Model-View-ViewModel (MVVM) architectural pattern, data communication integration using the MQTT protocol, and system testing through Black-box Testing and MQTT response time measurements. The application was developed to display monitoring data in the form of node activity status (online/offline) and Sound Pressure Level (SPL) values in real-time, while also providing volume and mute control features integrated with the backend system via an MQTT broker.

The results showed that the application successfully displayed real-time system monitoring data and was able to send volume and mute control instructions for processing by the backend system via the MQTT publish-subscribe mechanism. Black-box Testing results indicated that all key application features ran according to the established functional requirements. Furthermore, MQTT response time testing demonstrated that data communication between the Android application and the backend system was responsive, supporting real-time monitoring and control needs.

The contribution of this research includes the development of a Flutter-based Android mobile application to support monitoring and control of a multi-zone mosque audio system, the implementation of an MVVM architecture to support MQTT-based data communication management in Android applications, and evaluation of communication performance using MQTT response time parameters in a CPS-based audio system environment.

Keywords: Android, Cyber-Physical System, Flutter, Monitoring and Control, MQTT, Multi-Zone Audio, Smart Mosque.

DAFTAR ISI

HALAMAN PERNYATAAN KEASLIAN SKRIPSI	iii
LEMBAR PENGESAHAN	iv
HALAMAN NOTA KONSULTAN.....	v
NOTA PEMBIMBING	vi
HALAMAN MOTO	vii
HALAMAN PERSEMBAHAN	viii
KATA PENGANTAR	x
ABSTRAK.....	xi
<i>ABSTRACT</i>	xii
DAFTAR ISI.....	xiii
DAFTAR GAMBAR	xv
DAFTAR TABEL.....	xvi
BAB I PENDAHULUAN	1
A. Latar Belakang	1
B. Rumusan Masalah.....	2
C. Batasan Masalah	2
D. Tujuan Penelitian	3
E. Manfaat Penelitian	3
F. Kontribusi Penelitian	4
G. Keaslian Penelitian.....	4
H. Keterbaruan Penelitian.....	5
BAB II TINJAUAN PUSTAKA	6
A. Penelitian Terkait	6
B. Landasan Teori.....	16
C. Studi Terkait.....	22
BAB III METODOLOGI PENELITIAN.....	23
A. Waktu dan Tempat Penelitian.....	23
B. Alat dan Bahan.....	24
C. Prosedur Penelitian	25
D. Analisis Data.....	45

E.	Jadwal Penelitian	47
BAB IV HASIL DAN PEMBAHASAN.....		49
A.	Hasil	49
B.	Pembahasan.....	66
BAB V KESIMPULAN.....		72
A.	Kesimpulan	72
B.	Saran/Rekomendasi.....	72
DAFTAR PUSTAKA		74
DAFTAR LAMPIRAN		77

DAFTAR GAMBAR

Gambar 2.1 <i>Mind Map</i> Penelitian	14
Gambar 2.2 Taksonomi Bloom	15
Gambar 3.1 Diagram Alur Penelitian.....	26
Gambar 3.2 Flowchart Sistem.....	29
Gambar 3.3 <i>Login page</i> dan Dashboard <i>Mobile</i> Android	31
Gambar 3.4 Arsitektur Sistem Distribusi Audio Berbasis MQTT	32
Gambar 3.5 Integrasi dengan MQTT broker dan Kontrol Zona	33
Gambar 3.6 Pengembangan Aplikasi dan Implementasi Sistem	34
Gambar 3.7 Pengujian Aplikasi dan Sistem.....	39
Gambar 4.1 Arsitektur Sistem yang digunakan Pada Penelitian Ini	50
Gambar 4.2 Pemantauan <i>Backend</i> melalui CMD	51
Gambar 4.3 Halaman <i>Login</i> Aplikasi.	52
Gambar 4.4 Dashboard <i>Monitoring</i> Aplikasi.....	53
Gambar 4.5 Halaman Detail Zona	54
Gambar 4.6 Kontrol Volume dan <i>Mute</i>	55
Gambar 4.7 Tampilan Halaman Notifikasi Perangkat Tidak Normal.....	56
Gambar 4.8 Alur Komunikasi MQTT yang diterapkan pada Sistem.....	56
Gambar 4.9 <i>Code Connect</i> ke Alamat Broker	57
Gambar 4.10 Tampilan Aplikasi Terhubung ke MQTT broker	58
Gambar 4.11 <i>Code Subscribe</i> ke Topik	58
Gambar 4.12 Validasi Kontrol Volume	58
Gambar 4.13 Tampilan Aplikasi <i>Mute</i> Zona.....	59
Gambar 4.14 Validasi Kontrol <i>Mute</i>	59
Gambar 4.15 Tampilan Aplikasi <i>Mute</i> Zona.....	60
Gambar 4.16 Tampilan <i>Monitoring</i> BE	61
Gambar 4.17 Tampilan Status dan Zona SPL pada Aplikasi.....	62
Gambar 4.18 Tampilan Kondisi Perangkat dan Status Node.....	63
Gambar 4.19 Pengujian <i>Response time</i>	65

DAFTAR TABEL

Tabel 2.1 Rangkuman Penelitian Terdahulu	6
Tabel 2.2 Rangkuman Gap Penelitian dan Novelty	11
Tabel 3.1 Alat dan Bahan yang digunakan dalam penelitian.....	24
Tabel 3.2 Pengembangan dan Pengujian Fitur Aplikasi	36
Tabel 3.3 Pengujian Fungsionalitas Aplikasi.....	40
Tabel 3.4 Jadwal Penelitian.....	48
Tabel 4.1 Hasil Pengujian Koneksi MQTT	61
Tabel 4.2 Pengujian Sinkronisasi Data <i>Monitoring</i>	62
Tabel 4.3 Pengujian Notifikasi Kondisi Perangkat	63
Tabel 4.4 Hasil Pengujian <i>Black-box Testing</i>	64
Tabel 4.5 Pengujian <i>Response Time</i> MQTT	66

BAB I PENDAHULUAN

A. Latar Belakang

Sistem distribusi audio merupakan komponen penting dalam mendukung kelancaran komunikasi di berbagai ruang publik, termasuk di masjid untuk kegiatan khotbah maupun salat[1], [2]. Di Indonesia, mayoritas masjid (sekitar 93,1%) masih menggunakan sistem pengeras suara *Public Address* (PA) analog yang terpusat untuk mengatur distribusi suara[2]. Meskipun sistem ini sudah umum digunakan, sistem PA terpusat memiliki berbagai keterbatasan teknis, seperti distribusi suara yang tidak merata di area belakang, munculnya zona mati (*dead zones*), serta gangguan akustik berupa gema (*echo*) atau *noise* akibat penempatan speaker yang tidak selaras dengan geometri ruangan[2]. Masalah-masalah ini secara signifikan mengurangi tingkat intelligibilitas suara (*Speech Transmission Index*) dan kenyamanan jamaah dalam beribadah[2].

Dalam beberapa tahun terakhir, perkembangan teknologi *Internet of Things* (IoT) dan *Cyber-Physical System* (CPS) menawarkan solusi yang lebih fleksibel dan efisien melalui koordinasi cerdas antara sistem komputasi dan proses fisik di lingkungan masjid[1]. Salah satu pendekatan inovatif yang mulai dipertimbangkan adalah sistem distribusi audio berbasis zonasi (*speaker zoning*)[1], [2]. Dalam sistem ini, setiap zona masjid memiliki kontrol audio independen, seperti pengaturan volume dan *mute*, yang memungkinkan distribusi suara lebih seragam (*uniform*) dan meminimalkan redaman sinyal di area tertentu. Pendekatan berbasis CPS ini mempermudah pengguna aplikasi dalam mengelola perangkat secara adaptif sesuai dengan luas area yang digunakan[1], [2].

Pengelolaan sistem audio masjid kini dapat dioptimalkan melalui *Android Development* untuk menciptakan solusi *Mobile Application* yang praktis dan terjangkau[3], [4]. Aplikasi *mobile* memungkinkan pengguna aplikasi untuk melakukan *monitoring* dan kontrol sistem secara *real-time* langsung dari *smartphone* mereka[4], [5], [6]. Untuk menjamin efisiensi komunikasi data antara aplikasi dan perangkat di lapangan, protokol MQTT (*Message Queuing Telemetry Transport*) digunakan sebagai standar komunikasi ringan berbasis *publish-subscribe*[7], [8]. Protokol ini sangat efisien karena membutuhkan bandwidth rendah dan cocok untuk perangkat dengan spesifikasi mesin terbatas, sehingga memastikan pengiriman instruksi kontrol ke setiap zona audio tetap stabil meskipun dalam kondisi jaringan yang tidak menentu[8].

Berdasarkan uraian tersebut, dapat disimpulkan bahwa pengembangan aplikasi *mobile* tidak hanya berfungsi sebagai pelengkap, tetapi menjadi komponen antarmuka (*interface*) yang mendukung *monitoring* dan kontrol sistem distribusi audio berbasis IoT untuk memastikan sistem dapat digunakan secara efektif oleh pengguna. Oleh karena itu, penelitian ini bertujuan mengembangkan aplikasi

mobile Android berbasis MQTT sebagai antarmuka *monitoring* dan kontrol pada sistem audio multi-zona masjid. Penelitian ini berfokus pada pengembangan aplikasi Android yang terintegrasi dengan *backend system* yang telah tersedia, sehingga aplikasi dapat digunakan untuk melakukan *monitoring* kondisi sistem serta mengirimkan instruksi kontrol secara *real-time* melalui protokol MQTT. Penelitian ini juga menekankan pada kemudahan operasional agar aplikasi dapat dijalankan dengan baik oleh pengguna. Untuk menjamin kualitas aplikasi yang dikembangkan, dilakukan pengujian fungsionalitas menggunakan metode *Black-box Testing* serta pengujian performa komunikasi melalui pengukuran *response time* MQTT guna memastikan bahwa seluruh fitur aplikasi berjalan sesuai kebutuhan sistem dan komunikasi data antara aplikasi Android dan *backend system* berlangsung secara responsif. Berdasarkan uraian latar belakang permasalahan di atas, penulis mengambil topik dengan judul **“APLIKASI MOBILE ANDROID BERBASIS MQTT UNTUK *MONITORING* DAN KONTROL SISTEM AUDIO MULTI-ZONA MASJID”**.

B. Rumusan Masalah

Berdasarkan latar belakang yang telah diuraikan, rumusan masalah dalam penelitian ini adalah sebagai berikut:

1. Bagaimana mengembangkan aplikasi *mobile* Android berbasis Flutter yang mampu menampilkan data *monitoring* serta menyediakan fitur kontrol volume dan *mute* secara fungsional?
2. Bagaimana mengimplementasikan mekanisme komunikasi data berbasis protokol MQTT agar aplikasi Android dapat berkomunikasi dengan *backend system* secara *real-time* dalam proses *monitoring* dan kontrol?
3. Bagaimana mengevaluasi kualitas aplikasi melalui pengujian fungsionalitas serta pengukuran *response time* komunikasi MQTT?

C. Batasan Masalah

Untuk menjaga fokus penelitian ini, beberapa batasan masalah yang ditetapkan adalah:

1. Pengembangan Aplikasi: Aplikasi dikembangkan khusus untuk platform Android menggunakan bahasa pemrograman Dart, *framework* Flutter, dan arsitektur MVVM dengan minimal SDK 26 dan target SDK 34.
2. Ruang Lingkup Monitoring: Fitur *monitoring* dibatasi pada visualisasi status keaktifan *node* (*online/offline*) dan nilai *Sound Pressure Level* (SPL) yang diperoleh dari *backend system* secara *real-time*.
3. Batasan Kontrol Audio: Perintah kontrol volume dan *mute* dibatasi pada proses pengiriman instruksi dari aplikasi Android hingga diproses oleh *backend system* melalui MQTT broker. Penelitian ini tidak mencakup implementasi maupun pengujian eksekusi kontrol pada perangkat keras audio atau *node* IoT.

4. Integrasi & Infrastruktur: *Backend system* yang meliputi MQTT broker, layanan *backend* (FastAPI), database PostgreSQL, dan *node* IoT telah tersedia sehingga tidak menjadi objek pengembangan maupun pengujian dalam penelitian ini. Penelitian hanya berfokus pada pengembangan aplikasi *mobile* Android dan integrasinya dengan infrastruktur tersebut.
5. Metode Pengujian: Pengujian dilakukan melalui *Black-box Testing* terhadap fungsi aplikasi serta pengukuran *response time* komunikasi MQTT. Penelitian tidak melibatkan pengujian *usability* menggunakan responden maupun kuesioner.

D. Tujuan Penelitian

Penelitian ini memiliki tujuan sebagai berikut:

1. Mengembangkan aplikasi *mobile* Android berbasis Flutter yang mampu menampilkan data *monitoring* serta menyediakan fitur kontrol volume dan *mute* secara fungsional.
2. Mengimplementasikan komunikasi data berbasis MQTT agar aplikasi Android dapat terintegrasi dengan *backend system* untuk mendukung proses *monitoring* dan kontrol secara *real-time*.
3. Mengevaluasi kualitas dan performa aplikasi melalui pengujian fungsionalitas fitur menggunakan metode *Black-box Testing* serta pengujian *response time* komunikasi MQTT.

E. Manfaat Penelitian

Manfaat penelitian ini diharapkan dapat memberikan kontribusi dalam aspek akademik, praktis, dan sosial bagi berbagai pihak yang terkait dengan pengembangan dan penerapan sistem yang diteliti.

1. Manfaat Akademik: Menjadi referensi dalam penelitian pengembangan aplikasi *mobile* Android berbasis *framework* Flutter dan arsitektur MVVM yang terintegrasi dengan protokol komunikasi MQTT untuk manajemen data sistem terdistribusi.
2. Manfaat Praktis: Menyediakan aplikasi *mobile* Android yang mampu menampilkan informasi *monitoring* status *node* dan nilai SPL secara *real-time* serta mendukung kontrol volume dan *mute* pada sistem audio multi-zona melalui komunikasi berbasis MQTT.
3. Manfaat Sosial: Memberikan kemudahan bagi pengelola masjid dalam memantau kondisi sistem audio multi-zona secara *real-time* sehingga proses pemeliharaan dan pengelolaan perangkat dapat dilakukan dengan lebih efektif dan efisien.

F. Kontribusi Penelitian

Penelitian ini memberikan kontribusi dalam pengembangan aplikasi *monitoring* dan kontrol sistem audio multi-zona masjid berbasis *Cyber-Physical System* (CPS) dan *Internet of Things* (IoT). Kontribusi penelitian tidak hanya berupa produk aplikasi yang dihasilkan, tetapi juga penerapan pendekatan pengembangan perangkat lunak dan komunikasi data yang mendukung sistem *monitoring* dan kontrol secara *real-time*.

1. Penerapan arsitektur perangkat lunak *Model-View-ViewModel* (MVVM) pada aplikasi Android berbasis Flutter untuk mendukung pengelolaan data dan komunikasi MQTT secara *real-time* sehingga antarmuka aplikasi tetap responsif dan mudah dikembangkan.
2. Perancangan model integrasi komunikasi data antara aplikasi Android, MQTT broker, dan *backend system* yang memungkinkan proses *monitoring* status *node*, visualisasi nilai *Sound Pressure Level* (SPL), serta pengiriman instruksi kontrol volume dan *mute* secara *real-time*.
3. Evaluasi performa komunikasi data berbasis MQTT menggunakan parameter *response time* sebagai dasar untuk menilai responsivitas komunikasi antara aplikasi Android dan *backend system* pada sistem audio multi-zona berbasis *Cyber-Physical System* (CPS).

G. Keaslian Penelitian

Keaslian penelitian ini terletak pada pengembangan aplikasi *mobile* Android yang terintegrasi dengan sistem distribusi audio masjid berbasis multi-zona menggunakan protokol MQTT untuk memungkinkan *monitoring* dan kontrol volume dan *mute* secara *real-time* melalui jaringan lokal. Penelitian sebelumnya menunjukkan bahwa sistem *Public Address* (PA) masjid masih menggunakan distribusi audio terpusat secara analog dan belum terintegrasi dengan sistem kontrol berbasis aplikasi *mobile* maupun komunikasi IoT secara *real-time*[9]. Selain itu, beberapa penelitian IoT berbasis MQTT umumnya hanya berfokus pada *monitoring* perangkat atau sensor tertentu dan belum mengimplementasikan kontrol sistem audio berbasis multi-zona melalui aplikasi *mobile* secara langsung[10].

Dengan demikian, keaslian penelitian ini terletak pada pengembangan aplikasi *mobile* Android menggunakan *framework* Flutter dan arsitektur MVVM yang difokuskan untuk melakukan *monitoring* status keaktifan *node*, visualisasi data SPL, serta manajemen pengiriman data kontrol volume dan *mute* berbasis multi-zona melalui protokol komunikasi MQTT.

H. Keterbaruan Penelitian

Penelitian ini menghadirkan kebaruan melalui pengembangan sistem *monitoring* dan kontrol distribusi audio masjid berbasis zonasi yang dapat dikendalikan secara *real-time* menggunakan aplikasi *mobile* Android. Berbeda dengan sistem audio konvensional yang umumnya masih dikendalikan secara manual dan terpusat pada perangkat fisik, penelitian ini mengintegrasikan protokol komunikasi MQTT sebagai mekanisme pertukaran data antara aplikasi *mobile* dan infrastruktur *backend* berbasis IoT.

Dengan demikian kebaruan dari penelitian ini adalah penerapan arsitektur MVVM pada aplikasi Android untuk mengelola *traffic* data komunikasi *publish-subscribe* MQTT secara *real-time*, yang dikhususkan untuk memantau nilai SPL dan kontrol volume dan *mute* pada sistem distribusi audio berbasis zonasi di masjid. Evaluasi kebaruan ini diukur berdasarkan performa parameter teknis jaringan (*response time*) dan kevalidan fungsionalitas sistem melalui eksperimen laboratorium.

BAB II TINJAUAN PUSTAKA

A. Penelitian Terkait

Untuk memperkuat landasan konseptual dan posisi penelitian ini, dilakukan kajian terhadap sejumlah penelitian terdahulu yang relevan dengan topik sistem audio masjid, *Internet of Things* (IoT), protokol MQTT, pengembangan aplikasi *mobile* Android, serta konsep *Cyber-Physical System* (CPS). Kajian ini bertujuan untuk mengidentifikasi pendekatan yang telah digunakan, capaian yang diperoleh, serta keterbatasan yang masih terdapat pada penelitian sebelumnya.

Ringkasan penelitian-penelitian tersebut disajikan pada Tabel 2.1, yang memuat informasi mengenai fokus penelitian, metode yang digunakan, serta gap yang masih terbuka. Melalui analisis komparatif tersebut, dapat diketahui posisi penelitian ini sekaligus menegaskan kontribusi dan kebaruan yang ditawarkan dibandingkan penelitian terdahulu.

Tabel 2.1 Rangkuman Penelitian Terdahulu

No.	Penulis (Tahun)	Keterangan
(1)	(2)	(3)
1	Rosad, Yudhana, et al., 2025b	Judul: <i>Zoned Audio Distribution: Systematic Review and Optimization Framework in Mosque</i> Metode: SLR, Survei Empiris, Simulasi Akustik Tools: ODEON, Ease Focus, Digital Signal Processing (DSP) Hasil: Sistem multi-zona meningkatkan <i>Speech Transmission Index</i> (STI)>0,6 dan kualitas suara lebih jelas.
2	Rosad, Irkhamudin, et al., 2025	Judul: <i>Edge-Based IoT EndPoint Performance Optimization for Smart Mosques</i> Metode: Eksperimental (<i>polling</i> vs <i>push</i>) Tools: ESP32, Raspberry Pi, WebSocket, SSE Hasil: Latensi turun dari 180 ms ke 50 ms dan tingkat ketersediaan sistem meningkat menjadi 99,3%.
3	Ariyanto et al., 2025	Judul: Perancangan Aplikasi <i>Mobile</i> Berbasis Flutter untuk Pemantauan Data Sensor IoT: Solusi bagi Manajemen Suhu dan Kelembapan Metode: <i>User acceptance Testing</i> (UAT) dengan 20 responden, SDLC Tools: Flutter, sensor DHT22, ESP32, protokol MQTT, database MySQL Hasil: Aplikasi Flutter berhasil melakukan <i>monitoring</i> data sensor IoT secara <i>real-time</i> melalui MQTT dengan pembaruan data yang stabil serta mendukung notifikasi kondisi sensor secara langsung.

No.	Penulis (Tahun)	Keterangan
(1)	(2)	(3)
4	Saiqul Umam et al., 2023	Judul: Implementasi Protokol MQTT Pada Aplikasi Smart Garden Berbasis Internet of Things Metode: Aplikasi smart garden berbasis IoT dengan protokol MQTT Tools: Protokol MQTT, mikrokontroler, sensor, MQTT broker, WiFi, web interface Hasil: Server MQTT berhasil menangkap data sensor dengan baik dan akurat, transmisi data stabil
5	Pratama & Irianto, 2026	Judul: Rancang Bangun Aplikasi Parkir Masjid Berbasis Android dan Firebase Metode: Model <i>Waterfall</i> Tools: Android Studio, Kotlin, Firebase Realtime Database Hasil: Aplikasi berhasil melakukan sinkronisasi data secara <i>real-time</i> dan seluruh fungsi sistem berjalan sesuai kebutuhan berdasarkan pengujian <i>Black-box Testing</i>.
6	Zainuddin Lapi et al., 2023	Judul: Rancang Bangun Purwarupa Aplikasi Masjid Pintar Menggunakan Platform Kodular Berbasis Internet di Masjid Metode: Pengembangan <i>prototype</i> Tools: NodeMCU Esp8266, relay, sensor LDR, Arduino IDE, Firebase, platform Kodular Hasil: Sistem berhasil mengontrol lampu dan memonitor intensitas cahaya dari jarak jauh
7	Thantawi et al., 2023	Judul: <i>Utilization and development of Smart Mosque Applications</i> Metode: Survei dan wawancara pengurus masjid Tools: G-Forms Hasil: Digitalisasi layanan masjid melalui aplikasi <i>mobile</i> membantu meningkatkan efektivitas pengelolaan operasional masjid dan akses informasi bagi pengguna.
8	Sanjaya et al., 2021	Judul: Penerapan <i>Internet of Things</i> Untuk Sistem <i>monitoring</i> Jemaah Masjid Sesuai Protokol Kesehatan Terhadap Virus Covid-19 Berbasis Arduino Metode: Simulasi dan pengujian akurasi sensor Tools: Arduino, sensor suhu, sensor jarak, LCD, web interface, bilik disinfektan Hasil: Sensor suhu akurasi 98,82%, sensor jarak akurasi 93,33%, <i>monitoring real-time</i> berhasil
9	Abdulfathah & Santoso, 2024	Judul: Pemanfaatan IoT (<i>Internet of Things</i>) Dalam <i>monitoring</i> Kadar Kepekatan Asap dan Kendali <i>Camera Tracking</i> Metode: Pengujian integrasi sistem dan pengujian fungsional

No.	Penulis (Tahun)	Keterangan
(1)	(2)	(3)
		Tools: Sensor MQ-2, Arduino Uno, ESP8266, motor servo, ponsel Android Hasil: Sistem berhasil terintegrasi, deteksi asap dan kontrol kamera berfungsi baik
10	Saputra et al., 2025	Judul: IoT-MQTT <i>Protocol-Based Water Sensor System</i> to Monitor Citarum River Water Quality Using Arduino Uno R4 WiFi Metode: Metode eksperimen IoT Tools: Arduino Uno R4 WiFi, sensor kualitas air, dan protokol MQTT Hasil: Sistem berhasil memonitor kualitas air secara <i>real-time</i> menggunakan MQTT dengan komunikasi data stabil, latensi rendah, dan efisien
11	Sosmita & Salam, 2024	Judul: Rancang Bangun Aplikasi <i>Mobile</i> Untuk Pemantauan Dan Kontrol Sistem Keamanan Pintu Rumah Berbasis IoT Metode: Pengujian sistem dan pengujian fungsional Tools: NodeMCU/ESP32, sensor PIR, solenoid door lock, relay, Kodular, Firebase Hasil: <i>monitoring</i> status pintu akurat, deteksi gerakan berhasil, notifikasi <i>real-time</i> tinggi
12	Habibi & Zaky, 2025	Judul: Sistem <i>monitoring</i> Tingkat Kualitas Udara Dan Optimasi Sensor Menggunakan <i>Internet of Things</i> Berbasis Android Metode: Kalibrasi ADC, optimasi sensor, pengujian stabilitas Tools: ESP32, sensor MQ135, sensor DHT11, RESTful API PHP, aplikasi Android Kotlin Hasil: Pembacaan sensor stabil dan presisi, <i>monitoring</i> berkelanjutan berhasil, respons cepat
13	Ahdan & Susanto, 2021	Judul: Implementasi Dashboard <i>Smart Energy</i> Untuk Pengontrolan Rumah Pintar pada Perangkat Bergerak Berbasis <i>Internet of Things</i> Metode: Pengujian sistem IoT dan evaluasi performa perangkat Tools: Sensor IoT, aplikasi Android, sistem smart home Hasil: Sistem <i>monitoring</i> dan kontrol perangkat IoT berjalan dengan baik serta penggunaan CPU dan memori tetap stabil selama operasi.
14	Ihsan et al., 2023	Judul: <i>Smart Home</i> Berbasis <i>Internet of Things</i> dan <i>Mobile Application</i> pada Pustaka Galeri Mandiri Padang Metode: Metodologi eksperimental dengan pendekatan <i>prototype</i> , pengujian akses intranet dan internet

No.	Penulis (Tahun)	Keterangan
(1)	(2)	(3)
		Tools: Aplikasi Blynk, <i>NodeMCU</i> v2, sistem relay, koneksi internet Hasil: Sistem berhasil diakses dari dalam rumah melalui koneksi intranet langsung ke <i>NodeMCU</i> v2, akses dari luar rumah berhasil melalui aplikasi Android Blynk via internet, kontrol perangkat lampu dan pagar/pintu sliding berfungsi optimal, sistem mendukung akses multiple platform
15	Muslim et al., 2022	Judul: Implementasi <i>Framework</i> Flutter Pada Sistem Informasi Perpustakaan Masjid Metode: Pengujian fungsional sistem, evaluasi performa aplikasi, dan kompatibilitas cross-platform Tools: <i>Framework</i> Flutter, Laravel, database Hasil: Sistem berhasil dipublikasikan pada platform Android dan iOS serta mendukung pengelolaan data perpustakaan masjid secara digital.
16	Hendrastuty & Ihza, 2021	Judul: Rancang Bangun Aplikasi <i>monitoring</i> Santri Berbasis Android Metode: Observasi langsung, perancangan sistem dengan UML, pendekatan OOP Tools: Platform Android, UML, database Hasil: Aplikasi berhasil memungkinkan orang tua mengakses informasi perkembangan santri menggunakan NISN dan tanggal lahir
17	Dillah et al., 2024	Judul: Rancang Bangun Aplikasi <i>monitoring</i> Operasi P2TL Pengukuran Tidak Langsung 2 Phasa di PT. PLN (PERSERO) Unit Pelaksana Pelayanan Pelanggan (UP3) Metro Metode: <i>Waterfall</i> dan <i>Black-box Testing</i> Tools: Android Studio, <i>Smartphone</i> Android Hasil: Aplikasi <i>monitoring</i> berjalan stabil dan seluruh fungsi utama sistem dapat digunakan dengan baik.
18	Dewi & Fikri, 2023	Judul: Optimalisasi Keamanan Rumah dengan Implementasi Sistem Notifikasi Gerbang Cerdas Berbasis <i>Internet of Things</i> (IoT) Metode: Uji coba lapangan selama 10 hari, pengujian sinkronisasi perangkat, evaluasi efektivitas protokol MQTT Tools: <i>Smartphone</i> Android, ESP8266, MQTT, sensor magnet, buzzer, telegram Hasil: <i>monitoring</i> dan notifikasi <i>real-time</i> berhasil
19	Anggoro & Widiasari, 2021	Judul: Perancangan dan Penerapan Kendali Lampu Ruangan Berbasis IoT (<i>Internet of Things</i>) Android

No.	Penulis (Tahun)	Keterangan
(1)	(2)	(3)
		Metode: Metode <i>Prototype</i> , pengujian fungsionalitas sistem, evaluasi performa kontrol jarak jauh Tools: <i>NodeMCU</i> , Bot Telegram Hasil: Kontrol jarak jauh berhasil
20	Rahmadi et al., 2023	Judul: Rancangan Aplikasi Kuis Anggota Kelompok Penyelenggara Pemungutan Suara (KPPS) Berbasis Android Metode: Metode <i>prototype</i> dan <i>blackbox testing</i> Tools: Platform Android, <i>Smartphone</i> berbagai versi Android untuk testing kompatibilitas, tools pengembangan <i>mobile</i> Hasil: Aplikasi memiliki tingkat kompatibilitas yang baik pada berbagai perangkat Android dan seluruh fitur berjalan sesuai rancangan.
21	Yulisman et al., 2021	Judul: Penerapan <i>Internet of Things</i> (IoT) Kontrol Lampu Menggunakan <i>NodeMCU</i> ESP8266 dan <i>Smartphone</i> Metode: Metode <i>Prototype</i> , pengujian fungsional kontrol jarak jauh, evaluasi efisiensi energi Tools: <i>NodeMCU</i> ESP8266, modul Relay 5V, <i>smartphone</i> , bahasa pemrograman C, Arduino IDE. Hasil: Kontrol lampu berhasil <i>real-time</i>
22	Rosad, Yudhana, et al., 2025a	Judul: <i>Initial Study of Public Address Usage in Mosque Audio Systems</i> Metode: Deskriptif kualitatif Tools: Kuesioner dan analisis statistik deskriptif terhadap 200 masjid di Indonesia Hasil: 93,1% masjid menggunakan sistem PA konvensional, tingkat kepuasan pengguna hanya 47,2%, dan 60% pengelola masjid hanya berperan sebagai pengguna tanpa kemampuan teknis dalam pengelolaan sistem audio.

Berdasarkan analisis terhadap penelitian-penelitian terdahulu yang telah dirangkum pada Tabel 2.1, selanjutnya dilakukan identifikasi kesenjangan penelitian serta perumusan kebaruan yang ditawarkan dalam penelitian ini. Identifikasi tersebut dilakukan dengan membandingkan fokus, metode, dan hasil penelitian sebelumnya terhadap kebutuhan pengembangan sistem distribusi audio masjid berbasis IoT.

Hasil analisis tersebut disajikan pada Tabel 2.2 yang memuat ringkasan gap penelitian dan novelty penelitian. Tabel ini menunjukkan secara sistematis perbedaan dan kontribusi yang diberikan penelitian ini dibandingkan dengan penelitian sebelumnya.

Tabel 2.2 Rangkuman Gap Penelitian dan Novelty

No	Penelitian	Fokus	Gap Penelitian	Novelty
(1)	(2)	(3)	(4)	(5)
1	Rosad, Yudhana, et al., 2025b	Optimasi distribusi audio multi-zona masjid menggunakan DSP dan simulasi akustik	Belum menyediakan aplikasi <i>mobile</i> untuk <i>monitoring</i> dan kontrol sistem audio secara <i>real-time</i>	Pengembangan aplikasi Android berbasis MQTT untuk <i>monitoring</i> SPL dan kontrol audio multi-zona secara <i>real-time</i>
2	Rosad, Irkhamudin, et al., 2025	Optimasi performa endpoint IoT berbasis <i>edge</i> pada <i>Smart Mosque</i>	Fokus pada optimasi endpoint IoT dan belum menyediakan antarmuka kontrol audio bagi pengguna	Integrasi aplikasi Android dengan MQTT dan <i>backend</i> untuk <i>monitoring</i> serta kontrol sistem audio masjid
3	Ariyanto et al., 2025	Aplikasi <i>mobile monitoring</i> sensor IoT berbasis MQTT	Hanya berfokus pada <i>monitoring</i> sensor dan tidak mendukung kontrol perangkat maupun sistem audio multi-zona	Implementasi <i>monitoring</i> dan kontrol sistem audio masjid berbasis MQTT dalam aplikasi Android
4	Saiqul Umam et al., 2023	Implementasi MQTT pada smart garden	Tidak diterapkan pada sistem audio dan tidak mendukung pengelolaan multi-zona	Penerapan MQTT untuk <i>monitoring</i> SPL dan kontrol audio masjid multi-zona
5	Pratama & Irianto, 2026	Aplikasi parkir masjid berbasis Android dan Firebase	Tidak berkaitan dengan sistem audio maupun komunikasi MQTT	Pengembangan aplikasi Android untuk <i>monitoring</i> dan kontrol sistem audio masjid berbasis MQTT
6	Zainuddin Lapi et al., 2023	<i>Smart Mosque</i> untuk kontrol lampu berbasis IoT	Belum mendukung <i>monitoring</i> dan kontrol sistem distribusi audio	Implementasi aplikasi <i>mobile</i> untuk <i>monitoring</i> dan kontrol sistem audio masjid berbasis IoT
7	Thantawi et al., 2023	Pemanfaatan aplikasi <i>Smart Mosque</i>	Tidak mengembangkan sistem kontrol audio maupun integrasi MQTT <i>real-time</i>	Pengembangan aplikasi Android yang terintegrasi dengan sistem audio masjid berbasis MQTT

No	Penelitian	Fokus	Gap Penelitian	Novelty
(1)	(2)	(3)	(4)	(5)
8	Sanjaya et al., 2021	<i>monitoring</i> protokol kesehatan masjid berbasis IoT	Tidak membahas distribusi audio dan pengelolaan zona audio	Implementasi <i>monitoring</i> dan kontrol sistem audio multi-zona berbasis Android
9	Abdulfathah & Santoso, 2024	<i>monitoring</i> asap dan kontrol kamera berbasis IoT	Tidak diterapkan pada sistem distribusi audio	Implementasi <i>monitoring</i> SPL dan kontrol audio berbasis MQTT
10	Saputra et al., 2025	<i>monitoring</i> kualitas air menggunakan MQTT	Berfokus pada <i>monitoring</i> sensor dan tidak menyediakan aplikasi kontrol audio	Integrasi MQTT dengan aplikasi Android untuk <i>monitoring</i> dan kontrol audio masjid
11	Sosmita & Salam, 2024	<i>monitoring</i> dan kontrol pintu berbasis IoT	Tidak mendukung pengelolaan audio multi-zona	Implementasi <i>monitoring</i> dan kontrol sistem audio masjid multi-zona berbasis MQTT
12	Habibi & Zaky, 2025	<i>monitoring</i> kualitas udara berbasis Android dan IoT	Fokus pada <i>monitoring</i> sensor, bukan kontrol sistem audio	Pengembangan aplikasi Android untuk <i>monitoring</i> SPL dan kontrol audio secara <i>real-time</i> audio multi-zona
13	Ahdan & Susanto, 2021	Dashboard <i>smart home</i> berbasis IoT	Tidak diterapkan pada sistem audio masjid dan tidak mendukung multi-zona	Pengembangan dashboard Android untuk <i>monitoring</i> dan kontrol audio masjid berbasis MQTT
14	Ihsan et al., 2023	<i>Smart home</i> berbasis IoT dan <i>mobile application</i>	Tidak mendukung distribusi audio dan pengelolaan zona audio	Implementasi aplikasi Android untuk <i>monitoring</i> dan kontrol sistem audio multi-zona
15	Muslim et al., 2022	Sistem informasi perpustakaan masjid berbasis Flutter	Tidak membahas komunikasi MQTT maupun sistem audio	Pemanfaatan Flutter untuk aplikasi <i>monitoring</i> dan kontrol sistem audio masjid

No	Penelitian	Fokus	Gap Penelitian	Novelty
(1)	(2)	(3)	(4)	(5)
16	Hendrastuty & Ihza, 2021	Aplikasi <i>monitoring</i> berbasis Android	Tidak mengintegrasikan MQTT dan perangkat IoT	Integrasi Android, MQTT, dan <i>backend</i> untuk <i>monitoring</i> sistem audio masjid
17	Dillah et al., 2024	Aplikasi <i>monitoring</i> berbasis Android	Tidak menggunakan MQTT dan tidak mendukung kontrol perangkat audio	Implementasi komunikasi MQTT <i>real-time</i> pada aplikasi <i>monitoring</i> dan kontrol audio
18	Dewi & Fikri, 2023	Sistem notifikasi berbasis MQTT	Tidak diterapkan pada sistem distribusi audio multi-zona	Pemanfaatan MQTT untuk <i>monitoring</i> dan kontrol sistem audio masjid
19	Anggoro & Widiyastuti, 2021	Kontrol lampu berbasis IoT Android	Tidak mendukung <i>monitoring</i> SPL dan pengelolaan multi-zona	I Implementasi <i>monitoring</i> SPL dan kontrol audio multi-zona berbasis Android
20	Rahmadi et al., 2023	Aplikasi Android berbasis <i>prototype</i>	Tidak terintegrasi dengan IoT dan MQTT	Pengembangan aplikasi Android berbasis Flutter yang terintegrasi dengan MQTT untuk sistem audio masjid
21	Yulisman et al., 2021	Kontrol lampu berbasis IoT	Tidak mendukung distribusi audio multi-zona	Implementasi <i>monitoring</i> dan kontrol audio masjid berbasis MQTT
22	Rosad, Yudhana, et al., 2025a	Studi penggunaan sistem PA masjid	Tidak mengembangkan sistem kontrol digital atau aplikasi <i>mobile</i>	Pengembangan sistem distribusi audio digital berbasis MQTT dan Android

Berdasarkan analisis gap penelitian pada Tabel 2.2, dapat disimpulkan bahwa meskipun berbagai penelitian telah membahas penerapan IoT, aplikasi Android, dan protokol MQTT untuk sistem *monitoring* dan kontrol, belum terdapat penelitian yang mengintegrasikan sistem distribusi audio masjid berbasis multi-zona dengan komunikasi *real-time* melalui aplikasi *mobile*. Sebagian besar penelitian masih

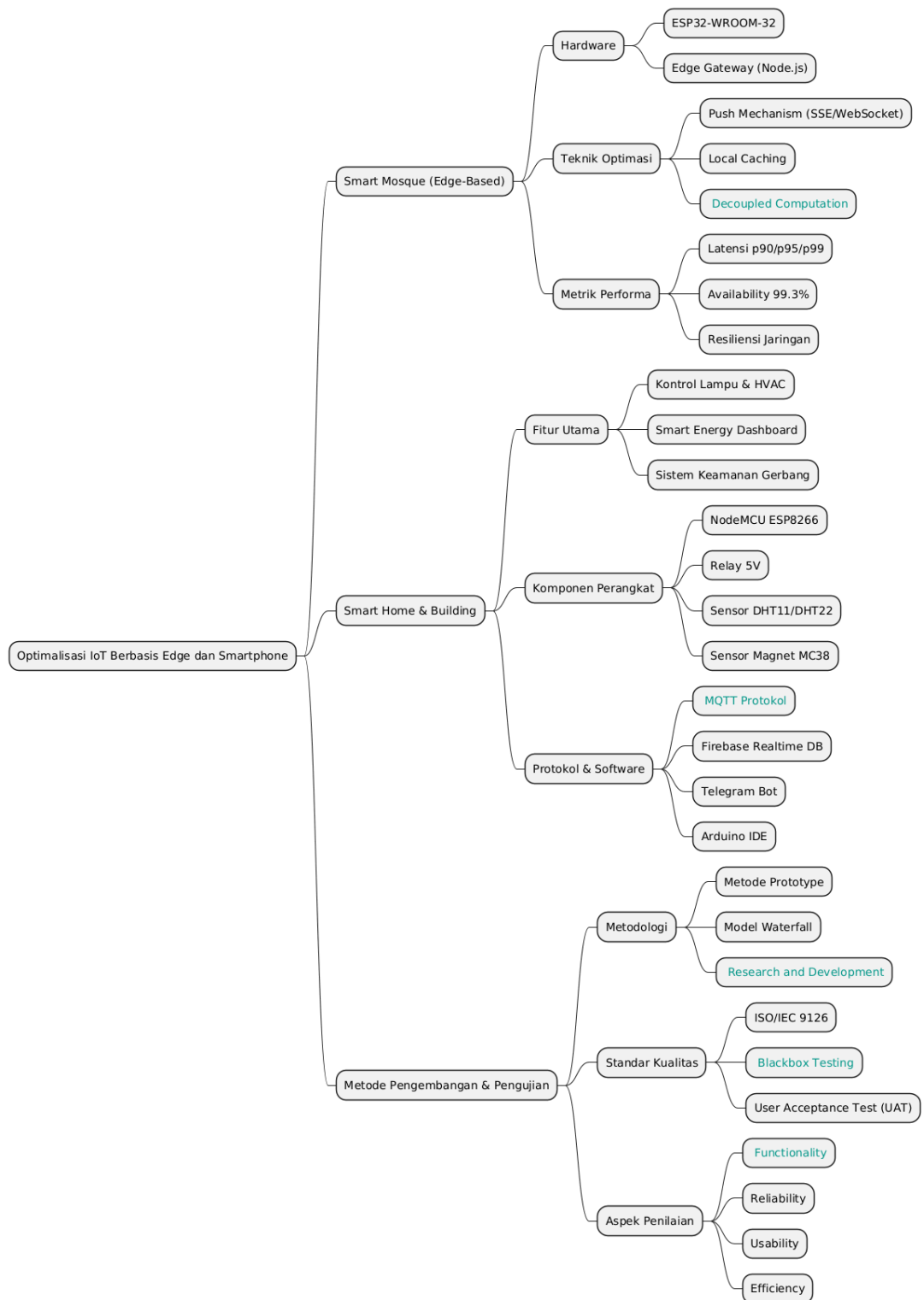
berfokus pada *monitoring* atau kontrol perangkat secara terpisah tanpa menggabungkan aspek distribusi audio dan arsitektur terdistribusi.

Oleh karena itu, penelitian ini menawarkan kebaruan berupa pengembangan sistem distribusi audio masjid berbasis IoT yang mengintegrasikan konsep *Cyber-Physical System* (CPS), komunikasi MQTT berbasis *publish-subscribe*, serta aplikasi Android sebagai antarmuka *monitoring* dan kontrol secara *real-time*. Integrasi ini dirancang untuk mendukung pengelolaan audio multi-zona yang lebih fleksibel, responsif, dan mudah digunakan oleh pengguna, sehingga memberikan kontribusi terhadap pengembangan sistem audio masjid yang lebih adaptif.

Berdasarkan gap dan kebaruan tersebut, penelitian ini tidak hanya mengadopsi teknologi IoT dan MQTT, tetapi juga mengintegrasikannya secara komprehensif dalam sistem distribusi audio multi-zona berbasis aplikasi *mobile*. Untuk memperjelas kerangka konseptual penelitian, berikut disajikan *mind map* penelitian dan Taksonomi Bloom.



Gambar 2.1 *Mind Map* Penelitian



Gambar 2.2 Taksonomi Bloom

B. Landasan Teori

Landasan teori yang digunakan pada penelitian “Aplikasi *Mobile* Android Berbasis MQTT Untuk *Monitoring* dan Kontrol Sistem Audio Multi-Zona Masjid” sebagai berikut :

1. Sistem Audio Masjid

Sistem audio masjid memiliki peran penting dalam mendukung komunikasi selama kegiatan ibadah, seperti adzan, khotbah, dan ceramah. Sistem audio ini bertujuan agar suara dapat didengar dengan jelas oleh jamaah di ruang utama maupun area lain seperti serambi atau luar masjid. Dalam praktiknya, tantangan utama meliputi geometri ruang, penempatan speaker, *reverberasi*, dan pemerataan distribusi suara[2], [9]. Perkembangan teknologi digital juga memungkinkan sistem audio diintegrasikan dengan mekanisme *monitoring* sehingga kondisi perangkat dapat dipantau secara *real-time* untuk mendukung keandalan operasional sistem[11].

Sistem audio masjid umumnya melibatkan rantai perangkat *input–proses–output* (misalnya mikrofon, penguat, dan speaker) yang perlu dikalibrasi sesuai karakter ruang[2], [12]. Pendekatan PA terpusat yang masih dominan sering menimbulkan ketidakmerataan cakupan suara pada ruang dengan kondisi akustik kompleks. Oleh karena itu, sistem distribusi audio multi-zona menjadi pendekatan yang lebih tepat untuk meningkatkan *intelligibility* dan kenyamanan dengar[1], [2].

Penelitian oleh Rosad, Yudhana, et al. (2025) menunjukkan bahwa penerapan sistem distribusi audio multi-zona pada masjid mampu meningkatkan pemerataan suara dan nilai *Speech Transmission Index* (STI) sehingga informasi yang disampaikan dapat diterima jamaah dengan lebih jelas[2]. Hasil tersebut menunjukkan bahwa pengelolaan audio berdasarkan zona merupakan pendekatan yang efektif untuk mengatasi perbedaan karakteristik akustik pada setiap area masjid. Seiring bertambahnya jumlah zona yang dikelola, diperlukan mekanisme *monitoring* data serta kontrol yang mampu menampilkan kondisi setiap zona serta memudahkan pengelolaan logika sistem secara *real-time*. Oleh karena itu, pemanfaatan aplikasi *mobile* yang terintegrasi dengan infrastruktur *backend* IoT menjadi salah satu solusi untuk mendukung proses *monitoring* dan simulasi manajemen data sistem audio multi-zona secara lebih efektif, fleksibel, dan *real-time*.

2. Cyber-Physical Systems (CPS)

CPS merupakan integrasi komputasi, komunikasi jaringan, dan proses fisik untuk *monitoring* serta kontrol *real-time*[1], [2]. Pada konteks audio masjid, prinsip CPS relevan untuk pengendalian terdistribusi antarzona

sehingga keputusan kontrol tidak bertumpu pada satu titik pusat[1], [2]. Implementasi CPS juga didukung oleh sistem *monitoring* berbasis IoT yang mampu mengintegrasikan sensor dan perangkat kontrol sehingga respons sistem terhadap perubahan kondisi menjadi lebih cepat dan terkoordinasi[13].

Implementasi berbasis *edge* dan arsitektur terdistribusi juga meningkatkan respons sistem saat kondisi jaringan berubah, termasuk ketika koneksi internet tidak stabil[1]. Dengan demikian, CPS menjadi fondasi teknis bagi transisi dari model terpusat menuju model multi-zona yang lebih adaptif[1], [2].

Konsep CPS tidak hanya menghubungkan dunia fisik dan digital, tetapi juga memungkinkan terjadinya pertukaran data secara kontinu antara perangkat lapangan dan sistem komputasi. Dalam arsitektur *Smart Mosque*, data dari perangkat audio dapat dikirim menuju *backend system* untuk diproses dan selanjutnya digunakan sebagai dasar *monitoring* kondisi sistem maupun pengelolaan logika parameter perangkat. Selain itu, penerapan CPS berbasis arsitektur terdistribusi memungkinkan setiap *node* tetap beroperasi dan mempertahankan komunikasi meskipun terjadi gangguan pada *node* lain, sehingga meningkatkan keandalan sistem dibandingkan pendekatan terpusat. Pendekatan ini mendukung pengelolaan sistem audio multi-zona yang lebih responsif, terukur, dan terintegrasi.

3. *Internet of Things* (IoT) dalam Sistem Audio

IoT adalah konsep di mana perangkat fisik dilengkapi dengan sensor, aktuator, dan kemampuan komunikasi sehingga dapat berinteraksi dengan sistem lainnya melalui internet. IoT memungkinkan perangkat fisik terhubung dan bertukar data untuk *monitoring* serta kontrol jarak jauh secara *real-time*[4]. Dalam berbagai implementasi, pendekatan ini digunakan untuk kendali perangkat, sinkronisasi data aplikasi *mobile*, dan pemantauan status sistem[7], [14]. Penelitian menunjukkan bahwa sistem *monitoring* berbasis IoT mampu menghasilkan pembacaan sensor yang stabil dan presisi serta mendukung pemantauan kondisi sistem secara berkelanjutan melalui aplikasi *mobile*[15].

Dalam sistem audio masjid, konsep IoT dapat diterapkan untuk memantau status perangkat audio per zona dan mengirim perintah kontrol melalui aplikasi *mobile* pengguna[1], [16]. Pendekatan ini mendukung pengelolaan operasional yang lebih cepat, terukur, dan fleksibel[2], [8]. Selain itu, penerapan IoT pada sistem perangkat terhubung menunjukkan bahwa visualisasi data dapat dipantau secara jarak jauh melalui aplikasi *mobile* dengan koneksi jaringan yang stabil dan mendukung pengelolaan multi-*node* secara terintegrasi[17].

Berbagai penelitian menunjukkan bahwa teknologi IoT telah banyak digunakan untuk mendukung sistem *monitoring* dan transmisi data berbasis aplikasi *mobile*. Dengan memanfaatkan komunikasi data secara *real-time*,

sistem dapat memperoleh informasi kondisi perangkat serta melakukan pengendalian logika dari lokasi yang berbeda. Kemampuan ini menjadikan IoT sebagai fondasi utama dalam pengembangan sistem *monitoring* dan kontrol audio masjid yang terintegrasi dengan aplikasi Android.

4. MQTT sebagai Protokol Komunikasi

MQTT (Message Queuing Telemetry Transport) merupakan protokol ringan berbasis *publish-subscribe* yang banyak dipakai pada implementasi IoT untuk telemetri dan kontrol[7], [8]. Dalam studi implementatif, MQTT mendukung pengiriman data sensor secara stabil serta kendali perangkat jarak jauh dengan beban komunikasi yang efisien[5], [8]. Pada berbagai sistem kendali perangkat berbasis IoT, komunikasi antara perangkat dan aplikasi *mobile* terbukti mampu berjalan dengan baik melalui jaringan internet sehingga memungkinkan pengendalian perangkat secara *real-time* dengan respons yang stabil[18].

Untuk sistem distribusi audio masjid, MQTT sesuai untuk kanal perintah (misalnya kontrol volume/*mute* per zona) dan publikasi status perangkat secara *real-time*. Karakteristik ini mendukung kebutuhan latensi rendah dan kontinuitas layanan pada sistem audio berbasis IoT[1], [8]. Keunggulan MQTT terletak pada penggunaan bandwidth yang efisien, dukungan untuk komunikasi *real-time*, dan kemampuan untuk mengirimkan pesan meskipun terjadi gangguan atau pemutusan sementara pada jaringan.

MQTT banyak digunakan pada sistem IoT karena menerapkan mekanisme *publish-subscribe* yang memungkinkan pengiriman data berlangsung secara ringan dan efisien. Karakteristik tersebut menjadikan MQTT cocok digunakan pada aplikasi *monitoring* dan kontrol yang membutuhkan pembaruan data secara *real-time* dengan latensi rendah. Pada penelitian ini, MQTT digunakan sebagai media pertukaran data antara aplikasi Android dan *backend system* sehingga perubahan status kontrol maupun data *monitoring* dapat diterima secara cepat dan sinkron.

5. Pemrograman Dart dan *Framework* Flutter

Dart merupakan bahasa pemrograman yang dikembangkan untuk membangun aplikasi modern berbasis client, khususnya dalam pengembangan aplikasi *mobile*. Dart digunakan secara luas dalam *framework* Flutter yang menyediakan pustaka pengembangan aplikasi Android dengan efisiensi kompilasi kode yang optimal[19].

Flutter merupakan *framework* yang mendukung pengembangan antarmuka pengguna berbasis komponen-komponen visual (*widget*), sehingga memudahkan pengembang dalam membangun aplikasi yang responsif dan

konsisten. Pengembangan aplikasi menggunakan Flutter memberikan performa rendering UI yang tinggi karena menggunakan mesin grafis mandiri, sehingga transisi antarmuka berjalan mulus.

Selain itu, Flutter juga telah digunakan dalam pengembangan aplikasi *monitoring* berbasis IoT yang terintegrasi dengan protokol MQTT. Hasil penelitian menunjukkan bahwa aplikasi *mobile* berbasis Flutter mampu menampilkan data secara *real-time* dengan manajemen *state* antarmuka yang stabil[5]. Hal ini menunjukkan bahwa kombinasi Dart dan Flutter sangat sesuai untuk pengembangan aplikasi *mobile* berbasis IoT yang membutuhkan respons data cepat dan antarmuka yang interaktif. Dalam konteks penelitian ini, penggunaan Dart dan Flutter mendukung pengembangan aplikasi Android yang mampu melakukan *monitoring* dan kontrol sistem audio masjid secara *real-time* dengan tampilan yang intuitif dan performa yang optimal.

6. *Model-View-ViewModel (MVVM)*

Model-View-ViewModel (MVVM) merupakan pola arsitektur perangkat lunak yang memisahkan aplikasi ke dalam tiga komponen utama, yaitu Model, View, dan ViewModel. Model bertanggung jawab terhadap pengelolaan data dan logika bisnis, View berfungsi menampilkan antarmuka pengguna, sedangkan ViewModel menjadi penghubung antara Model dan View. Melalui pemisahan tersebut, perubahan data dapat dikelola secara lebih terstruktur tanpa menyebabkan ketergantungan langsung antara antarmuka pengguna dan logika aplikasi. Penerapan MVVM juga memungkinkan proses pengembangan, pengujian, dan pemeliharaan aplikasi dilakukan dengan lebih mudah karena setiap komponen memiliki tanggung jawab yang jelas[20].

Penelitian Riyadhhi, Purnamasari, dan Prihandani (2023) menunjukkan bahwa penerapan MVVM pada aplikasi Android mampu memisahkan proses bisnis dengan antarmuka pengguna sehingga struktur kode menjadi lebih rapi, mudah dikembangkan, dan lebih mudah dipelihara. Selain itu, MVVM memanfaatkan mekanisme *state management* yang membantu sinkronisasi data antara View dan ViewModel secara lebih efektif[20].

Pada penelitian ini, arsitektur MVVM digunakan untuk memisahkan komponen antarmuka pengguna, logika aplikasi, dan pengelolaan data MQTT sehingga proses *monitoring* status *node*, *monitoring* nilai *Sound Pressure Level* (SPL), serta kontrol volume dan *mute* dapat dikelola secara terstruktur dan mudah dikembangkan.

7. *Black-box Testing*

Black-box Testing merupakan metode pengujian perangkat lunak yang dilakukan dengan mengevaluasi fungsionalitas sistem berdasarkan masukan

(*input*) dan keluaran (*output*) tanpa memperhatikan struktur atau kode program. Tujuan pengujian ini adalah untuk memastikan bahwa setiap fitur aplikasi telah berjalan sesuai dengan spesifikasi kebutuhan yang telah ditentukan serta mendeteksi adanya kesalahan pada fungsi, antarmuka, maupun proses pengolahan data[21].

Pada pengembangan aplikasi Android, *Black-box Testing* banyak digunakan untuk memverifikasi fungsi-fungsi utama aplikasi seperti proses autentikasi, navigasi antarmuka, pengolahan data, hingga komunikasi dengan *backend system*. Pengujian dilakukan dengan memberikan berbagai skenario masukan pada setiap fitur, kemudian membandingkan keluaran yang dihasilkan dengan hasil yang diharapkan sehingga dapat diketahui apakah fungsi aplikasi telah berjalan dengan benar[22].

Penelitian oleh Dillah et al. (2024) menerapkan *Black-box Testing* untuk menguji aplikasi *monitoring* berbasis Android yang dikembangkan. Hasil pengujian menunjukkan bahwa seluruh fungsi aplikasi dapat berjalan sesuai rancangan tanpa ditemukan kesalahan yang mengganggu operasional sistem. Demikian pula penelitian Rahmadi et al. (2023) menggunakan *Black-box Testing* untuk mengevaluasi aplikasi Android berbasis *prototype*. Hasil pengujian menunjukkan bahwa seluruh fungsi aplikasi berjalan dengan baik dan memenuhi kebutuhan fungsional yang telah ditetapkan.

Pada penelitian ini, *Black-box Testing* digunakan untuk memverifikasi seluruh fitur utama aplikasi Android, meliputi proses *Login*, koneksi MQTT, *monitoring* status *node* dan nilai *Sound Pressure Level* (SPL), kontrol volume dan *mute*, notifikasi kondisi perangkat, mekanisme *auto reconnect*, serta sinkronisasi data setelah koneksi kembali normal. Pengujian dilakukan dengan membandingkan hasil yang diperoleh terhadap hasil yang diharapkan sehingga dapat dipastikan bahwa seluruh fungsi aplikasi bekerja sesuai kebutuhan sistem.

8. *Response time* MQTT

Response time merupakan parameter performa yang menunjukkan waktu yang dibutuhkan sistem untuk memberikan respons terhadap suatu permintaan atau instruksi yang diberikan pengguna. Pada sistem *monitoring* dan kontrol berbasis IoT, *response time* digunakan untuk mengevaluasi kecepatan komunikasi data dan tingkat responsivitas sistem dalam mendukung layanan *real-time*[23].

Pengukuran *response time* umumnya dilakukan dengan menghitung selisih waktu (*timestamp*) antara saat pesan dikirim (*publish*) dan saat pesan diterima pada sistem tujuan. Penelitian Dizdarevic, Michalke, dan Jukan (2023) menggunakan pendekatan tersebut untuk melakukan *benchmarking*

performa berbagai implementasi MQTT broker. Hasil penelitian menunjukkan bahwa nilai *response time* dipengaruhi oleh kapasitas broker, kondisi jaringan, ukuran *payload*, dan jumlah koneksi yang dilayani secara bersamaan[23].

Penelitian oleh Rosad, Irkhamudin, et al. (2025) menunjukkan bahwa optimasi endpoint IoT berbasis *edge* pada lingkungan *Smart Mosque* mampu menurunkan latensi komunikasi dari 180 ms menjadi 50 ms, sedangkan Saputra et al. (2025) menunjukkan bahwa komunikasi berbasis MQTT mampu mempertahankan latensi yang rendah dan stabil pada sistem *monitoring* IoT[1], [10]. Hasil kedua penelitian tersebut menunjukkan bahwa *response time* merupakan salah satu indikator penting dalam mengevaluasi performa sistem *monitoring* dan kontrol berbasis IoT.

Pada penelitian ini, pengukuran *response time* mengacu pada metode yang digunakan oleh Dizdarevic et al. (2023), yaitu menghitung selisih *timestamp* antara saat instruksi kontrol dikirim dari aplikasi Android dan saat pesan diterima oleh *backend system* melalui MQTT broker. Untuk memperoleh nilai tersebut, penelitian ini menambahkan fitur pencatatan *timestamp* pada aplikasi sehingga pengukuran dapat dilakukan secara langsung terhadap proses komunikasi yang berlangsung pada sistem.

9. Metode *Research and Development* (R&D)

Metode *Research and Development* (R&D) merupakan pendekatan penelitian yang berorientasi pada proses menghasilkan suatu produk sekaligus mengevaluasi produk tersebut agar sesuai dengan kebutuhan pengguna dan tujuan penelitian. Produk yang dihasilkan dalam penelitian R&D tidak terbatas pada perangkat pembelajaran, tetapi juga dapat berupa perangkat lunak, sistem informasi, aplikasi *mobile*, maupun produk teknologi lainnya. Oleh karena itu, karakteristik utama metode R&D meliputi tahapan analisis kebutuhan, perancangan, pengembangan, implementasi, dan evaluasi terhadap produk yang dikembangkan sehingga produk yang dihasilkan memiliki tingkat kelayakan sesuai dengan tujuan penelitian[24].

Perkembangan penelitian terkini menunjukkan bahwa metode R&D tidak hanya digunakan pada bidang pendidikan, tetapi juga banyak diterapkan dalam pengembangan sistem dan perangkat lunak karena mampu mengintegrasikan proses penelitian dengan proses pengembangan produk secara sistematis. Dalam pengembangan perangkat lunak, tahapan penelitian R&D sering mengadopsi pendekatan *Software Development Life Cycle* (SDLC) sebagai acuan teknis dalam proses analisis kebutuhan, perancangan, implementasi, hingga pengujian sistem. Namun demikian, SDLC berfungsi sebagai pendekatan teknis dalam pengembangan perangkat lunak, sedangkan R&D

merupakan metode penelitian yang berorientasi pada pengembangan, validasi, dan evaluasi produk yang dihasilkan[25].

Berdasarkan karakteristik tersebut, metode R&D dipilih pada penelitian ini karena tujuan penelitian tidak hanya menganalisis permasalahan pada sistem audio multi-zona masjid, tetapi juga menghasilkan produk berupa aplikasi *mobile* Android berbasis MQTT yang mampu mendukung proses *monitoring* dan kontrol sistem. Selain menghasilkan produk, penelitian ini juga melakukan implementasi serta evaluasi terhadap aplikasi melalui pengujian fungsionalitas menggunakan *Black-box Testing* dan pengukuran performa komunikasi menggunakan parameter *response time* MQTT. Dengan demikian, penggunaan metode R&D dinilai sesuai dengan tujuan penelitian yang berorientasi pada pengembangan dan evaluasi produk, sedangkan tahapan pengembangan perangkat lunak yang diterapkan merupakan pendekatan teknis untuk merealisasikan aplikasi yang dikembangkan[25].

C. Studi Terkait

Berbagai penelitian sebelumnya telah membahas pengembangan sistem audio masjid berbasis teknologi modern. Penelitian oleh Al-Oudat menunjukkan bahwa penerapan *Internet of Things* (IoT) pada sistem kontrol level suara dapat membantu mengatasi permasalahan distribusi suara yang tidak merata pada pengeras suara masjid melalui mekanisme pengaturan volume yang lebih adaptif dan terukur[26]. Selain itu, penelitian oleh Rosad, Yudhana, dan Sutikno mengungkapkan bahwa sistem audio terpusat masih rentan menimbulkan *dead zone*, *reverberasi*, dan penurunan *speech intelligibility*, sehingga pendekatan distribusi audio berbasis zonasi direkomendasikan untuk meningkatkan kualitas distribusi suara di lingkungan masjid[2].

Penelitian lain oleh Supriyadi et al. menunjukkan bahwa penerapan zonasi speaker dan pengelolaan distribusi sinyal audio yang tepat mampu menghasilkan kualitas suara yang lebih merata dan jelas pada berbagai area masjid[27]. Namun, sebagian besar penelitian tersebut masih berfokus pada aspek akustik dan perangkat keras sistem audio, serta belum mengintegrasikan aplikasi *mobile* untuk *monitoring* dan kontrol secara *real-time*. Oleh karena itu, penelitian ini mengembangkan aplikasi Android yang terintegrasi dengan sistem audio berbasis IoT dan komunikasi MQTT untuk mendukung *monitoring* serta kontrol audio multi-zona secara *real-time*.

BAB III METODOLOGI PENELITIAN

A. Waktu dan Tempat Penelitian

Waktu dan tempat penelitian merupakan aspek penting yang menjelaskan periode pelaksanaan serta lokasi yang digunakan dalam proses perancangan, pengembangan, dan pengujian sistem. Penentuan waktu dan lokasi penelitian dilakukan untuk memastikan seluruh tahapan penelitian dapat berjalan secara terstruktur serta sesuai dengan kebutuhan pengembangan aplikasi kontrol dan *monitoring* sistem audio masjid berbasis IoT. Adapun waktu dan tempat pelaksanaan penelitian ini dijelaskan sebagai berikut.

1. Waktu Penelitian

Penelitian ini dilakukan pada bulan Desember 2025 hingga Juni 2026. Proses pengumpulan data dimulai pada bulan Desember 2025 melalui observasi dan diskusi dengan pengelola masjid untuk mengidentifikasi kebutuhan sistem audio yang ada. Setelah itu, perancangan aplikasi dan sistem dilakukan pada bulan Januari hingga Februari 2026, dilanjutkan dengan pengujian fungsional, pengujian latensi komunikasi MQTT, serta pengujian performa aplikasi pada bulan Maret hingga Mei 2026. Selanjutnya, proses analisis hasil pengujian, penyusunan laporan penelitian, dan penyelesaian skripsi dilakukan pada bulan Juni 2026.

2. Tempat Penelitian

Penelitian ini dilaksanakan pada tiga lokasi yang memiliki peran berbeda sesuai dengan tahapan penelitian yang dilakukan, yaitu tahap pengumpulan kebutuhan sistem, pengembangan aplikasi, dan pengujian sistem.

- a. Lokasi pertama adalah Shatomeia Yogyakarta yang digunakan sebagai tempat perancangan dan pengembangan sistem. Pada lokasi ini dilakukan berbagai kegiatan, meliputi analisis kebutuhan sistem, perancangan arsitektur aplikasi berbasis MQTT, perancangan antarmuka pengguna (UI/UX), implementasi aplikasi Android menggunakan *framework* Flutter, serta integrasi aplikasi dengan MQTT broker dan basis data. Selain itu, dilakukan pula proses pengembangan dan penyempurnaan fitur aplikasi sesuai dengan kebutuhan sistem yang telah diidentifikasi.
- b. Lokasi kedua adalah Masjid Al-Muhajirin yang beralamat di Jl. Wates KM.11 Perum Griya Kencana Permai Sedayu, Bantul, Yogyakarta. Lokasi ini digunakan sebagai tempat pengumpulan data awal melalui observasi dan diskusi dengan pengelola masjid. Kegiatan tersebut bertujuan untuk mengidentifikasi permasalahan yang dihadapi dalam pengelolaan sistem audio masjid serta memperoleh informasi

mengenai kebutuhan fungsional dan nonfungsional yang menjadi dasar dalam perancangan aplikasi kontrol dan *monitoring* berbasis *smartphone*.

- c. Lokasi ketiga adalah Shatomeia Kesugihan, Kabupaten Cilacap, yang digunakan sebagai tempat pelaksanaan pengujian sistem. Pada lokasi ini dilakukan pengujian fungsional menggunakan metode *Black-box Testing* untuk memastikan setiap fitur aplikasi berjalan sesuai dengan kebutuhan yang telah dirancang. Selain itu, dilakukan pengujian latensi komunikasi data berbasis MQTT untuk mengukur waktu respons pengiriman dan penerimaan data, serta pengujian performa aplikasi Android untuk mengevaluasi penggunaan sumber daya perangkat, stabilitas aplikasi, dan kinerja sistem secara keseluruhan. Pemilihan lokasi ini didasarkan pada ketersediaan fasilitas pengujian yang memadai sehingga proses pengujian dapat dilakukan secara terkontrol dan menghasilkan data yang akurat.

B. Alat dan Bahan

Alat dan bahan yang digunakan dalam penelitian ini terdiri dari perangkat keras dan perangkat lunak yang digunakan untuk mendukung pengembangan dan pengujian aplikasi. Berikut adalah daftar alat dan bahan yang digunakan dalam penelitian ini:

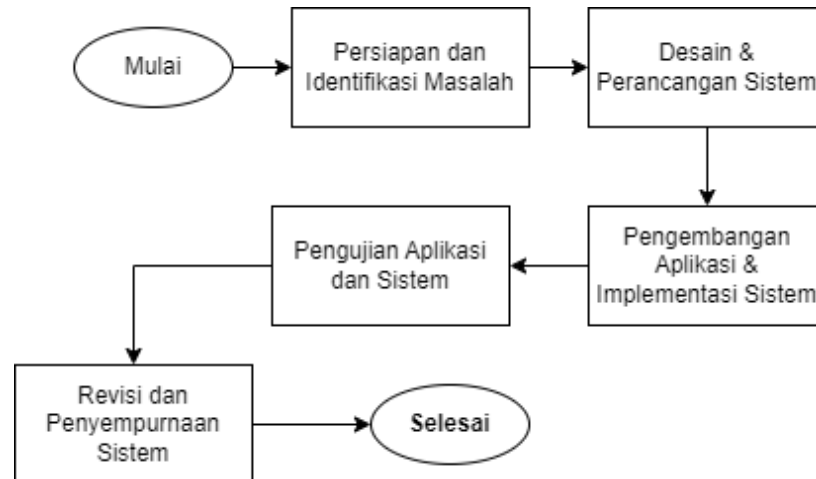
Tabel 3.1 Alat dan Bahan yang digunakan dalam penelitian

Kategori	Alat dan Bahan	Keterangan
Perangkat Keras	<i>Smartphone</i> Android	Digunakan oleh pengguna untuk menjalankan aplikasi <i>mobile monitoring</i> dan kontrol.
	Mikrokontroler IoT	Menggunakan modul ESP32 (atau <i>NodeMCU</i> ESP8266 dalam beberapa literatur terkait) sebagai <i>endpoint</i> pengirim data sensor.
	Sensor Suara (SPL)	Komponen <i>node</i> sensor untuk menangkap dan mengukur tingkat intensitas suara (<i>Sound Pressure Level</i>) di setiap zona.
	Speaker Aktif	Menggunakan speaker dengan spesifikasi ACR KSV 4 Inch 8 Ω / 20 Watt.
	Power Amplifier	Menggunakan modul penguat daya tipe PAM8403 2 x 3 Watt
Perangkat Lunak	Android Studio	Digunakan untuk mengembangkan aplikasi <i>mobile</i> berbasis Android.

Kategori	Alat dan Bahan	Keterangan
	<i>Framework</i> Flutter	Kerangka kerja utama lintas platform yang digunakan untuk merancang dan membangun antarmuka aplikasi Android.
	Bahasa Pemrograman Dart	Bahasa pemrograman utama yang digunakan untuk menulis seluruh logika aplikasi pada <i>framework</i> Flutter.
	Arsitektur Perangkat Lunak	Menerapkan pola arsitektur MVVM (<i>Model-View-ViewModel</i>) untuk memisahkan logika bisnis aplikasi dengan lapisan tampilan.
	Library mqtt_client	<i>Package</i> atau pustaka perangkat lunak di Flutter yang digunakan untuk mengimplementasikan fungsi komunikasi protokol MQTT.
Komunikasi dan Jaringan	Wi-Fi / LAN	Digunakan untuk menghubungkan perangkat <i>mobile</i> dengan MQTT broker dan perangkat audio.
	Protokol	Protokol komunikasi data ringan berbasis <i>publish-subscribe</i> yang menjadi standar pengiriman instruksi kontrol dan data <i>monitoring</i> secara <i>real-time</i> .
Server dan Database	MQTT broker (Mosquitto)	Untuk komunikasi antara aplikasi <i>mobile</i> dan perangkat audio masjid.
	Firebase Database	Menyimpan data terkait status perangkat dan pengaturan sistem audio masjid.

C. Prosedur Penelitian

Prosedur penelitian ini dilakukan melalui tahapan-tahapan yang terstruktur dan sistematis untuk mencapai tujuan penelitian, yaitu pengembangan aplikasi *mobile* Android untuk *monitoring* dan kontrol *real-time* sistem distribusi audio masjid berbasis MQTT. Berikut adalah diagram alur penelitian yang menggambarkan tahapan-tahapan yang dilakukan dalam penelitian ini, mulai dari persiapan hingga revisi dan penyempurnaan sistem:



Gambar 3.1 Diagram Alur Penelitian

Tahapan-tahapan lebih rinci dijelaskan pada setiap langkah berikut:

1. Persiapan dan Identifikasi Masalah

Tahap pertama dalam penelitian ini adalah Persiapan dan Identifikasi Masalah, yang sangat penting karena akan membentuk dasar dari seluruh proses penelitian. Pada tahap ini, dilakukan persiapan teknis dan non-teknis yang dibutuhkan serta identifikasi masalah yang akan menjadi fokus dari penelitian. Penjelasan lebih rinci mengenai tahapan ini adalah sebagai berikut:

a. Persiapan Awal

Pada tahap ini, peneliti melakukan persiapan untuk memulai penelitian. Proses persiapan ini melibatkan beberapa aspek, antara lain:

- 1) Penyusunan Rencana Penelitian: Peneliti menyusun rencana penelitian yang mencakup tujuan, metodologi, dan jadwal penelitian. Hal ini bertujuan untuk memastikan bahwa penelitian dilakukan dengan sistematis dan terstruktur.
- 2) Pemilihan Alat dan Bahan: Peneliti mempersiapkan perangkat keras dan perangkat lunak yang akan digunakan dalam penelitian ini. Beberapa alat yang disiapkan antara lain:
 - a) Perangkat Keras: *Smartphone* Android untuk menjalankan aplikasi, perangkat audio masjid (speaker, amplifier), dan MQTT broker (*Mosquitto*) yang digunakan untuk komunikasi antar perangkat.
 - b) Perangkat Lunak: Flutter untuk pengembangan aplikasi *mobile* Android, Dart sebagai bahasa pemrograman, serta `mqtt_client` untuk implementasi protokol MQTT.
- 3) Penyusunan Daftar Pustaka: Peneliti mengumpulkan referensi terkait dengan sistem audio masjid, teknologi IoT, MQTT, serta

desain aplikasi *mobile*, yang akan menjadi dasar untuk pengembangan aplikasi.

b. Identifikasi Masalah

Setelah persiapan teknis selesai, tahap berikutnya adalah Identifikasi Masalah, yang merupakan langkah penting dalam penelitian ini. Pada tahap ini, peneliti melakukan observasi dan diskusi dengan pengelola masjid untuk mengidentifikasi permasalahan yang dihadapi dalam pengelolaan sistem audio serta menentukan kebutuhan sistem yang akan dikembangkan. Beberapa langkah dalam identifikasi masalah ini adalah sebagai berikut:

1) Diskusi dengan Pengelola Masjid

Peneliti melakukan diskusi dengan pengelola masjid untuk memperoleh informasi mengenai kondisi sistem audio yang digunakan serta kebutuhan yang diperlukan dalam pengelolaan audio multi-zona. Beberapa aspek yang dibahas antara lain:

- a) Apa saja kendala yang dihadapi dalam mengatur distribusi suara pada berbagai zona masjid?
- b) Apakah sistem audio yang digunakan saat ini telah mendukung pengaturan volume dan status audio secara terpisah pada setiap zona?
- c) Bagaimana proses pemantauan dan pengendalian perangkat audio yang dilakukan saat ini?
- d) Informasi apa saja yang diperlukan untuk mendukung *monitoring* kondisi sistem audio secara *real-time*?

2) Observasi Sistem Audio yang Ada

Peneliti juga melakukan observasi terhadap sistem audio masjid yang digunakan saat ini. Observasi ini bertujuan untuk memahami karakteristik sistem serta mengidentifikasi beberapa keterbatasan yang ada, antara lain:

- a) Pengaturan distribusi suara pada beberapa zona masih dilakukan secara manual.
- b) Belum tersedianya mekanisme *monitoring* status perangkat audio secara terpusat dan *real-time*.
- c) Keterbatasan dalam melakukan pengendalian dan pemantauan sistem audio dari perangkat bergerak (*mobile device*).

3) Analisis Kebutuhan Sistem

Berdasarkan hasil diskusi dan observasi yang telah dilakukan, peneliti menganalisis kebutuhan sistem yang akan

dikembangkan. Beberapa kebutuhan yang berhasil diidentifikasi meliputi:

- a) Kontrol Multi-Zona: Sistem harus mampu menyediakan pengaturan volume dan fungsi *mute* secara independen pada setiap zona audio.
- b) *Monitoring Real-time*: Sistem harus mampu menampilkan informasi kondisi perangkat audio, status koneksi perangkat, serta data *monitoring* yang dikirimkan melalui protokol MQTT secara *real-time*.
- c) Komunikasi Data *Real-time*: Sistem harus mendukung pertukaran data antara aplikasi Android dan perangkat *edge* secara *real-time* menggunakan protokol MQTT.
- d) Pemantauan Status Sistem: Sistem harus mampu menampilkan status operasional perangkat sehingga memudahkan proses *monitoring* dan pengelolaan sistem audio multi-zona.

c. Tujuan Identifikasi Masalah

Tujuan Tujuan dari identifikasi masalah ini adalah untuk:

- 1) Menentukan Fokus Penelitian: Dengan memahami masalah yang ada, peneliti dapat merumuskan fokus penelitian ini, yaitu pengembangan aplikasi *mobile* Android untuk *monitoring* dan kontrol *real-time* sistem distribusi audio masjid berbasis MQTT.
- 2) Mengidentifikasi Kebutuhan Pengguna: Peneliti dapat mengidentifikasi fitur-fitur utama yang harus ada dalam aplikasi, seperti kontrol volume dan *mute* per zona, *monitoring* status perangkat, serta kemudahan penggunaan bagi pengelola masjid.
- 3) Mendapatkan Dasar Perancangan Sistem: Identifikasi masalah ini juga menjadi dasar bagi perancangan sistem audio berbasis zona dan aplikasi *mobile* yang akan dikembangkan.

d. *Output* dari Tahap Identifikasi Masalah

Setelah tahap identifikasi masalah, peneliti menghasilkan beberapa *output* penting, yaitu:

- 1) Laporan Identifikasi Masalah: Sebuah laporan yang merangkum hasil wawancara, observasi, dan analisis kebutuhan pengguna.
- 2) Dokumen Kebutuhan Sistem: Dokumen yang merinci kebutuhan sistem audio yang lebih baik, serta fitur-fitur yang harus ada dalam aplikasi *mobile* yang akan dikembangkan.
- 3) Dasar Perancangan Aplikasi: Data dan informasi yang diperoleh digunakan untuk merancang aplikasi *mobile* Android yang dapat

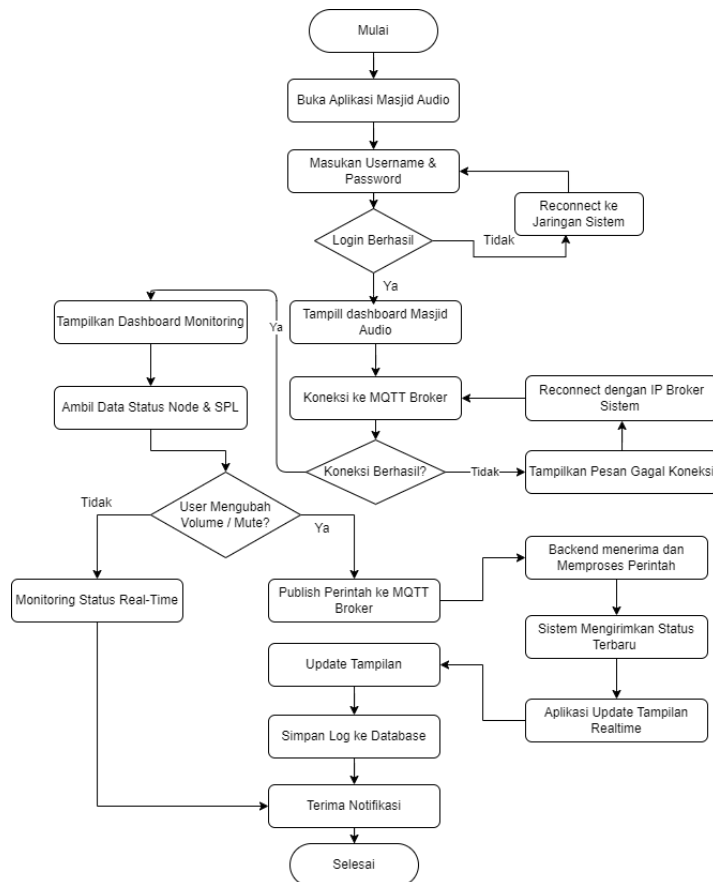
memenuhi kebutuhan pengelola masjid dan mengatasi masalah yang ada pada sistem audio tradisional.

2. Desain dan Perancangan Sistem

Pada tahap ini dilakukan perancangan arsitektur sistem, desain antarmuka aplikasi, serta mekanisme komunikasi menggunakan protokol MQTT. Selain itu, disusun flowchart sistem untuk menggambarkan alur kerja aplikasi dan komunikasi antara aplikasi *mobile*, MQTT broker, dan perangkat *node* audio. Flowchart ini memvisualisasikan proses *monitoring* status perangkat, pengiriman perintah kontrol, serta pembaruan data secara *real-time*.

a. Perancangan Sistem Audio Multi-Zona

Pada Sistem distribusi audio berbasis zona dirancang agar pengelola masjid dapat mengontrol volume dan *mute* secara independen pada setiap zona (seperti ruang salat, serambi kanan dan serambi kiri). Untuk memperjelas alur operasional sistem, mekanisme kerja divisualisasikan dalam bentuk flowchart sistem yang menunjukkan proses komunikasi dan kontrol audio secara *real-time* pada setiap zona. Berikut disajikan flowchart sistem pada gambar di bawah ini:



Gambar 3.2 Flowchart Sistem

Flowchart sistem menggambarkan alur kerja aplikasi android dalam melakukan *monitoring* dan kontrol sistem distribusi audio masjid berbasis MQTT. Proses dimulai ketika pengguna membuka aplikasi dan melakukan koneksi ke MQTT broker untuk menghubungkan aplikasi dengan *node* audio pada setiap zona masjid. Jika koneksi berhasil, aplikasi akan menampilkan dashboard *monitoring* yang memuat status *node* dan nilai *Sound Pressure Level* (SPL) secara *real-time*, sedangkan jika koneksi gagal sistem akan menampilkan pesan kesalahan. Selanjutnya, pengguna dapat melakukan kontrol seperti mengatur volume atau mengaktifkan fitur *mute* pada zona tertentu. Perintah tersebut dikirim melalui MQTT broker ke *node* audio untuk dieksekusi, kemudian *node* mengirimkan status terbaru ke aplikasi sehingga tampilan *monitoring* dapat diperbarui secara *real-time* serta aktivitas sistem dapat tercatat pada database *backend*..

Berdasarkan alur kerja sistem tersebut, perancangan sistem distribusi audio berbasis zona bertujuan untuk mengatasi permasalahan distribusi suara yang tidak merata di lingkungan masjid. Oleh karena itu, sistem dirancang dengan beberapa komponen utama yang mendukung fleksibilitas pengelolaan audio pada setiap area masjid, yaitu sebagai berikut:

- 1) Kontrol Zona: Setiap zona akan dilengkapi dengan pengaturan volume dan *mute* yang dapat dikontrol secara terpisah.
- 2) Sumber Suara: Sistem menampilkan informasi sumber suara untuk setiap zona (misalnya mikrofon atau ceramah). Implementasi fitur pemilihan/pergantian sumber suara secara aktif ditetapkan sebagai pengembangan lanjutan.

b. Desain Aplikasi *Mobile* Android

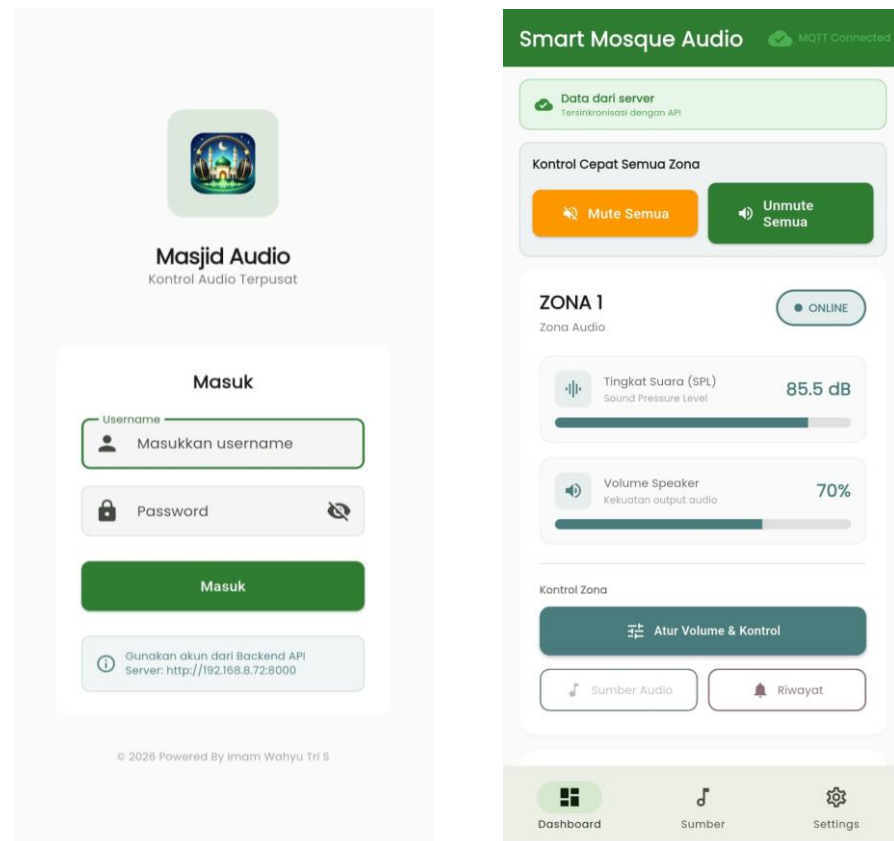
Aplikasi *mobile* dirancang untuk memberikan kontrol penuh kepada pengelola masjid dalam melakukan *monitoring* dan pengendalian sistem distribusi audio berbasis zona. Desain aplikasi menekankan kemudahan penggunaan (*user-friendly*) sehingga dapat dioperasikan oleh pengguna non-teknis tanpa memerlukan pemahaman teknis yang mendalam.

Perancangan aplikasi mencakup penyusunan struktur navigasi, pengelompokan fitur utama, serta penyajian informasi secara jelas dan sederhana. Fitur yang disediakan meliputi *monitoring* status *node* audio, tampilan nilai *Sound Pressure Level* (SPL) secara *real-time*, serta kontrol volume dan *mute* pada setiap zona. Selain itu, antarmuka dirancang dengan prinsip konsistensi dan kejelasan visual agar

pengguna dapat dengan mudah memahami kondisi sistem dan melakukan tindakan yang diperlukan secara cepat.

Untuk mendukung keamanan akses dan kemudahan penggunaan, aplikasi juga dilengkapi dengan mekanisme autentikasi pengguna melalui halaman *Login*. Setelah berhasil masuk, pengguna akan diarahkan ke halaman dashboard yang menampilkan ringkasan kondisi sistem secara keseluruhan. Dashboard ini menjadi pusat kontrol utama yang memungkinkan pengguna memantau status perangkat dan melakukan pengaturan audio pada setiap zona secara *real-time*.

Sebagai representasi dari perancangan tersebut, berikut disajikan contoh desain antarmuka aplikasi *mobile* Android yang meliputi tampilan *Login* page dan dashboard sebagai bagian utama dalam interaksi pengguna dengan sistem.



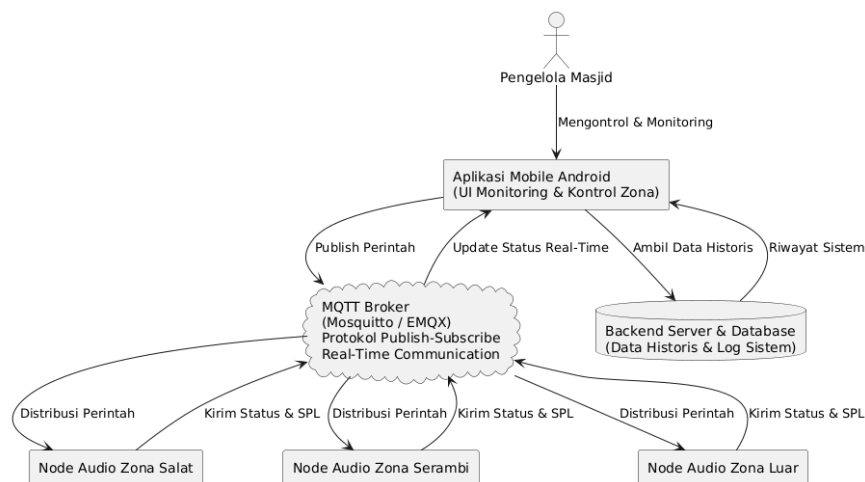
Gambar 3.3 *Login page* dan Dashboard *Mobile* Android

Fitur utama yang ada dalam aplikasi adalah:

- 1) Dashboard: Menampilkan status perangkat audio secara keseluruhan, seperti jumlah *node* aktif dan status perangkat (*online/offline*).

- 2) *Monitoring Zona*: Menampilkan status masing-masing zona (volume, SPL, status *online/offline*).
 - 3) *Kontrol Zona*: Pengelola dapat mengatur volume dan *mute* per zona. Informasi sumber suara yang digunakan pada tiap zona juga ditampilkan, sedangkan kontrol aktif untuk memilih/mengganti sumber suara ditetapkan sebagai pengembangan lanjutan.
 - 4) *Histori Sistem*: Pengelola akan menerima pemberitahuan kondisi perangkat audionya, seperti perangkat *offline* atau status perangkat tidak normal.
- c. Arsitektur Sistem dan Integrasi MQTT

Sistem menggunakan protokol komunikasi MQTT untuk memastikan komunikasi *real-time* antara aplikasi *mobile* dan perangkat audio masjid.



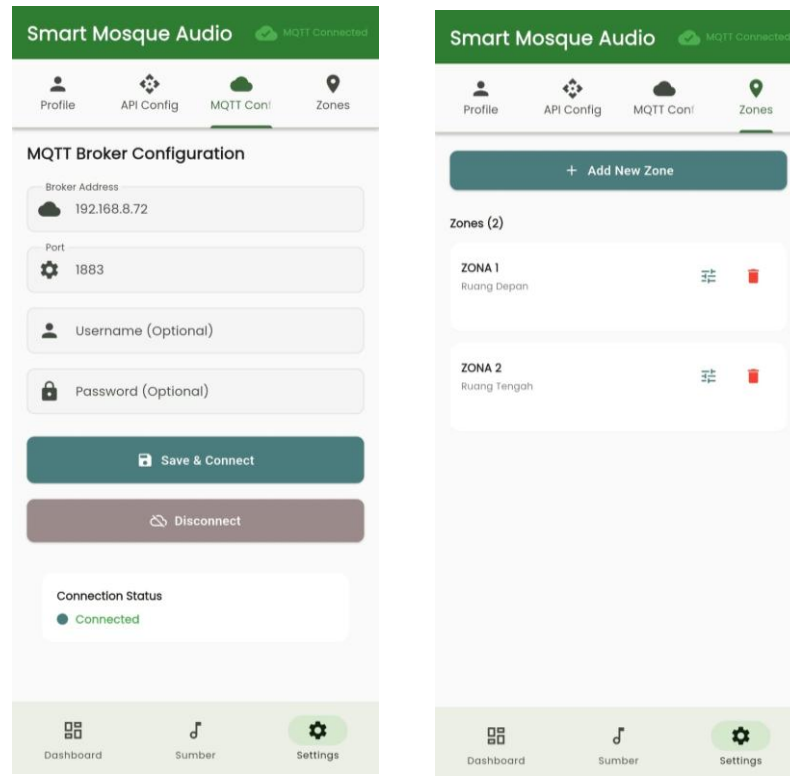
Gambar 3.4 Arsitektur Sistem Distribusi Audio Berbasis MQTT

Berikut adalah rincian arsitektur sistem yang dirancang:

- 1) *MQTT broker*: Berfungsi sebagai pusat komunikasi antara aplikasi *mobile* dan perangkat audio yang terhubung di masjid.
- 2) *Perangkat Audio IoT*: Perangkat audio di setiap zona masjid akan terhubung ke MQTT broker untuk menerima perintah kontrol dan mengirimkan status perangkat.
- 3) *Aplikasi Mobile*: Aplikasi ini berfungsi untuk memantau dan mengontrol status perangkat audio secara *real-time*, serta mengirimkan perintah kontrol berdasarkan *input* pengguna.

d. Desain Antarmuka Pengguna (UI/UX)

Desain antarmuka pengguna (UI) difokuskan pada kesederhanaan dan kemudahan penggunaan. Berikut contoh UI tampilan dari aplikasinya :



Gambar 3.5 Integrasi dengan MQTT broker dan Kontrol Zona

Beberapa elemen desain yang dipertimbangkan adalah:

- 1) Kemudahan Navigasi: Menu dan tombol yang mudah dijangkau dengan ukuran yang sesuai, sehingga pengguna dapat mengoperasikan aplikasi dengan cepat.
- 2) Feedback Visual: Aplikasi memberikan umpan balik langsung setelah setiap tindakan pengguna, seperti perubahan volume atau status perangkat.
- 3) Responsivitas: Desain antarmuka aplikasi dirancang agar responsif dan dapat digunakan pada berbagai ukuran layar perangkat Android.

e. Prototipe dan Uji Coba

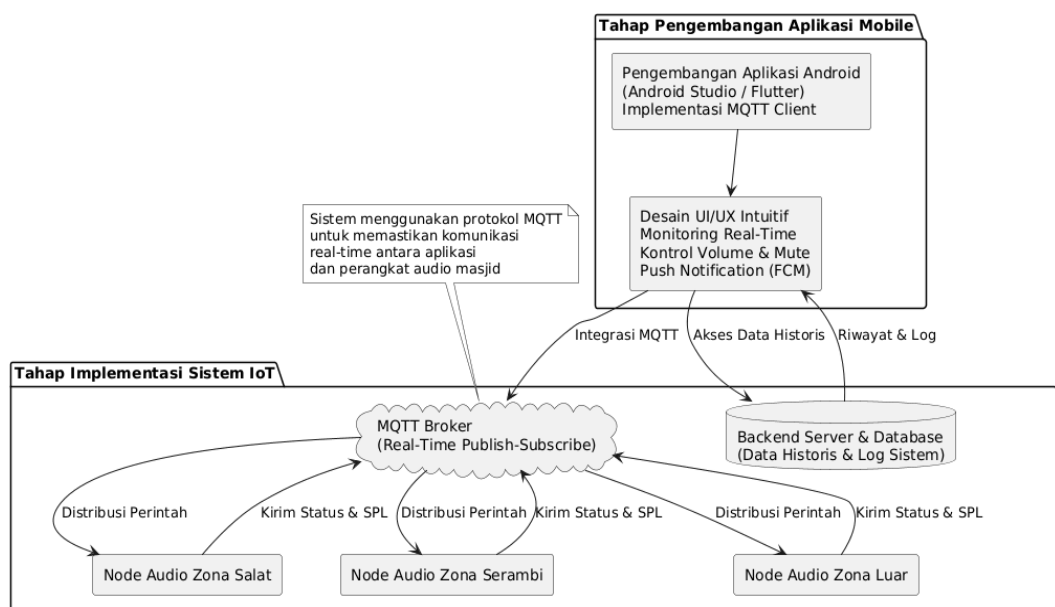
Setelah tahap perancangan antarmuka dan arsitektur kode diselesaikan, peneliti membangun prototipe awal (minimum viable product) aplikasi Android menggunakan *framework* Flutter dan arsitektur MVVM. Prototipe perangkat lunak ini diuji coba secara

mandiri oleh peneliti di dalam lingkungan laboratorium (eksperimental teknis).

Uji coba pada tahap prototipe ini bertujuan untuk memvalidasi fungsionalitas dasar aliran data (data stream) serta memastikan bahwa seluruh menu kontrol (volume dan *mute*) serta indikator nilai numerik SPL secara *real-time* telah terimplementasi dan terintegrasi dengan baik ke sisi MQTT broker. Hasil dari uji coba teknis awal ini digunakan sebagai acuan dasar oleh peneliti untuk melakukan perbaikan bug, optimalisasi penanganan memori, serta penyempurnaan baris kode aplikasi sebelum dilanjutkan ke tahap pengujian performa jaringan yang lebih komprehensif.

3. Pengembangan Aplikasi dan Implementasi Sistem

Pada tahap ini, peneliti mengimplementasikan desain aplikasi *mobile* dan sistem distribusi audio yang telah dirancang. Tahap ini melibatkan pengembangan perangkat lunak (aplikasi *mobile* Android) dan integrasi sistem distribusi audio berbasis IoT.



Gambar 3.6 Pengembangan Aplikasi dan Implementasi Sistem

Aplikasi *mobile* dikembangkan menggunakan Android Studio dengan bahasa pemrograman Dart. Beberapa fitur utama yang dikembangkan dalam aplikasi adalah:

a. Implementasi Sistem Audio Berbasis IoT

Sistem distribusi audio multi-zona masjid diimplementasikan menggunakan teknologi *Internet of Things* (IoT). Beberapa langkah yang dilakukan pada tahap implementasi sistem audio adalah:

- 1) Integrasi Perangkat Audio dengan IoT: Setiap perangkat audio (misalnya, speaker dan amplifier) yang terpasang di zona masjid dihubungkan dengan mikrokontroler (seperti ESP32) dan terhubung ke jaringan Wi-Fi untuk komunikasi *real-time*. Perangkat audio ini terhubung ke MQTT broker untuk menerima perintah kontrol dari aplikasi *mobile* dan mengirimkan data status perangkat secara berkala.
- 2) Pengaturan Zona Audio: Setiap zona audio dipetakan secara terstruktur untuk dapat dipantau tingkat intensitas suaranya melalui grafik nilai *Sound Pressure Level* (SPL) secara *real-time*. Adapun fitur kontrol *volume* dan status *mute* pada aplikasi ini dibatasi sebagai simulasi pengiriman instruksi data hingga level sistem (*backend* / MQTT broker) dan tidak mencakup eksekusi pada perangkat keras fisik karena belum mendukung fitur kontrol gain otomatis. Sementara itu, fitur pemilihan sumber suara tidak dicakup dalam penelitian ini dan dialokasikan sebagai tahap pengembangan lanjutan.
- 3) Pengaturan MQTT broker: MQTT broker (Mosquitto) diimplementasikan untuk mengelola komunikasi antara aplikasi *mobile* dan perangkat audio masjid. Broker ini akan menjadi perantara yang mengirimkan dan menerima pesan dalam bentuk topik-topik MQTT yang telah ditentukan, seperti:

```
smart-mosque/app/spl/zone/{zoneId}/status (status
    perangkat audio: online/offline)
smart-mosque/app/spl/zone/{zoneId}/spl
    (monitoring SPL real-time dengan min/max threshold)
smart-mosque/app/spl/zone/{zoneId}/volume
    (update nilai volume zona)
smart-mosque/app/spl/zone/{zoneId}/mute (perintah
    mute/unmute per zona)
```

- 4) Pengujian Koneksi dan Respons Sistem: Setiap fungsionalitas komunikasi data diuji untuk memastikan aplikasi *mobile* Android dapat berinteraksi dengan MQTT broker secara *real-time*. Pengujian difokuskan pada validasi kecepatan respons pengiriman pesan perintah kontrol (*volume* dan *mute*) dari aplikasi menuju *backend system*, serta pengukuran kestabilan koneksi jaringan untuk memastikan nilai latensi yang dihasilkan tetap rendah.

b. Pengembangan dan Pengujian Fitur Aplikasi

Setelah aplikasi *mobile* android dan sistem audio berbasis IoT dikembangkan, dilakukan serangkaian pengujian untuk memastikan bahwa aplikasi dan sistem berfungsi dengan baik.

Tabel 3.2 Pengembangan dan Pengujian Fitur Aplikasi

No.	Fitur	Deskripsi	Hasil Pengujian
1	<i>Monitoring Status Node</i>	Aplikasi menampilkan status <i>node</i> audio (<i>online/offline</i>) secara <i>real-time</i> melalui komunikasi MQTT broker.	Berhasil – Status <i>node</i> ditampilkan secara <i>real-time</i> sesuai kondisi perangkat dengan waktu respon cepat.
2	<i>Monitoring Sound Pressure Level (SPL)</i>	Aplikasi menampilkan nilai <i>Sound Pressure Level</i> (dB) dari setiap <i>node</i> untuk memantau tingkat kebisingan audio masjid.	Berhasil – Nilai SPL ditampilkan sesuai data sensor dan diperbarui secara <i>real-time</i> .
3	Kontrol Volume <i>Node</i>	Aplikasi mengirimkan parameter perubahan nilai intensitas volume berdasarkan pergeseran komponen slider kontrol volume pada antarmuka.	Berhasil – Perubahan parameter angka volume sukses diterbitkan (<i>published</i>) ke topik MQTT broker dan tercatat secara valid di <i>backend</i> .
4	Fungsi <i>Mute/Unmute</i>	Aplikasi menyediakan fitur untuk membisukan (<i>mute</i>) dan mengaktifkan kembali (<i>unmute</i>) status audio.	Berhasil – Perintah <i>mute</i> dan <i>unmute</i> sukses dikirimkan menuju MQTT broker dan status indikator pada aplikasi berubah sesuai <i>input</i> .
5	<i>Monitoring Multi-Zona</i>	Aplikasi menampilkan pemisahan visual dan data kontrol dari beberapa <i>node</i> berdasarkan pembagian zona masjid.	Berhasil – Setiap <i>node</i> terpetakan secara independen sesuai zona, di mana kontrol pada satu zona tidak memengaruhi zona lainnya.

No.	Fitur	Deskripsi	Hasil Pengujian
6	Sinkronisasi dengan MQTT broker	Aplikasi terhubung dengan MQTT broker untuk komunikasi <i>real-time</i> antara aplikasi dan perangkat <i>node</i> .	Berhasil – Aplikasi berhasil <i>subscribe</i> dan <i>publish</i> data secara <i>real-time</i> tanpa <i>error</i> .
7	Integrasi Database Backend (REST API)	Aplikasi melakukan pemanggilan data registrasi <i>node</i> serta log riwayat peristiwa dari database <i>backend</i> (PostgreSQL) melalui protokol HTTP REST API.	Berhasil – Aplikasi Flutter sukses melakukan <i>request</i> data dan menerima respon JSON secara valid untuk menampilkan riwayat log secara terstruktur.
8	Rendering Antarmuka Pengguna (UI/UX)	Aplikasi menyajikan tata letak visual komponen kontrol, indikator monitoring, teks Bahasa Indonesia, serta perpindahan halaman menu (<i>navigation menu</i>).	Berhasil – Seluruh komponen visual <i>widget</i> Flutter ter-render secara presisi, responsif terhadap sentuhan simulasi, dan navigasi antar-tab berjalan lancar.

Pengujian dilakukan untuk mengukur berbagai aspek, antara lain:

1) Pengujian Fungsionalitas Aplikasi:

Memastikan fitur-fitur yang telah diimplementasikan pada aplikasi *mobile* berfungsi dengan baik, seperti kontrol volume, *mute*, *monitoring* status perangkat, serta penampilan informasi sumber suara pada tiap zona. Pengujian dilakukan menggunakan beberapa perangkat Android dan perangkat audio yang terhubung ke sistem. Pada penelitian ini, fitur pemilihan/pergantian sumber suara secara aktif belum diimplementasikan dan ditetapkan sebagai pengembangan lanjutan.

2) Pengujian Kinerja Sistem:

Pengujian untuk memastikan latensi komunikasi antara aplikasi dan perangkat audio minimal (di bawah 500ms). Mengukur penggunaan daya pada perangkat Android dan efisiensi penggunaan bandwidth pada jaringan Wi-Fi.

c. Integrasi dan Uji Coba Sistem

Setelah komponen aplikasi *mobile* Android, MQTT broker, dan *backend system* terintegrasi, dilakukan pengujian sistem secara menyeluruh di lingkungan laboratorium (eksperimental teknis). Pada tahap ini, pengujian meliputi:

1) Integrasi Aplikasi dan MQTT broker :

Pengujian ini bertujuan untuk memastikan bahwa arsitektur komunikasi dua arah (*publish-subscribe*) menggunakan protokol MQTT antara aplikasi Android dan broker dapat berjalan lancar secara *real-time*. Proses pengujian difokuskan pada kevalidan perubahan data, di mana setiap instruksi yang dikirimkan melalui menu kontrol (*volume* dan *mute*) pada aplikasi terbukti berhasil diterima, diproses, dan mengubah status data di sisi MQTT broker serta tercatat pada *log database backend*.

2) Pengujian Simulasi dan Performa Jaringan (Eksperimen Laboratorium):

Pengujian dilakukan melalui simulasi fungsional tanpa melibatkan perangkat keras fisik audio di lapangan maupun pengguna langsung. Validasi pengujian diukur berdasarkan:

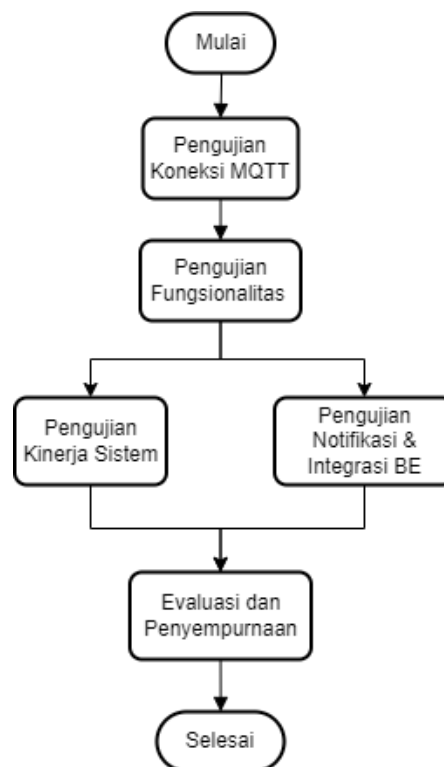
- a) Kevalidan Data Kontrol: Memastikan kecocokan parameter nilai *volume* dan status logika *mute* yang dikirim dari aplikasi dengan data yang tertangkap di *MQTT client monitoring tools* (seperti MQTT Explorer atau Mosquitto CLI).
- b) Kinerja Transmisi (*Response time*): Mengukur efisiensi dan kecepatan waktu respon (*latency*) pengiriman paket data perintah dari aplikasi menuju MQTT broker untuk memastikan sistem bekerja secara responsif dalam skala laboratorium.

4. Pengujian Aplikasi dan Sistem

Pada tahap awal penyusunan proposal, pengujian aplikasi direncanakan menggunakan metode *User acceptance Testing* (UAT) untuk mengevaluasi tingkat penerimaan pengguna terhadap aplikasi yang dikembangkan. Namun, pada tahap pelaksanaan penelitian, metode pengujian disesuaikan menjadi *Black-box Testing* karena evaluasi pada penelitian difokuskan pada validasi fungsionalitas aplikasi serta pengukuran performa komunikasi data MQTT melalui parameter *response time*. Penyesuaian ini dilakukan karena penelitian tidak melibatkan pengguna akhir sebagai responden, sehingga evaluasi difokuskan pada pengujian kesesuaian fungsi aplikasi terhadap kebutuhan

fungsional. Oleh karena itu, metode *Black-box Testing* dipilih sebagai metode yang lebih sesuai dengan tujuan dan ruang lingkup penelitian.

Tahap pengujian Aplikasi dan Sistem bertujuan untuk memverifikasi kesesuaian fungsi aplikasi serta mengevaluasi performa komunikasi data secara teknis. Evaluasi dilakukan melalui pengujian fungsionalitas fitur kontrol pada aplikasi *mobile* Android dan pengukuran parameter *response time* transmisi data protokol MQTT dari aplikasi menuju *backend system* dan MQTT broker. Pengujian dilakukan secara eksperimental pada lingkungan laboratorium tanpa melibatkan pengguna langsung maupun eksekusi perangkat keras fisik audio di lapangan.



Gambar 3.7 Pengujian Aplikasi dan Sistem

Pengujian ini dilakukan dalam beberapa aspek penting, yaitu fungsionalitas, notifikasi dan kinerja sistem.

a. Pengujian Integrasi dan Koneksi MQTT

Pengujian integrasi teknis dilakukan untuk memastikan arsitektur komunikasi data dua arah (*publish-subscribe*) antara aplikasi *mobile* Android, *backend system*, dan MQTT broker berfungsi secara sinkron. Aspek pengujian ini meliputi:

1) Pengujian Koneksi MQTT broker:

Memastikan bahwa aplikasi *mobile* dapat terhubung dengan

MQTT broker dengan baik, dan bahwa perangkat audio dapat berkomunikasi dengan aplikasi secara *real-time* melalui broker tersebut.

2) Pengujian Sinkronisasi Data Kontrol dan Monitoring:

Memastikan data status yang dikirimkan dari aplikasi langsung tercatat pada topik di MQTT broker dan database *backend*, serta sebaliknya, visualisasi data status *node* (*online/offline*) dan nilai *Sound Pressure Level* (SPL) dari *backend system* berhasil diterima dan ditampilkan oleh aplikasi secara *real-time*.

b. Pengujian Fungsionalitas Aplikasi

Pengujian fungsionalitas menggunakan metode *Black-box Testing* difokuskan untuk memvalidasi bahwa seluruh komponen antarmuka pengguna (UI/UX) dan menu kontrol status pada aplikasi Android berfungsi dengan benar sesuai rancangan logis arsitektur MVVM.

Tabel 3.3 Pengujian Fungsionalitas Aplikasi

No.	Fitur yang Diuji	Deskripsi Pengujian	Hasil yang Diharapkan
1	Kontrol Volume per Zona	Mengubah parameter nilai <i>slider</i> volume pada zona tertentu melalui antarmuka aplikasi <i>mobile</i> Android	Data nilai volume berubah pada topik MQTT broker yang sesuai dan tercatat di database <i>backend</i> .
2	Kontrol <i>Mute</i> per Zona	Menekan tombol aktif/nonaktif <i>mute</i> audio pada zona tertentu melalui antarmuka aplikasi.	Logika status data berubah pada topik MQTT broker dan database <i>backend</i> .
3	<i>Monitoring</i> Status Keaktifan <i>Node</i>	Mengamati visualisasi komponen indikator status operasional <i>node</i> audio pada antarmuka aplikasi.	Aplikasi menampilkan status <i>Online</i> atau <i>Offline</i> secara tepat berdasarkan pasokan data dari <i>backend</i> .

No.	Fitur yang Diuji	Deskripsi Pengujian	Hasil yang Diharapkan
4	Monitoring Nilai SPL	Mengamati pembaruan nilai suara (<i>Sound Pressure Level</i>) per zona pada aplikasi.	UI Aplikasi secara <i>realtime</i> menampilkan nilai SPL sesuai limpahan data <i>stream</i> dari <i>backend</i> .

c. Pengujian Performa Jaringan (*Response time* MQTT)

Pengujian performa jaringan dilakukan untuk mengukur efisiensi, responsivitas, dan stabilitas pengiriman pesan kontrol berbasis protokol MQTT dari aplikasi Android menuju *backend system*. Pengujian teknis ini mencakup:

1) Pengukuran Waktu Respons dan Latensi:

Menghitung durasi (dalam satuan milidetik/*ms*) yang dibutuhkan sejak tombol kontrol (*volume* atau *mute*) ditekan pada antarmuka aplikasi hingga pesan kontrol tersebut berhasil diterima, diproses, dan diterbitkan (*published*) oleh MQTT broker.

2) Pengujian Kinerja Jaringan dan Bandwidth

Mengukur beban konsumsi data (*bandwidth*) yang dihasilkan oleh aktivitas pengiriman pesan kontrol serta penerimaan data *monitoring stream* SPL secara konstan melalui protokol MQTT untuk memastikan konsumsi data jaringan tetap efisien dalam kondisi laboratorium.

d. Pengujian Notifikasi dan Integrasi *Backend system*

Pengujian notifikasi dan integrasi *backend system* dilakukan untuk memastikan bahwa proses pertukaran data antara aplikasi Android, MQTT broker, dan *backend system* dapat berjalan dengan baik. Pengujian ini bertujuan untuk memverifikasi bahwa setiap perubahan data yang terjadi pada sistem dapat diterima dan ditampilkan oleh aplikasi secara *real-time* serta seluruh komponen sistem mampu berkomunikasi secara sinkron.

Aspek pengujian yang dilakukan meliputi:

1) Pengujian Integrasi Aplikasi dengan *Backend system*

Pengujian ini bertujuan untuk memastikan bahwa aplikasi Android berhasil terhubung dengan *backend system* dan dapat menerima data *monitoring* yang disediakan oleh *backend*. Data yang diuji meliputi status *node* (*online/offline*), nilai *Sound Pressure Level* (SPL), serta informasi status kontrol yang tersimpan pada sistem.

Pengujian dinyatakan berhasil apabila data yang tersedia pada *backend system* dapat diterima dan ditampilkan oleh aplikasi Android tanpa terjadi kehilangan data maupun ketidaksesuaian informasi.

2) Pengujian Sinkronisasi Status Kontrol

Pengujian ini dilakukan untuk memastikan bahwa setiap perintah kontrol yang dikirimkan melalui aplikasi Android dapat diteruskan menuju MQTT broker dan diproses oleh *backend system*. Selain itu, perubahan status yang terjadi pada *backend* juga harus dapat diterima kembali oleh aplikasi sehingga kondisi sistem yang ditampilkan selalu sesuai dengan data terbaru.

Pengujian dinyatakan berhasil apabila perubahan status volume dan *mute* yang dikirimkan melalui aplikasi dapat tercatat pada *backend system* dan selanjutnya diperbarui kembali pada antarmuka aplikasi secara *real-time*.

3) Pengujian Pembaruan Data *Monitoring* Secara *Real-time*

Pengujian ini bertujuan untuk memastikan bahwa data *monitoring* yang berasal dari *backend system* dapat diterima oleh aplikasi Android secara otomatis tanpa memerlukan penyegaran (*refresh*) manual. Data yang diuji meliputi status keaktifan *node* dan nilai SPL pada setiap zona yang dimonitor.

Pengujian dinyatakan berhasil apabila setiap perubahan data pada *backend* dapat ditampilkan pada aplikasi dalam waktu yang cepat dan sesuai dengan kondisi aktual sistem.

e. Revisi dan Penyempurnaan Sistem

Setelah tahap pengujian aplikasi dan sistem dilakukan, tahap berikutnya adalah Revisi dan Penyempurnaan Sistem. Pada tahap ini, peneliti melakukan analisis mendalam terhadap hasil pengujian dan umpan balik yang diperoleh untuk memperbaiki dan menyempurnakan aplikasi serta sistem distribusi audio masjid berbasis IoT. Tujuan utama dari tahap ini adalah untuk memastikan

bahwa sistem yang dikembangkan dapat memenuhi kebutuhan pengguna dengan lebih baik dan meningkatkan performa aplikasi.

1) Analisis Hasil Pengujian Sistem

Setelah melakukan pengujian fungsionalitas, performa jaringan, dan integrasi sistem secara teknis di laboratorium, peneliti mengumpulkan data hasil pengujian untuk dianalisis lebih lanjut, yang mencakup:

- a) Hasil Pengujian Fungsionalitas: Peneliti menganalisis apakah seluruh fitur pada antarmuka aplikasi Android berfungsi dengan benar berdasarkan arsitektur MVVM, termasuk kontrol *volume*, logika status *mute* per zona, serta visualisasi data *monitoring* status *node* dan grafik SPL. Jika ditemukan ketidaksesuaian data antara *input* pada aplikasi dengan data yang diterima oleh MQTT broker, maka struktur pemetaan data pada kode program (*ViewModel* dan *Model*) perlu diperbaiki.
- b) Hasil Pengujian Performa Jaringan: Data performa berupa parameter waktu respon (*latency*) pengiriman pesan perintah dari aplikasi menuju MQTT broker akan dianalisis untuk mengukur tingkat responsivitas sistem. Jika hasil pengukuran menunjukkan nilai *latency* atau waktu tunggu yang tinggi di dalam lingkungan jaringan laboratorium, maka optimasi pada pengelolaan *traffic* data komunikasi dan algoritma *publish-subscribe* MQTT perlu dilakukan.
- c) Hasil Pengujian Integrasi Sistem: Analisis ini mengevaluasi sejauh mana aplikasi *mobile* Android dapat terintegrasi dan bertukar data secara konsisten dengan *backend system* melalui perantara MQTT broker. Fokus analisis ditujukan pada stabilitas sinkronisasi data operasional zona, di mana jika ditemukan kendala seperti kegagalan pencatatan log pada *database backend* atau putusnya sesi komunikasi (*keep-alive*) pada broker, maka aspek konfigurasi koneksi *client* dan integrasi sistem harus diperbaiki.

2) Penyempurnaan Fitur Aplikasi

Berdasarkan hasil analisis, beberapa fitur aplikasi yang perlu diperbaiki atau disempurnakan dapat meliputi:

- a) Peningkatan Responsivitas Transmisi Data:
 - (1) Optimalisasi Latensi MQTT: Jika ditemukan bahwa waktu respons pengiriman pesan terlalu lama, peneliti

akan melakukan optimalisasi pada manajemen *traffic* data komunikasi dan struktur pemetaan fungsi *publish-subscribe* untuk mengurangi latensi transmisi antara aplikasi dan MQTT broker.

- (2) Efisiensi Manajemen State UI : Mengoptimalkan pemrosesan aliran data (*asynchronous stream*) pada arsitektur MVVM agar pembaruan data grafik SPL dan status keaktifan *node* pada antarmuka aplikasi dapat berjalan lebih ringan tanpa membebani kinerja memori perangkat Android.

b) Penyempurnaan Sinkronisasi Data Kontrol:

Peningkatan Akurasi Kontrol Volume: Jika terdapat ketidaksesuaian antara nilai parameter yang diinput pada antarmuka aplikasi dengan data yang diterima oleh MQTT broker, peneliti akan memeriksa kembali algoritma pemrosesan data di sisi *ViewModel* dan *Model* untuk memastikan instruksi simulasi *volume* dan status *mute* terkirim serta tercatat secara valid pada database *backend*.

3) Penyempurnaan Sistem Perangkat Lunak dan Integrasi Data

Beberapa aspek dalam implementasi aplikasi Android dan arsitektur data yang perlu disempurnakan setelah proses analisis hasil pengujian adalah:

a) Peningkatan Komunikasi MQTT:

Stabilitas Koneksi: Jika hasil pengujian teknis menunjukkan adanya pemutusan sesi (*keep-alive*) atau gangguan dalam komunikasi data, peneliti akan melakukan perbaikan pada konfigurasi parameter koneksi *client* di dalam aplikasi atau optimasi pada pengaturan MQTT broker demi menjamin kelancaran pertukaran data *real-time*.

b) Pengelolaan Data Historis:

Optimasi Penyimpanan Database *Backend*: Memaksimalkan efisiensi penyimpanan log data historis nilai SPL dan status operasional zona pada database *backend* (PostgreSQL) melalui pembatasan periode penyimpanan (*data retention policy*) atau pengaturan query yang lebih efisien, sehingga performa aplikasi saat memanggil data historis tetap cepat.

c) Pengembangan Manajemen Struktur Zona:

Skalabilitas Manajemen Zona: Mengoptimalkan struktur kode pada lapisan *Model* dan *ViewModel* agar arsitektur aplikasi Flutter dapat mendukung pengelolaan visualisasi

data serta simulasi status kontrol untuk jumlah zona audio yang lebih banyak secara simultan apabila diperlukan pengembangan sistem di masa mendatang.

4) Uji Coba dan Validasi Hasil Revisi

Setelah dilakukan proses perbaikan dan optimasi sistem perangkat lunak, peneliti akan melakukan uji coba ulang dalam skala laboratorium untuk memastikan bahwa semua perubahan kode yang diimplementasikan memberikan hasil teknis yang diinginkan. Uji coba ulang ini meliputi:

- a) Uji Fungsionalitas: Memastikan bahwa semua fitur yang diperbaiki berfungsi sesuai dengan harapan.
- b) Uji Kinerja: Mengukur apakah latensi dan penggunaan daya telah berkurang setelah dilakukan optimasi.
- c) Uji Stabilitas Sistem: Memastikan bahwa aliran data (*asynchronous stream*) antara aplikasi Android, MQTT broker, dan database *backend* dapat beroperasi secara konsisten dan stabil tanpa mengalami kendala kehilangan data (*packet loss*) atau kegagalan pencatatan log sistem selama periode pengujian di laboratorium.

D. Analisis Data

Pada tahap ini, data yang telah dikumpulkan dari proses pengujian fungsionalitas fitur kontrol status dan pengujian performa jaringan (*response time*) pesan MQTT akan dianalisis untuk mengevaluasi apakah kualitas perangkat lunak aplikasi Android yang dikembangkan telah memenuhi tujuan penelitian teknis. Analisis data ini dilakukan dengan pendekatan eksperimental laboratorium untuk memberikan gambaran logis tentang responsivitas aliran data (*asynchronous stream*) dan kevalidan sinkronisasi integrasi sistem. Berikut adalah diagram alur yang menggambarkan tahapan-tahapan dalam analisis data yang dilakukan pada penelitian ini.

1. Analisis Koneksi dan Komunikasi MQTT

Analisis dilakukan untuk memastikan mekanisme komunikasi data berbasis MQTT dapat berjalan dengan baik antara aplikasi Android dan *backend system*. Aspek yang dianalisis meliputi:

- a. Keberhasilan Koneksi MQTT : Analisis dilakukan dengan mengamati kemampuan aplikasi dalam membangun koneksi ke MQTT broker serta mempertahankan status koneksi selama proses pengujian berlangsung.

- b. Pengiriman dan Penerimaan Pesan : Analisis dilakukan dengan membandingkan data yang dikirimkan aplikasi melalui mekanisme *publish* dengan data yang diterima oleh *backend system* melalui topik MQTT yang sesuai. Hasil pengujian digunakan untuk memastikan tidak terjadi kegagalan pengiriman pesan selama proses komunikasi berlangsung.

2. Analisis Fungsionalitas Aplikasi (*Black-box Testing*)

Analisis fungsionalitas dilakukan menggunakan metode *Black-box Testing* untuk memastikan setiap fitur aplikasi berjalan sesuai dengan kebutuhan sistem tanpa memperhatikan struktur kode program.

- a) Fitur *monitoring* Sistem : Pengujian dilakukan untuk memastikan aplikasi mampu menampilkan informasi status *node* (*online/offline*) dan nilai *Sound Pressure Level* (SPL) yang diterima dari *backend* secara benar dan *real-time*.
- b) Fitur Kontrol Volume : Pengujian dilakukan untuk memastikan perubahan nilai volume yang dilakukan melalui slider aplikasi dapat menghasilkan data kontrol yang sesuai dan berhasil dikirimkan ke MQTT broker.
- c) Fitur Kontrol *Mute* : Pengujian dilakukan untuk memastikan tombol *mute* dapat mengubah status kontrol sesuai kondisi yang dipilih pengguna dan berhasil diteruskan ke *backend system* melalui MQTT.
- d) Fitur Navigasi dan Antarmuka : Pengujian dilakukan untuk memastikan seluruh halaman, menu, tombol, dan komponen antarmuka aplikasi dapat berfungsi sesuai rancangan sistem.

3. Analisis Kinerja Sistem (*Response time* MQTT)

Analisis kinerja sistem dilakukan untuk mengetahui kecepatan komunikasi data yang terjadi antara aplikasi Android dan *backend system* melalui MQTT broker. Parameter yang dianalisis adalah:

Response time Pengiriman Perintah: Pengukuran dilakukan dengan menghitung selisih waktu sejak pengguna mengirimkan perintah kontrol melalui aplikasi hingga pesan diterima dan diproses oleh *backend system*.

Perhitungan *response time* menggunakan persamaan:

$$Response\ Time = T_{exec} - T_{user}$$

Keterangan:

- a) T_{user} = waktu saat pesan dikirim aplikasi
- b) T_{exec} = waktu saat pesan diterima *backend*

Hasil pengukuran disajikan dalam satuan milidetik (ms) dan digunakan untuk mengevaluasi kemampuan sistem dalam mendukung komunikasi *real-time*.

4. Analisis Notifikasi dan Integrasi *Backend*

Analisis dilakukan untuk memastikan aplikasi mampu menerima pembaruan status dari *backend system* dan menampilkan informasi kondisi perangkat secara tepat.

- a) Notifikasi Kondisi Perangkat : Pengujian dilakukan untuk memastikan aplikasi dapat menampilkan informasi kondisi perangkat berdasarkan data yang diterima dari *backend*, baik dalam kondisi normal maupun tidak normal.
- b) Sinkronisasi Data Monitoring: Pengujian dilakukan untuk memastikan perubahan status perangkat yang terjadi pada *backend system* dapat diterima dan ditampilkan oleh aplikasi Android secara *real-time*.

5. Evaluasi dan Penyajian Hasil

Setelah analisis data dilakukan, hasil pengujian akan disajikan dalam bentuk tabel pengujian *Black-box Testing*, tabel hasil pengukuran *response time* MQTT, nilai rata-rata *response time*, serta pembahasan deskriptif terhadap hasil yang diperoleh. Penyajian data tersebut digunakan untuk mengevaluasi keberhasilan implementasi aplikasi Android berbasis MQTT dalam mendukung *monitoring* dan kontrol sistem audio multi-zona secara *real-time*.

E. Jadwal Penelitian

Jadwal penelitian ini mencakup tahapan-tahapan yang telah dijelaskan sebelumnya, dengan perkiraan waktu pelaksanaan untuk setiap tahap yang diperlukan. Waktu yang dibutuhkan untuk penelitian ini adalah sekitar 7 bulan, dimulai dari penyusunan rencana penelitian, perancangan, pengujian, revisi dan penyempurnaan sistem hingga penyusunan laporan skripsi dan sidang. Di bawah ini adalah rincian jadwal penelitian yang mencakup setiap tahapan kegiatan beserta perkiraan waktu yang diperlukan mulai dari bulan Desember 2025 sampai dengan bulan Juni 2026 yang ditunjukkan pada Tabel 3.4 Jadwal Penelitian.

Tabel 3.4 Jadwal Penelitian

No.	Kegiatan	Des 2025	Jan 2026	Feb 2026	Mar 2026	Apr 2026	Mei 2026	Jun 2026
1	Penyusunan Rencana Penelitian							
2	Pengumpulan Literatur dan Wawancara							
3	Perancangan Sistem Audio Multi-Zona dan Aplikasi							
4	Pengembangan Aplikasi <i>Mobile</i> dan Sistem Audio IoT							
5	Pengujian Aplikasi dan Sistem							
6	Revisi dan Penyempurnaan Sistem							
7	Penulisan Laporan Skripsi dan Sidang							

BAB IV HASIL DAN PEMBAHASAN

A. Hasil

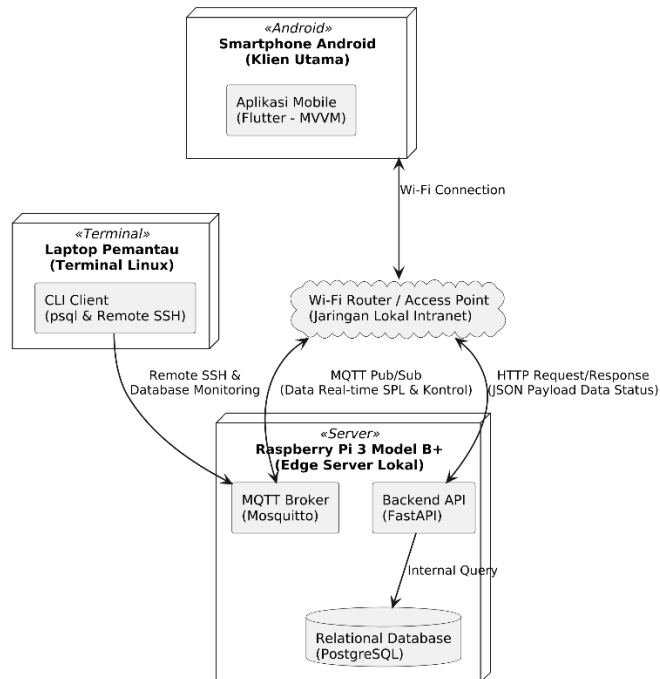
1. Gambaran Umum Sistem

Penelitian ini menghasilkan sebuah aplikasi *mobile* berbasis Android yang digunakan untuk melakukan *monitoring* dan kontrol pada sistem audio multi-zona masjid melalui protokol MQTT. Aplikasi dikembangkan menggunakan *framework* Flutter dengan menerapkan pola arsitektur *Model-View-ViewModel* (MVVM) untuk memisahkan logika bisnis, pengelolaan data, dan antarmuka pengguna.

Sistem yang dibangun memanfaatkan mekanisme komunikasi data berbasis *publish-subscribe* melalui MQTT broker sebagai perantara pertukaran data antara aplikasi Android dan *backend system*. Aplikasi menerima data *monitoring* berupa status *node* dan nilai *Sound Pressure Level* (SPL) secara *real-time* dari *backend*, kemudian menampilkan informasi tersebut pada antarmuka pengguna. Selain itu, aplikasi juga dapat mengirimkan perintah kontrol berupa perubahan nilai volume dan status *mute* yang selanjutnya diteruskan ke MQTT broker untuk diproses oleh *backend system*.

Sesuai dengan batasan penelitian yang telah ditetapkan, fitur kontrol volume dan *mute* pada penelitian ini dibatasi hingga proses pengiriman instruksi, penerimaan pesan pada MQTT broker, serta sinkronisasi data pada *backend system*. Penelitian tidak mencakup implementasi kontrol fisik terhadap perangkat audio karena perangkat *node* yang digunakan belum mendukung mekanisme *Automatic Gain Control* (AGC).

Berdasarkan arsitektur jaringan yang disajikan pada Gambar 4.1, implementasi seluruh infrastruktur kontrol dan penyimpanan data dirancang menggunakan model *Local Distributed Network* (Jaringan Lokal Terdistribusi). Seluruh komponen perangkat lunak dijalankan di dalam lingkungan laboratorium menggunakan jaringan *Intranet* (lokal) yang dijumpatani oleh sebuah Wi-Fi Router tanpa bergantung pada koneksi internet publik.



Gambar 4.1 Arsitektur Sistem yang digunakan Pada Penelitian Ini

Aliran data dan interaksi antar-komponen di dalam arsitektur ini dibagi menjadi tiga bagian utama:

a. Sisi Klien (*Frontend App*):

Aplikasi *mobile* Android yang dibangun dengan *framework* Flutter dan arsitektur MVVM bertindak sebagai antarmuka utama. Aplikasi ini terhubung ke Wi-Fi Router untuk melakukan dua aksi komunikasi simultan :

- 1) Protokol MQTT: Melakukan *publish* payload JSON berisi instruksi kontrol (*volume* dan *mute*) serta melakukan *subscribe* secara *real-time* untuk menangkap perubahan data numerik desibel (dB) SPL dari broker.
- 2) Protokol HTTP (REST API): Mengirimkan *request* GET/POST menuju FastAPI untuk kebutuhan manajemen data statis seperti autentikasi *Login* operator dan penarikan riwayat log peristiwa.

b. Sisi *Edge* Server (Raspberry Pi 3 Model B+):

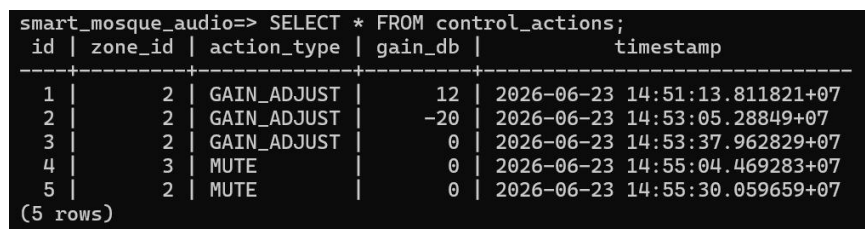
Raspberry Pi 3 Model B+ bertindak sebagai pusat pemrosesan data lokal (*edge server*). Di dalam perangkat ini, tiga service utama berjalan secara terintegrasi:

- 1) Mosquitto MQTT broker mengelola antrean pesan *publish-subscribe* dengan latensi rendah.
- 2) FastAPI *Backend* bertindak sebagai perantara logis yang memproses request HTTP dari aplikasi Android.
- 3) PostgreSQL Database berfungsi sebagai media penyimpanan permanen untuk mengunci data konfigurasi zona, data akun users, dan tabel *log_events*.

c. Sisi *Monitoring* Independen (Laptop Pemantau):

Untuk memastikan transparansi dan validitas data selama proses pengujian, digunakan sebuah komputer pemantau yang terhubung ke *backend system*. Komputer ini berfungsi untuk memverifikasi bahwa setiap instruksi kontrol yang dikirim dari aplikasi Android berhasil diterima dan disimpan oleh *backend system*.

Proses pemantauan dilakukan menggunakan utilitas *psql* melalui Command Prompt (CMD) pada sistem operasi Windows seperti yang ditunjukkan pada Gambar 4.2. Melalui koneksi ke database PostgreSQL, peneliti dapat menjalankan perintah *query* secara langsung untuk melihat data yang tersimpan pada tabel sistem. Dengan metode ini, setiap instruksi kontrol seperti perubahan nilai volume (*gain adjustment*) maupun status *mute* yang dikirim melalui aplikasi Android dapat diverifikasi secara *real-time* melalui pencatatan data pada database *backend*.



id	zone_id	action_type	gain_db	timestamp
1	2	GAIN_ADJUST	12	2026-06-23 14:51:13.811821+07
2	2	GAIN_ADJUST	-20	2026-06-23 14:53:05.28849+07
3	2	GAIN_ADJUST	0	2026-06-23 14:53:37.962829+07
4	3	MUTE	0	2026-06-23 14:55:04.469283+07
5	2	MUTE	0	2026-06-23 14:55:30.059659+07

(5 rows)

Gambar 4.2 Pemantauan *Backend* melalui CMD

2. Implementasi Antarmuka Aplikasi Android

Antarmuka aplikasi dirancang untuk memudahkan pengguna dalam melakukan *monitoring* kondisi sistem audio multi-zona serta melakukan kontrol secara *real-time*. Desain antarmuka dikembangkan dengan memperhatikan aspek kemudahan penggunaan, keterbacaan informasi, dan responsivitas tampilan pada perangkat Android.

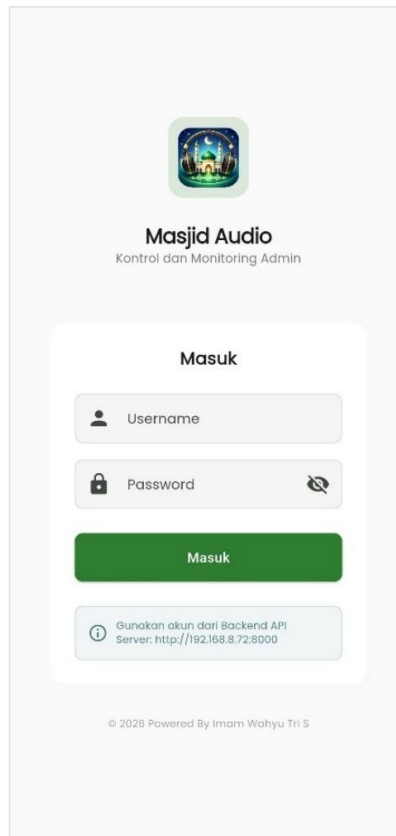
Fitur utama yang tersedia pada aplikasi meliputi halaman autentikasi pengguna, dashboard monitoring, halaman detail zona, menu kontrol volume dan

mute, serta halaman notifikasi kondisi perangkat. Setiap halaman dirancang agar mampu menampilkan informasi yang relevan sesuai kebutuhan pengguna dalam mengelola sistem audio multi-zona.

a. Halaman *Login*

Halaman *Login* merupakan halaman awal yang ditampilkan ketika aplikasi dijalankan. Halaman ini berfungsi untuk melakukan autentikasi pengguna sebelum mengakses fitur *monitoring* dan kontrol sistem.

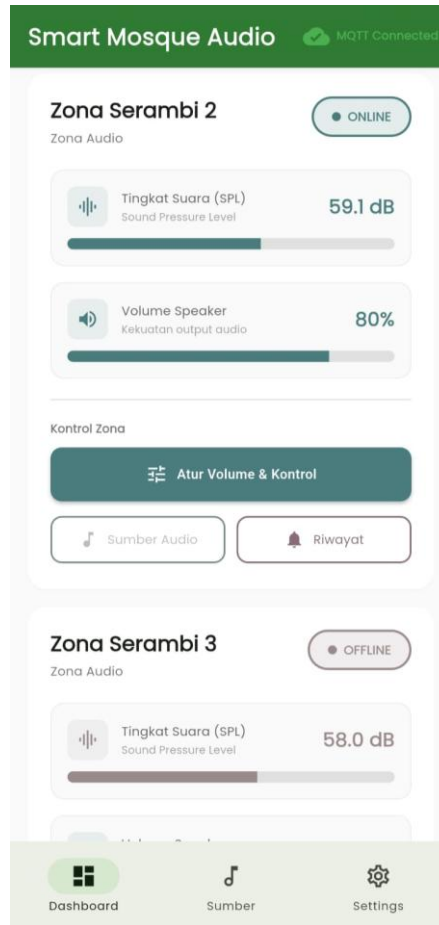
Pada halaman ini, pengguna diminta memasukkan informasi akun yang telah terdaftar pada sistem. Setelah proses autentikasi berhasil, pengguna akan diarahkan menuju halaman dashboard aplikasi. Berikut tampilan *Login* aplikasi yang ditunjukkan oleh Gambar 4.3.



Gambar 4.3 Halaman *Login* Aplikasi.

b. Halaman *Monitoring*

Halaman dashboard berfungsi sebagai pusat *monitoring* sistem audio multi-zona. Informasi yang ditampilkan meliputi status konektivitas *node*, kondisi perangkat, serta nilai *Sound Pressure Level* (SPL) yang diterima secara *real-time* dari *backend system*. Halaman dashboar *monitoring* ditunjukkan oleh Gambar 4.4.



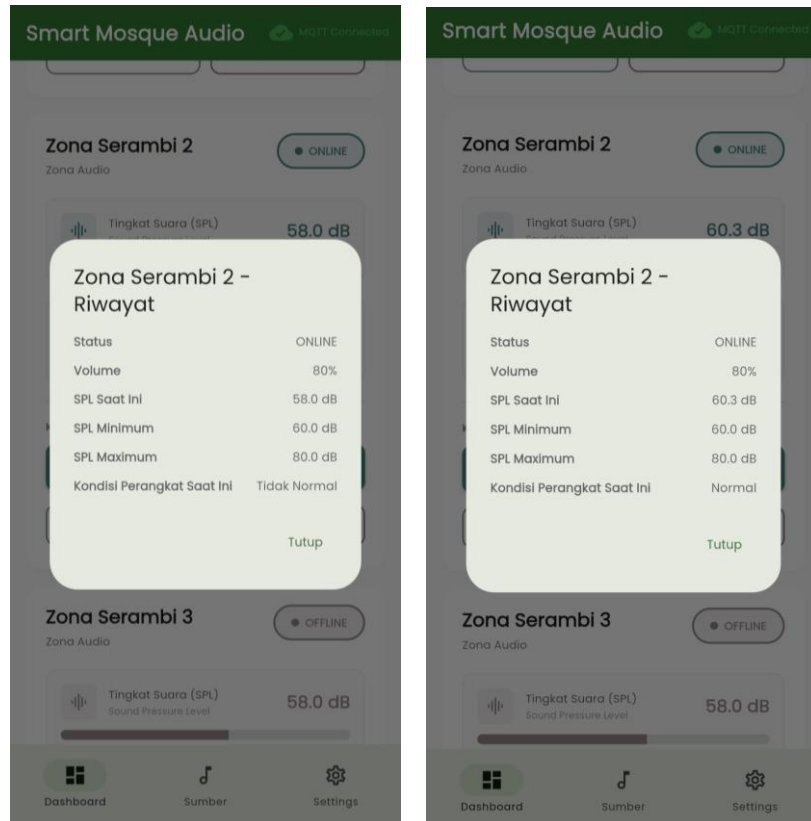
Gambar 4.4 Dashboard *Monitoring* Aplikasi

Informasi tersebut ditampilkan dalam bentuk indikator visual sehingga pengguna dapat dengan mudah mengetahui kondisi masing-masing zona tanpa harus melakukan pemeriksaan secara langsung ke lokasi perangkat.

c. Halaman Detail Zona

Halaman detail zona digunakan untuk menampilkan informasi yang lebih rinci mengenai setiap zona audio yang tersedia pada sistem. Informasi yang ditampilkan meliputi identitas zona, status perangkat, nilai SPL terkini, dan kondisi operasional sistem.

Melalui halaman ini pengguna dapat melakukan pemantauan kondisi setiap zona secara lebih spesifik dibandingkan tampilan ringkasan pada dashboard utama. Berikut gambaran kondisi perangkat ketika tidak normal ditunjukkan pada Gambar 4.5.

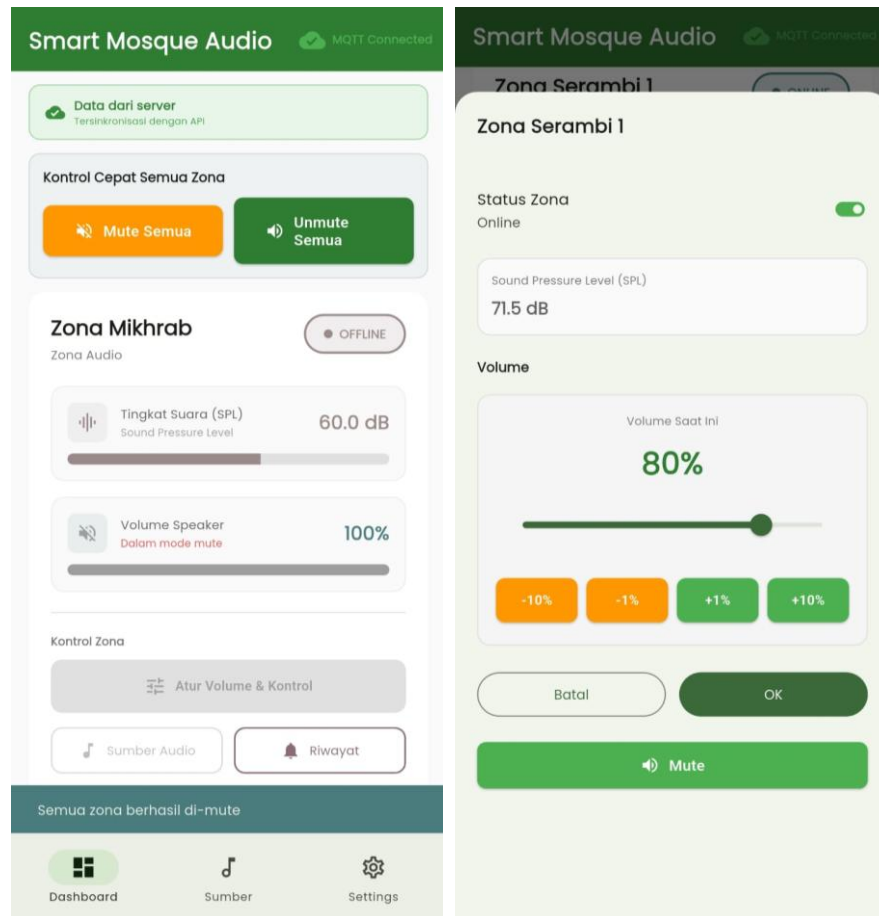


Gambar 4.5 Halaman Detail Zona

d. Halaman Kontrol Volume dan *Mute*

Halaman kontrol digunakan untuk mengirimkan instruksi kontrol kepada sistem melalui MQTT broker. Fitur yang tersedia meliputi pengaturan nilai volume menggunakan komponen slider serta pengaturan status *mute* menggunakan tombol kontrol.

Ketika pengguna melakukan perubahan nilai volume atau status *mute*, aplikasi akan mengirimkan pesan MQTT menuju topik yang telah ditentukan. Selanjutnya pesan tersebut diterima oleh *backend system* dan diproses sesuai mekanisme yang telah dirancang. Berikut tampilan aplikasi ketika pengguna menggunakan *mute* dan tampilan ketika mau kontrol volume diperlihatkan pada Gambar 4.6.

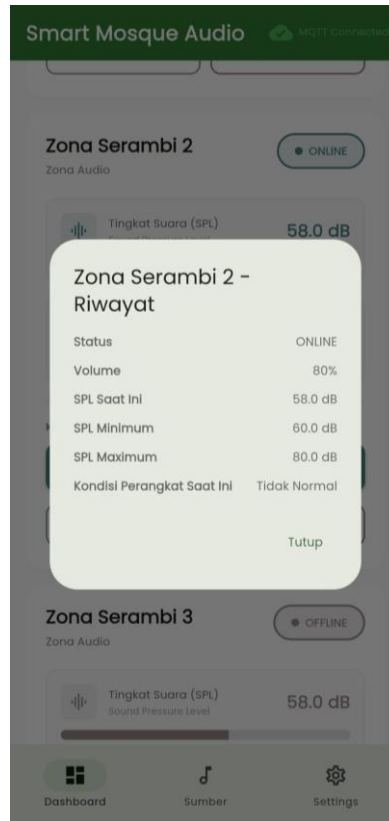


Gambar 4.6 Kontrol Volume dan *Mute*

e. Halaman Notifikasi

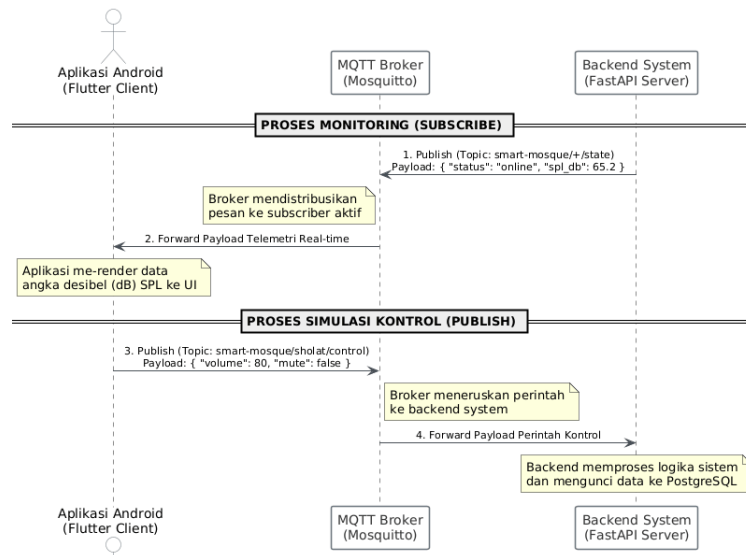
Halaman notifikasi berfungsi untuk memberikan informasi terkait kondisi operasional perangkat pada sistem audio multi-zona. Notifikasi ditampilkan ketika terjadi perubahan status perangkat, baik dalam kondisi normal maupun ketika terdeteksi kondisi yang memerlukan perhatian pengguna.

Informasi yang ditampilkan pada halaman ini membantu pengguna dalam melakukan pemantauan kondisi sistem secara lebih cepat tanpa harus membuka halaman detail setiap zona. Ketika kondisi perangkat tidak normal berarti SPL berada di atas atau di bawah ambang ketentuan dB yang sebelumnya ditentukan, begitupun sebaliknya. Pada tampilan ini ditunjukkan oleh Gambar 4.7.



Gambar 4.7 Tampilan Halaman Notifikasi Perangkat Tidak Normal

3. Implementasi Komunikasi MQTT



Gambar 4.8 Alur Komunikasi MQTT yang diterapkan pada Sistem.

Implementasi komunikasi data pada penelitian ini menggunakan protokol MQTT dengan mekanisme *publish-subscribe*. MQTT broker berfungsi sebagai

perantara komunikasi antara aplikasi Android dan *backend system* sehingga pertukaran data dapat berlangsung secara *real-time*.

Lihat Gambar 4.8 pada proses monitoring, *backend system* mengirimkan data status *node* dan nilai SPL melalui topik MQTT tertentu yang kemudian diterima aplikasi melalui mekanisme *subscribe*. Sebaliknya, pada proses kontrol, aplikasi mengirimkan instruksi volume dan *mute* melalui mekanisme *publish* menuju topik yang telah ditentukan.

Penggunaan MQTT pada penelitian ini dipilih karena memiliki karakteristik ringan, efisien, dan mendukung komunikasi data secara *real-time* sehingga sesuai dengan kebutuhan sistem *monitoring* dan kontrol berbasis IoT.

4. Pengujian Koneksi MQTT

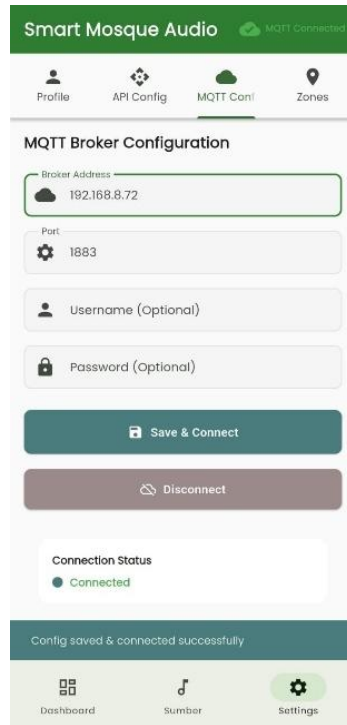
Pengujian koneksi MQTT dilakukan untuk memastikan aplikasi Android dapat terhubung dengan MQTT broker dan mampu melakukan komunikasi data menggunakan mekanisme *publish-subscribe*. Pengujian ini bertujuan untuk memverifikasi bahwa proses pengiriman data kontrol dari aplikasi serta penerimaan data *monitoring* dari *backend system* dapat berjalan dengan baik. Pada pengujian Koneksi MQTT ada beberapa hal yang diuji antara lain :

- a. Aplikasi melakukan koneksi ke MQTT broker

```
class MqttService {  
    static const String defaultBrokerAddress = '192.168.8.72';  
    static const int defaultBrokerPort = 1883;  
}
```

Gambar 4.9 *Code Connect* ke Alamat Broker

Seperti yang ditampilkan pada gambar 4.9 bahwasannya aplikasi sudah dihubungkan dengan IP Broker “192.168.8.72” dengan port “1883” dan hal ini dibuktikan dengan Gambar 4.10 yaitu tampilan aplikasi yang sudah terhubung langsung ke MQTT broker, sehingga komunikasi data yang digunakan pada aplikasi terjadi secara *real-time*.



Gambar 4.10 Tampilan Aplikasi Terhubung ke MQTT broker

- b. Aplikasi melakukan *subscribe* topik monitoring

```
class MqttService {
    static const String defaultBrokerAddress = '192.168.8.72';
    static const int defaultBrokerPort = 1883;

    static const String baseTopicSuffix = 'smart-mosque/app/spl';
    String _baseTopicSuffix = baseTopicSuffix;
}
```

Gambar 4.11 Code Subscribe ke Topik

Aplikasi telah terhubung ke MQTT broker kemudian menghubungkan ke topik dasar pada aplikasi, seperti yang ditunjukkan pada Gambar 4.11 dan melakukan *subscribe* pada topik *monitoring* utama beserta seluruh subtopiknya, sehingga setiap perubahan data seperti SPL, status, volume, *mute/unmute* dan alarm dapat diterima secara *real-time*. Dengan mekanisme ini, aplikasi dapat memantau kondisi sistem secara terus-menerus tanpa perlu melakukan permintaan data secara manual.

- c. Aplikasi melakukan *publish* data kontrol volume

```
smart_mosque_audio=> SELECT * FROM control_actions;
id | zone_id | action_type | gain_db | timestamp
-----+-----+-----+-----+-----
1 | 2 | GAIN_ADJUST | 12 | 2026-06-23 14:51:13.811821+07
2 | 2 | GAIN_ADJUST | -20 | 2026-06-23 14:53:05.28849+07
3 | 2 | GAIN_ADJUST | 0 | 2026-06-23 14:53:37.962829+07
```

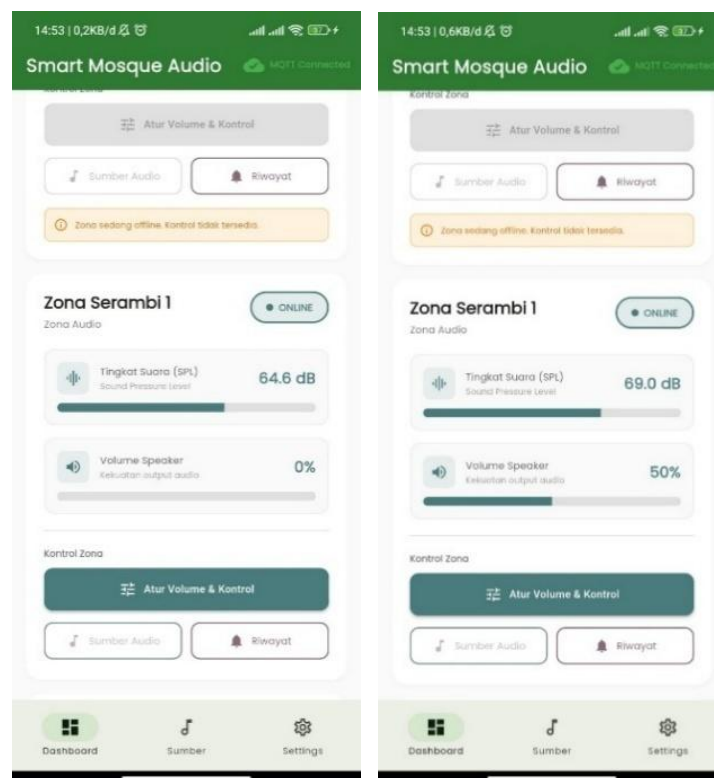
Gambar 4.12 Validasi Kontrol Volume

Pada pengujian ini dilakukan verifikasi terhadap proses pengiriman data kontrol volume dari aplikasi Android menuju *backend system*. Pengguna melakukan perubahan nilai volume melalui slider pada aplikasi seperti

yang ditunjukkan pada Gambar 4.12 kemudian sistem mengonversi nilai tersebut menjadi parameter kontrol yang selanjutnya dikirim melalui MQTT dan disimpan pada database PostgreSQL.

Hasil pengujian pada Gambar 4.12 menunjukkan bahwa setiap perubahan nilai volume yang dilakukan pada aplikasi berhasil diterima dan dicatat oleh *backend system*. Data yang tersimpan pada tabel *control_actions* menunjukkan kesesuaian antara perintah yang dikirim dari aplikasi dengan data yang diterima oleh *backend system*.

Berdasarkan hasil tersebut, dapat disimpulkan bahwa mekanisme pengiriman data kontrol volume berjalan dengan baik, di mana setiap perubahan nilai pada antarmuka aplikasi berhasil diproses dan tersinkronisasi ke database secara *real-time* sesuai dengan perancangan sistem hal.



Gambar 4.13 Tampilan Aplikasi *Mute* Zona

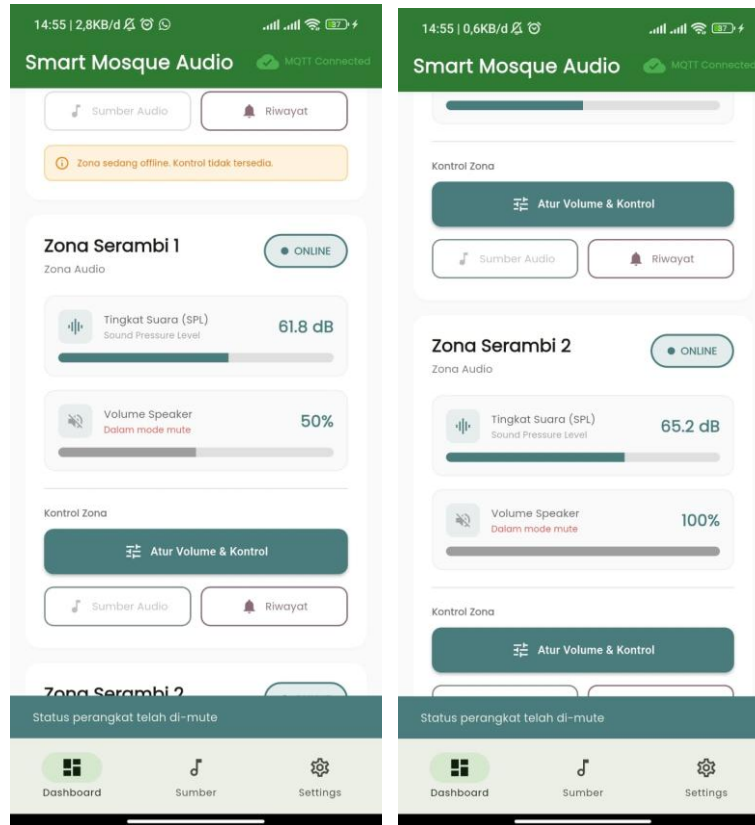
d. Aplikasi melakukan *publish* data *mute*

4	3	MUTE	0	2026-06-23 14:55:04.469283+07
5	2	MUTE	0	2026-06-23 14:55:30.059659+07

Gambar 4.14 Validasi Kontrol *Mute*

Pada pengujian kali ini peneliti melakukan pengujian terhadap 2 zona yang aktif. Pada pengujian pertama terhadap zona 2 yaitu zona serambi 1 yang ditunjukkan pada Gambar 4.15 sebelah kiri dan zona 3 yaitu zona serambi

2 yang ditunjukkan pada Gambar 4.15 sebelah kanan. Kedua zona tersebut sudah membuktikan bahwa kedua zona sudah sukses melakukan *mute* yang ditunjukkan pada tampilan di UI aplikasi yaitu “Status perangkat telah di-*mute*”. Hal ini diperkuat atau dibuktikan dengan tampilan pada Gambar 4.14.



Gambar 4.15 Tampilan Aplikasi *Mute* Zona

Pada pengujian ini dilakukan beberapa skenario, yaitu pengujian koneksi aplikasi ke broker, pengujian *subscribe* data *monitoring*, dan pengujian *publish* data kontrol yang sudah dibuktikan pada masing-masing pengujian yang kali ini dirangkum pada sebuah tabel pengujian koneksi MQTT. Berikut hasil pengujian ditunjukkan pada Tabel 4.1.

Tabel 4.1 Hasil Pengujian Koneksi MQTT

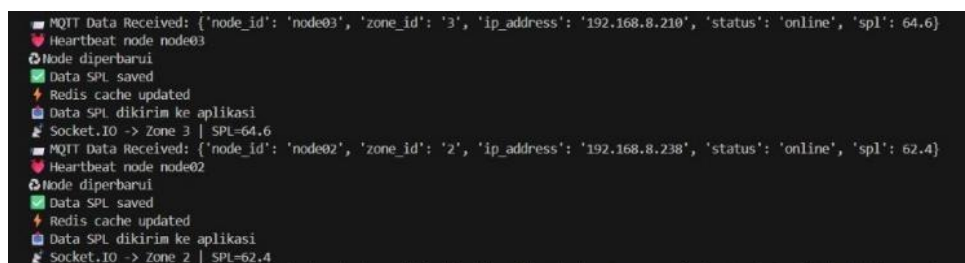
No	Skenario Pengujian	Hasil yang Diharapkan	Hasil Pengujian	Status
1	Aplikasi melakukan koneksi ke MQTT broker	Status koneksi berhasil terhubung	Aplikasi berhasil terhubung ke broker	Berhasil
2	Aplikasi melakukan <i>subscribe</i> topik monitoring	Data <i>monitoring</i> diterima aplikasi	Data <i>monitoring</i> berhasil diterima	Berhasil
3	Aplikasi melakukan <i>publish</i> data kontrol volume	Data diterima MQTT broker	Data berhasil diterima broker	Berhasil
4	Aplikasi melakukan <i>publish</i> data <i>mute</i>	Status <i>mute</i> diterima broker	Data berhasil diterima broker	Berhasil

Berdasarkan hasil pengujian pada Tabel 4.1, seluruh proses komunikasi antara aplikasi Android dan MQTT broker berjalan dengan baik. Aplikasi berhasil melakukan koneksi ke broker, menerima data *monitoring* melalui mekanisme *subscribe*, serta mengirimkan data kontrol melalui mekanisme *publish* tanpa ditemukan kegagalan komunikasi.

5. Pengujian Notifikasi dan Integrasi *Backend*

Pengujian notifikasi dan integrasi *backend* dilakukan untuk memastikan data yang dikirimkan oleh *backend system* dapat diterima dan ditampilkan oleh aplikasi secara *real-time*. Selain itu, pengujian ini juga bertujuan untuk memverifikasi bahwa fitur notifikasi mampu memberikan informasi kondisi perangkat berdasarkan status yang diterima dari *backend system*.

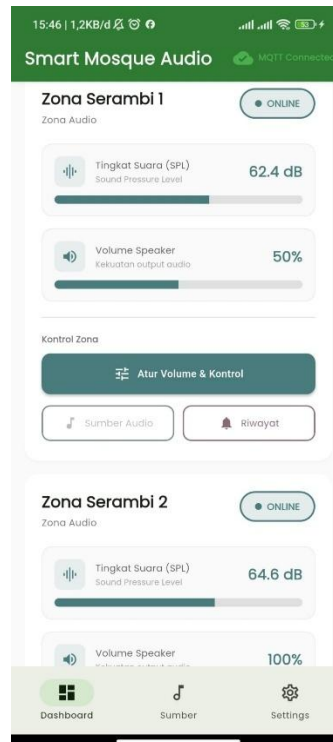
a. Pengujian Sinkronisasi Data *Monitoring*



Gambar 4.16 Tampilan *Monitoring* BE

Pada gambar 4.16 itu tampilan *monitoring* BE dari hasil pengujian dimana aplikasi UI sudah sinkron dengan *Backend system* yang mana untuk status

dan nilai SPL yang dikontrol simulasikan pada tampilan aplikasi seperti yg ditunjukkan pada Gambar 4.0



Gambar 4.17 Tampilan Status dan Zona SPL pada Aplikasi

Setelah pengujian dilakukan dengan membandingkan data yang tersimpan pada *backend system* dengan data yang ditampilkan pada aplikasi Android.

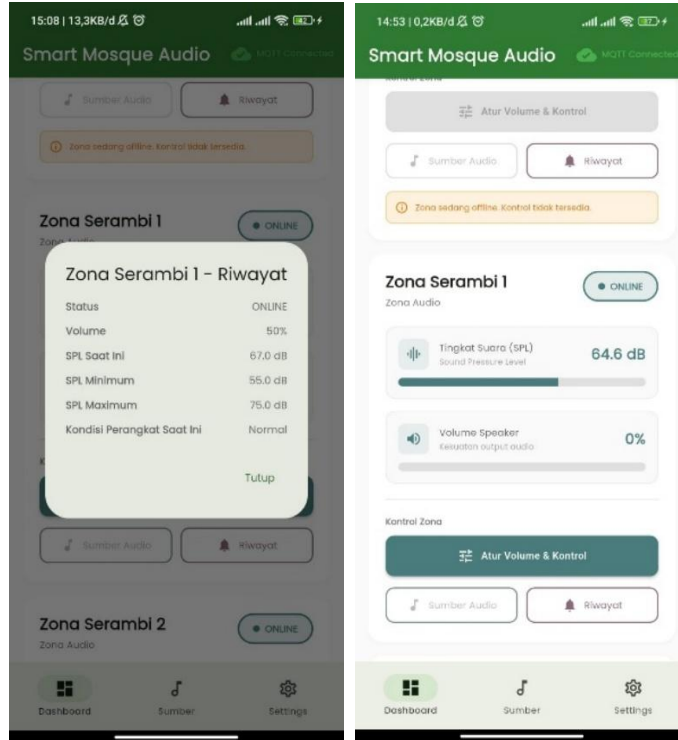
Tabel 4.2 Pengujian Sinkronisasi Data *Monitoring*

No	Parameter	Data Backend	Data Aplikasi	Status
1	Status <i>Node</i> Zona 2	<i>Online</i>	<i>Online</i>	Sesuai
2	Status <i>Node</i> Zona 3	<i>Online</i>	<i>Online</i>	Sesuai
3	Nilai SPL Zona Serambi 1	62.4 dB	62.4 dB	Sesuai
4	Nilai SPL Zona Serambi 2	64.6 dB	64.6dB	Sesuai

Berdasarkan hasil pengujian, seluruh data *monitoring* yang dikirimkan *backend* berhasil diterima dan ditampilkan oleh aplikasi secara akurat tanpa adanya perbedaan nilai.

b. Pengujian Notifikasi Kondisi Perangkat

Pengujian notifikasi dilakukan untuk memastikan aplikasi mampu memberikan informasi kondisi perangkat berdasarkan status yang diterima dari *backend system* .



Gambar 4.18 Tampilan Kondisi Perangkat dan Status *Node*

Pada Gambar 4.18 ditunjukkan bahwa kondisi perangkat ketika SPL di bawah atau melebihi batas yang ditentukan oleh pengguna menunjukkan notifikasi kondisi perangkat tidak normal dan juga status *node* juga sesuai pada tampilan aplikasi. Berikut tabel hasil pengujiannya :

Tabel 4.3 Pengujian Notifikasi Kondisi Perangkat

No	Kondisi Sistem	Notifikasi yang Diharapkan	Hasil Pengujian	Status
1	<i>Node</i> aktif	Status <i>Online</i>	Tampil	Berhasil
2	<i>Node offline</i>	Peringatan <i>Node Offline</i>	Tampil	Berhasil
3	SPL normal	Kondisi Normal	Tampil	Berhasil
4	SPL melebihi batas	Peringatan Kondisi Tidak Normal	Tampil	Berhasil

Hasil pengujian menunjukkan bahwa aplikasi mampu menampilkan notifikasi sesuai kondisi perangkat yang diterima dari *backend system*

sehingga memudahkan pengguna dalam melakukan *monitoring* kondisi sistem audio multi-zona.

6. Pengujian Fungsionalitas (*Black-box Testing*)

Pengujian *Black-box Testing* dilakukan untuk mengevaluasi kesesuaian fungsi aplikasi terhadap kebutuhan sistem tanpa melihat struktur kode program. Pengujian difokuskan pada seluruh fitur utama aplikasi yang digunakan oleh pengguna.

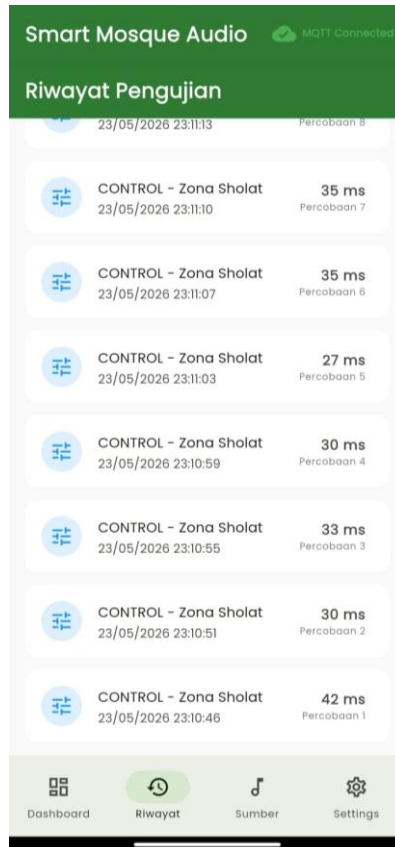
Tabel 4.4 Hasil Pengujian *Black-box Testing*

No	Deskripsi Singkat	Langkah Utama (Ringkas)	Hasil yang Diharapkan	Hasil Uji (Pass/Fail)
1	<i>Login</i>	Memasukan Akun Valid	Masuk Ke Dashboard	<i>Pass</i>
2	Koneksi ke MQTT broker	Isi konfigurasi broker, tekan <i>Connect</i>	Status <i>Connected</i> , tidak ada <i>error</i>	<i>Pass</i>
3	<i>Monitoring</i> status <i>node</i> dan SPL	Amati Dashboard/ <i>Nodes</i> saat <i>node</i> aktif	<i>Online/offline</i> , SPL, suhu, baterai tampil dan ter-update	<i>Pass</i>
4	Kontrol volume per zona	Ubah slider volume satu zona	Level suara di zona berubah sesuai slider	<i>Pass</i>
5	Kontrol <i>mute/unmute</i> per zona	Tekan tombol <i>mute</i> lalu <i>unmute</i> untuk satu zona	Audio di zona berhenti dan kembali normal	<i>Pass</i>
6	Master <i>mute</i> semua zona	Tekan tombol master <i>mute</i>	Perintah terkirim dan diterima oleh <i>Backend</i>	<i>Pass</i>
7	Fitur Sistem Kondisi Tidak Normal	Simulasikan kondisi abnormal (mis. <i>node offline</i> /SPL tinggi)	Tampilan muncul sesuai skenario	<i>Pass</i>
8	Auto- <i>reconnect</i>	Putuskan WiFi saat terhubung, nyalakan kembali	Aplikasi <i>reconnect</i> otomatis, status kembali ter-update	<i>Pass</i>
9	Sinkronisasi data setelah <i>reconnect</i>	Setelah <i>reconnect</i> , amati status <i>node</i> dan histori	Data status dan histori konsisten dengan kondisi terbaru	<i>Pass</i>

Berdasarkan hasil pengujian pada Tabel 4.4, seluruh fitur aplikasi berhasil berjalan sesuai dengan kebutuhan fungsional yang telah ditetapkan. Tidak ditemukan kegagalan fungsi selama proses pengujian sehingga aplikasi dinyatakan memenuhi kebutuhan sistem.

7. Pengujian *Response time* MQTT

Pengukuran *response time* dilakukan menggunakan mekanisme pencatatan timestamp pada sisi aplikasi Android dan *backend system*. Timestamp pertama dicatat ketika pengguna mengirimkan instruksi kontrol melalui aplikasi, sedangkan timestamp kedua dicatat ketika pesan berhasil diterima dan diproses oleh *backend system*. Berikut hasil pengujian *Response time* ditunjukkan pada Gambar 4.19.



Gambar 4.19 Pengujian *Response time*

Nilai *response time* dihitung sebagai selisih antara kedua timestamp tersebut menggunakan Persamaan persamaan berikut :

$$\Delta t = T_{exec} - T_{user}$$

Keterangan:

- a. Δt = *Response time*
- b. T_{user} = waktu saat pesan dikirim aplikasi
- c. T_{exec} = waktu saat pesan diterima *backend*

Metode ini digunakan untuk mengukur end-to-end latency komunikasi data pada tingkat aplikasi (*application-level latency*), sehingga mampu merepresentasikan waktu respon sistem yang dirasakan pengguna secara langsung. Berikut hasil tabel pengujiannya ditunjukkan pada Tabel 4.5.

Tabel 4.5 Pengujian *Response Time* MQTT

Zona	Percobaan ke-	t_{user} (ms)	t_{exec} (ms)	Δt (ms)	Catatan
Z1	1	0	42	42	Data Z1
Z1	2	0	30	30	Data Z1
Z1	3	0	33	33	Data Z1
Z1	4	0	30	30	Data Z1
Z1	5	0	27	27	Data Z1
Z1	6	0	35	35	Data Z1
Z1	7	0	35	35	Data Z1
Z1	8	0	30	30	Data Z1
Z2	9	0	35	35	Data Z2
Z3	10	0	42	42	Data Z3
	Rata-rata			33,9	Target ≤ 100 ms

Berdasarkan hasil pengujian pada Tabel 4.5 rata-rata *response time* komunikasi MQTT sebesar 33,9 ms. Nilai tersebut menunjukkan bahwa komunikasi data antara aplikasi Android dan *backend system* berlangsung secara responsif dan mampu mendukung kebutuhan *monitoring* serta kontrol secara *real-time*.

B. Pembahasan

1. Pembahasan Implementasi Komunikasi MQTT

Berdasarkan hasil pengujian koneksi MQTT pada Tabel 4.1, aplikasi Android berhasil melakukan koneksi ke MQTT broker, menerima data *monitoring* melalui mekanisme *subscribe*, serta mengirimkan data kontrol volume dan *mute* melalui mekanisme *publish*. Hasil ini menunjukkan bahwa implementasi protokol MQTT pada aplikasi telah berjalan sesuai dengan rancangan sistem yang menggunakan pola komunikasi *publish-subscribe*.

Keberhasilan proses komunikasi tersebut menunjukkan bahwa aplikasi mampu menjalankan fungsi sebagai antarmuka *monitoring* dan kontrol sistem audio multi-zona sesuai dengan rancangan yang telah ditetapkan. Penggunaan MQTT juga mendukung kebutuhan pertukaran data secara *real-time* karena setiap perubahan data dapat langsung diteruskan melalui broker menuju pihak yang berlangganan (*subscriber*). Hasil ini sejalan dengan penelitian Saiqul Umam et al. (2023) yang menunjukkan bahwa MQTT mampu mendukung komunikasi data yang stabil dan akurat pada sistem berbasis *Internet of Things*.

Selain itu, keberhasilan komunikasi data antara aplikasi Android dan *backend system* menunjukkan bahwa integrasi perangkat lunak yang dibangun telah mampu mendukung arsitektur *Cyber-Physical System* (CPS) yang digunakan pada sistem audio multi-zona masjid. Dengan demikian, hasil

tersebut menunjukkan bahwa integrasi aplikasi Android dengan *backend system* melalui MQTT telah berhasil diimplementasikan sehingga aplikasi mampu mendukung proses *monitoring* dan kontrol secara *real-time* sesuai ruang lingkup penelitian.

2. Pembahasan Integrasi *Backend* dan Fitur Monitoring

Hasil pengujian sinkronisasi data *monitoring* pada Tabel 4.2 menunjukkan bahwa seluruh data yang dikirimkan oleh *backend system* berhasil ditampilkan pada aplikasi tanpa adanya perbedaan nilai. Status *node* dan nilai *Sound Pressure Level* (SPL) yang ditampilkan pada aplikasi sesuai dengan data yang tersimpan pada *backend system*.

Keberhasilan sinkronisasi tersebut menunjukkan bahwa mekanisme pertukaran data antara *backend system*, MQTT broker, dan aplikasi Android berjalan secara konsisten. Kondisi ini sangat penting karena fungsi utama aplikasi pada penelitian ini adalah melakukan *monitoring* kondisi sistem audio multi-zona secara *real-time*.

Pada pengujian notifikasi kondisi perangkat, aplikasi juga berhasil menampilkan informasi kondisi normal maupun kondisi abnormal sesuai dengan status yang diterima dari *backend system*. Fitur ini memberikan kemudahan bagi pengelola masjid dalam memantau kondisi perangkat tanpa harus melakukan pengecekan langsung terhadap setiap *node* yang terpasang pada sistem audio.

Dengan adanya visualisasi status *node* dan notifikasi kondisi sistem, aplikasi mampu mendukung proses *monitoring* kondisi perangkat secara *real-time* sehingga memudahkan pengelola dalam memperoleh informasi kondisi sistem.

3. Pembahasan Fungsionalitas Aplikasi

Hasil pengujian *Black-box Testing* pada Tabel 4.4 menunjukkan bahwa seluruh fitur utama aplikasi berhasil dijalankan sesuai dengan spesifikasi kebutuhan sistem. Seluruh skenario pengujian memperoleh hasil *pass* tanpa ditemukan kegagalan fungsi.

Fitur-fitur yang diuji meliputi proses *Login*, koneksi MQTT, *monitoring* status *node*, *monitoring* nilai SPL, kontrol volume, kontrol *mute*, master *mute*, notifikasi kondisi abnormal, auto *reconnect*, serta sinkronisasi data setelah proses *reconnect*. Keberhasilan seluruh skenario pengujian menunjukkan

bahwa implementasi aplikasi telah memenuhi kebutuhan fungsional yang dirancang pada tahap analisis dan perancangan sistem.

Selain itu, keberhasilan fitur auto *reconnect* menunjukkan bahwa aplikasi mampu mempertahankan kontinuitas layanan ketika terjadi gangguan jaringan sementara. Kemampuan tersebut menjadi penting karena aplikasi dirancang untuk mendukung sistem *monitoring* dan kontrol yang membutuhkan komunikasi data secara berkelanjutan.

Secara keseluruhan, hasil *Black-box Testing* menunjukkan bahwa seluruh fungsi yang dirancang pada penelitian ini telah berjalan sesuai kebutuhan fungsional. Namun, pengujian ini hanya mengevaluasi aspek fungsionalitas aplikasi dan belum mencakup aspek *usability*, pengalaman pengguna (*user experience*), maupun tingkat penerimaan pengguna terhadap aplikasi.

4. Pembahasan Performa *Response time* MQTT

Pengujian *response time* dilakukan untuk mengetahui tingkat responsivitas komunikasi data antara aplikasi Android dan *backend system* melalui MQTT broker. Berdasarkan hasil pengujian pada Tabel 4.5 diperoleh rata-rata *response time* sebesar 33,9 ms.

Nilai tersebut berada jauh di bawah target performa yang ditetapkan yaitu ≤ 100 ms. Hasil ini menunjukkan bahwa sistem mampu mengirimkan dan memproses data kontrol dengan sangat cepat sehingga pengguna dapat memperoleh respons sistem hampir secara langsung setelah melakukan interaksi pada aplikasi.

Performa komunikasi yang diperoleh juga menunjukkan bahwa penggunaan MQTT sangat sesuai untuk kebutuhan *monitoring* dan kontrol sistem audio multi-zona yang membutuhkan pertukaran data secara *real-time*. Semakin kecil nilai *response time* yang diperoleh, maka semakin baik kemampuan sistem dalam memberikan respons terhadap perubahan kondisi maupun instruksi pengguna.

Hasil penelitian ini juga mendukung penelitian Rosad, Irkhamudin, et al. (2025) yang menunjukkan bahwa optimasi komunikasi pada lingkungan *Smart Mosque* mampu menghasilkan latensi komunikasi yang rendah. Rata-rata *response time* sebesar 33,9 ms menunjukkan bahwa komunikasi data berbasis MQTT pada aplikasi berlangsung secara responsif dan mampu mendukung proses *monitoring* serta kontrol secara *real-time* sesuai dengan ruang lingkup penelitian. Hasil pengujian ini menunjukkan performa komunikasi data yang baik pada aspek *response time*, namun tidak digunakan untuk mengevaluasi

aspek *usability*, efektivitas penggunaan, maupun tingkat penerimaan pengguna terhadap aplikasi.

5. Pembahasan Hasil

a. Pembahasan Perancangan UI/UX Aplikasi Android

Rumusan masalah pertama membahas bagaimana mengembangkan aplikasi *mobile* Android berbasis Flutter yang mampu menampilkan data *monitoring* serta menyediakan fitur kontrol volume dan *mute* secara fungsional.

Berdasarkan hasil implementasi yang telah dilakukan, aplikasi berhasil dikembangkan menggunakan *framework* Flutter dengan arsitektur *Model-View-ViewModel* (MVVM). Aplikasi menyediakan berbagai fitur utama, meliputi proses autentikasi pengguna, *monitoring* status *node* (*online/offline*), *monitoring* nilai *Sound Pressure Level* (SPL), kontrol volume, kontrol *mute*, master *mute*, notifikasi kondisi perangkat, serta mekanisme auto *reconnect* ketika koneksi MQTT terputus.

Hasil *Black-box Testing* pada Tabel 4.4 menunjukkan bahwa seluruh fitur utama aplikasi memperoleh status *pass*, sehingga seluruh fungsi yang dirancang dapat berjalan sesuai dengan kebutuhan fungsional penelitian. Selain itu, aplikasi mampu menampilkan data *monitoring* secara *real-time* berdasarkan data yang diterima dari *backend system* serta menyediakan fitur kontrol volume dan *mute* yang terintegrasi dengan komunikasi MQTT.

Pengujian yang dilakukan pada penelitian ini berfokus pada validasi fungsionalitas aplikasi, sehingga belum mencakup evaluasi terhadap aspek *usability*, pengalaman pengguna (*user experience*), maupun tingkat penerimaan pengguna (*user acceptance*).

Dengan demikian, rumusan masalah pertama berhasil terjawab karena aplikasi *mobile* Android berbasis Flutter berhasil dikembangkan dan mampu menyediakan fungsi *monitoring* serta kontrol sesuai dengan kebutuhan fungsional penelitian.

b. Pembahasan Implementasi Komunikasi MQTT

Rumusan masalah kedua membahas bagaimana mengimplementasikan mekanisme komunikasi data berbasis protokol

MQTT agar aplikasi Android dapat berkomunikasi dengan *backend system* secara *real-time* dalam proses *monitoring* dan kontrol.

Hasil pengujian koneksi MQTT pada Tabel 4.1 menunjukkan bahwa aplikasi berhasil melakukan koneksi ke MQTT broker, melakukan proses subscribe terhadap data monitoring, serta melakukan proses publish terhadap data kontrol volume dan *mute* tanpa mengalami kegagalan komunikasi. Selain itu, hasil pengujian sinkronisasi data *monitoring* pada Tabel 4.2 menunjukkan bahwa data yang diterima aplikasi sesuai dengan data yang tersedia pada *backend system*.

Implementasi MQTT menggunakan mekanisme *publish-subscribe* memungkinkan proses pertukaran data berlangsung secara dua arah antara aplikasi Android dan *backend system*. *Backend system* mengirimkan data *monitoring* melalui topik MQTT yang kemudian diterima aplikasi melalui proses subscribe, sedangkan aplikasi mengirimkan instruksi kontrol melalui proses publish menuju MQTT broker untuk diproses oleh *backend system*.

Hasil pengujian tersebut menunjukkan bahwa mekanisme komunikasi MQTT berhasil diimplementasikan dan mampu mendukung proses *monitoring* serta kontrol secara *real-time* sesuai dengan ruang lingkup penelitian.

Dengan demikian, rumusan masalah kedua berhasil terjawab karena implementasi komunikasi data berbasis MQTT mampu mendukung pertukaran data antara aplikasi Android dan *backend system* secara *real-time* sesuai dengan kebutuhan penelitian.

c. Pembahasan Evaluasi Kualitas Aplikasi

Rumusan masalah ketiga membahas bagaimana mengevaluasi kualitas aplikasi melalui pengujian fungsionalitas serta pengukuran *response time* komunikasi MQTT.

Berdasarkan hasil *Black-box Testing* pada Tabel 4.4, seluruh fitur aplikasi memperoleh hasil *pass* tanpa ditemukan kegagalan fungsi pada proses *Login*, *monitoring* data, kontrol volume, kontrol *mute*, master *mute*, notifikasi kondisi perangkat, maupun mekanisme auto *reconnect*. Hasil tersebut menunjukkan bahwa seluruh fungsi aplikasi telah berjalan sesuai dengan kebutuhan fungsional yang ditetapkan pada tahap perancangan.

Selain pengujian fungsionalitas, evaluasi performa komunikasi dilakukan melalui pengukuran *response time* MQTT sebagaimana ditunjukkan pada Tabel 4.5. Hasil pengujian menunjukkan rata-rata *response time* sebesar 33,9 ms, dengan nilai tertinggi 42 ms dan nilai terendah 27 ms. Nilai tersebut berada di bawah target performa yang ditetapkan, yaitu ≤ 100 ms. Hasil ini menunjukkan bahwa komunikasi data berbasis MQTT pada aplikasi berlangsung secara responsif sehingga mampu mendukung proses *monitoring* dan kontrol secara *real-time* sesuai dengan ruang lingkup penelitian.

Hasil pengujian pada penelitian ini menunjukkan bahwa aplikasi telah memenuhi kebutuhan fungsionalitas serta memiliki performa komunikasi data yang responsif berdasarkan hasil *Black-box Testing* dan pengukuran *response time* MQTT. Namun demikian, pengujian yang dilakukan terbatas pada evaluasi fungsionalitas aplikasi dan performa komunikasi data, sehingga belum mencakup evaluasi terhadap aspek *usability*, efektivitas penggunaan, maupun tingkat penerimaan pengguna terhadap aplikasi. Oleh karena itu, kualitas aplikasi secara menyeluruh masih memerlukan evaluasi lebih lanjut pada penelitian berikutnya.

Dengan demikian, rumusan masalah ketiga berhasil terjawab karena aplikasi telah memenuhi kebutuhan fungsional dan menunjukkan performa komunikasi data yang responsif berdasarkan hasil *Black-box Testing* dan pengukuran *response time* MQTT sesuai dengan ruang lingkup penelitian.

Secara keseluruhan, hasil implementasi dan pengujian menunjukkan bahwa penelitian ini telah mencapai seluruh tujuan penelitian yang dirumuskan pada Bab I sesuai dengan ruang lingkup dan batasan penelitian yang telah ditetapkan. Meskipun demikian, capaian penelitian ini masih terbatas pada evaluasi aspek fungsionalitas aplikasi dan performa komunikasi data. Evaluasi terhadap aspek *usability*, efektivitas penggunaan, dan tingkat penerimaan pengguna belum dilakukan sehingga menjadi peluang untuk penelitian selanjutnya.

BAB V KESIMPULAN

A. Kesimpulan

Berdasarkan hasil penelitian yang telah dilakukan mengenai aplikasi *mobile* Android berbasis MQTT untuk *monitoring* dan kontrol sistem audio multi-zona masjid, diperoleh kesimpulan sebagai berikut:

1. Aplikasi Android berhasil dirancang dan diimplementasikan menggunakan *framework* Flutter dengan pola arsitektur MVVM (*Model-View-ViewModel*). Aplikasi mampu menampilkan informasi *monitoring* berupa status *node* (*online/offline*) dan nilai *Sound Pressure Level* (SPL) secara *real-time*. Selain itu, aplikasi juga berhasil menyediakan fitur kontrol volume dan *mute* yang dapat digunakan sesuai kebutuhan sistem.
2. Mekanisme komunikasi data berbasis protokol MQTT berhasil diimplementasikan menggunakan pola *publish-subscribe* sehingga proses pertukaran data *monitoring* dan pengiriman instruksi kontrol antara aplikasi Android dan *backend system* dapat berlangsung secara *real-time*. Hasil pengujian integrasi menunjukkan bahwa aplikasi mampu melakukan koneksi ke MQTT broker, menerima data *monitoring* melalui proses *subscribe*, serta mengirimkan instruksi kontrol volume dan *mute* melalui proses *publish* tanpa ditemukan kegagalan komunikasi.
3. Evaluasi aplikasi dilakukan melalui *Black-box Testing* dan pengujian performa *response time* MQTT. Hasil *Black-box Testing* menunjukkan bahwa seluruh fitur aplikasi berjalan sesuai kebutuhan fungsional dengan tingkat keberhasilan 100%. Sementara itu, hasil pengujian *response time* MQTT menunjukkan rata-rata waktu respons sebesar 33,9 ms, berada di bawah target pengujian sebesar 100 ms. Hasil tersebut menunjukkan bahwa aplikasi memenuhi aspek fungsionalitas dan memiliki performa komunikasi data yang responsif sesuai dengan ruang lingkup penelitian.

B. Saran/Rekomendasi

Berdasarkan hasil penelitian yang telah dilakukan, beberapa saran yang dapat diberikan untuk pengembangan penelitian selanjutnya adalah sebagai berikut:

1. Evaluasi *Usability* dan Penerimaan Pengguna

Penelitian ini mengevaluasi aplikasi melalui *Black-box Testing* dan pengukuran *response time* komunikasi MQTT sehingga berfokus pada aspek fungsionalitas dan performa teknis. Penelitian selanjutnya disarankan untuk melakukan evaluasi terhadap aspek *usability*, kemudahan penggunaan, pengalaman pengguna (*user experience*), dan tingkat penerimaan pengguna menggunakan metode seperti System

Usability Scale (SUS) atau *User acceptance Testing* (UAT). Evaluasi tersebut diharapkan dapat memberikan gambaran yang lebih komprehensif mengenai kualitas aplikasi dari sudut pandang pengguna.

2. Implementasi Kontrol pada Perangkat IoT Secara Fisik

Penelitian ini membatasi fitur kontrol volume dan *mute* hingga proses pengiriman instruksi dan pemrosesan data pada MQTT broker serta *backend system*. Oleh karena itu, penelitian selanjutnya dapat mengembangkan integrasi dengan perangkat audio yang telah mendukung fitur kontrol gain atau *Automatic Gain Control* (AGC) sehingga instruksi yang dikirimkan melalui aplikasi dapat dieksekusi secara langsung pada perangkat audio di setiap zona.

3. Pengujian pada Lingkungan Operasional Nyata

Pengujian pada penelitian ini dilakukan dalam lingkungan laboratorium dengan skenario terkontrol. Penelitian selanjutnya disarankan melakukan implementasi dan pengujian langsung pada lingkungan masjid yang sebenarnya untuk mengevaluasi kestabilan sistem, keandalan komunikasi MQTT, serta performa pada kondisi operasional yang sesungguhnya.

4. Analisis Performa pada Skala yang Lebih Besar

Pengujian *response time* pada penelitian ini dilakukan pada jumlah *node* yang terbatas. Penelitian selanjutnya dapat mengevaluasi performa sistem pada jumlah *node* yang lebih banyak untuk mengetahui kemampuan skalabilitas MQTT broker, *backend system*, dan aplikasi Android dalam menangani komunikasi data secara bersamaan pada lingkungan dengan jumlah perangkat yang lebih besar.

DAFTAR PUSTAKA

- [1] S. Rosad, F. Irkhamudin, C. Yuniar Pramudita, and S. Rizqi Alifah, “Edge-Based IoT EndPoint Performance Optimization for Smart Mosques,” *COMPILER*, vol. 14, pp. 87–96, Nov. 2025, doi: 10.28989/compiler.v14i2.3496.
- [2] S. Rosad, A. Yudhana, and T. Sutikno, “Zoned Audio Distribution: Systematic Review and Optimization *Framework* in Mosque,” *ELKHA : Jurnal Teknik Elektro*, vol. 17, no. 2, pp. 148–54, Oct. 2025.
- [3] A. M. Thantawi, S. Budilaksono, N. Anwar, and S. Komsiah, “Utilization and development of Smart Mosque Applications,” *Sinkron : Jurnal dan Penelitian Teknik Informatika*, vol. 8, no. 1, pp. 119–129, Jan. 2023, doi: 10.33395/sinkron.v8i1.11911.
- [4] R. A. Pratama and K. D. Irianto, “Rancang Bangun Aplikasi Parkir Masjid Berbasis Android dan Firebase,” *Technologia : Jurnal Ilmiah*, vol. 17, no. 1, pp. 197–206, Jan. 2026, doi: 10.31602/tji.v17i1.21416.
- [5] A. S. S. Ariyanto, I. S. M. Negara, D. N. Triwibowo, and Y. Feriyanto, “Perancangan Aplikasi Mobile Berbasis Flutter untuk Pemantauan Data Sensor IoT Solusi bagi Manajemen Suhu dan Kelembapan,” *JSAI: Journal Scientific and Applied Informatics*, vol. 8, no. 1, pp. 204–211, Jan. 2025, doi: 10.36085.
- [6] S. Ahdan and E. R. Susanto, “Implementasi Dashboard Smart Energy Untuk Pengontrolan Rumah Pintar Pada Perangkat Bergerak Berbasis *Internet of Things*,” *Jurnal Teknoinfo*, vol. 15, no. 1, pp. 26–31, Jan. 2021, doi: 10.33365/jti.v15i1.954.
- [7] I. P. Dewi and R. Fikri, “Optimalisasi Keamanan Rumah dengan Implementasi Sistem Notifikasi Gerbang Cerdas Berbasis *Internet of Things* (IoT),” *Journal of Computer System and Informatics (JoSYC)*, vol. 4, no. 4, pp. 816–829, Aug. 2023, doi: 10.47065/josyc.v4i4.4004.
- [8] M. Saiqul Umam, S. Adi Wibowo, and Y. Agus Pranoto, “Implementasi Protokol MQTT pada Aplikasi SmartGarden Berbasis IoT (*Internet of Things*),” Feb. 2023.
- [9] S. Rosad, A. Yudhana, and T. Sutikno, “Initial Study of Public Address Usage in Mosque Audio Systems,” *VoteTEKNIKA*, 2025, doi: <https://doi.org/10.24036/voteteknika.v13i1.133053>.

- [10] W. T. Saputra *et al.*, “IoT-MQTT Protocol-Based Water Sensor System to Monitor Citarum River Water Quality using Arduino Uno R4 Wifi,” *Jurnal ELTIKOM*, vol. 9, no. 1, pp. 11–22, Jun. 2025, doi: 10.31961/eltikom.v9i1.1335.
- [11] A. J. Sanjaya, Y. A. Pranoto, and F. Santi Wahyuni, “Penerapan IoT (*Internet of Things*) Untuk Sistem Monitoring Jemaah Masjid Sesuai Protokol Kesehatan Terhadap Virus Covid-19 Berbasis Arduino,” Mar. 2021.
- [12] I. Zainuddin Lapi, N. Kholis, and L. Anifah, “Rancang Bangun Purwarupa Aplikasi Masjid Pintar Menggunakan Platform Kodular Berbasis Internet di Masjid,” Jul. 2023. Accessed: Mar. 02, 2026. [Online]. Available: <https://jurnal.risetilmiah.ac.id/index.php/jti>
- [13] A. Abdulfathah and D. B. Santoso, “Pemanfaatan IoT (*Internet of Things*) dalam Monitoring Kadar Kepekatan Asap dan Kendali Camera Tracking,” *Aisyah Journal of Informatics and Electrical Engineering*, vol. 6, no. 1, pp. 125–129, Feb. 2024, doi: <https://doi.org/10.30604/jti.v6i1.221>.
- [14] Y. Yulisman, I. Ikhsan, A. Febriani, and R. Melyanti, “Penerapan *Internet of Things* (IoT) Kontrol Lampu Menggunakan *NodeMCU* ESP8266 dan *Smartphone*,” *Jurnal Ilmu Komputer*, vol. 10, no. 2, pp. 136–143, Oct. 2021, doi: 10.33060/jik/2021/vol10.iss2.231.
- [15] F. A. Habibi and U. Zaky, “Sistem Monitoring Tingkat Kualitas Udara dan Optimasi Sensor Menggunakan *Internet of Things* Berbasis Android,” *Jurnal Informatika Teknologi dan Sains*, vol. 7, no. 2, pp. 650–658, May 2025, [Online]. Available: <https://commons.wikimedia.org/>
- [16] Y. Sosmita and R. I. Salam, “Rancang Bangun Aplikasi Mobile Untuk Pemantauan dan Kontrol Sistem Keamanan Pintu Rumah Berbasis IoT,” *Jurnal Pusat Akses Kajian Teknologi Artificial Intelligence (Jurnal Pustaka AI)*, vol. 4, pp. 85–89, 2024, doi: 10.55382/jurnalpustakaai.v4i33.853.
- [17] H. Ihsan, I. Ikhsan, and R. Asmara, “Smart Home Berbasis *Internet of Things* dan Mobile Application pada Pustaka Galeri Mandiri Padang,” *Jurnal Pustaka Robot Sister (Jurnal Pusat Akses Kajian Robotika, Sistem Tertanam, dan Sistem Terdistribusi)*, vol. 1, no. 1, pp. 6–10, Jan. 2023, doi: 10.55382/jurnalpustakarobotsister.v1i1.331.
- [18] W. W. Anggoro and I. R. Widiyari, “Perancangan dan Penerapan Kendali Lampu Ruangan Berbasis IoT (*Internet of Things*) Android,” *Jurnal Teknik Informatika dan Sistem Informasi*, vol. 8, no. 3, pp. 1596–1606, Sep. 2021, [Online]. Available: <http://jurnal.mdp.ac.id>

- [19] Muslim, R. Puspita Sari, and S. Rahmayuda, "Implementasi *Framework* Flutter Pada Sistem Informasi Perpustakaan Masjid (Studi Kasus: Masjid di Kota Pontianak)," 2022.
- [20] I. M. Riyadhhi, Intan Purnamasari, and Kamal Prihandani, "PENERAPAN POLA ARSITEKTUR MVVM PADA PERANCANGAN APLIKASI PENGADUAN MASYARAKAT BERBASIS ANDROID," *INFOTECH journal*, vol. 9, no. 1, pp. 147–158, May 2023, doi: 10.31949/infotech.v9i1.5246.
- [21] A. Dillah, G. F. Nama, D. Budiyanto, and M. A. Muhammad, "Rancang Bangun Aplikasi Monitoring Operasi P2TL Pengukuran Tidak Langsung 2 Phasa di PT. PLN (PERSERO) Unit Pelaksana Pelayanan Pelanggan (UP3) Metro," *Jurnal Informatika dan Teknik Elektro Terapan*, vol. 12, no. 3, pp. 1844–1851, Aug. 2024, doi: 10.23960/jitet.v12i3.4458.
- [22] Rahmadi, I. Quraisy, and M. R. Rasyid, "Rancangan Aplikasi Kuis Anggota Kelompok Penyelenggara Pemungutan Suara (KPPS) Berbasis Android," *Journal of Computer and Information System (J-CIS)*, vol. 6, no. 1, pp. 54–63, Mar. 2023, doi: <https://doi.org/10.31605/jcis.v6i1>.
- [23] J. Dizdarevic, M. Michalke, and A. Jukan, "Engineering and Experimentally Benchmarking Open Source MQTT broker Implementations," May 2023, [Online]. Available: <http://arxiv.org/abs/2305.13893>
- [24] Ade Rahayu, "Metode Penelitian dan Pengembangan (R&D) : Pengertian, Jenis dan Tahapan," *DIAJAR: Jurnal Pendidikan dan Pembelajaran*, vol. 4, no. 3, pp. 459–470, Jul. 2025, doi: 10.54259/diajar.v4i3.5092.
- [25] N. Rachma and I. Muhlas, "Comparison Of Waterfall And Prototyping Models In Research And Development (R&D) Methods For Android-Based Learning Application Design," *Jurnal Inovatif: Inovasi Teknologi Informasi dan Informatika*, vol. 5, no. 1, pp. 36–39, 2022, doi: 10.32832/inovatif.
- [26] N. Al-Oudat, "IoT-based sound-level control for audio amplifiers: Mosques as a case study," *International Journal of Electrical and Computer Engineering*, vol. 12, no. 1, pp. 859–867, Feb. 2022, doi: 10.11591/ijece.v12i1.pp859-867.
- [27] T. Supriyadi, A. Ashari, S. Slameta, T. Utomo, and R. Solihin, "Perancangan dan Implementasi Sistem Tata Suara Masjid dengan Dua Posisi Mimbar yang berbeda," *Jurnal Pengabdian Masyarakat Sultan Indonesia*, vol. 2, no. 1, pp. 35–44, Mar. 2025, doi: 10.58291/abdisultan.v2i1.346.

DAFTAR LAMPIRAN

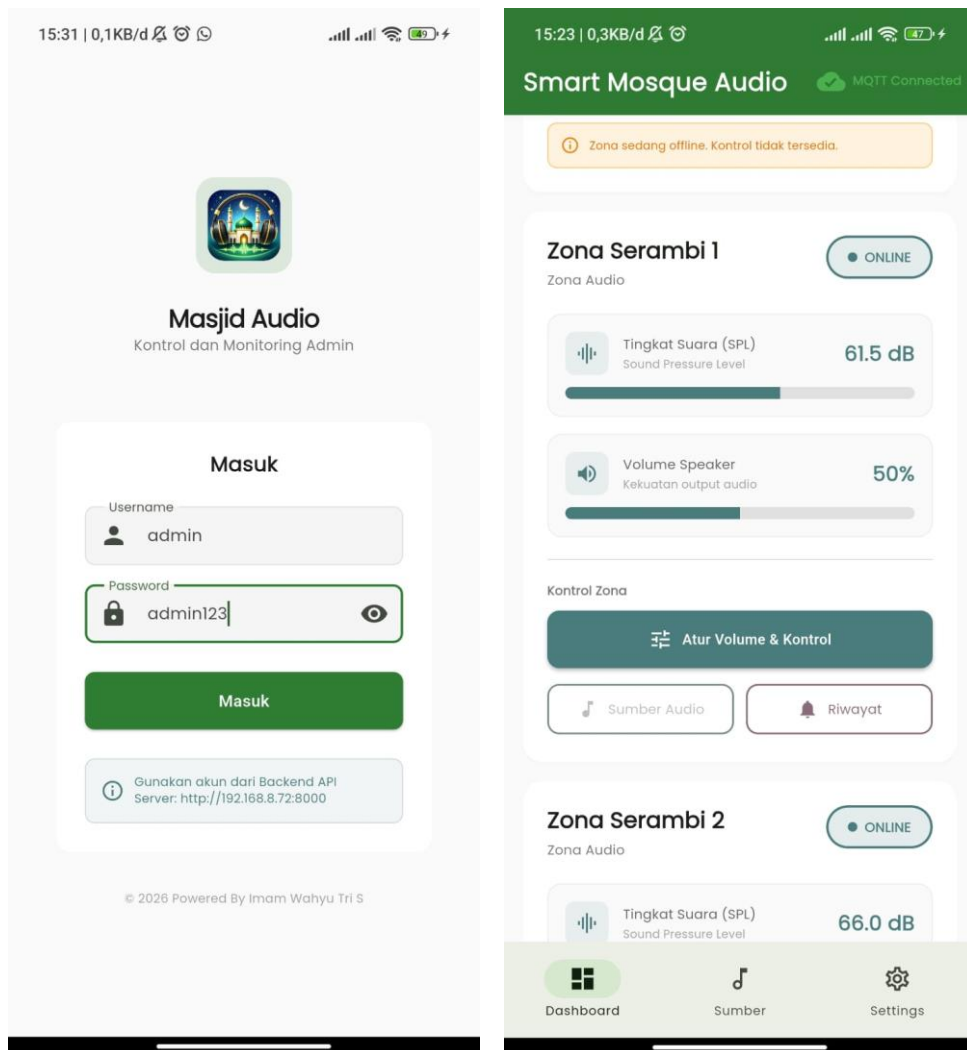
- A. Dokumentasi Pendukung :
<https://drive.google.com/drive/folders/1fkID8juHZ1JMVrFVjTsZSPIJOOGohuyI?usp=sharing>
- B. *Code* Aplikasi Masjid Audio :
https://github.com/chr1xz/ImamWahyuTriS_Masjid_audio_app.git



Scan : *Code* Aplikasi

- C. Dokumen Pendukung : <https://drive.google.com/drive/folders/1-lrsODEOj3zjFcDkOyHSAYwLqVRpNNrQ?usp=sharing>

D. Tampilan *Login Page* dan Zona Aktif



E. Tampilan Zona *Offline* dan Konfigurasi MQTT broker

