IMPLEMENTASI WEB PUSH NOTIFICATION PADA SISTEM INFORMASI MANAJEMEN ARSIP MENGGUNAKAN PUSHJS

by Id936 Jtiik

Submission date: 17-Oct-2018 10:09AM (UTC+0700)
Submission ID: 1021408308
File name: 936-2795-1-RV.docx (549.06K)
Word count: 3120
Character count: 20331

Jurnal Teknologi Informasi dan Ilmu Komputer (JTIIK) DOI: 10.25126/jtiik...
Vol. x, No. x, April 2014, hlm. x-x p-ISSN: 2355-7699
Akreditasi KEMRISTEKDIKTI, No 48a/E/KPT/2017 e-ISSN: 2528-6579

IMPLEMENTASI WEB PUSH NOTIFICATION PADA SISTEM INFORMASI MANAJEMEN ARSIP MENGGUNAKAN PUSHJS

(Naskah masuk: dd mmm yyyy, diterima untuk diterbitkan: dd mmm yyyy)

Abstrak

Teknologi terus menerus berkembang, berbagai jenis teknologi terus bermunculan seperti sistem informasi manajemen arsip. Namun para pekerja kadang melakukan pekerjaan lain di komputer sehingga arsip tidak terkontrol. Penerapan *Web Push Notification* dapat menampilkan pemberitahuan berbasis *website*. *Web Push Notification* merupakan mekanisme pemberitahuan menggunakan *Javascript* pada *web browser*. Fitur ini tersedia dalam *Push API HTML5* dengan menggunakan *Push Service* atau *Messaging server* yang mengirim pemberitahuan ke *web browser* yang telah berlangganan tanpa membuka *website* sehingga dapat melakukan *broadcast message* dan *Notification API HTML5* tidak memerlukan *Push Service* atau *Messaging server* tetapi harus membuka *website*, tetapi belum didukung semua *web browser* sehingga pada makalah ini dibahas Implementasi *Web Push Notification* pada sistem informasi manajemen arsip menggunakan *PushJS*. *PushJS* merupakan gabungan dari *Push API HTML5* dengan *Notification API HTML5* dengan kompatibilitas yang telah dikembangkan sehingga menggunakan *script* yang sama ketika berbeda *web browser*. Teknologi pemberitahuan yang cocok untuk sistem informasi manajemen arsip berbasis *web* yaitu *Notification API HTML5* karena tidak akan mengirim pemberitahuan yang sama ke semua pengguna. Namun tidak ada proses di belakang layar sehingga tidak akan dijalankan secara otomatis, masalah tersebut diatasi dengan menggunakan *AJAX* dengan mengambil *JSON* kemudian dijalankan berulang-ulang pada *web browser* dan meminimalisir bentrokan antara *script web push notification* di *multi tab window* atau *window web browser* diatasi menggunakan *localStorage* dari WebStorage API HTML5.

Kata kunci: *ajax, json, notification api html5, pushjs, webstorage api html5*

Abstract

Technology is constantly evolving, various types of technology keep emerging like archive information management system. But the workers sometimes do other work on the computer so the archives are not controlled. Implementing Web Push Notification can display website-based notifications. Web Push Notification is a notification mechanism using Javascript in a web browser. This feature is available in Push API HTML5 by using Push Service or Messaging server that sends notification to web browser that has been subscribed without opening website so that it can do broadcast message and Notification API HTML5 does not require Push Service or Messaging server but must open website but not supported all web browsers so in this paper discussed Implementation of Web Push Notification on the archive management information system using PushJS. PushJS is a composite of Push API HTML5 with HTML5 Notification API with developed compatibility so that it uses the same script when different web browsers. The notification technology suitable for a web-based archive management information system is HTML5 Notification API because it will not send the same notification to all users. But there is no process behind the scenes so it will not run automatically, the problem is solved by using AJAX by taking JSON and then running repeatedly on the web browser and minimize clash between push notification web scripts in multi-tab window or web browser window resolved using localStorage from the HTML5 web storage API.

Keywords: *ajax, json, notification api html5, pushjs, webstorage api html5*

1. PENDAHULUAN

Pada umumnya sebuah aplikasi berbasis web tidak mempunyai fasilitas notifikasi atau pemberitahuan terbaru kepada pengguna, sehingga proses penyampaian informasi terutama informasi penting tidak diketahui secara langsung. Hal tersebut tentunya menjadi permasalahan, terutama pada sistem informasi manajemen arsip yang memerlukan penanganan yang cepat terhadap informasi arsip yang baru masuk. Jika tidak segera ditangani arsip terus bertumpuk dan semakin banyak, sementara pengguna aplikasi atau petugas tidak mungkin terus standby memperhatikan aplikasi manajemen arsip.

Dari permasalahan tersebut, perlu adanya pengembangan pada aplikasi dengan menerapkan sistem notifikasi secara otomatis tanpa harus membuka atau memperhatikan aplikasi terus menerus. Teknik-teknik dalam penyajian data secara

realtime yang telah dilakukan pada aplikasi web diantaranya dapat menggunakan *Asynchronous JavaScript and XML (AJAX)* dan *Web Socket* (Husen, Rahmatulloh, & Sulastri, 2018).

Bentuk pemberitahuan saat ini yang sedang tren yaitu pemberitahuan pada *header* halaman *web* (Bahtiar & Mulwinda, 2016) dengan menggunakan AJAX atau sejenisnya namun pemberitahuan ini hanya dapat terlihat ketika membuka halaman web secara langsung. Pemberitahuan pada perangkat Android menggunakan Google *Cloud Messaging* (GCM) dengan menerapkan *Calendar Provider*. *Calendar Provider merupakan* jenis data yang dapat digunakan pada aplikasi kalender di perangkat Android (Setiawan, Kristianto, & Fredicia, 2015) namun jenis pemberitahuan ini hanya berjalan pada perangkat Android dan GCM jika diterapkan pada web hanya dapat digunakan pada Push API dengan jenis pesan yang dikirim secara masal apabila telah berlangganan. Pemberitahuan pada perangkat Android dengan konsep *alarm* jadwal perkuliahan (Ramadhan & Utomo, 2014). Pemberitahuan monitoring menggunakan *SMS Gateway* (Fiade, Maula, & Suseno, 2013) atau sistem pemilihan umum menggunakan SMS Gateway (Satrio, Herlambang, Setyawan, & Arwani, 2018) namun pemberitahuannya menggunakan biaya *SMS*. Adapun pemberitahuan menggunakan *email* (Asri, Hamzah, & Sholeh, 2014).

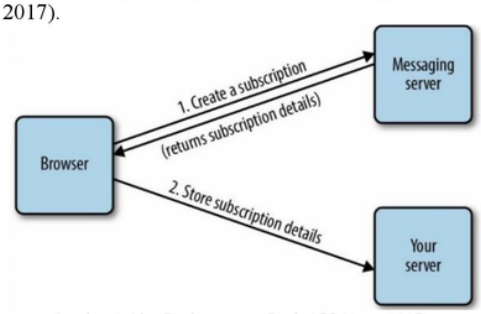
Sementara pada penelitian Rosidin menyatakan bahwa *Web Push Notification* merupakan teknologi pemberitahuan yang harus diterapkan pada *web* masa kini (Rosidin, 2017). Sehingga fokus penelitian ini, untuk mendukung teknologi informasi dalam pengiriman informasi pemberitahuan yang terkontrol dibutuhkan implementasi *Web Push Notification* pada Sistem Informasi Manajemen Arsip menggunakan PushJS, karena fitur teknologi *Push API HTML5* dan *Notification API HTML5* belum didukung oleh semua *web browser* sehingga menggunakan PushJS sebagai *library* pada lintas platform. (Nickerson, 2017)

2. *WEB PUSH NOTIFICATION*

Web Push Notification merupakan pemberitahuan yang dapat dikirim ke pengguna melalui *web desktop* dan *web* seluler. Pemberitahuan ini merupakan pesan gaya lansiran yang tampil pada sudut kanan atas atau bawah layar *desktop*, tergantung pada sistem operasi, atau muncul di perangkat seluler dengan cara yang hampir identik dengan *Push Notification* yang dikirim dari aplikasi. *Web Push Notification* dikirimkan di *desktop* atau layar seluler pengguna kapan pun ketika *web browser* dijalankan meskipun pengguna membuka halaman *web* atau tidak. (Airship, Urban, 2018)

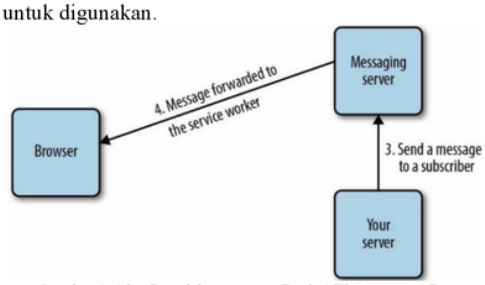
2.1 *Push API HTML5*

Push API HTML5 merupakan jenis notifikasi pada *web browser* yang menggunakan *Push Service* atau *Messaging server* sebagai tempat *broadcast messaging* dengan syarat *web browser* telah berlangganan (Ater, 2017). *Push API HTML5* memungkinkan pengiriman pesan *push* ke *webapp* melalui layanan *push*. *Server* dapat mengirim pesan *push* kapan saja, bahkan ketika *webapp* atau *user agent* tidak aktif. Layanan push memastikan pengiriman yang andal dan efisien kepada agen pengguna. Pesan *push* dikirim ke *Service Worker* yang berjalan di *webapp*, yang dapat menggunakan informasi dalam pesan untuk memperbarui status lokal atau menampilkan pemberitahuan kepada pengguna. Push API memungkinkan *webapp* untuk berkomunikasi dengan *user agent* secara asinkron. Ini memungkinkan server aplikasi untuk memberikan informasi kapan tanpa membuka *webapp*. (W3C, 2017).



Gambar 1 Alur Berlangganan Push API (Ater, 2017)

Gambar 1 merupakan alur dalam penggunaan pertama *Push API* dengan memanggil *subscriber()*. Pusat server pengiriman pesan akan menyimpan rincian langganan baru ini dan mengembalikan ke halaman, selanjutnya halaman dapat mengirim rincian berlangganan ke *server* dimana data disimpan untuk digunakan.



Gambar 2 Alur Pengiriman pesan Push API (Ater, 2017)

Gambar 2 merupakan alur pengiriman *Push API* dengan memanfaatkan rincian langganan dengan mengirim pesan ke *message server* kemudian meneruskan ke *web browser* pengguna, pada saat itu *ServiceWorker* bekerja apabila telah terdaftar di *web browser*.

2.2 *Notification API HTML5*

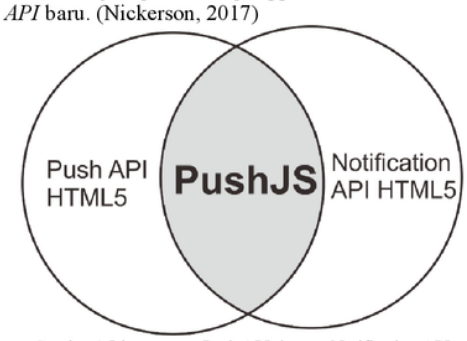
Notification API HTML5 memungkinkan halaman web atau pekerja layanan (*service worker*) untuk membuat dan mengontrol tampilan pemberitahuan sistem. Pemberitahuan ini

ditampilkan di luar web browser (di antarmuka perangkat) dan dengan demikian ada di luar konteks jendela atau *tab web browser* tunggal. Karena tidak bergantung pada jendela atau tab peramban apa pun, mereka dapat dibuat bahkan setelah pengguna meninggalkan situs Anda. Sebelum Anda dapat menampilkan pemberitahuan kepada pengguna, Anda harus terlebih dahulu meminta izin pengguna. Keseluruhan prosesnya relatif sederhana dan lugas (Ater, 2017).

Notification API untuk menampilkan pemberitahuan untuk memperingatkan pengguna di luar konteks halaman web. Pemberitahuan ini mengacu pada menampilkan pemberitahuan pada "desktop". ini dirancang agar kompatibel dengan platform pemberitahuan yang ada sebanyak mungkin, tetapi juga menjadi platform-independen. Karena platform umum tidak menyediakan fungsi yang sama, spek ini akan menunjukkan peristiwa apa yang dijamin dan mana yang tidak. Khususnya, pemberitahuan yang ditentukan di sini hanya dapat berisi konten teks dan ikon. (W3C, 2015)

2.3 *PushJS*

PushJS adalah salah satu cara untuk membangun dan menjalankan pemberitahuan *desktop* Javascript. Tambahan yang cukup baru untuk spesifikasi resmi, *Notification API* memungkinkan browser *modern* seperti Chrome, Safari, Firefox, dan IE 9+ untuk mendorong pemberitahuan ke desktop pengguna. PushJS bertindak sebagai solusi lintas peramban untuk *API* ini, *fallback* menggunakan penerapan yang lebih lama jika peramban pengguna tidak mendukung *API* baru. (Nickerson, 2017)



Gambar 3 Irisan antara Push API dengan Notification API menjadi PushJS

Gambar 3 merupakan perumpamaan bahwa *PushJS* memiliki kemampuan dari *Push API HTML5* dan *Notification API HTML5*. Fungsi utama notifikasi menggunakan *Notification API HTML5* dan menggunakan *serviceWorker* untuk menggunakan teknologi *Push API HTML5* sehingga tidak langung menggunakan fasilitas dari *Web Browser*.

2.4 *WebStorage API HTML5*

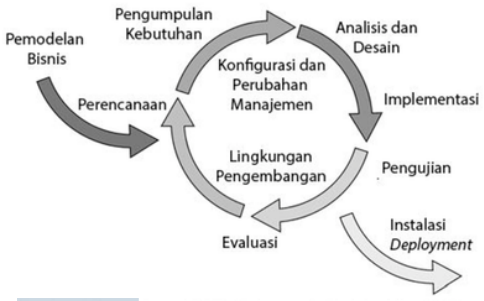
Web Storage API HTML5 atau disebut DOMStorage merupakan sebuah *API* yang membuatnya mudah untuk menyimpan data di seluruh *web*. Sebelum ada *Web Storage API*, *remote server* diperlukan untuk menyimpan data apa pun yang bertahan dengan mengirimnya bolak-balik dari klien ke *server*. Dengan munculnya *Web Storage API*, pengembang sekarang dapat menyimpan data secara langsung di sisi klien pada *web browser* untuk akses berulang di seluruh permintaan atau untuk diambil secara lama setelah Anda menutup *browser* sepenuhnya, sehingga mengurangi lalu lintas jaringan. *Web Storage* berbeda dari *cookie* dari bagaimana cara menyimpan dan mengambil data. Dengan menggunakan *API* sederhana ini, pengembang dapat menyimpan nilai dalam objek JavaScript yang mudah diambil yang bertahan di seluruh beban halaman. Dengan menggunakan *sessionStorage* atau *localStorage*, pengembang dapat memilih untuk membiarkan nilai-nilai tersebut bertahan baik di seluruh beban halaman dalam satu jendela atau *tab* atau di seluruh *browser* saat di *restart*. Data yang tersimpan tidak ditransmisikan di seluruh jaringan dan mudah diakses pada kunjungan kembali ke halaman. (Ater, 2017)

2.5 PHP *Data Objects*

PDO (*PHP Data Objects*) merupakan fitur dari PHP yang telah ada sejak versi 5. *PDO* menjadi pilihan karena mendukung berbagai jenis database, dimana untuk untuk memanggil seluruh fungsi setiap database tidak memperlukan perubahan *script* cara fungsi koneksi ke *database*. PDO mendukung *SQL injection* dengan menggunakan PDO *BindParam statement parameter* atau masukan dari pengguna secara otomatis telah di*filter*. *PDO* menggunakan *data-access abstaction layer* untuk berhubungan dengan *database* (Abdul, 2008).

3. METODE PENELITIAN

Metodologi pengembangan perangkat lunak pada penelitian ini menggunakan *RUP (Rational Unified Process)* mulai dari tahapan permulaan (*inception*), perencanaan (*elaboration*), kontruksi (*contruction*) dan transisi (*transition*). Proses pengulangan atau iteratif pada *RUP* secara global dapat dilihat pada Gambar 4.

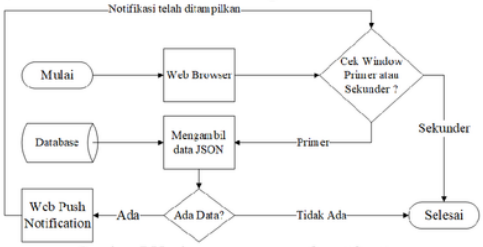


Gambar 4 Proses Iterasi RUP (Sukamto & Shalahuddin, 2013)

4. HASIL DAN PEMBAHASAN

4.1. Tahap *Inception* (Permulaan)

Proses bisnis yang akan diterapkan pada Sistem Informasi Manajemen Arsip hanya pada bagian Petugas Perpus dan Admin setelah *login*. Sistem yang diimplementasikan berupa pemberitahuan menggunakan *PushJS*, *JQuery* dan *AJAX* dengan bentuk pertukaran data menggunakan *JSON*. Pertukaran data *JSON* dapat diamankan menggunakan *JSON Web Token (JWT)* (Rahmatulloh, Sulastri, & Nugroho, 2018).



Gambar 7 Usulan rancangan *push notification*

Gambar 7 merupakan rancangan usulan *push notification* menggunakan *PushJS* berbasis Javascript. Setelah melakukan *login, web browser* akan menjalankan *script* dengan mengecek status tab window atau *window web browser* primer atau sekunder, dilihat dari waktu pembuatan *tab window* atau *window* dalam satuan *milisecond* atau biasa disebut *timestamp* disimpan dalam *localStorage* menggunakan *Web Storage API HTML5*. Apabila status *tab window* atau *window* primer maka akan menjalankan *script* pengambilan data berbentuk *JSON* menggunakan *AJAX* jika ada maka akan ditampilkan sedangkan jika tidak ada atau sudah semua data maka akan kembali lagi ke proses pengecekan status *tab window* atau *window web browser* yang digunakan.

Adapun aktor-aktor yang berhubungan dengan sistem ini yaitu: Petugas SKPD (Satuan Kerja Pemerintah Daerah), Petugas DISPERPUSKA dan Pengunjung. Kebutuhan fungsional sistem informasi manajemen arsip adalah Manajemen Arsip, Manajemen Instansi, Manajemen Pengguna, Manajemen Ruangan, Manajemen Rak, Manajemen Box, Manajemen Surat, ScannerBox dan Pencarian Arsip. Kebutuhan non fungsional sistem informasi manajemen arsip adalah sistem notifikasi pemberitahuan yang bisa memberikan informasi atau pengingat ketika tanpa harus memperhatikan aplikasi atau sedang mengerjakan aktifitas lain di komputer.

4.2. Tahap *Elaboration* (Perencanaan)

Pada tahap kedua ini proses perencanaan dimulai dari menganalisa, identifikasi dan evaluasi proses bisnis yang sedang berjalan. Sehingga akan didapatkan solusi alternatif yang mengarah kepada perbaikan dan pengembangan yang lebih baik serta sesuai dengan kebutuhan.

Untuk mengetahui sistem yang sedang berjalan dan untuk mempelajari sistem yang ada, diperlukan suatu penggambaran aliran-aliran informasi dari bagian-bagian yang terkait baik dari dalam maupun dari luar sistem. Adapun aliran informasi yang sedang berjalan pada sistem manajemen arsip adalah sebagai berikut :

1. SKPD/Kecamatan menyerahkan Arsip untuk di dititipkan di depo arsip.
2. DIPERPUSKA menerima Arsip dari SKPD/kecamatan yang akan ditipkan di depo arsip.
3. DIPERPUSKA menyeleksi Arsip yang akan dititipkan
4. DIPERPUSKA membuat Jadwal Resensi Arsip (JRA).

4.3. Tahap *Construction* (Kontruksi)

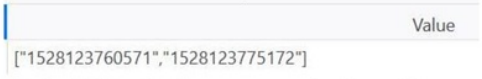
Implementasi sistem menggunakan *PushJS* ke dalam Sistem Informasi Manajemen Arsip digabungkan dengan *Javascript*, *JQuery*, *AJAX*, *WebStorage API HTML5*. Konsep pemberitahuan yang digunakan seperti *running text* pada *channel* televisi yang memberikan informasi terus menerus sampai informasi tersebut di ubah oleh operator. Konsep ini pula menjadi inspirasi untuk diimplementasikan pada Sistem Informasi Manajemen Arsip yang dapat memberikan pemberitahuan terus menerus sampai ditindak lanjuti oleh pengguna. Pemberitahuan yang dimunculkan pada *Web Push Notification* seperti Arsip yang belum diatur JRA dan Arsip yang sudah melewati masa Inaktif harus ditindak sesuai jenis Arsip.

Konsep teknis yang digunakan *Javascript* mengecek ketersediaan *web* dibuka di *tab window* yang masa saja, kemudian script *Web Push Notification* akan berjalan pada *tab window* yang terakhir dibuka berdasarkan waktu pembuatan *tab window* menggunakan *localStorage* dan *sessionStorage* dari *Web Storage API HTML5*. Setelah itu *script Web Push Notification* yang menggunakan *PushJS* akan mengambil data berbentuk *JSON* dari *database* dengan menggunakan *AJAX* secara *realtime* dan dihitung jumlah data yang akan di munculkan pada pemberitahuan dengan nilai *timeout* atau waktu tampil yang telah ditentukan. Apabila ada perubahan dalam pemberitahuan akan di eksekusi pada saat element data yang terakhir telah dimunculkan Gambar 8 merupakan tampilan dari antarmuka *Web Push Notification*.

Gambar 8 Tampilan Antarmuka *Web Push Notification*

4.4. Tahap *Transition* (Transisi)

Sebelum proses transisi, pengujian merupakan hal terpenting yang bertujuan untuk menemukan kesalahan-kesalahan atau kekurangan-kekurangan pada perangkat lunak yang akan diuji. Metode pengujian menggunakan *black box* yang berfokus pada persyaratan fungsional dari sistem yang dibangun. Pada proses pengecekan menggunakan *localStorage* menghasilkan *timestamp* dalam format *milisecond* atau milidetik kemudian dibandingkan waktu yang paling terbaru membuat *window* atau *tab window* pada *web browser* dengan menghasilkan data berbentuk JSON seperti pada Gambar 9 bernilai ["1528123760571","1528123775172"] yang berarti 06-04-2018 14:49:20 dan 06-04-2018 14:49:35 pada zona *Universal Time Coordinated (UTC)* sedangkan Asia/Jakarta ditambah 7 jam lebih cepat. Jadi, *tab window* atau *window* yang akan menjalankan *web push notification* yang dibuat pada 1528123775172 atau 06-04-2018 14:49:35 pada zona *UTC*.



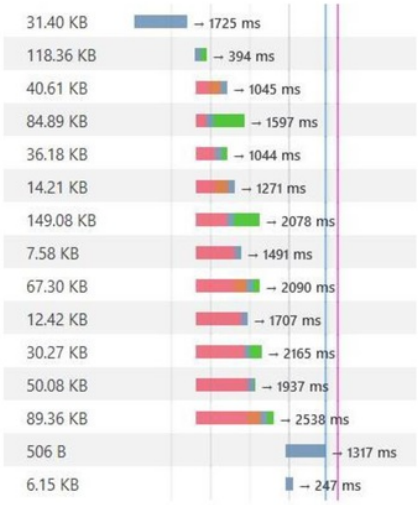
Gambar 9 Isi data *localStorage* menggunakan *Inspect Element* pada *Web Browser*

Pengujian selanjutnya fungsional sistem menggunakan skenario uji yang dapat dilihat pada Tabel 1.

Tabel 1 Bagian Uji Fungsional Sistem

Nama Bagian	Skenario Uji	Hasil yang diharapkan	Kesimpulan
Web Push Notification	Belum melakukan Login	Tidak menampilkan pemberitahuan	[√] Berhasil [ ] Tidak Berhasil
	Setelah Pengguna Login (Petugas Perpus)	Menampilkan pemberitahuan	[√] Berhasil [ ] Tidak Berhasil
	Setelah Pengguna Login (Petugas SKPD)	Tidak Menampilkan pemberitahuan	[√] Berhasil [ ] Tidak Berhasil
	Membuka *web* di *tab window* yang lain	Tidak terjadi bentrok *script* yang berjalan	[√] Berhasil [ ] Tidak Berhasil

Dalam memproses *script Web Push Notification*, *web browser* membutuhkan beberapa detik untuk membuka halaman setelah selesai dengan pengecekan validasi *window* selama 1000 *ms* (*milisecond*) atau 1 detik dan menampilkan notifikasi selama 5000 ms (*milisecond*) atau 5 detik. Jadi, jika ada 10 notifikasi maka dibutuhkan waktu 5x10 = 50 detik untuk menampilkan semua pemberitahuan. Proses tersebut belum termasuk lama proses membuka halaman *web* dan data *JSON*.



Gambar 10 Tampilan Visual Lama Waktu Proses

Gambar 10 menunjukan hasil analisa menggunakan *Inspect Element* pada *web browser* Mozilla Firefox diperoleh 1725 milidetik (ms) atau 1,7 detik.

Gambar 11 Proses Pengulangan pengambilan *JSON*

Berdasarkan Gambar 11 diperoleh 10 sampel kemudian dimasukan kedalam Tabel 2 dengan dihitung rata-rata lama waktu pemrosesan dengan karakteristik 9500 data arsip pada *database* dengan hasil *compressing* menjadi 506 *Bytes*. Hasil rata-rata diperoleh 2161,2 milidetik yang dapat dilihat pada Tabel 2.

Tabel 2. Uji Waktu Pemrosesan Data JSON

No	milidetik (*ms*)
1	1773
2	2427
3	2118
4	2275
5	2070
6	2094
7	2069
8	2264
9	2094
10	2428
Jumlah	21612
Rata-rata	2161,2

Gambar 12 *Unit Testing Web Push Notification* menggunakan QUnit

Gambar 12 merupakan hasil *unit testing Web Push Notification* menggunakan QUnit menunjukan fungsi *webpush()* pada HTML berjalan sesuai dengan harapan.

5. KESIMPULAN

Penerapan *Web Push Notification* dalam pemberitahuan pada Sistem Informasi Manajemen Arsip dapat membantu dalam proses penyampaian informasi sehingga lebih interaktif dan *realtime*. Sementara penggunaan *Web Storage API HTML5* berfungsi untuk mengecek *status tab window* pada *web browser* yang sama, sehingga *Javascript* tidak bentrok saat dijalankan pada *multi tab window*.

Sebagai bahan penelitian selanjutnya teknologi *Web Push Notification* dapat dikembangkan lebih lanjut dengan fitur *input* atau *multiple choice* sehingga pengguna dapat memasukan data langsung atau memilih opsi yang ada pada notifikasi tanpa harus membuka halaman *web*. PushJS dapat dikembangkan menggunakan *serviceWorker* dengan teknologi *Push API HTML5* dengan memanfaatkan *Firebase Cloud Messaging*.

DAFTAR PUSTAKA

Abdul, K. (2008). *Dasar Pemrograman Web Dinamis Menggunakan PHP* (1 ed.). Yogyakarta: Penerbit Andi.

Airship, Urban. (2018). *Web Push Notifications Explained | Urban Airship*. Dipetik 07 15, 2018, dari https://www.urbanairship.com/web-push-notifications-explained

Asri, N. F., Hamzah, A., & Sholeh, M. (2014). NAGIOS UNTUK MONITORING SERVER DENGAN PENGIRIMAN NOTIFIKASI GANGGUAN SERVER MENGGUNAKAN EMAIL DAN SMS GATEWAY (STUDI KASUS : PT. GAMATECHNO INDONESIA - YOGYAKARTA). *Jurnal JARKOM, 1*(2), 151-161.

Ater, T. (2017). *Building Progressive Web Apps* (1 ed.). Sebastopol: O'Reilly Media.

Bahtiar, M. R., & Mulwinda, A. (2016). Pengembangan Fitur Notifikasi pada Website Application Comic Strip rupi.co Menggunakan Metode Agile. *Jurnal Teknik Elektro, 8*(1), 25-30.

Fiade, A., Maula, A. A., & Suseno, H. B. (2013). Aplikasi Monitoring Jaringan Berbasis Mobile Web dengan Sistem Notifikasi Berbasis SMS Gateway. *JURNAL TEKNIK INFORMATIKA, 6*(2).

Husen, H., Rahmatulloh, A., & Sulastri, H. (2018, 4 1). IMPLEMENTASI KOMUNIKASI FULL DUPLEX MENGGUNAKAN WEBSOCKET PADA SISTEM INFORMASI PENGELOLAAN GGARAN UNIVERSITAS ABC. *Simetris: Jurnal Teknik Mesin, Elektro dan Ilmu Komputer, 9*(1), 597-606.

Nickerson, T. (2017). *Installing | Push v1.0*. Dipetik 07 15, 2018, dari https://pushjs.org/docs/introduction

Rahmatulloh, A., Sulastri, H., & Nugroho, R. (2018). Keamanan RESTful Web Service Menggunakan JSON Web Token (JWT) HMAC SHA-512. *Jurnal Nasional Teknik Elektro dan Teknologi Informasi (JNTETI)*.

Ramadhan, T., & Utomo, V. G. (2014). Rancang Bangun Aplikasi Mobile Untuk Notifikasi Jadwal Kuliah Berbasis Android (Studi Kasus: Stmik Provisi Semarang). *Jurnal Teknologi Informasi dan Komunikasi, 5*(2), 47-55.

Rosidin. (2017). *Manfaat Menggunakan Push Notification Untuk Website - JURNAL ROSID*. Dipetik 07 15, 2018, dari https://jurnal.rosid.net/manfaat-menggunakan-push-notification-untuk-website/

Satrio, W. A., Herlambang, A. D., Setyawan, R. A., & Arwani, I. (2018). ANCANG BANGUN SISTEM E2OV (ELECTRONIC – ELECTION OBSERVATION AND VOTING) MENGGUNAKAN SMS. *Jurnal Teknologi Informasi dan Ilmu Komputer, 5*(1), 1-6.

Setiawan, ., Kristianto, E., & Fredicia. (2015). Implementasi Push Notification pada Informasi Perkuliahan dan Kegiatan Mahasiswa Berbasis Android. *Jurnal Teknik dan Ilmu Komputer, 4*(14), 211-219.

Sukamto, R. A., & Shalahuddin, M. (2013). *Rekayasa Perangkat Lunak (Terstruktur dan Berorientasi Objek)*. Bandung: BI-Obses.

W3C. (2015). *Notification API : W3C Recommendation 22 October 2015*. Dipetik 07 15, 2018, dari https://www.w3.org/TR/notifications/

W3C. (2017). *Push API : W3C Working Draft 15 December 2017*. Dipetik 07 15, 2018, dari https://www.w3.org/TR/push-api/

IMPLEMENTASI WEB PUSH NOTIFICATION PADA SISTEM INFORMASI MANAJEMEN ARSIP MENGGUNAKAN PUSHJS

ORIGINALITY REPORT

12%	11%	2%	6%
SIMILARITY INDEX	INTERNET SOURCES	PUBLICATIONS	STUDENT PAPERS

PRIMARY SOURCES

1	Submitted to Universitas Brawijaya Student Paper	3%
2	elib.unikom.ac.id Internet Source	1%
3	media.neliti.com Internet Source	1%
4	widuri.raharja.info Internet Source	1%
5	id.123dok.com Internet Source	1%
6	journal.akprind.ac.id Internet Source	1%
7	ejnteti.jteti.ugm.ac.id Internet Source	1%
8	docplayer.info Internet Source	1%
9	doaj.org	

	Internet Source	<1%
10	id.portalgaruda.org Internet Source	<1%
11	www.journal.unipdu.ac.id Internet Source	<1%
12	www.repository.uinjkt.ac.id Internet Source	<1%
13	jagocoding.com Internet Source	<1%
14	Zakky Zamrudi, Teguh Wicaksono. "Social Commerce Adoption in SME's", JEMA: Jurnal Ilmiah Bidang Akuntansi dan Manajemen, 2018 Publication	<1%
15	eprints.mdp.ac.id Internet Source	<1%
16	docobook.com Internet Source	<1%

Exclude quotes	Off	Exclude matches	Off
Exclude bibliography	Off		