

PENGEMBANGAN SISTEM AUTOMASI *DEPLOYING WEBSITE* ABSENSI DI KISEL INDONESIA

Natasya Salsabila¹⁾, Asri Wulandari²⁾, Reza Yusuf Merdekantara³⁾, R.Tubagus Khaidir Akbar⁴⁾

¹ Jurusan Teknik Elektro, Politeknik Negeri Jakarta, Depok, 16425

² Jurusan Teknik Elektro, Politeknik Negeri Jakarta, Depok, 16425

³ Koperasi Telekomunikasi Selular, Jakarta, 12310

⁴ Koperasi Telekomunikasi Selular, Jakarta, 12310

E-mail: natasyasalsabila19@gmail.com

Abstract

The increasing development of Information Technology makes companies need to optimize their performance. In its implementation, the website deployment process uses a manual method, so it takes a long time, and there is no website code version information which makes it difficult to find out the website version. To overcome this problem, an Automation System for deploying the website was created to speed up the Deployment process, Scalable Service, minimize errors, and develop monitoring of website code changes that will be used in the website presence deployment process. The results of testing the Automation System Deploying website with the standards of The Five Pillars of the Well-Architected Framework on the Operational Excellence aspect prove that the system is running well. The reliability aspect shows that the Website Deployment Automation System is reliable. In the aspect of Performance efficiency, it shows that Automation System for deploying the website is more efficient than manual Deployment websites and the system can be used to simplify the deployment process when code changes occur and create code documentation versions and the status of the website deployment process.

Keywords: *Automation System, Pipeline, Deploying Website, AWS*

PENDAHULUAN

Deploying merupakan proses mengunggah sebuah aplikasi agar aplikasi tersebut dapat diakses oleh publik. *Deploying* ini sangat dibutuhkan oleh programmer untuk meluncurkan aplikasi yang telah dibuat sebelumnya. Seperti disebutkan oleh (Kurniawan et al., 2019) *Deployment* berarti tahap untuk membuat sebuah model implementasi dalam sebuah *tools* yang dapat dibangun dengan berbagai jenis pemograman.

Namun pada implementasinya, sering kali terjadi kesalahan saat melakukan proses *deploying* ini. Kesalahan yang terjadi saat melakukan proses *deploying* secara manual seperti kesalahan dalam mengkonfigurasi *environment* yang dibutuhkan, kesalahan dalam menginstall *web server* dan *database*, kesalahan dalam mengkonfigurasi *frontend* dan *backend*, dan masih banyak kesalahan lain.

Bedasarkan paparan di atas, diperlukan sistem automasi *deploying* agar kesalahan dapat diminimalkan dan efisiensi waktu dapat dicapai. Menurut Humble dan Farley proses automasi *deploying* atau yang biasa disebut pipeline adalah proses perubahan pada sumber kode yang dapat dilihat oleh *end-users* atau dengan kata lain sebuah proses yang terdapat kontrol versi pada kode aplikasi atau *website* disebut *production environment* (Mohammad, 2018).

Pipeline merupakan alat yang digunakan untuk mengotomatisasi tugas dalam proses *continuous integration and delivery (CI/CD)* dari suatu aplikasi. *Pipeline* dapat dibuat sesuai dengan kebutuhan untuk menjadi sebuah sistem automasi (Farid & Anugrah, 2021). Dengan adanya *pipeline* ini langkah-langkah seperti mengkonfigurasi *environment*, menginstall *web server* dan *database*, mengkonfigurasi *frontend* dan *backend* tidak lagi dilakukan oleh programmer sehingga dapat meminimalisir kesalahan ketika langkah-langkah tersebut dilakukan secara manual. Selain itu, *deploying website* menjadi lebih cepat karena menggunakan sistem automasi tanpa campur tangan manusia.

Tujuan dari penulisan ini adalah merancang dan merealisasikan sistem automasi *deploying website* absensi, serta menganalisis hasil *deployment website* absensi terhadap *The Five Pillars of Well-Architected Framework AWS* yang diharapkan dapat membantu para staf untuk mempermudah dan mempercepat proses *deployment*.

METODE PENELITIAN

Sistem automasi *deploying website* terbagi menjadi dua *tools* yang dibutuhkan untuk melakukan pengambilan analisis, yaitu *website absensi* dan *pipeline*. *Website absensi* digunakan sebagai contoh produk yang digunakan untuk melakukan proses *deployment* menggunakan sistem automasi *deploying*, berikut adalah diagram alir dari *website absensi* yang digunakan:

Sistem automasi *deploying website* ini diuji dengan tiga pillars pada *The Five Pillars of Well-Architected Framework AWS* (Raza et al., 2021) yaitu *Operational Excellence pillars*, *Reliability Pillars* dan *Performance Efficiency Pillars* serta uji test case pembaharuan *code website*.

HASIL DAN PEMBAHASAN

Berdasarkan metode penelitian, pengujian dilakukan untuk memverifikasi keberhasilan sistem automasi *deploying website*. Secara lengkap hasil pengujian tersebut dapat dijelaskan sebagai berikut.

A. Pengujian karakteristik aspek *Operational Excellence*

Pengujian aspek *Operational Excellence* dilakukan untuk mengetahui sistem automasi *deploying website* dapat menyediakan fungsionalitas yang dibutuhkan oleh pengguna lalu dilakukan perhitungan persentase keberhasilan sistem. Pengujian pada karakteristik aspek *Operational Excellence* dilakukan dengan menggunakan pengujian penilaian ahli (*expert judgment*) dan alpha test (Ter Berg et al., 2019). Hasil dari pengujian ditunjukkan pada Tabel 1.

Table 1 Hasil Ketercapaian Uji Karakteristik Aspek *Operational Excellence*

Ketercapaian			
<i>Expert Judgment</i>		<i>Alpha Test</i>	
Ya	Tidak	Ya	Tidak
6	0	6	0

Persentase Keberhasilan dari hasil uji karakteristik aspek *Operational Excellence* dapat dihitung dengan rumus sebagai berikut:

$$\begin{aligned} \text{Presentase Keberhasilan (\%)} &= \frac{i}{r} \times 100\% \\ &= \frac{6}{6} \times 100\% = 100\% \end{aligned}$$

Dari hasil perhitungan persentase keberhasilan didapatkan hasil yaitu 100%. Hal tersebut menunjukkan bahwa semua langkah dan aktifitas yang terdapat pada Sistem Automasi *Deploying website* dapat berjalan dengan baik. Maka Sistem Automasi *Deploying website* terbukti memenuhi standar aspek *Operational Excellence*.

B. Pengujian karakteristik aspek *Reliability*

Pengujian aspek *reliability* dilakukan untuk menguji bagaimana kemampuan sistem untuk pulih dari gangguan, infrastruktur dapat memperoleh sumber daya komputasi secara dinamis untuk memenuhi permintaan, dan mengurangi gangguan seperti kesalahan konfigurasi. Pengujian aspek *reliability* ini dilakukan dengan menjalankan sistem automasi *deploying website* pada *Bitbucket* dengan sistem operasi *Ubuntu 18.04* dan *NodeJS* sebagai Bahasa pemograman yang digunakan. Hasil dari pengujian ditunjukkan pada Tabel 2

Table 2 Hasil Ketercapaian Uji Karakteristik Aspek Reliability

Ketercapaian	
Ya	Tidak
5	0

Hasil ketercapaian yang didapatkan pada hasil uji karakteristik aspek *Reliability* menggunakan metode alpha test (Masripah & Ramayanti, 2020) yaitu berjumlah 5 *test case* yang tercapai dan tidak ada *test case* yang tidak tercapai. Presentase Keberhasilan dari hasil uji karakteristik aspek *Reliability* dapat dihitung dengan rumus sebagai berikut:

$$\begin{aligned} \text{Presentase Keberhasilan (\%)} &= \frac{i}{r} \times 100\% \\ &= \frac{5}{5} \times 100\% = 100\% \end{aligned}$$

Dari hasil perhitungan presentase kelayakan didapatkan hasil yaitu 100%. Hal ini menyatakan bahwa sistem automasi *deploying website* terbukti handal.

C. Pengujian karakteristik aspek *Performance efficiency*

Pengujian karakteristik aspek *Performance efficiency* dilakukan untuk mengetahui penggunaan sumber daya komputasi yang efisien. Pengujian karakteristik aspek *performance efficiency* dilakukan dengan membandingkan cara kerja serta infrastruktur antara sistem automasi *deploying website* absensi dengan *deploying website* absensi secara manual di *AWS Instance*. Hasil dari pengujian ditunjukkan pada Tabel 3

Table 3 Perbandingan Proses Deploying website

Aksi	Langkah yang dilakukan	Waktu yang dibutuhkan
Manual	10 Langkah	85 Menit
Otomatis	1 Langkah	20 Menit

Dari tabel diatas dapat disimpulkan bahwa proses *deploying website* dengan cara manual dilakukan dengan 10 langkah dan menghabiskan waktu 85 menit untuk *deploy* satu *website* sedangkan proses *deploying website* dengan cara otomatis dilakukan dengan 1 langkah dan menghabiskan waktu 20 menit untuk *deploy* satu *website*. Hal ini menyatakan bahwa Sistem Automasi *Deploying website* terbukti lebih efisien baik dari segi aktifitas yang harus dilakukan untuk *deploying website* maupun dari segi waktu yang digunakan.

D. Pengujian *Deploying website* dengan pembaharuan *code website*

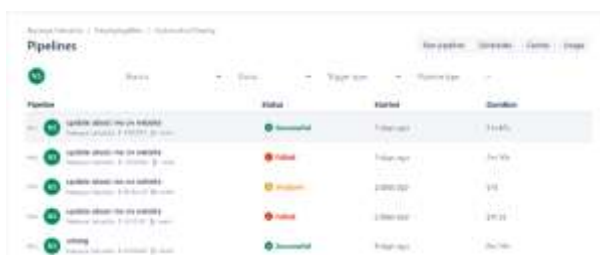
Pengujian *Deploying website* dengan pembaharuan *code website* dilakukan dengan *deploying code* website yang telah di perbaharui menggunakan sistem automasi *deploying website*, lalu dilakukan pengamatan apakah fitur baru website dapat diakses oleh publik.

Penambahan *script* yang dilakukan adalah folder *About* yang berisi *code* fitur about me, setelah *code* website berjalan dengan baik di local server, dilakukan push *code* dari *local server* ke *bitbucket pipeline* untuk melakukan proses *deploying* secara otomatis menggunakan sistem automasi *deploying website*. Hasil *deploying website* dengan fitur baru sebagai berikut:



Gambar 3 About me website pada Instance AWS

Website berhasil di deploy pada *Instance* AWS dan dapat diakses oleh semua pengguna menggunakan IP *Instance* yang tertera di *Pipeline* *Bitbucket* ataupun Console Manager AWS. Proses ini juga memiliki *monitoring changes* atau dokumentasi perubahan *code* serta perbedaan versi proses *deploying* menggunakan *pipeline* *bitbucket* yang berfungsi untuk mengetahui dan mengontrol versi *code website* yang akan di *deploy* untuk kebutuhan *production* (Light et al., 2021).



Pipeline	Status	Started	Duration
Deploy ke AWS S3	Completed	14:00:00	1:45
Deploy ke AWS S3	Failed	14:00:00	1:45
Deploy ke AWS S3	Completed	14:00:00	1:45
Deploy ke AWS S3	Failed	14:00:00	1:45
Deploy ke AWS S3	Completed	14:00:00	1:45

Gambar 4 Dokumentasi monitoring changes

Dari hasil proses diatas terbukti sistem automasi *deploying website* dapat dilakukan dengan adanya perubahan yang terjadi pada *website*. Hal ini dibuktikan oleh *website* yang memiliki fitur baru dan dapat di akses oleh seluruh pengguna. Dengan ini pengujian *deploying website* dengan pembaharuan *code website* berhasil dilakukan.

SIMPULAN

Hasil pengujian Sistem Automasi *Deploying website* dengan standar *The Five Pillars of the Well-Architected Framework* pada aspek *Operational Excellence* memperoleh nilai sebesar 100% terbukti bahwa memenuhi standar aspek *Operational Excellence*. Pada aspek *Reliability* memperoleh nilai sebesar 100% yang menandakan bahwa Sistem Automasi *Deploying Website* terbukti handal. Dan pada aspek *Performance efficiency* hanya melakukan satu langkah dengan waktu 20 menit yang menandakan bahwa Sistem Automasi *Deploying website* sangat efisien dibandingkan *Deploying website* secara manual. Sistem automasi *deploying website* juga dapat digunakan untuk mempermudah proses *deploying* saat terjadi perubahan *code* dan terciptanya dokumentasi versi *code* serta status proses *deploying website*.

DAFTAR PUSTAKA

- Farid, A., & Anugrah, I. G. (2021). Implementasi CI/CD Pipeline Pada Framework Androbase Dengan Menggunakan Jenkins (Studi Kasus: PT. Andromedia). *Jurnal Nasional Komputasi Dan Teknologi Informasi (JNKTI)*, 4(6), 522–527. <https://doi.org/10.32672/jnkti.v4i6.3703>
- Kurniawan, S., Gata, W., Puspitawati, D. A., -, N., Tabrani, M., & Novel, K. (2019). Perbandingan Metode Klasifikasi Analisis Sentimen Tokoh Politik Pada Komentar Media Berita Online. *Jurnal RESTI (Rekayasa Sistem Dan Teknologi Informasi)*, 3(2), 176–183. <https://doi.org/10.29207/resti.v3i2.935>
- Light, J., Pfeiffer, P., & Bennett, B. (2021). An evaluation of continuous integration and delivery frameworks for classroom use. *Proceedings of the 2021 ACMSE Conference - ACMSE 2021: The Annual ACM Southeast Conference*, 204–208. <https://doi.org/10.1145/3409334.3452085>
- Masripah, S., & Ramayanti, L. (2020). Penerapan Pengujian Alpha Dan Beta Pada Aplikasi Penerimaan Siswa Baru. *Swabumi*, 8(1), 100–105. <https://doi.org/10.31294/swabumi.v8i1.7448>
- Mohammed. (2018). *Streamlining DevOps automation for Cloud applications*. 6(4).
- Raza, M., Hussain, F. K., Hussain, O. K., Rehman, Z. ur, & Zhao, M. (2021). Imputing sentiment intensity for SaaS service quality aspects using T-nearest neighbors with correlation-weighted Euclidean distance. *Knowledge and Information Systems*, 63(9), 2541–2584. <https://doi.org/10.1007/s10115-021-01591-3>
- Ter Berg, C. J. A., Leontaris, G., van den Boomen, M., Spaan, M. T. J., & Wolfert, A. R. M. (2019). Expert judgement based maintenance decision support method for structures with a long service-life. *Structure and Infrastructure Engineering*, 15(4), 492–503. <https://doi.org/10.1080/15732479.2018.1558270>