ANALISIS UNJUK KERJA *SINGLE PAGE WEB APPLICATION : CYBER MEDIAWALL* PERPUSTAKAAN UNILA

Mardiana [1*], Roby Syah Putra[2], Meizano Ardhi Muhammad[3]
[1,2,3] Jurusan Teknik Elektro Unila

[1*]mardiana@eng. nila.ac.id

ABSTRAK

Cyber mediawall sangat dibutuhkan untuk mendukung penyampaian suatu informasi menjadi lebih cepat, tepat dan akurat, terutama untuk penyampaian informasi yang realtime dan bersifat terus menerus. Demikian juga dengan Perpustakaan, user membutuhkan informasi mengenai content dan event yang ada di Perpustakaan baik dalam bentuk teks, animasi, video dan suara yang dapat diperoleh dengan cepat, mudah dan menarik. Cyber mediawall ini dikembangkan dengan menerapkan teknologi Single Page Web Application dan menggunakan WebSocket terutama untuk pengambilan data secara real time dan penjadwalan content. Manajemen informasi pun dilakukan dengan mengatur seluruh data yang akan ditampilkan mulai dari proses pemasukkan content, sinkronisasi seluruh content dari server pusat dengan lokal, dan penjadwalan sehingga content dapat ditampilkan secara real time atau sesuai dengan waktu tayang yang diinginkan. Untuk analisa unjuk kerja, aplikasi dibangun dengan dalam dua versi, yaitu dengan menggunakan WebSocket dan dengan Ajax JSON. Hasil pengujian pada Cyber mediawall ini membuktikan bahwa seluruh fitur sudah berjalan sesuai yang diharapkan, WebSocket memiliki kecepatan page load lebih cepat daripada AJAX. Proses yang membutuhkan waktu paling lama pada AJAX adalah proses scripting sedangkan pada Websocket adalah proses rendering.

***Kata kunci** : Cyber Mediawall, Perpustakaan, Single Page Web Application, WebSocket, AJAX*

LATAR BELAKANG

Mediawall merupakan aplikasi berbasis web yang berguna untuk menampilkan media informasi seperti berita singkat, foto, video melalui monitor seperti LCD TV, Monitor, Projector baik local, intranet (LAN & WAN) bahkan internet. Biasanya, sebuah monitor hanya dapat menampilkan satu layar saja untuk menampilkan video atau gambar pada saat bersamaan. Tetapi sistem Mediawall memungkinkan kita untuk membagi satu layar menjadi beberapa bagian yang bisa menampilkan content yang berbeda pada saat yang bersamaan. Mediawall ini banyak digunakan di perusahaan layanan jasa seperti bandara, showroom, mall, bank dan tempat publik lainnya sebagai media promosi, dan informasi layanan bagi para konsumen jasa. Selain itu juga dapat dimanfaatkan untuk pengawasan keamanan yang di kendalikan dari ruang tertentu pada perkantoran ataupun perusahaan (Levi, 2006).

Kendala dan permasalahan pada penyampaian informasi termasuk juga yang dialami di perpustakaan Unila yaitu penyebaran informasi yang biasanya hanya dilakukan melalui media seperti papan pengumuman, pamplet, leaflet, brosur, dan sebagainya. Kelemahan cara penyebaran informasi tersebut biasanya kurang mendapatkan perhatian dari pengunjung, bahkan brosur seringkali tidak dibaca isinya. Sehingga informasi menjadi tidak mengenai sasaran dan sia-sia. Kendala penyebaran informasi tersebut jika terus menerus terjadi akan memberikan dampak langsung terhadap kualitas pelayanan maupun buruknya pemahaman pengguna. Dengan menggunakan mediawall, informasi dapat ditampilkan secara langsung dan saat itu juga kepada pengguna sehingga penyebaran informasi diharapkan menjadi lebih tepat sasaran. Selain itu informasi dapat ditampilkan dalam berbagai format tampilan yang lebih menarik yang dapat dilihat maupun didengar oleh pengunjung, dibandingkan

dalam format kertas. Update informasipun menjadi lebih mudah dan cepat dilakukan oleh pengelola.

Berdasarkan permasalahan yang telah diuraikan diatas, maka sangatlah perlu adanya sistem mediawall yang berfungsi untuk penyebaran informasi di Perpustakaan Unila. Selain itu sistem tersebut harus dapat dikembangkan dengan cara yang efektif dan efisien dengan menggunakan teknologi pengembangan aplikasi berbasis web terkini. Sistem harus dapat juga memanfaatkan teknologi terkini pada pengembangan sistem mediawall terutama pada penjadwalan (scheduler) yang bersifat real time. Untuk mengetahui unjuk kerja sistem maka aplikasi perlu dibangun dengan dalam dua versi, dalam hal ini dengan menggunakan *WebSocket* dan *Ajax JSON*. Dengan demikian pada tahapan pengujian dan evaluasi pun akan lebih mudah dilakukan. Dari hasil evaluasi dan analisis dapat dibuat rekomendasi yang akan digunakan untuk memperbaiki kualitas aplikasi dan pelayanan perpustakaan secara berkelanjutan.

TINJAUAN PUSTAKA

Sistem Mediawall

Salah satu teknologi informasi yang banyak digunakan untuk menampilkan informasi di ruangan publik adalah teknologi mediawall atau yang biasa disebut dengan videowall, yang berisikan informasi umum dalam bentuk text, video, audio, gambar, animasi yang dapat ditayangkan kepada masyarakat umum. Sehingga setiap individu yang membutuhkan informasi dapat langsung memperolehnya tanpa harus mengakses sendiri sistem informasi. Sistem Mediawall dapat menampilkan data dalam format yang bervariasi, mulai dari gambar (BMP, JPG, PNG, GIF), flash video, text, html, dan sebagainya. Sistem dapat berjalan pada TV Plasma, Monitor LCD, Monitor VGA, Videowall, dan Projector. Sistem pengaturan pada Mediawall dapat mengoperasikan banyak tampilan dan diakses melalui LAN, WAN atau internet. Pengontrolan berbagai tampilan juga dapat dilakukan dari jarak jauh tanpa harus ke lokasi unit tampilan untuk berbagai sumber content yang terletak pada lokasi yang berbeda.

Evolusi Teknologi Aplikasi Berbasis Web

Pada awalnya semua halaman web adalah bersifat statis, pengguna meminta resource dan server memberikan resource tersebut. Halaman web hanyalah sebuah copy dari teks yang berbentuk elektronik. Browser digunakan untuk pertukaran hasil penelitian oleh ilmuwan dan sebagai informasi online yang digunakan oleh universitas. Teknik antar muka yang dikembangkan akhir akhir ini adalah Single Page Application(SPA). SPA merupakan suatu aplikasi berbasis web yang hanya menggunakan satu halaman saja, dengan kata lain jika pengguna beralih menu, pada URL tidak akan menunjukkan perubahan halaman. Dalam SPA, komponen CSS, image, skrip dan resources yang lain akan dimuat pada satu waktu di halaman utama dan kemudian kontennya akan dimuat secara dinamis berdasarkan permintaan pengguna (Jadhav et al, 2015). SPA dapat memberikan responsivitas yang tinggi dan kenyamanan penggunaan, sehingga tidak membingungkan pengguna (Mikowski , 2014).

Ajax adalah sebuah teknik yang digunakan untuk membuat rich application pada browser yang menggunakan JavaScript sebagai komponen utama. Teknologi yang dimanfaatkan oleh Ajax untuk memungkinkan komunikasi antara klien dan server adalah XMLHttpRequest yang pada awalnya hanya didukung oleh Internet Explorer versi 5 atau lebih, namun saat ini telah didukung oleh banyak browser yang ada. Istilah Ajax dibuat oleh Jesse James Garet dari Adaptive Path. Hal yang ditekankan oleh Ajax adalah proses pengiriman data ke server secara asynchronous dan penggunaan objek XMLHttpRequest yang pendekatan dengan cara ini belum pernah digunakan sebelumnya. Hal ini memungkinkan proses pengiriman data dari dan ke server dilakukan secara asynchronous dan perubahannya dapat langsung ditampilkan di klien tanpa me-refresh ulang halaman web (Asleson, 2006).

Websocket di HTML5 merupakan kemajuan di bidang komunikasi HTTP. WebSocket menawarkan solusi yang lebih baik, dirancang untuk diimplementasikan di web browser dan web server. Dengan teknik ini permintaan yang datang dari sisi klien cukup dilakukan sekali saat proses handshaking, selanjutnya akan membuka satu jalur komunikasi sehingga data yang ada pada sisi server dapat dikirimkan ke sisi klien. Dengan begitu klien tidak perlu melakukan permintaan yang tidak diperlukan seperti pada teknik polling. Spesifikasi dari Websocket memungkinkan saluran komunikasi dua arah single-socket untuk mengirim dan menerima informasi antara browser dan server. Saat ini Websocket di HTML5 adalah sarana terdepan untuk memfasilitasi full-duplex, pertukaran data di web secara real time. Protokol WebSocket protocol telah distandarisasi oleh Internet Engineering Task Force (IETF) pada RFC 6455 (Fette, 2011) and WebSockets (Whatwg, 2017) telah distandarisasi oleh World Wide Web Consortium (W3C).

METODE PENELITIAN

Metode penelitian yang digunakan adalah sebagai berikut:

1. Studi literatur berkaitan dengan pengembangan sistem berbasis web, real time scheduling, HTML5 WebSocket dan sistem mediawall. Identifikasi masalah dan pola solusi yang hendak dikerjakan.
2. Analisa kebutuhan sistem dan desain sistem.
3. Pengembangan sistem berbasis web menggunakan metoda pengembangan perangkat lunak *Prototype* (Pressman, 2010) dengan menerapkan teknologi Single Page Web Application.
4. Modul pengambilan data dan penjadwalan (scheduler) dibangun dengan menggunakan teknologi WebSocket, sedangkan untuk yang lain sebagian aplikasi dibuat dalam dua versi, yaitu versi dengan WebSocket dan versi Ajax. Sebagai dasar pembuatan aplikasi, user requirement untuk aplikasi dalam penelitian ini diperoleh dari pihak pengelola Perpustakaan Unila.

HASIL DAN PEMBAHASAN

Rancangan sistem *Cyber mediawall* yang dikembangkan dapat dilihat pada penelitian sebelumnya (Roby, 2015). Pada penelitian ini dilakukan dua pengujian pada *Cyber Mediawall* yaitu pengujian fitur dan pengujian unjuk kerja.

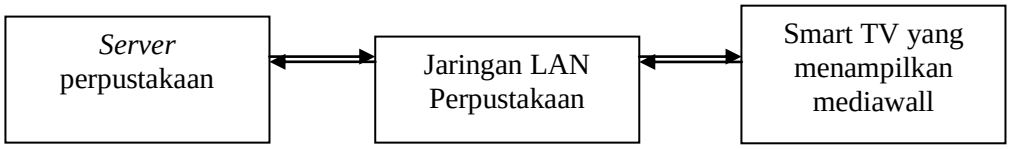
A. Pengujian Fitur

Pada tahap pengujian ini *Cyber Mediawall* ditampilkan pada TV layar datar yang ditampilkan di ruangan pusat informasi perpustakaan unila (Universitas Lampung). Aplikasi web ini diinstall pada *server* perpustakaan dan ditampilkan melalui jaringan internet yang dihubungkan pada TV layar datar tersebut.

Tabel 1. Lingkup pengujian fitur.

No	Perangkat	Spesifikasi	Kegunaan	Jumlah
1	Server	mendukung spesifikasi sebagai *server* utama.	Perangkat yang digunakan sebagai *server* pada pengujian fitur.	1
2.	Layar Televisi	Mendukung spesifikasi resolusi video dan dapat menampilkan browser.	Perangkat pengujian fitur sistem disisi client.	1
3.	*Internet access connection*	Wireless Minimum 32kbps	Jaringan yang digunakan sebagai penghubung antara sisi client dan sisi *server* pada pengujian fitur.	1

Berikut ini adalah topologi dari pengujian fitur :



Gambar 1. Topologi Fisik Pengujian Fitur Cyber Mediawall

Dengan menggunakan perlengkapan dan topologi tersebut dilakukan pengujian berdasarkan tabel sekenario pengujian fitur *Cyber Mediawall*.

Tabel 2. Tabel Skenario Pengujian Fitur *Cyber Mediawall*

No	Fungsional ID	Prosedur Pengujian	Yang diHarapkan	Hasil
1.	MW-TV-01	Melihat informasi apa yang ditampilkan	Menampilkan daftar buku terbaru	Sesuai
2.	MW-TV-02	Melihat informasi apa yang ditampilkan	Menampilkan slide gambar sebagai media promosi dan dokumentasi perpustakaan	Sesuai
3.	MW-TV-03	Melihat informasi apa yang ditampilkan	Menampilkan jadwal operasional perpustakaan	Sesuai
4.	MW-TV-04	Melihat informasi apa yang ditampilkan	Menampilkan pengumuman – pengumuman yang perlu diketahui oleh pelanggan perpustakaan baik berita seputar kampus, beasiswa, dan lainnya.	Sesuai
5.	MW-TV-05	Melihat informasi apa yang ditampilkan	Menampilkan Jam Digital	Sesuai
6.	MW-TV-06	Melihat informasi apa yang ditampilkan	Menampilkan Tanggal berisi hari, bulan, dan tahun.	Sesuai
7.	MW-TV-07	Melihat informasi apa yang ditampilkan	Menampilkan Video promosi perpustakaan	Sesuai
8.	MW-TV-08	Melihat informasi apa yang ditampilkan	Menampilkan Informasi dalam bentuk Running Text	Sesuai

Berikut tampilah hasil akhir pengujian.

Gambar 2. Hasil akhir pengujian fitur

B. Pengujian Unjuk Kerja

Aplikasi ini menggunakan Ajax JSON, dan *Websocket* sebagai *web service*nya. Sehingga pada unjuk kerja aplikasi dilakukan pengujian *page load* kecepatan dari masing – masing *web service* yang digunakan. Pengujian unjuk kerja ini dilakukan pada satu laptop menggunakan *server* apache pada XAMPP yang bersifat localhost. Spesifikasi perangkat intel celeron 1,4GHz, RAM 2GB, VGA 1GB intel HD graphics. Berikut ini adalah skenario pengujian pada *page load web service* aplikasi :

Tabel 3. Tabel skenario pengujian *web service*.

No	Fungsional ID	Prosedur Pengujian	Yang Diharapkan	Hasil
1	MW-WS-01	Melakukan test *page load* pada ajax JSON	Mengatahui kecepatan *page load*	1,303642 detik.
2	MW-WS-02	Melakukan test *page load* pada *websocket*	Mengetahui kecepatan *page load*	759,927 milidetik

Pengujian *page load* dilakukan menggunakan *google chrome* dan melakukan test *page load* dengan melihat *time-line* pada kedua *web service* dengan sampel masing-masing 10 sampel.

1. Pengujian *Page load* Ajax JSON

Dari pengujian *page load* dari ajax JSON sebanyak 10 kali pengujian menggunakan localhost dengan diperoleh hasil sebagai berikut :

Tabel 4. Tabel Hasil pengujian *page load* Ajax JSON

No	Nomor Pengujian	Loading	Scripting	Rendering	Painting	Jumlah
1.	Pengujian 1	1,12 s	780,28 ms	117,92 ms	197,34 ms	2215,54 ms
2.	Pengujian 2	377,23 ms	444,66 ms	156,00 ms	264,81 ms	1242,7 ms
3.	Pengujian 3	245,32 ms	1,02 s	104,23 ms	244,42 ms	1613,97 ms
4.	Pengujian 4	346,66 ms	502,85 ms	67,63 ms	203,76 ms	1120,9 ms
5.	Pengujian 5	304,16 ms	672,57 ms	194,46 ms	162,95 ms	1334,14 ms
6.	Pengujian 6	259,43 ms	385,93 ms	113,80 ms	221,13 ms	980.29 ms
7.	Pengujian 7	234,70 ms	436,16 ms	123,35 ms	247, 72 ms	1041,93 ms
8.	Pengujian 8	281,92 ms	473,84 ms	124,71 ms	221,60 ms	1102.07 ms
9.	Pengujian 9	252,31 ms	816,31 ms	131,53 ms	156,69 ms	1356,84 ms
10.	Pengujian 10	258,56 ms	406,33 ms	102,24 ms	260,91 ms	1028,04 ms
Jumlah total		3680.29 ms	5938,93 ms	1235,87 ms	2181,33 ms	13036,42 ms
Rata-rata		368,029 ms	593,893 ms	123,587 ms	218,133 ms	1303,642 ms

Dari hasil pengujian *page load* ajax JSON diperoleh hasil rata – rata total *page load* adalah sebesar 1303,642 milidetik atau 1,303642 detik.

2. Pengujian *Page load* Html5 *Websocket*

Pada pengujian *page load* dari html5 *websocket* diperoleh hasil sebagai berikut :

Dari hasil pengujian tersebut dibuat tabel hasil pengujian *page load* untuk mengetahui kecepatan rata – rata dari 10 kali sampel pengujian, adapun tebel hasil pengujian Html5 *Websocket* sebagai berikut :

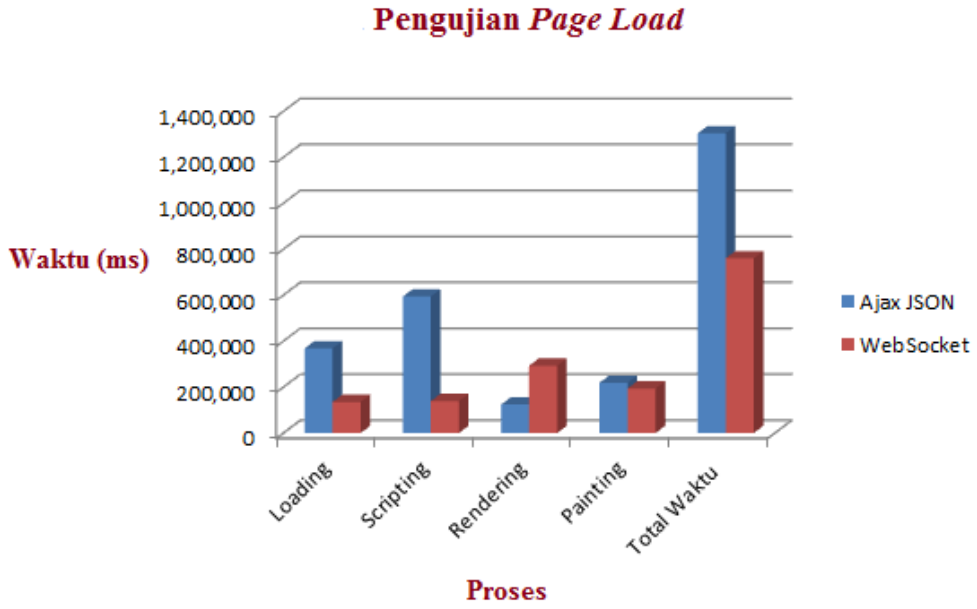
Tabel 5. Tabel Hasil pengujian *page load* Html5 *Websocket*

No	Nomor Pengujian	Loading	Scripting	Rendering	Painting	Jumlah
1.	Pengujian 1	259,89 ms	164,86 ms	251,65 ms	151,02 ms	827,42 ms
2.	Pengujian 2	82,82 ms	138,37 ms	321,75 ms	233,17 ms	776,11 ms
3.	Pengujian 3	80,38 ms	134,17 ms	356,14 ms	271, 16 ms	841,85 ms
4.	Pengujian 4	85,84 ms	118,62 ms	345,29 ms	197,40 ms	747,15 ms
5.	Pengujian 5	127,91 ms	132,33 ms	278,40 ms	147,47 ms	686.11 ms
6.	Pengujian 6	93,78 ms	128,25 ms	264,85 ms	286,71 ms	773.59 ms
7.	Pengujian 7	208,99 ms	153,18 ms	343,91 ms	207,10 ms	913,18 ms
8.	Pengujian 8	213,20 ms	153,44 ms	362,87 ms	186,39 ms	915,9 ms
9.	Pengujian 9	95,14 ms	140,41 ms	184,03 ms	98,98 ms	518,56 ms
10.	Pengujian 10	100,60 ms	126,12 ms	217,34 ms	155,34 ms	599,4 ms
Jumlah total		1348,55 ms	1389,75 ms	2926.23 ms	1934,74 ms	7599,27 ms
Rata-rata		134,855 ms	138,975 ms	292,623 ms	193,474 ms	759,927 ms

Dari hasil pengujian *page load* Html5 *Websocket* diperoleh hasil rata – rata total *page load* adalah sebesar 759,927 milidetik.

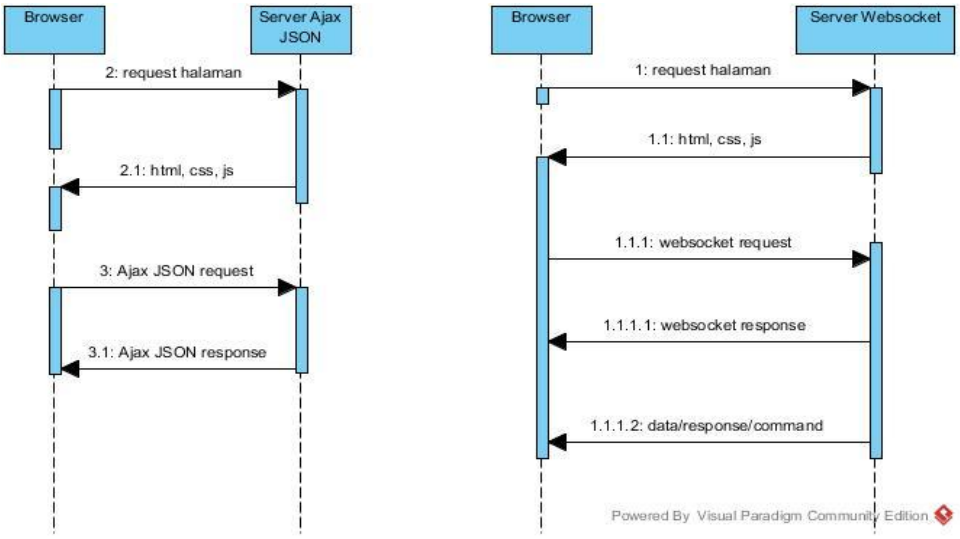
3. Evaluasi Unjuk kerja

Berdasarkan hasil pengujian dapat dilihat bahwa jika dibandingkan Ajax JSON, *web service* Html5 *Websocket* memiliki kinerja *page load* yang lebih cepat berdasarkan pengujian page load pada masing-masing *web service*. Hal ini terlihat dari perbedaan poin yaitu Ajax JSON menghasilkan rata-rata jumlah pengujian sebesar 1,303642 detik dan Html5 *Websocket* menghasilkan rata-rata jumlah pengujian sebesar 0,759927 detik. Sehingga dapat dilihat pada aplikasi *websocket* memiliki kecepatan *page load* cepat dibandingkan Ajax JSON.



Gambar 3. Grafik Rata-Rata Pengujian *Page Load*

Untuk sistem kerja dari ajax JSON dan html5 *websocket* pada *Cyber Mediawall* dapat dilihat pada diagram sequance berikut ini :



Gambar 4. diagram *sequence* perbandingan ajax JSON dan html5 *websocket*

Pada ajax JSON sistem *Cyber Mediawall* ini ketika browser dari client melakukan *request* ke *server* maka *server* akan mengirim form html, css, dan js dalam tamplate *mediawall*.html ke browser client, kemudian *server* akan mulai melakukan *load* data dari *database* ke tampilan *Cyber Mediawall* menggunakan JSON.php sebagai penarik datanya dan merubah dalam bentuk JSON, kemudian ajax load JSON pada tampilan *mediawall*.html akan mem*parsing* data yang diambil oleh JSON.php dan menampilkannya dalam bentuk informasi aslinya sesuai dengan yang terdapat pada *database*. JSON.php akan terus menarik informasi dari *database*, sehingga setiap ada informasi baru yang diinputkan oeh *web service* maka JSON.php akan mulai me*load* data, lalu kemudian ajax load JSON yang bertugas mem*parsing* yang terdapat pada *mediawall*.html akan melakukan *request* dan *load* sehingga informasi baru tersebut dapat ditampilkan pada *mediawall*.html disisi client. Pada ajax *Cyber Mediawall* shake hand antara sisi browser dan *server* terjadi *request* berkali-kali agar tetap menjaga informasi tetap berada pada posisi *realtime*.

Sedangkan pada sisi *websocket Cyber Mediawall* ini ketika browser dari client melakukan *request* ke *server* maka *server* akan mengirim form html, css, dan js dalam tamplate *mediawall*.htm ke browser client. Setelah browser melakukan request dan menerima tamplate *mediawall*.html maka hubungan antara browser dan *server* akan trus terjaga *realtime* hingga ada close request dari sisi browser, setelah itu java jdbc akan menarik informasi dari *database* dan menampilkannya ke dalam servlet kemudian servlet ini akan ditampilkan ke dalam salah konten pada tamplate *mediawall*.html dan java jdbc akan terus melakukan load data untuk menampilkannya ke dalam servlet. *Websocket* pada konten *Cyber Mediawall* akan menjaga koneksi antara browser dan *server* agar tetap terkoneksi, sehingga tanpa browser melakukan *request* pun data pada konten akan terus ter*load* secara *realtime*. Namun pada penggunaannya *websocket* memerlukan *server* khusus agar dapat beroperasi dengan baik. Pada *Cyber Mediawall* ini *server* khusus yang digunakan berupa Glassfish 4.1 yang terdapat pada EE7 java netbeans dan menggunakan JDBC pada java netbeans untuk menarik data dari *database* dan menampilkannya melalui servlet.

Pada hasil pengujian dapat dilihat bahwa pada Ajax JSON proses *scripting* membutuhkan waktu paling lama dibandingkan dengan proses lainnya yaitu selama 593,893 milidetik perbedaannya waktunya pun sangat jauh jika dibandingkan dengan proses lainnya yaitu *rendering* yang selama 123,587 milidetik, loading selama 368,029 milidetik dan *painting* selama 218,133 milidetik. Hal ini terjadi karena pada Ajax JSON dilakukan proses *scripting* berkali-kali untuk mendapat *respon* dari server, lalu setelah itu melakukan *rendering* untuk ditampilkan ke dalam mediawall.

Sedangkan untuk html5 *websocket* hasil pengujian menunjukan proses yang membutuhkan waktu paling lama adalah *rendering* dengan proses yang membutuhkan waktu selama 292,623 milidetik,

namun jika dibandingkan dengan proses lainnya perbandingan waktu prosesnya tidak terlalu jauh. Hal ini dapat dilihat jika dibandingkan dengan angka berikut yaitu proses *scripting* membutuhkan waktu 138,975 milidetik, *loading* membutuhkan waktu 134,855 milidetik, dan *painting* membutuhkan waktu 193,474 milidetik. Hal ini disebabkan karena pada html5 websocket hanya dilakukan proses *scripting* hanya sekali dan seterusnya *server websocket* akan mengirimkan *respon* terus menerus tanpa perlu melakukan *scripting* kembali dan menyebabkan prosses *rendering* terjadi berulang-ulang kali dan membutuhkan waktu paling lama.

KESIMPULAN

1. Pada pengujian fitur, seluruh prosedur yang diuji pada scenario pengujian menunjukkan hasil yang sesuai
2. Pada aplikasi *websocket* memiliki kecepatan *page load* lebih cepat dibandingkan Ajax JSON. Test *page load* html5 *websocket* hasil rata-rata pegujiannya adalah sebesar 0,759927 detik dan pada Ajax JSON diperoleh hasil rata-rata pengujiannya sebesar 1,303642 detik.
3. Pada Ajax JSON proses scripting membutuhkan waktu yang paling lama yaitu selama 593,893 milidetik, sedangkan untuk html5 *websocket* proses rendering membutuhkan waktu paling lama adalah yaitu selama 292,623 milidetik.

DAFTAR PUSTAKA

Asleson, Ryan. and Schutta, Nathaniel T., "Foundation Of Ajax", Appress, Springer-Verlag, New York, 2006.

Fette, I. and A. Melnikov, A., "The WebSocket Protocol," RFC 6455 (Proposed Standard), Internet Engineering Task Force, Dec. 2011. [Online]. Available: http://www.ietf.org/rfc/rfc6455.txt

Jadhav, M. A., Sawant, B. R. & Deshmukh, A. (2015). Single Page Application using AngularJS, International Journal of Computer Science and Information Technologies, 6 (3): 2876.

Levi, Albert, Proceeding of Computer and information sciences - ISCIS 2006 21th International Symposium, SpringerLink, Istanbul, Turkey, 2006.

Mikowski M. S. & Powell, J. C. (2014). Single Page Web Application. Shelter Island: Manning Publications Company.

Pressman, Roger, S, "Software Engineering : A practitioner's Approach", Seventh Edition. New York: McGraw-Hill ., 2010.

Roby S.P., Mardiana, Meizano A.M., 2015, "Rancang Bangun Sistem Informasi *Cyber Mediawall* Perpustakaan Universitas Lampung", Prosiding SATEK VI ISBN 978-602-0860-02-2

Whatwg Community, "Websockets", Nov. 2017. [Online]. Available: https://html.spec.whatwg.org/multipage/web-sockets.html#network