IMPLEMENTASI *LOAD BALANCING BERBASIS CLOUD COMPUTING* UNTUK MENDUKUNG *INFRASTRUCTURE AS A SERVICE* MENGGUNAKAN METODE *QUICK SORT*

Hero Wintolo[1], Lalu Suwandi Yusuf [2]
Departemen Informatika
Sekolah Tinggi Teknologi Adisutjipto Yogyakarta
Jl. Janti, Blok-R, Lanud Adisucipto Yogyakarta
[1]herowintolo@stta.ac.id, [2] alhamdulillahyusuf@gmail.com

ABSTRACT

Implementation of load balancing in cloud computing applications using quick sort method is a study to determine the access speed of each server. Design a cloud computing application using 4 laptops as application users and 3 laptops or PCs as servers in data storage. In sending data performed by the user application, the data will be distributed to the server that has been selected by the system with a quick sort method that sort the rest of the server capacity and then choose the server with the largest capacity. The stored data is placed in the htdocs file on the server. Based on the tests performed, cloud computing applications can run on at least 2 laptops using the windows version. Testing the server is also done by directly using the command prompt. During testing each server receives ping from the application to determine server performance in using load balancing. Testing without load balancing that sends data 50 times of 10 Mb to one server yields speed 15 ms, with 2 server divided by two in its data delivery that is as much as 25 times 10 Mb then speed 12 ms.

***Keywords** : Load Balancing, Cloud Computing, Quick Sort.*

1. Latar Belakang Masalah

Perkembangan dunia teknologi informasi saat ini sangatlah pesat, di mana kebutuhan teknologi oleh masyarakat pada umumnya untuk membantu pekerjaan sehari-hari sangat diperlukan. Oleh karenanya bermacam-macam aplikasi atau *softwere* dibuat guna membantu pekerjaan sehari-hari untuk digunakan oleh masyarakat pada umumnya, sehingga dapat dirasakan manfaatnya dan membantu meringankan beban kerja. Teknologi c*loud computing* dihadirkan sebagai upaya untuk memungkinkan akses sumber daya dan aplikasi dari mana saja melalui jaringan Internet, sehingga keterbatasan pemanfaatan infrastruktur ICT yang sebelumnya ada dapat diatasi[1]. Jaringan komputer dalam bentuk local area network (LAN) dapat dipergunakan untuk cloud computing dalam bentuk layanan Cloud Infrastructure as a Service (IAAS)[2].

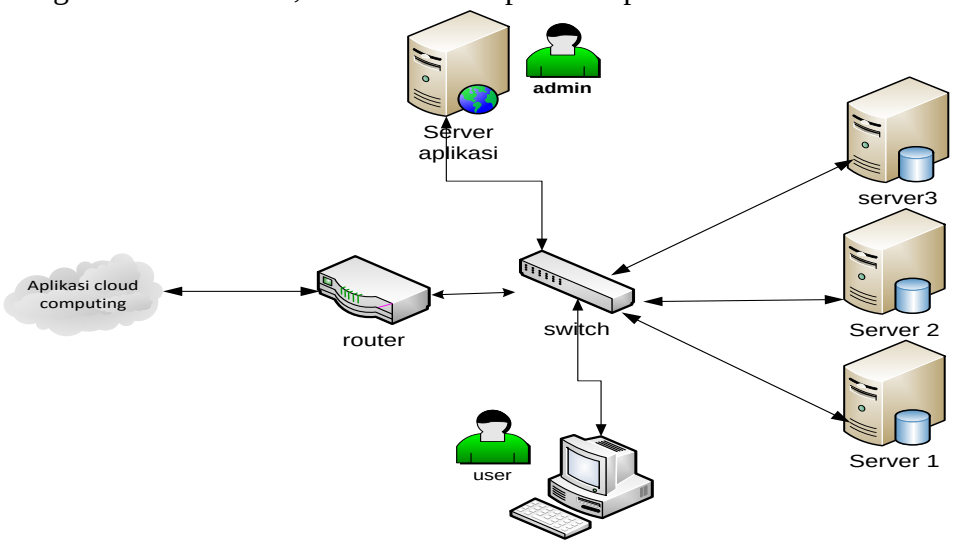
Salah satu layanan Cloud Computing adalah Infrastructure as a Service untuk melayani layanan infrastruktur seperti switch, prosesor, RAM atau memori, dan penyimpanan disk yang cocok untuk kebutuhan pengguna[3]. Dengan infrastruktur seperti ini maka layanan cloud computing dapat dibuat dan diperuntukan pada hal-hal yang berguna bagi masyarakat, salah satunya Sistem Informasi Manajemen (Simpuskesmas). Simpuskesmas Berbasis Cloud Computing merupakan Simpuskesmas yang dulu dalam penerapannya menggunakan client server dirubah ke Simpuskesmas yang di upload di web sehingga puskesmas-puskesmas di Kabupaten Demak dapat mengakses SIMPUS tersebut melalui internet[4].

Dalam kebanyakan kasus, organisasi/ perusahaan yang menggunakan fasilitas komputasi awan pada dasarnya tidak perlu lagi mengetahui dimana sesungguhnya layanan yang diperlukannya dilakukan/diproses, serta tidak perlu tahu lagi dimana data mereka disimpan, karena semuanya sudah diatur oleh vendor komputasi awan. Sementara spesifikasi secara internalnya relatif tidak

sama untuk masing-masing vendor, komputasi awan dapat kita pikirkan sebagai sumberdaya perangkat keras (komputer, *hardisk*, dan jaringan komputer) dan perangkat lunak yang bersifat koheren, berukuran raksasa, dan dapat diakses secara meluas. Jadi pada umumnya teknologi *cloud computing* secara tidak langsung membutuhkan yang sumberdaya-sumberdaya yang ada guna mendukung kelancaran akses oleh pengguna. Maka daripada itu perlu adanya sumberdaya yang bisa mengatur jalannnya lalu lintas data dalam mengakses *cloud computing*, agar proses akses berjalan dengan lancar dan tanpa hambatan. Salah satu sumberdaya yang digunakan dalam mengatur lalu lintas data yaitu dengan menggunakan metode *load balancing*. Hasil pengurutan menggunakan metode *Quick Sort* ini dapat digunakan untuk *load balancing* dari instruksi yang akan dikirimkan oleh *server* ke komputer *client* untuk dilakukan sebuah proses, sehingga keseimbangan kinerja dari aplikasi pemrosesan paralel ini dapat dicapai[5].

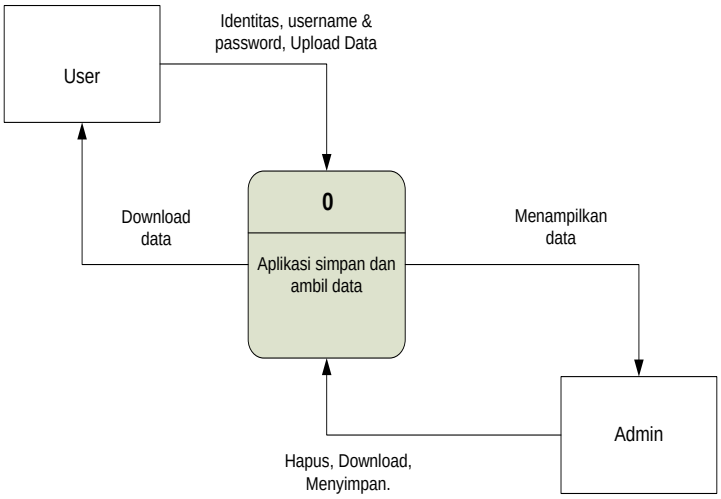
2. Metodologi Penelitian

Teknologi *cloud computing* merupakan sebuah model yang memungkinkan untuk *ubiquitous* (Di manapun dan kapanpun), nyaman digunakan, lebih leluasa akses jaringan ke sumber daya komputasi (contoh: jaringan, server, *storage*, aplikasi, dan layanan) yang dapat dengan cepat dirilis atau ditambahkan[6]. Gambar 1 menunjukkan konsep dari implementasi *load balancing* yang akan dibangun. Aplikasi ini merupakan aplikasi berbasis *web* dan memiliki beberapa *server* data yang berfungsi sebagai tempat penyimpanan data. Proses awal dimulai dari *user* melakukan *upload* sebuah data pada aplikasi. Data-data tersebut akan disimpan pada yang *server* data dengan cara *load balancing* ke *server* yang tersedia. Kemudian, ketika data akan disimpan oleh aplikasi, data tersebut akan tersimpan kepada *server* yang lebih banyak kapasitas penyimpanannya dengan metode *quick sort* yaitu *server* akan diurutkan sesuai dengan kapasistas sisa penyimpanannya masih paling besar. Kemudian, barulah data dapat disimpan.



Gambar 1. Gambaran Sistem Aplikasi *Cloud Computing*

Gambar 2 merupakan diagram konteks aplikasi *cloud balancing*. Pada Sistem ini hanya terdapat dua entitas yaitu *user* dan admin. Proses-proses yang dilakukan *user* terhadap sistem yaitu *user* dapat melakukan registrasi *member* dengan memasukan data-data identitas. Agar dapat melakukan *upload* data, *user* harus melakukan *login member* terlebih dahulu dengan memasukan *username* dan *password*. Setelah *upload* data, data tersebut akan ditampilkan ke *user*. Proses – proses yang dilakukan oleh entitas admin adalah melakukan *login* admin, melihat data *member*, *input* data *server*, hapus *member* dan hapus data.

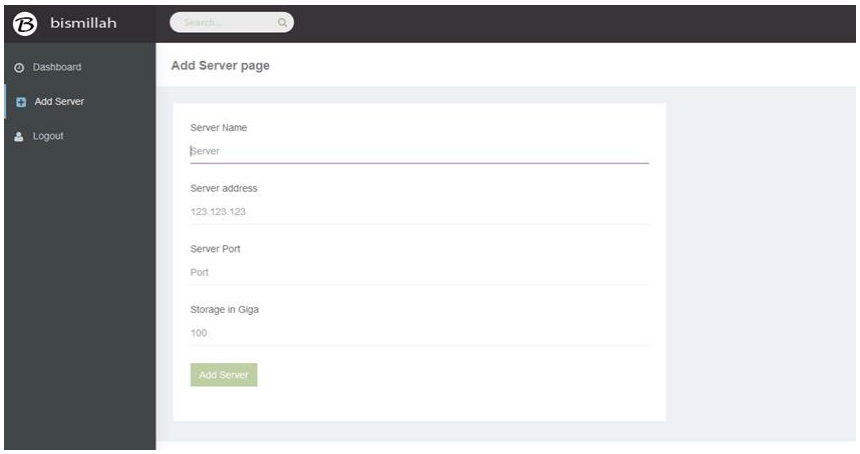


Gambar 2. Diagram Konteks Aplikasi *Cloud Computing*

3. Hasil dan Pembahasan

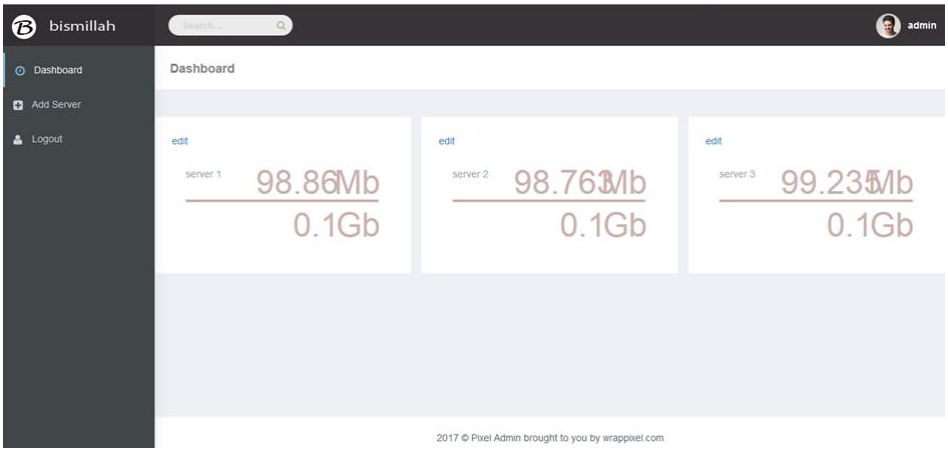
3.1. Penjelasan Hasil Rancangan

Hasil rancangan merupakan hasil penerapan yang telah di konversi menjadi sebuah sistem secara utuh. Pada bagian ini akan mengurai bagaimana hasil rancangan tersebut dalam sebuah keutuhan program. Berikut adalah bagian-bagian dari hasil perancangan yang telah di konversi menjadi sebuah sistem. Pada Gambar 3 terdapat tombol *Add server* pada sebelah kiri gambar yang fungsinya untuk membuat *server* baru. Dan tombol *logout* untuk keluar dari halaman akses.

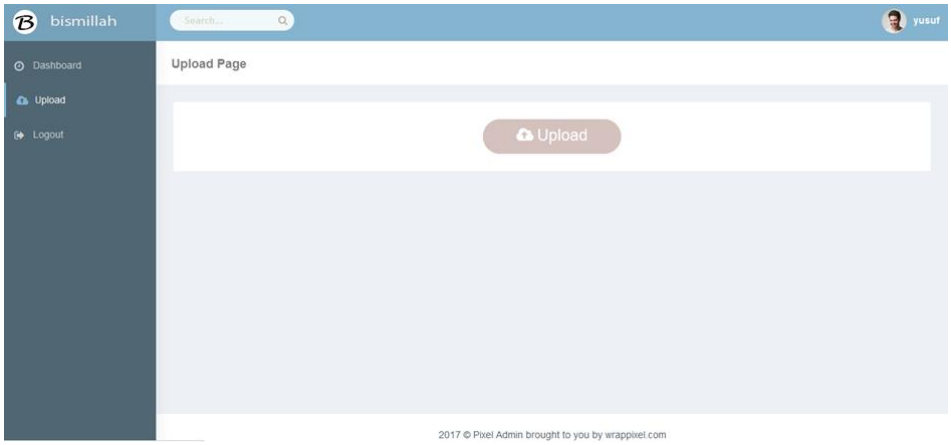


Gambar 3. *Form Create Server*

Gambar 4 merupakan halaman untuk server, disini terlihat beberapa *server* yang telah dibuat dan diisi sebelumnya sehingga menampilkan kapasitas *server* . Dan pada halaman ini terdapat tombol *add server* yang berfungsi untuk membuat sebuah *server* baru sesuai dengan kebutuhan aplikasi. Dalam percobaan ini terdapat 3 buat *server yang* dimana keseluruhan *server* diatur oleh *admin*. Di setiap *server* terdapat alamat IP yang berbeda-beda. Ketika sebuah data akan disimpan ke *server*, data yang masih dalam pengiriman akan memilih *server* yang kapasistas pengimpanannya masih banyak atau masih belum terpakai dengan metode *quick sort.*

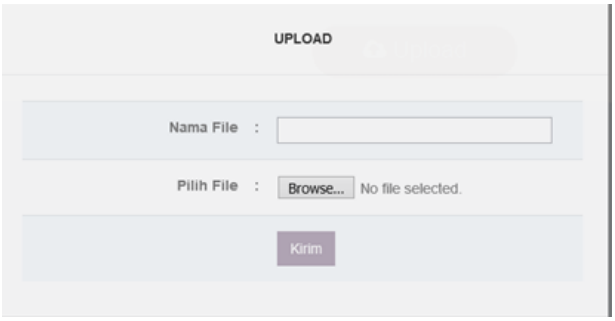


Gambar 4. *Form* Halaman *server*



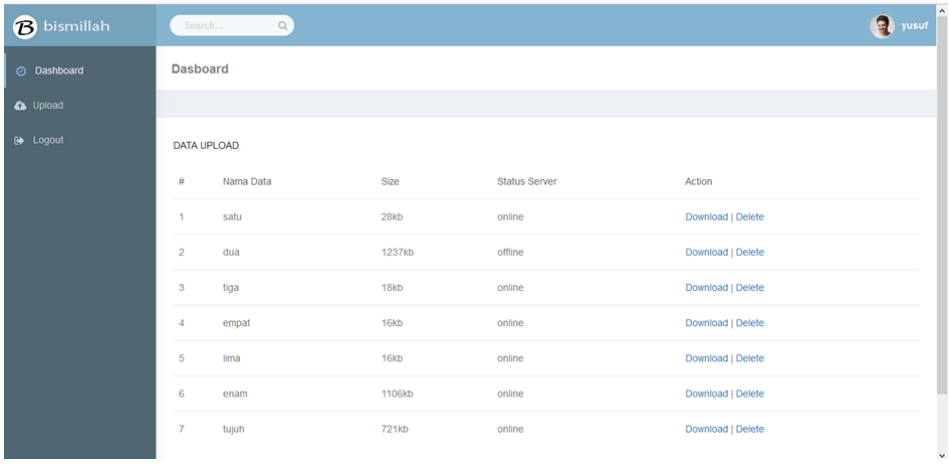
Gambar 5. *Form* Halaman *Upload*

Form upload ini berfungsi sebagai saranan untuk membatu *user* dalam meng*upload document*. Di form ini tersedia pilihan *upload* yang kemudian *user* menekan tombol *upload*. Kemudian akan menampilkan form untuk menamai data dan memilih data untuk transaksi *upload*.



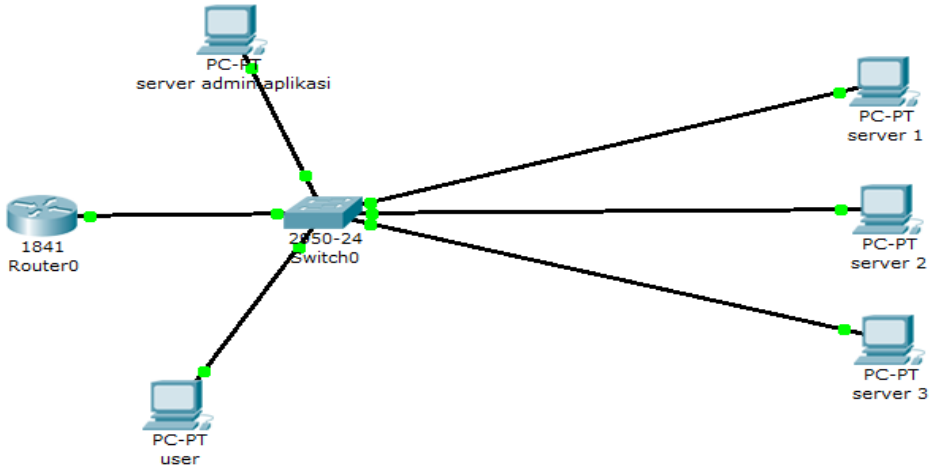
Gambar 6. *Form* Transaksi *Upload*

Proses transaksi *Upload* dilakukan pada halaman transaksi seperti terlihat pada Gambar 6, *user* diharuskan mengisi nama data dan *file* data yang akan disimpan ke *server*. Setelah *user* mengisi data, klik tombol kirim untuk menyimpan data yang telah dipilihnya. *File* yang telah diunggah akan terlihat seperti pada gambar 7. Pada halaman daftar ini, *user* bisa melihat data yang sudah di*upload*nya. *User* bisa *mengakses* data miliknya sendiri dengan mengklik tombol *download* dan bisa juga untuk menghapus data miliknya sendiri dan *user* bisa mengetahui bahwa status server tempat data *user* berada dalam keadaan *online* atau *offline*. Meskipun dalam keadaan *offline user* masih bisa mengakses data yang telah disimpan sebelumnya.



Gambar 7. *Form* Halaman Daftar *File Upload*

3.2. Pengujian



Gambar 8. Skema Pengujian

Pengujian yang pertama kali merupakan pengunjian fungsi untuk menilai kinerja hasil rancangan sudah sesuai atau belum dengan tujuan penelitian. Selama uji fungsi menggunakan simulasi, terlebih dahulu membuat alamat *IP* yang sifatnya statis guna memudahkan dalam pengujian. Dalam pengalamatan IP dapat di lihat pada Tabel 1yaitu daftar *IPAddress.*

Tabel 1. Daftar *IP Address*

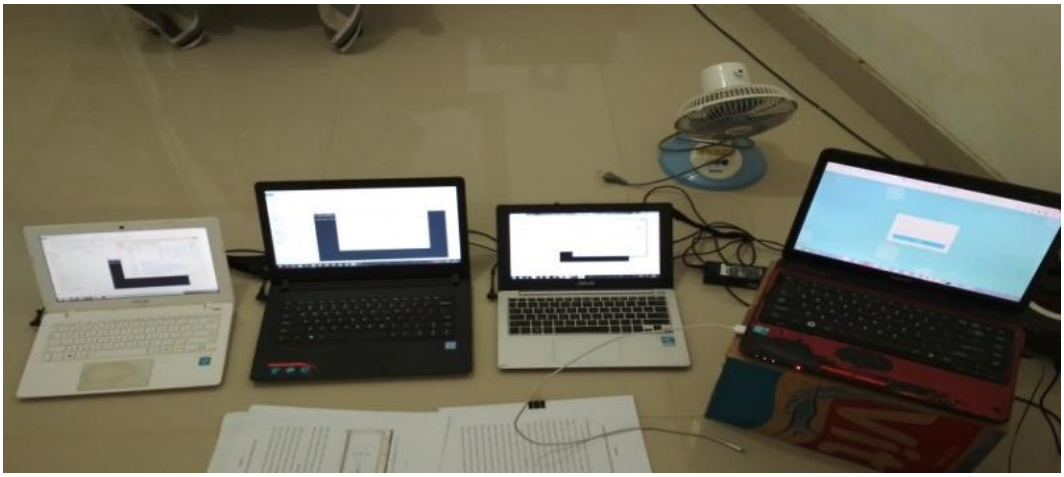
Nama Perangkat	IP	Subnet	Server
Router	192.168.100.1	255.255.255.0	-
Server Aplikasi	192.168.100.2	255.255.255.0	192.168.100.1
Server Data 1	192.168.100.3	255.255.255.0	192.168.100.1
Server Data 2	192.168.100.4	255.255.255.0	192.168.100.1
Server Data 3	192.168.100.5	255.255.255.0	192.168.100.1
Komputer User	192.168.100.5	255.255.255.0	192.168.100.1

Setiap *server* mempunyai alamat *IP* yang berbeda-beda yaitu *server* 1 *IP Address*nya 192.168.10.3, *server* 2 *IP Address*nya 192.168.10.4, *server* 3 *IP Address*nya 192.168.10.5. sedangkan komputer *user* 192.168.10.6 dan *server* aplikasi IP *Address*nya 192.168.10.2. Dalam uji coba mendapatkan hasil proses bahwa ketika *user* mengakses setiap *server*, waktu yang dibutuhkan berbeda-beda pada saat menggunakan alat simulasi *softwere packet tracer*.

Tabel 2. Hasil Proses Pengujian Menggunakan *Packet Tracer*

Server	Waktu Min	Waktu Max	Rata-rata
Server 1/192.168.10.3	46 ms	63 ms	57 ms
Server 2/192.168.10.4	31 ms	63 ms	58 ms
Server 3/192.168.10.5	31 ms	63 ms	59 ms

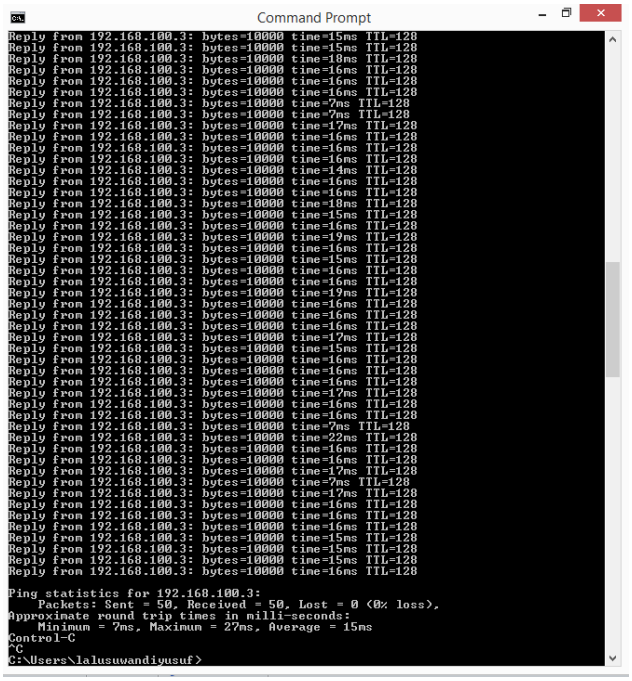
Pada Tabel 2, hasil proses pengujian menggunakan *packet tracer*, waktu rata-rata akses antara server satu dengan yang lainnya berbeda-beda. Hal ini mempengaruhi kecepatan dalam pengiriman sebuah paket data.



Gambar 9. Alat Uji Aplikasi dan *Server*

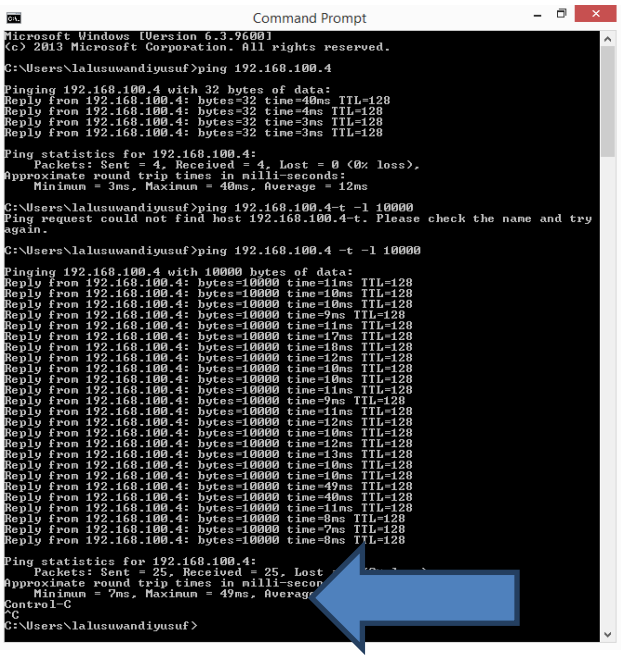
Pengujian selanjutnya dilakukan dengan mengimplementasikannya pada peralatan computer dan jaringan computer secara langsung. Dalam pengujian alat terdapat 4 buah komputer, 1 diantaranya digunakan sebagai *admin/user* selebihnya sebagai *server*. Dalam pengujian akan dilakukan dua cara untuk mengetahui perbedaan dari *load balancing* melalui *cmd online* dan data yang dikirim dalam pengujian, sebesar 10.000 *byte* atau 10 *Mb*.

Pengujian yang pertama menggunakan satu *server* (*tanpa load balancing*) yaitu pengujian yang dilakukan oleh *user* guna melakukan pengiriman paket data ke salah 1 *server* dari 3 *server* yang ada sebanyak 50 kali. Hasilnya, kecepatan *minimum*nya 7 *ms* rata-rata yang didapatkan 15 *ms*.

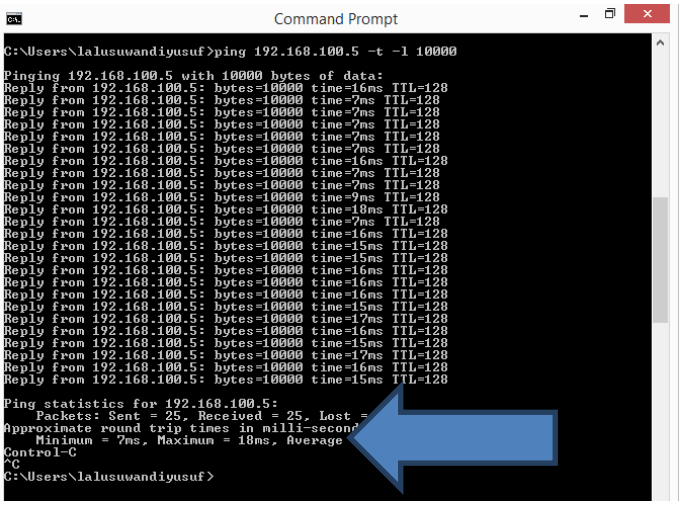


Gambar 10. Hasil Pengujian dengan 1 *Server*

Pengujian yang kedua dengan melibatkan 2 *server* (*load balancing*) yaitu untuk membuktikan bahwa penggunaan *load balancing* dapat dapat lebih mengefisiensikan waktu pengiriman data. Pada proses dan hasil pengujian dengan 2 server bagian satu merupakan hasil perbandingan dari pengiriman *packet* data ke *server*, yang mana hasil setiap kecepatan yang didapatkan rata-rata 13 *ms* akan menjadi perbandingan untuk pengujian yang menggunkan 2 *server*.



Gambar 11. Proses dan Hasil Pengujian dengan 2 *Server* Bagian Satu.



Gambar 12. Proses dan Hasil Pengujian dengan 2 *Server* Bagian Dua

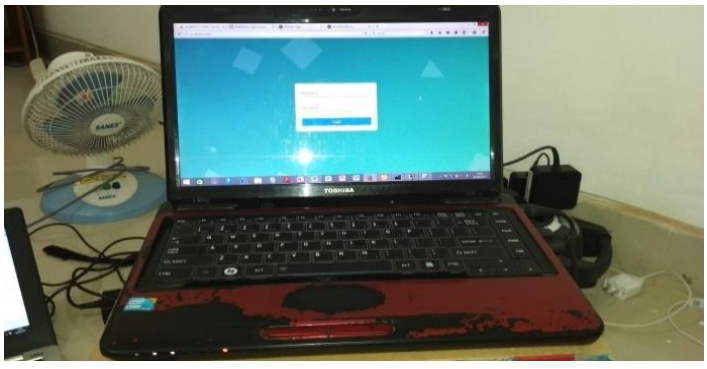
Selanjutnya proses dan hasil pengujian dengan 2 *server* bagian dua juga mendapatkan kecepatan dengan rata-rata 12 *ms.*

Tabel 3. Hasil proses pengujian menggunakan *online*

Nama/Alamat IP	Waktu Min	Waktu Max	Rata-rata	Load balancing
Server 1 /192.168.100.3	7 ms	27 ms	15 ms	Tidak
Server 2 /192.168.100.4	7 ms	49 ms	13 ms	Pakai
Server 3 /192.168.100.5	7 ms	18 ms	12 ms	Pakai

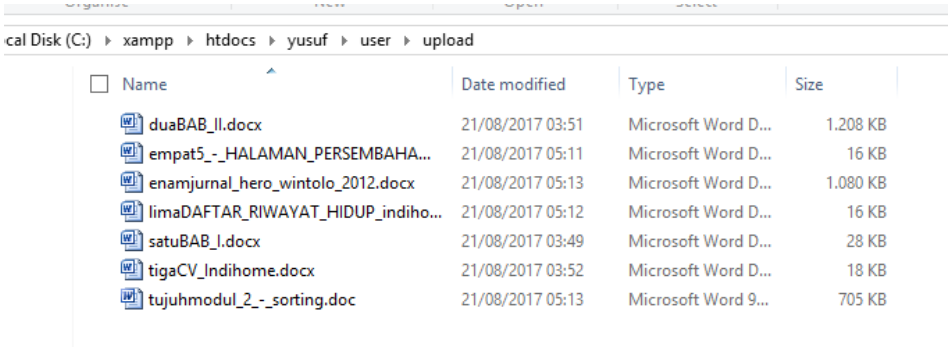
Dalam pengujian tersebut, dapat disimpulkan bahwa penggunaan dua *server* atau lebih (*load balancing*) mengakibatkan proses yang dihasilkan lebih ringan dalam memproses suatu pengiriman paket data dari pada tidak menggunakan *load balancing*. Seperti pada gambar 4.9, uji coba dilakukan dengan memberikan 10.000 *byte* atau 10 *Mb* selama 50 kali akses kepada *server* 1, kemudian 25 kali akses pada *server* 2 dan 3 guna mengetahui kinerja dari setiap pada *server*.

Kemudian melakukan dan memperlihatkan uji alat dari beberapa *server* yang sudah dikirimkan data dari aplikasi. Dalam berbagai percobaan yang dilakukan, data yang dikirimkan oleh *user* dapat berbeda tempat penyimpanannya dikarenakan aplikasi akan membaca kapasitas *server* yang tersedia dan kemudian kapasitas *server* yang paling besar akan didahulukan dalam pengisian data yang dikirimkan oleh *user.*



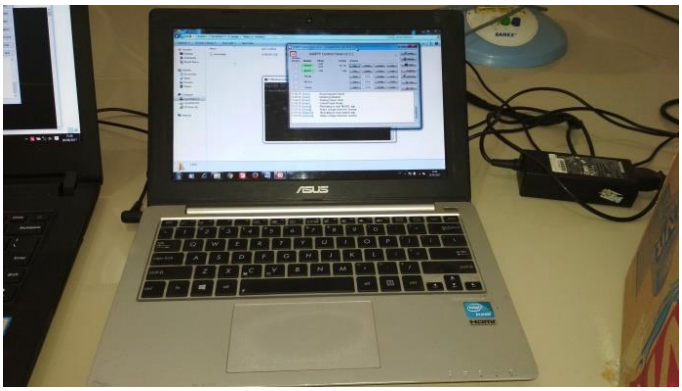
Gambar 13. Alat Uji Satu Sebagai Aplikasi

Pada alat uji satu berfungsi sebagai *user* sekaligus sebagai *admin* dalam aplikasi. Selain itu juga berfungsi sebagai *server* aplikasi yang mem*backup file* setiap kali ada transaksi pengiriman data ke *server*. sehingga bilamana ada salah satu dari 3 *server* dalam keaadaan *offline*, data yang ada di dalamnya masih bisa diakses.



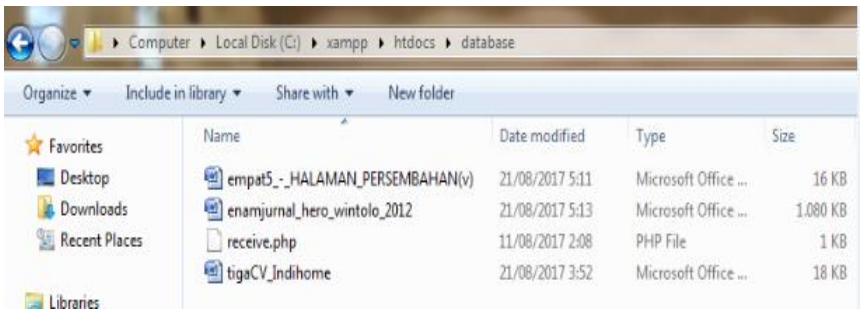
Gambar 14. Bukti *Backup* Data Pada *Server* Aplikasi

Gambar bukti *backup* data pada *server* aplikasi menunjukkan data *user* juga berada pada *server* aplikasi. Untuk mengantisipasi *user* yang akan mengaskes datanya jika sewaktu-waktu *server* ada yang *offline.*

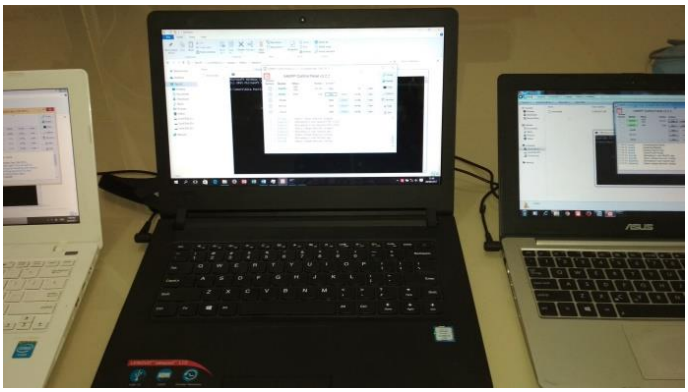


Gambar 15. Alat Uji Dua Sebagai *Server* 1

Pada alat uji dua merupakan salah satu *server* yang menjadi *server* 1 dalam menyimpan data *user* dan telah menyisipkan alamat *IP* 192.168.100.3. Hasil dari penyimpanan *file user* ke *server* 1 menunjukan bahwa keberhasilan transaksi yang dilakukan *user* dalam menyimpan *file* data pada *server* 1.

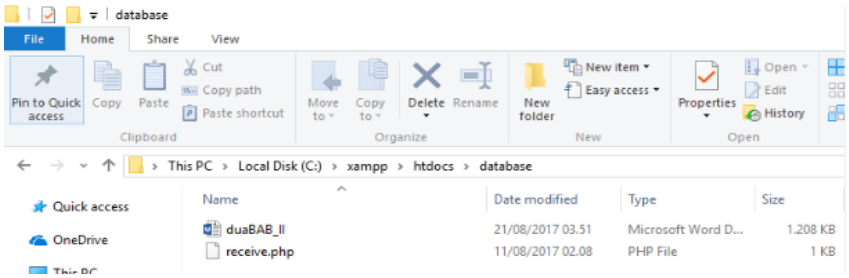


Gambar 16. Bukti Terkirimnya *File* Data Pada Laptop 1 Sebagai *Server* 1



Gambar 17. Alat Uji Tiga Sebagai *Server* 2

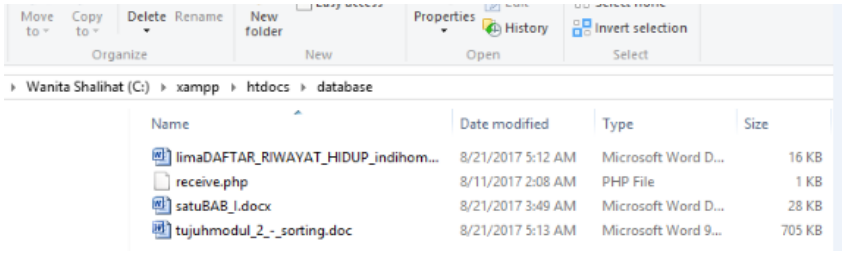
Pada alat uji tiga yang berfungsi sebagai *server* 2 sudah di *setting* memiliki alamat *IP* 192.168.100.4 dan kapasitas untuk *server*. Pada *server* alat uji tiga, data yang dikirimkan oleh user dari aplikasi ke *server* bahwa data tersebut sudah masuk dan bertempat pada folder penyimpanan yang ada di *server* 2.



Gambar 18. Bukti Terkirimnya *File* Data Pada Laptop 2 Sebagai *Server* 2

Gambar 19. Alat Uji Empat Sebagai *Server* 3

Selanjutnya laptop terakhir yang digunakan sebagai alat uji yang ke empat, fungsi utamanya juga tidak jauh berbeda dengan *server* 1 dan server 2 yaitu sebagai *server* 3. Pada *server* 3 sudah ada alamat IP 192.168.100.5 yang sudah di *setting* sebelumnya. Kemudian dari hasil beberapa percobaan yang dilakukan, data *user* disimpan pada *server* 3. Artinya ketika *user* mengirimkan *file* data, aplikasi akan mengarahkan data *user* ke tempat *server* yang kapasitas ruang penyimpanannya masih tersisa banyak. Meskipun dalam pengiriman data cukup banyak, akan tetapi yang terhitung pada aplikasi adalah kapasitas *file* data.



Gambar 20. Bukti Terkirimnya *File* Data Pada Laptop 3 Sebagai *Server* 3

4. Kesimpulan

Kesimpulan yang diperoleh dari hasil uji fungsi dan uji alat aplikasi *cloud computing* ini adalah sebagai berikut :

1. Pengiriman data sebanyak 50 kali dan data yang dikirim sebesar 10.000 *byte* atau 10 *Mb* kepada 1 *server* yang menghasilkan kecepatan 15 *ms*, dengan 2 *server* dibagi dua dalam pengiriman datanya yatu sebanyak 25 kali dan sebesar 10.000 maka kecepatan 12 *ms*.
2. Penerapan metode *quick sort* mengurutkan sisa kapasitas yang paling besar sehingga bisa dipastikan data tersebut berada pada *server* yang kapasitasnya paling besar.
3. Data yang tersimpan tetap bisa diakses oleh user. Data tersebut tersimpan pada *server* aplikasi dan juga disalah satu *server* yang sudah ada. Sehingga data yang disimpan bisa diakses dalam keadaan *off*.

Daftar Pustaka

[1] Ashari, A. and Setiawan, H., 2011. Cloud Computing: Solusi ICT?. *Jurnal Sistem Informasi*, *3*(2).

[2] Wintolo, H., Kusumaningrum, A. and Kusuma, H.W., 2017. Use of Automation Codecs Streaming Video Applications Based on Cloud Computing. *Telkomnika*, *15*(3).

[3] Sulistyowati, L., Sulistyo, W. and Bayu, T.I., 2012. Implementasi Cloud Computing Sebagai Infrastructure as a Service untuk Penyediaan Web Server.

[4] Wibisono, S. and Munawaroh, S., 2012. Sistem Informasi Manajemen Puskesmas (Simpuskesmas) berbasis Cloud Computing. *Dinamik-Jurnal Teknologi Informasi*, *17*(2).

[5] Wintolo, H., Penerapan Quick Sort pada Pemrosesan Parallel Berbasis Personal Komputer (PC) dalam Local Area Network (LAN) Guna Mendukung Load Balancing.

[6] Mell, P. and Grance, T., 2011. The NIST definition of cloud computing.