Pengembangan Manajemen Multi Server Berbasis Web Menggunakan Framework Django

Development of Web-Based Multi-Server Management Using the Django Framework

Dadan Irwan[1*], Taufiqur Rokhman[2], Sisferi Hikmawan[3]
[1]Department of Computer Engineering/ Faculty of Engineering
[2]Department of Machine Engineering/Faculty of Engineering
Islamic University 45, Cut Mutia No. 83 Bekasi City phone 021-8801027
[3]Department of Computer Science, STMIK Nusa Mandiri Jakarta
dadanirwan@unismabekasi.ac.id[1*], rokhman_taufik@gmail.com[2], gansisferi@gmail.com[3]

Abstrak *- Sebuah server dirancang dengan tujuan untuk dapat memberikan layanan secara terus menerus dengan ketersediaan layanan di atas 99.9% tanpa gangguan kepada pengguna. Pada setiap server perlu dilakukan manajemen sistem seperti update sistem dan aktivitas lainnya. Proses untuk meremote server memerlukan aplikasi Putty SSH agar administrator dapat masuk ke dalam sistem dan dapat melakukan manajemen pada server. Penelitian ini bertujuan membangun aplikasi manajemen multi-server untuk melakukan pengelolaan pada beberapa mesin server (multi-server) secara otomatis dan terintegrasi menggunakan framework Django. Tahapan yang dilakukan meliputi analisis sistem dan inventarisasi permasalahan yang sedang berjalan, perancangan aplikasi mengunakan aplikasi framework Django, pengujian aplikasi dilakukan dengan 2 skenario berbeda, kemudian implementasi sistem dan analisa sistem. Berdasarkan pengujian dengan melakukan aktivitas update sistem yang dilakukan secara manual menggunakan Putty SSH pada 1 unit server diperoleh durasi waktu sekitar 39 detik. Sementara pengujian dengan aplikasi manajemen multi-server memerlukan waktu sekitar 32 detik pada 2 unit server berbeda, sehingga diperoleh tingkat efisiensi waktu sebesar 50%.*

Kata Kunci: *Server, Framework, Django, Multi-Server, Putty SSH*

Abstract *– A server is designed to be able to provide services continuously with the availability of services above 99.9% without interruption to users. On each server system management needs to be done such as system updates and other activities. The process for a remote server requires the Putty SSH utilization so that administrators can enter the system and can do management on the server. This research proposes to build a multi-server management application to manage multiple server machines (multi-server) automatically and integrated using the Django framework. Steps being taken include system analysis and inventory of ongoing problems, application design using the Django framework application, application testing is carried out with 2 different scenarios, then system implementation and system analysis. Based on testing by doing a system update activity that is done manually using Putty SSH on 1 server unit, the duration time is around 39 seconds. While testing with a multi-server management application takes about 32 seconds on 2 different server units, a time efficiency level of 50% is obtained.*

Keywords*: Server, Framework, Django, Multi-Server, Putty SSH*

1. Pendahuluan

Server merupakan perangkat penting pada jaringan komputer yang memiliki fungsi dalam menyediakan pusat layanan (*services*), data, informasi dan sumber daya yang diperlukan *client*

[1]. Sebuah Server dibangun untuk dapat memberikan layanan secara terus menerus pada tingkat *high availability* di atas 99.9% tanpa gangguan kepada pengguna [2]. Saat ini Server dikembangkan dengan berbagai varian *hardware* dan *software* yang bermacam-macam, sehingga pada penerapannya memerlukan administrator sistem yang memiliki keahlian khusus dalam proses pengelolaannya. Pengelolaan *server* mutlak dilakukan secara rutin dan terjadwal dengan baik agar layanannya dapat terus berjalan sesuai dengan fungsinya sehingga terhindar dari status *server down*. Pada implementasinya, seorang administrator biasanya menangani lebih dari satu mesin *server* (*multi-server*) seperti *web server*, *database server*, *mail server*, dan layanan *server* lainnya. Setiap layanan atau *services* dipasang pada satu mesin *native server* yang sama atau mesin *server* yang berbeda (*multi-server*). Bertambahnya kebutuhan server juga dapat mempersulit administrator dalam manajemen server dan juga menambah besarnya biaya yang dikeluarkan untuk membeli tambahan server [3].

Sebuah native server yang memiliki spesifikasi tinggi dengan teknologi virtualisasi dapat dimanfaatkan untuk memberikan lebih dari satu layanan. Teknologi virtualisasi yang ditanamkan pada setiap mesin server dapat digunakan dan dimanfaatkan untuk dibangun beberapa mesin server secara virtual yang memberikan layanan berbeda [4]. Secara teknis, pada setiap *server* sudah dipasang sistem operasi, system layanan (*services*), dan aplikasi lainnya. *Server-server* tersebut harus dikelola secara rutin seperti *update* sistem, *update* aplikasi, pemasangan aplikasi baru, dan monitoring penggunaan sumber daya *hardware* dan *software*. Permasalahan yang muncul yaitu ketika seorang *administrator server* melakukan pengelolaan pada banyak *server* (*multi-server*) yang dilakukan secara manual sehingga menjadi tidak efektif dan efesien baik dari sisi waktu maupun kinerja.

Pada penelitian terdahulu mengenai aplikasi berbasis smartphone untuk manajemen server pada private cloud computing telah dilakukan oleh Andi Gita Novianti, Muh. Niswar, dan Amil Ahmad Ilham. Pada penelitian ini telah dibangun aplikasi untuk pengelolan server yang dapat diakses melalui smartphone sehingga dapat diakses secara mobile [5]. Pada penelitian ini peneliti berfokus untuk membangun sistem aplikasi berbasis *web* yang dapat melakukan pengelolaan pada multi-server yang dapat bekerja secara otomatis dan terintegrasi. Penelitian ini menerapkan teknik virtualisasi agar dapat mengoptimalkan sumber daya secara maksimal dengan membangun mesin server-server virtual menggunakan aplikasi Proxmox [6]. Manfaat dari penelitian ini adalah agar pengelolaan pada beberapa mesin multi *server* dapat dilakukan secara efektif pada sisi *effort* dan efesien pada sisi waktu. Hasil yang ingin didapat dari penelitian ini adalah terdapatnya sebuah sistem aplikasi berbasis *web* yang dapat melakukan pengelolaan aktivitas layanan pada multi-server secara otomatis dan terintegrasi dan dapat menyediakan informasi setiap server secara spesifik dan komprehensif.

2. Metode Penelitian

Beberapa tahapan penelitian yang telah dilakukan antara lain:

2.1. Studi Pustaka

Dalam melakukan studi pustaka, kami menggunakan sistem operasi Ubuntu server sebagai studi kasus dalam penelitian kami karena Ubuntu server merupakan sistem operasi khusus untuk server yang termasuk dalam sistem operasi yang terbanyak digunakan di Indonesia.

2.2. Inventarisasi Masalah

Pada komunitas Ubuntu Indonesia, permasalahan yang paling sering dibahas adalah seputar Secure Shell (SSH) dan Backup [7]. Dari sini kami mencoba menjadikan perancangan aplikasi manajemen multi-server sebagai sebuah solusi tambahan dalam mengelola kebutuhan server menggunakan *protocol* yang populer dipakai oleh para SysAdmin atau Pengelola Server yaitu SSH. Dalam pengelolaan hampir seluruh kegiatan dilakukan menggunakan *protocol* SSH. Maka

ISBN: 978-602-60581-1-9

dari itu, aplikasi ini juga menambahkan fitur diantaranya, perintah-perintah standar yang sering digunakan dalam penggunaan SSH serta juga menambahkan kustomisasi perintah di SSH.

2.3. Analisa Kebutuhan

Kebutuhan dalam pembuatan aplikasi Manajemen Server diantaranya :

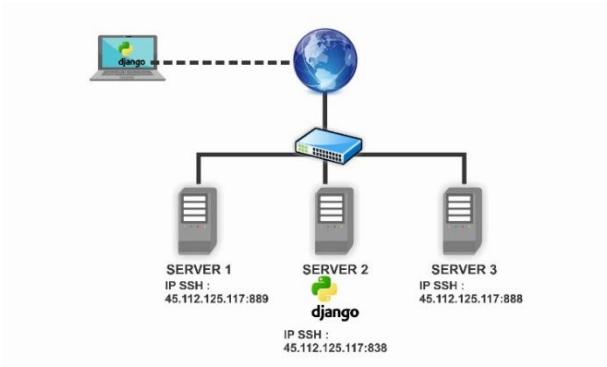
1. Sebuah server utama dan 2 server client dengan spesifikasi RAM 2 GB dan dan HDD 30GB
2. Sistem Operasi Ubuntu 18.0 yang sudah terinstal Python, MySQL, Apache, SSH
3. Koneksi jaringan Internet dan intranet

2.4. Perancangan Aplikasi dan Pengujian

Dalam perancangan aplikasi hanya menggunakan Python dan Django!, namun dalam pengerjaannya penggunaan Django membutuhkan beberapa plugin tambahan yaitu Celery, sebuah plugin tambahan yang bekerja di latar belakang untuk mengatur tugas pada Python jika ada pekerjaan yang memakan waktu agar tidak *time out* sehingga bisa melakukan pekerjaan lain selama yang sebelumnya belum selesai.

3. Arsitektur Sistem Jaringan

Pada perancangan sistem jaringan yang meliputi arsitektur atau topologi yang dikembangkan memerlukan beberapa perangkat keras diantaranya: 1) Tiga unit server, 2 unit server yang digunakan sebagai perangkat yang menjadi objek ketika pengujian dilakukan dan 1 unit server diperlukan untuk penyimpanan aplikasi Django. Ketiga server tersebut dibangun secara virtual menggunakan sistem virtualisasi Proxmox; 2) Satu unit laptop yang digunakan untuk mengakses asplikasi; dan 3) Jaringan Internet.



Gambar 3. Arsitektur sistem jaringan dengan django.

Sementara perangkat lunak yang digunakan antara lain: 1) Aplikasi Proxmox; 2) Sistem operasi Linux; 3) Aplikasi Django; dan 4) Aplikasi Putty

4. Rancangan Sistem

Aplikasi yang dirancang dengan menggunakan framework Django secara garis besar terdapat 2 file yang memiliki fungsi untuk melakukan remote SSH yaitu `urls.py` dan `views.py`. File `urls.py` disimpan di komputer server 2 pada direktori */home/server2/unismaserver/server/urls.py*. File tersebut berfungsi untuk koneksi ke server yang

ISBN: 978-602-60581-1-9

di remote menggunakan protokol SSH. Berikut *script* yang berfungsi untuk mengeksekusi perintah dari server local:

```
run=subprocess.run(command,stdout=subprocess.PIPE,stderr=subprocess.ST
DOUT, timeout=100)
```

Sementara *script* yang berfungsi untuk melakukan remote SSH ke server lain yaitu:

```
client.connect(hostname=hostname,port=port,username=username,password=
password)
stdin,stdout,stderr=client.exec_command('sudo '+ command, get_pty=True)
```

File `views.py` berfungsi untuk menyimpan daftar server dan isi *command linux* yang disajikan untuk pengguna/pengelola server. File tersebut disimpan pada server2 dengan direktori */home/server2/unismaserver/server/views.py*. Pada file tersebut dirancang untuk melakukan aktifitas dalam pengelolaan layanan server diantaranya: 1) Melakukan tes koneksi (Ping) ke gateway server Django; 2) Melakukan Restart server; 3) Melakukan Shutdown server; 4) Melakukan Update sistem operasi dan aplikasi pada server. Beberapa script yang digunakan untuk melakukan pengelolaan pada server adalah sebagai berikut:

Mengirim perintah ping ke gateway server Django:

```
server=get_object_or_404(Server, pk=server_id)
result=server_command.delay(server.alamat_ip,server.port,.username,ser
ver.password, 'ping -c 3 45.112.125.114')
```

Mengirim perintah reboot ke server:

```
server = get_object_or_404(Server, pk=server_id) result =
server_command.delay(server.alamat_ip, server.port, server.username,
server.password, 'reboot')
```

Mengirim perintah shutdown ke server:

```
server=get_object_or_404(Server, pk=server_id)
result=server_command.delay(server.alamat_ip,server.port,server.userna
me,server.password,'shutdown')
```

Mengirim perintah apt-get update ke server:

```
server=get_object_or_404(Server, pk=server_id)
result=server_command.delay(server.alamat_ip,server.port,server.userna
me,server.password,'apt-get update')
```

Aplikasi ini bekerja menggunakan protokol SSH untuk meremote server berbasis linux, maka sangat memungkinkan juga untuk meremote platform lain selain server misalnya: Mikrotik Router, Cisco Router, Cisco Access point dan perangkat lainnya.

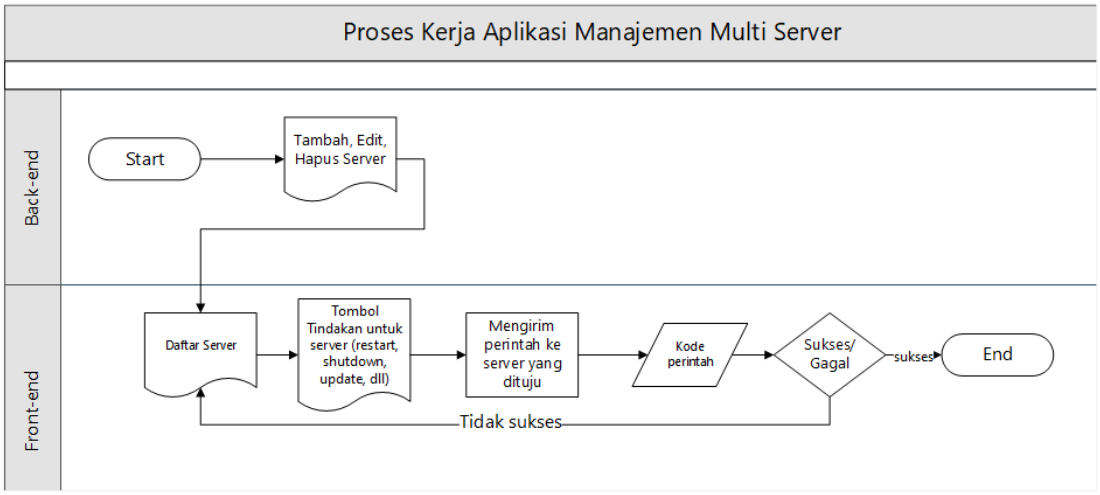
5. Hasil dan Pembahasan

Prinsip kerja dari aplikasi yang dirancang yaitu mampu menggantikan aktivitas pengelolaan pada server yang selama ini dilakukan secara manual dengan menggunakan aplikasi Putty SSH untuk meremote server. Pada aplikasi yang dikembangkan, pengelolaan server mulai dari remote SSH dan aktivitas layanan lainnya seperti *update* sistem dan administrasi sistem dapat dilakukan secara otomatis dan terintegrasi pada satu aplikasi. Pada Gambar 4 dapat dilihat proses bisnis kerja aplikasi manajemen multi server menggunakan framework Django.

Proses pengelolaan secara terintegrasi mampu menghemat aktivitas atau *effort* administrator dan durasi waktu kerja yang diperlukan. Pada sisi *back-end*, proses bisnis nya meliputi penambahan, edit, dan hapus server, sementara pada sisi *front-end* proses bisnis meliputi

ISBN: 978-602-60581-1-9

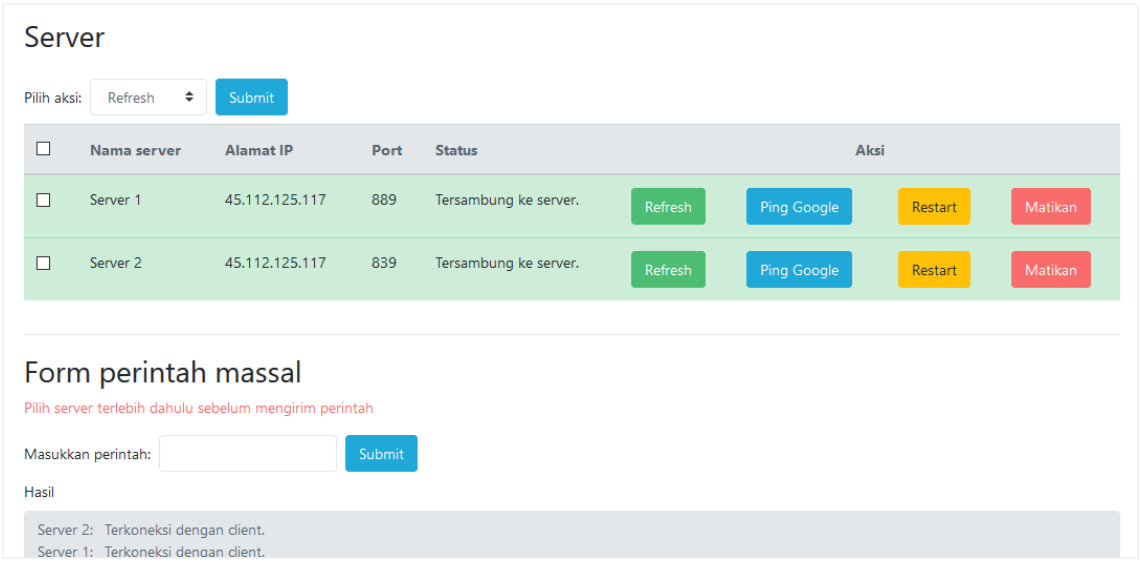
monitoring daftar server, kemudian memilih layanan server yang akan dilakukan, lalu perintah akan terkirim ke server yang diremote, dan apabila berhasil maka proses tersebut akan dijalankan oleh server, dan jika terjadi kegagalan maka proses akan kembali ke tahapan pemilihan server.



Gambar 4. Proses bisnis aplikasi manajemen multi-server.

5.1. Front-End User Interface

Aspek tampilan *user interface* pada aplikasi yang digunakan oleh pengelola server (administrator) dirancang untuk mempermudah dalam pengelolaan sistem. Pada penelitian ini terdapat 2 (dua) unit server yang digunakan untuk proses pengujian. Sementara aktivitas yang dapat dilakukan oleh administrator meliputi: 1) Test Koneksi (Ping) ke server; 2) Test koneksi (Ping) ke salah satu server Internet dalam contoh ini yaitu google.com; 3) Ping ke Server Utama; 4) Melakukan Restart ke setiap server; 5) Melakukan proses Shutdown ke setiap server; dan 5) Melakukan proses *Update* sistem pada setiap server. Pada Gambar 5, dapat dilihat bahwa seorang administrator server dapat melakukan aktivitas pengelolaan pada kedua server dalam waktu bersamanaan dan hanya 1 (satu) halaman antar muka.

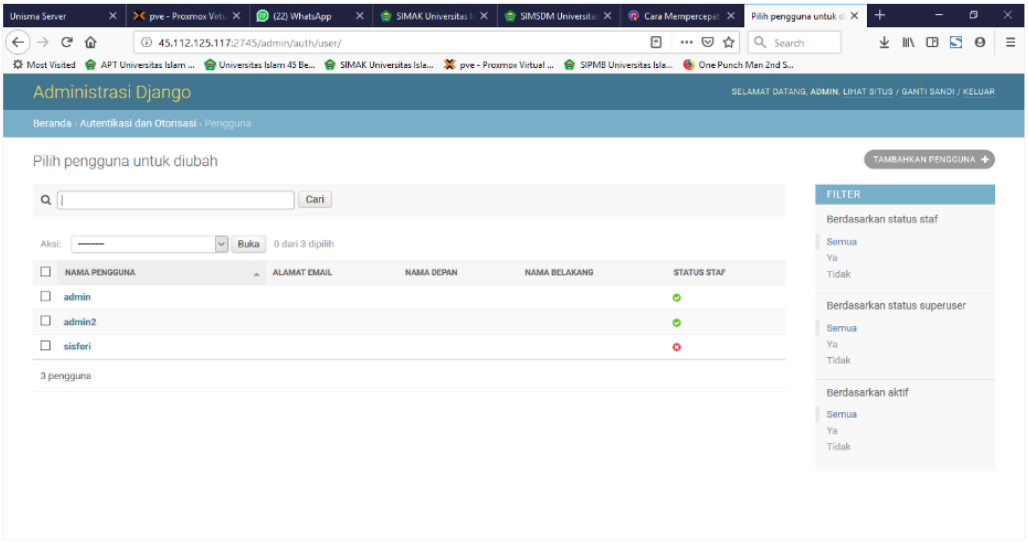


Gambar 5. *User interface* aplikasi.

ISBN: 978-602-60581-1-9

5.2. User Interface Administrator

Pada halaman sebagai admin disediakan beberapa fitur atau fungsi diantaranta: 1) Pembuatan pengguna baru; 2) Penambahan server yang dikelola; 3) Penambahan aktivitas yang dilakukan oleh administrator. Pada Gambar 6 tampak antar muka aplikasi untuk akun admin.

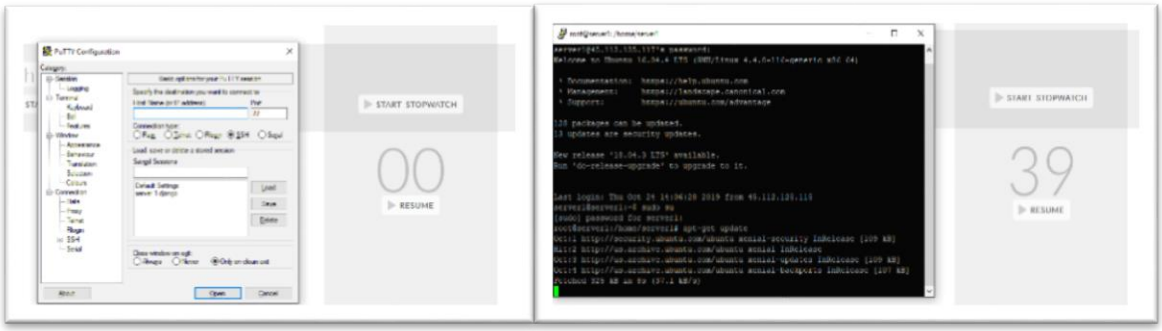


Gambar 6. Halaman admin.

5.3. Pengujian Sistem

Pada proses pengujian sistem dilakukan 2 (dua) skenario, skenario pertama aktivitas remote server menggunakan aplikasi Putty SSH dan aplikasi pengukur waktu secara online [8]. Pengujian dilakukan secara manual pada satu server dengan diawali penggunaan aplikasi Putty SSH. Kemudian di-*input* perintah IP: 45.112.125.117 port 889, kemudian klik Open**.** Setelah masuk ke command putty, login dengan mengetik username dan password linux. Kemudian login dan ketikkan perintah, sudo apt-get update setelah itu ketik password sudo, dan terakhir *Stopwatch* dihentikan.

Berdasarkan proses pengujian yang telah dilakukan, maka diperoleh waktu yang diperlukan mulai dari membuka aplikasi Putty sampai melakukan update sistem operasi yaitu 39 detik. Pada Gambar 7, ditunjukkan hasil peroleh durasi waktu dari aplikasi timer-online.

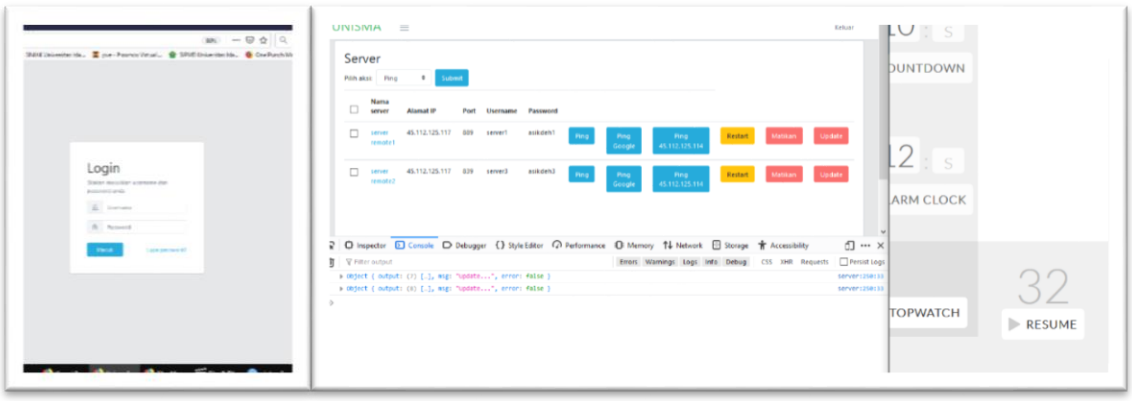


Gambar 7. Proses pengujian sistem secara manual.

ISBN: 978-602-60581-1-9

Pada skenario kedua, telah dilakukan proses pengujian menggunakan aplikasi manajemen server mulai dari kondisi telah tampil halaman login, lalu melakukan aktivitas yang bertujuan untuk update sistem, dengan tahapan antara lain: 1) Login menggunakan akun admin; 2) Setelah tampil halaman daftar server, klik tombol update untuk server yang akan di update, disini kami menguji dua server sekaligus untuk di-*update*; 3) Setelah status sukses (tertulis error: false) maka *stopwatch* dihentikan.

Berdasarkan hasil pengukuran yang diperoleh dari aplikasi *stopwatch* menunjukkan angka 32 detik untuk mengupdate dua server secara bersamaan seperti tampak pada Gambar 8.



Gambar 8. Proses pengujian sistem menggunakan aplikasi.

5.4. Analisa Sistem

Pengelolaan pada server yang dilakukan menggunakan aplikasi yang dikembangkan menggunakan *framework* Django dapat bekerja secara otomatis dan terintegrasi. *Effort* yang dilakukan oleh seorang pengelola server dapat lebih efektif, karena hanya memerlukan satu *interface* untuk melakukan remote SSH dan aktivitas pengelolaan pada beberapa server. Keuntungan lainnya, durasi waktu yang diperlukan lebih cepat sehingga administrator tidak memerlukan waktu yang lebih lama untuk melakukan pengelolaan layanan pada beberapa server. Berdasarkan proses pengujian yang dilakukan, pengelolaan server yang dilakukan secara manual memerlukan waktu sekitar 39 detik untuk melakukan proses update sistem pada 1 (satu) unit server mulai dari proses awal sampai selesai. Sementara dengan menggunakan aplikasi manajemen server, hanya memerlukan durasi wakti sekitar 32 detik pada 2 (dua) unit server yang berbeda. Pengelolaan dengan menggunakan aplikasi manajemen multi server diperoleh nilai efisiensi waktu sebesar 50%.

6. Simpulan dan Saran

Aplikasi manajemen multi-server yang telah dikembangkan dapat bekerja secara otomatis dan terintegrasi pada satu *interface* sehingga mampu menghemat *effort* dan durasi waktu yang diperlukan oleh seorang administrator dalam pengelolaan multi-server. Pada proses update sistem, pengelolaan berbasis aplikasi manajemen multi-server mampu menghemat waktu sekitar 50% jika dibandingkan dengan waktu pengelolaan pada server yang dilakukan secara manual.

Saran untuk pengembangan penelitian kedepannya antara lain perlu penambahan fitur *report* untuk administrator dan tersedianya aplikasi berbasis *mobile*.

Ucapan Terima Kasih

Terima kasih kami sampaikan kepada Kementrian Riset Teknologi dan Pendidikan Tinggi (Kemenristek DIKTI) yang telah mendukung dan mendanai penelitian kami melalui program

hibab penelitian dosen pemula (PDP) tahun 2019, sehingga penelitian ini dapat kami selesaikan dan mendapatkan luaran sesuai dengan yang diharapkan. Ucapan terima kasih juga kami sampaikan kepada Lembaga Penelitian dan Pengabdian Masyarakat (LPPM) Universitas Islam 45 Bekasi yang telah memberikan dukungan dan memfasilitasi setiap keperluan mulai penyusunan proposal dan pembuatan laporan akhir.

DAFTAR PUSTAKA

[1] N. Aliya, "Pengertian Server, Fungsi Server Beserta Cara Kerja dan Jenis-jenis Server." Oct, 2018. [Online]. Available: https://www.nesabamedia.com/pengertian-server-dan-fungsi-server/. [Accessed June, 20, 2019].

[2] Afriandi, A. (2012). Perancangan, Implementasi, dan Analisis Kinerja Virtualisasi Server Menggunakan Proxmox, VMwrae ESX, dan Openstack. *Jurnal Teknologi Informasi UGM*.

[3] Widarma, A., & Siregar, Y. H. (2019). Analisis Kinerja Teknologi Virtualisasi Server (Studi Kasus: Universitas Asahan). *Prosiding Seminar Nasional Multidisiplin* (pp. 688-698). Kisaran: Universitas Asahan.

[4] Irwan, D. (2017). Service Availability dan Performa Sumber Daya Prosessor Pada Infrastruktur Server Virtual. *Jurnal Penelitian Ilmu Komputer System Embedded dan Logic*, 42-45.

[5] Novianti, A. G., Niswar, M., & Ilham, A. A. (2011). *Aplikasi Berbasis Smartphone untuk Manajemen Server Pada Private Cloud Computing.* Papua: Fakultas Ilmu Komputer dan Manajemen, Universitas Sains dan Teknologi Jayapura.

[6] Harfadzi, & Irwan, D. (2016). Perancangan dan Implementasi Virtualisasi Server Menggunakan Proxmox VE 3.4. *Jurnal Piksel*, 89-97.

[7] Andi. (2017, Juli 10). *Post top List*. Retrieved from Ubuntu Indonesia: https://ubuntu-indonesia.com/toplist?sid=6def770bbcff43fc920d87d3cc5bad70.

[8] Online Timer, *Stopwactch.* http://www.timer-tab.com, 2019.

ISBN: 978-602-60581-1-9