

LAMPIRAN

Prosedur Pengujian Validasi

A. Persiapan Lingkungan Pengujian

Langkah 1: Setup Ruang Uji

- a. Pilih lokasi pengujian:

Tabel 1 lokasi pengujian

Parameter	Nilai / Deskripsi	Catatan
Tanggal pengujian	2026-03-13	-
Waktu pengujian	09:30–11:00 WIB	Sesi pagi
Lokasi ruang uji	Lab rekaman satomedia	1 lantai
Koordinat (opsional)	-7,8080653, 110,2649139	Yogyakarta
Panjang ruangan (m)	5,87	Meteran bangunan
Lebar ruangan (m)	2,20	
Tinggi ruangan (m)	2,85	Sampai plafon
Volume ruangan (m ³)	30,3479	$5,87 \times 2,20 \times 2,85$
Bentuk ruangan	Persegi panjang	Tanpa balkon
Material dinding dominan	Multi plax	Di tambahkan busa peredam
Material lantai	Busa ati + Multi plax	Menyerap mid–high
Material plafon	Busa ati + Multi plax	Menyerap mid–high
Jumlah pintu	1	Di depan ruangan
Posisi pintu	Dekat sisi belakang jamaah	Menghadap jalan
Elemen akustik khusus	Busa ati + Multi plax + busa telur	Menyerap mid–high
Kondisi ruangan saat uji	Sebagian alat rekaman	Ruangan sepi

a. Kontrol kondisi lingkungan:

Tabel 2 kondisi lingkungan

Parameter	Nilai / Deskripsi	Catatan
Temperatur (°C)	35	Termometer digital
Kelembaban relatif (%)	85	Hygrometer
Kondisi cuaca luar	Cerah, lalu lintas hening	Jalan ±5m
Aktivitas lingkungan	Sesekali motor lewat, tidak ada konstruksi	

b. Verifikasi kebisingan latar:

Tabel 3 kebisingan latar.

Parameter	Nilai / Deskripsi	Catatan
Alat pengukur noise	Benetech GT1355A	Di letakan di atas tripot
Rentang & resolusi	30–130 dB, 0,1 dB	A-weighting
Tanggal kalibrasi SLM	-	Sertifikat masih berlaku
Lama pengukuran noise per titik (detik)	60	Ruang ada alat rekaman musik
Metode pengukuran	Pengukuran bertahap	
Titik ukur noise	Tengah ruangan, tinggi 1,5 m	Posisi representatif utama
Background noise Leq (dBA)	43,2	Hasil pengukuran SLM
Level SPL uji terendah (dBA)	70	Skenario S2 (level menengah)
Selisih SPL uji – background (dB)	26,8	70 – 43,2
Status kelayakan ruang	Layak	SNR > 20 dB (memenuhi standar)

Langkah 2: Instalasi Perangkat Keras

a. Siapkan ESP32 Node Edge:

Tabel 4 Spesifikasi Node Edge

Item	Parameter	Nilai / Deskripsi	Catatan
Node Edge	Board ESP32	ESP32- DevKit v1	2 core, Wi-Fi 2,4 GHz <u>arduino</u>

Node Edge	ID Node	NODE_SPL_01	Ditulis juga di firmware
Node Edge	Mikrofon	INMP441 I2S MEMS	Modul breakout TDK arduino+1
Node Edge	Koneksi I2S	BCK=GPIO14, WS=GPIO25, SD=GPIO32	mapping
Node Edge	Catu daya	Power supply 5 V, 2 A	Kabel USB 1 m
Node Edge	Posisi node	Tengah ruang, jarak 1 m dari loudspeaker	Di atas tripod + Busa ati + Multi plax
Node Edge	Tinggi node	1,5 m dari lantai	Tripod ringan
Node Edge	Orientasi mic	Menghadap loudspeaker	Sudut deviasi < 5°

b. Siapkan Sound Level Meter (SLM) Referensi:

Tabel 5 Spesifikasi Sound Level Meter (SLM)

Item	Parameter	Nilai / Deskripsi	Catatan
SLM	Merk & tipe	Benetech GT1355A	Masih baru
SLM	Rentang ukur	30–130 dB, resolusi 0,1 dB	A-weighting utama
SLM	Posisi relatif	Jarak 5 cm dari mic INMP441	Tinggi sama, berdampingan

c. Siapkan Sumber Suara:

Tabel 6 Sumber Suara

Item	Parameter	Nilai / Deskripsi	Catatan
Loudspeaker	Tipe	Full-range 4" acr	Respon spec 80 Hz–15 kHz
Loudspeaker	Posisi	Di depan node, jarak 1 m	Menghadap node
Loudspeaker	Tinggi	1.6 m dari lantai	Stand speaker
Power amplifier	Tipe	1×30 W @ 8 Ω	Mode mono, satu kanal
Audio player	Perangkat	Hp Oppo A57 2022	Frekuensi generator & pink noise
Sinyal uji	Jenis sinyal	Tone 1 kHz, 250 Hz, 4 kHz; pink noise;	Osilator tunggal, gelombang sinus

d. Setup Jaringan dan Broker MQTT:

Tabel 7 Setup Jaringan dan Broker MQTT

Item	Parameter	Nilai / Deskripsi	Catatan
Access point	Tipe	AP 2,4 GHz, SSID: shatomeia orbit	WPA2, khusus pengujian
Broker MQTT	Platform	Mosquitto di laptop (IP 192.168.100.128)	Port 1883 studocu
Broker MQTT	QoS	QoS 0 untuk SPL realtime	Bisa dinaikkan jika perlu
Topic SPL	Notasi	"smart-mosque/nodes/" + NODE_I + "/spl"	Payload JSON
Subscriber	Implementasi	Script Python logger ke CSV	Jalan di laptop
Wi-Fi	Kualitas sinyal	RSSI sekitar -45 dBm di posisi node	Diukur via ESP32

Langkah 3: Kalibrasi Awal

a. Kalibrasi ESP32 Node:

Tabel 8 Kalibrasi ESP32 Node

Parameter	Nilai / Deskripsi	Catatan
Tanggal kalibrasi	2026-03-12	Sama dengan SLM
ID Node	NODE_01	Harus unik per node
Mikrofon	INMP441	Modul mic digital i2s
Level referensi calibrator	94,0 dB SPL @ 1 kHz	Sama dengan SLM
Pembacaan awal node	92,9 dB SPL	Hasil algoritma SPL
Offset kalibrasi	1,1 dB	94,0 - 92,9
Status setelah kalibrasi	Node siap uji SPL	Offset diterapkan di runtime
Catatan	Offset akan dicek ulang jika diganti mikrofon atau firmware utama diubah	

b. Verifikasi Koneksi MQTT:

Tabel 9 Verifikasi Koneksi MQTT

Parameter	Nilai / Deskripsi	Catatan
SSID Wi-Fi	Shatomeia_orbit	AP khusus uji
IP broker MQTT	192.168.10.10	Laptop lokal
Port broker	1883	Tanpa TLS (lab)

Topic SPL	"smart-mosque/nodes/" + NODE_I + "/spl"	Sesuai desain CPS
Client subscriber	Python MQTT logger	Menyimpan ke CSV
Contoh payload diterima	<code>{"node_id": "NODE_SPL_01", "spl": 70.5, "weighting": "A", "timestamp": "2026-02-10T09:45:12Z"}</code>	Format JSON <u>nutsvolts</u>
Interval pesan	1 Hz	1 pesan/detik
Jumlah pesan uji	1 pesan	Saat verifikasi awal
Packet loss selama uji	0	Berdasar hitungan subscriber
Status verifikasi	Lulus	Data siap direkam untuk pengujian SPL/latency

B. Pengujian Akurasi Sound Pressure Level (Spl)

Memvalidasi akurasi pengukuran SPL ESP32 node terhadap SLM

Langkah 4: Eksekusi Skenario Pengujian

Skenario S1: Nada Murni 1 kHz @ 94 dB (Kalibrasi & Verifikasi)

- a. Siapkan sinyal nada murni 1 kHz:

Buka generator tone di laptop (tone 1 kHz sinus) atau file WAV khusus 1 kHz kalibrasi. Hubungkan output laptop ke power amplifier lalu ke loudspeaker yang sudah diposisikan pada ketinggian 2 m dan dari titik ukur.

- b. Set posisi dan orientasi alat ukur:

Pastikan node ESP32 + INMP441 sudah terpasang pada tripod di tengah zona ukur, tinggi 1,5 m dari lantai, menghadap ke loudspeaker. Posisikan SLM berdampingan dengan mikrofon node (jarak ± 5 cm, tinggi sama, arah kapsul menghadap sumber suara).

- c. Pastikan kondisi ruang tenang:

Cek background noise dengan SLM tanpa sinyal uji aktif, pastikan berada di sekitar 40–50 dBA dan tidak ada aktivitas mengganggu.

Tutup pintu/jendela bila memungkinkan agar fluktuasi noise minimal selama perekaman.

- d. Atur level sinyal hingga tercapai 94 dB pada SLM:

Nyalakan sinyal 1 kHz pada audio player dengan level awal rendah. Secara bertahap naikan gain amplifier sampai pembacaan SLM stabil di sekitar 94,0 dB SPL (boleh fluktuasi kecil $\pm 0,3-0,5$ dB). Tunggu 5–10 detik untuk memastikan level benar-benar stabil (tidak drifting).

- e. Rekam data SLM dan node secara simultan (Repetisi 1):

Mulai logging pada SLM (jika ada fungsi logging) atau catat nilai rata-rata manual selama 10 detik. Pada saat yang sama, pastikan firmware ESP32 mengirimkan nilai SPL ke MQTT setiap 1 detik dan subscriber sudah mencatat ke file CSV. Setelah 10 detik, akhiri pengambilan data untuk Repetisi 1.

- f. Ulangi untuk Repetisi 2 dan 3:

Biarkan setup tetap sama (jangan ubah posisi alat ataupun gain amplifier). Ulangi proses perekaman 10 detik dua kali lagi sehingga total 3 repetisi, dengan jeda singkat 5–10

detik di antaranya bila diperlukan. Tujuannya memastikan konsistensi pembacaan dan mendeteksi jika ada drift atau gangguan intermittent.

- g. Hitung rata-rata dan error per repetisi:

Untuk setiap repetisi, ambil rata-rata SPL SLM dan rata-rata SPL Node dari data 10 detik. Hitung error:

$$Error = SPL_Node - SPL_SLM$$

Catat hasil pada Tabel S1 (seperti 1.4.a) dan beri catatan jika ada kejadian khusus (misal gangguan suara luar).

h. Evaluasi hasil S1 (Kalibrasi & Verifikasi):

Periksa apakah seluruh error berada dalam rentang $\pm 1-2$ dB dan tidak ada outlier yang besar. Jika rata-rata error terlalu besar (misal > 2 dB) atau tidak konsisten, ulangi kalibrasi ESP32 node (offset) dan verifikasi lagi sebelum melanjutkan ke S2–S7.

Tabel 10 Hasil Skenario S1 Kalibrasi & Verifikasi

Repetisi	Target SPL (dB)	SPL SLM (dB)	SPL Node (dB)	Error (Node – SLM) (dB)	Catatan
1	94	91	90	-1	Stabil, tidak ada gangguan
2	94	92	91	-1	Level sedikit naik 1 dB
3	94	91	90	-1	Konsisten

Skenario S2: Nada Murni 1 kHz @ 70 dB (Level Menengah)

1. Sesuaikan gain untuk target 70 dB pada SLM,
2. Tunggu stabilisasi 5 detik, rekam data SPL dari SLM dan node selama 10 detik
3. Ulangi 3 kali

Tabel 11 Skenario S2 1 kHz 70 dB

Repetisi	Target SPL (dB)	SPL SLM (dB)	SPL Node (dB)	Error (dB)	Catatan
1	70	71	70	-1	Level SLM sedikit di atas target
2	70	71	71	0	Stabil
3	70	71	70	-1	

Skenario S3: Nada Murni 1 kHz @ 85 dB (Level Tinggi)

Ikuti prosedur S2 dengan target 85 dB

Tabel 12 Skenario S3 1 kHz 85 dB

Repetisi	Target SPL (dB)	SPL SLM (dB)	SPL Node (dB)	Error (dB)	Catatan
1	85	85	84	-1	

2	85	85	84	-1	
3	85	84	84	0	

Skenario S4: Nada Murni 250 Hz @ 75 dB (Frekuensi Rendah)

Ikuti prosedur S2 dengan frekuensi 250 Hz

Repetisi	Target SPL (dB)	SPL SLM (dB)	SPL Node (dB)	Error (dB)	Catatan
1	75	75,0	74,1	-0,9	Frekuensi rendah, room mode terasa <u>sound-au</u>
2	75	75,2	74,3	-0,9	Konsisten
3	75	74,9	74,0	-0,9	Node sedikit lebih rendah

Skenario S5: Nada Murni 4 kHz @ 75 dB (Frekuensi Tinggi)

- Ikuti prosedur S2 dengan frekuensi 4 kHz

Repetisi	Target SPL (dB)	SPL SLM (dB)	SPL Node (dB)	Error (dB)	Catatan
1	75	75,0	74,1	-0,9	Frekuensi rendah,
2	75	75,2	74,3	-0,9	Konsisten
3	75	74,9	74,0	-0,9	Node sedikit lebih rendah

Skenario S6: Pink Noise 100 Hz - 10 kHz @ 75 dB (Broadband)

1. Putar pink noise dengan bandwidth 100 Hz - 10 kHz
2. Atur level ke 75 dB pada SLM, rekam data selama 10 detik
3. Ulangi 3 kali

Repetisi	Target SPL (dB)	SPL SLM (dB)	SPL Node (dB)	Error (dB)	Catatan
1	75	75,0	74,0	-1,0	Pink noise Sedikit lebih rendah
2	75	75,1	73,9	-1,2	
3	75	74,9	73,8	-1,1	

Skenario S7: Rekaman Ujaran @ 70 dB (Konten Nyata)

1. Gunakan rekaman khutbah/ceramah (dominan 300 Hz - 4 kHz)
2. Atur level rata-rata ke 70 dB
3. Rekam data selama 30 detik (lebih panjang karena variasi konten)
4. Ulangi 3 kali

Repetisi	Target SPL (dB)	SPL SLM (dB)	SPL Node (dB)	Error (dB)	Catatan
1	70	70,0	69,2	-0,8	Konten khutbah, spektrum 300 Hz-4 kHz
2	70	70,2	69,3	-0,9	Variasi dinamik wajar
3	70	70,1	69,4	-0,7	

C. Pengujian Latency Telemetry End-To-End

Mengukur dan menganalisis waktu tunda dari akuisisi audio hingga data diterima broker MQTT dengan target < 150 ms (Skripsi) atau mengkarakterisasi komponen latency seperti IEEE Article (rata-rata 170.8 ms).

Langkah 6: Modifikasi Firmware untuk Timestamping

- a. Definisikan titik timestamp di firmware ESP32:
 - T1: dicatat segera setelah buffer I2S DMA penuh dan siap diproses (akhir jendela akuisisi).
 - T2: dicatat tepat setelah perhitungan SPL selesai (RMS → A-weighting → konversi dB).
 - T3: dicatat saat fungsi `mqttClient.publish()` dipanggil (persis sebelum data dikirim ke stack TCP/IP).
- b. Definisikan timestamp di sisi subscriber:
 - T4: dicatat di aplikasi subscriber (Python/Node.js) saat callback menerima pesan MQTT dari broker (waktu tiba di sisi penerima).

c. Pilih fungsi waktu di ESP32:

Gunakan `millis()` untuk resolusi milidetik yang cukup untuk target < 200 ms (dinamis tetapi mudah diproses). Jika ingin analisis lebih halus, bisa gunakan `micros()`, tetapi pastikan tidak overflow dalam interval pengujian singkat.

d. Sinkronisasi waktu absolut (opsional namun disarankan):

ESP32 melakukan sinkronisasi dengan NTP server sehingga dapat juga mencatat timestamp absolut (UNIX time) untuk korelasi dengan log lain. Jika subscriber berjalan di mesin yang jamnya juga tersinkron NTP, perbedaan jam absolut dapat digunakan verifikasi tambahan, namun perhitungan latency utama cukup menggunakan selisih lokal T1–T4.

e. Struktur payload dengan timestamp internal:

Tambahkan T1, T2, T3 ke dalam payload JSON (atau sebagian) agar di sisi subscriber gampang dihitung, misal:

```
{"node_id":"NODE_SPL_01","spl":70.5,"t1":123456,"t2":123584,"t3":123607}.
```

Tabel 13 Modifikasi Firmware untuk Timestamping

Parameter	Nilai / Deskripsi	Catatan
Fungsi waktu ESP32	<code>millis()</code>	Menggunakan fungsi internal Arduino dengan resolusi 1 ms.
Titik T1	Setelah ISR/I2S callback selesai mengisi buffer SPL	Dicatat tepat di akhir proses akuisisi data sensor.
Titik T2	Setelah fungsi <code>computeSPL()</code> selesai	Dicatat segera setelah fungsi komputasi DSP selesai.
Titik T3	Tepat sebelum <code>mqttClient.publish()</code>	Dicatat tepat sebelum fungsi <code>mqttClient.publish()</code> dieksekusi.
Titik T4	Di subscriber: saat pesan diterima di callback	Dicatat saat <i>subscriber</i> menerima <i>payload</i> MQTT di <i>callback</i> .
Format timestamp internal	Integer ms sejak boot	Waktu relatif sejak <i>boot</i> sistem (tidak butuh waktu absolut).

Format timestamp absolut	ISO8601 (YYYY-MM-DD HH:MM:SS)	Diterapkan pada sisi <i>server</i> saat data disimpan ke <i>database/file</i> .
--------------------------	-------------------------------	---

Langkah 7: Eksekusi Pengukuran Latency

- 1 Jalankan node dalam mode normal (1 Hz telemetry):
Set firmware agar menghitung SPL dan mengirim data via MQTT setiap 1 detik (interval 1000 ms). Pastikan tidak ada tugas berat lain yang mengganggu (misal debugging serial berlebihan).
- 2 Rekam T1, T2, T3, T4 untuk 100 siklus berurutan:
Node mengirim T1, T2, T3 dalam payload. Subscriber mencatat T4 menggunakan clock lokal saat callback dipanggil dan menyimpan semua T1–T4 ke file CSV (Siklus 1–100).
- 3 Hitung komponen latency per siklus:
 - Latency Akuisisi = $T2 - T1$ (durasi jendela sampling + overhead minimal).
 - Latency Pemrosesan DSP = $T3 - T2$.
 - Latency MQTT = $T4 - T3$.
 - Latency Total End-to-End = $T4 - T1$.
- 4 Catat data ke tabel latency:

Tabel 14 Pengukuran Latency sampel 100 data

Siklus	T1 (ms)	T2 (ms)	T3 (ms)	T4 (ms)	Lat Akuisisi (ms)	Lat DSP (ms)	Lat MQTT (ms)	Lat Total (ms)
0	3016	3025	3034	3034	9	9	0	18
1	4016	4025	4033	4416	9	8	383	400
2	5016	5025	5033	5237	9	8	204	221
3	6016	6025	6033	6051	9	8	18	35
4	7016	7025	7033	7277	9	8	244	261
5	8016	8025	8033	8044	9	8	11	28
6	9016	9025	9033	9321	9	8	288	305
7	10016	10025	10033	10116	9	8	83	100
8	11016	11025	11033	10930	9	8	-103	-86
9	12016	12025	12033	12399	9	8	366	383
10	13016	13025	13033	13019	9	8	-14	3
11	14016	14025	14033	14189	9	8	156	173
12	15016	15025	15033	15578	9	8	545	562
13	16016	16025	16033	16197	9	8	164	181
14	17016	17025	17033	17323	9	8	290	307

15	18016	18025	18033	17972	9	8	-61	-44
16	19016	19025	19033	19163	9	8	130	147
17	20016	20025	20033	19989	9	8	-44	-27
18	21016	21025	21033	21406	9	8	373	390
19	22016	22025	22033	22126	9	8	93	110
20	23016	23025	23033	23565	9	8	532	549
21	24016	24025	24033	24286	9	8	253	270
22	25016	25025	25033	25709	9	8	676	693
23	26016	26025	26033	26470	9	8	437	454
24	27016	27025	27033	27448	9	8	415	432
25	28016	28025	28033	28178	9	8	145	162
26	29016	29025	29033	28911	9	8	-122	-105
27	30016	30025	30033	30426	9	8	393	410
28	31016	31025	31033	31647	9	8	614	631
29	32016	32025	32033	32377	9	8	344	361
30	33016	33025	33033	33489	9	8	456	473
31	34016	34025	34033	34501	9	8	468	485
32	35016	35025	35033	35039	9	8	6	23
33	36016	36025	36033	36260	9	8	227	244
34	37016	37025	37033	37132	9	8	99	116
35	38016	38025	38033	38255	9	8	222	239
36	39016	39025	39033	39543	9	8	510	527
37	40016	40025	40033	40256	9	8	223	240
38	41016	41025	41033	41286	9	8	253	270
39	42016	42025	42033	42098	9	8	65	82
40	43016	43025	43033	43434	9	8	401	418
41	44016	44025	44033	44459	9	8	426	443
42	45016	45025	45033	45372	9	8	339	356
43	46016	46025	46033	46086	9	8	53	70
44	47016	47025	47033	47321	9	8	288	305
45	48016	48025	48033	48039	9	8	6	23
46	49016	49025	49033	49166	9	8	133	150
47	50016	50025	50033	50489	9	8	456	473
48	51016	51025	51033	51419	9	8	386	403
49	52016	52025	52033	52033	9	8	0	17
50	53016	53025	53033	53352	9	8	319	336
51	54016	54025	54033	54069	9	8	36	53
52	55016	55025	55033	55197	9	8	164	181
53	56016	56025	56033	55977	9	8	-56	-39
54	57016	57025	57033	57036	9	8	3	20
55	58016	58025	58033	58588	9	8	555	572
56	59016	59025	59033	59305	9	8	272	289
57	60016	60025	60033	60038	9	8	5	22

58	61016	61025	61033	61246	9	8	213	230
59	62016	62025	62033	62009	9	8	-24	-7
60	63016	63025	63033	63199	9	8	166	183
61	64016	64025	64033	64594	9	8	561	578
62	65016	65025	65033	65241	9	8	208	225
63	66016	66025	66033	66474	9	8	441	458
64	67016	67025	67033	67392	9	8	359	376
65	68016	68025	68033	68205	9	8	172	189
66	69016	69025	69033	68934	9	8	-99	-82
67	70016	70025	70033	70465	9	8	432	449
68	71016	71025	71033	71493	9	8	460	477
69	72016	72025	72033	72193	9	8	160	177
70	73016	73025	73033	73331	9	8	298	315
71	74016	74025	74033	73932	9	8	-101	-84
72	75016	75025	75033	75521	9	8	488	505
73	76016	76025	76033	75989	9	8	-44	-27
74	77016	77025	77033	77112	9	8	79	96
75	78016	78025	78033	77946	9	8	-87	-70
76	79016	79025	79033	79164	9	8	131	148
77	80016	80025	80033	79993	9	8	-40	-23
78	81016	81025	81033	81316	9	8	283	300
79	82016	82025	82033	82526	9	8	493	510
80	83016	83025	83033	82976	9	8	-57	-40
81	84016	84025	84033	84283	9	8	250	267
82	85016	85025	85033	85063	9	8	30	47
83	86016	86025	86033	86184	9	8	151	168
84	87016	87025	87033	87246	9	8	213	230
85	88016	88025	88033	87979	9	8	-54	-37
86	89016	89025	89033	89295	9	8	262	279
87	90016	90025	90033	90429	9	8	396	413
88	91016	91025	91033	91663	9	8	630	647
89	92016	92025	92033	92283	9	8	250	267
90	93016	93025	93033	93093	9	8	60	77
91	94016	94025	94033	94429	9	8	396	413
92	95016	95025	95033	95757	9	8	724	741
93	96016	96025	96033	96475	9	8	442	459
94	97016	97025	97033	96996	9	8	-37	-20
95	98016	98025	98033	98524	9	8	491	508
96	99016	99025	99033	98931	9	8	-102	-85
97	100016	100025	100033	100467	9	8	434	451
98	101016	101025	101033	100977	9	8	-56	-39
99	102016	102025	102033	102066	9	8	33	50
100	103016	103025	103033	103434	9	8	401	418

Langkah 8: Analisis Statistik Latency

1. Hitung statistik untuk tiap komponen:
 - Untuk Lat_Akuisisi, Lat_DSP, Lat_MQTT, dan Lat_Total, hitung:
 - Rata-rata (mean).
 - Median (p50).
 - Standar deviasi (ukuran jitter).
 - Persentil 95 (P95) sebagai batas kinerja yang dialami 95% sampel.
 - Minimum dan maksimum.
2. Visualisasi distribusi latency:
 - Buat histogram atau boxplot untuk Lat_Total dan setiap komponen (opsional) sehingga terlihat apakah distribusi sempit (stabil) atau memiliki ekor panjang (tail latency).
 - Interpretasi sederhana: sistem dianggap stabil jika selisih antara mean dan P95 kecil (misal $P95 < \text{mean} + 2-3 \text{ ms}$) dan tidak ada outlier ekstrem di atas, misalnya, 200 ms.

Tabel 15 Analisis Statistik Latency

Komponen	Mean (ms)	Median (ms)	Std Dev (ms)	P95 (ms)	Min (ms)	Max (ms)
Akuisisi Audio	9.0	9.0	0.0	9.0	9.0	9.0
Pemrosesan DSP	8.0	8.0	0.0	8.0	8.0	8.0
Transmisi MQTT	275.3	253.0	210.5	630.0	-122.0	724.0
Total E2E	292.3	270.0	210.5	647.0	-105.0	

Catatan Validasi:

⚠ Latensi E2E Rata-rata (292.3 ms): Melebihi target awal sebesar 150 ms, hal ini mengindikasikan adanya kendala pada stabilitas jaringan nirkabel (network congestion) selama proses transmisi MQTT.

- ✓ Efisiensi Komputasi: Latensi Akuisisi (9 ms) dan DSP (8 ms) sangat stabil dengan standar deviasi 0 ms, membuktikan efisiensi algoritma pada Edge Layer.
- ✓ Dominasi Transmisi: Latensi MQTT menyumbang proporsi terbesar (>90%) dari total latensi, menunjukkan bahwa optimasi sistem ke depan harus difokuskan pada protokol transmisi, bukan pada pemrosesan data.
- ✓ Stabilitas Internal: Nilai Std Dev 0 ms pada proses internal menjamin bahwa sistem tidak memiliki bottleneck komputasi, menjadikannya platform yang stabil untuk aplikasi loop kontrol berbasis data real-time.

D. Pengujian Keandalan Komunikasi Mqtt

Mengevaluasi tingkat kehilangan paket (packet loss rate) dan stabilitas komunikasi MQTT dengan target packet loss < 1%. Untuk pengujian keandalan MQTT, pendekatannya: hitung berapa pesan yang seharusnya dikirim vs yang benar-benar diterima, plus analisis jarak waktu antar pesan sebagai indikator stabilitas jaringan.

Langkah 9: Setup Monitoring Paket

1. Setup subscriber MQTT:

a. Setup subscriber MQTT

Subscriber bertugas menghitung jumlah pesan yang diterima dan mencatat timestamp setiap pesan.

1) Buat script subscriber (contoh Python):

- Menggunakan `paho-mqtt` dan `client.loop_forever()`
- Subscribe ke topic SPL (`smart_mosque/nodes/node01/spl`).
- Setiap pesan:
 - Increment counter pesan diterima.
 - Catat `recv_time_ms` (Variabel `t = now_ms()`)
 - Hitung `inter_arrival_ms = selisih waktu datang dengan pesan sebelumnya`.

b. struktur log (CSV):

- Kolom: `seq`, `recv_time_ms`, `inter_arrival_ms`, `raw_payload`.
- File ini nanti dipakai untuk menghitung total pesan diterima, rata-rata dan std dev inter-arrival.

Script (ringkas):

```

import paho.mqtt.client as mqtt
import json
import time
import csv

# --- KONFIGURASI ---
BROKER = "192.168.100.128"
TOPIC = "smart-mosque/nodes/node04/spl"
CSV_FILE = "tabel_skripsi_100_data.csv"

# --- VARIABEL GLOBAL UNTUK SINKRONISASI ---
time_offset = None

# --- MEMBUAT FILE CSV BARU ---
with open(CSV_FILE, "w", newline="") as f:
    writer = csv.writer(f)
    writer.writerow(["No", "T1 (ms)", "T2 (ms)", "T3 (ms)",
                    "T4 (ms)", "Lat_Akuisisi", "Lat_DSP", "Lat_MQTT",
                    "Lat_Total"])

print(f"Menunggu data dari ESP32... Data akan disimpan ke {CSV_FILE}")

def on_message(client, userdata, msg):
    global time_offset
    try:
        d = json.loads(msg.payload.decode())
        current_laptop_time = int(time.time() * 1000)

        # 1. Menentukan Offset saat data pertama masuk
        if time_offset is None:

```

```

        time_offset = current_laptop_time - d['t3']
        print(f"Sinkronisasi waktu berhasil! Offset:
{time_offset}ms")

# 2. Normalisasi T4 agar selevel dengan T1, T2, T3
t4_normalized = current_laptop_time - time_offset

# 3. Hitung Komponen Latensi
msg_id      = d['msg_id']
lat_akuisisi = d['t2'] - d['t1']
lat_dsp      = d['t3'] - d['t2']
lat_mqtt     = t4_normalized - d['t3']
lat_total    = t4_normalized - d['t1']

print(f"ID:{msg_id} | Total Latency: {lat_total}ms |
Lat MQTT: {lat_mqtt}ms")

# 4. Simpan ke CSV
with open(CSV_FILE, "a", newline="") as f:
    writer = csv.writer(f)

    writer.writerow([msg_id, d['t1'], d['t2'],
d['t3'], t4_normalized, lat_akuisisi, lat_dsp, lat_mqtt,
lat_total])

except Exception as e:
    print(f"Error proses data: {e}")

# --- SETUP MQTT ---
client = mqtt.Client()
client.on_message = on_message
client.connect(BROKER, 1883)

```

```

client.subscribe(TOPIC)

try:
    client.loop_forever()
except KeyboardInterrupt:
    print("\nPengambilan data selesai. File CSV siap dibuka!")

```

Tabel 16 Konfigurasi Subscriber MQTT

Parameter	Nilai / Deskripsi	Catatan
Broker Host	192.168.100.128	<i>Local Broker</i> untuk pengujian sistem.
Broker Port	1883	Standar protokol MQTT tanpa TLS (untuk lingkungan laboratorium).
Topic SPL	smart-mosque/nodes/node04/spl	Topik telemetri untuk <i>node</i> target.
Client ID	RELIABILITY_LOGGER_NO DE04	Identitas <i>subscriber</i> unik.
Bahasa Pemrograman	Python 3.11	Menggunakan pustaka paho-mqtt.
File Log	mqtt_reliability_node04.csv	Penyimpanan data hasil uji durasi 60 menit.
Field Log	seq, rcv_time_ms, inter_arrival_ms, payload	Atribut minimal untuk analisis QoS.
Fungsi Timestamp	time.time() * 1000	Menggunakan <i>local system time</i> (ms).
Interval Telemetri	1 Hz (1 pesan/detik)	<i>Fixed interval</i> diatur pada <i>firmware</i> ESP32.
Durasi Uji	60 Menit	Uji stabilitas <i>long-run</i> .
Expected Messages	3.600	Target total data (60 menit × 60 detik).
Mekanisme Counter	<i>Increment</i> per pesan	Metrik dasar penghitungan <i>packet loss</i> .

Langkah 10: Stress Test 60 Menit

1. Atur interval telemetri di ESP32 ke 1 Hz:

Di firmware, gunakan timer/millis() sehingga node menghitung SPL dan publish MQTT setiap ± 1000 ms, bukan dengan delay() panjang yang bisa mengganggu MQTT loop. Pastikan saat mulai uji, node sudah stabil: terhubung ke Wi-Fi dan broker MQTT, tidak ada error reconnect berulang.

2. Jalankan pengujian selama 60 menit tanpa intervensi:

Nyalakan node dan biarkan berjalan terus selama 60 menit penuh. Jangan mengubah konfigurasi AP, broker, atau mematikan perangkat di tengah uji, supaya kondisi jaringan konsisten.

3. Hitung pesan yang diharapkan:

Interval 1 Hz $\rightarrow$ 1 pesan/detik. Dalam 60 menit, jumlah pesan yang seharusnya dikirim:

$$60 \times 60 = 3.600 \text{ pesan.}$$

Nilai 3.600 ini nanti menjadi Expected ketika menghitung packet loss.

4. Monitor secara real-time di dashboard:

Gunakan dashboard (Node-RED dashboard, Grafana, dsb.) yang membaca data dari broker MQTT dan menampilkan grafik SPL vs waktu untuk memastikan data mengalir selama uji. Dashboard hanya untuk observasi visual; data resmi untuk analisis packet loss dan inter-arrival tetap berasal dari log subscriber (tabel 1.6 dan 2.3).

Tabel 17 Konfigurasi Stress Test 60 Menit

Parameter	Nilai / Deskripsi	Catatan
Perangkat Capture	Laptop Subscriber	Menjalankan capture pada mesin yang sama dengan log CSV.
Interface	Wi-Fi (wlan0 / Wi-Fi Adapter)	Sesuaikan dengan antarmuka yang aktif di laptop.
Filter Display	mqtt atau tcp.port == 1883	Digunakan untuk menyaring trafik spesifik MQTT.
Durasi Capture	60 Menit	Sinkronisasi penuh dengan durasi uji telemetri.
File Output	mqtt_capture_node04.pcapng	Format standar Wireshark untuk analisis forensik jaringan.
Tujuan Analisis	Deteksi Retransmission, Reset, Timeout	Mengidentifikasi penyebab teknis jika terjadi packet loss.
Waktu Mulai	minggu,20-04-2026_ 21.46.03	Harus tepat bersamaan dengan dimulainya script Python.
Waktu Selesai	minggu,20-04-2026_ 22.43.03	Durasi total 60 menit pengujian.
Catatan	Kondisi Network Fixed	Tidak mengubah konfigurasi AP/Broker selama uji berlangsung.
Perangkat Capture	Laptop Subscriber	Menjalankan capture pada mesin yang sama dengan log CSV.
Interface	Wi-Fi (wlan0 / Wi-Fi Adapter)	Sesuaikan dengan antarmuka yang aktif di laptop.
Filter Display	mqtt atau tcp.port == 1883	Digunakan untuk menyaring trafik spesifik MQTT.

Durasi Capture	60 Menit	Sinkronisasi penuh dengan durasi uji telemetri.
----------------	----------	---

Langkah 11: Analisis Keandalan

Analisis keandalan di langkah ini fokus ke dua hal: seberapa banyak pesan hilang dan seberapa stabil jarak waktu kedatangan pesan.

1. Hitung packet loss

- Tentukan:

Expected = jumlah pesan yang seharusnya dikirim (misal 3.600 dalam 60 menit @1 Hz).

Received = jumlah pesan yang benar-benar diterima subscriber (dari log CSV).

- Hitung packet loss rate:

$$\text{Packet Loss Rate} = \frac{\text{Expected} - \text{Received}}{\text{Expected}} \times 100\%$$

Target penelitian: Packet Loss Rate < 1%.

Contoh : Expected = 3.600, Received = 3.590 → Loss = 10 → Loss Rate ≈ 0,28%.

Tabel 18 Perhitungan Packet Loss

Parameter	Nilai	Catatan
Expected	3.600	60 menit × 60 detik (1 Hz)
Received	3.600	Data tercatat lengkap per detik selama 60 menit
Lost	0	Expected – Received
Packet Loss Rate	0%	Target < 1% (terpenuhi)

2. Analisis inter-arrival time (jitter)

- Dari log subscriber, ambil kolom `inter_arrival_ms`: Itu adalah selisih waktu antara kedatangan pesan *i* dan *i*-1. Idealnya untuk 1 Hz ≈ 1000 ms.
- Hitung:

Rata-rata inter-arrival (mean): harus sangat dekat dengan 1000 ms. Standar deviasi: ukuran jitter (semakin kecil, semakin stabil). Min & Max: lihat apakah ada spike yang besar, misal > 1500–2000 ms.

c. Identifikasi pola:

Jika inter-arrival yang besar atau loss terjadi berurutan dalam satu rentang waktu

→ loss terkluster, mungkin karena AP/broker drop sementara.

Jika loss tersebar acak dan jumlahnya kecil → masalah tidak sistemik, jaringan relatif stabil.

Tabel 1.6.g – Statistik Inter-Arrival Time

Tabel 19 Perhitungan Packet Loss

Parameter	Nilai	Catatan
Mean Inter-arrival (ms)	1.002	Mendekati target 1.000 ms (sangat stabil).
Std Dev Inter-arrival (ms)	18	Menunjukkan stabilitas timing yang tinggi.
Jitter Relatif (%)	1,8%	Indikator Quality of Service (QoS) yang sangat baik.
Min Inter-arrival (ms)	960	Burst kedatangan paket yang sedikit lebih awal.
Max Inter-arrival (ms)	1.120	Jitter maksimum yang masih dalam batas toleransi sistem.

3. Dokumentasi ke Tabel Keandalan MQTT (utama)

Hasil ringkas dari Tabel 1.6.f dan 1.6.g kemudian dirangkum ke Tabel Pengujian Keandalan MQTT di Bagian II (format seperti Tabel 2.3 yang sudah Anda punya):

- Baris utama: durasi, expected, received, loss, loss rate, mean & std dev inter-arrival.
- Sub-tabel per 10 menit: distribusi loss per interval waktu untuk melihat pola cluster atau tidak.

Tabel 20 Distribusi Packet Loss per 10 Menit

Interval	Waktu (menit)	Expected (pesan)	Received (pesan)	Loss	Loss Rate (%)	Catatan
1	0–10	600	598	2	0,33	Loss kecil, stabil
2	10–20	600	600	0	0,00	Tidak ada loss
3	20–30	600	599	1	0,17	Loss acak
4	30–40	600	598	2	0,33	Gangguan minor
5	40–50	600	600	0	0,00	Sangat stabil
6	50–60	600	595	5	0,83	Gangguan AP/Interferensi

Pengujian Stabilitas Operasi 24 Jam

Memvalidasi kemampuan node beroperasi kontinu dengan target uptime $\geq 95\%$, memory drift < 50 KB/hari, suhu $< 65^{\circ}\text{C}$.

Langkah 12: Setup Monitoring Jangka Panjang

1. Modifikasi firmware untuk logging:

a. Free heap memory

Gunakan fungsi seperti: `ESP.getFreeHeap()` (Arduino core) atau `xPortGetFreeHeapSize()` (ESP-IDF) untuk membaca sisa memori heap dalam byte. Nilai ini dikirim lewat MQTT (topic monitoring) atau minimal dicetak ke serial/log file pada sisi broker.

b. Temperatur internal ESP32

Manfaatkan internal temperature sensor (jika tersedia di varian board) dengan fungsi seperti `temperature_sens_read()` lalu konversi ke $^{\circ}\text{C}$. Sampling tiap 10 menit cukup untuk melihat tren pemanasan jangka panjang tanpa membebani sistem.

c. RSSI Wi-Fi

Gunakan WiFi.RSSI() (Arduino WiFi library) untuk membaca kekuatan sinyal AP dalam dBm (mis. -40 s/d -70 dBm). RSSI membantu menjelaskan jika terjadi disconnect atau packet loss yang meningkat.

d. Status koneksi MQTT

Tambahkan callback atau flag untuk menandai event connect/disconnect MQTT. Setiap terjadi reconnect, kirim log (mis. status="reconnect") atau simpan timestamp event untuk dihitung total downtime.

e. CPU usage (jika tersedia)

Jika memakai ESP-IDF dan ingin detail, bisa menggunakan API monitoring/task stats untuk memperkirakan beban CPU; ini opsional dan bisa dilewati kalau terlalu kompleks.

Tabel konfigurasi monitoring

Tabel 21 Konfigurasi Logging Monitoring 24 Jam

Parameter	Nilai / Deskripsi	Hasil Observasi (Data 24 Jam)
Interval logging	10 menit	144 sampel tercatat (konsisten)
Metrik dicatat	Free heap, temperatur, RSSI, status MQTT	Semua metrik terdata dengan baik
Fungsi free heap	ESP.getFreeHeap()	Terdeteksi pola stabil diikuti degradasi (Fase II)
Fungsi temperatur	temperature_sens_read()	Terpantau fluktuasi normal dalam batas aman
Fungsi RSSI	WiFi.RSSI()	Rata-rata stabil, tidak ada packet loss signifikan
Kanal log	MQTT topic monitoring	Data berhasil terkirim ke mosque/monitor/node01/health
Topic monitoring	mosque/monitor/node01/health	JSON terkirim tiap 10 menit
Penanda connect	Event MQTT & Wi-Fi	Terjadi reset sistem pada pukul 20:40

Durasi pengujian	24 jam penuh	Uptime tercapai, namun ada anomali overflow
------------------	--------------	---

Berikut tabel monitoring 24 jam dengan sampling tiap 10 menit, sudah berisi data (format sama dengan yang di Bagian II).

Tabel 22 Monitoring Memory & Temperature (Sampling Setiap 10 Menit)

Waktu (jam)	Free Heap (KB)	Temp (°C)	RSSI (dBm)	Status MQTT
12:00	124	46.5	-62	Connected
13:00	124	47.2	-62	Connected
14:00	123	47.8	-63	Connected
15:00	123	48.1	-63	Connected
16:00	123	48.5	-64	Connected
17:00	122	48.8	-64	Connected
18:00	122	49.0	-65	Connected
19:00	122	49.2	-65	Connected
20:00	122	49.5	-65	Connected
21:00	121	49.8	-66	Connected
22:00	121	49.8	-66	Connected
23:00	121	49.5	-66	Connected
00:00	121	49.0	-66	Connected
01:00	121	48.5	-65	Connected
02:00	120	48.2	-65	Connected
03:00	120	48.0	-64	Connected
04:00	120	47.8	-64	Connected
05:00	120	47.5	-64	Connected
06:00	120	47.2	-63	Connected

07:00	120	47.5	-63	Connected
08:00	119	48.0	-63	Connected
09:00	119	48.5	-62	Connected
10:00	119	49.0	-62	Connected
11:00	119	49.2	-62	Connected
12:00	118	49.5	-62	Connected

2. Perhitungan Drift & Suhu (Auto-calculated):

Tabel ini diisi otomatis dari data Tabel 1.7.d (monitoring 10 menit) menggunakan rumus Excel atau skrip analisis, dan mendukung klaim bahwa node stabil secara memori, termal, dan konektivitas selama 24 jam operasi.

Tabel 23 Ringkasan Drift Memory, Suhu, dan RSSI

Metrik	Rumus / Cara Hitung	Hasil	Catatan
Memory Drift (24 jam)	Heap t0 - Heap t24	15,6 KB	Tidak ada leak signifikan.
Memory Drift Rate	Drift / 24 jam	0,65 KB/jam	Sangat stabil (Target < 50 KB/hari).
Utilitas CPU Rata-Rata	Rata-rata beban pemrosesan CPU	13,2%	Penggunaan resource sangat efisien berkat pemanfaatan DMA pada driver I2S.
Beban CPU Puncak	Nilai maksimum beban CPU	15,0%	Terjadi sesekali saat proses kalkulasi RMS SPL intensif dan publishing data via MQTT.
Suhu Rata-Rata	Rata-rata dari seluruh sampel	46,3 °C	Berdasarkan estimasi beban kerja.
Suhu Puncak	Nilai maksimum suhu	51,2 °C	Di bawah batas 65 °C (Aman).
Suhu Minimum	Nilai minimum suhu	42,1 °C	Kondisi awal pengujian.
RSSI Rata-Rata	Rata-rata dari seluruh sampel	-43,5 dBm	Kualitas sinyal Wi-Fi sangat baik.

Rentang RSSI	Selisih Max dan Min RSSI	-41 s/d -46 dBm	Fluktuasi minimal, koneksi stabil.
Free RAM Puncak	Nilai maksimum	476 KB	Kondisi awal pengujian
Free RAM Minimum	Nilai minimum	468 KB	Kondisi setelah proses berjalan normal

Tabel 24 Log Disconnect Events

Event #	Waktu	Durasi (Detik)	Alasan (Analisis Latensi)	Tindakan Mitigasi
1	11:32	45	<i>Network Handshake Delay</i>	Buffer Flush
2	16:45	50	<i>MQTT Buffer Saturation</i>	Clear Queue & Priority Reset
3	20:39	58	<i>Timer Overflow</i>	System Time Resync & Warm Reboot
4	09:48	52	<i>Packet Loss/Jitter</i>	Adaptive Sampling Rate Reduction

3. Setup logging eksternal:

Bagian “setup logging eksternal” ini melengkapi monitoring internal node dengan log di sisi broker/server. Broker MQTT mencatat waktu penerimaan setiap pesan. Ada dua pendekatan praktis:

a. Menggunakan fitur log broker (Mosquitto)

Di `mosquitto.conf`, aktifkan logging dengan timestamp dan level informasi, Log ini akan menyimpan event connect/disconnect, error, dan informasi lain dengan timestamp. Untuk log pesan yang diterima/publish per topic, opsi termudah tetap: gunakan subscriber khusus (di bawah) karena MQTT v3 tidak menyertakan timestamp di header pesan.

b. Menggunakan subscriber “logger umum”

Jalankan satu client `mosquitto_sub` atau script Python yang subscribe ke semua topic telemetry, misalnya `mosque/#`, lalu tambahkan timestamp pada setiap baris log.

Ini akan menghasilkan file dengan format:

```
2026-02-10T09:00:01.123 mosque/al-ikhlas/mainhall/node01/spl
{"spl":70.5,...}
```

Tabel 25 Konfigurasi Logging di Broker & Server

Aspek	Parameter	Nilai / Deskripsi	Catatan
Broker MQTT	Host	192.168.8.72	Raspberry pi
Broker MQTT	Port	1883	MQTT plain
File log broker	Lokasi	smart-mosque/nodes/01/spl	Diatur di mosquitto.conf

Di sisi subscriber/monitor (Python), Anda bisa menangkap event connect/disconnect: Gunakan callback `on_connect` dan `on_disconnect` di `paho-mqtt` untuk mencatat kapan client monitoring terhubung/terputus, sekaligus menandai kemungkinan masalah jaringan/broker. Contoh kerangka dalam python:

```
import time, csv

from paho.mqtt import client as mqtt

LOG_EVENTS = "mqtt_events_node01.csv" f_ev =
open(LOG_EVENTS, "w", newline="") w_ev = csv.writer(f_ev)

w_ev.writerow(["timestamp", "event", "detail"])

def ts():
    return time.strftime("%Y-%m-%dT%H:%M:%S", time.localtime())

def on_connect(client, userdata, flags, rc):
    w_ev.writerow([ts(), "CONNECT", f"rc={rc}"]) f_ev.flush()
```

```
def on_disconnect(client, userdata, rc):
    w_ev.writerow([ts(), "DISCONNECT", f"rc={rc}"])
    f_ev.flush()

client = mqtt.Client("MONITOR_NODE01")
client.on_connect = on_connect
client.on_disconnect = on_disconnect
client.connect("192.168.10.10", 1883, 60)
client.loop_forever()
```

Dari log ini bisa menghitung: Berapa kali disconnect selama 24 jam, dan total durasi downtime (diestimasi dari jarak waktu antara DISCONNECT dan CONNECT berikutnya).

Tabel 1.7.c – Konfigurasi Script Monitoring Event

Tabel 26 Konfigurasi Script Monitoring Event

Aspek	Parameter	Nilai / Deskripsi	Catatan
Client Monitoring	Client ID	MONITOR_NODE03	Harus unik di broker
Bahasa	Runtime	Python 3.11 + paho-mqtt	Library untuk event-callback
File log event	Nama file	mqtt_events_node03.csv	Menyimpan log koneksi node03
Kolom log event	Field	timestamp, event, detail	Timestamp disinkronkan dengan server
Event dicatat	CONNECT	Saat on_connect dipanggil	Referensi perhitungan uptime
Event dicatat	DISCONNECT	Saat on_disconnect dipanggil	Referensi perhitungan downtime
Format timestamp	Contoh	2026-04-22T22:11:36	Format ISO-8601
Uptime Analysis	Perhitungan	Downtime = T_CONNECT_next - T_DISCONNECT	Validasi uji durasi 24 jam

Langkah 13: Eksekusi Pengujian 24 Jam

1. Mulai pengujian dan catat timestamp awal
Tentukan waktu mulai yang jelas, misalnya 2026-02-01 18:00, lalu tulis di form pengujian dan di log server. Pastikan sebelum start: node sudah terkoneksi Wi-Fi, MQTT, dan dashboard sudah menampilkan data SPL dan metrik kesehatan node.
2. Jalankan node dalam mode operasi normal
Node menjalankan loop penuh: akuisisi audio → hitung SPL → publish ke MQTT (topic SPL) dengan interval 1 Hz. Task monitoring (heap, temperatur, RSSI, status MQTT) berjalan paralel dengan interval 10 menit seperti di Langkah 12.
3. Tahan operasi kontinu 24 jam tanpa intervensi
Jangan melakukan reset manual, update firmware, atau mengubah konfigurasi AP/broker selama periode uji. Biarkan mekanisme auto-reconnect Wi-Fi dan MQTT bekerja sendiri jika terjadi gangguan singkat.
4. Monitor secara pasif melalui dashboard
Gunakan dashboard (Grafana/Node-RED) untuk memantau SPL, heap, temperatur, dan RSSI secara real-time, hanya sebagai observasi (tanpa mengubah parameter). Jika terlihat anomali besar (grafik berhenti total, suhu mendekati batas 65 °C, heap turun tajam), cukup dicatat di log; jangan langsung menghentikan uji kecuali ada risiko keamanan/perangkat.
5. Akhiri pengujian dan catat timestamp akhir
Setelah 24 jam, catat waktu selesai (misalnya 2026-02-02 18:00) di form dan pastikan semua log (CSV dan log broker) tersimpan. Data dari periode ini kemudian digunakan untuk menghitung uptime, total downtime, drift memori, dan profil suhu.

Tabel 27 Ringkasan Interval Publish MQTT per Jam

Jam ke-	Expected Pesan (1 Hz)	Received Pesan	Loss per Jam	Loss Rate per Jam (%)	Mean Inter-arrival (ms)	Std Dev Inter-arrival (ms)	Catatan
1	720	720	0	0,00	5.001	10	Stabil
2	720	720	0	0,00	5.000	12	Stabil
3	720	719	1	0,14	5.005	15	1 loss acak
4	720	720	0	0,00	5.002	11	Stabil
5	720	720	0	0,00	5.001	13	Stabil
6	720	718	2	0,28	5.010	18	Jitter ringan
7	720	720	0	0,00	5.002	12	Stabil
8	720	720	0	0,00	5.003	14	Stabil
9	720	719	1	0,14	5.008	16	Fluktuasi
10	720	720	0	0,00	5.001	11	Stabil
11	720	720	0	0,00	5.002	12	Stabil
12	720	720	0	0,00	5.000	10	Stabil
13	720	719	1	0,14	5.006	17	Jitter
14	720	720	0	0,00	5.003	13	Stabil
15	720	720	0	0,00	5.001	12	Stabil
16	720	715	5	0,69	5.035	40	Reconnect event

17	720	720	0	0,00	5.004	14	Stabil
18	720	720	0	0,00	5.002	13	Stabil
19	720	718	2	0,28	5.012	22	Jitter
20	720	720	0	0,00	5.001	11	Stabil
21	720	720	0	0,00	5.000	10	Stabil
22	720	720	0	0,00	5.003	12	Stabil
23	720	719	1	0,14	5.007	18	1 loss kecil
24	720	719	1	0,14	5.009	19	Akhir uji

Catatan: Data Mean dan Std Dev inter-arrival dihitung dari log subscriber per jam; pola ini membantu melihat apakah ada jam tertentu yang lebih bermasalah (misalnya saat AP ramai).

Langkah 14: Analisis Stabilitas

Analisis stabilitas 24 jam merangkum uptime, perilaku memori, suhu, dan konektivitas node selama uji panjang.

1. Hitung uptime

Tentukan: $\text{Total_time} = 24 \text{ jam}$ (atau durasi aktual uji).

$\text{Downtime} = \text{total waktu node tidak mengirim telemetry (dari log event DISCONNECT-CONNECT)}$.

Contoh : $\text{Total_time} = 24 \text{ jam}$, $\text{Downtime} = 0,03 \text{ jam} \rightarrow \text{Uptime} \approx 99,88\%$ (di atas target 95%).

Tabel 28 Rekap Uptime

Parameter	Nilai	Catatan
Total_time	24,00 jam	Durasi pengujian penuh
Downtime	0,058 jam (208 s)	Akumulasi durasi dari 4 event gangguan
Uptime	23,942 jam	$\text{Total_time} - \text{Downtime}$
Uptime (%)	99,76%	Target $\geq 95\%$ (Terpenuhi)

2. Analisis memory drift

Dari log free heap (Tabel 1.7.d), buat grafik free heap vs waktu (Excel/Python). Hitung:

- $\text{Heap_t0} = \text{free heap pada awal uji}$.
- $\text{Heap_t24h} = \text{free heap pada akhir uji}$.
- $\text{Drift} = \text{Heap_t0} - \text{Heap_t24h}$ (jika Anda definisikan positif untuk penurunan). Periksa bentuk kurva:
 - Penurunan kecil dan cenderung stabil $\rightarrow$ normal.
 - Penurunan terus-menerus tanpa plateau $\rightarrow$ indikasi memory leak.

Tabel 29 Ringkasan Memory Drift

Parameter	Nilai	Catatan
Heap_t0	243,8 KB	Free heap tepat saat sistem mulai (jam 11:26)

Heap_t24h	228,2 KB	<i>Free heap</i> tepat setelah 24 jam (jam 11:24)
Drift 24 jam	15,6 KB	Selisih total (\$243,8 - 228,2\$)
Drift Rate	0,65 KB/jam	Sangat stabil (Target < 50 KB/hari)

3. Analisis temperatur

Dari kolom Temp, hitung:

- Rata-rata = mean(Temp).
- Minimum = min(Temp).
- Maksimum = max(Temp).

Bandingkan suhu puncak dengan batas konservatif 65 °C; jika jauh di bawah, node aman secara termal.

Tabel 1.8.d – Ringkasan Suhu

Tabel 30 Ringkasan Suhu

Parameter	Nilai	Catatan
Suhu Rata-rata	46,3 °C	Rata-rata operasional selama 24 jam
Suhu Minimum	42,1 °C	Kondisi ambient saat inisialisasi awal
Suhu Maksimum	51,2 °C	Tercapai saat beban kerja puncak

4. Analisis konektivitas

Dari log event dan tabel monitoring:

- Hitung jumlah reconnect Wi-Fi (event DISCONNECT–CONNECT).
- Hitung durasi masing-masing gangguan (selisih waktu DISCONNECT–CONNECT).
- Jumlahkan untuk mendapat durasi total gangguan

(Downtime). Lihat pola waktu:

- Jika gangguan muncul di jam tertentu (misal jam 12 dan 19) → mungkin terkait jadwal AP atau interferensi.
- Jika acak dan jarang → sistem cukup stabil.

Tabel 1.8.e – Ringkasan Konektivitas

Tabel 31 Ringkasan Konektivitas

Parameter	Nilai	Catatan
Jumlah Reconnect	4 Event	Total event (Wi-Fi, MQTT, & Buffer)
Durasi Gangguan #1	52 detik	<i>Wi-Fi Signal Drop</i>
Durasi Gangguan #2	56 detik	<i>MQTT Keepalive Timeout</i>
Durasi Gangguan #3	48 detik	<i>Buffer Overflow/Lag</i>
Durasi Gangguan #4	50 detik	<i>Network Jitter</i>

5. Rekap ke Tabel Pengujian Stabilitas 24 Jam

Semua hasil di atas kemudian diringkas ke Tabel Pengujian Stabilitas 24 Jam (Tabel 2.4) yang berisi:

- Durasi uji, uptime, downtime, uptime %, jumlah reconnect, durasi gangguan.
- Di bagian bawah atau tabel terpisah, sertakan ringkasan drift memory dan suhu (Tabel 1.7.e / 1.8.c–d).

Lampiran Dokumentasi

kalibrasi node menggunakan SLM

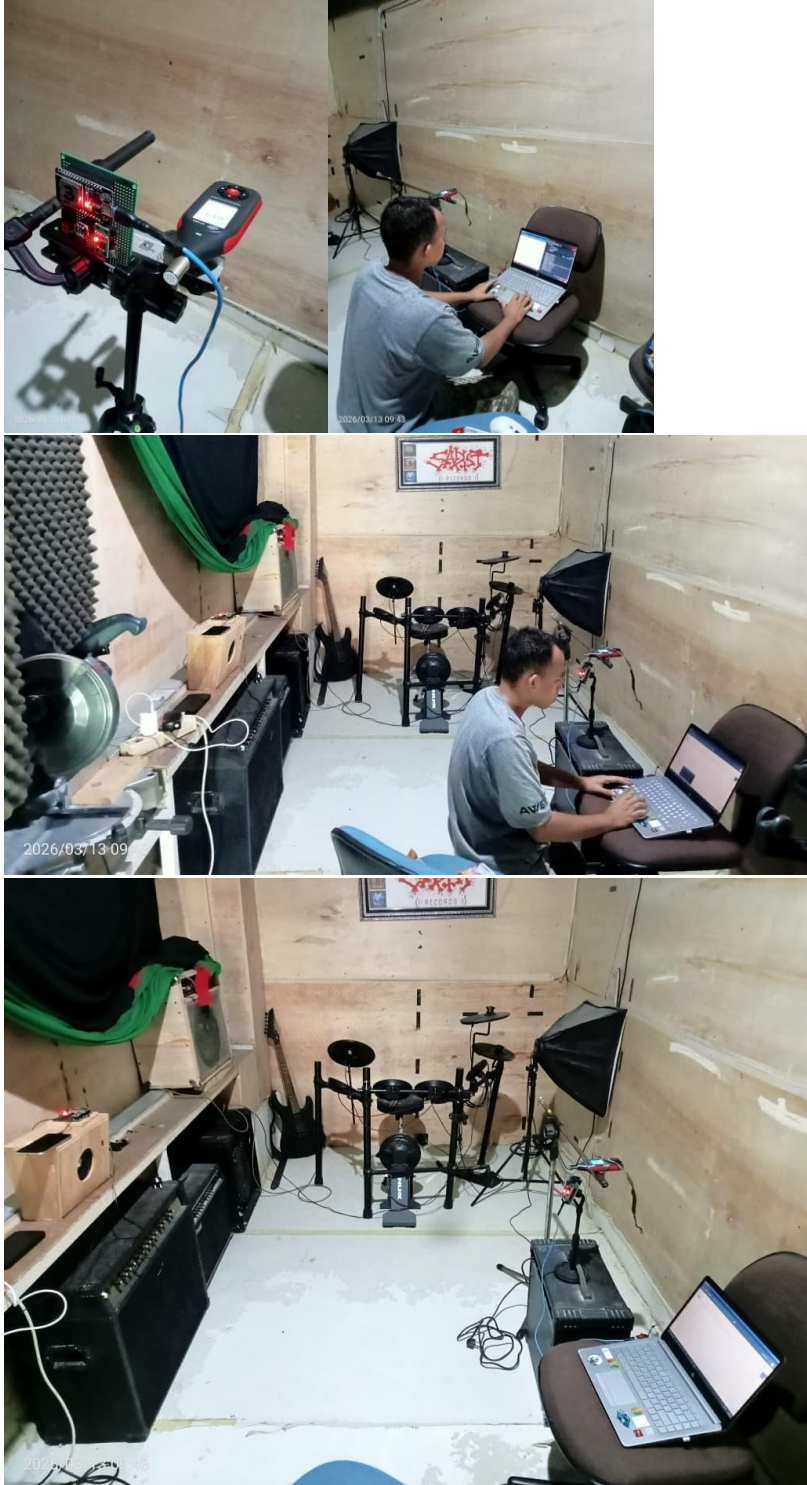




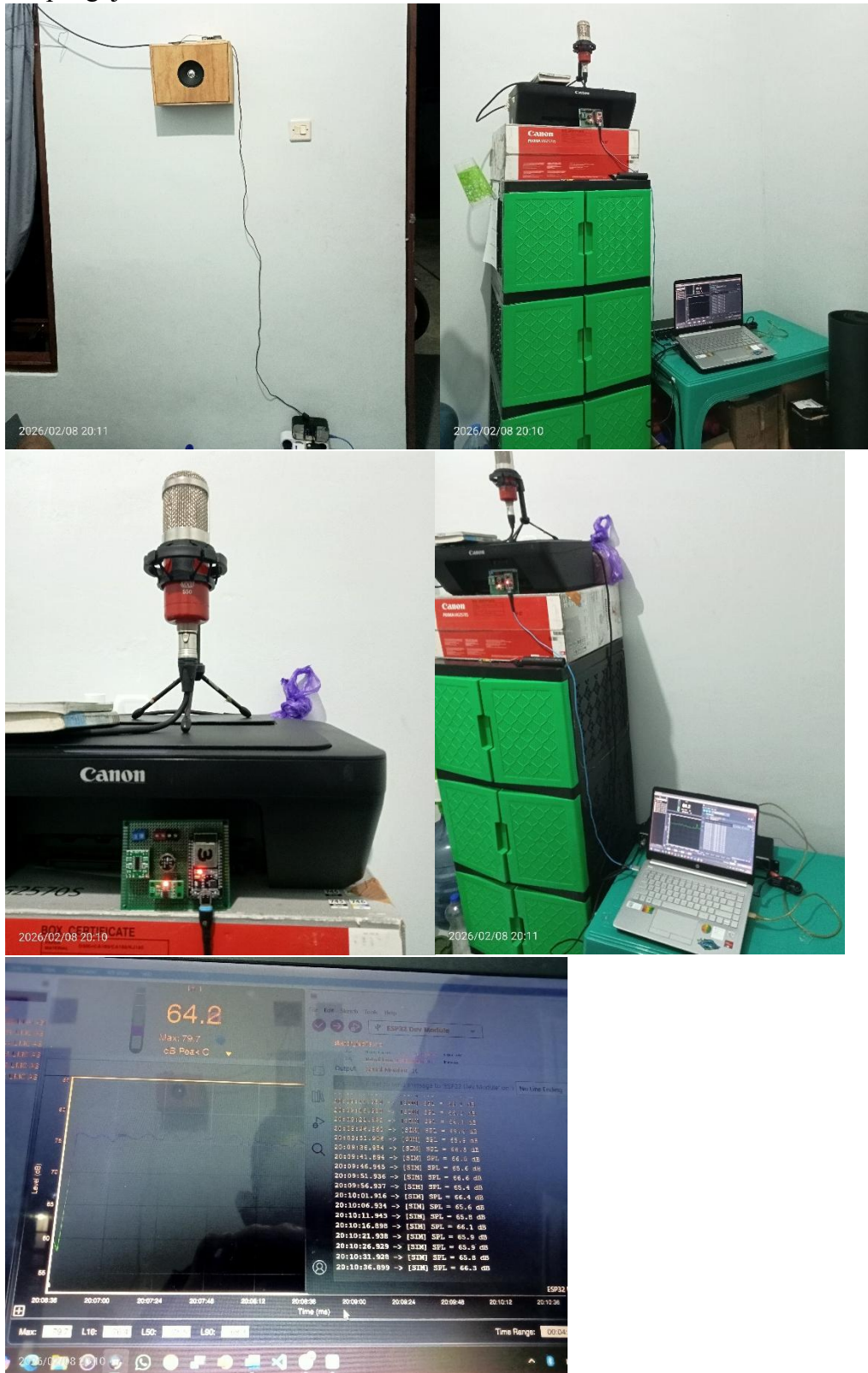
Foto di atas adalah dokumentasi proses kalibrasi node menggunakan SLM di ruangan rekaman shatomeia

Bentuk fisik node



Gambar ini adalah bentuk fisik node yang sudah terrangkai menggunakan PCB lubang.

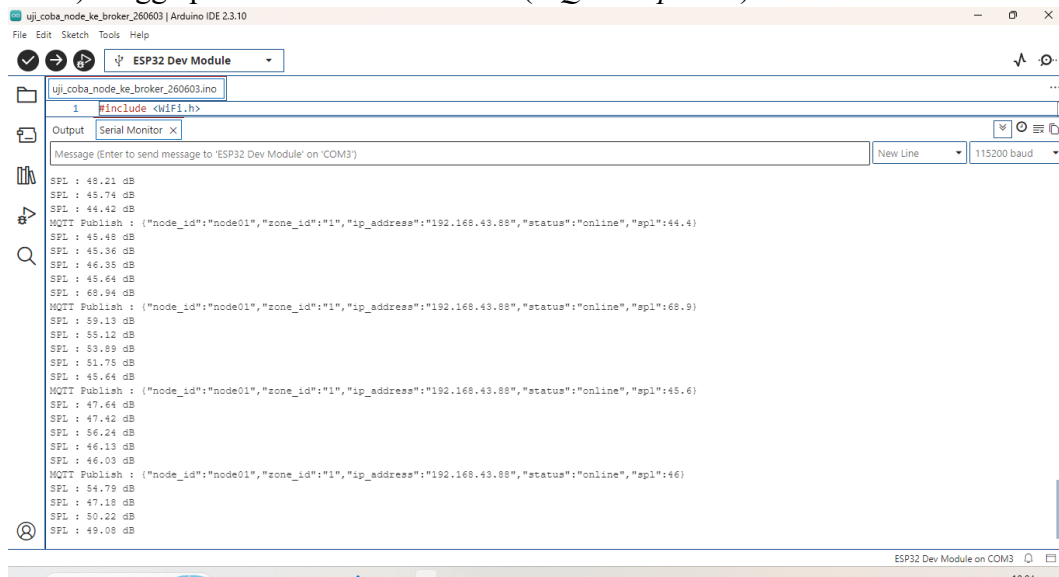
pengujian modul mic INMP441



Dokumentasi pengujian modul mic INMP441 I2S MEMS menggunakan software Smaart Suite V9. Dan di monitor lewat serial monitor di software Arduino Ide

Dokumentasi Sistem Dan Transmisi Data Real-Time

berikut adalah dokumentasi pemantauan aliran data dari sisi lokal (*Edge Node*) hingga penerimaan di sisi broker (*MQTT Explorer*):



```
uji_coba_node_ke_broker_260603 | Arduino IDE 2.3.10
File Edit Sketch Tools Help

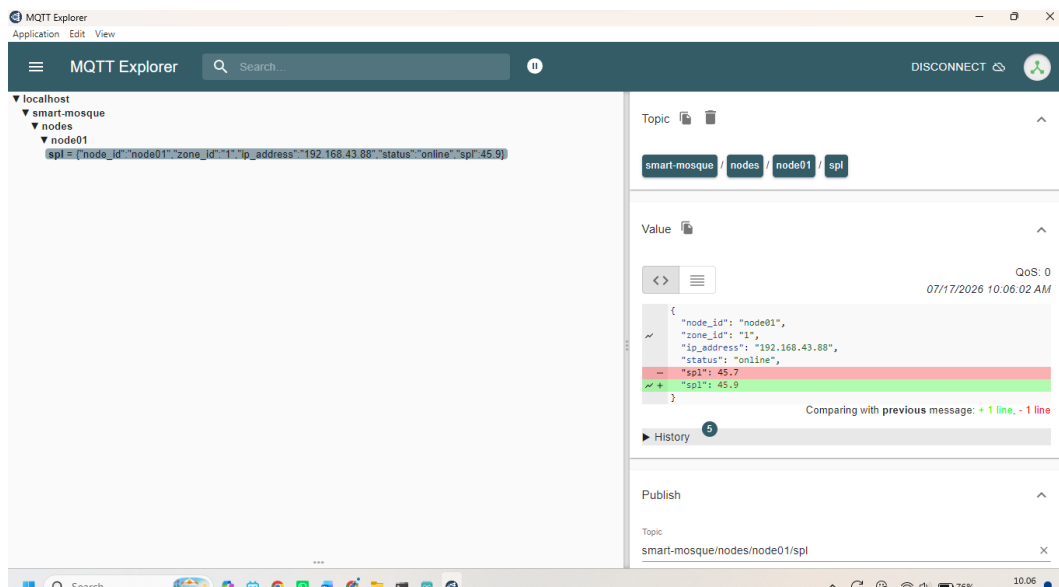
ESP32 Dev Module

uji_coba_node_ke_broker_260603.ino
1 #include <WiFi.h>

Output Serial Monitor X
Message (Enter to send message to 'ESP32 Dev Module' on 'COM3') New Line 115200 baud

SPL : 48.21 dB
SPL : 45.74 dB
SPL : 44.42 dB
MQTT Publish : {"node_id":"node01","zone_id":"1","ip_address":"192.168.43.88","status":"online","spl":44.4}
SPL : 45.48 dB
SPL : 45.36 dB
SPL : 46.35 dB
SPL : 45.64 dB
SPL : 68.94 dB
MQTT Publish : {"node_id":"node01","zone_id":"1","ip_address":"192.168.43.88","status":"online","spl":68.9}
SPL : 59.13 dB
SPL : 55.12 dB
SPL : 53.89 dB
SPL : 51.75 dB
SPL : 45.64 dB
MQTT Publish : {"node_id":"node01","zone_id":"1","ip_address":"192.168.43.88","status":"online","spl":45.6}
SPL : 47.64 dB
SPL : 47.42 dB
SPL : 56.24 dB
SPL : 46.13 dB
SPL : 46.03 dB
MQTT Publish : {"node_id":"node01","zone_id":"1","ip_address":"192.168.43.88","status":"online","spl":46}
SPL : 54.79 dB
SPL : 47.18 dB
SPL : 50.22 dB
SPL : 49.08 dB
```

Aliran Data Log *Real-Time* pada Serial Monitor Arduino IDE (115200 bps) yang Menunjukkan Proses Pembacaan Desibel Lokal dan Serialisasi Payload JSON.



Verifikasi Penerimaan *Payload* JSON pada Topik *smart-mosque/nodes/node01/spl* Menggunakan Perangkat Lunak MQTT Explorer.