BAB IV
PENGUJIAN DAN ANALISA

4.1 Tujuan Pengujian

Pada bab ini akan dilakukan pengujian dan pengukuran alat kerja yang bertujuan untuk mengevaluasi fungsionalitas, akurasi, dan performa prototipe gerbang tol *free flow* yang telah dirancang, guna memastikan seluruh mekanisme transaksi dan sistem pengolahan data kendaraan berbasis Firebase dapat beroperasi sesuai dengan spesifikasi yang telah direncanakan. Melalui pengujian terstruktur ini, tingkat keberhasilan integrasi antara perangkat keras pengidentifikasi kendaraan dengan *real-time database* akan diukur untuk mengidentifikasi potensi keterlambatan (*delay*) pengiriman data, memvalidasi validitas pencatatan transaksi, serta mendeteksi *error* pada sistem. Seluruh nilai rata-rata pada tabel-tabel hasil pengujian di bab ini dihitung menggunakan Persamaan (3.3), yaitu jumlah seluruh data hasil percobaan dibagi dengan jumlah percobaan yang dilakukan. Dengan demikian, data pengujian yang diperoleh dapat menjadi landasan objektif untuk menganalisis keandalan prototipe sistem secara menyeluruh serta menentukan efektivitas implementasi teknologi berbasis IoT ini dalam mendukung visualisasi dan pengolahan data lalu lintas tol yang dinamis.

4.2 Pengujian Konektivitas ESP32 ke Firebase

Pengujian konektivitas ESP32 ke Firebase dibagi menjadi dua bagian. Bagian pertama membahas pengujian koneksi awal untuk mengukur kekuatan sinyal *Wi-Fi* (RSSI), waktu koneksi ke jaringan *Wi-Fi*, serta waktu koneksi ke Firebase Realtime Database saat sistem pertama kali dinyalakan. Bagian kedua membahas pengujian kekuatan sinyal *Wi-Fi* saat sistem sedang melakukan proses pengiriman data transaksi secara langsung, untuk mengetahui apakah kondisi sinyal turut memengaruhi keberhasilan pengiriman data pada kondisi operasional yang sesungguhnya.

4.2.1 Pengujian Konektivitas Awal

Pengujian konektivitas ESP32 ke Firebase ini dilakukan sebanyak 10 kali percobaan untuk mengukur kekuatan sinyal *Wi-Fi* (RSSI), waktu koneksi ke jaringan *Wi-Fi*, serta waktu koneksi ke Firebase Realtime Database, dengan hasil yang ditunjukkan pada **Tabel 4.1**. Tingkat keberhasilan koneksi dihitung menggunakan Persamaan (3.2), yaitu perbandingan jumlah percobaan yang berhasil terhubung dengan total percobaan yang dilakukan.

Jaringan *Wi-Fi* yang digunakan pada seluruh rangkaian pengujian ini bukan berasal dari *access point* tetap seperti router, melainkan memanfaatkan fitur *hotspot* (*tethering*) pada *smartphone*. Pemilihan ini didasarkan pada pertimbangan kepraktisan pengujian, mengingat lokasi prototipe tidak selalu memiliki akses ke jaringan *Wi-Fi* tetap, sehingga *hotspot smartphone* dipilih sebagai sumber koneksi internet portabel yang lebih fleksibel. Karakteristik ini turut menjadi salah satu faktor yang perlu dipertimbangkan dalam menginterpretasikan nilai RSSI dan waktu koneksi pada hasil pengujian, mengingat performa *smartphone* dapat sedikit berbeda dengan jaringan *Wi-Fi* dari router tetap, baik dari segi stabilitas maupun kekuatan sinyal yang dipancarkan.

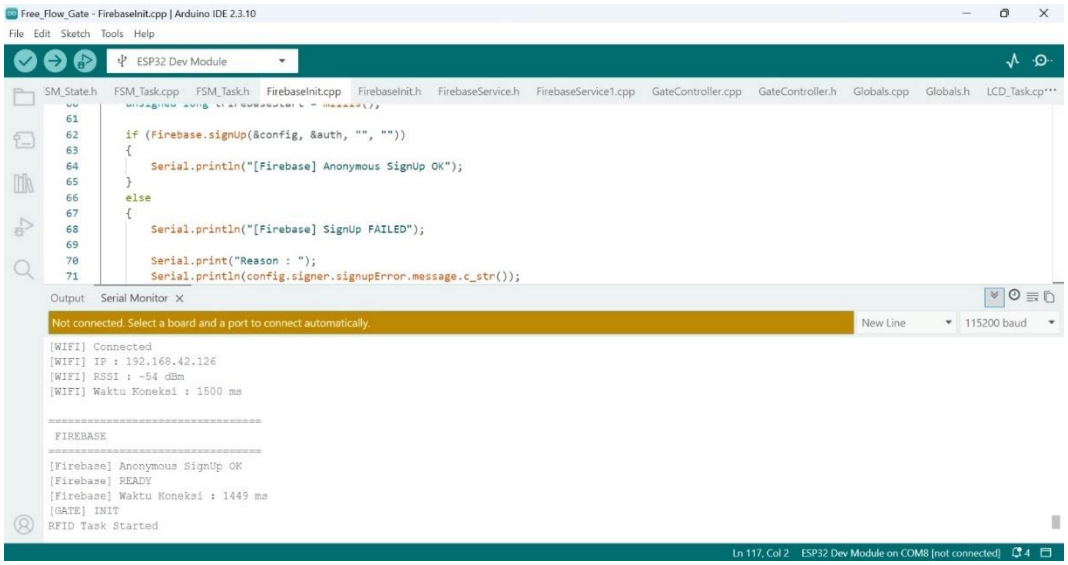
Tabel 4. 1 Hasil Pengujian Konektivitas ESP32 ke Firebase

Percobaan	*Wi-Fi* RSSI	ESP - *Wi-Fi* (s)	ESP - Firebase (s)	Status Koneksi
1	-54	1,5	1,449	Berhasil
2	-55	1,5	1,449	Berhasil
3	-62	1,499	1,414	Berhasil
4	-53	1,5	1,896	Berhasil
5	-54	1,5	1,862	Berhasil
6	-56	1,5	1,180	Berhasil
7	-52	1,499	1,133	Berhasil
8	-56	1,5	1,667	Berhasil
9	-60	1,5	2,166	Berhasil
10	-59	1,5	2,289	Berhasil
Rata-Rata	**-56,1**	**1,4998**	**1,6505**	**100%**

Sumber: Hasil Pengujian Penulis

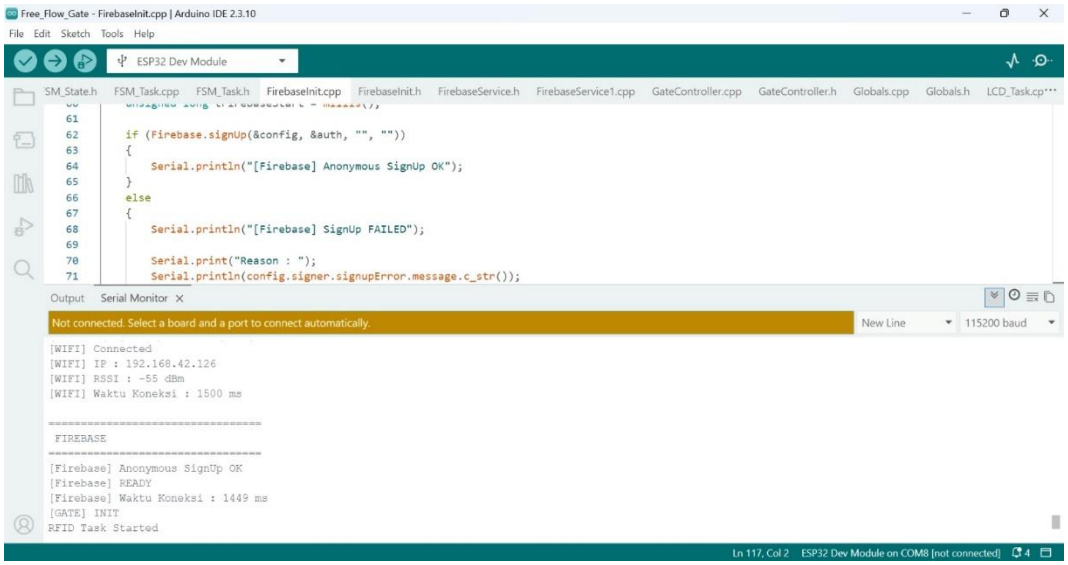
Berdasarkan **Tabel 4.1**, seluruh percobaan (10 dari 10) berhasil terhubung baik ke *Wi-Fi* maupun ke Firebase, sehingga tingkat keberhasilan koneksi mencapai 100%, dengan rata-rata RSSI sebesar -56,1 dBm, waktu koneksi *Wi-Fi* rata-rata 1,49 s, dan waktu koneksi Firebase rata-rata 1,65 s. Nilai RSSI rata-rata tersebut berada pada rentang -50 dBm hingga -60 dBm yang secara umum tergolong kategori baik, hal ini sejalan dengan waktu koneksi *Wi-Fi* yang sangat konsisten pada setiap percobaan, yang menandakan proses *handshake* ESP32 dengan *access point* berjalan stabil tanpa dipengaruhi fluktuasi kondisi jaringan lokal. Berbeda dengan waktu koneksi *Wi-Fi*, waktu koneksi ke Firebase menunjukkan variasi yang jauh lebih besar, yaitu antara 1,13 s (percobaan ke-7) hingga 2,28 s (percobaan ke-10), dengan selisih mencapai 1,15 s, variasi ini tidak berkorelasi langsung dengan kekuatan RSSI. Pada percobaan ke-3, RSSI tercatat paling lemah (-62 dBm) tetapi waktu koneksi Firebase justru relatif cepat (1,41 s), sementara pada percobaan ke-4 dan ke-5, RSSI tercatat lebih baik (-53 dBm dan -54 dBm) tetapi waktu koneksi Firebase justru lebih lambat (1,89 s dan 1,86 s), ini mengindikasikan bahwa lamanya waktu koneksi ke Firebase lebih dipengaruhi oleh faktor di luar kekuatan sinyal *Wi-Fi* lokal, seperti latensi jaringan internet menuju *server* Firebase, waktu respons proses otentikasi pada sisi *cloud*, maupun kondisi lalu lintas data pada jaringan internet penyedia layanan saat pengujian berlangsung. Jika dibandingkan dengan waktu pelayanan transaksi sistem SLFF/Flo sebesar 0,6 detik per kendaraan [47], waktu inisialisasi koneksi ESP32 ke Firebase pada penelitian tugas akhir ini (total rata-rata sekitar 3,15 s) memang tampak lebih lambat, namun kedua metrik ini mengukur hal yang berbeda, yaitu waktu pelayanan transaksi pada sistem yang sudah berjalan (*steady state*) dibandingkan waktu inisialisasi awal (*bootstrapping*) yang hanya terjadi satu kali saat sistem dinyalakan dan tidak berulang pada setiap kendaraan yang melintas, sehingga tidak akan mempengaruhi kecepatan pelayanan transaksi per kendaraan setelah sistem berada pada kondisi *standby*. Keberhasilan koneksi 100% pada seluruh percobaan ini juga menjawab keterbatasan pada penelitian-penelitian sebelumnya yang masih mengandalkan *database* lokal seperti Microsoft Access [9] dan Raspberry Pi [48] yang tidak mendukung sinkronisasi data secara *real-time* melalui *cloud*, sehingga hasil pengujian ini membuktikan

bahwa integrasi ESP32 dengan Firebase Realtime Database pada penelitian tugas akhir ini mampu berjalan secara konsisten dan andal. Bukti keberhasilan koneksi tersebut turut didukung oleh catatan *log Serial Monitor* pada **Gambar 4.1** dan **Gambar 4.2**, yang menunjukkan status koneksi *Wi-Fi* dan Firebase secara berurutan pada saat pengujian berlangsung.



Gambar 4. 1 Log *Serial* Monitor Koneksi *Wi-Fi* dan Firebase (Percobaan ke-1)
Sumber: Dokumentasi Penulis

Berdasarkan **Gambar 4.1**, ditampilkan *log serial monitor* koneksi saat pengambilan data dilakukan. Selanjutnya, ditampilkan pada **Gambar 4.2**.



Gambar 4. 2 Log *Serial Monitor* Koneksi *Wi-Fi* dan Firebase (Percobaan ke-2)
Sumber: Dokumentasi Penulis

Berdasarkan **Gambar 4.1** dan **Gambar 4.2**, terlihat bahwa ESP32 berhasil mencatat status koneksi *Wi-Fi* yang tersambung beserta nilai RSSI dan waktu koneksi, yang kemudian diikuti oleh status koneksi Firebase yang tersambung beserta waktu koneksinya masing-masing. Kesesuaian antara nilai yang tercatat pada *log Serial Monitor* dengan data yang direkap pada **Tabel 4.1** menunjukkan bahwa proses pengambilan data pengujian telah dilakukan secara konsisten dan validitasnya dapat dipertanggungjawabkan.

4.2.2 Pengujian Kekuatan Sinyal (RSSI) Selama Proses Transaksi

Perlu ditegaskan bahwa label UID seperti MOBIL0, MOBIL1, MOBIL2, dan MOBIL3 yang tercantum pada tabel pengujian berikut dan tabel-tabel pengujian selanjutnya merupakan penamaan internal yang digunakan penulis untuk merepresentasikan masing-masing *RFID tag* uji coba, bukan merujuk pada merek atau jenis kendaraan yang sesungguhnya. Penamaan ini digunakan untuk mempermudah proses identifikasi dan pelacakan data transaksi selama pengujian berlangsung. Selain pengujian awal koneksi ESP32 ke *Wi-Fi* dan Firebase pada **Tabel 4.1**, penulis juga melakukan pencatatan nilai RSSI secara berkelanjutan saat sistem benar-benar melakukan proses pengiriman data transaksi, bukan hanya pada tahap inisialisasi koneksi. Pengujian ini dilakukan sebanyak 18 kali percobaan dengan skenario yang bervariasi, meliputi transaksi berhasil, transaksi gagal akibat saldo tidak mencukupi, hingga transaksi gagal akibat UID kendaraan yang tidak terdaftar pada *database*, dengan tujuan untuk mengetahui apakah kekuatan sinyal *Wi-Fi* turut memengaruhi keberhasilan proses pengiriman data ke Firebase pada kondisi operasional yang sesungguhnya. Hasil pengujian ditunjukkan pada **Tabel 4.2**.

Sama seperti pengujian sebelumnya, jaringan *Wi-Fi* yang digunakan tetap menggunakan *hotspot smartphone*, dan seluruh percobaan dilakukan dalam rentang waktu yang berdekatan (sekitar pukul 19.10-19.20 WIB), sehingga kondisi lingkungan pengujian relatif konsisten antarpercobaan.

Tabel 4. 2 Hasil Pengujian Kekuatan Sinyal (RSSI) Selama Proses Transaksi Pengiriman Data

Percobaan	Waktu	UID	Nama	Tarif	Saldo Awal	Saldo Akhir	Status	*Wi-Fi* RSSI (dBm)
1	2026/07/27 19:19:57	MOBIL2	ceo	5000	50	50	Gagal saldo kurang	-57
2	2026/07/27 19:19:49	MOBIL2	ceo	5000	50	50	Gagal saldo kurang	-59
3	2026/07/27 19:19:44	MOBIL3	dava	5000	98415000	98410000	Berhasil	-57
4	2026/07/27 19:17:43	MOBIL2	ceo	5000	50	50	Gagal saldo kurang	-58
5	2026/07/27 19:17:39	MOBIL3	dava	5000	98420000	98415000	Berhasil	-60
6	2026/07/27 19:17:10	MOBIL2	ceo	5000	50	50	Gagal saldo kurang	-58
7	2026/07/27 19:17:03	MOBIL3	dava	5000	98425000	98420000	Berhasil	-60
8	2026/07/27 19:16:48	MOBIL1	ell	5000	450000	445000	Berhasil	-59
9	2026/07/27 19:16:39	MOBIL3	dava	5000	98430000	98425000	Berhasil	-57
10	2026/07/27 19:16:28	MOBIL3	dava	5000	98435000	98430000	Berhasil	-60
11	2026/07/27 19:16:23	MOBIL1	ell	5000	455000	450000	Berhasil	-59
12	2026/07/27 19:15:46	MOBIL1	ell	5000	460000	455000	Berhasil	-52
13	2026/07/27 19:15:40	MOBIL3	dava	5000	98440000	98435000	Berhasil	-52

Percobaan	Waktu	UID	Nama	Tarif	Saldo Awal	Saldo Akhir	Status	*Wi-Fi* RSSI (dBm)
14	2026/07/27 19:15:30	MOBIL2	ceo	5000	50	50	Gagal saldo kurang	-54
15	2026/07/27 19:11:35	MOBIL1	ell	5000	465000	460000	Berhasil	-52
16	2026/07/27 19:11:17	MOBIL0	-	5000	-	-	Gagal tidak terdaftar	-52
17	2026/07/27 19:10:56	MOBIL3	dava	5000	98450000	98445000	Berhasil	-54
18	2026/07/27 19:10:48	MOBIL3	dava	5000	98455000	98450000	Berhasil	-53
Rata-Rata								**-56,28**

Sumber Hasil Pengujian Penulis

Berdasarkan **Tabel 4.2**, dari 18 kali percobaan pengiriman data, nilai RSSI yang tercatat berada pada rentang -52 dBm hingga -60 dBm dengan rata-rata sebesar -56,28 dBm, hampir identik dengan rata-rata RSSI pada pengujian konektivitas awal pada **Tabel 4.1** (-56,1 dBm). Kesamaan nilai ini menunjukkan bahwa kekuatan sinyal *hotspot smartphone* yang digunakan relatif stabil dari waktu ke waktu, meskipun sifatnya portabel dan berbeda dengan *access point* tetap seperti router. Jika mengacu pada klasifikasi umum RSSI [57], rentang -52 dBm hingga -60 dBm tergolong dalam kategori baik (*good*), sehingga secara keseluruhan kondisi jaringan selama pengujian tergolong mendukung pengiriman data secara *real-time*.

Temuan menarik yang perlu dianalisis lebih lanjut adalah ketiadaan korelasi antara kekuatan RSSI dan keberhasilan pengiriman data. Pada percobaan ke-5 dan ke-7, misalnya, RSSI tercatat paling lemah pada rentang pengujian ini (-60 dBm), namun transaksi tetap berhasil dan data tetap terkirim serta tersimpan di Firebase. Sebaliknya, pada percobaan ke-16, RSSI justru tercatat paling kuat (-52 dBm), tetapi transaksi berstatus gagal karena UID kendaraan (MOBIL0) tidak terdaftar pada *database*. Begitu pula pada percobaan ke-1, ke-2, ke-4, ke-6, dan ke-14 yang

seluruhnya berstatus “gagal saldo kurang” meskipun RSSI pada percobaan tersebut berada pada rentang -54 dBm hingga -59 dBm, yang tidak jauh berbeda dari percobaan-percobaan lain yang berhasil. Temuan ini mengindikasikan bahwa status kegagalan transaksi pada pengujian ini bukan disebabkan oleh lemahnya sinyal *Wi-Fi* atau kegagalan pengiriman data ke Firebase, melainkan murni disebabkan oleh logika validasi pada sisi sistem, yaitu pengecekan saldo dan status pendaftaran UID kendaraan. Dengan kata lain, sistem mampu mengirimkan dan mencatat data transaksi secara konsisten ke Firebase pada seluruh 18 percobaan, terlepas dari hasil validasi yang diperoleh (berhasil maupun gagal), yang membuktikan bahwa reliabilitas jalur komunikasi ESP32-Firebase tidak terganggu oleh fluktuasi RSSI selama nilainya masih berada pada kategori baik. Temuan ini sejalan dengan hasil analisis statistik RSSI pada sistem berbasis WLAN *indoor* [57] yang menyatakan bahwa performa transmisi data cenderung stabil selama RSSI belum menyentuh ambang batas sinyal lemah, serta memperkuat hasil pengujian pada **Tabel 4.1** yang juga menunjukkan bahwa variasi kekuatan sinyal tidak berkorelasi langsung dengan waktu maupun keberhasilan koneksi ke Firebase.

4.3 Pengujian Identifikasi Kendaraan

Pengujian Identifikasi Kendaraan ini bertujuan untuk mengetahui tingkat akurasi pembacaan *RFID Tag* oleh modul UHF RFID *Reader* EL-UHF-RMT01 serta kecepatan proses pengiriman hasil identifikasi tersebut dari ESP32 ke Firebase Realtime Database. Pengujian dilakukan sebanyak 10 kali percobaan dengan cara mendekatkan 4 *RFID Tag* yang mewakili kendaraan berbeda (MOBIL0 hingga MOBIL3) secara acak ke modul *reader*, di mana MOBIL0 merepresentasikan kondisi UID yang tidak terdaftar pada *database*, sedangkan MOBIL1, MOBIL2, dan MOBIL3 merepresentasikan kendaraan terdaftar dengan kondisi saldo yang bervariasi. Skenario ini sengaja dirancang untuk mencakup 3 kemungkinan status keluaran sistem, yaitu identifikasi berhasil, gagal karena UID tidak terdaftar, dan gagal karena saldo tidak mencukupi, sebagaimana telah didefinisikan dalam alur *Finite State Machine* (FSM) pada sistem. Parameter yang diukur pada pengujian ini meliputi kesesuaian antara UID yang didekatkan dengan UID yang berhasil dibaca,

status hasil identifikasi, waktu proses dari pembacaan RFID hingga data tersimpan di Firebase, serta kesesuaian data yang tercatat pada *database* dengan kondisi aktual pengujian. Tingkat keberhasilan identifikasi UID dihitung menggunakan Persamaan (3.2), yaitu perbandingan jumlah percobaan yang berhasil terhubung dengan total percobaan yang dilakukan. Hasil pengujian ditunjukkan pada **Tabel 4.3**.

Tabel 4. 3 Hasil Pengujian Identifikasi Kendaraan

Percobaan	UID yang Didekatkan	UID yang Terbaca	Status	RFID-Firebase (s)	Hasil Database
1	MOBIL1	MOBIL1	Berhasil	0,441	Sesuai
2	MOBIL2	MOBIL2	Berhasil	0,604	Sesuai
3	MOBIL0	MOBIL0	Gagal tidak terdaftar	1,331	Sesuai
4	MOBIL3	MOBIL3	Gagal saldo kurang	2,492	Sesuai
5	MOBIL1	MOBIL1	Berhasil	1,713	Sesuai
6	MOBIL0	MOBIL0	Gagal tidak terdaftar	0,446	Sesuai
7	MOBIL2	MOBIL2	Berhasil	0,306	Sesuai
8	MOBIL3	MOBIL3	Gagal saldo kurang	0,514	Sesuai
9	MOBIL3	MOBIL3	Gagal saldo kurang	1,432	Sesuai
10	MOBIL0	MOBIL0	Gagal tidak terdaftar	0,394	Sesuai

Percobaan	UID yang Didekatkan	UID yang Terbaca	Status	RFID-Firebase (s)	Hasil Database
Rata-rata				**0,9673**	**100%**

Sumber: Hasil Pengujian Penulis

Berdasarkan **Tabel 4.3**, dari 10 kali percobaan, seluruh UID yang didekatkan berhasil terbaca dengan tepat oleh modul UHF RFID *Reader* tanpa satu pun kesalahan pembacaan, sehingga tingkat akurasi identifikasi mencapai 100%. Kesesuaian ini juga berlaku pada kolom Hasil *Database*, di mana seluruh status hasil identifikasi (baik berhasil, gagal tidak terdaftar, maupun gagal saldo kurang) yang tercatat pada Firebase sesuai dengan kondisi yang seharusnya terjadi pada setiap skenario pengujian, tanpa ditemukan satu pun kondisi status yang salah tercatat, baik kendaraan yang seharusnya berhasil tercatat sebagai gagal maupun sebaliknya. Hal ini menunjukkan bahwa logika validasi pada sisi sistem, mulai dari pencocokan UID, pengecekan status pendaftaran, hingga pengecekan saldo, telah bekerja secara konsisten selama pengujian berlangsung. Dari sisi distribusi skenario, pengujian ini mencakup 4 kali status berhasil, 3 kali status gagal akibat UID tidak terdaftar dan 3 kali status gagal akibat saldo kurang, sehingga ketiga jalur logika utama pada sistem identifikasi telah teruji seluruhnya.

Dari sisi waktu proses, rata-rata waktu yang dibutuhkan mulai dari RFID membaca *tag* hingga data tersimpan di Firebase adalah 0,97 s, dengan rentang waktu tercepat 0,306 s (percobaan ke-7) dan terlama 2,49 s (percobaan ke-4), atau selisih hingga 2,18 s antarpercobaan. Variasi ini tergolong cukup besar apabila dibandingkan dengan kecepatan baca modul EL-UHF-RMT01 itu sendiri, yang menurut spesifikasi pada **Tabel 2.5** mampu membaca lebih dari 50 *tag* per detik, atau setara dengan kurang dari 0,02 s untuk satu kali pembacaan. Dengan demikian, proses pembacaan RFID bukan merupakan kontributor utama terhadap total waktu yang tercatat, melainkan proses pengiriman dan penulisan data ke Firebase Realtime Database yang lebih banyak memengaruhi durasi keseluruhan, sejalan dengan temuan pada **Tabel 4.1** dan **Tabel 4.2** sebelumnya yang juga menunjukkan variasi waktu respons Firebase yang cukup signifikan antarpercobaan.

Jika data waktu proses dikelompokkan berdasarkan status hasil identifikasi, status berhasil memiliki rata-rata 0,76 s (percobaan 1, 2, 5, dan 7), status gagal tidak

terdaftar memiliki rata-rata 0,72 s (percobaan 3, 6, dan 10), sedangkan status gagal saldo kurang memiliki rata-rata tertinggi, yaitu 1,47 s (percobaan 4, 8, dan 9). Secara logika sistem, hal ini cukup masuk akal karena proses validasi saldo memerlukan langkah tambahan berupa pembacaan nilai saldo pengguna pada *node database* setelah UID dinyatakan terdaftar, sedangkan proses gagal tidak terdaftar dapat langsung dihentikan begitu UID tidak ditemukan dalam daftar kendaraan terdaftar, tanpa melakukan pembacaan data saldo. Namun demikian, pola ini tidak sepenuhnya konsisten pada seluruh percobaan, terlihat dari percobaan ke-8 yang berstatus gagal saldo kurang, namun hanya membutuhkan waktu 0,51 s, lebih cepat dibandingkan beberapa percobaan berstatus berhasil, sedangkan percobaan ke-4 pada status yang sama justru menjadi percobaan dengan waktu paling lambat pada keseluruhan pengujian. Kondisi ini mengindikasikan bahwa selain jumlah langkah validasi pada sisi logika sistem, durasi waktu proses turut dipengaruhi oleh faktor lain di luar kendali langsung sistem, seperti kondisi jaringan internet dan waktu respons *server* Firebase saat data dikirimkan, sebagaimana juga telah diidentifikasi dalam analisis pada **Tabel 4.1** dan **Tabel 4.2** sebelumnya.

Keberhasilan pembacaan UID sebesar 100% pada seluruh percobaan ini turut menjawab temuan [50] yang menyatakan bahwa keandalan pembacaan *tag* kendaraan pada sistem *Automatic Lane Barrier* (ALB) sangat dipengaruhi oleh konfigurasi daya dan sudut antena RFID, sehingga hasil pengujian ini membuktikan bahwa penempatan dan konfigurasi modul UHF RFID *Reader* pada prototipe tugas akhir ini telah cukup optimal untuk mendukung proses identifikasi kendaraan pada jarak baca yang telah ditentukan. Hasil ini juga sejalan dengan pernyataan [24] bahwa RFID memiliki keunggulan dalam kecepatan serta mengurangi kemungkinan kesalahan pembacaan dibandingkan dengan metode identifikasi konvensional seperti *barcode*. Selain itu, konsistensi status yang tercatat pada Firebase di seluruh percobaan turut memperkuat hasil penelitian [59] pada sistem palang pintu RFID berbasis ESP32 dengan integrasi *cloud database*, yang menyatakan bahwa seluruh data akses kendaraan dapat tercatat secara *real-time* dengan tingkat akurasi validasi yang tinggi, meskipun keandalan pencatatan tersebut tetap bergantung pada kestabilan koneksi jaringan pada sisi perangkat. Jika

dibandingkan dengan waktu pelayanan transaksi sistem SLFF/Flo sebesar 0,6 detik per kendaraan [47], rata-rata waktu identifikasi hingga penyimpanan data pada pengujian ini (0,97 detik) memang masih lebih lambat, angka tersebut baru mencakup tahapan identifikasi dan penulisan data ke Firebase, belum termasuk tahapan validasi saldo dan pembaruan status transaksi secara menyeluruh. Meski demikian, tingkat akurasi identifikasi sebesar 100% pada seluruh percobaan menunjukkan bahwa integrasi UHF RFID *Reader* EL-UHF-RMT01 dengan ESP32 dan Firebase Realtime Database pada prototipe ini mampu melakukan proses identifikasi kendaraan secara andal.

4.4 Pengujian Validasi Saldo dan Transaksi

Pengujian ini bertujuan untuk mengetahui keandalan logika validasi saldo pada sistem serta mengukur waktu komunikasi dua arah antara ESP32 dan Firebase Realtime Database selama proses transaksi berlangsung, yaitu waktu pengambilan data dari Firebase ke ESP32 dan waktu pengiriman data hasil transaksi dari ESP32 ke Firebase. Pengujian dilakukan sebanyak 37 kali percobaan dengan skenario yang dirancang untuk mencakup 4 kondisi saldo, yaitu saldo lebih, saldo pas, saldo kurang, dan UID tidak terdaftar. Keempat kondisi tersebut direpresentasikan melalui 3 *tag* kendaraan terdaftar dan 1 *tag* tidak terdaftar, yaitu MOBIL1 (ell) dengan saldo besar yang terus berkurang seiring banyaknya transaksi untuk merepresentasikan kondisi saldo lebih, MOBIL2 (ceo) yang saldonya sengaja diisi ulang senilai persis dengan tarif (Rp5.000) sebelum setiap percobaan untuk merepresentasikan kondisi saldo pas, MOBIL3 (dava) yang diuji pada dua kondisi berbeda yaitu saldo sedikit di atas tarif (Rp6.000) dan saldo kurang, serta MOBIL0 yang tidak didaftarkan pada *database* untuk merepresentasikan kondisi UID tidak terdaftar. Pada skenario saldo pas, ketika transaksi MOBIL2 berhasil, saldo pada *tag* akan terpotong penuh sehingga tersisa Rp0. Apabila *tag* yang sama didekatkan kembali sebelum sempat diisi ulang oleh penulis, sistem akan mencatat status gagal saldo kurang karena saldo yang tersedia Rp0 sudah berada di bawah tarif. Hasil pengujian selengkapnya ditunjukkan pada **Tabel 4.4**.

Tabel 4. 4 Hasil Pengujian Validasi Saldo dan Transaksi (Saldo Lebih, Saldo Pas, Saldo Kurang, Tidak Terdaftar)

Percobaan	UID	Nama	Tarif	Saldo Awal	Saldo Akhir	Status	Firebase – ESP32 (s)	ESP32 - Firebase (s)
1	MOBIL2	ceo	5000	0	0	Gagal saldo kurang	0,409	0,814
2	MOBIL2	ceo	5000	5000	0	Berhasil	0,410	0,410
3	MOBIL1	ell	5000	235000	230000	Berhasil	0,410	1,432
4	MOBIL3	dava	5000	0	0	Gagal saldo kurang	1,434	0,407
5	MOBIL0	-	5000	-	-	Gagal tidak terdaftar	0,615	1,175
6	MOBIL2	ceo	5000	0	0	Gagal saldo kurang	0,820	0,404
7	MOBIL2	ceo	5000	5000	0	Berhasil	0,406	0,613
8	MOBIL1	ell	5000	240000	235000	Berhasil	0,420	0,407
9	MOBIL1	ell	5000	245000	240000	Berhasil	0,367	0,408
10	MOBIL3	dava	5000	0	0	Gagal saldo kurang	0,409	0,403
11	MOBIL2	ceo	5000	5000	0	Berhasil	0,417	0,414
12	MOBIL1	ell	5000	250000	245000	Berhasil	1,433	0,408
13	MOBIL1	ell	5000	255000	250000	Berhasil	0,412	0,408
14	MOBIL0	-	5000	-	-	Gagal tidak terdaftar	0,409	1,164
15	MOBIL2	ceo	5000	0	0	Gagal saldo kurang	0,413	0,4
16	MOBIL3	dava	5000	6000	1000	Berhasil	0,408	0,409

Percobaan	UID	Nama	Tarif	Saldo Awal	Saldo Akhir	Status	Firebase – ESP32 (s)	ESP32 - Firebase (s)
17	MOBIL2	ceo	5000	5000	0	Berhasil	0,406	0,612
18	MOBIL1	ell	5000	260000	255000	Berhasil	0,409	0,611
19	MOBIL1	ell	5000	265000	260000	Berhasil	0,367	0,408
20	MOBIL2	ceo	5000	0	0	Gagal saldo kurang	0,409	0,405
21	MOBIL2	ceo	5000	5000	0	Berhasil	0,408	0,409
22	MOBIL1	ell	5000	270000	265000	Berhasil	0,404	0,613
23	MOBIL1	ell	5000	275000	270000	Berhasil	0,410	0,415
24	MOBIL3	dava	5000	0	0	Gagal saldo kurang	0,410	0,610
25	MOBIL2	ceo	5000	5000	0	Berhasil	0,406	0,412
26	MOBIL1	ell	5000	280000	275000	Berhasil	0,409	0,408
27	MOBIL1	ell	5000	285000	280000	Berhasil	0,411	0,404
28	MOBIL0	-	5000	-	-	Gagal tidak terdaftar	0,376	1,576
29	MOBIL0	-	5000	-	-	Gagal tidak terdaftar	0,409	1,372
30	MOBIL1	ell	5000	290000	285000	Berhasil	0,413	0,410
31	MOBIL2	ceo	5000	5000	0	Berhasil	0,818	0,412
32	MOBIL3	dava	5000	6000	1000	Berhasil	0,611	0,410
33	MOBIL1	ell	5000	295000	290000	Berhasil	0,412	0,408
34	MOBIL0	-	5000	-	-	Gagal tidak terdaftar	0,616	1,367
35	MOBIL2	ceo	5000	5000	0	Berhasil	0,407	0,411

Percobaan	UID	Nama	Tarif	Saldo Awal	Saldo Akhir	Status	Firebase – ESP32 (s)	ESP32 - Firebase (s)
36	MOBIL3	dava	5000	0	0	Gagal saldo kurang	0,411	0,626
37	MOBIL1	ell	5000	300000	295000	Berhasil	0,409	0,408
Rata-Rata							**0,5**	**0,6**

Sumber: Hasil Pengujian Penulis

Untuk melengkapi data numerik pada **Tabel 4.4**, penulis juga mendokumentasikan kondisi fisik prototipe saat pengujian, mencakup tampilan LCD dan status palang, untuk 3 skenario yang mewakili kondisi berbeda. Dokumentasi skenario saldo lebih (MOBIL1) ditunjukkan pada **Gambar 4.3**.

Gambar 4. 3 Tampilan LCD pada Skenario Saldo Lebih (MOBIL1)
Sumber: Dokumentasi Penulis

Berdasarkan **Gambar 4.3**, LCD menampilkan keterangan nama pengguna, saldo awal, tarif, dan sisa saldo secara lengkap, yang menunjukkan bahwa sistem berhasil mengambil data saldo dari Firebase, melakukan perhitungan tarif, dan menampilkan hasil pemotongan secara langsung kepada pengguna. Dokumentasi berikutnya diambil pada skenario saldo pas (MOBIL2), sebagaimana ditunjukkan pada **Gambar 4.4**.

Gambar 4. 4 Tampilan LCD pada Skenario Saldo Pas (MOBIL2)
Sumber: Dokumentasi Penulis

Kondisi serupa juga terlihat pada **Gambar 4.4**, di mana LCD menampilkan saldo, tarif, dan sisa, membuktikan bahwa sistem tetap dapat memproses transaksi dengan benar meskipun saldo pengguna hanya cukup untuk membayar tarif, tanpa harus memiliki saldo lebih. Terakhir, dokumentasi pada skenario UID tidak terdaftar (MOBIL0) ditunjukkan pada **Gambar 4.5**.

Gambar 4. 5 Tampilan LCD pada Skenario UID Tidak Terdaftar (MOBIL0)
Sumber: Dokumentasi Penulis

Berbeda dengan 2 kondisi sebelumnya, pada **Gambar 4.5** LCD hanya menampilkan pesan "RFID TIDAK TERDAFTAR" tanpa rincian saldo maupun tarif, yang menunjukkan bahwa proses validasi dihentikan lebih awal pada tahap pengecekan status pendaftaran UID, sebelum sistem sempat melakukan pengambilan data saldo dari Firebase.

Berdasarkan **Tabel 4.4**, dari 37 kali percobaan diperoleh 24 kali status berhasil, 8 kali status gagal saldo kurang, dan 5 kali status gagal tidak terdaftar, dengan seluruh status tercatat sesuai dengan kondisi saldo dan pendaftaran yang telah dirancang pada setiap skenario. Pada kelompok pengujian MOBIL1 (ell) yang merepresentasikan kondisi saldo lebih, nilai saldo awal tercatat menurun secara konsisten dengan kelipatan tarif (Rp5.000) pada setiap percobaan berikutnya, mulai

dari Rp300.000 pada percobaan (ke-37 berdasarkan urutan waktu) hingga Rp235.000 pada percobaan terakhir (percobaan ke-3), tanpa satu pun selisih yang tidak sesuai dengan tarif yang berlaku. Pola ini membuktikan bahwa proses pemotongan saldo pada Firebase berjalan akurat dan konsisten, serta tidak terjadi pemotongan ganda maupun kesalahan pencatatan meskipun pengujian dilakukan berulang kali dalam rentang waktu yang berdekatan. Pada kelompok pengujian MOBIL2 (ceo) yang merepresentasikan kondisi saldo pas, seluruh percobaan dengan saldo awal Rp5.000 (persis sama dengan tarif) tercatat berhasil dengan saldo akhir Rp0, yang menunjukkan bahwa sistem menerapkan aturan validasi saldo cukup atau lebih besar sama dengan tarif (bukan harus lebih besar dari tarif), sehingga kondisi batas (saldo pas) tetap dapat diproses sebagai transaksi yang sah. Adapun status gagal saldo kurang yang tercatat pada MOBIL2 dan MOBIL3 seluruhnya terjadi ketika saldo pada *tag* dalam keadaan kosong (Rp0), yaitu ketika pengujian dilakukan berulang tanpa pengisian saldo terlebih dahulu, sehingga hasil ini turut memastikan bahwa sistem secara konsisten menolak transaksi begitu saldo tidak lagi mencukupi.

Dari sisi waktu komunikasi, rata-rata waktu pengambilan data dari Firebase ke ESP32 tercatat sebesar 0,5 s dan rata-rata waktu pengiriman data dari ESP32 ke Firebase sebesar 0,6 s. Apabila data waktu tersebut dikelompokkan berdasarkan status hasil transaksi, diperoleh pola sebagai berikut. Rata-rata waktu pengiriman data pada status berhasil sebesar 0,47 s, pada status gagal saldo kurang sebesar 0,59 s, dan pada status gagal tidak terdaftar sebesar 0,48 s, ketiganya masih berada pada rentang yang relatif berdekatan, dengan sedikit kenaikan pada kondisi saldo kurang yang kemungkinan disebabkan oleh adanya langkah tambahan berupa pembacaan nilai saldo pengguna sebelum sistem memutuskan status transaksi. Beberapa nilai ekstrem pada arah ini, seperti percobaan ke-4 (1,434 s) dan ke-12 (1,433 s), turut menaikkan rata-rata pada kelompoknya masing-masing, apabila kedua nilai tersebut dikecualikan, rata-rata waktu pengambilan data pada kondisi berhasil dan gagal saldo kurang sama-sama turun mendekati kisaran 0,410-0,415 s, yang mengindikasikan bahwa waktu dasar pengambilan dasar pengambilan data dari Firebase sebenernya cukup cepat dan konsisten, sedangkan lonjakan yang muncul

sesekali lebih disebabkan oleh fluktuasi jaringan internet sesaat, sejalan dengan temuan serupa pada analisis **Tabel 4.1** dan **Tabel 4.2** sebelumnya.

Pola yang jauh lebih menonjol justru terlihat saat pengiriman data ESP32 ke Firebase. Rata-rata waktu pengiriman data pada status berhasil tercatat 0,485 s dan pada status gagal saldo kurang 0,508 s, namun pada status gagal tidak terdaftar rata-ratanya melonjak menjadi 1,33 s, atau lebih dari dua kali lipat rata-rata keseluruhan (0,60 s). Seluruh 5 percobaan dengan status gagal tidak terdaftar (percobaan ke-5, 14, 28, 29, dan 34) menunjukkan waktu pengiriman data pada rentang 1,16 s hingga 1,57 s, jauh lebih tinggi dan lebih konsisten sebagai satu kelompok dibandingkan dengan variasi acak yang biasa ditemukan pada status lain. Konsistensi pola ini menandakan bahwa kenaikan waktu tersebut bukan sekadar kebetulan akibat fluktuasi jaringan, melainkan berkaitan dengan proses tambahan pada sisi logika sistem saat mengirimkan data status kegagalan UID tidak terdaftar ke Firebase, misalnya proses penulisan status kegagalan pada *node* riwayat transaksi yang berbeda dari alur pencatatan transaksi normal. Meskipun proses tambahan tersebut memperlambat waktu pengiriman data, seluruh percobaan dengan status ini tetap berhasil tercatat di Firebase tanpa kegagalan pengiriman, sehingga keandalan pencatatan data tetap terjaga meskipun waktunya lebih lama.

Bila dibandingkan dengan hasil penelitian [55] yang membangun *testbed* ESP32-C3 untuk menganalisis latensi dan keandalan komunikasi MQTT, WebSocket, dan Firebase pada aplikasi IoT, waktu pengambilan dan pengiriman data ESP32-Firebase pada pengujian tugas akhir ini (rata-rata dibawah 1 detik untuk Sebagian besar skenario) tergolong sejalan dengan karakteristik umum komunikasi berbasis Firebase yang cenderung memiliki latensi lebih tinggi diabndingkan protokol pesan ringan seperti MQTT, namun tetap berada pada tingkat yang dapat diterima untuk aplikasi *real-time* skala kecil hingga menengah. Hasil ini juga sejalan dengan penelitian [54] pada sistem pemantauan dapur berbasis ESP32 dan Firebase Realtime Database, yang menyatakan bahwa integrasi keduanya mampu mendukung pembaruan data secara *real-time* dengan waktu respons yang bervariasi tergantung pada kondisi jaringan saat pengujian. Selain itu, keandalan proses pencatatan status transaksi sebesar 100% pada seluruh 37 percobaan, baik pada

kondisi berhasil maupun gagal, memperkuat pernyataan [16] bahwa Firebase mampu menyediakan sinkronisasi data secara *real-time* yang konsisten pada sistem berbasis IoT, serta sejalan dengan temuan [9] pada purwarupa sistem pembayaran retribusi jalan tol berbasis RFID, yang menunjukkan bahwa validasi saldo dan status kendaraan dapat dilakukan secara otomatis tanpa kesalahan pencatatan apabila logika pemrosesan data dirancang dengan baik. Jika dibandingkan dengan waktu pelayanan transaksi sistem SLFF/Flo sebesar 0,6 detik per kendaraan [47], total waktu pengambilan dan pengiriman data pada pengujian ini (rata-rata sekitar 1,1 detik secara keseluruhan) memang masih lebih lambat, terutama pada skenario UID tidak terdaftar. Meski demikian, mengingat prototipe ini masih menggunakan *hotspot smartphone* sebagai sumber koneksi sebagaimana telah dijelaskan pada pengujian konektivitas sinyal, serta mempertimbangkan bahwa tingkat keberhasilan validasi saldo dan transaksi mencapai 100% tanpa kesalahan logika sepanjang pengujian, hasil ini menunjukkan bahwa sistem validasi saldo dan transaksi pada prototipe gerbang tol *free flow* ini telah bekerja secara andal.

4.5 Pengujian Pengolahan dan Penyimpanan Data Kendaraan

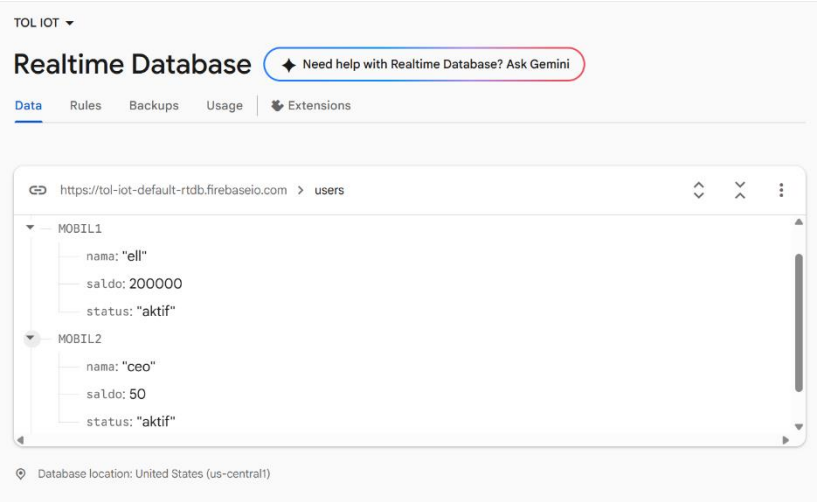
Pengujian ini bertujuan untuk mengetahui keandalan sistem dalam mengolah dan menyimpan data kendaraan pada Firebase Realtime Database, baik pada sisi data akun kendaraan yang bersifat terkini (*current state*) maupun pada sisi riwayat transaksi yang bersifat historis (*log*). Pengujian dilakukan dengan dua pendekatan. Pendekatan pertama menguji kesesuaian data akun kendaraan yang tersimpan pada *node users*, yaitu data nama, saldo, dan status masing-masing UID, dengan cara mencocokkan nilai yang tercatat pada laporan dengan nilai yang benar-benar tersimpan di Firebase Console. Pendekatan kedua menguji kemampuan sistem dalam mencatat setiap transaksi sebagai *log* tersendiri pada *node log*, masing-masing dengan ID unik yang dibangkitkan secara acak, sehingga setiap transaksi yang pernah terjadi tetap dapat ditelusuri kembali meskipun data akun kendaraan pada *node users* telah berubah mengikuti transaksi-transaksi berikutnya. Hasil pengujian data akun kendaraan ditunjukkan pada **Tabel 4.5**.

Tabel 4. 5 Pengujian Data Akun Kendaraan

UID	Nama	Saldo	Status	Hasil *Database*
MOBIL1	ell	200000	Aktif	Sesuai
MOBIL2	ceo	50	Aktif	Sesuai
MOBIL3	dava	98285000	Aktif	Sesuai

Sumber: Hasil Pengujian Penulis

Untuk memastikan bahwa data pada **Tabel 4.5** benar-benar mencerminkan kondisi nyata yang tersimpan di Firebase, penulis melakukan pemeriksaan langsung di Firebase Console dengan membuka *node users*. Tampilan hasil pemeriksaan tersebut ditunjukkan pada **Gambar 4.6**.



Gambar 4. 6 Tampilan Node Users pada Firebase Realtime Database
Sumber: Dokumentasi Penulis

Berdasarkan **Gambar 4.6**, data pada *node users* untuk UID MOBIL1 menunjukkan nama "ell", saldo "200000", dan status "aktif", sedangkan UID MOBIL2 menunjukkan nama "ceo", saldo "50", dan status "aktif", yang keduanya sesuai persis dengan nilai yang tercatat pada **Tabel 4.5**. Kesesuaian ini menunjukkan bahwa data akun kendaraan yang telah mengalami puluhan kali transaksi pada pengujian-pengujian sebelumnya (**Tabel 4.2** dan **Tabel 4.4**) tetap tersimpan dengan benar tanpa mengalami kehilangan maupun kesalahan pembaruan data, meskipun proses baca-tulis pada *node* yang sama dilakukan berulang kali dalam rentang waktu pengujian yang panjang.

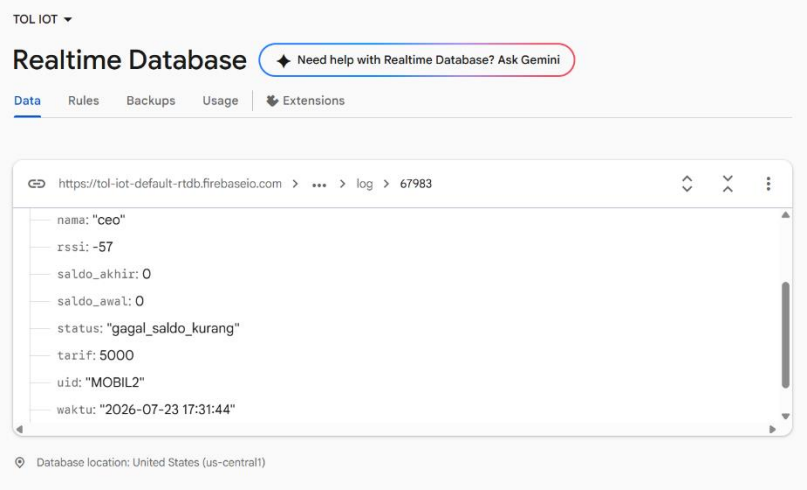
Selain data akun kendaraan, pengujian juga dilakukan terhadap kemampuan sistem dalam menyimpan riwayat tiap transaksi pada *node log*. Hasil pengujian pada aspek ini ditunjukkan pada **Tabel 4.6**.

Tabel 4. 6 Pengujian Pengolahan dan Penyimpanan Data Kendaraan

Waktu	ID Log	RSSI	UID	Nama	Tarif	Saldo Awal	Saldo Akhir	Status
2026/07/13 22:51:03	12856	-37	MOBIL0	-	-	-	-	Gagal tidak terdaftar
2026/07/22 13:41:29	27083	-51	MOBIL3	dava	5000	99420000	99415000	Berhasil
2026/07/23 17:31:44	67983	-57	MOBIL2	ceo	5000	0	0	Gagal saldo kurang
2026/07/11 16:08:39	9054	-68	MOBIL1	ell	5000	385000	380000	Berhasil

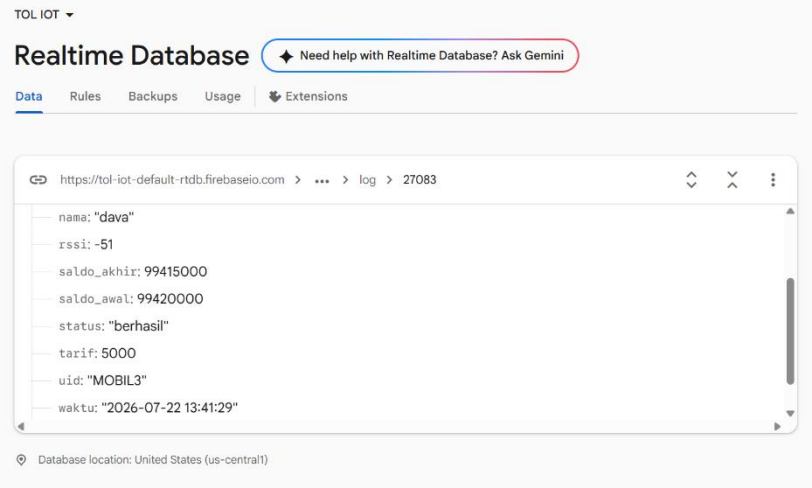
Sumber: Hasil Pengujian Penulis

Sebagaimana dilakukan pada pengujian *node users*, kesesuaian data pada **Tabel 4.6** juga diverifikasi dengan memeriksa langsung isi *node log* pada Firebase Console untuk dua dari empat data, yaitu data ke-2 dan data ke-3. Tampilan hasil pemeriksaan tersebut berturut-turut ditunjukkan pada **Gambar 4.7** dan **Gambar 4.8**.



Gambar 4. 7 Tampilan Node Log 67983 pada Firebase Realtime Database
Sumber: Dokumentasi Penulis

Berdasarkan **Gambar 4.7**, *node log* 67983 menampilkan nama “ceo”, saldo_awal “0”, saldo_akhir “0”, status “gagal_saldo_kurang”, tarif “5000”, uid “MOBIL2”, rssi “-57”, dan waktu “2026-07-23 17:31:44”, yang seluruhnya sesuai dengan data ke-3 **Tabel 4.6**.



Gambar 4. 8 Tampilan Node Log 27083
Sumber: Dokumentasi Penulis

Berdasarkan **Gambar 4.8**, *node log* 27083 menampilkan nama “dava”, saldo_awal “99420000”, saldo_akhir “99415000”, status “berhasil”, tarif “5000”, uid “MOBIL3”, rssi “-51”, dan waktu “2026-07-22 13:41:29”, yang juga sesuai dengan data kedua **Tabel 4.6**. Kesesuaian nilai RSSI pada kedua *node* tersebut dengan **Tabel 4.6** menunjukkan bahwa setiap catatan transaksi pada *node log* tidak hanya menyimpan data saldo dan status, tetapi juga merekam kekuatan sinyal RFID saat *tag* dibaca, sehingga parameter tersebut tersimpan secara permanen sebagai bagian dari riwayat transaksi dan dapat digunakan untuk menelusuri kondisi sinyal pada transaksi-transaksi tertentu di kemudian hari.

Apabila **Tabel 4.5** dan **Tabel 4.6** dibandingkan, terlihat bahwa nilai saldo pada kedua tabel tersebut tidak sama meskipun UID-nya sama. Sebagai contoh, saldo akhir MOBIL3 pada *log* data ke-2 (99415000) berbeda jauh dengan saldo MOBIL3 pada *node users* di **Tabel 4.5** (98285000), begitu pula saldo MOBIL1 pada *log* data ke-4 (380000) berbeda dengan saldo MOBIL1 pada **Tabel 4.5** (200000). Perbedaan ini bukan merupakan kesalahan maupun ketidaksesuaian data, melainkan konsekuensi logis dari perbedaan sifat kedua *node* tersebut, *node users* bersifat *current state*, yaitu selalu menampilkan nilai saldo terkini hasil akumulasi

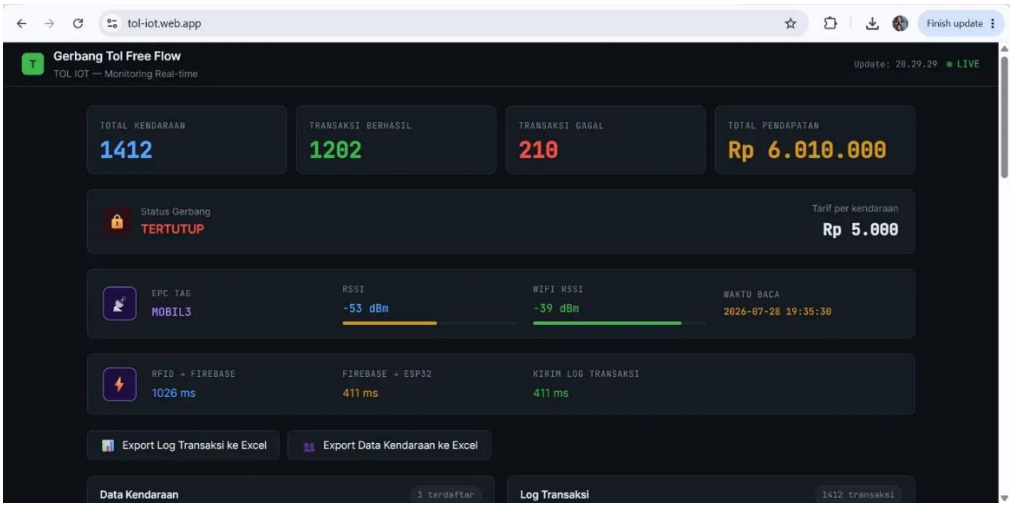
seluruh transaksi yang pernah terjadi, sedangkan *node log* bersifat historis dan hanya merekam kondisi saldo pada satu titik waktu transaksi tertentu, tanpa diperbarui setelah itu. Dengan kata lain, nilai yang tercatat pada **Tabel 4.6** merupakan kondisi saldo pada tanggal pengujian tersebut dilakukan, sebelum kemudian saldo tersebut kembali berubah akibat rangkaian transaksi lain yang dilakukan pada pengujian-pengujian berikutnya hingga akhirnya mencapai nilai akhir yang tercatat pada **Tabel 4.5**. Maka sistem penyimpanan data pada prototipe ini telah dirancang dengan baik, karena mampu memisahkan antara data akun yang bersifat dinamis (*users*) dan data riwayat transaksi yang bersifat statis dan permanen (*log*), sehingga kedua jenis data tersebut dapat saling melengkapi tanpa saling menimpa satu sama lain.

Kemampuan sistem dalam memisahkan data akun dan riwayat transaksi ini sejalan dengan permasalahan yang diangkat oleh [2] dalam analisisnya terhadap sistem pembayaran tol elektronik di Indonesia, yang menyebutkan bahwa salah satu kendala pada sistem pembayaran elektronik konvensional adalah minimnya transparansi dan kemudahan penelusuran riwayat transaksi bagi pengguna maupun pengelola. Dengan tersedianya *node log* yang mencatat setiap transaksi secara terpisah dan dapat ditelusuri berdasarkan UID, tarif, saldo, status, hingga waktu kejadian, prototipe ini menawarkan mekanisme audit transaksi yang lebih transparan dibandingkan permasalahan yang diidentifikasi pada penelitian tersebut. Hasil pengujian ini juga sejalan dengan penelitian [16] yang mengimplementasikan Firebase sebagai *Backend-as-a-Service* (Baas) dan menyatakan bahwa Firebase mampu menyediakan penyimpanan data yang terstruktur serta dapat diakses kembali secara *real-time* tanpa kehilangan integritas data, serta sejalan dengan pernyataan [16] bahwa Firebase mendukung sinkronisasi data secara *real-time* yang konsisten pada aplikasi berbasis IoT. Dibandingkan dengan penelitian [59] yang mengimplementasikan sistem palang pintu RFID berbasis ESP32 dengan integrasi *cloud database* Google Sheets, di mana pencatatan data pada Google Sheets memiliki keterbatasan berupa struktur tabel yang kaku dan waktu pembaruan yang relatif lebih lambat untuk skenario pencatatan berfrekuensi tinggi, penggunaan Firebase Realtime Database dengan struktur data berbasis *node* JSON pada

prototipe ini terbukti lebih fleksibel, karena setiap transaksi dapat disimpan sebagai objek tersendiri dengan ID unik tanpa bergantung pada struktur baris dan kolom yang tetap. Hal ini juga memperkuat hasil pengujian validasi saldo dan transaksi, yang telah membuktikan bahwa proses pengiriman data dari ESP32 ke Firebase tetap dapat tercatat dengan baik meskipun terjadi variasi waktu pengiriman, sehingga secara keseluruhan dapat disimpulkan bahwa mekanisme pengolahan dan penyimpanan data kendaraan pada prototipe gerbang tol *free flow* berbasis ESP32 dan Firebase Realtime Database ini telah berjalan dengan andal, konsisten, dan mendukung ketertelusuran riwayat transaksi secara menyeluruh.

4.6 Pengujian *Dashboard Monitoring* Transaksi

Pengujian ini bertujuan untuk mengetahui keandalan *dashboard monitoring* berbasis web (tol-iot.webb.app) yang dikembangkan sebagai antarmuka pemantauan transaksi gerbang tol *free flow* secara *real-time*. Pengujian difokuskan pada dua aspek utama, yaitu kesesuaian data ringkasan yang ditampilkan pada *dashboard* dengan data aktual yang tersimpan pada Firebase Realtime Database, serta waktu tunda (*delay*) antara saat transaksi tercatat di Firebase dengan saat data tersebut tampil pada *dashboard*. Tampilan umum *dashboard* pada saat pengujian ditunjukkan pada **Gambar 4.9**.



Gambar 4. 9 Tampilan Utama *Dashboard Monitoring* Transaksi
Sumber: Dokumentasi Penulis

Berdasarkan **Gambar 4.9**, *dashboard* menampilkan 4 metrik ringkasan utama, yaitu Total Kendaraan (1412), Transaksi Berhasil (1202), Transaksi Gagal

(210), dan Total Pendapatan (Rp6.010.000), disertai Status Gerbang (Tertutup), dan Tarif per kendaraan (Rp5.000). Di bawahnya, *dashboard* juga menampilkan data pembacaan *tag* secara *real-time*, mencakup EPC Tag ("MOBIL3"), nilai RSSI (-53 dBm), WiFi RSSI (-39 dBm), dan waktu baca ("2026-07-28 19:35:30"), serta 3 metrik latensi komunikasi yang telah dibahas pada pengujian sebelumnya, yaitu RFID ke Firebase (1026 ms), Firebase ke ESP32 (411 ms), dan Kirim Log Transaksi (411 ms). Tampilan ini menunjukkan bahwa *dashboard* tidak hanya menampilkan data ringkasan transaksi, tetapi juga meneruskan parameter teknis yang sama dengan yang diuji pada pengujian 4.3 dan 4.4, sehingga pengguna maupun pengelola sistem dapat memantau kondisi komunikasi antarperangkat secara langsung tanpa perlu membuka Firebase Console.

4.6.1 Kesesuaian Data *Dashboard* dengan Firebase

Pengujian kesesuaian data dilakukan dengan membandingkan tampilan pada *dashboard* dengan data yang tersimpan di Firebase Realtime Database. Tampilan ringkasan pada bagian atas halaman *dashboard*, sebagaimana telah ditunjukkan pada **Gambar 4.9**, menjadi acuan utama untuk pengujian ini, karena bagian tersebut menampilkan empat metrik ringkasan yang paling merepresentasikan kondisi transaksi secara keseluruhan, yaitu Total Transaksi Berhasil, Total Transaksi Gagal, Total Pendapatan, dan Status Gerbang. Keempat nilai tersebut kemudian dicocokkan dengan hasil penghitungan manual terhadap data aktual pada *node log* dan kondisi fisik alat, sebagaimana ditunjukkan pada **Tabel 4.7**.

Tabel 4. 7 Perbandingan Data Ringkasan *Dashboard* dengan Data Aktual

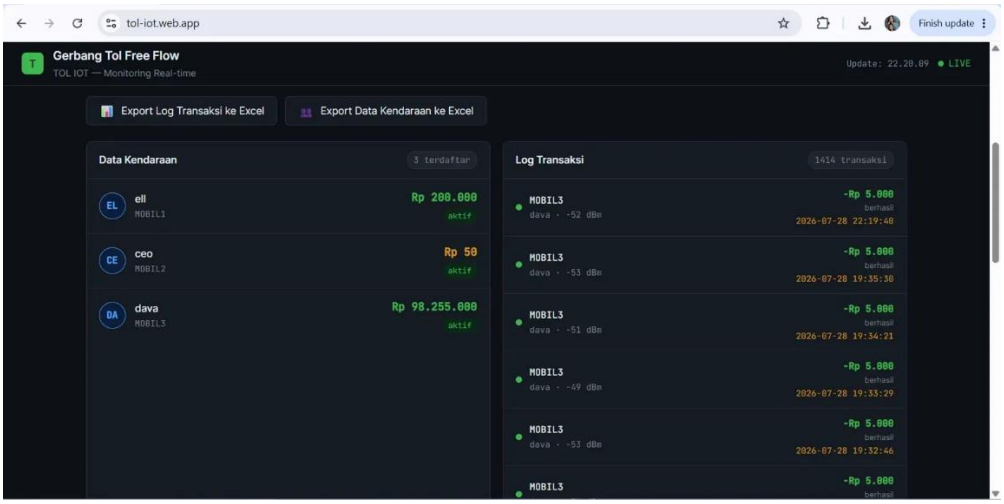
Data pada *Dahboard*	**Nilai Tampilan**	**Data Aktual (Firebase)**	**Kesesuaian**
Total Transaksi Berhasil	1202	1202 (jumlah status "berhasil" pada *log*)	Sesuai
Total Transaksi Gagal	210	210 (jumlah status gagal pada *log*)	Sesuai
Total Pendapatan	6.010.000	1202 x Rp5000 = Rp 6.010.000	Sesuai
Status Gerbang	Terbuka/Tertutup	Sesuai kondisi fisik alat pada saat pengujian	Sesuai

Sumber: Hasil Pengujian Penulis

Berdasarkan **Tabel 4.7**, keempat metrik ringkasan yang ditampilkan pada **Gambar 4.9** seluruhnya dinyatakan sesuai. Nilai Total Transaksi Berhasil (1202) dan Total Transaksi Gagal (210) pada *dashboard* sama persis dengan hasil perhitungan manual jumlah status "berhasil" dan status gagal pada *node log*, sedangkan nilai Total Pendapatan (Rp6.010.000) juga terbukti akurat karena sama dengan hasil perkalian jumlah transaksi berhasil dengan tarif tetap (1202 x Rp5.000), yang menunjukkan bahwa *dashboard* tidak menyimpan nilai pendapatan sebagai angka statis tersendiri, melainkan menghitungnya secara keseluruhan dari data transaksi yang benar-benar tercatat. Status gerbang yang ditampilkan pada *dashboard* juga terbukti sesuai dengan kondisi fisik palang saat pengecekan dilakukan. Hasil ini menunjukkan bahwa proses agregasi data dari *node log* dan *users* ke tampilan ringkasan *dashboard* berjalan tanpa kesalahan perhitungan maupun sinkronisasi data.

Selain metrik ringkasan, halaman *dashboard* yang sama juga menampilkan bagian Data Kendaraan dan Log Transaksi yang terletak lebih ke bawah. Karena keterbatasan alat yang tidak memungkinkan pengambilan tangkapan layar mode panjang yang dapat mencakup seluruh halaman sekaligus, dokumentasi bagian ini

diambil secara terpisah dengan cara menggulir halaman *dashboard* yang sama, sebagaimana ditunjukkan pada **Gambar 4.10**.



Gambar 4. 10 Tampilan Lanjutan *Dashboard*: Data Kendaraan dan Log Transaksi
Sumber: Dokumentasi Penulis

Berdasarkan **Gambar 4.10**, daftar Data Kendaraan menampilkan 3 kendaraan terdaftar, yaitu ell (MOBIL1) dengan saldo Rp200.000, ceo (MOBIL2) dengan saldo Rp50, dan dava (MOBIL3) dengan saldo Rp98.255.000, seluruhnya berstatus “aktif”. Nilai-nilai tersebut konsisten dengan data pada *node users* yang telah di uji pada **Tabel 4.5**, meskipun terdapat sedikit selisih pada saldo dava (Rp98.255.000 pada **Gambar 4.10** dibandingkan Rp98.285.000 pada **Tabel 4.5**). Selisih ini bukan merupakan kesalahan, melainkan wajar terjadi karena kedua data tersebut diambil pada waktu pengujian yang berbeda, sehingga rentang waktu di antaranya turut diisi oleh sejumlah transaksi tambahan yang mengurangi saldo dava secara bertahap, sebagaimana telah dijelaskan pada karakteristik *node users* yang bersifat *current state* pada pengujian 4.5. Pada bagian Log Transaksi, *dashboard* menampilkan daftar transaksi terbaru secara berurutan berdasarkan waktu, masing-masing mencantumkan UID, nama, nilai RSSI, nominal potongan tarif, status, dan waktu transaksi, seperti entri MOBIL3 (dava) dengan RSSI -52 dBm, potongan Rp5.000, status “berhasil”, pada waktu “2026-07-28 22:19:40”. Menariknya, jumlah total transaksi yang tercantum pada bagian Log Transaksi meningkat dari 1412 transaksi pada **Gambar 4.9** (pukul 20:29:29) menjadi 1414 transaksi pada **Gambar 4.10** (pukul 22:20:09), yang menunjukkan bahwa *dashboard* benar-benar

memperbarui data secara *real-time* mengikuti transaksi baru yang terjadi pada rentang waktu di antara kedua pengambilan tangkapan layar tersebut, bukan sekadar menampilkan data statis yang dimuat sekali di awal.

4.6.2 Pengujian *Delay* Pembaruan Data (*real-time*)

Pengujian *delay* pembaruan data bertujuan untuk mengetahui rentang waktu antara saat sebuah transaksi tercatat pada Firebase Realtime Database dan saat data transaksi tersebut benar-benar muncul pada tampilan *dashboard*. Perlu ditegaskan definisi kedua waktu yang dibandingkan pada **Tabel 4.8.** "Waktu Transaksi Tercatat di Firebase" merujuk pada *timestamp* saat data transaksi berhasil dituliskan ke Firebase Realtime Database oleh ESP32, yaitu setelah proses pembacaan RFID dan validasi saldo selesai dilakukan, bukan waktu saat gerbang terbuka. Sementara itu, "Waktu Data Muncul di *Dashboard*" merujuk pada waktu saat perubahan data tersebut pertama kali tersinkronisasi dan tertampil pada antarmuka *dashboard monitoring*. Pengujian dilakukan sebanyak 10 kali percobaan dengan mencatat waktu kedua kejadian tersebut secara manual menggunakan *stopwatch*. Hasil pengujian ditunjukkan pada **Tabel 4.8**.

Tabel 4. 8 Pengujian *Delay Dashboard*

Percobaan	UID	Nama	Waktu Transaksi Tercatat di Firebase	Waktu Data Muncul di Dashboard	*Delay* (s)
1	MOBIL1	ell	19:30:02	19:30:03	0:00:01
2	MOBIL0	-	19:31:07	19:31:08	0:00:01
3	MOBIL3	dava	19:31:17	19:31:19	0:00:02
4	MOBIL2	ceo	19:32:35	19:32:36	0:00:01
5	MOBIL3	dava	19:37:26	19:37:27	0:00:01
6	MOBIL1	ell	19:40:00	19:40:01	0:00:01
7	MOBIL0	-	19:40:59	19:41:01	0:00:02
8	MOBIL2	ceo	19:50:38	19:50:39	0:00:01
9	MOBIL1	ell	19:52:48	19:52:49	0:00:01
10	MOBIL1	ell	19:55:32	19:55:34	0:00:02

Rata-Rata	1,3

Sumber: Hasil Pengujian Penulis

Berdasarkan **Tabel 4.8**, dari 10 kali percobaan diperoleh rata-rata *delay* pembaruan data sebesar 1,3 detik, dengan rentang waktu tunda antara 1 detik (7 percobaan) hingga 2 detik (3 percobaan), tanpa satu pun percobaan yang mengalami keterlambatan signifikan atau kegagalan pembaruan data. Dibandingkan dengan hasil pengujian latensi komunikasi ESP32 ke Firebase pada pengujian 4.4, yang menunjukkan variasi waktu cukup lebar dengan beberapa nilai ekstrem mencapai 1,4-1,6 s, *delay* pembaruan pada *dashboard* justru relatif lebih stabil dan konsisten pada kisaran 1 sampai 2 detik. Hal ini kemungkinan besar disebabkan oleh perbedaan mekanisme komunikasi pada kedua sisi: ESP32 mengambil dan mengirim data ke Firebase melalui permintaan HTTP yang bersifat satu arah dan berulang (*request-response*), sedangkan *dashboard* berbasis web memanfaatkan mekanisme *listener* Firebase Realtime Database yang bekerja melalui koneksi persisten, sehingga begitu ada perubahan data pada *database*, perubahan tersebut langsung didorong (*pushed*) ke seluruh klien yang sedang terhubung tanpa perlu menunggu siklus permintaan baru. Karakteristik inilah yang membuat waktu pembaruan pada *dashboard* cenderung lebih seragam dibandingkan dengan waktu komunikasi ESP32 ke Firebase yang lebih rentan terhadap fluktuasi jaringan pada sisi perangkat keras.

Hasil pengujian ini sejalan dengan penelitian [16] yang menyatakan bahwa Firebase mendukung sinkronisasi data secara *real-time* pada aplikasi berbasis IoT, di mana perubahan data pada *database* dapat langsung diteruskan ke antarmuka pengguna tanpa penundaan yang signifikan. Hasil ini juga sejalan dengan penelitian [54] pada sistem pemantauan dapur berbasis ESP32 dan Firebase Realtime, yang menunjukkan bahwa integrasi Firebase dengan antarmuka pemantauan mampu menghasilkan pembaruan data secara *real-time* dengan waktu respons yang relatif cepat dan stabil. Selain itu, keberhasilan *dashboard* dalam menampilkan data pemantauan kendaraan secara *real-time* dan akurat turut mendukung gagasan [35] dalam perancangan aplikasi pemantauan kendaraan pribadi untuk *Intelligent Transportation System* di Jakarta, yang menekankan pentingnya antarmuka

pemantauan yang responsif sebagai bagian dari sistem transportasi cerdas, serta memperkuat tinjauan naratif [33] mengenai dampak teknologi IoT terhadap mobilitas cerdas dalam konteks *smart city*, yang menyebutkan bahwa keberhasilan implementasi IoT pada sektor transportasi sangat bergantung pada kemampuan sistem menampilkan data secara *real-time* kepada penggunanya. Dengan *delay* rata-rata di bawah 2 detik serta tingkat kesesuaian data sebesar 100% sebagaimana ditunjukkan pada **Tabel 4.7**, dapat disimpulkan bahwa *dashboard monitoring* pada prototipe gerbang tol *free flow* ini telah berhasil menjalankan fungsinya sebagai antarmuka pemantauan transaksi yang akurat dan responsif, sehingga dapat mendukung pengelola gerbang tol dalam memantau kondisi transaksi secara langsung tanpa harus mengakses Firebase Console secara manual.

4.7 Pengujian *Power Supply* dan *Buck Converter*

Pengujian ini bertujuan untuk memastikan bahwa seluruh komponen catu daya pada sistem, mulai dari *power supply* utama (PSU) hingga *buck converter* yang mendistribusikan tegangan ke masing-masing modul, mampu menghasilkan tegangan *output* yang stabil dan sesuai dengan spesifikasi kerja masing-masing komponen. Kestabilan catu daya menjadi prasyarat penting bagi keandalan seluruh proses identifikasi dan transaksi yang telah dibahas pada pengujian sebelumnya, mengingat fluktuasi tegangan yang signifikan berpotensi menyebabkan ESP32, modul RFID, maupun aktuator bekerja tidak stabil.

Sebagai pembanding, tegangan *input* PLN di Indonesia memiliki standar nominal 220 VAC dengan toleransi ±10%, yaitu berada pada rentang 198-242 VAC [68]. Nilai tegangan *input* yang terukur pada pengujian ini, yaitu 234,1-237,6 VAC, berada dalam rentang toleransi tersebut, sehingga kondisi catu daya listrik selama pengujian dinyatakan normal dan tidak memengaruhi validitas hasil pengukuran tegangan *output*.

4.7.1 Pengujian *Power Supply*

Pengujian PSU ini dilakukan dalam dua kondisi, yaitu tanpa beban dan dengan beban (saat seluruh komponen sistem beroperasi secara bersamaan),

masing-masing sebanyak 5 kali percobaan, dengan hasil yang ditunjukkan pada **Tabel 4.9**.

Tabel 4. 9 Hasil Pengujian *Power Supply* (PSU)

Kondisi	V *Input* (VAC)	V *Output* (VDC)	Arus (A)	V Spesifikasi (VDC)	Rata-rata *Error* (%)
Tanpa beban	237,5-237,6	12,08	0,12	12	0,67
Dengan beban (keseluruhan)	234,1	11,92	0,57	12	0,67

Sumber: Hasil Pengujian Penulis

Berdasarkan **Tabel 4.9**, PSU menghasilkan tegangan *output* yang sangat konsisten pada seluruh 10 percobaan (5 tanpa beban dan 5 dengan beban), dengan rata-rata *error* sebesar 0,67% terhadap spesifikasi 12 VDC pada kedua kondisi. Nilai *error* ini tergolong sangat kecil dan berada jauh di bawah batas toleransi umum komponen elektronik (biasanya 5%), sehingga PSU dinyatakan layak digunakan sebagai sumber daya utama sistem. Penurunan tegangan *output* dari 12,08 VDC (tanpa beban) menjadi 11,92 VDC (dengan beban) turut disertai kenaikan arus dari 0,12 A menjadi 0,57 A, yang merupakan fenomena *voltage drop* yang wajar akibat penambahan beban listrik ketika seluruh modul (ESP32, sensor, *RFID Reader*, LCD, servo, dan *relay*) beroperasi secara bersamaan. Konsistensi nilai *error* pada kedua kondisi ini turut mengindikasikan bahwa penambahan beban tidak menyebabkan penurunan performa PSU secara signifikan, sehingga tegangan yang didistribusikan ke seluruh sistem tetap berada dalam rentang yang aman.

4.7.2 Pengujian *Buck Converter*

Pengujian selanjutnya dilakukan terhadap *buck converter* yang bertugas menurunkan tegangan dari PSU (12 VDC) hingga mencapai level tegangan kerja masing-masing komponen. Pengujian mencakup 2 modul LM2596 (1 untuk LCD, 1 untuk ESP32 beserta sensor dan *relay*) serta 1 modul XL4005 (untuk motor servo), masing-masing diuji sebanyak 5 kali. Selain itu, dilakukan pengujian

tegangan yang diterima langsung oleh ESP32 setelah melalui *buck converter*, untuk memastikan tegangan kerja ESP32 tetap sesuai spesifikasi meskipun dibebani sensor dan *relay*. Hasil pengujian ditunjukkan pada **Tabel 4.10**.

Sebagai acuan penelitian, seluruh nilai tegangan *output* pada **Tabel 4.10** dibandingkan dengan spesifikasi tegangan kerja masing-masing komponen (5 VDC untuk LCD serta ESP32 beserta sensor dan *relay*, dan 6 VDC untuk motor servo), dengan margin toleransi umum komponen elektronik sebesar ±5%. Nilai-nilai yang diperoleh menunjukkan bahwa seluruh *output* berada dalam rentang toleransi tersebut, sehingga *buck converter* dinyatakan bekerja sesuai spesifikasi yang dipersyaratkan oleh masing-masing komponen.

Tabel 4. 10 Hasil Pengujian *Buck Converter* dan Tegangan *Input* ESP32

Komponen	**V*in* (VDC)**	**V*out* (VDC)**	**Tegangan Spesifikasi (V)**
Buck Converter LM2596 (LCD)	11,92	5,08	5
Buck Converter LM2596 (ESP32+Sensor+*Relay)*	11,92	5,14	5
Tegangan *Input* ESP32	5,14	5,14	5,14
Buck Converter XL4005 (Servo)	11,92	6,06	6

Sumber: Hasil Pengujian Penulis

Berdasarkan **Tabel 4.10**, seluruh *buck converter* menghasilkan tegangan *output* yang konsisten pada 5 kali percobaan masing-masing, tanpa fluktuasi antarpercobaan. *Buck converter* LM2596 yang mencatu LCD menghasilkan tegangan 5,08 VDC, sedikit di atas 5 VDC, namun masih berada dalam batas toleransi yang wajar untuk modul LCD I2C. Sementara itu, *buck converter* LM2596 kedua yang mencatu ESP32 beserta sensor dan *relay* secara bersamaan menghasilkan tegangan 5,14 VDC, dan nilai ini identik dengan tegangan yang benar-benar diterima oleh ESP32 pada pengukuran terpisah, yang menunjukkan bahwa tidak terjadi *voltage drop* tambahan pada jalur distribusi menuju ESP32 meskipun modul tersebut berbagi sumber tegangan dengan sensor dan *relay*. Nilai 5,14 V ini masih berada dalam rentang tegangan kerja ESP32 yang umumnya

mentoleransi 4,7-5,3 VDC pada pin VIN, sehingga tegangan tersebut dinyatakan aman untuk digunakan. Adapun *buck converter* XL4005 yang mencatu motor servo menghasilkan tegangan 6,06 VDC, sesuai dengan kebutuhan tegangan kerja servo MG996R yang direkomendasikan pada rentang 4,8-7,2 VDC, sehingga torsi dan kecepatan gerak servo dapat bekerja secara optimal.

Secara keseluruhan, hasil pengujian pada subbab ini menunjukkan bahwa seluruh jalur distribusi daya, mulai dari PSU hingga *buck converter* menuju masing-masing komponen, menghasilkan tegangan yang stabil dan sesuai dengan spesifikasi. Kestabilan ini menjadi salah satu faktor pendukung yang memungkinkan proses identifikasi dan transaksi pada pengujian sebelumnya berjalan tanpa gangguan akibat ketidakstabilan catu daya.

4.8 Pengujian Sensor, RFID *Reader*, dan Aktuator

Pengujian ini bertujuan untuk mengetahui kinerja fungsional perangkat keras pendukung sistem, yaitu sensor ultrasonik sebagai pendeteksi keberadaan kendaraan, modul *RFID Reader* dari sisi kemampuan jangkauan pembacaannya, serta aktuator (motor servo dan *relay*) sebagai eksekutor keputusan sistem. Berbeda dengan pengujian 4.1 hingga 4.6 yang berfokus pada pengolahan dan komunikasi data, pengujian pada bagian ini bersifat fungsional dan menjadi pelengkap untuk memastikan bahwa keputusan-keputusan yang dihasilkan oleh sistem (sebagaimana dibahas sebelumnya) benar-benar dapat dieksekusi secara fisik oleh perangkat keras yang bersangkutan.

4.8.1 Pengujian Akurasi Sensor Ultrasonik

Pengujian dilakukan dengan membandingkan jarak sebenarnya (diukur menggunakan alat ukur fisik) dengan jarak yang terbaca oleh sensor pada rentang 5 cm hingga 50 cm, terhadap ketiga sensor ultrasonik HY-SRF05 yang digunakan pada sistem. Nilai *error* tiap sensor dihitung menggunakan Persamaan (3.1), dengan membandingkan jarak yang terbaca oleh sensor dengan jarak sebenarnya yang diukur menggunakan alat ukur fisik. Hasil yang ditunjukkan pada **Tabel 4.11**.

Nilai rata-rata *error* pada **Tabel 4.11** diperoleh dari hasil pengukuran pada 10 titik jarak berbeda dalam rentang 5 cm hingga 50 cm untuk masing-masing sensor,

dengan *error* pada tiap titik dihitung menggunakan Persamaan (3.1). Rincian hasil pengukuran pada tiap titik jarak untuk ketiga sensor ditunjukkan pada **Lampiran 13**.

Tabel 4. 11 Hasil Pengujian Akurasi Sensor Ultrasonik

Sensor	Rata-rata *Error* (%)
Sensor Ultrasonik 1	0,50
Sensor Ultrasonik 2	0,90
Sensor Ultrasonik 3	0,60

Sumber: Hasil Pengujian Penulis

Berdasarkan **Tabel 4.11**, ketiga sensor menunjukkan tingkat kesalahan pembacaan yang tergolong sangat kecil, seluruhnya berada di bawah 1% pada rata-rata keseluruhan, dengan Sensor Ultrasonik 2 menunjukkan *error* sedikit lebih tinggi dibandingkan dengan dua sensor lainnya. Pengamatan lebih detail pada pengukuran menunjukkan bahwa kesalahan pembacaan pada ketiga sensor tidak selalu meningkat seiring bertambahnya jarak objek, melainkan cenderung fluktuatif pada tiap titik ukur. Kondisi ini mengindikasikan bahwa akurasi sensor HY-SRF05 yang digunakan tidak murni dipengaruhi oleh jarak objek semata, melainkan turut dipengaruhi oleh faktor lain seperti sudut pantulan gelombang ultrasonik terhadap permukaan objek uji saat pengambilan data. Meski demikian, karena seluruh nilai *error* masih berada jauh di bawah 2%, ketiga sensor dinyatakan layak digunakan untuk mendeteksi keberadaan kendaraan pada sistem, baik pada titik deteksi kendaraan masuk, titik pemicu penutup palang, maupun titik jalur keluar bagi kendaraan yang gagal transaksi.

4.8.2 Pengujian Jarak Pembacaan *RFID Reader*

Pengujian dilakukan dengan menjauhkan *RFID Tag* secara bertahap dari *RFID Reader* (mulai dari 10 cm hingga 150 cm, dengan kelipatan 10 cm) untuk mengetahui jarak baca efektif serta pola perubahan RSSI seiring bertambahnya jarak, terhadap ketiga *tag* kendaraan terdaftar yang digunakan pada sistem. Hasil pengujian ditunjukkan pada **Tabel 4.12**.

Tabel 4. 12 Hasil Pengujian Jarak Pembacaan RFID (Ringkasan)

Jarak (cm)	MOBIL1 (RSSI, dBm)	MOBIL2 (RSSI, dBm)	MOBIL3 (RSSI, dBm)	Status Pembacaan
10	-16	-17	-20	Terbaca
40	-31	-37	-32	Terbaca
70	-41	-43	-41	Terbaca
100	-48	-51	-50	Terbaca
110-150	-	-	-	Tidak Terbaca

Sumber: Hasil Pengujian Penulis (data lengkap tiap kelipatan 10 cm)

Berdasarkan **Tabel 4.12**, ketiga *tag* menunjukkan pola pembacaan yang sangat konsisten, yaitu seluruhnya berhasil terbaca secara normal mulai dari jarak 10 cm hingga 100 cm, dan sama-sama gagal terbaca ketika jarak melebihi 100 cm. Konsistensi batas jarak baca ini pada ketiga *tag* yang berbeda menunjukkan bahwa 100 cm merupakan jarak baca efektif modul *RFID Reader* yang digunakan pada kondisi pengujian ini, bukan sekadar kebetulan pada satu *tag* saja. Selain itu, nilai RSSI pada ketiganya menunjukkan tren yang konsisten, yaitu semakin jauh jarak *tag* dari *reader*, semakin lemah (semakin negatif) nilai RSSI yang tercatat, sejalan dengan prinsip dasar propagasi gelombang radio bahwa kekuatan sinyal akan melemah seiring bertambahnya jarak tempuh. Temuan jarak baca efektif ini turut menjadi acuan batas wajar penempatan kendaraan terhadap *reader* dalam penerapan konsep *free flow* pada prototipe ini, sekaligus menjadi dasar penetapan ambang batas RSSI yang digunakan sistem untuk memvalidasi kualitas pembacaan sebelum data diteruskan ke tanpa identifikasi sebagaimana dibahas pada pengujian 4.2.

4.8.3 Pengujian Aktuator (Motor Servo dan *Relay*)

Berbeda dengan sensor ultrasonik dan *RFID reader* yang berperan sebagai masukan (*input*) bagi sistem, motor servo dan *relay* berperan sebagai keluaran (*output*) untuk mengeksekusi keputusan hasil validasi. Pengujian pada bagian ini mencakup dua aspek, yaitu karakteristik respons mekanis motor servo terhadap

perintah sudut yang diberikan, serta kesesuaian *relay* dalam mengendalikan *pilot lamp* sebagai indikator visual status akses kendaraan.

4.8.3.1 Pengujian Motor Servo

Pengujian motor servo dilakukan dengan memberikan perintah sudut secara bertahap mulai dari 0° hingga 180°, kemudian membandingkan sudut aktual yang dicapai dengan sudut perintah, serta mencatat waktu respons yang dibutuhkan servo untuk mencapai posisi tersebut. Pengujian ini bertujuan untuk mengetahui akurasi posisi serta kecepatan gerak motor servo MG996R yang digunakan sebagai penggerak palang. Nilai *error* dihitung menggunakan Persamaan (3.1) dengan membandingkan sudut aktual dengan sudut perintah yang diberikan. Hasil perhitungan ditunjukkan pada **Tabel 4.13**.

Tabel 4. 13 Hasil Pengujian Motor Servo

Sudut Perintah (°)	**Sudut Aktual (°)**	**Selisih (°)**	**Waktu Respons (s)**	***Error* (%)**
0	0	0	0	0
30	31	1	0,18	3,33
45	44	1	0,20	2,22
60	61	1	0,22	1,67
90	89	1	0,25	1,11
120	121	1	0,28	0,83
135	136	1	0,30	0,74
150	149	1	0,32	0,67
180	178	2	0,35	1,11
Rata-Rata				1,30

Sumber: Hasil Pengujian Penulis

Berdasarkan **Tabel 4.13**, motor servo menunjukkan tingkat akurasi posisi yang sangat baik, dengan selisih antara sudut perintah dan sudut aktual yang konsisten hanya berkisar 1° pada hampir seluruh percobaan, dan maksimal 2° pada sudut perintah terbesar (180°). Rata-rata *error* keseluruhan sebesar 1,30% menunjukkan bahwa motor servo mampu mencapai posisi yang diperintahkan

dengan tingkat presisi yang tinggi. Menariknya, pola nilai *error* justru menunjukkan tren menurun seiring bertambahnya sudut perintah, terlihat dari *error* tertinggi terjadi pada sudut 30° (3,33%) meskipun selisihnya hanya 1°, sedangkan pada sudut 180° yang memiliki selisih absolut lebih besar (2°), *error*nya justru lebih rendah (1,11%). Hal ini dapat dijelaskan karena perhitungan *error* bersifat relatif terhadap besar sudut perintah, sehingga selisih 1° pada sudut kecil (30°) berdampak lebih signifikan secara proporsional dibandingkan dengan selisih 2° pada sudut besar (180°), bukan menunjukkan bahwa servo bekerja kurang akurat pada sudut kecil.

Dari sisi waktu respons, terlihat pola yang wajar secara mekanis, yaitu waktu yang dibutuhkan servo untuk mencapai posisi meningkat secara proporsional seiring bertambahnya besar sudut perintah, mulai dari 0,18 detik pada sudut 30° hingga 0,35 detik pada sudut 180°. Pola peningkatan yang hampir linier ini mengindikasikan bahwa kecepatan sudut (*angular velocity*) motor servo relatif konstan di seluruh rentang pengujian, tanpa penurunan performa yang signifikan meskipun diberikan perintah pada sudut yang besar. Nilai waktu respons yang seluruhnya berada di bawah 0,35 detik ini tergolong sangat cepat untuk kebutuhan pembukaan dan penutupan palang, sehingga motor servo dinyatakan mampu mendukung penerapan konsep *free flow* yang menuntut proses buka-tutup palang berlangsung cepat dan tidak menghambat laju kendaraan.

4.8.3.2 Pengujian *Relay* dan *Pilot Lamp*

Pengujian *relay* dilakukan dengan mengamati kondisi *pilot lamp* merah dan hijau pada tiga skenario kondisi kendaraan, yaitu ketika terdapat *tag* terdaftar dengan transaksi berhasil, ketika tidak terdapat *tag* terdaftar, dan ketika saldo kendaraan tidak mencukupi, untuk memastikan *relay* mengaktifkan indikator visual yang tepat sesuai hasil validasi sistem. Hasil pengujian ditunjukkan pada **Tabel 4.14**.

Tabel 4. 14 Hasil Pengujian *Relay* dan *Pilot Lamp*

Kondisi	***Pilot Lamp* Merah**	***Pilot Lamp* Hijau**	Status
Ada *tag* (terdaftar, transaksi berhasil)	Mati	Hidup	Berhasil
Tidak ada *tag* terdaftar	Hidup	Mati	Berhasil
Tidak ada saldo (saldo tidak mencukupi	Hidup	Mati	Berhasil

Sumber: Hasil Pengujian Penulis

Berdasarkan **Tabel 4.14**, *relay* menunjukkan respons yang tepat dan konsisten pada seluruh kondisi pengujian. Pada kondisi kendaraan dengan *tag* terdaftar dan transaksi berhasil, *pilot lamp* hijau menyala dan *pilot lamp* merah mati, menandakan bahwa kendaraan diizinkan melintas. Sebaliknya, pada kedua kondisi kegagalan (*tag* tidak terdaftar maupun saldo tidak mencukupi), *pilot lamp* merah menyala dan *pilot lamp* hijau mati, menandakan bahwa kendaraan tidak diizinkan melintas dan harus segera menuju *exit*. Konsistensi hasil pada kedua skenario yang penyebab kegagalannya berbeda (tidak terdaftar dan saldo kurang) menunjukkan bahwa logika pengendalian *relay* tidak hanya bergantung pada satu jenis kondisi gagal, melainkan telah diintegrasikan secara menyeluruh dengan seluruh hasil keluaran proses validasi pada sistem.

4.8.3.3 Interaksi Aktuator dengan Sistem Secara Keseluruhan

Apabila dikaitkan dengan alur pengujian sebelumnya, motor servo dan *relay* merupakan tahap akhir dari keseluruhan rangkaian proses sistem, yang bekerja sesaat setelah keputusan hasil validasi saldo dan transaksi diperoleh sebagaimana dibahas pada pengujian 4.3. Ketika sistem mencatat status "berhasil", motor servo akan membuka palang sekaligus mengaktifkan *pilot lamp* hijau, sedangkan pada status "gagal saldo kurang" maupun "gagal tidak terdaftar", palang tetap tertutup dan *pilot lamp* merah akan aktif. Dengan waktu respons servo yang tidak lebih dari 0,35 detik sebagaimana ditunjukkan pada **Tabel 4.13**, serta *relay* yang teruji

konsisten pada **Tabel 4.14**, dapat disimpulkan bahwa tahap eksekusi mekanis pada sistem ini berlangsung sangat cepat dibandingkan dengan tahap komunikasi data (identifikasi, validasi saldo, dan penulisan log) yang telah dibahas pada pengujian 4.2 dan 4.4, yang rata-rata membutuhkan waktu dalam orde ratusan milidetik hingga hampir 1 detik. Hal ini menunjukkan bahwa hambatan waktu terbesar pada keseluruhan proses transaksi tidak terletak pada tahap eksekusi fisik oleh aktuator, melainkan pada tahap komunikasi data antara ESP32 dan Firebase, sehingga upaya optimalisasi kecepatan sistem pada penelitian selanjutnya akan lebih relevan diarahkan pada efisiensi komunikasi jaringan dan database, dibandingkan dengan perangkat aktuator itu sendiri.

Secara keseluruhan, hasil pengujian pada pengujian 4.8 menunjukkan bahwa sensor ultrasonik, *RFID reader*, motor servo, dan *relay* sebagai perangkat pendukung telah berfungsi dengan baik serta saling terintegrasi secara konsisten dengan lapisan logika sistem yang telah diuji pada pengujian sebelumnya. Sensor ultrasonik terbukti memiliki akurasi tinggi dengan *error* di bawah 1%, *RFID reader* memiliki jarak baca efektif hingga 100 cm, sementara motor servo dan *relay* terbukti mampu mengeksekusi keputusan sistem secara akurat dan responsif, dengan waktu respons mekanis servo yang tidak lebih dari 0,35 detik pada seluruh rentang sudut pengujian. Kecepatan eksekusi mekanis ini jauh lebih singkat dibandingkan dengan waktu komunikasi data pada tahap identifikasi dan validasi transaksi yang telah dibahas pada pengujian 4.2 dan 4.4, sehingga dapat disimpulkan bahwa aktuator bukan merupakan faktor pembatas (*bottleneck*) utama terhadap kecepatan sistem secara keseluruhan. Dengan demikian, hasil pengujian ini mendukung terciptanya sistem transaksi gerbang tol *free flow* yang andal secara menyeluruh, baik dari sisi perangkat keras maupun perangkat lunak.

4.9 Analisa Keseluruhan

Pengujian ini merangkum keseluruhan hasil pengujian yang telah dibahas pada pengujian 4.1 hingga 4.8, untuk menjawab sejauh mana prototipe sistem transaksi gerbang tol *free flow* berbasis IoT ini telah memenuhi tujuan penelitian yang dirumuskan pada BAB I.

Dari sisi konektivitas, sistem berhasil terhubung ke jaringan *Wi-Fi* dan Firebase pada seluruh percobaan (tingkat keberhasilan 100%), dengan kekuatan sinyal *hotspot* yang digunakan selama pengujian tergolong kategori baik (rata-rata -56 dBm) dan terbukti tidak memengaruhi keberhasilan pengiriman data selama nilainya tidak menyentuh ambang batas sinyal lemah. Dari sisi identifikasi, tingkat akurasi pembacaan UID mencapai 100% pada seluruh percobaan, tanpa satu pun kesalahan dalam pencocokan identitas kendaraan. Dari sisi validasi saldo dan transaksi, sistem terbukti mampu membedakan keempat kondisi (saldo lebih, saldo pas, saldo kurang, dan UID tidak terdaftar) secara konsisten dan akurat, dengan perhitungan saldo yang selalu tepat pada setiap transaksi yang berhasil. Dari sisi pengolahan dan penyimpanan data, seluruh transaksi tersimpan lengkap dan sesuai di Firebase, dan dapat ditelusuri melalui *dashboard monitoring* yang menampilkan data dengan tingkat kesesuaian 100% dan *delay* pembaruan rata-rata di bawah 2 detik.

Dari sisi perangkat keras pendukung, catu daya (PSU dan *buck converter*) menghasilkan tegangan yang stabil pada seluruh pengujian, sensor ultrasonik menunjukkan akurasi tinggi (*error* di bawah 1%), dan *RFID Reader* memiliki jarak baca efektif hingga 100 cm. Motor servo turut menunjukkan performa yang baik, dengan tingkat akurasi posisi mencapai rata-rata *error* 1,30% serta waktu respons yang tidak lebih dari 0,35 detik pada seluruh rentang sudut pengujian, sementara *relay* terbukti konsisten mengendalikan *pilot lamp* sesuai hasil validasi pada seluruh skenario kegagalan maupun keberhasilan transaksi. Kecepatan eksekusi mekanis oleh aktuator yang berada pada orde ratusan milidetik ini jauh lebih cepat dibandingkan dengan waktu komunikasi data pada tahap identifikasi dan validasi (yang berada pada orde ratusan milidetik hingga mendekati 1 detik), sehingga hambatan waktu terbesar pada keseluruhan proses transaksi teridentifikasi berasal dari tahap komunikasi data antara ESP32 dan Firebase, bukan dari keterbatasan mekanis perangkat keras.

Sebagai gambaran performa waktu proses secara menyeluruh, apabila ketiga tahapan waktu komunikasi data yang telah diuji secara terpisah pada pengujian sebelumnya digabungkan, yaitu waktu identifikasi hingga data tersimpan (rata-rata

0,97 s pada **Tabel 4.3**), waktu pengambilan data pengguna dari Firebase (rata-rata 0,50 s pada **Tabel 4.4**), dan waktu penulisan hasil transaksi ke *log* (rata-rata 0,60 s pada **Tabel 4.4**), Estimasi total waktu proses dihitung menggunakan Persamaan (3.4) sebagai berikut: 0,97 + 0,50 + 0,60 = 2,07 detik. Dengan demikian, estimasi total waktu proses dari kendaraan teridentifikasi hingga hasil transaksi tercatat sekitar 2,07 detik. Waktu ini memang masih lebih lambat dibandingkan waktu pelayanan transaksi pada sistem SLFF/Flo yang telah beroperasi secara komersial (0,6 detik per kendaraan), namun perlu digarisbawahi bahwa perbandingan ini belum sepenuhnya setara, mengingat pengujian pada penelitian ini masih menggunakan jaringan *hotspot smartphone* yang notabene memiliki karakteristik sinyal berbeda dengan infrastruktur jaringan tetap yang digunakan pada sistem komersial, serta prototipe ini belum diuji dalam kondisi kendaraan bergerak melintasi gerbang secara langsung.

Berdasarkan keseluruhan hasil pengujian tersebut, dapat disimpulkan bahwa prototipe sistem transaksi gerbang tol *free flow* berbasis IoT yang dikembangkan dalam penelitian ini telah berhasil memenuhi tujuan penelitian yang dirumuskan, yaitu mengimplementasikan sistem identifikasi dan transaksi kendaraan secara otomatis, mengintegrasikan proses tersebut dengan Firebase Realtime Database untuk pengolahan dan penyimpanan data secara *real-time*, serta menyediakan antarmuka pemantauan berupa *dashboard* yang akurat dan responsif. Meski demikian, beberapa keterbatasan pengujian yang teridentifikasi selama penelitian ini, di antaranya penggunaan jaringan *hotspot* yang bersifat sementara, menjadi dasar bagi penulis dalam merumuskan saran pengembangan lebih lanjut pada BAB V.