

Perbandingan Server Sent Event & Websocket untuk Integrasi Website & Module RFID

Rio Irawan¹, I Made Suartana²

^{1,2}Jurusan Teknik Informatika, Fakultas Teknik, Universitas Negeri Surabaya

¹rioirawan.20084@mhs.unesa.ac.id

²madesuartana@unesa.ac.id

Abstrak— Perkembangan teknologi, khususnya dalam bidang sistem dan informasi, telah memungkinkan digitalisasi dan otomatisasi berbagai aspek kehidupan, termasuk industri ritel melalui adopsi *Self Checkout System*. *Self Checkout System* memanfaatkan teknologi *RFID (Radio Frequency Identification)* untuk mendukung transaksi belanja tanpa kasir. *RFID* memungkinkan identifikasi dan pengambilan data nirkabel secara otomatis melalui komponen utama berupa *tag* dan *reader*. Namun, integrasi antara aplikasi *web* dan perangkat *RFID* menghadirkan tantangan, terutama dalam hal komunikasi data secara real-time. Penelitian ini mengkaji dua protokol komunikasi data, *WebSocket* dan *Server Sent Event (SSE)*, dalam mengintegrasikan sistem website dengan perangkat *RFID*. *WebSocket* menawarkan jalur komunikasi cepat dan efisien dengan koneksi tunggal, sementara *SSE* lebih efisien dalam penggunaan memori pada komunikasi berbasis *event*. Penelitian ini membandingkan kecepatan pengiriman dan penerimaan data, serta penggunaan memori dan *bandwidth* dari kedua protokol, untuk menentukan protokol yang lebih efektif dalam sistem *self checkout*. Hasil penelitian didapatkan *WebSocket* lebih unggul daripada *Server Sent Event (SSE)* dengan delay waktu 3.5 - 4.2 miliseconds dengan *SSE* 4.1 - 5.2 miliseconds. Sedangkan untuk penggunaan memori *websocket* menggunakan sekitar 210 - 340K dan *SSE* menggunakan 208-260K memori. Akan tetapi dengan mempertimbangkan Keunggulan *websocket* dalam komunikasi nya yang mendukung dua arah atau full duplex, juga mendukung format data yang lebih banyak daripada *SSE* yang hanya mendukung text atau tipe data String. Bisa disimpulkan dalam penelitian ini protokol komunikasi untuk integrasi antara *RFID* dengan aplikasi berbasis website, *websocket* lebih unggul dalam Delay dan beberapa aspek lain dibandingkan dengan *Server Sent Event*.

Kata Kunci— *Server Sent Event, WebSocket, Radio Frequency Identification, website, Website, Integration, Self Checkout*

I. PENDAHULUAN

Pekembangan teknologi apalagi yang berhubungan dengan sistem dan informasi di beberapa tahun terakhir semakin *modern* dan membuat hampir semua hal dapat didigitalisasi dan di automisasi dengan memanfaatkan peran teknologi [1]. Salah satu adopsi teknologi pada industri adalah *Self checkout system*. *Self checkout system* merupakan sistem yang mendukung

kegiatan berbelanja agar saat proses pembayaran dan pengecekan bisa secara otomatis bisa dilakukan tanpa memerlukan bantuan manusia sebagai kasir seperti sebelumnya [2]. Salah satu teknologi yang mendukung dan biasa digunakan untuk membangun dan membuat *self checkout system* yaitu *RFID* atau *Radio Frequency Identification*. Teknologi *RFID* adalah suatu sistem otomatis yang bekerja atau berfungsi dengan *tag* dan *reader*, dimana *tag(transponder)* dan *reader* adalah komponen utama dari *RFID* yang berfungsi untuk pengambilan data *nirkabel* secara otomatis[9]. *Reader RFID* dapat membaca *chip* silikon yang ada pada didalam tag yang dapat dibaca gelombang *radio tag*, identifikasi ini bersifat unik dan berbeda setiap tag satu dengan tag lainnya. *Reader* dapat menangkap atau mendeteksi gelombang radio yang dipancarkan pada radius tiga sampai tiga puluh kaki dan mampu membaca data digital yang terdapat pada chip, jarak dan jangkauan dari sinyal *RFID* tergantung pada frekuensi radio yang digunakan dan spesifikasi dari perangkat *RFID* itu sendiri. [3].

Tantangan yang sering dihadapi dalam proses pembuatan aplikasi yang bisa mendukung *self checkout system* ini adalah proses integrasi antara aplikasi yang berbasis *website* dengan perangkat *RFID*. *RFID* yang sudah membaca atau memindai tag harus bisa mengirim data berupa kode yang mengidentifikasi suatu *tag* kedalam sistem aplikasi berbasis *web* yang sudah dibuat. Juga sebaliknya, dari sistem *web* mengirim data berupa kode ke perangkat *RFID* dan perangkat mengirimkannya kembali kedalam *tag* agar bisa menyimpan data kode kedalam suatu tag tertentu. Dengan begitu harus ada jalur komunikasi data yang *Realtime* atau secara langsung bisa menghubungkan aplikasi yang berbasis *website* dengan perangkat *RFID*, pengiriman data tersebut harus cepat dan efisien agar data yang dibaca oleh *RFID* bisa langsung diolah oleh sistem web aplikasi.

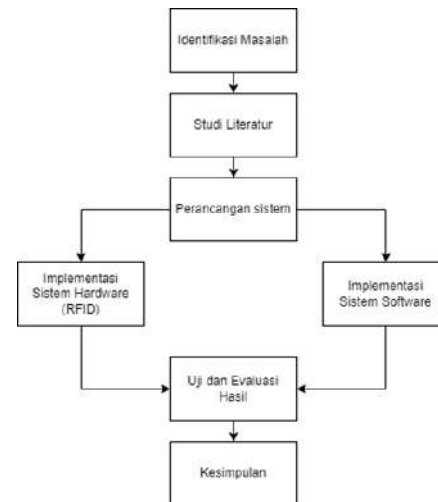
Sejauh ini metode komunikasi pengiriman data yang paling efektif untuk pengiriman data dari perangkat *hardware* kepada *website* adalah dengan menggunakan *websocket*, itu dikarenakan *websocket* akan memberikan data yang baru dan langsung ketika *server websocket* menerima data atau ada data yang masuk dari sistem yang ada didalam maupun dari luar sistem itu sendiri. Maka dari itu *websocket* disebut sebagai *protocol TCP* yang menggunakan koneksi tunggal, dengan begitu *websocket* hanya membutuhkan satu jalur untuk mengirim data ke sisi klien yang memiliki jalur hubungan pada server dan juga port [4]. Salah satu penelitian sebelumnya yang berjudul "*Implementasi Sistem Kontrol dan Monitoring Smart*

Home Menggunakan Integrasi Protokol Websocket dan MQTT” menyebutkan jika komunikasi dari perangkat *hardware* dengan sistem *IoT* (*Internet of Things*) cukup lancar dan efektif dengan menggunakan *websocket* sebagai jalur komunikasi yang menjembatani antara aplikasi *software* yang berbasis *website* dengan perangkat *hardware*[8]. Tetapi dalam penelitian tersebut penulis mengkombinasikan jalur komunikasi menggunakan *MQTT* (*Message Queueing Telemetry Transport*) yang memang protokol yang biasa digunakan untuk pembuatan *IoT*[5]. Selain itu ada protokol komunikasi lain yaitu *Server Sent Event* (*SSE*) yang dimana *Server Sent Event* protokol komunikasi yang berbasis pada *event* tertentu untuk mengirim data, berbeda dengan *websocket* yang berbasis *polling* dimana akan terus menerima dan *update* data secara berkala *Server Sent Event* berfokus pada *Event* tertentu sebagai pemicu untuk pengiriman data. *Server Sent Event* (*SSE*) ini adalah protokol komunikasi satu arah dimana ada sisi *Client* dan *Server*, jadi tidak memungkinkan untuk komunikasi dua arah seperti *Websocket*. Akan tetapi *Server Sent Event*(*SSE*) dinilai lebih efisien untuk komunikasi data yang berbasis pada *event* tertentu sehingga penggunaan memori pada proses komunikasi bisa lebih rendah daripada menggunakan protokol komunikasi *polling* seperti *websocket*.

Dengan mempertimbangkan masalah integrasi yang cukup menjadi tantangan dalam proses menghubungkan aplikasi yang berbasis *website* dengan perangkat *hardware*, Penulis mencoba melakukan implementasi *websocket* dan *Server Sent Event*(*SSE*) dalam proses komunikasi data secara *Realtime* dengan perangkat *RFID* dalam sistem *self checkout*, dan dalam penelitian ini penulis mencoba membandingkan jalur komunikasi *websocket* dengan *Server Sent Event* (*SSE*) untuk mempertimbangkan protokol komunikasi data yang lebih cepat dalam proses pengiriman dan penerimaan data, sekaligus penggunaan memori yang lebih rendah agar tidak memberatkan sistem ketika sudah terhubung dengan *website*.

II. METODE PENELITIAN

Sebelum akhirnya mengimplementasikan dan menguji sistem yang akan dibuat, ada beberapa langkah yang harus dilakukan dalam penelitian, dan ada beberapa alur dari penelitian yang harus dilakukan.



Gbr 1 alur penelitian

A. Identifikasi Masalah

Tahap ini untuk mengidentifikasi masalah yang ada untuk akhirnya dijadikan sebagai suatu syarat dan pedoman untuk menyusun tugas akhir yang merupakan tujuan dari penelitian ini untuk mencari suatu solusi dari suatu masalah dan membuat aplikasi yang bisa.

B. Studi Literatur

Studi literatur ini dilakukan dengan mencari sumber informasi yang berhubungan dengan masalah yang diangkat sebagai fokus penelitian melalui berbagai media, seperti jurnal, artikel, internet dan berbagai sumber informasi lainnya yang nantinya akan dijadikan sebagai referensi untuk penelitian ini. Beberapa hal penting yang dalam penelitian ini seperti tentang *Websocket*, *RFID*, *Rest API* yang merupakan komponen penting dari penelitian ini, jadi sebelum merancang sistem penulis mencari referensi tentang dasar teori yang dibutuhkan untuk penyusunan dan implementasi sistem.

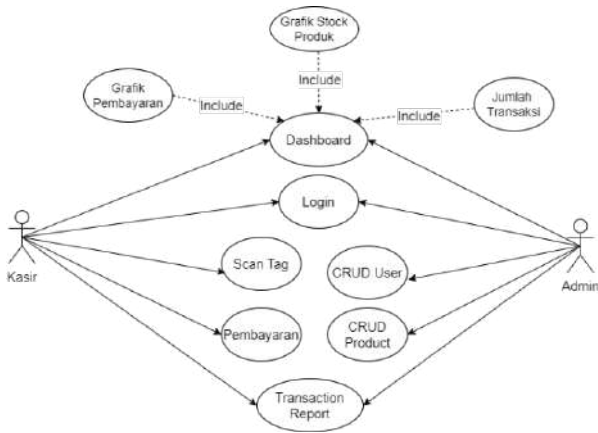
C. Perancangan Sistem

Pada tahap ini akan perancangan sistem meliputi analisis dan perancangan struktur sistem yang akan dibuat. Perancangan sistem sangat penting untuk Implementasi *Websocket* di sistem *self checkout* berbasis *website* dan *module RFID* (*Radio Frequency Identification*), karena perancangan sistem bisa mempermudah ke implementasi yang lebih detail nantinya. Dalam perancangan ini meliputi mulai dari analisis kebutuhan sistem, perancangan sistem *hardware*, perancangan sistem *software*.

1) Perancangan Sistem Software (Website)

Dalam perencanaan sistem, termasuk dalam perancangan alur kerja sistem, terdapat beberapa tahapan seperti merancang *Use Case Diagram*, *Activity Diagram*, *Sequence Diagram*.

a. Use Case Diagram

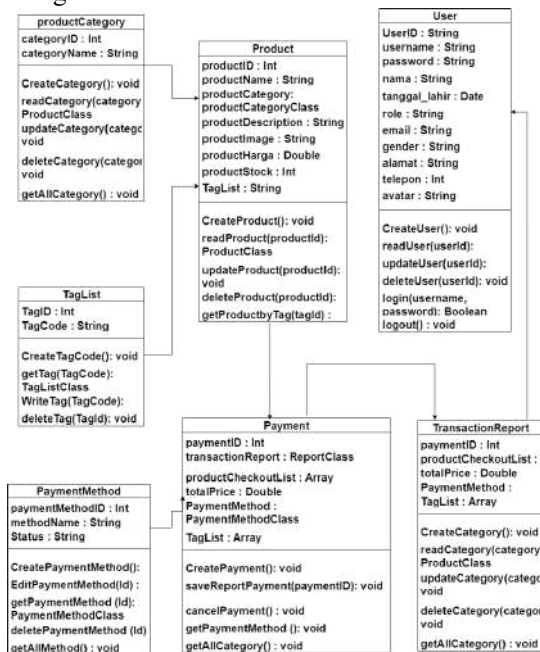


Gbr 2. use case diagram

Pada Gbr 2 adalah menjelaskan diagram *Use Case* yang mencakup peran Admin dan User sebagai Aktor, di mana setiap Aktor memiliki perannya masing-masing terkait fitur-fitur dalam sistem.

b. Class Diagram

Class diagram adalah jenis diagram dalam pemodelan UML (*Unified Modeling Language*) yang digunakan untuk menggambarkan struktur statis dari sistem dengan memperlihatkan kelas-kelas yang ada, hubungan antara kelas-kelas tersebut.

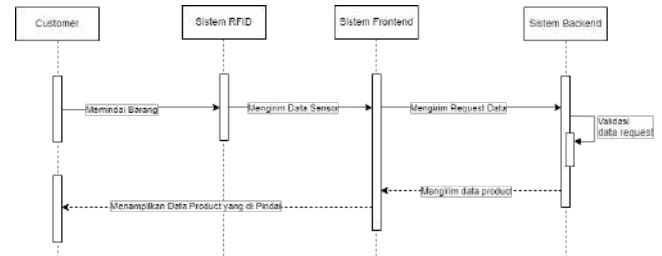


Gbr 3 class diagram

Pada Gbr 3 dijelaskan tentang *class diagram* yang menyusun sistem ini, diagram ini menjelaskan dan menampilkan komponen – komponen dan *class* beserta relasinya yang akan digunakan dalam pembuatan sistem *self checkout* ini.

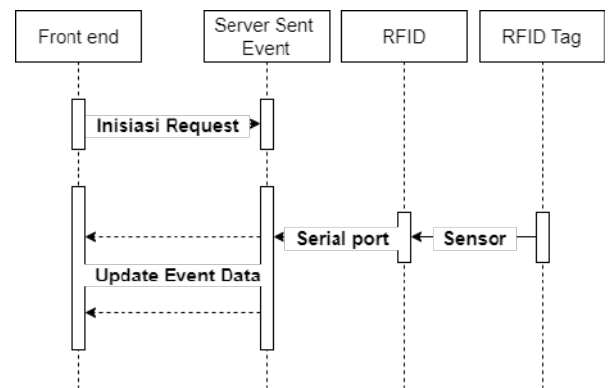
2) Perancangan Sistem RFID (Radio Frequency Identification)

Dalam perancangan Sistem RFID sangat sederhana sekali karena interaksi yang dibutuhkan ketika membuat sistem ini adalah bagaimana *RFID Reader* sebagai sensor dapat terhubung dengan program javascript untuk akhirnya dibaca dan di teruskan kepada *frontend* melalui *websocket*.



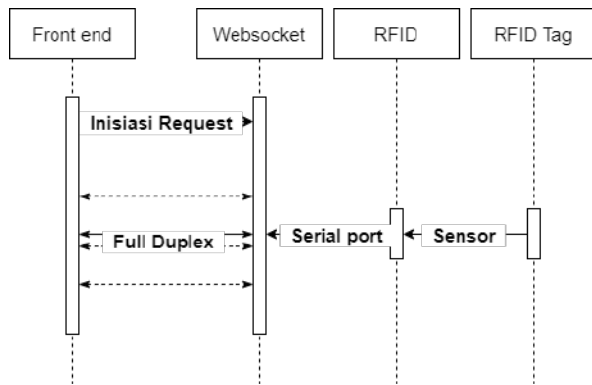
Gbr 4 scan tag sequence diagram

Pada Gbr 4 Sequence Diagram tersebut adalah skema atau alur pembacaan *RFID Tag* yang nantinya dapat dibaca oleh program *javascript* sehingga bisa dikirimkan melalui *websocket* untuk kebutuhan sistem *website* dalam identifikasi produk berdasarkan tag yang sudah ada.



Gbr 5 sequence diagram server sent events

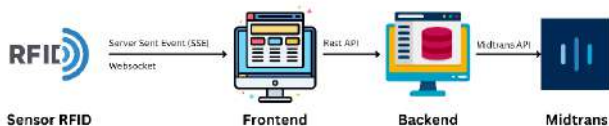
Pada Gbr 5 bisa dilihat interaksi komponen *RFID* dengan sisi *Front End*, dengan *Sever Sent Event* sebagai perantara yang membuat koneksi dengan *Front End*. Pertama diawali dengan inisiasi *Request* untuk membuat koneksi dengan protokol komunikasi agar bisa menerima semua data dari *RFID* yang sudah masuk ke kedalam Sensor dan dilanjutkan dengan *Serial Port*, dan *Server Event*.



Gbr 6 sequence diagram websocket

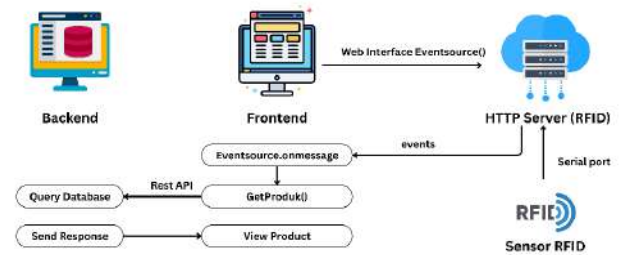
Pada Gbr 6 adalah interaksi RFID dengan sisi Frontend dengan Websocket Sebagai protokol komunikasi antara RFID dengan sisi Client pada Frontend, dan dalam komunikasi ini memakai rangkaian alur data yang sama dengan server sent events dimana ketika sensor menerima data dan Sensor RFID akan menerima data yang terdeteksi, maka Sistem RFID akan membuat Event Listener untuk membuat koneksi data dengan websocket setiap data baru yang dikirim melalui serial port. Perbedaan dari Server Sent Events disini adalah koneksi websocket yang dengan frontend merupakan koneksi full duplex dimana koneksi tersebut merupakan koneksi dua arah dimana client dan sever bisa saling mengirim pesan atau data melalui emitter, salah satu event listener websocket yang bisa digunakan untuk mengirim data yang masuk dengan event listener tertentu.

3) Arsitektur Sistem



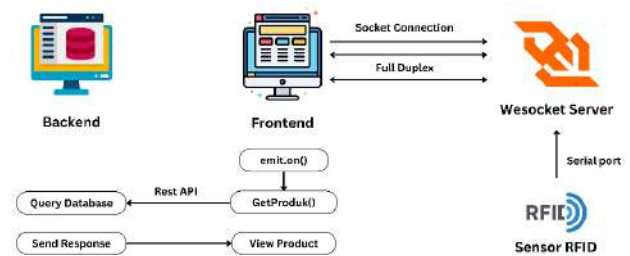
Gbr 7 Arsitektur Sistem Websocket

Pada Gbr 7 menunjukkan arsitektur sistem dari selfcheckout ini yang terdiri dari Sisi Client atau Frontend, sistem RFID dan Server atau Backend. Sistem Frontend dan Sistem RFID akan di hubungkan dengan Websocket dan SSE untuk melihat perbandingan, dan sistem frontend dan system backend akan dihubungkan dengan Rest API.



Gbr 8 arsitektur sistem server sent events

Pada Gbr 8 adalah Arsitektur yang lebih detail dalam sistem server sent event, dimana diawali dengan insisiasi web interfas dari javascript dan server akan langsung mengirimkan data ke Frontend setiap ada event yang masuk.



Gbr 9 arsitektur sistem websocket

Pada Gbr 9 bisa dilihat Arsitektur websocket sedikit berbeda dari SSE, Websocket memiliki koneksi full duplex yang dapat dengan langsung mengirimkan data tanpa harus menunggu event baru masuk, karena setiap detik kedua koneksi websocket melakukan Ping koneksi sever.

D. Implementasi Sistem

Implementasi sistem disini akan menjadi bagian utama dalam penelitian karena dalam penelitian fokus pada implementasi Websocket pada sistem self checkout yang berbasis website dimana hasil dari implementasi ini nantinya akan menjadi semacam hasil yang bisa di analisis. Implementasi sistem disini juga akan dibagi menjadi dua bagian dimana ada bagian software dan ada bagian hardware. Pembagian ini ditujukan untuk lebih terstruktur dalam proses penelitian yang akan dilakukan.

E. Pengujian Sistem

Hasil implementasi yang sudah bisa digunakan dan bisa berfungsi dengan baik akan dilanjutkan ke proses pengujian untuk membandingkan protokol komunikasi mana yang lebih baik dengan memerhatikan beberapa kondisi yaitu waktu delay, dan penggunaan memori. Software yang digunakan sebagai alat bantu untuk mengukur delay dan penggunaan memori adalah timestamp yang ada dalam javascript dan Chrome Task Manager. Timestamp javascript sendiri bisa digunakan untuk mengukur delay dengan mencari selisih respon dibuat dan respon diterima pada client, dengan penjelasan berikut :

D = Delay dari setiap pengiriman data
RS (*Response Send*) = Waktu pembuatan *respons* pada *server*
RR (*Response Receive*) = Waktu *respons* diterima di *client*

$$D = RR (\text{Response Receive}) - RS (\text{Response Send}) \dots \dots \dots (1)$$

Lalu *Chrome Task Manager* yang dimana adalah salah satu fitur yang sudah disediakan oleh *chrome* untuk melihat penggunaan memori yang digunakan oleh suatu *website* yang sudah berjalan, pengukuran ini akan digunakan untuk membandingkan banyak *resource* yang terpakai Ketika menggunakan *Server Sent Event* dan *Websocket*.

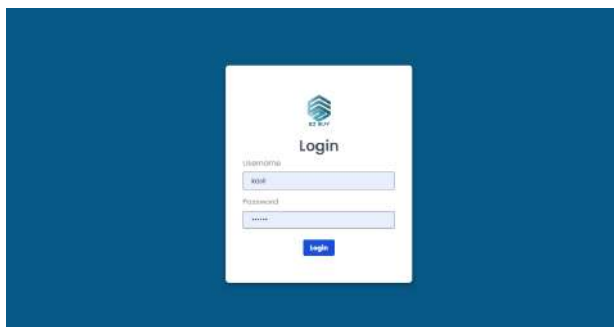
III. HASIL DAN PEMBAHASAN

A. Pengembangan Aplikasi

Penelitian ini akan menggunakan 2 sistem yang berbeda dimana ada *sistem RFID* untuk pembacaan *RFID Tag*, dan proses pengiriman data *RFID Tag* ke aplikasi *website* untuk identifikasi data produk yang memiliki *ID* sesuai dengan yang sudah di baca oleh *RFID Reader*.

1) Aplikasi Website

Aplikasi Website ini disusun dengan *Vue.js* dan *Express.js* untuk *server*, aplikasi ini berguna untuk menerima data dari *RFID* dan mengidentifikasinya dengan data *RFID Tag* yang sudah tersimpan didalam *database*.



Gbr 10 Tampilan Login Website

Pada Gbr 10 adalah tampilan awal sebelum masuk kedalam sistem utama dari aplikasi *self checkout* pada login ini bisa digunakan untuk keamanan dari aplikasi. Login disini mengguakan *username* dan *password* yang didapatkan dari fungsi penambahan data user yang ada di admin.

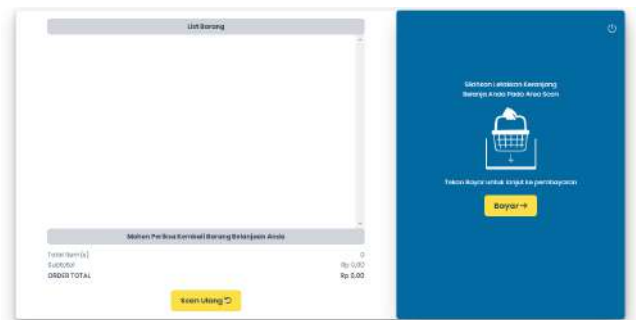


Gbr 11 Tampilan Dashboard

Pada Gbr 11 adalah tampilan *Dashboard* yang ditampilkan setelah berhasil *login*, dalam *dashboard* ini terdapat beberapa tampilan yang menunjang informasi , seperti Riwayat transaksi, Riwayat transaksi yang berhasil, dan *stock* produk.

Gbr 12 Tampilan Report Transaksi

Pada Gbr 12 adalah tampilan laporan transaksi dalam aplikasi yang sudah dilakukan, lengkap dengan status transaksi, mulai dari transaksi tersebut sukses, gagal, atau dibatalkan. yang nantinya bisa didownload kedalam bentuk pdf, sebagai *invoice* atau bukti pembayaran.



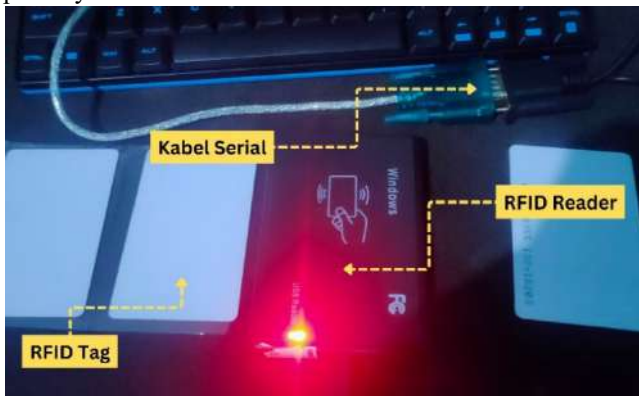
Gbr 13 Tampilan Website pemindaian RFID

Tampilan pada Gbr 13, terlihat tampilan awal aplikasi *website* sebelum menerima data dari sensor *RFID* , ini akan menampilkan data produk yang sesuai dengan identifikasi *RFID tag* yang terhubung ke *database*.

2) Sistem RFID

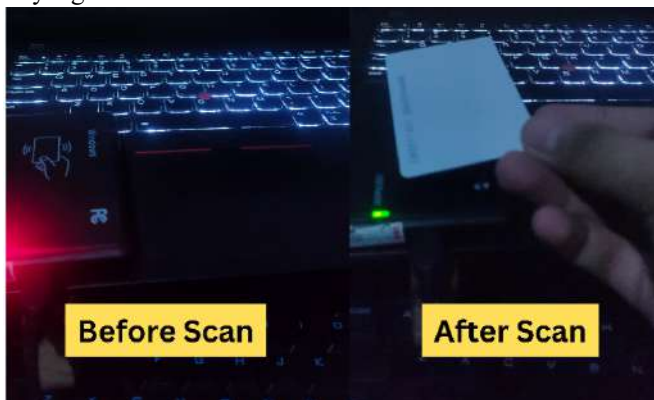
Sistem *RFID* akan berisi *server Websocket* atau *Server Sent Event (SSE)* yang akan digunakan untuk membaca data dari sensor *RFID* dan mengirimkannya ke sisi *Frontend*

atau *Client* sebagai penerima untuk diproses ke identifikasi pembayaran.



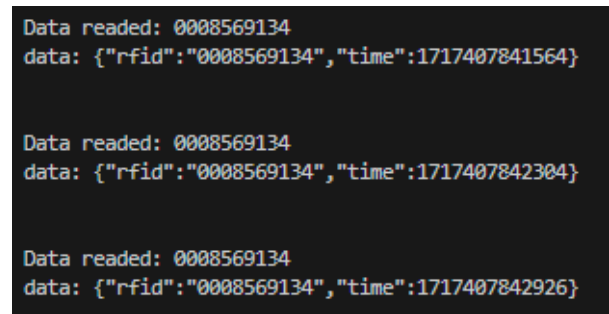
Gbr 14 dokumentasi pengujian

Pada Gbr 14 adalah dokumentasi alat yang digunakan sebelum pengujian dimulai dimana terdapat beberapa *hardware* yang harus disiapkan meliputi *RFID Reader*, *RFID Tag*, dan kabel *serial* untuk transfer data kedalam komputer tujuan. *RFID Tag* yang akan dibaca juga harus sudah memiliki ID *Tag* yang tersimpan pada memori dalam *RFID Tag*, untuk bisa membaca *RFID Tag* perangkat harus disambungkan kepada komputer dari *USB Port* sebagai daya utama untuk *RFID Reader* yang akan digunakan untuk memindai *RFID Tag*. Kabel *Serial* juga harus dihubungkan ke-komputer untuk sensor *RFID* bisa mengirimkan data yang sudah terdeteksi atau terbaca.



Gbr 15 proses pengujian scan tag

Pada Gbr 15 adalah proses pemindaian *Tag RFID*. Bisa dilihat lampu dan *buffer* dari *RFID Reader* akan berwarna merah dan akan berubah berwarna hijau juga akan ditandai dengan bunyi *buffer* dari *RFID Reader* yang menunjukkan *RFID Tag* berhasil di scan dengan baik dan *sensor* berhasil mengirimkan data kepada komputer melalui kabel serial yang sudah dihubungkan, tahap selanjutnya adalah pembuatan skema pembacaan data melalui *serial port* agar bisa mengatur apa yang akan dilakukan oleh sistem *RFID* Ketika data dari sensor berhasil membaca *RFID tag*.



Gbr 16 Hasil pembacaan sensor *RFID*

Pada gambar 16 terlihat hasil output yang dihasilkan saat data terbaca oleh server dan membuat variable time sebagai timestamp untuk mengukur delay atau response time.

B. Hasil Penelitian *RFID* dengan *Websocket*

Hasil penelitian menggunakan *websocket* akan dilakukan dengan mengirimkan data dari sensor *RFID* ke sisi *Client* atau *Frontend* sebanyak 3 kali pengujian, dan 10 kali pengiriman data disetiap pengujian. Pengujian akan berfokus pada waktu delay pengiriman data dari *RFID* sensor ke *Frontend website (Client)*. Pengujian ini menggunakan *websocket* sebagai jalur komunikasi antara sistem *RFID* dengan Aplikasi *Frontend*.

TABEL I PENGUJIAN 1 WEBSOCKET

Parameter	Pengujian									
	1	2	3	4	5	6	7	8	9	10
Delay (milliseconds)	3	3	4	6	6	5	3	3	4	5
Rata- rata	4,2 milliseconds									
Penggunaan Resource	210,880 K									

Pada Tabel I terlihat pengujian pertama menggunakan jalur komunikasi *websocket*, delay dengan rata – rata di 4.2 milliseconds, dan penggunaan *resource* sebesar 210,880 K.

Berikut adalah hasil pengujian ke-2 dari jalur komunikasi *Websocket* :

TABEL II PENGUJIAN 2 WEBSOCKET

Parameter	Pengujian									
	1	2	3	4	5	6	7	8	9	10
Delay (milliseconds)	3	3	4	3	3	4	5	3	3	4
Rata- rata	3,5 milliseconds									

Penggunaan Resource	276,432 K
---------------------	-----------

Pada Tabel II terlihat pengujian ke-2 menggunakan *websocket*, *delay* dengan rata – rata di 3.5 *milliseconds*, dan penggunaan *resource* yaitu 276,432 K.

Berikut adalah hasil pengujian ke-3 dari jalur komunikasi *Websocket* :

TABEL III PENGUJIAN 3 WEBSOCKET

Parameter	Pengujian									
	1	2	3	4	5	6	7	8	9	10
Delay (<i>milliseconds</i>)	3	3	3	4	5	5	3	3	4	4
Rata- rata	3,7 <i>milliseconds</i>									
Penggunaan Resource	341,200 K									

Pada Tabel III terlihat pengujian ke-3 menggunakan jalur komunikasi *websocket*, *delay* pengiriman data pada setiap komunikasi dengan data rata – rata di 3.7 *milliseconds*, dan penggunaan *resource* atau memori yaitu 341,200 K.

Berikut adalah hasil pengujian ke-4 dari jalur komunikasi *Websocket* :

TABEL IV PENGUJIAN 4 WEBSOCKET

Parameter	Pengujian									
	1	2	3	4	5	6	7	8	9	10
Delay (<i>milliseconds</i>)	3	4	4	5	5	3	5	4	3	3
Rata- rata	3,9 <i>milliseconds</i>									
Penggunaan Resource	309,060 K									

Pada Tabel IV terlihat pengujian ke-4 menggunakan jalur komunikasi *websocket*, *delay* pengiriman data dengan rata – rata di 3.9 *milliseconds*, dan penggunaan *resource* yaitu 309,060K.

Berikut adalah hasil pengujian ke-5 dari jalur komunikasi *Websocket* :

TABEL V PENGUJIAN 5 WEBSOCKET

Parameter	Pengujian									
	1	2	3	4	5	6	7	8	9	10
Delay (<i>milliseconds</i>)	6	3	4	5	3	3	6	5	3	3

Rata- rata	4.1 <i>milliseconds</i>
Penggunaan Resource	296,132 K

Pada Tabel V terlihat pengujian ke-4 menggunakan jalur komunikasi *websocket*, *delay* pengiriman data pada setiap komunikasi dengan data rata – rata di 4.1 *milliseconds*, dan penggunaan *resource* atau memori yaitu 296,132 KB.

TABEL VI HASIL RATA - RATA PENGUJIAN WEBSOCKET

Parameter	Pengujian				
	Uji ke -1	Uji ke -2	Uji ke -3	Uji ke -4	Uji ke -5
Delay <i>milliseconds</i>	4.2 ms	3.5 ms	3.7 ms	3.9 ms	4.1 ms
Penggunaan Resource	210,88 K	276,43 K	341,20 K	309,06 K	296,132 K

Pada Tabel VI bisa dibandingkan Rata – rata *delay* dan penggunaan *resource* dari setiap pengujian. Dimana rata – rata *delay* stabil di 3,5 – 4,2 *milliseconds* , dan rata – rata penggunaan *resource* yang kurang lebih berada di sekitar 200 -350 K.

C. Hasil Penelitian RFID dengan Server Sent Event (SSE)

Pengiriman data menggunakan *SSE* sedikit berbeda dengan *Websocket* karena *SSE* berbasis pada event, maka *SSE* hanya akan mengirimkan data kepada *client* ketika ada data yang masuk melalui sensor, server sent ini menggunakan jalur komunikasi *HTTP* pada umumnya, akan tetapi menggunakan *EventSource()* dari javascript untuk menangkap *response*, dan inisiasi *request* hanya dilakukan satu kali dan data dari *server* akan terus dikirimkan ketika ada *event* atau data baru yang masuk dari sensor. Pengujian ini menggunakan *Server Sent Event (SSE)* sebagai jalur komunikasi antara *RFID* dengan sisi *Frontend*.

TABEL VII PENGUJIAN 1 SERVER SENT EVENT

Parameter	Pengujian									
	1	2	3	4	5	6	7	8	9	10
Delay (<i>milliseconds</i>)	6	5	5	6	4	5	6	5	4	5
Rata- rata	5,1 <i>milliseconds</i>									
Penggunaan Resource	208,194 K									

Tabel VII terlihat pada pengujian pertama menggunakan jalur komunikasi *Server Sent Event (SSE)*, *delay* pengiriman data pada setiap komunikasi dengan data rata – rata di 5.1 *milliseconds*, dan penggunaan *resource* sebesar 210,880 K.

Berikut adalah hasil pengujian ke-2 dari jalur komunikasi *Server Sent Event (SSE)* :

TABEL VIII PENGUJIAN 2 SERVER SENT EVENT

Parameter	Pengujian									
	1	2	3	4	5	6	7	8	9	10
Delay (milliseconds)	6	5	3	3	4	3	5	3	5	4
Rata- rata	4.1 milliseconds									
Penggunaan Resource	216,390 K									

Pada Tabel VIII terlihat pengujian ke-2 menggunakan jalur komunikasi *Server Sent Event*, delay pengiriman data pada setiap komunikasi dengan data rata – rata di 4.1 milliseconds, dan penggunaan resource yaitu 216,390 K. Berikut adalah hasil pengujian ke-3 dari jalur komunikasi *Server Sent Event (SSE)* :

TABEL IX PENGUJIAN 3 SERVER SENT EVENT

Parameter	Pengujian									
	1	2	3	4	5	6	7	8	9	10
Delay (milliseconds)	7	4	5	7	4	3	3	5	4	4
Rata- rata	4.6 milliseconds									
Penggunaan Resource	260,390 K									

Pada Tabel IX terlihat pengujian ke-3 menggunakan jalur komunikasi *Server Sent Event*, delay pengiriman dengan rata – rata di 4.6 milliseconds, dan penggunaan resource yaitu 260,390K.

Berikut adalah hasil pengujian ke-4 dari jalur komunikasi *Server Sent Event (SSE)* :

TABEL X PENGUJIAN 4 SERVER SENT EVENT

Parameter	Pengujian									
	1	2	3	4	5	6	7	8	9	10
Delay (milliseconds)	7	4	5	7	4	3	3	5	4	4
Rata- rata	4.6 milliseconds									
Penggunaan Resource	254,311 K									

Pada Tabel X terlihat pengujian ke-4 menggunakan jalur komunikasi *Server Sent Event*, delay pengiriman data pada setiap komunikasi dengan data rata – rata di 4.6 milliseconds, dan penggunaan resource yaitu 254,311 K.

Berikut adalah hasil pengujian ke-5 dari jalur komunikasi *Server Sent Event (SSE)* :

TABEL XI PENGUJIAN 5 SERVER SENT EVENT

Parameter	Pengujian									
	1	2	3	4	5	6	7	8	9	10
Delay (milliseconds)	7	4	5	7	4	4	6	5	4	6
Rata- rata	5.2 milliseconds									
Penggunaan Resource	257,370 K									

Pada Tabel XI terlihat pengujian ke-5 menggunakan *Server Sent Event*, delay pengiriman data dengan rata – rata di 5.2 milliseconds, dan penggunaan resource yaitu 257,370K.

TABEL XII HASIL RATA - RATA PENGUJIAN SERVER SENT EVENT

Parameter	Pengujian				
	Uji ke -1	Uji ke -2	Uji ke -3	Uji ke -4	Uji ke -5
Delay (milliseconds)	5.1 ms	4.1 ms	4.6 ms	4.6 ms	5.2 ms
Penggunaan Resource	208,194 K	216,390 K	260,390 K	254,311 K	257,370 K

Pada Tabel XII bisa dibandingkan Rata – rata delay dan penggunaan resource dari setiap pengujian. Dimana rata – rata delay di 4.1 – 5.2 milliseconds , dan rata – rata penggunaan resource yang kurang lebih berada di sekitar 200 -270 K.

D. Perbandingan Delay & Penggunaan Resource Websocket dengan Server Sent Event

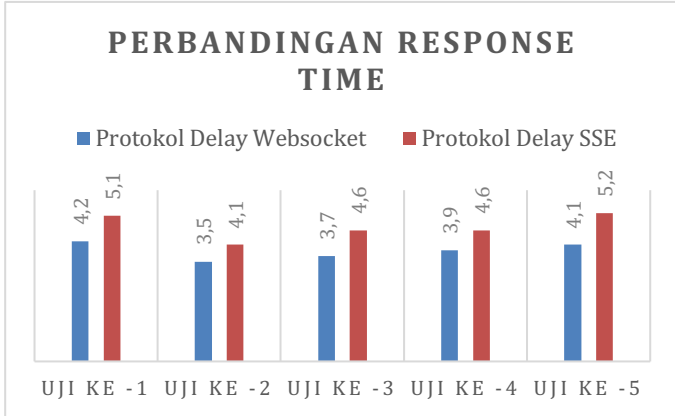
Setelah mendapatkan data hasil pengujian komunikasi *RFID* dengan *Websocket* dan komunikasi *RFID* dengan *Server Sent Event (SSE)* , hasil tersebut dapat dibandingkan untuk mengetahui *performa* metode komunikasi yang diukur dari Delay dan penggunaan memori.

TABEL XIII PERBANDINGAN DELAY SERVER SENT EVENT DENGAN WEBSOCKET

Pengujian	Delay Websocket	Delay SSE
Uji ke -1	4.2 ms	5.1 ms
Uji ke -2	3.5 ms	4.1 ms
Uji ke -3	3.7 ms	4.6 ms
Uji ke -4	3.9 ms	4.6 ms
Uji ke -5	4.1 ms	5.2 ms

Pada Tabel XIII bisa dibandingkan Rata – rata delay dari setiap pengujian antara *Websocket* dan *Server Sent Event*, bisa

dilihat dalam awal sampai akhir pengujian *delay websocket* dan *SSE* memiliki perbedaan dimana *Websocket* lebih unggul daripada *Server Sent Event* dengan *delay waktu 3.5 - 4.2 milliseconds* dengan *Server Sent Event 4.1 - 5.2 milliseconds* , yang berarti *websocket* memiliki *response time* yang lebih cepat daripada *Server Sent Event*.



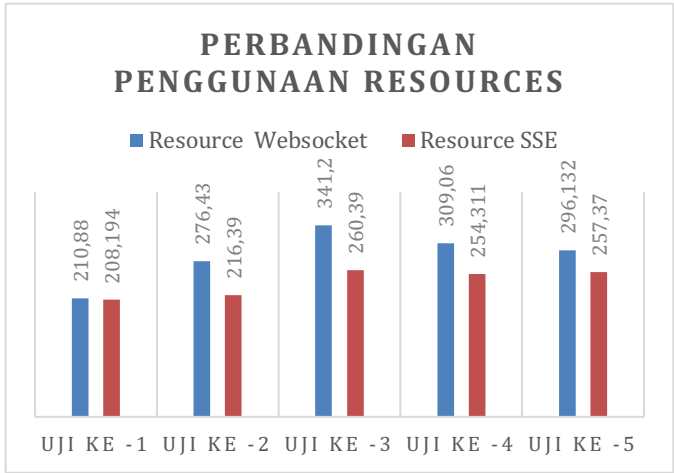
Gbr 17 Grafik perbandingan Response Time (Milliseconds)

Pada Gbr 17 bisa dilihat dalam grafik jika *Response Time* atau *Delay* dari *websocket* selalu dibawah *Server Sent Event (SSE)*.

TABEL XIV PERBANDINGAN PENGGUNAAN RESOURCES SERVER SENT EVENT DENGAN WEBSOCKET

Pengujian	Resource Websocket	Resource SSE
Uji ke -1	210,88 K	208,194 K
Uji ke -2	276,43 K	216,390 K
Uji ke -3	341,20 K	260,390 K
Uji ke -4	309,060 K	254,311 K
Uji ke -5	296,132 K	257,370 K

Pada Tabel XIV bisa dibandingkan Penggunaan *Resource* atau memori dari setiap pengujian antara *Websocket* dan *Server Sent Event*, bisa dilihat dalam awal sampai akhir pengujian penggunaan memori *websocket* lebih besar dari pada *Server Sent Event* , dimana *websocket* menggunakan sekitar 210 – 340K dan *Server Sent Event* menggunakan 208-260K memori , jadi dari perbandingan *resources* ini *Server Sent Event* lebih unggul dengan penggunaan memori yang lebih sedikit.



Gbr 18 Perbandingan Penggunaan Resources

Pada Gbr 18 bisa dilihat dalam grafik jika Penggunaan *Resources* dari *websocket* selalu tinggi atau lebih besar *Server Sent Event (SSE)*.

TABEL XV PERBANDINGAN AKHIR PERFORMA DAN FITUR SERVER SENT EVENT DENGAN WEBSOCKET

Aspek / Fitur	Websocket	Server Sent Event
Delay / Latensi	3.5-4.2 milliseconds	4.1-5.2 milliseconds
Penggunaan Resources	210 – 340K	208-260K
Auto Reconnect	Mendukung	Mendukung
Komunikasi 2 Arah	Mendukung	Tidak Mendukung
Kemudahan Implementasi	Lebih Rumit	Lebih Mudah
Format Data	Teks, Array, Biner	Hanya Teks

Dengan mempertimbangkan hasil dari penelitian bisa dilihat *Websocket* lebih unggul dalam hal *delay* atau *response time* , tetapi *server sent event* lebih baik dan menghemat penggunaan *resources*, maka penulis menambahkan beberapa parameter atau aspek yang lain yang bisa dijadikan acuan untuk mempertimbangkan penggunaan protokol komunikasi, dimana ada beberapa fitur seperti komunikasi dua arah (*full duplex*) dan fleksibilitas format data yang hanya dimiliki *websocket* dan tidak dimiliki *Server Sent Event* [6][7].

IV. KESIMPULAN

Penelitian ini bertujuan untuk membandingkan performa dari protokol komunikasi untuk integrasi dengan perangkat *hardware* yang dimana disini menggunakan *RFID* sebagai sensor pembacaan data yang nantinya akan digunakan untuk sistem *Self Checkout*. Parameter utama dalam perbandingan *Websocket* dengan *Server Sent Event* adalah Waktu *Delay* atau *response time* dan Penggunaan *Resources*, dan beberapa parameter atau aspek lain yang merupakan fitur utama yang didukung oleh masing - masing protokol komunikasi. Dengan parameter utama dan beberapa aspek lain.

Setelah implementasi sistem *Software* dan *Hardware*, proses integrasi dilanjutkan dengan menggunakan *Server Sent Event (SSE)* dan *Websocket*. *Event Listener* menggunakan serial port, agar saat data dari *sensor RFID* yang masuk akan langsung dikirim secara *Realtime* kepada *client* atau *frontend* aplikasi agar data bisa di proses. Berdasarkan uji coba yang sudah dilakukan menunjukkan *Websocket* lebih unggul daripada *Server Sent Event (SSE)* dengan *delay* waktu 3.5 - 4.2 *miliseconds* dengan *SSE* 4.1 - 5.2 *miliseconds*. Sedangkan untuk penggunaan memori *websocket* menggunakan sekitar 210 - 340K dan *SSE* menggunakan 208-260K memori. Akan tetapi dengan mempertimbangkan Keunggulan *websocket* dalam komunikasi nya yang mendukung dua arah atau *full duplex*, juga mendukung format data yang lebih banyak daripada *SSE* yang hanya mendukung *text* atau tipe data *String*. Bisa disimpulkan dalam penelitian ini protokol komunikasi untuk integrasi antara *RFID* dengan aplikasi berbasis *website*, *websocket* lebih unggul dalam *Delay* dan beberapa aspek lain dibandingkan dengan *Server Sent Event*.

REFERENSI

- [1] Anwari, V. B., Faras, F., & Samsinar. (2020). Implementasi Sistem Informasi Kasir Pada Rakab Mercon Berbasis Web. *Seminar Nasional Inovasi Teknologi*, 1–8.
- [2] Riferro Lim, R. (2021). Perancangan Sistem Kasir Layanan Mandiri Berbasis Web Pada Supermarket Bless Dengan Metode Sdlc (Vol. 1, Issue 1
- [3] Abugabah, A., Nizamuddin, N., & Abuqabbah, A. (2020). A review of challenges and barriers implementing RFID technology in the Healthcare sector. *Procedia Computer Science*, 170, 1003–1010. <https://doi.org/10.1016/j.procs.2020.03.094>
- [4] Zulham. (2019). IOT (INTERNET OF THINGS) GUNA Mendukung Era Industri 4.0.
- [5] Soewito, B., Christian, Gunawan, F. E., Diana, & Gede Putra Kusuma, I. (2019). Websocket to support real time smart home applications. *Procedia Computer Science*, 157, 560–566. <https://doi.org/10.1016/j.procs.2019.09.014>
- [6] MDN Web Docs. 2024. Websocket URL: <https://developer.mozilla.org/en-US/docs/Web/API/Server>. Diakses tanggal 07 Juni 2024.
- [7] MDN Web Docs. 2024. Server Sent Events URL: https://developer.mozilla.org/en-US/docs/Web/API/Server-sent_events. Diakses tanggal 07 Juni 2024.
- [8] A.B., A.-I. (2017). Implementasi Teknologi Websocket dalam Pengembangan Sistem Berbagi Lokasi Berbasis Web. *Jurnal Pengembangan Teknologi Informasi Dan Ilmu Komputer*, 1(9), 950–959
- [9] Dedi Jubaedi, A., & Guna Herlambang, A. (2017). RANCANG BANGUN SISTEM INFORMASI KEANGGOTAAN PENGGUNA FASILITAS LAPANGAN OLAHRAGA PADA GEDUNG OLAHRAGA MANDIRI BERBASIS RFID. 4(1).
- [10] Louis de la torre, dkk (2019). "Using Server-Sent Events for Event-Based Control in Networked Control Systems"