Prototipe Sinkronisasi Penyimpanan Dokumen Secara *Realtime* Menggunakan *Websocket* Pada *Cloud Storage*

Muhammad Mustafa[1], Anwar[2], Indrawati[3*]

Jurusan Teknologi Informasi dan Komputer Politeknik Negeri Lhokseumawe
Jln. B.Aceh Medan Km.280 Buketrata 24301 INDONESIA

[1]mm982603@gmail.com,
[2]anwarsy@pnl.ac.id,
[3]indrawati@pnl.ac.id

Abstrak— **Perkembangan teknologi saat ini sudah berkembang dengan sangat pesat, khususnya pada penggunaan teknologi komputasi yang semakin meningkat. Seiring dengan perkembangan zaman, teknologi komputasi telah mencapai kemudahan dan kenyamanan yang luar biasa dalam melakukan kegiatan, diantaranya adalah penggunaan internet. Hampir semua aktifitas internet memerlukan tempat penyimpanan, keterbatasan penyimpanan data sering dijumpai ketika akan menyimpan data-data pada komputer atau laptop. Dengan keterbatasan yang ada maka diperlukan tempat penyimpanan *file* yang perlu dipindahkan ke database internet seperti pada website. Namun untuk meningkatkan kenyamanan dan kemudahan pengaksesan data di berbagai perangkat yang berbeda diperlukan sistem yang secara otomatis tersinkronisasi secara *realtime*. Realtime adalah dimana ketika ada perubahan data, maka saat itu juga *website* di *browser* juga ada perubahan atau setidaknya muncul notifikasi. Dengan demikian dapat diterapkan websocket sebagai solusi untuk sinkronisasi data secara realtime, websocket dirancang untuk diterapkan di *browser web* dan server web. Hasil implementasi dan pengujian yang dilakukan menunjukkan website penyimpanan dokumen secara realtime dapat dilakukan menggunakan metode *websocket, websocket* dapat menerima lebih banyak *request* dibandingkan website tanpa *websocket* yaitu 8000 *request* saat websocket berjalan dalam 5 detik pengujian. Persentase keberhasilan *websocket* saat pengiriman data mencapai 100%. Hasil pengukuran *QoS* didapatkan nilai rata-rata delay file ukuran 124 KB sebesar 0,2479 detik , file ukuran 505 *KB* sebesar 0,2874 detik dan *file* ukuran 1,32 MB sebesar 0,8231 detik. Nilai rata-rata *jitter file* ukuran 124 *KB* sebesar 0,1108 detik , file ukuran 505 *KB* sebesar 0,1252 detik dan *file* ukuran 1,32 *MB* sebesar 0,1137 detik dan dari keseluruhan data yang diuji *packet loss* yang didapatkan sebesar 0%.**
Kata kunci— *Website* Penyimpanan, *Websocket*, *Realtime*, *Cloud Storage*.

Abstract— **The development of technology is currently growing very rapidly, especially in the increasing use of computing technology. Along with the times, computing technology has achieved extraordinary convenience and comfort in carrying out activities, including the use of the internet. Almost all internet activities require storage space, data storage limitations are often encountered when storing data on a computer or laptop. With the existing limitations, it is necessary to store files that need to be moved to an internet database such as on a website. However, to increase the convenience and ease of accessing data on various different devices, a system that is automatically synchronized in real time is needed. Realtime is where when there is a change in data, at that time the website in the browser also changes or at least a notification appears. Thus, websocket can be applied as a solution for realtime data synchronization, websocket is designed to be implemented in web browsers and web servers. The results of the implementation and testing carried out show that real-time document storage websites can be done using the websocket method, websocket can receive more requests than websites without websocket, namely 8000 requests when websocket runs in 5 seconds of testing. The percentage of websocket success when sending data reaches 100%. The QoS measurement results obtained the average delay value of 124 KB file size of 0.2479 seconds, 505 KB file size of 0.2874 seconds and 1.32 MB file size of 0.8231 seconds. The average jitter value for a 124 KB file is 0.1108 seconds, a file size of 505 KB is 0.1252 seconds and a file size of 1.32 MB is 0.1137 seconds and from the overall data tested, packetloss is obtained at 0%.**
Keywords— Website Storage, Websocket, Realtime, Cloud Storage.

I. PENDAHULUAN

Perkembangan teknologi khususnya pada teknologi komputasi semakin meningkat. Hampir semua aktifitas internet memerlukan tempat penyimpanan, Namun keterbatasan penyimpanan data sering jumpai ketika akan menyimpan data-data pada komputer atau laptop. Dengan keterbatasan yang ada maka diperlukan tempat penyimpanan *file* yang perlu dipindahkan ke *database* internet seperti pada *website*.

Website adalah kumpulan halaman *web* yang saling terkoneksi dan memiliki data informasi yang saling terkait. *Website* terdiri dari halaman dan sekumpulan halaman yang disebut *homepage*. *Website* dibuat untuk dapat diakses secara luas melalui sebuah aplikasi peramban menggunakan *URL* (*Uniform Resource Locator*) [1].

Penggunaan *websocket* sudah menjadi standar baru yang dipergunakan untuk melakukan komunikasi *realtime* di situs web dan aplikasi *mobile*. Ini dikarenakan *Websocket* merupakan suatu protokol yang dipergunakan untuk komunikasi dua arah. Penggunaan *websocket* tidak hanya digunakan untuk implementasi web *browser* dan web server, tetapi dapat digunakan oleh aplikasi klien atau server [2].

Dengan adanya teknologi penyimpanan di awan yang dikenal dengan *cloud computing*. *Cloud computing* merupakan pemanfaatan teknologi komputer (komputasi) dalam suatu jaringan dengan pengembangan berbasis internet (awan) yang mempunyai fungsi untuk menjalankan program atau aplikasi melalui komputer yang terkoneksi internet. Namun untuk meningkatkan kenyamanan dan kemudahan pengaksesan data di berbagai perangkat yang berbeda diperlukan sistem yang secara otomatis tersinkronisasi secara *realtime*. Dengan demikian dapat diterapkan *websocket* sebagai solusi untuk sinkronisasi data secara *realtime*.

Penelitian ini berhubungan erat dengan beberapa penelitian yang telah dilakukan sebelumnya. Salah satu penelitian dengan judul "Pemanfaatan *Private Cloud storage* Sebagai Media Penyimpanan Data *E-Learning* Pada Lembaga Pendidikan", peneltian ini bertujuan agar tidak menghambat kinerja performa *e-learning* pada lembaga pendidikan tersebut. Pengembangan dilakukan dengan metode *incremental* dan penerapan *cloud* pada objek penelitian ini dilakukan menggunakan satu Server dan tiga *Client* [3].

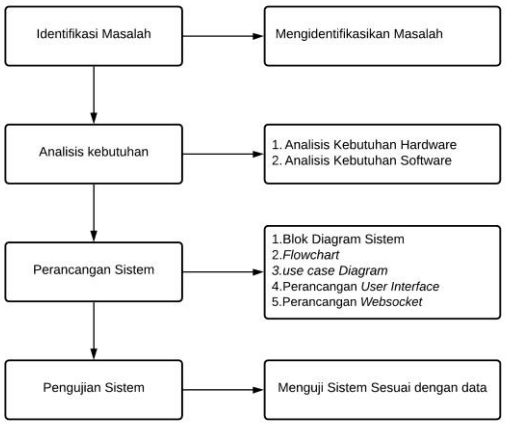
Pada penelitian lain yang berjudul "*Websocket* untuk optimasi kecepatan data transfer pada *real time chatting*", penelitian ini bertujuan untuk berkirim pesan secara *realtime* menggunakan teks melalui jaringan *internal* dengan memanfaatkan teknologi *websocket*. Pengujian dilakukan pada aplikasi *chatting* dengan menggunakan Mozilla Firefox selama 30 menit dengan perbandingan antara aplikasi *chatting websocket* dan *ajax* secara bersamaan [4].

Dalam penelitian lain yang berjudul "Penerapan *Google Drive* Sebagai Media Penyimpanan Bahan Perkuliahan Dalam Mendukung Aplikasi *Mobile App*", penelitian ini bertujuan untuk mencari solusi alternatif untuk menekan biaya di server tanpa mengurangi kualitas bahan ajar yang digunakan dan mengimplementasikan bagaimana bahan ajar tersebut dapat diterapkan pada aplikasi *smartphone* [5].

Pada penelitian lain yang berjudul "Pengembangan *Push Notification* Menggunakan *Websocket*", penelitian ini bertujuan untuk mengembangkan mekanisme pengiriman *push notification* menggunakan protokol *websocket* dengan server yang telah diimplementasikan *broker*, sehingga diharapkan pengiriman notifikasi lebih cepat. Notifikasi yang dikirimkan berupa judul berita dan tautan berita dan akan dilakukan dalam ruang lingkup jaringan lokal [6].

II. METODOLOGI PENELITIAN

Penelitian ini dilakukan melalui beberapa tahapan sehingga membentuk alur yang sistematis untuk mencapai tujuan dari penelitian. Tahapan-tahapan dalam proses penelitian dapat dilihat pada Gambar 1.



Gambar 1. Tahapan Penelitian

A. Identifikasi Masalah

Pada tahap ini dilakukan penelusuran terhadap berbagai macam literatur yang relevan dan terpercaya seperti buku, jurnal ilmiah, dan referensi-referensi lainnya. Literatur yang digunakan dititik beratkan pada komunikasi *realtime websocket* yang dirancang untuk diimplementasikan di *server web* dan *browser web* dan perhitungan QoS (*Quality of Service*), serta literatur dari penelitian yang sejenis.

B. Analisis Kebutuhan

Analisis kebutuhan dilakukan untuk memperoleh informasi kebutuhan dalam pengembangan sistem dan gambaran dari *website* yang akan dirancang. Perangkat keras yang dibutuhkan pada penelitian ini yaitu sebagai berikut :

1. Satu unit *server* berupa *cloud platform* untuk penyimpanan data dari *website*.
2. Dua unit PC/laptop untuk melakukan konfigurasi server, pembuatan *website* serta untuk pengujian *website*.

Perangkat lunak yang dibutuhkan pada penelitian ini antara lain sebagai berikut :

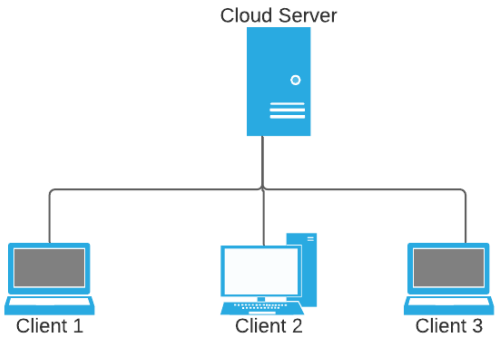
1. Aplikasi text editor untuk melakukan pemrograman pada website.
2. Aplikasi Putty untuk melakukan remote terhadap server.
3. Aplikasi FileZilla untuk upload file ke server.
4. Aplikasi Wireshark sebagai tool untuk menguji website.

C. Perancangan Sistem

Perancangan sistem dibuat untuk menjelaskan gambaran mengenai sistem yang akan dibuat. Dalam penelitian "prototipe sinkronisasi penyimpanan dokumen secara *realtime* menggunakan pada *cloud storage*" dilakukan perancangan yang meliputi perancangan blok diagram sistem, *flowchat*, perancangan *use case* diagram, perancangan *user interface* dan Perancangan *Websocket*.

D. Block Diagram Sistem

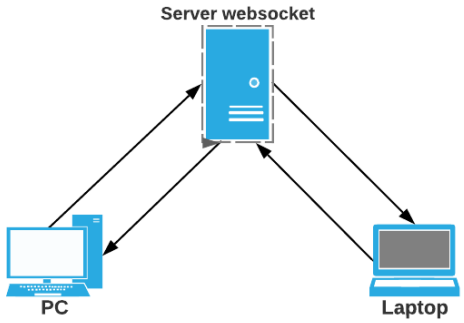
Berikut merupakan perancangan diagram blok sistem dapat dilihat pada Gambar 2.



Gambar 2 blok diagram *website* Penyimpanan

Sinkronisasi *website* penyimpanan dokumen yang diimplementasikan pada *cloud storage* menggunakan metode *websocket* terdiri atas beberapa komponen yang *cloud* server sebagai media menghubungkan antara komputer dan jaringan yang berbasis *internet* dan sebagai media penyimpanan dan pengelolaan data, komputer dan laptop sebagai *client* yang meminta *request* berupa data ke *server*. Secara singkat cara kerja prototipe sinkronisasi penyimpanan dokumen secara *realtime* menggunakan *websocket* pada *cloud storage* yaitu *client* satu melakukan *upload* data ke *server*, pada *client* dua dan *client* tiga data akan langsung diterima tanpa perlu melakukan *reload page*.

Blok diagram perancangan *websocket* yang dibangun pada penelitian ini dapat dilihat pada Gambar 3 berikut ini.

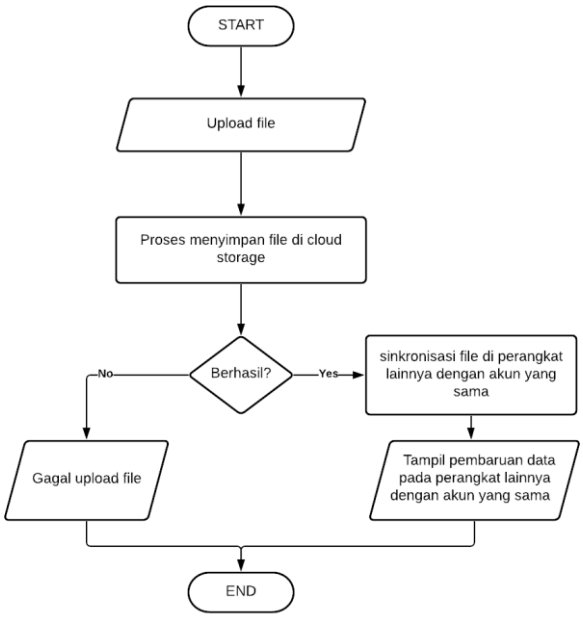


Gambar 3. Perancangan *Websocket*

Pada perancangan sistem *website* penyimpanan menggunakan teknologi *websocket*, terdapat server yang telah di integrasikan dengan *webscoket* cara kerjanya saat komputer/laptop yang mengakses *website* yang telah dimasukkan *library* dari *websocket* maka terjadi sinkronisasi secara *realtime* dari *website* tersebut. Proses sinkronisasi terjadi pada *user* yang mengakses *website* yang telah di integrasikan dengan *websocket*.

E. *Flowchart*

Adapun *Flowchart* perancangan sistem *Interface User* dapat dilihat pada Gambar 3 berikut ini.

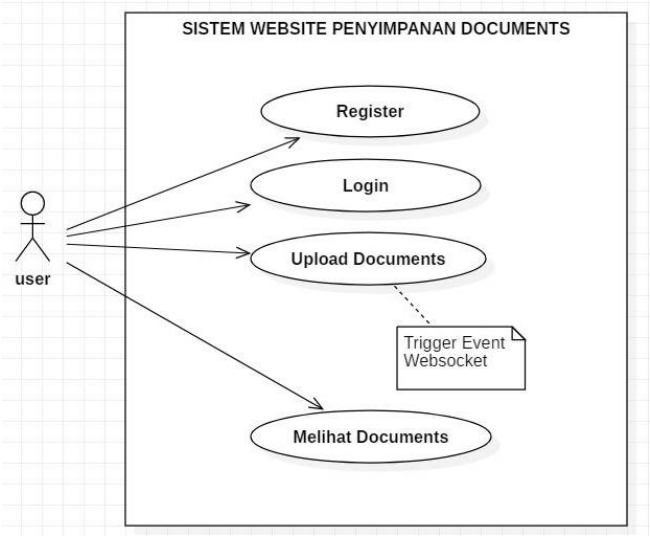


Gambar 4. *Flowchart* cara kerja dari *website* penyimpanan

Sistem *interface* menampilkan tahapan proses dari *website*, Adapun cara kerja dari *website* sinkronisasi penyimpanan dokumen secara *realtime* menggunakan *websocket* pada *cloud storage* yaitu *client* satu melakukan *upload* data ke *server*, pada server terjadi proses penyimpanan data. jika data berhasil diupload maka *websocket* akan melakukan sinkronisasi data pada perangkat/*user* lainnya dan langsung menampilkan hasil dari pembaharuan data yang tersinkronisasi secara *realtime*.

F. *Use Case* Diagram

Adapun *use case* diagram dari prototipe sinkronisasi penyimpanan dokumen secara *realtime* menggunakan *websocket* pada *cloud storage* dapat dilihat pada gambar 4 berikut.



Gambar 5. *Use Case* Diagram *Website* Penyimpanan

Gambar 5 menunjukkan *actor/user* dapat melakukan registrasi akun serta *login* dan *user* juga dapat mengelola *documents* yang ingin di *upload* saat proses *upload* data terjadi proses *websocket* supaya data tersinkronisasi serta dapat melihat seluruh *document* yang di *upload*.

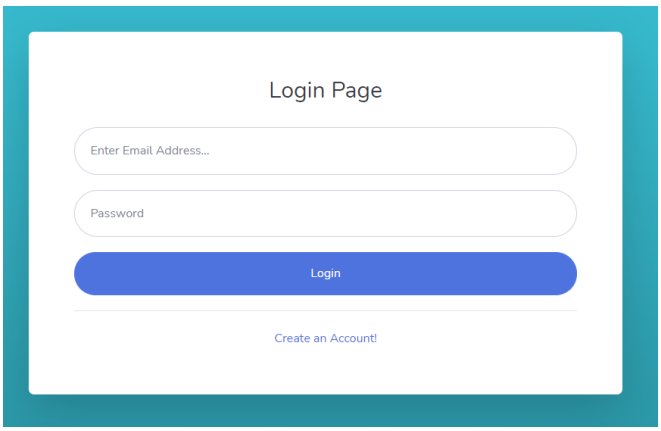
III. HASIL DAN PEMBAHASAN

A. *User Interface*

Sistem *website* sinkronisasi penyimpanan dokumen secara *realtime* menggunakan *websocket* pada *cloud storage* yang dibuat terdapat u*ser interface* sebagai visual dari *website*. *Website* ini memiliki beberapa halaman seperti halaman *login*/masuk, *Dashboard* , *My Profile*, *Edit Profile*, *Change Password*, dan *Group*.

1. Tampilan Halaman Masuk

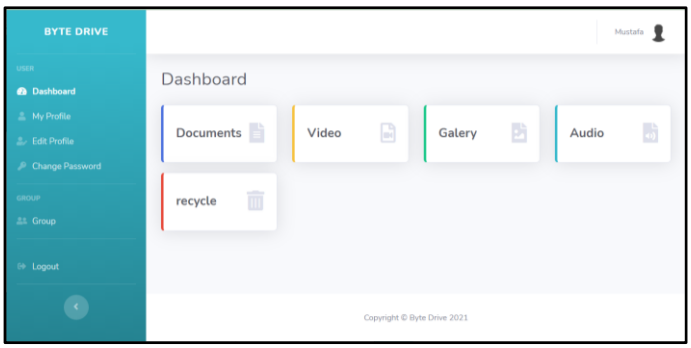
Pada halaman ini pengguna melakukan proses autentikasi masuk pada aplikasi *website*. Untuk melakukan proses autentikasi, pengguna diminta memasukkan *Email* dan *password* untuk dapat masuk pada halaman utama *website*. Tampilan Halaman Masuk dapat dilihat pada Gambar 6.



Gambar 6. Tampilan Halaman *Login*

2. Tampilan Halaman *Dashboard*

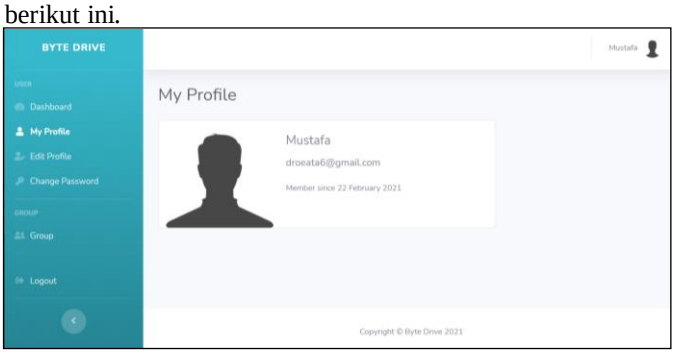
Pada halaman *dashboard* menampilkan menu yang digunakan untuk menyimpan beberapa data berupa *documen*, *video*, *galeri*, *audio* dan *reycle*/tempat sampah. Tampilan menu *dashboard* dapat dilihat pada Gambar 7.



Gambar 7. Tampilan Halaman *Dashboard*

3. Tampilan Halaman *Profile*

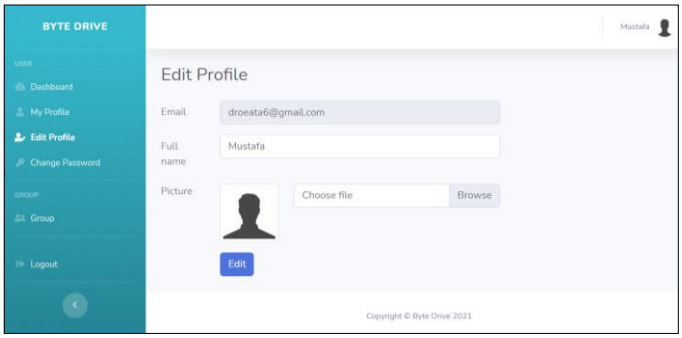
Halaman *profile* menampilkan beberapa informasi dari pengguna di antaranya nama, email dan tanggal pembuatan akun.Tampilan halaman *profile* dapat dilihat pada Gambar 8 berikut ini.



Gambar 8. Tampilan Hasil *Profile*

4. Tampilan Halaman *Edit Profile*

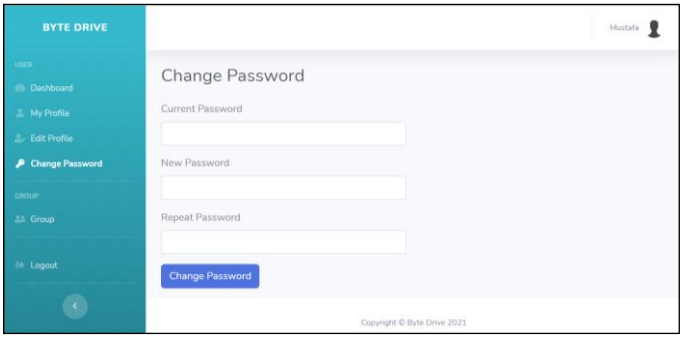
Pada halaman *edit profile* pengguna dapat mengubah data diri yaitu nama dari pengguna. Tampilan halaman *edit profile user* dapat dilihat pada Gambar 9 berikut.



Gambar 9. Tampilan Halaman *Edit Profile*

5. Tampilan Halaman *Change Password*

Halaman *Change Password* berfungsi untuk pengguna saat ingin mengubah *password* dari akun. Penggunaanya pengguna harus memasukkan *password* utama terlebih dahulu lalu memasukkan kata sandi baru. Tampilan halaman *Change Password* dapat dilihat pada Gambar 10 berikut ini.

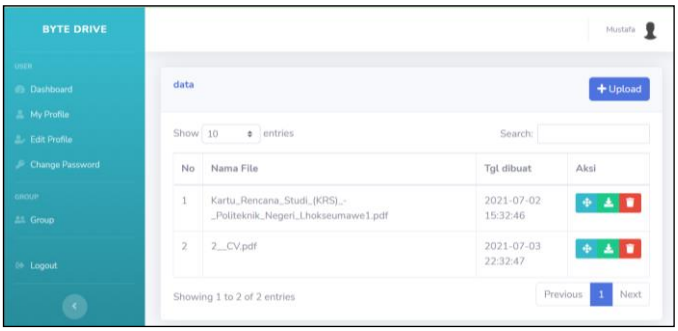


Gambar 10. Tampilan Halaman *Change Password*

6. Tampilan Halaman *Group*

Halaman *group* merupakan fitur untuk pengguna agar dapat melakukan *sharing* data dengan pengguna lainnya, saat beberapa pengguna *join* pada halaman ini pengguna dapat melakukan *sharing* data dalam i *group* . Pada halaman ini pengguna dapat menambahkan, *join group* dan

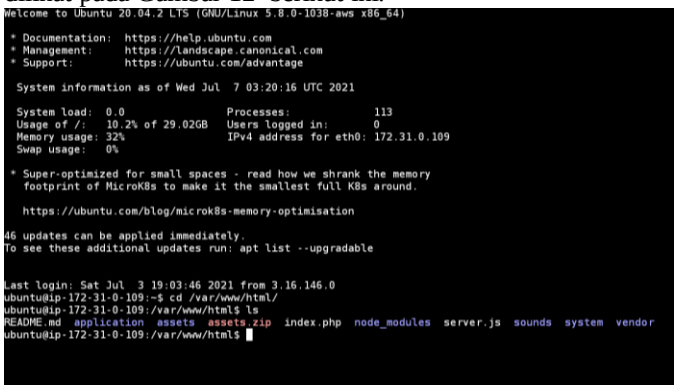
mengelola *group* . Tampilan halaman *group* dapat dilihat pada Gambar 11 berikut.



Gambar 11. Tampilan Halaman *Group*

7. Tampilan Halaman *Server*

Halaman *Server* merupakan fitur untuk admin agar dapat mengelola data, menyimpan *file* dan *database* yang dibutuhkan untuk *website*. Tampilan halaman *server* dapat dilihat pada Gambar 12 berikut ini.

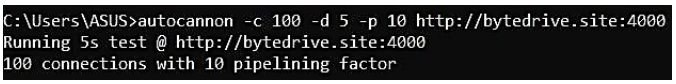


Gambar 12. Tampilan Halaman Server

B. Pengujian

1. Pengujian *Websocket*

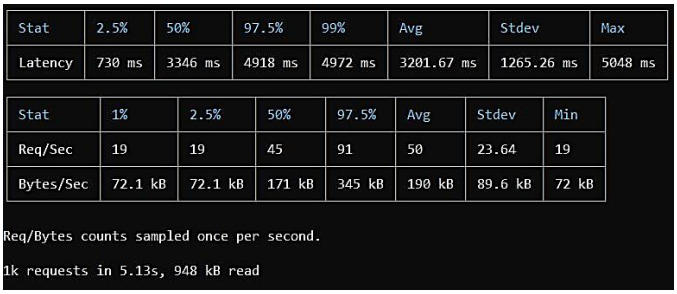
Pengujian *stress testing* menggunakan autocannon dengan menggunakan perintah dapat dilihat pada Gambar 13.



Gambar 13. Perintah *Stress Testing* Menggunakan *Autocannon*

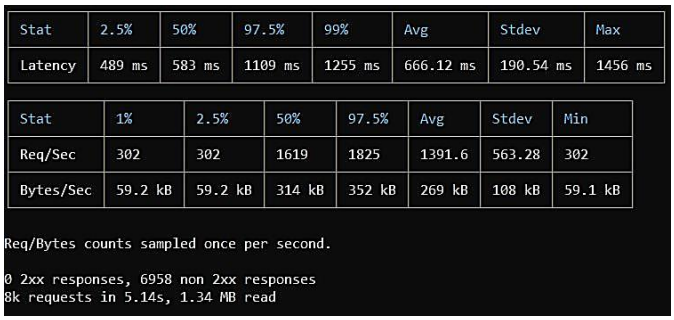
Dari perintah di atas dapat dijelaskan bahwa autocannon mensimulasikan (-c 100) 100 koneksi selama (-d 5) 5 detik dan (-p 10) 10 *multiple requests* secara bersamaan. Dilakukan 2 kali pengujian menggunakan *autocannon*, pengujian pertama dilakukan saat *websocket* tidak berjalan dan pengujian kedua dilakukan saat websocket dijalankan pada *port* 4000.

Gambar 14 dan 15 merupakan hasil pengujian *websocket* yang dijalankan menggunakan perintah pada Gambar 13.



Gambar 14. Hasil Pengujian Tanpa *Websocket*

Pada pengujian saat *websocket* tidak berjalan didapatkan hasil yaitu rata-rata *latency* 3201 ms, dan rata-rata *requests* per detik 50 *request*/detik dan rata-rata paket yang di unduh sebesar 190 KB/s.



Gambar 15. Hasil Pengujian Menggunakan W*ebsocket*

Pada pengujian saat *websocket* berjalan didapatkan hasil yaitu rata-rata *latency* 600 *ms*, dan rata-rata *requests* per detik 1391 *request*/detik dan rata-rata paket yang di unduh sebesar 269 *KB*/s. Dari 2 pengujian di atas dapat dilihat bahwa penggunaan *websocket* dapat merespon lebih banyak *request* yaitu 1000 *request* saat *websocke*t tidak berjalan dan 8000 *request* saat *websocket* berjalan dalam 5 detik.

2. Pengujian *Upload File*

Pada *website* penyimpanan dilakukan pengujian ukuran data yang di *upload* ke *server* dan didapatkan hasil pengujian seperti pada Tabel 1 berikut.

TABEL I
HASIL PENGUJIAN UKURAN UPLOAD FILE

Tipe Data	Ukuran Data	Waktu Upload	Keterangan
Galeri	1,6 Mb	3,01 s	Berhasil
Galeri	3,6 Mb	3,32 s	Berhasil
Galeri	5 Mb	5,18 s	Berhasil
Audio	10 Mb	16,14 s	Berhasil
Audio	14 MB	18,09 s	Berhasil
Audio	20 Mb	20,80 s	Berhasil
Document	30 Mb	22,25 s	Berhasil
Document	36 Mb	24,98 s	Berhasil
Document	45 Mb	27,43 s	Berhasil
Video	58 Mb	33,10 s	Berhasil
Video	80 Mb	49,77 s	Berhasil
Video	94 Mb	1,5 m	Berhasil
Galeri	1,6 Mb	3,01 s	Berhasil
Galeri	3,6 Mb	3,32 s	Berhasil
Galeri	5 Mb	5,18 s	Berhasil
Audio	10 Mb	16,14 s	Berhasil
Audio	14 MB	18,09 s	Berhasil
Audio	20 Mb	20,80 s	Berhasil
Document	30 Mb	22,25 s	Berhasil
Document	36 Mb	24,98 s	Berhasil

Dari hasil pengujian didapatkan hasil pengujian *upload file* ke *server* yaitu data yang diuji diambil tiga sampel data dari masing masing fitur dan di dapatkan total keberhasilan *upload file* pada *website* penyimpanan yaitu 100%.

3. Pengujian QoS (*Quality of Services*)

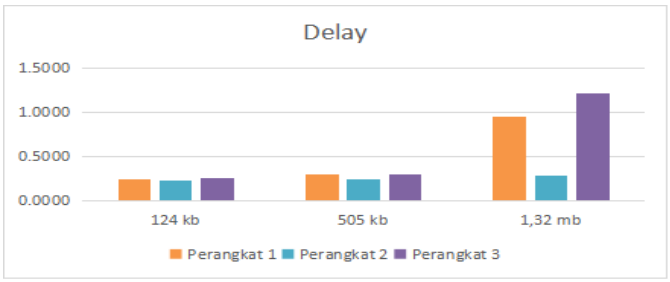
Pengujian dalam penelitian ini mencakup pengujian waktu pengiriman data (*delay*) dari *client* ke *server*, pengujian *jitter* (variasi *delay*). Hasil Pengujian *delay* dapat dilihat pada Tabel 2 berikut ini.

TABEL II
HASIL PENGUJIAN *DELAY* DENGAN *PACKET SIZE* YANG BERBEDA

Perangkat	***Delay* Dengan *Packet Size* -**		
	124 KB (s)	505 KB (s)	1,32 MB (s)
Perangkat 1	0,2477	0,3104	0,9610
Perangkat 2	0,2337	0,2499	0,2919
Perangkat 3	0,2623	0,3018	1,2164
Rata-rata	0,2479	0,2874	0,8231

Dari pengujian yang telah semakin besar data yang dikirim semakin besar waktu *delay* yang diperoleh ,pada paket data 124 *KB* selisih pengiriman paket tidak terlalu besar dikarenakan paket yang dikirim kecil ,namun pada paket 1,32 *MB* terdapat selisih data yang berbeda.

Gambar 16 merupakan grafik perbandingan *delay* pengujian pengiriman data dengan *packet size* yang berbeda.



Gambar 16. Grafik Perbandingan *Delay*

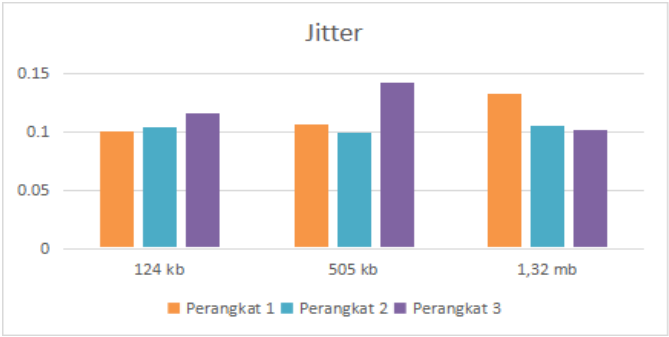
Jitter diperoleh dari variasi *delay* (perbedaan selang waktu) antar paket yang terjadi pada jaringan yang disebabkan oleh panjangnya antrian pada saat pengolahan data yang terjadi pada jaringan. Hasil pengujian *jitter* dapat dilihat pada Tabel 3.

TABEL III
HASIL PENGUJIAN *JITTER PACKET SIZE* YANG BERBEDA

Perangkat	***Jitter* Dengan *Packet Size* -**		
	124 KB (s)	505 KB (s)	1,32 MB (s)
Perangkat 1	0,1009	0,1065	0,1337
Perangkat 2	0,1050	0,0995	0,1051
Perangkat 3	0,1164	0,1423	0,1026
Rata-rata	0,1074	0,1161	0,1138

Dari pengujian yang telah dilakukan nilai *jitter* dalam jaringan dipengaruhi oleh kedalaman dari *buffer jitter* dalam peralatan jaringan. Jika *buffer* jitter tersedia lebih banyak, maka jaringan dapat mereduksi efek dari *jitter*.

Gambar 17 merupakan grafik perbandingan *jitter* pengujian pengiriman data dengan *packet size* yang berbeda.



Gambar 17. Grafik Perbandingan *Jitter*

Dari *capture* data yang telah dilakukan, maka didapatkan *packetloss* dengan cara perhitungan sebagai berikut.

TABEL IV
HASIL PENGUJIAN *PACKETLOSS* DENGAN *PACKET SIZE* YANG BERBEDA

Perangkat	***Packetloss* Dengan *Packet Size* -**		
	124 KB (s)	505 KB (s)	1,32 MB (s)
Perangkat 1	0	0	0
Perangkat 2	0	0	0
Perangkat 3	0	0	0

Dari pengujian yang telah dilakukan pada *transfer* data 124 KB, 505 *KB* dan 1,32 *MB* digolongkan kedalam kategori baik, dari keseluruhan data yang dikirim *packetloss* yang hasilkan sebesar 0%.

IV. KESIMPULAN

Berdasarkan pembahasan dan pengujian yang telah dilakukan mengenai *website* sinkronisasi penyimpanan dokumen secara *realtime* menggunakan *websocket* pada *cloud storage*, dapat disimpulkan sebagai berikut.

1. Hasil dari dua pengujian *websocket* dengan metode *stress testing* menggunakan *autocannon* didapatkan hasil bahwa penggunaan *websocket* dapat merespon lebih banyak *request* yaitu 1000 *request* saat *websocket* tidak berjalan dan 8000 *request* saat *websocket* berjalan dalam 5 detik pengujian
2. Berdasarkan pengujian pada masing-masing fitur pada *website*, persentase keberhasilan *websocket* untuk sinkronisasi penyimpanan pada *cloud storage* mencapai 100%.
3. Hasil pengukuran *QoS* didapatkan nilai rata-rata *delay file* ukuran 124 *KB* sebesar 0,2479 detik , *file* ukuran 505 *KB* sebesar 0,2874 detik dan *file* ukuran 1,32 MB sebesar 0,8231 detik. Nilai rata-rata *jitter file* ukuran

124 *KB* sebesar 0,1108 detik , file ukuran 505 *KB* sebesar 0,1252 detik dan *file* ukuran 1,32 MB sebesar 0,1137 detik dan dari keseluruhan data yang diuji packetloss yang didapatkan sebesar 0%.

SARAN

Dalam pengembangan *website* sinkronisasi penyimpanan dokumen secara *realtime* menggunakan *websocket* pada *cloud storage* masih memiliki kekurangan dan kelemahan. Oleh karena itu, diperlukan pengembangan selanjutnya untuk penyempurnaan *website* ini. Adapan saran untuk pengembangan *website* dalam pengembangan selanjutnya yaitu sebagai berikut.

1. *Website* ini diharapkan dapat dikembangkan untuk dijalankan pada sistem operasi *mobile*.
2. Sistem *website* penyimpanan dapat dikembangkan dari segi keamanan data menggunakan metode *enkripsi* agar *file* terjaga keamanan dan kerahasiaannya.
3. Pengembangan *user interface* diperlukan agar meningkatkan pengalaman pengguna dalam menggunakan *website*.

REFERENSI

[1] Salamadian, 2020. "Fungsi, Sejarah, Kegunaan, Jenis Jenis & Contoh Web [Online]. Sumber : https://salamadian.com/pengertian-website/

[2] Darsiwan.(2016). Apa itu WebSocket. In Codepolitan. https://www.codepolitan.com/menegtahui-apa-itu-websocket/

[3] Santiko, I., & Rosidi, R. (2018). Pemanfaatan Private Cloud Storage Sebagai Media Penyimpanan Data E-Learning Pada Lembaga Pendidikan. Jurnal Teknik Informatika, 10(2), 137–146. https://doi.org/10.15408/jti.v10i2.6992

[4] Maulana, A. R., & Rahmatulloh, A. (2019). Websocket untuk Optimasi Kecepatan Data Transfer pada Real Time Chatting. Innovation in Research of Informatics (INNOVATICS), 1(1), 7–12. https://doi.org/10.37058/innovatics.v1i1.667

[5] Warsito, A. B., & Yuliandini, E. (2017). Penerapan Google Drive Sebagai Media Penyimpanan Bahan Perkuliahan Dalam Mendukung Aplikasi Mobile App Deployment of Google Drive as Storage Media Materials In Support of Mobile App Applications. 7(2), 219–228.

[6] Yudianto P, A., Sakti P, E., & Amron, K. (2017). Pengembangan Push Notification Menggunakan Websocket. Jurnal Pengembangan Teknologi Informasi Dan Ilmu Komputer (JPTIIK) Universitas Brawijaya, 1(1), 1–7.