**Sistem Manajemen Layanan Web Berbasis Platform as a Service (PaaS) dengan API Openstack**

Ida Bagus Rathu Eka Surya Wibawa1, I Dewa Made Bayu Atmaja Darmawan 2, I Komang Ari Mogi 3

123 Program Studi Tekhnik Informatika, Fakultas Matematika dan Ilmu Pengetahuan Alam, Universitas Udayana

Jl. Kampus Bukit Jimbaran (80361), Badung, Bali, Indonesia

1guzekawibawa@gmail.com

2 dewabayu@cs.unud.ac.id

3 arimogi@cs.unud.ac.id

***Abstrak***

*Cloud computing merupakan teknologi yang saat ini mulai berkembang dalam banyak aktivitas teknologi informasi. Cloud computing merupakan model komputasi yang semua sumber daya yang ada dalam layanan cloud dijalankan dengan media jaringan internet. Dengan adanya cloud computing memudahkan para pengguna dalam melakukan komputasi tanpa harus melakukan instalasi aplikasi pada komputer, pengguna hanya perlu mengaksesnya melalui internet. Cloud computing memiliki beberapa fasilitas yang dapat dipilih oleh pengguna sesuai kebutuhan pengguna seperti Infrastructure as a Service(IaaS), Platform as a Service(PaaS). Serta Software as a Service(SaaS).*

*Pengembangan Platform as a Service sebagai salah satu teknologi cloud computing yang dapat digunakan oleh pengembang aplikasi untuk mengembangkan aplikasi yang akan dibuat tanpa perlu menyediakan infrastruktur, database, framework aplikasi dan lain sebagainya serta bersifat dinamis. Dalam pengembangan layanan cloud yang dikelola oleh seorang sistem administrator atau sysadmin, tugas menginstalasi dan menkonfigurasi sistem pada server maupun software aplikasi dilakukan dengan otomatis dengan menggunakan platform otomatisasi sistem linux untuk efisiensi waktu dan memanagemen penggunanya dengan lebih mudah.*

***Kata kunci : Cloud Computing, Platform as a Service, automation, ansible, API, Openstack.***

1. **Pendahuluan**

Cloud computing merupakan teknologi yang saat ini mulai berkembang dalam banyak aktivitas teknologi informasi. Cloud computing merupakan model komputasi yang semua sumber daya yang ada dalam layanan cloud dijalankan dengan media jaringan internet. Dengan adanya cloud computing memudahkan para pengguna dalam melakukan komputasi tanpa harus melakukan instalasi aplikasi pada komputer, pengguna hanya perlu mengaksesnya melalui internet. Cloud computing memiliki beberapa fasilitas yang dapat dipilih oleh pengguna sesuai kebutuhan pengguna seperti Infrastructure as a Service(IaaS) sebagai penyedia infrastruktur pada pengguna, Platform as a Service (PaaS) yang dapat digunakan oleh pengembang aplikasi untuk mengembangkan aplikasi yang akan dibuat tanpa perlu menyediakan infrastruktur, database, framework aplikasi dan lain sebagainya. Serta Software as a Service(SaaS) yang memberikan software yang siap digunakan oleh pengguna.

Dalam layanan cloud dikelola oleh seorang sistem administrator atau sysadmin yang bertugas menginstalasi dan menkonfigurasi server, menginstall dan mengkonfigurasi software aplikasi, membuat dan mengelola user, backup dan restore file, konfigurasi keamanan server, serta memonitor keamanan jaringan agar layanan yang dibutuhkan dapat berjalan dengan baik. Permasalahan yang muncul adalah untuk melakukan hal tersebut sysadmin harus mengkonfirmasi kebutuhan pengguna, serta untuk melakukan instalasi dan konfigurasi pada virtual server membutuhkan waktu yang cukup lama.

Dari masalah tersebut penelitian ini diharapkan dapat mempermudah tugas – tugas dari sistem administrator dengan sistem manajemen berbasis cloud yang dibangun menggunakan API dari openstack dan flask python, sistem administrator dapat mendukung kinerja seorang sistem administrator mengelola setiap virtual server yang dimiliki. Serta melakukan instalasi dan konfigurasi secara otomatis bila ada pengguna baru yang membutuhkan sebuah server virtual untuk website.

1. **Metode Penelitian**
2. **Analisis Kebutuhan**

Pada tahap ini dilakukan analisis kebutuhan sistem, analisis kebutuhan sistem meliputi data yang digunakan, pembelajaran dari referensi yang sudah ada dan perangkat yang digunakan baik perangkat lunak maupun perangkat keras:

Tahapan pertama yang dilakukan dalam penelitian ini adalah mengidentifikasi permasalahan. Tahap ini merupakan tahap yang paling penting dalam penelitian karena jalannya penelitian didasarkan atas permasalahan yang terjadi. Setelah menentukan masalah yang terjadi, tahapan yang diperlukan selanjutnya adalah menentukan rumusan masalah dan tujuan yang ingin dicapai dalam penenlitian. Pada penelitian ini identifikasi permasalahan dilakukan dengan menggunakan teknik observasi, dari teknik ini maka akan dapat diketahui mengenai keluhan – keluhan yang ada di lapangan.

Tahap kedua yang dilakukan dalam metodelogi penelitian ini adalah studi literatur. Studi literatur dilakukan dengan mengambil literatur – literatur pendukung dari jurnal – jurnal ilmiah, baik jurnal dalam negeri ataupun jurnal luar negeri dan dari beberapa buku. Dalam studi literatur ini, penulis mencari sumber terkait permasalahan – permasalahan yang perlu menjadi perbaikan dalam penelitian selanjutnya..

1. **Arsitektur Kerja Sistem**

Alur kerja web dari sisi front-side atau pada sisi user. Setelah user melakukan registrasi pada website, selanjutnya user akan diarahkan ke menu instalasi dan konfigurasi dalam membangun layanan VPS. Menu akan dibuat secara user friendly sehingga user hanya perlu memilih menu konfigurasi system. Ketika user sudah selesai memilih menu konfigurasi, system akan memvalidasi informasi yang dimasukkan oleh user, bila sudah benar system akan mengirimkan hasil instalasi dan konfigurasi dalam sebuah data berrntuk JSON ke dalam system yang nantinya akan dikirim ke virtual sever untuk melakukan konfigurasi dan instalasi modul kebutuhan aplikasi selanjutnya sistem akan melakukan backup berkala untuk menjaga keamanan data pengguna bila tejadi satu kesalahan dapat dilakukan restore data.



Gambar 1. Rancangan Implementasi Sistem

1. **Alur Proses**

Menjelaskan alur kerja sistem back-end sistem. Tahap pertama yang harus dilakukan adalah melakukan komunikasi dengan openstack agar dapat menggunakan API dari openstack. Melakukan komunikasi dengan melakukan request token authentikasi, hal ini dimaksudkan agar setiap komunikasi yang dibuat dapat melakukan update data pada API dan yang memiliki token dari autentikasi saja yang dapat berkomunikasi. Selanjutnya setelah melakukan request, sistem tetap akan melakukan validasi token id dari openstack, dikarena setiap melakukan request token memiliki batas waktu yang ditentukan. Setelah bisa melakukan komunikasi dengan openstack selanjutnya menerima masukan konfigurasi dari front-end dan sudah divalidasi oleh administrator, Setelah semua dicek dan sukses maka selanjutnya sistem akan menampilkan virtual mesin siap dan sebuah ip public untuk diakses oleh developer dan ssh key bila developer memiliki kebutuhan aplikasi khusus yang tidak dimiliki oleh sistem..

1. **Desain Antarmuka Sistem**



Gambar 2. Tampilan Menu Utama

Pengguna baru harus melakukan registrasi untuk dapat masuk ke dalam sistem, Setelah melakukan pendaftaran, pengguna akan diarahkan ke form login, selanjutnya pengguna dapat memasukan username dan password yang sudah di buat pada form registrasi.



Gambar 3. Tampilan Menu Buat Server

Pengguna memasukan nama project. Nama project nantinya akan menjadi nama instance baru sebagai hostname, untuk deskripsi bersifat opsional, dapat ditambahkan keterangan ataupun tidak. Lalu pilih siste operasi yang akan digunakan sebagai server pada lingkungan aplikasi yang akan dibuat. Lalu pilih CPU, CPU disini berisi berapa core yang di butuhkan untuk lingkungan sistem yang akan dibangun oleh pengguna, seberapa besar penyimpanan dan memori RAM yang dibutuhkan untuk mengembangkan sistem yang akan dibuat.



Gambar 4. Tampilan Menu List Server

Pada Menu ini sistem hanya akan membuat instance baru dan menginstall sistem operasi saja, bila pengguna ingin menginstall paket XAMPP / LAMP, pengguna hanya perlu memberi centang pada menu xampp dan selanjutnya pilih menu install. Maka sistem akan melakukan instalasi XAMPP pada instance server pengguna secara otomatis dengan menggunakan ansible.



Gambar 5. Tampilan Apache2 pada Virtual Server

Menguji webserver yang telah kita buat dapat mengakses ip address yang di miliki instance melalui web browser, bila muncul seperti gambar di atas web server yang di buat dinyatakan berhasil melakukan instalasi.



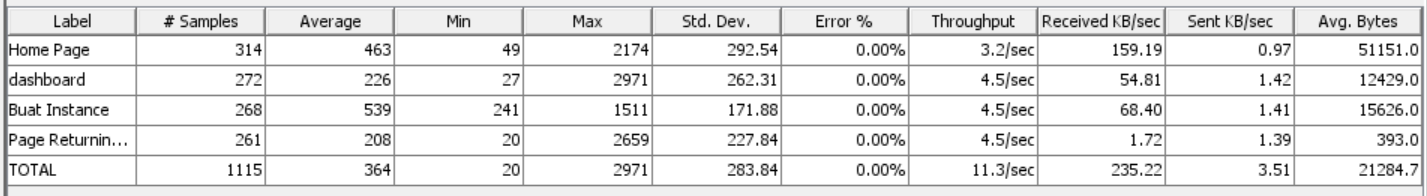
Gambar 6. Tampilan MySQL pada Virtual Server

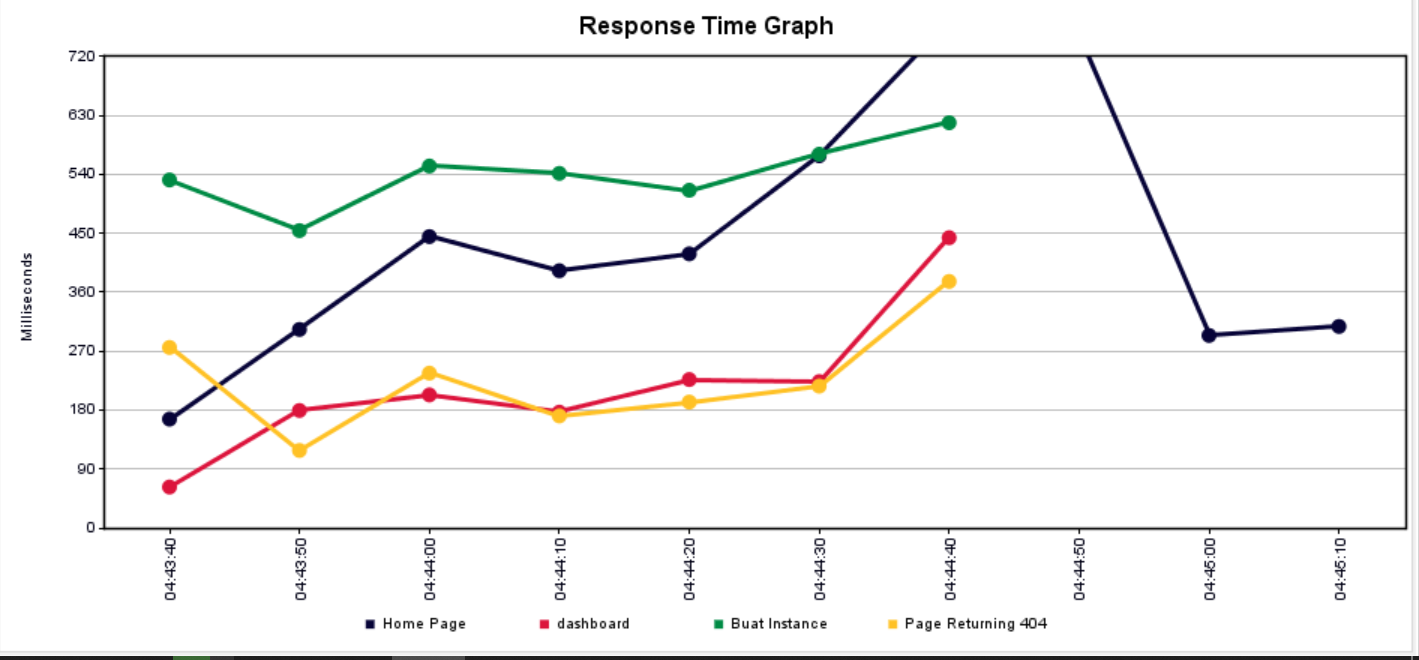
Tampilan dari database yang sudah terinstall pada instance, pengguna dapat melakukan pengecekan kembali dengan meremote instance server yangs sudah dibuat dengan menggunakan SSH

1. **Hasil dan Diskusi**

Pada penelitian ini pengujian dilakukan dengan menguji performa server yang digunakan untuk mengelola virtual server pengguna dengan hasil pengujian sebagai berikut.

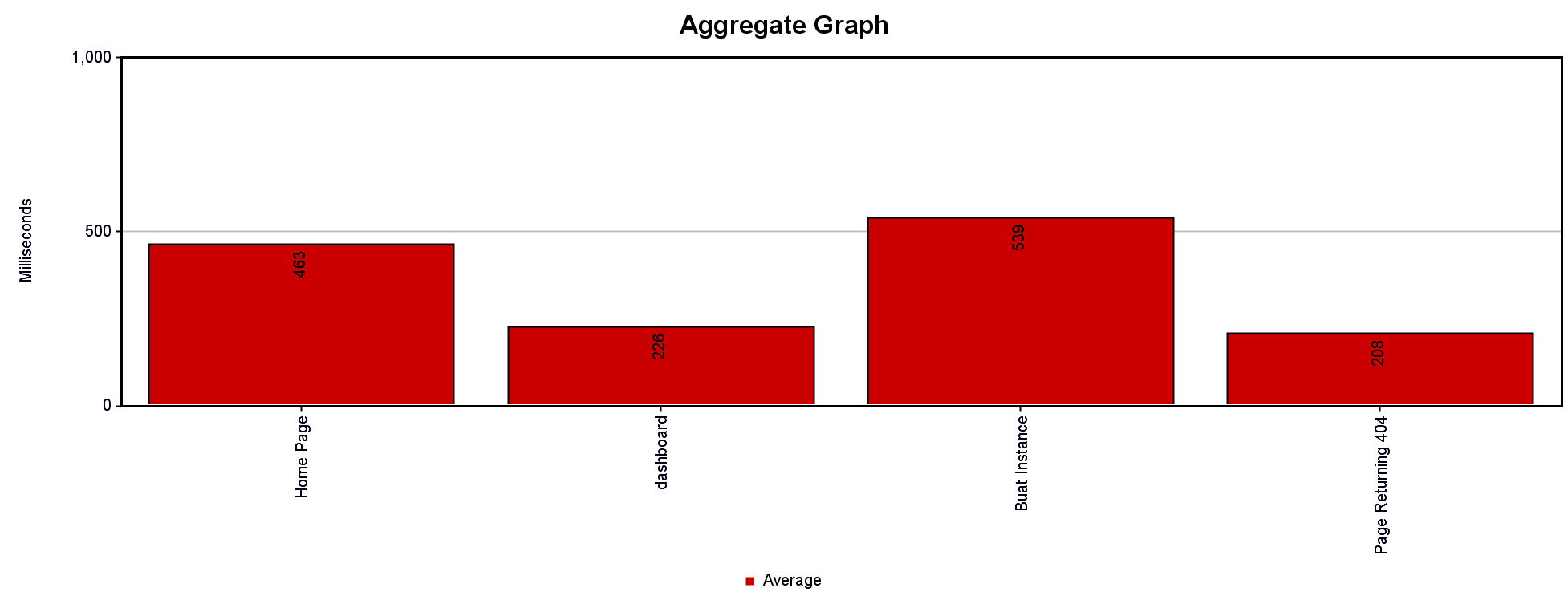
Tabel 1. Tabel Hasil Pengujian





Gambar 7. Grafik Waktu Respon

Pada gambar 7. Merupakan hasil dari pengujian waktu respon web server, dapat dilihat pada gambar waktu yang dibutuhkan pada pembuatan instance cukup lebih lama dibandingkan dalam mengakses alamat home page, dashboard atau page error, dikarenakan dalam alamat buat instance ini melakukan autentikasi pada server openstack. Sehingga membuat waktu respon yang dibutuhkan dalam menampilkan halaman membuat sebuah instance baru menjadi sedikit lebih lama melebihi 720 milliseconds dalam suatu waktu.



Gambar 8. Grafik Agregat

Pada gambar aggregate graph menunjukkan hasil dari rata – rata waktu yang dibutuhkan dalam mengakses setiap halaman yang di uji. Dari hasil yang ditampilkan rata – rata waktu dalam membuat sebuah instance mendapatkan waktu rata – rata yang cukup tinggi melebihi 500 Milliseconds dikarenakan setiap kali mengakses halaman buat baru, sistem akan selalu melakukan autentikasi API dan melakukan pengecekan menjadikan pada halaman buat instance menjadi sedikit lebih lama.

1. **Kesimpulan**

Berdasarkan penelitian yang telah dilakukan, dapat diambil beberapa kesimpulan sebagai berikut :

1. Sistem yang dibangun dapat diimplementasikan dengan Sistem Cloud computing secara otomatisasi dengan menggunakan ansible dengan prinsip kerja sistem membuat virtual server secara otomatis dan melakukan instalasi paket LAMP secara otomatis.
2. Berdasarkan hasil pengujian yang dilakukan didapatkan hasil bahwa membangun sebuah server dengan didukung oleh teknologi otomatisasi membuat waktu yang dibutuhkan lebih efisien dalam membuat sebuah server yang siap digunakan serta secara otomasis terinstal web server beserta sebuah database.
3. Dari hasil pengujian performa mendapatkan hasil yang sangat baik, server berjalan dengan baik ketika dikases oleh 100 pengguna per detiknya. Namun terjadi sedikit penurunan performa per 200 milliseconds ketika web managemen akan melakukan pengambilan data pada server openstack, tetapi tidak mempengaruhi performa server secara keseluruhan
4. **Saran**
5. Dikarenakan sistem ini berbasis cloud computing ada baiknya bila dikembangkan lebih lanjut dapat menambahkan fitur keamanan data pengguna, serta dapat dikembangkan secara dinamis untuk kebuhan penyimpanan, jaringan, maupun untuk komputasinya.
6. Dapat menambahkan monitoring instance untuk dapat melihat kesehatan setiap instance / virtual server untuk pengguna. Jadi pengguna pun dapat memantau keadaan virtual servernya sendiri. Hal tersebut dapat menjadi pertimbangan untuk penggunanya.

**Referensi**

[1]. Xia, Q., Lan, Y., & Xiao, L. (2015). A Heuristic Adaptive Threshold Algorithm on IaaS Clouds.

[2]. Anggeriana, H. (2011). *Cloud Computing.*

[2]. Ansible. (t.thn.). *ansible.com*. Diambil kembali dari http://docs.ansible.com/ansible/latest/index.html: http://docs.ansible.com/ansible/latest/index.html

[3]. Collings, T., & Kurt , W. (2015). *Duties of the System Administrator. In Red Hat Linux Networking System Administrator (chap. 1).*

[4]. Corporation, Exabyte. (2004). *The Basic Backup Guide.* cororado.

[5]. Doty, S. (2008). *Python Basics.*

[6]. Gerald D. Everett, R. M. (2007). *Software Testing: Testing Across the Entire Software Development Life Cycle 1st Edition.* Canada: IEEE Press.

[7]. Lei Xiaojiang, S. Y. (2013). The Design and Implementation of Resource Monitoring for Cloud Computing Service Platform.

[8]. Liyun Zuo, L. S. (2015). A Dynamic Self-adaptive Resource-Load Evaluation Method in Cloud Computing.

[9]. Masse, M. (2012). REST API. Dalam *REST API Design Rulebook* (hal. 5). America: O’Reilly Media, Inc.

[10]. Mulyana, E. (2017, 10 05). *Pengantar Openstack*. Diambil kembali dari https://eueung.gitbooks.io: https://eueung.gitbooks.io/buku-komunitas-sdn-rg/content/index.html

[11]. Nicole Ng, H. C. (2011). An Adaptive Threshold Method to Address Routing Issues in Delay-Tolerant Networks.

[12]. Saleh, Y. W. (2013). Adaptive Resource Management for Service Workflows in Cloud Environments.

[13]. Sosinsky, B. (2011). *Cloud Computing Bible.* Canada: Wiley Publishing, Inc.