INTISARI

PERANCANGAN SISTEM *CONTINUOUS INTEGRATION* DAN *CONTINUOUS DEPLOYMENT* PADA INSTANCE EC2 AWS MENGGUNAKAN DOCKER DAN GITLAB UNTUK APLIKASI BERBASIS WEB HK ENDURANCE DI PT HUTAMA KARYA

Aga Riskika
18/425688/SV/14830

PT Hutama Karya adalah Badan Usaha Milik Negara (BUMN) yang bergerak di bidang konstruksi, pengembangan, dan penyedia jasa jalan tol. Meskipun berfokus pada bidang konstruksi, perusahaan ini juga memiliki beberapa aplikasi yang dijalankan di server untuk memenuhi kebutuhan bisnis ataupun kegiatan yang sering diselenggarakan oleh perusahaan. Semua aplikasi yang dimiliki PT Hutama Karya dijalankan pada server fisik di gedung HK Tower dan masih menggunakan metode *deployment* secara manual sehingga apabila ada perubahan dari suatu aplikasi akan cukup memakan waktu dan bergantung kepada seseorang yang perlu melakukan proses yang sama secara manual dan berulang-ulang setiap ada perubahan kode pada aplikasi. Untuk itu perlu adanya sistem CI/CD dalam proses *deployment* agar aplikasi dapat dirilis dan diperbarui dengan cepat dan otomatis setiap kali ada perubahan kode di suatu *repository*. Sistem CI/CD yang diterapkan menggunakan *tools Docker* dan GitLab serta diterapkan pada *cloud environmet* EC2 AWS. Setiap ada perubahan kode di *repository* nantinya akan ada *trigger* terjadinya *build* aplikasi yang dilakukan oleh *Docke*r agar aplikasi tersebut dikemas menjadi sebuah *Docker Image* dan dilakukan *push* ke *container registry* milik GitLab, selanjutnya akan ada proses *deployment* di server *staging* dimana server tersebut nantinya akan melakukan *pull image* hasil *build image* dari proses atau *stage* sebelumnya dan menjalankannya sebagai *container* di server tersebut. Setelah *stage deployment* di server *staging* berhasil *stage* selanjutnya yang merupakan *stage* untuk *deploymen*t ke server *production*

akan diblok terlebih dahulu dan hanya akan dijalankan apabila aplikasi di server staging sudah berjalan dengan baik untuk menghindari adanya kesalahan atau *bug* yang mungkin ada pada aplikasi, sehingga server *production* akan jauh lebih aman dan terhindar dari kemungkinan adanya *bug*. Setelah aplikasi pada server *staging* dipastikan berjalan dengan baik selanjutnya *stage deployment production* bisa dijalankan dengan melakukan klik pada *stage* tersebut dan akan menjalankan rangkaian proses yang mirip dengan *deployment* di server *staging* tetapi perbedaannya *stage* terakhir ini dilakukan untuk *deployment* ke server *production* sehingga menjadi sebuah sistem *Multi-Stage Deployment*.

Kata kunci: CI/CD, *Docker*, *Docker Image*, GitLab, *Container*, *Container Registry*, *Multi-Stage Deployment*.

ABSTRACT

DESIGN OF MULTI-STAGE CONTINUOUS INTEGRATION AND CONTINUOUS DEPLOYMENT SYSTEM ON AWS EC2 INSTANCE USING DOCKER AND GITLAB CI FOR HK ENDURANCE WEB-BASED APPLICATION AT PT HUTAMA KARYA

PT Hutama Karya is a State-Owned Enterprise (BUMN) which is engaged in construction, development, and toll road service provider. Although focused on the construction sector, this company also has several applications that run on servers to meet business needs or activities that are often hosted by the company. All applications owned by PT Hutama Karya are run on a physical server in the HK Tower building and still use the manual application method so that there is a change from an application that will take time and depend on someone who needs to do the same process manually and over and over again every time. code changes to the application. For this reason, it is necessary to have a CI/CD system in the deployment process so that applications can be released quickly and automatically every time there is a code change in a repository. The CI/CD system implemented using Docker and GitLab tools and applied to the AWS EC2 cloud environment. Every time there is a code change in the repository, it will trigger an application build by Docker so that the application is packaged into a Docker Image and pushed to the GitLab registry container, then there will be a deployment process on the staging server where the server will pull the build image. image of the previous process or stage and run it as a container on that server. After the deployment stage on the staging server is successful, the next stage which is the stage for deployment to the production server will be blocked first and will only be run if the application on the staging server is running well to avoid errors or bugs that may exist in the application, so that the production server will much safer and avoid the possibility of bugs. After the application on

the staging server is confirmed to be running properly, the next stage of production will be run by clicking on that stage and will run a series process similar to deployment on the staging server, but the difference is that this last stage is carry out for deployment to the production server into a multi-stage Deployment.

Keywords: CI/CD, Docker, Docker Image, GitLab, Container, Container Registry, Multi-Stage Deployment.