Pembuatan Sistem Indikator RGB Led Berbasis ESP32 Dan Aplikasi Android Untuk Monitoring Parkir Di Area Parkir Luar TULT

1st Matthew Jonathan El Rindengan
Telkom University
Fakultas Teknik Elektro
Bandung, Indonesia
matthewrindengan@student.telkomuniveristy.ac.id

2nd Uke Kurniawan Usman
Telkom University
Fakultas Teknik Elektro
Bandung, Indonesia
ukeusman@telkomuniveristy.ac.id

3rd Sri Astuti
Telkom University
Fakultas Teknik Elektro
Bandung, Indonesia
sriastuti@telkomuniveristy.ac.id

***Abstrak* — Meningkatnya jumlah kendaraan di area luar Gedung TULT (Telkom University Landmark Tower) menyebabkan kesulitan dalam menemukan slot parkir yang tersedia, mengakibatkan ketidakteraturan dalam penggunaan lahan parkir. Untuk mengatasi hal ini, dikembangkan sistem parkir pintar berbasis IoT yang mengintegrasikan Firebase *Realtime Database*, indikator LED WS2812B, dan aplikasi mobile berbasis Flutter guna memberikan informasi parkir secara *real-time* serta mendukung reservasi *slot* dan manajemen *check-in/check-out* berbasis QR Code. Sistem ini bekerja dengan memperbarui status *slot* parkir langsung di Firebase, yang kemudian ditampilkan di aplikasi pengguna serta dikontrol pada indikator LED menggunakan ESP32-S3. Fitur utama yang diimplementasikan meliputi pemantauan ketersediaan slot, reservasi, notifikasi status parkir, serta override admin untuk menyelesaikan konflik parkir. Hasil pengujian menunjukkan bahwa sistem ini mampu menampilkan status parkir dengan latensi rata-rata di bawah 200 ms, sementara indikator LED memiliki visibilitas yang baik dalam berbagai kondisi pencahayaan. Evaluasi pengalaman pengguna melalui System Usability Scale (SUS) menghasilkan skor 73.86, yang menunjukkan bahwa aplikasi memiliki kemudahan penggunaan yang baik. Dengan hasil ini, sistem parkir pintar berbasis IoT ini dapat menjadi solusi efektif dalam mengoptimalkan pengelolaan parkir di lingkungan kampus.**

***Kata kunci*— Firebase, IoT, Smart Parking, WS2812B, ESP32-S3, Hybrid App**

I. PENDAHULUAN

Meningkatnya jumlah kendaraan di area luar Gedung TULT (Telkom University Landmark Tower) menyebabkan kesulitan dalam menemukan slot parkir yang tersedia, mengakibatkan waktu pencarian parkir yang lebih lama dan penggunaan lahan parkir yang tidak teratur. Sistem parkir konvensional yang mengandalkan sensor fisik sering kali memiliki biaya implementasi yang tinggi serta membutuhkan perawatan yang kompleks.

Penelitian ini mengembangkan sistem parkir pintar berbasis *Intertet of Things* yang memanfaatkan Firebase Realtime Database sebagai pusat data, ESP32-S3 untuk mengontrol indikator parkir berbasis WS2812B RGB LED, serta aplikasi *mobile* berbasis Flutter untuk memantau dan mengelola parkir secara *real-time*. Sistem ini memungkinkan pengguna untuk melihat status parkir, melakukan reservasi, *serta check-in* dan *check-out* menggunakan QR Code.

Melalui pengujian kualitas layanan (QoS) dan evaluasi pengalaman pengguna, sistem ini terbukti mampu memberikan pembaruan data secara *real-time* dengan latensi rata-rata 127,45 ms, serta mendapatkan skor System Usability Scale (SUS) sebesar 73,86, yang menunjukkan tingkat penerimaan yang baik. Dengan hasil ini, sistem parkir pintar berbasis IoT diharapkan dapat menjadi solusi efektif dalam mengoptimalkan pengelolaan parkir di lingkungan kampus.

II. DASAR TEORI

Pada sistem parkir pintar ini, platform utama yang digunakan adalah Firebase *Realtime Database* sebagai pusat penyimpanan data parkir secara *real-time* serta ESP32-S3 untuk mengontrol indikator parkir berbasis WS2812 RGB LED Matrix. Aplikasi mobile dikembangkan menggunakan Flutter, yang memungkinkan pengguna untuk melihat status parkir, melakukan reservasi, serta *check-in* dan *check-out* menggunakan QR Code.

Fitur Firebase yang digunakan dalam sistem ini meliputi Realtime Database untuk menyimpan status ketersediaan slot parkir, Firebase *Authentication* untuk mengelola akses pengguna, serta Firebase *Cloud Messaging* untuk mengirimkan notifikasi terkait status parkir. Pengujian sistem dilakukan dengan mengukur *Quality of Service* (QoS), yang mencakup latensi komunikasi, *throughput*, dan keandalan pembaruan data, serta evaluasi *usability* menggunakan *System Usability Scale* (SUS).

A. ESP32-S3

ESP32-S3 adalah *System-on-Chip* (SoC) berdaya rendah yang dikembangkan oleh *Espressif Systems*, dilengkapi dengan prosesor dual-core Xtensa LX7 yang beroperasi hingga 240 MHz. Perangkat ini memiliki konektivitas Wi-Fi 2.4 GHz dan Bluetooth 5 (LE), menjadikannya sangat cocok untuk berbagai aplikasi Internet of Things (IoT). ESP32-S3 memiliki 512 KB SRAM, 384 KB ROM, dan 16 KB RTC SRAM, dengan dukungan untuk memori flash eksternal dan PSRAM melalui antarmuka Quad SPI atau Octal SPI. Salah satu fitur unggulannya adalah kemampuan akselerasi AI melalui instruksi vektor yang meningkatkan performa dalam tugas-tugas pembelajaran mesin. Selain itu, ESP32-S3 juga menyediakan berbagai periferal seperti GPIO, ADC, dan USB OTG, yang memungkinkan fleksibilitas dalam interaksi dengan perangkat keras[1].

B. WS2812 RGB MATRIX

WS2812 adalah sumber cahaya LED pintar yang mengintegrasikan sirkuit kontrol dan chip RGB dalam satu paket berukuran 5050. Setiap LED memiliki sirkuit reshaping sinyal bawaan, yang memastikan keutuhan transmisi data dan memungkinkan pencascadingan banyak LED tanpa kehilangan sinyal. WS2812 beroperasi dengan satu pin input data, memungkinkan kontrol banyak LED hanya dengan satu pin mikrokontroler. Sistem ini mendukung kecepatan transmisi data hingga 800 Kbps, memastikan transisi warna yang mulus dan animasi berkualitas tinggi. Setiap LED dapat menampilkan 256 tingkat kecerahan untuk masing-masing warna, menghasilkan total 16.777.216 kombinasi warna, sehingga ideal untuk tampilan warna penuh dan aplikasi pencahayaan dekoratif[2].

C. FIREBASE

Firebase adalah platform pengembangan aplikasi berbasis cloud yang dikembangkan oleh Google, dirancang untuk menyediakan berbagai layanan backend tanpa perlu mengelola infrastruktur server. Firebase memungkinkan sinkronisasi data real-time, otentikasi pengguna, pengiriman notifikasi, dan berbagai layanan lainnya yang membantu pengembang membangun aplikasi dengan cepat dan efisien. Dalam penelitian ini, Firebase digunakan sebagai backend utama, dengan fitur utama yang diimplementasikan meliputi Realtime Database dan Firebase Authentication.

1. Realtime Database

Firebase Realtime Database adalah database NoSQL berbasis cloud yang menyimpan data dalam format JSON dan memungkinkan sinkronisasi data secara real-time antara berbagai perangkat yang terhubung. Dengan teknologi ini, setiap perubahan status parkir yang terjadi akan langsung diperbarui dalam sistem tanpa perlu polling berkala, sehingga meningkatkan efisiensi komunikasi dan mengurangi latensi. Selain itu, Firebase Realtime Database memiliki dukungan mode offline, yang memungkinkan perangkat tetap dapat membaca dan menulis data meskipun tidak terhubung ke internet. Saat koneksi kembali, Firebase secara otomatis menyinkronkan perubahan yang terjadi selama mode offline[3].

2. Firebase Authentication

Firebase Authentication adalah layanan autentikasi berbasis cloud yang mendukung berbagai metode login, termasuk email dan kata sandi, verifikasi nomor telepon, serta autentikasi melalui layanan pihak ketiga seperti Google, Facebook, dan Twitter. Dengan menggunakan JSON Web Token (JWT), Firebase *Authentication* memastikan bahwa setiap transaksi dalam sistem berasal dari pengguna yang sah, sehingga mengurangi risiko penyalahgunaan dan meningkatkan keamanan sistem[4].

B. *Quality of Service Testing*

Pengujian Quality of Service (QoS) merupakan langkah krusial dalam menilai kinerja dan keandalan suatu jaringan atau sistem komunikasi[5]. Parameter-parameter utama yang dievaluasi meliputi latensi, throughput, packet loss, dan retransmission rate. Evaluasi menyeluruh terhadap metrik-metrik ini memberikan pemahaman mendalam tentang kemampuan sistem dalam memenuhi kebutuhan aplikasi yang sensitif terhadap waktu dan data.

1. Latency

Latensi mengacu pada waktu yang dibutuhkan oleh data untuk berpindah dari sumber ke tujuan dalam suatu jaringan. Ini mencakup berbagai komponen, seperti processing delay (waktu yang diperlukan perangkat untuk memproses header paket), queuing delay (waktu yang dihabiskan paket dalam antrian), transmission delay (waktu untuk mentransmisikan bit paket ke media), dan propagation delay (waktu yang diperlukan sinyal untuk merambat melalui media transmisi). Latensi yang rendah sangat penting untuk aplikasi real-time, seperti video conferencing dan online gaming, di mana keterlambatan dapat mempengaruhi pengalaman pengguna secara signifikan.

2. Throughput

Throughput adalah jumlah data yang berhasil ditransfer dari satu titik ke titik lain dalam jaringan dalam jangka waktu tertentu. Ini mencerminkan kapasitas efektif dari jaringan dan dipengaruhi oleh berbagai faktor, termasuk bandwidth yang tersedia, latensi, dan overhead protokol. Throughput yang tinggi menunjukkan bahwa jaringan dapat menangani volume data yang besar dengan efisien, yang penting untuk aplikasi seperti streaming video dan transfer file berukuran besar.

3. Packet Loss

Packet loss terjadi ketika satu atau lebih paket data gagal mencapai tujuan yang dituju, yang dapat disebabkan oleh kemacetan jaringan, kesalahan transmisi, atau kerusakan perangkat keras. Kehilangan paket dapat berdampak signifikan pada kualitas layanan, terutama dalam aplikasi seperti streaming video dan komunikasi suara, di mana kehilangan data dapat menyebabkan penurunan kualitas yang nyata. Penelitian telah mengidentifikasi bahwa faktor-faktor seperti penundaan, variasi penundaan, dan packet loss mempengaruhi Quality of Service (QoS).

4. Retransmission Rate

Retransmission rate mengukur frekuensi pengiriman ulang paket data yang gagal mencapai tujuan atau tiba dalam keadaan rusak. Tingkat retransmisi yang tinggi dapat mengindikasikan masalah dalam kualitas koneksi jaringan, seperti gangguan sinyal atau kemacetan, dan dapat menyebabkan peningkatan latensi serta penurunan throughput. Pengelolaan tingkat retransmisi yang efektif sangat penting untuk menjaga kinerja jaringan yang optimal dan memastikan pengalaman pengguna yang memuaskan.

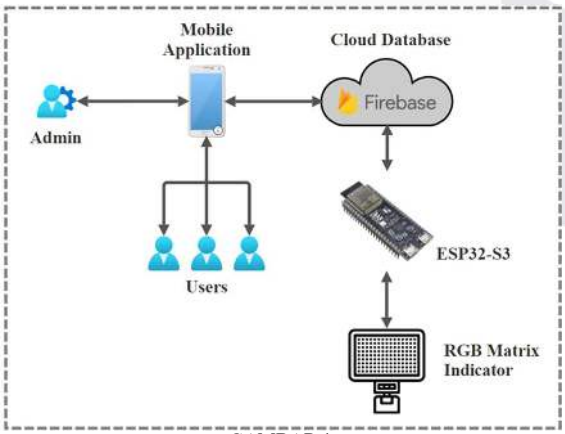
III. PERANCANGAN SISTEM

Sistem parkir pintar berbasis IoT yang dikembangkan dalam penelitian ini bertujuan untuk menciptakan arsitektur yang mampu mengintegrasikan ESP32-S3 sebagai mikrokontroler, Firebase Realtime Database sebagai pusat penyimpanan data, WS2812B RGB LED sebagai indikator status parkir, dan aplikasi mobile berbasis Flutter sebagai antarmuka utama bagi pengguna. Keberhasilan sistem ini ditandai dengan kemampuan sinkronisasi data secara real-time, yang memungkinkan pengguna melihat informasi parkir terkini, melakukan reservasi, serta check-in dan check-out dengan cepat dan mudah. Dengan pendekatan ini, sistem parkir dapat beroperasi dengan latensi rendah, throughput optimal, serta akurasi tinggi dalam memperbarui status ketersediaan slot parkir.

A. Implementasi

Sistem parkir pintar berbasis IoT ini dirancang untuk menyediakan informasi real-time tentang ketersediaan slot parkir dengan menghubungkan berbagai komponen perangkat keras dan perangkat lunak dalam satu ekosistem yang terintegrasi. ESP32-S3 berfungsi sebagai unit pemrosesan utama, yang berkomunikasi dengan Firebase Realtime Database untuk memperbarui status parkir, serta mengontrol WS2812B RGB LED yang berfungsi sebagai indikator visual di lapangan. Aplikasi mobile berbasis Flutter memungkinkan pengguna untuk melihat status slot parkir, melakukan reservasi, serta menyelesaikan proses check-in dan check-out menggunakan QR Code.

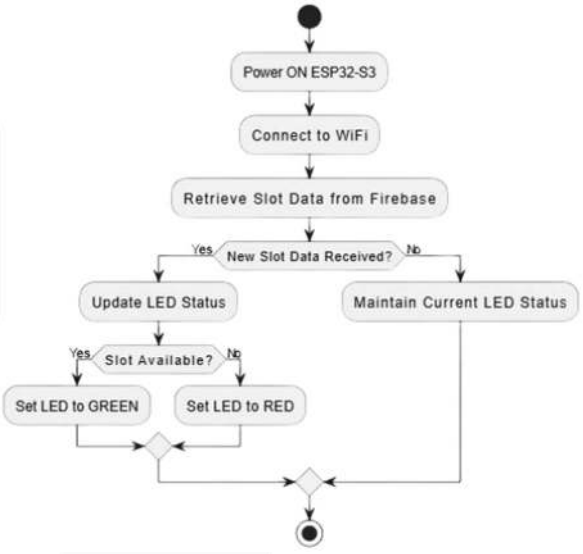
Pada tingkat sistem, Firebase bertindak sebagai backend pusat yang menyimpan informasi status slot, data reservasi, dan histori check-in/check-out pengguna. Setiap kali pengguna melakukan reservasi atau check-in melalui aplikasi, Firebase secara otomatis memperbarui status slot tersebut dan mengirimkan pembaruan ke ESP32-S3. ESP32-S3 kemudian mengontrol tampilan indikator WS2812B RGB LED, yang berubah menjadi merah saat slot terisi dan kembali ke hijau saat slot kosong. Gambar 1 menggambarkan arsitektur sistem, yang mencakup alur komunikasi antara aplikasi, Firebase, dan mikrokontroler.



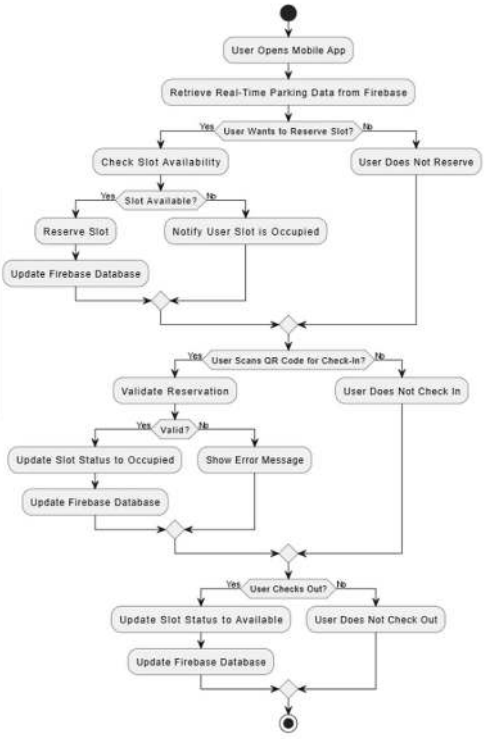
GAMBAR 1
Diagram Arsitektur Sistem

Selain pembaruan status parkir secara otomatis, sistem juga menerapkan Firebase Authentication untuk mengelola akses pengguna. Hanya pengguna yang telah terdaftar yang dapat melakukan reservasi dan check-in, sementara admin memiliki hak akses untuk melakukan override status slot jika diperlukan. Dengan adanya autentikasi berbasis Firebase, sistem ini dapat memastikan keamanan data serta mencegah penyalahgunaan fitur parkir.

Untuk mendukung pengolahan data dan komunikasi antara aplikasi mobile, Firebase, dan ESP32-S3, sistem ini menggunakan pendekatan event-driven synchronization, di mana setiap perubahan yang terjadi pada Firebase langsung diproses oleh ESP32-S3 dalam waktu nyata. Gambar 2 dan 3 menunjukkan bagaimana proses komunikasi ini berlangsung, mulai dari pengguna yang melakukan check-in hingga perubahan status yang ditampilkan di indikator LED.



GAMBAR 2
Diagram Alur Perangkat Keras



GAMBAR 3
Diagram Alur Aplikasi Mobile

Agar sistem dapat berjalan dengan lancar, perancangan perangkat keras juga menjadi aspek penting dalam implementasi ini. ESP32-S3 dihubungkan dengan WS2812B RGB LED yang digunakan sebagai indikator visual, serta catu daya 5V 30A untuk memastikan suplai daya yang stabil. Tabel 1 merangkum spesifikasi perangkat keras yang digunakan dalam implementasi sistem ini.

TABEL 1
Spesifikasi Perangkat Keras

Item	Jumlah
ESP32-S3	1
WS2812B RGB LED Matrix (16x16, 256 LEDs)	7
Catu Daya WZ-0530J (5V 30A)	1
Kabel Listrik AC	1

Implementasi ini juga memperhatikan efisiensi komunikasi data, di mana sistem dioptimalkan untuk mengurangi latensi pembaruan status slot, memastikan bahwa setiap perubahan dapat diterapkan dalam waktu kurang dari 200 ms. Dengan demikian, sistem ini dapat memberikan pengalaman pengguna yang lebih responsif serta meningkatkan efisiensi pengelolaan parkir di lingkungan kampus.

B. Langkah-langkah Implementassi

Berikut langkah-langkah dari implementasi sistem:

1. Merancang arsitektur sistem dengan mengintegrasikan ESP32-S3, Firebase Realtime Database, Firebase Authentication, aplikasi mobile Flutter, dan indikator LED WS2812B.
2. Mengonfigurasi Firebase Realtime Database untuk menyimpan informasi parkir, termasuk status slot, reservasi pengguna, serta data check-in dan check-out.
3. Menghubungkan ESP32-S3 dengan Firebase, sehingga mikrokontroler dapat membaca status parkir dan memperbarui tampilan indikator LED secara otomatis.
4. Mengembangkan aplikasi mobile berbasis Flutter, dengan fitur seperti monitoring parkir real-time, reservasi slot, check-in/check-out berbasis QR Code, serta notifikasi pengguna.
5. Menerapkan Firebase Authentication sebagai sistem keamanan yang memastikan hanya pengguna terdaftar yang dapat menggunakan fitur reservasi dan check-in.
6. Mengembangkan program pada ESP32-S3 untuk membaca data dari Firebase dan mengontrol WS2812B RGB LED, sehingga setiap perubahan status parkir ditampilkan secara real-time.
7. Melakukan pengujian fungsional untuk memastikan bahwa setiap perubahan status di aplikasi dapat tersinkronisasi dengan Firebase dan diperbarui ke indikator LED.
8. Melakukan evaluasi kinerja sistem, dengan mengukur latensi komunikasi, throughput data, packet loss, serta keandalan sistem, guna memastikan sistem dapat bekerja secara efisien dalam penggunaan nyata.

IV. PENGUJIAN DAN ANALISIS

Pengujian *Quality of Service* (QoS) dilakukan untuk mengukur performa komunikasi sistem parkir pintar berbasis IoT. Pengujian ini berfokus pada latensi, throughput, packet loss, dan retransmission rate dalam tiga segmen utama, yaitu Aplikasi ke Backend (*App-Backend*), *Backend* ke Mikrokontroler (*Backend*-ESP32-S3), serta pengukuran menyeluruh (*End-to-End*).

A. QoS Aplikasi - *Backend*

Pengujian ini mengukur efisiensi komunikasi antara aplikasi mobile dan Firebase Realtime Database menggunakan Wireshark untuk menangkap paket data serta Firebase Performance Monitoring untuk menganalisis waktu respons server.

Gambar 5.1 menunjukkan contoh packet capture yang digunakan sebagai contoh perhitungan QoS berdasarkan data dari uji ke-11 dalam tabel pengujian. Gambar ini menampilkan pertukaran paket antara klien (10.215.173.1) dan server Firebase (35.186.236.207), termasuk proses resolusi DNS, *handshake* TCP, negosiasi TLS 1.3, dan pertukaran *application* data antara aplikasi dan *backend*.

GAMBAR 4
Packet Capture Tes 11

Latensi dihitung berdasarkan selisih waktu antara pengiriman data oleh aplikasi (Paket 17) dan respons dari server Firebase (Paket 19). Pada uji ke-11, perhitungannya sebagai berikut:

$$Latensi = (0.430349 - 0.350407) \times 1000 = 79.94\ ms$$

Rata-rata latensi dari 15 pengujian tercatat 70,5 ms, dengan rentang 7,86 ms hingga 144,05 ms.

Throughput dihitung berdasarkan total data yang dipertukarkan selama durasi komunikasi. Pada uji ke-11, dengan total data sebesar 1812 byte dan durasi transmisi 0,079942 detik, perhitungannya sebagai berikut:

$$Throughput = \frac{1812 \times 8}{0.079942 \times 1000} = 181.35\ Kbps$$

Rata-rata throughput adalah 324,25 Kbps, dengan variasi antara 54,47 Kbps hingga 1350,65 Kbps. Tidak ditemukan packet loss atau retransmission, menandakan komunikasi antara aplikasi dan Firebase berjalan stabil.

Tabel 2 menyajikan hasil pengujian QoS segmen App-Backend.

TABEL 2
Hasil Pengujian QoS Aplikasi-Backend

Uji Ke	Request Timestamp (ms)	Response Timestamp (ms)	Latency (ms)	Packet Loss (%)	Throughput (Kbps)	Retransmission Rate (%)
1	1.433549	1.577599	144.05	0	100.58	0
2	0.242403	0.268851	26.45	0	54.47	0
3	0.207161	0.262385	55.224	0	262.05	0
4	0.183340	0.197392	14.052	0	751.9	0
5	0.265539	0.311861	46.322	0	310.79	0
6	0.423088	0.449874	26.786	0	537.50	0
7	0.258313	0.398226	139.913	0	104.24	0
8	0.419901	0.427762	7.861	0	1350.65	0
9	0.449864	0.370279	79.585	0	182.1	0
10	0.318480	0.389898	71.418	0	202.13	0
11	0.350407	0.430349	79.94	0	181.39	0
12	0.413001	0.542387	129.386	0	111.97	0
13	0.285286	0.346556	61.27	0	236.6	0
14	0.305865	0.398196	92.33	0	156.76	0
15	0.148796	0.205949	57.15	0	254.09	0

B. QoS Backend-ESP32 S3

Pengujian ini mengevaluasi kinerja komunikasi antara Firebase *Realtime Database* dan ESP32-S3, yang bertanggung jawab untuk memperbarui indikator LED parkir berdasarkan perubahan status di database. Gambar 5.2 menunjukkan tampilan hasil pengukuran QoS menggunakan Serial Monitor ESP32-S3, di mana waktu perubahan status di backend dibandingkan dengan waktu penerimaan oleh ESP32-S3.

GAMBAR 5
Contoh Hasil Pengukuran QoS Serial Monitor pada ESP32-S3

Latensi dihitung dengan membandingkan timestamp pembaruan di Firebase (databaseChangeTime) dengan waktu penerimaan di ESP32 (slotMatrixUpdateTime). Pada uji ke-11, perhitungannya adalah:

$$Latensi = 1739870878347 - 1739870878196 = 151\ ms$$

Dari 15 uji coba, latensi rata-rata tercatat 100 ms, dengan variasi antara 64 ms hingga 151 ms.

Throughput dihitung berdasarkan jumlah data JSON yang diterima oleh ESP32 dalam periode tertentu. Pada uji ke-11, dengan total data 1024 byte dan durasi transmisi 0,151 detik, perhitungannya sebagai berikut:

$$Throughput = \frac{1812 \times 8}{0.151 \times 1000} = 5.17\ Kbps$$

Rata-rata throughput adalah 6,8 Kbps, dengan variasi 4,75 Kbps hingga 12,21 Kbps. Tidak ditemukan packet loss atau retransmission, yang menunjukkan stabilitas jaringan antara backend dan ESP32-S3.

Tabel 3 menyajikan hasil pengujian QoS segmen Backend-ESP32-S3.

TABEL 3
Hasil Pengujian QoS Aplikasi-Backend

Uji Ke	Timestamp Perubahan Backend (UNIX Epoch)	Timestamp Respons Mikrokontroler (Unix Epoch)	Latency (ms)	Packet Loss (%)	Throughput (Kbps)	Retransmission Rate (%)
1	1739852654986	1739852655050	64	0	12.21	0
2	1739857797543	1739857797644	101	0	5.65	0
3	1739860162067	1739860162192	125	0	6.25	0
4	1739864414411	1739864414479	68.0	0	8.39	0
5	1739866511944	1739866512070	126.0	0	6.20	0
6	1739867849111	1739867849215	104	0	5.48	0
7	1739869052353	1739869052426	73	0	10.70	0
8	1739869325161	1739869325228	67	0	8.51	0
9	1739869814633	1739869814710	77	0	10.15	0
10	1739870130176	1739870130296	120	0	4.75	0
11	1739870878196	1739870878347	151	0	5.17	0
12	1739871513761	1739871513827	66	0	8.64	0
13	1739872658667	1739872658797	130	0	6.01	0
14	1739873747323	1739873747409	86	0	6.63	0
15	1739874686825	1739874686947	122	0	6.40	0

B. QoS End-to-End

Pengujian ini mengukur performa sistem secara keseluruhan, mencakup komunikasi dari aplikasi hingga indikator parkir pada ESP32-S3. Latensi total dihitung dengan menjumlahkan latensi dari *App-Backend* dan *Backend*-ESP.

Pada uji ke-11, perhitungannya adalah:

$$Latensi_{total} = 79.94 + 151 = 230.94\ ms$$

Hasil pengukuran menunjukkan latensi total berkisar antara 74,86 ms hingga 230,94 ms, dengan rata-rata 127,45 ms.

Throughput end-to-end dihitung dengan mengambil nilai throughput terendah antara kedua segmen karena transmisi data dilakukan secara berurutan. Pada uji ke-11, perhitungannya adalah:

$$Throughput_{total} = min(181.35,5.17) = 5.17\ Kbps$$

Rata-rata throughput adalah 6,5 Kbps, dengan variasi 4,75 Kbps hingga 12,21 Kbps. Tidak ditemukan packet loss maupun retransmission selama pengujian.

Tabel 4 menyajikan hasil pengujian QoS End-to-End.

TABEL 4
Hasil Pengujian QoS Aplikasi-Backend

Uji Ke	Total Latency (ms)	Packet Loss (%)	Throughput (Kbps)	Retransmission Rate (%)
1	208.05	0	12.21	0
2	127.45	0	5.65	0
3	180.22	0	6.25	0
4	82.05	0	8.39	0
5	172.32	0	6.20	0
6	130.79	0	5.48	0
7	212.91	0	10.70	0
8	74.86	0	8.51	0
9	156.59	0	10.15	0
10	191.42	0	4.75	0
11	230.94	0	5.17	0
12	195.39	0	8.64	0
13	191.27	0	6.01	0
14	178.33	0	6.63	0
15	178.33	0	6.40	0

V. KESIMPULAN

Penelitian ini berhasil mengembangkan sistem parkir pintar berbasis IoT yang mengintegrasikan ESP32-S3, Firebase *Realtime Database*, WS2812B RGB LED, dan aplikasi *mobile* berbasis Flutter untuk menyediakan pembaruan status parkir secara real-time. Pengujian *Quality of Service* (QoS) menunjukkan bahwa sistem memiliki rata-rata latensi end-to-end sebesar 127,45 ms, dengan *throughput* mencapai 6,5 Kbps, serta tidak ditemukan *packet loss* maupun retransmission selama pengujian. Hasil ini menunjukkan bahwa sistem mampu beroperasi dengan stabil dan efisien, memastikan bahwa setiap perubahan status parkir dapat ditampilkan secara akurat baik pada aplikasi maupun indikator LED di lapangan.

Dari hasil pengujian, dapat disimpulkan bahwa sistem ini memiliki performa yang cukup baik dalam menangani pembaruan data secara *real-time* dan dapat diimplementasikan dalam lingkungan parkir kampus untuk meningkatkan efisiensi dan kenyamanan pengguna. Dengan komunikasi yang handal dan responsif, sistem ini dapat membantu mengurangi waktu pencarian parkir serta mengoptimalkan pemanfaatan lahan parkir. Meskipun demikian, terdapat potensi peningkatan lebih lanjut, seperti optimasi pemrosesan event Firebase atau penerapan teknik kompresi data untuk meningkatkan efisiensi transmisi informasi.

REFERENSI

[1] Espressif Systems, “ESP32-S3 Series Datasheet v1.9,” 2024. Accessed: Feb. 19, 2025. [Online]. Available: https://www.espressif.com/documentation/esp32-s3_datasheet_en.pdf

[2] World Semi, “WS2812 Intelligent Control LED Integrated Light Source,” 2007. [Online]. Available: http://www.world-semi.com

[3] Google Firebase, “Firebase Realtime Database Documentation,” Google. Accessed: Feb. 19, 2025. [Online]. Available: 2024

[4] Google Firebase, “Firebase Authentication Documentation,” Google. Accessed: Feb. 19, 2025. [Online]. Available: https://firebase.google.com/docs/auth

[5] M. Singh and G. Baranwal, “Quality of Service (QoS) in Internet of Things,” in *Proceedings - 2018 3rd International Conference On Internet of Things: Smart Innovation and Usages, IoT-SIU 2018*, Institute of Electrical and Electronics Engineers Inc., Nov. 2018. doi: 10.1109/IoT-SIU.2018.8519862.