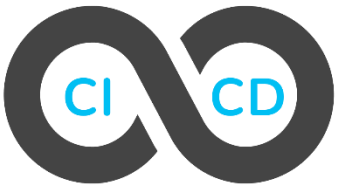
BAB 2
TEORI DAN TINJAUAN PUSTAKA

2.1 Praktik CI/CD

Praktik CI/CD merupakan upaya mengubah proses pengembangan website atau aplikasi yang tadinya masih manual menjadi otomatis, sehingga proses pengembangan dapat dilaksanakan dengan lebih cepat, tepat dan efisien. Secara lebih jelas praktik CI/CD merupakan pengembangan website atau aplikasi dengan mengotomatisasi setiap proses yang dilakukan. Tujuannya website atau aplikasi yang dihasilkan punya performa yang andal dan minim *bug*. Secara sederhana praktik CI/CD dilakukan otomatisasi berlangsung dari sejak penulisan *source code*, pengujian (*testing*), hingga produksi (*deployment*).



Gambar 2. 1 Logo CI/CD

Logo CI/CD biasanya digambarkan dengan *infinity*, seperti pada Gambar 2. 1 Logo CI/CD. Praktik CI/CD banyak menitikberatkan pada kolaborasi, maka hal ini menjadi bagian dari DevOps. *DevOps* merupakan upaya integrasi antara tim development (*Dev*) yang berisi para developer dan tim operations (*Ops*) yang dihuni oleh software engineer. Maka dari itu fungsi CI/CD di dalam *DevOps* adalah untuk dapat menjembatani aktivitas yang dilakukan tim *Devs* dan tim *Ops*. Sehingga praktik CI/CD ini dianggap sebagai tulang punggung dari *DevOps* modern, yang dikenal dengan CI/CD Pipeline. Praktik CI/CD terbagi menjadi 3 tahapan yaitu ada :

1. *Continuous Integration* (CI) :

   Proses penggabungan atau pengintegrasian kode yang telah dibuat oleh tim developer ke dalam repositori, selanjutnya *source code* yang ada pada repositori akan ditarik untuk dijalankan pengecekan adanya *bug* atau

masalah keteraturan *souce kode* secara otomatis dan berkelanjutan. Setelah lolos pengecekan *source code* akan di *build* menjadi sebuah file *image*.

2. *Continuous Delivery* (CD) :

   Proses lanjutan setelah file *image* berhasil terbuat adalah pengiriman file *image* kedalam sebuah *registry* atau penyimpanan (GHCR) secara otomatis. Didalam *registry* ini akan terdapat berbagai versi dari *image* dari yang paling lama hingga yang paling baru dengan segala perbaikan.

3. *Continuous Deployment* (CD) :

   Proses penarikan *image* dari *registry* ke dalam server. Setelah penarikan selesai dan berhasil, *image* tadi akan di jalankan dalam bentuk *Container* untuk dilakukan evaluasi kinerjanya. Dari hasil evaluasi inilah yang akan menentukan langkah selanjutnya, apakah akan dirilis atau akan ditahan terlebih dahulu untuk kembali dilakukan perbaikan dan penyesuaian.

Dalam praktik CI/CD terdapat 2 istilah yang sering muncul dan digunakan, yaitu ada *build* dan *test* :

1. *Build* :

   Build merupakan proses yang dilakukan untuk menghasilkan *image* dari website yang dapat disebut juga dengan *artefak*. Image ini yang akan dapat dieksekusi dari *source code* dalam repositori. Proses ini melibatkan kompilasi kode, pengemasan *dependensi*, dan pengintegrasian komponen-komponen yang diperlukan untuk membuat website atau aplikasi akhirnya dapat dijalankan didalam server untuk selanjutnya dilakukan evaluasi atau *test*.

2. *Test* :

   Test merupakan serangkaian proses yang dilakukan dan digunakan untuk memvalidasi website atau aplikasi secara otomatis setelah proses "*build*". Tujuan dari proses *test* ini adalah memastikan bahwa perubahan, perbaikan dan penyesuaian yang dilakukan pada *source code* tidak menambahkan atau bahkan malah memperparah *bug* yang dapat

menyebabkan kerusakan yang tidak diinginkan. Sehingga proses ini tentunya tidak boleh dilewatkan.

2.1.1 *GitHub Action*

Saat ini ada banyak sekali bermunculan tools yang menawarkan kemudahan dalam menjalankan praktik CI/CD, dari beberapa tools tersebut salah satunya adalah *GitHub Action*. *GitHub Action* ini merupakan salah satu produk resmi dari *GitHub* repositori yang sering digunakan oleh para *Developer* untuk menyimpan dan memanajemen souce kode dengan rapi dan teratur. Kembali ke *GitHub Action*, *GitHub Actions* merupakan tools integrasi berkelanjutan dan pengiriman berkelanjutan (CI/CD) yang dapat memungkinkan para tim *DevOps* untuk melakukan mengotomatiskan alur pembangunan, pengujian, dan penerapan. Tim *DevOps* dibantu dengan *GitHub Action* ini akan membuat alur kerja untuk membangun dan menguji setiap permintaan *commit* dan *push* ke repositori. Sedikit berbeda dengan logo GitHub, logo *GitHub Action* dapat dilihat pada Gambar 2. 2 Logo *GitHub Action* dibawah ini.

Gambar 2. 2 Logo *GitHub Action*

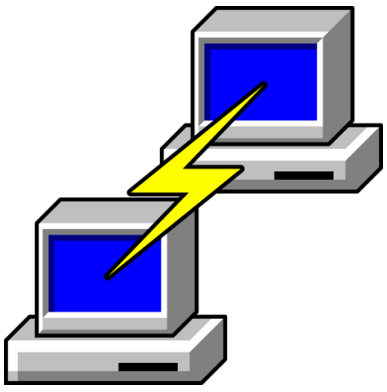
GitHub Actions hadir sebagai tools yang sangat membantu bagi para *DevOps*, dengan menggunakan *GitHub Action* dapat memungkinkan *Devops* untuk menjalankan dan mengotomatisasi alur kerja ketika terjadi perubahan *source code* di repositori. Setelah *DevOps* selesai melakukan konfigurasi *GitHub Action*, nantinya praktik CI/CD akan dapat berjalan atau bekerja untuk secara otomatis, salah satu fitur yang bermanfaat adalah *GitHub Action* dapat menambahkan label yang sesuai setiap kali terjadi *commit* dan *push* di repositori. Hal ini sangat berguna

sebagai penanda versi dari website. Nantinya akan terdata versi website yang ada mulai dari versi terlama hingga yang terbaru dengan perbaikan dari *bug* yang terdeteksi.

GitHub Action dapat support dengan sistem operasi Linux, Windows, dan macOS, *GitHub Action* juga dapat menghosting *runner* yang dihosting sendiri di pusat data atau infrastruktur *cloud* sendiri seperti yang dilakukan pada Proyek Akhir ini. Penulis melakukan hosting *runner* pada Server Linux Ubuntu yang sudah dikonfigurasi.

2.1.2 Putty

PuTTY merupakan software Open Source yang digunakan untuk melakukan remote access, seperti SSH atau Telnet. Dengan software ini, tim DevOps akan dapat mengakses komputer server dari jarak jauh,serta menghindari beratnya beban laptop ketika harus melekukan konfigurasi pada server. PuTTY tidak memiliki antarmuka grafis (GUI). Nantinya ketika berhasil terhubung, hanya akan ada tampilan baris perintah (command line) yang tersedia. Logo tool ini dapat dilihat pada Gambar 2. 3 Logo PuTTY.

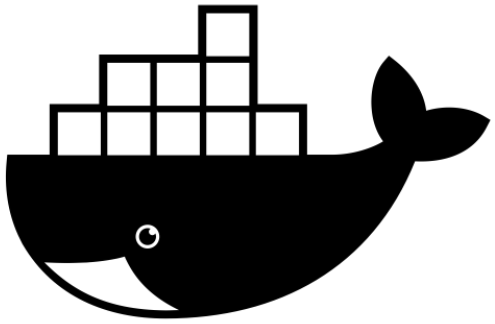


Gambar 2. 3 Logo PuTTY

2.1.3 Docker

Docker merupakan layanan yang mempunyai kemampuan untuk mengemas dan menjalankan sebuah website atau aplikasi dalam sebuah lingkungan terisolasi yang disebut dengan Container. Dengan adanya isolasi dan keamanan yang

memadai memungkinkan DevOps untuk menjalankan banyak Container di waktu yang bersamaan pada host tertentu. Salah satu fitur yang sering diguanakan adalah Docker Engine yang merupakan fitur untuk membuat image dan Container. Logo Docker digambarkan dengan paus yang membawa beberapa Container, seperti pada Gambar 2. 4 Logo Docker dibawah ini.



Gambar 2. 4 Logo Docker

Berikut beberapa istilah yang perlu diketahui yang mendukung praktik CI/CD :

1. *Dockerfile* :

   Dockerfile adalah file konfigurasi yang digunakan untuk membangun docker image (Ekaputra & Affandi, 2023). Secara lebih rinci dockerfile merupakan sebuah file yang berisi *source code* instruksi dan perintah yang nantinya akan digunakan oleh Docker Engine untuk membuild atau membuat sebuah Image Docker.

2. *Docker image* :

   Docker image merupakan hasil dari proses build menggunakan dockerfile sebelumnya. Dimana image ini berisi kode, pacakge, runtime, environment dan library yang dikemas dalam sebuah image.

3. *Docker Container* :

   Docker Container adalah teknologi yang dapat digunakan untuk memperkuat pemanfaatan layanan CDN dan cloud computing terhadap pengembangan aplikasi web (Ekaputra & Affandi, 2023). Dari docker Container inilah DevOps melakukan monitoring kinerja dari website atau aplikasi yang sudha dibuat untuk di cek apakah sudah siap untuk dirilis atau belum.

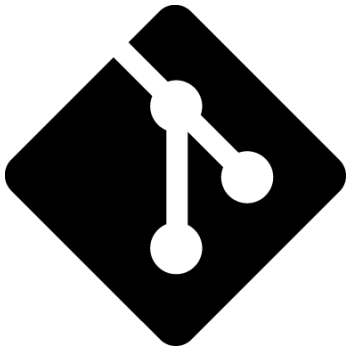
2.1.4 Node.js

Node.js adalah platform berbasis javascript runtime dengan skalabilitas tinggi yang dapat mengeksekusi kode javascript di luar lingkungan browser (Nurhayati & Agussalim, 2023). Dengan Node.js kita dapat menjalankan kode JavaScript di mana pun, tidak hanya terbatas pada lingkungan browser. Node.js juga menyediakan banyak library/module JavaScript yang membantu menyederhanakan pengembangan website atau aplikasi. Node.js ini digunakan sebagai *base image* dalam proyek akhir website **Rekomendasi Film**. Logo Node.js dapat dilihat pada Gambar 2. 5 Logo Node.js.

Gambar 2. 5 Logo Node.js

2.1.5 Git Bash

Git Bash merupakan software untuk lingkungan Microsoft Windows yang menyediakan lapisan emulasi untuk pengalaman baris perintah Git. Bash adalah singkatan dari Bourne Again Shell. Shell adalah software terminal yang digunakan untuk berinteraksi dengan sistem operasi melalui perintah tertulis. Bash adalah shell default yang populer di Linux dan macOS. Git Bash ini digunakan untuk membantu developer untuk melakukan *commit* dan *push* pada repositori. Dengan Git Bash ini DevOps juga menggunakan untuk melakukan *commit* dan *push* untuk Dockerfile dan file action yang selesai dibuat. Logo tools ini dapat dilihat pada Gambar 2. 6 Logo Git Bash.



Gambar 2.6 Logo Git Bash

2.1.6 SonarQube

SonarQube adalah tool yang digunakan untuk analisis *source code*, mengidentifikasi dan mengatasi masalah dalam kode mereka. Dengan menggunakan SonarQube penulis dapat membantu perusahaan untuk meningkatkan kualitas dan keamanan website dengan memberikan informasi terkait dengan masalah bug dan keamanan dari *source code* yang sudah dibuat dan dikembangkan. Logo SonarQube dapat dilihat pada Gambar 2. 7 Logo SonarQube.

Gambar 2.7 Logo SonarQube

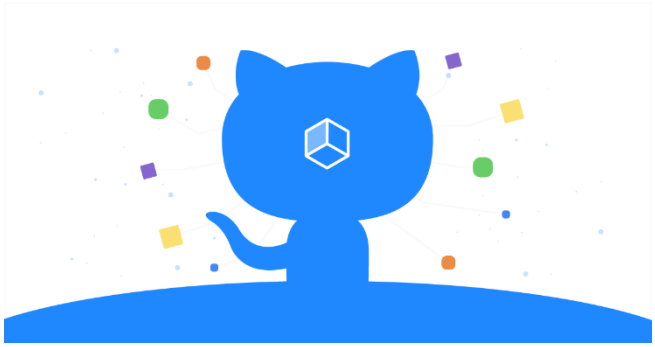
2.1.7 Trivy File Scan

Trivy File Scan adalah tool scanner keamanan untuk mengevaluasi keamanan Docker Container dan Dockerimage. Trivy digunakan untuk scanning masalah di dalam Dockerimage dan memberikan informasi tentang masalah tersebut. Trivy ini penulis gunakan untuk melakukan pemindaian keamanan pada sistem file/dependecy yang ada. Untuk logo tool ini dapat dilihat pada Gambar 2. 8 Logo Trivy File Scan

Gambar 2.8 Logo Trivy File Scan

2.1.8 GitHub Container Registry (GHCR)

GitHub Container Registry merupakan layanan penyimpanan dan manajemen docker image yang disediakan oleh GitHub. Ini memungkinkan DevOps untuk menyimpan, mengelola, dan mendistribusikan docker image secara terpusat melalui platform GitHub. Logo GitHub Container Registry hampir mirip GitHub Action yang bertema siluet kucing seperti yang terlihat pada Gambar 2. 9 Logo GitHub Container Registry.



Gambar 2.9 Logo GitHub Container Registry

Dengan GitHub Container Registry, DevOps dapat membuat dan menyimpan berbagai versi dari docker image proyek akhir dengan berbagai versi yang ada. Dengan GHCR ini tentunya dapat membantu dalam manajemen docker image secara teratur dan rapi.

GHCR ini terintegrasi dengan platform GitHub, memudahkan pengelolaan repositori docker image dan memanajemen versi kode dari proyek. Dengan demikian, GitHub Container Registry memungkinkan tim DevOps untuk lebih

mudah berkolaborasi dan menyebarkan aplikasi yang dikemas dalam bentuk docker image.

2.1.9 Grafana

Grafana adalah tool analisis dan visualisasi yang digunakan untuk memonitor dan mengamati data. Dengan Grafana, dashboard akan menjadi menarik dan nyaman untuk dilihat. Dashboard yang dibuat tentunya digunakan untuk menampilkan data dari berbagai sumber seperti database, sistem pemantauan, dan aplikasi lainnya, dalam hal ini digunakan menampilkan data yang berasal dari Prometheus. Logo tool ini dapat dilihat pada Gambar 2. 10 Logo Grafana.

Gambar 2.10 Logo Grafana

2.1.10 Prometheus

Prometheus adalah tool yang dirancang untuk memantau dan mengumpulkan metrik dari lingkungan infrastruktur. Prometheus dirancang untuk bekerja dengan baik dalam lingkungan Container dan memberikan fleksibilitas dalam pengambilan metrik. Dalam hal ini Prometheus digunakan membantu untuk mengumpulkan data dari Cadvisor untuk nantinya akan ditampilkan pada Grafana. Logo tool ini dapat dilihat pada Gambar 2. 10 Logo Prometheus.

Gambar 2.11 Logo Prometheus

2.1.11 cAdvisor

cAdvisor adalah tool yang digunakan untuk menyediakan informasi detail tentang penggunaan sumber daya dan kinerja Container. cAdvisor dirancang khusus untuk bekerja dengan Container Docker dan Container runtime lainnya. Dalam hal ini cAdvisor digunakan membantu untuk mendata kinerja dari semua Container yang berjalan pada server. Logo tool ini dapat dilihat pada Gambar 2. 10 Logo cAdvisor.

Gambar 2.12 Logo cAdvisor

2.2 Tinjauan Pustaka

Tabel 2.1 Tabel Tinjauan Pustaka

Peneliti	Judul Peneltian	Hasil
Zaidan Zulhakim, Ari Kurniawan	Implementasi *Continuous Integration* Dan *Continuous Deployment* Pada	Mendapatkan hasil implementasi dan pengujian pada Implementasi Continuous Integration dan Continuous Deployment pada Pengembangan Aplikasi Website 10

	Pengembangan Aplikasi Website Menggunakan *Docker* Dan *Github Actions*	Menggunakan Docker, GitHub Actions dan Jenkins lalu dibandingkan mana yang lebih cepat diantara 2 itu. Dan didapatkank bahwa yang lebih cepat adalah GitHub Action. (Zulhakim [1], Kurniawan[2])
Jaeni, Nicko Aji S, Arid Dwi L.	Implementasi *Continuous Integration/Continuous Delivery* (CI/CD) Pada Performance Testing DevOps	Dalam Penelitian ini didapatkan hasil, bahwa proses deployment menggunakan CI/CD mampu mempersingkat proses deployment dikarenakan proses deployment dengan CI/CD memiliki standarisasi yang telah dibuat sebelumnya lalu berjalan secara otomatis dan menggunakan tools. Sehingga hanya sedikit campur tangan dari manusia saat terjadi proses deployment. Selain itu CI/CD memberikan hasil yang lebih teliti dengan penemuan bug pada pengujian tersebut (Laksito, 2022).
Argian Raditya Putra, Hariandi Maulid, Muhammad Purwadi	Implemantasi CI/CD Pada Proyek E-RBA	Dalam penelitian ini digunakan untuk menyajikan sebuah studi mengenai implementasi Continuous Integration/Continuous Deployment (CI/CD) pipeline di lingkungan kantor untuk mendukung pengembangan proyek E-RBA dengan source code yang kompleks. CI/CD merupakan metodologi pengembangan perangkat lunak yang mengintegrasikan integrasi kode, dan otomatisasi proses deploy guna meningkatkan efisiensi dan kualitas pengembangan.
Kurniawan Farid Jadmiko	Penerapan GitHub Action Runner untuk Praktik CI/CD pada Website Rekomendasi Film Studi Kasus PT Cloufina Nata Karya	Menjalankan praktik CI/CD dan tujuan utamannya mendapatkan hasil berupa *bug* kinerja secara menyeluruh, mulai dari penggunaan resource sistem operasi hingga penggunaan jaringan. Dalam mengintegrasikan beberapa *tool* dan File Action YAML dapat mempermudah dalam proses deployment aplikasi pada PT Cloufina Nata Karya.