BAB 2
TEORI DAN TINJAUAN PUSTAKA

Pada bab 2 akan dibahas tentang dasar teori dan tinjauan pustaka yang digunakan dalam pembuatan Proyek Akhir ini.

2.1 DevOps

DevOps merupakan singkatan dari Development dan Operations, dengan istilah lain DevOps merupakan konsep kolaborasi dan komunikasi antara tim pengembang software dengan tim operasional software. DevOps menjadi strategi yang praktis dengan tujuan mendapatkan software yang efisien sesuai dengan kebutuhan perusahaan karena dapat mengurangi beberapa tahapan pengembangan yang ada pada metode sebelumnya tanpa mengurangi kualitas pada software yang sedang dikembangkan tersebut.

Pengadopsian DevOps di beberapa perusahaan menunjukkan manfaat bagi pengembangan perangkat lunak (aplikasi/website). Devops mendukung kecepatan produksi perangkat lunak dan mengurangi usaha yang dibutuhkan ketika menyiapkan perilisan, sehingga memungkinkan perusahaan untuk rilis dalam waktu yang cepat sesuai keperluan. Penerapan DevOps dinilai menghemat 4 penggunaan sumber daya untuk pengembangan dan pemeliharaan, karena adanya otomatisasi.

DevOps memiliki dampak positif dari sisi karyawan salah satunya membantu mengurangi stress akibat kecemasan, dengan hal itu akan meningkatkan kesejahteraan. Hal itu membuktikan DevOps tidak hanya membawa manfaat untuk perusahaan, namun juga bagi pekerja (Farid & Anugrah, 2021).

2.2 CI/CD

CI/CD merupakan singkatan dari Continuous Integration dan Continuous Delivery/Deployment. Tahapan dari Continuous Integration yaitu melakukan integrasi otomatis dan terus-menerus dari perubahan *source code* yang dilakukan oleh developer ke dalam repositor (GitHub). Jika developer melakukan perubahan

source code dan mengirimkannya ke repository, sistem otomatis akan melakukan proses integrasi, termasuk kompilasi dan pengujian otomatis melalui Jenkins. Sehingga distribusi perangkat lunak akan lebih efektif dan efisien. Continuous Integration bertujuan untuk meminimalkan konflik dan masalah integrasi yang dapat terjadi ketika banyak developer bekerja pada proyek yang sama (Laksito, 2022).

Tahapan dari Continuous Delivery yaitu proses sebuah pipeline Jenkins yang sudah berhasil "sukses" dari tahap integrasi serta pengujian, akan dikirimkan ke Nexus Repository. Kemudian untuk tahapan Continuous Deployment akan dilaksanakan ketika Continuous Delivery telah berhasil dilakukan, maka tahap Continuous Deployment akan memasukkan hasil build image ke dalam Server.

Pengaplikasian CI/CD pada PT Cloufina Nata Karya dapat memberikan keunggulan dengan mempercepat siklus pengembangan, meningkatkan kualitas perangkat lunak, dan meningkatkan responsivitas. Dengan otomatisasi pengujian dan penyebaran, CI/CD mengurangi risiko kesalahan manusia dan memungkinkan perusahaan untuk secara konsisten dan mempercepat pengiriman fitur baru tetapi juga mengidentifikasi bug lebih awal dalam siklus pengembangan.

Gambar 2. 1 Alur CI/CD

Terlihat pada Gambar 2. 1 Alur CI/CD, merupakan alur dari CI/CD yang digunakan Penulis sebagai acuan dalam proses CI/CD pada pengembangan Website Company Profile PT Cloufina Nata Karya.

2.1.1 GitHub

GitHub merupakan platform penyimpanan Repository yang mengijinkan developers untuk melakukan pelacakan permasalah *source code* dan manajemen *source code* secara gratis (Wijayanto & Adhinata, 2021). GitHub menyediakan

sistem kontrol yang kuat sehingga developer dapat menyimpan, melacak, dan berkolaborasi pada perubahan *source code* menggunakan Git yang memungkinkan tim untuk mengelola riwayat perubahan dengan efisien. Logo dari GitHub dapat dilihat pada Gambar 2. 2 Logo GitHub.

GitHub juga memiliki fitur untuk menyimpan variabel-variabel rahasia salah satunya Credential. Credential adalah informasi otentikasi yang digunakan untuk mengakses sumber daya terbatas, mencakup username, password, token API, atau kunci rahasia lainnya yang diperlukan untuk meng-autentikasi dan otorisasi pengguna atau aplikasi. Hal ini membantu dalam pengelolaan konfigurasi aplikasi yang dapat diakses oleh sistem CI/CD. Penggunaan GitHub dalam praktik CI/CD juga memberikan manfaat bagi developer karena mengadopsi siklus pengembangan perangkat lunak yang terotomatisasi, konsisten, dan terukur sehingga meningkatkan produktivitas dan kualitas perangkat lunak secara keseluruhan.

Gambar 2. 2 Logo GitHub

2.1.2 Jenkins

Jenkins adalah tools open source yang dapat melakukan build dan deploy sebuah proyek dengan ringan dan cepat secara otomatis. Jenkins menjadi tools yang paling populer untuk CI/CD dan dilengkapi ratusan plugin agar mendukung optimalisasi dalam proses CI/CD (Shama & Chandra, 2021). Logo Jenkins cukup menarik dan dapat dilihat pada Gambar 2. 3 Logo Jenkins.

Jenkins memberikan fleksibilitas dan optimalisasi dalam menjalankan proses CI/CD. Keunggulan Jenkins tidak hanya terletak pada kemampuannya untuk

mengotomatisasi build dan deployment proyek, tetapi juga pada integrasi yang mudah dengan berbagai alat pengembangan. Melalui integrasi otomatis dengan GitHub, Jenkins memastikan setiap perubahan *source code* diuji secara otomatis, meningkatkan keandalan dan konsistensi perangkat lunak. Selain itu, kemampuan Jenkins untuk berintegrasi dengan alat *containerisasi* seperti Docker menambah dimensi fleksibilitas dalam manajemen dan penyebaran aplikasi. Dengan memfasilitasi pembuatan pipeline CI/CD yang dapat disesuaikan, Jenkins memberikan solusi yang dapat diadaptasi untuk memenuhi berbagai alur kerja dan kebutuhan pengembangan perangkat lunak.

Gambar 2. 3 Logo Jenkins

2.1.3 Server (Ubuntu 22.04.3)

Dalam konteks IT, server memiliki peran utama dalam menyediakan layanan, menyimpan data, atau menjalankan aplikasi yang dapat diakses oleh pengguna atau perangkat lain melalui jaringan, seperti internet atau jaringan lokal. Dalam proses CI/CD, server memegang peran krusial sebagai tempat eksekusi untuk tahapan-tahapan dalam siklus CI/CD. Server menyimpan *Docker Image* atau hasil dari proses CI/CD seperti file biner, *source code* yang telah dikompilasi, atau file lainnya. *Docker Image* ini nantinya dapat digunakan untuk tahapan selanjutnya, seperti distribusi atau penyebaran ke tahap produksi. Logo terbaru dari Ubuntu dapat dilihat pada Gambar 2. 4 Logo Ubuntu 22.04.3.

Server berfungsi sebagai tempat utama untuk alat-alat CI/CD seperti Jenkins, GitLab CI, atau alat CI/CD lainnya. Hal ini memungkinkan otomatisasi dan koordinasi efisien dalam setiap tahap CI/CD. Dalam Proyek Akhir ini, menggunakan Ubuntu 22.04.3. Ubuntu secara umum diakui sebagai pilihan yang populer untuk server dalam berbagai lingkungan pengembangan dan produksi.

Gambar 2. 4 Logo Ubuntu 22.04.3

2.1.4 Docker

Docker adalah sebuah projek open source yang ditujukan untuk software engineer, DevOps engineer atau sysadmin untuk membangun, mengemas dan menjalankan aplikasi dimana pun di dalam sebuah *container*. Docker berfungsi sebagai virtualisasi di dalam sebuah sistem operasi atau sebuah server atau sebuah web server atau bahkan sebuah database server, dimana dengan menggunakan virtualisasi ini, diharapkan developer dapat mengembangkan aplikasi sesuai dengan spesifikasi server (Shama & Chandra, 2021). Logo Docker bisanya digambarkan dengan beberapa container seperti pada Gambar 2. 5 Logo Docker. Beberapa istilah Docker yang digunakan dalam CI/CD :

a. *Dockerfile*

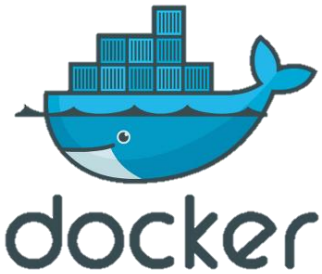
   Berisi dokumen text atau script yang berisi semua perintah yang biasanya dilakukan manual untuk membangun sebuah image. Dengan menggunakan perintah docker build dari terminal, Docker akan membangun image secara bertahap berdasarkan eksekusi perintah dalam script (Ekaputra & Affandi, 2023).

b. *Docker Image*

Paket eksekusi yang mencakup semua yang diperlukan untuk menjalankan aplikasi, termasuk *source code*, runtime, library, variabel lingkungan, dan pengaturan sistem. *Docker Image* merupakan read-only template untuk menjalankan *container*.

c. *Docker Container*

Sebuah abstraksi layer aplikasi yang membungkus package diperlukan oleh aplikasi. Saat menjalankan *Docker Image*, *container* menjadi lingkungan yang terisolasi di mana aplikasi dapat berjalan tanpa mengganggu sistem host. *Container* dapat dimulai, dihentikan, dihapus, dan dikelola dengan mudah (Ekaputra & Affandi, 2023).



Gambar 2. 5 Logo Docker

2.1.5 Nexus

Sonatype Nexus adalah platform manajemen repository yang digunakan untuk menyimpan dan mengelola *Docker Image* dari sebuah perangkat lunak. *Docker Image* dari perangkat lunak ini dapat berupa berbagai jenis, termasuk file JAR, WAR, dan berbagai dependensi atau komponen yang digunakan dalam pengembangan perangkat lunak.

Dalam tahap Continuous Delivery, setelah proses build dan pengujian selesai di Jenkins, Nexus memastikan pengambilan otomatis dan konsisten dari *Docker Image* perangkat lunak yang dihasilkan selama proses build dan pengujian. Nexus bertindak sebagai repository pusat yang menyimpan hasil build yang telah

diuji dan diverifikasi, memastikan *Docker Image* siap diterapkan ke tahap selanjutnya yaitu tahap Continuous Deployment.

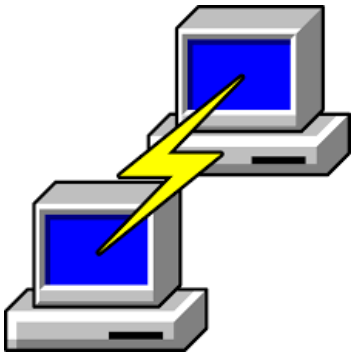
Dengan menyimpan beberapa versi *Docker Image* di dalam Nexus Sonatype Repository dan versi yang benar-benar diperlukan disimpan dalam Server Ubuntu, hal ini akan menghemat ruang penyimpanan di Server Ubuntu sehingga dapat meringankan beban kerja dan penyimpanan dari Server. Logo dari Nexus bisa dilihat pada Gambar 2. 6 Logo Sonatype Nexus dibawah ini.

Gambar 2. 6 Logo Sonatype Nexus

2.1.6 PuTTY

Seperi yang ada pada Gambar 2. 7 Logo PuTTY, PuTTY adalah sebuah aplikasi terminal emulator, serial console, dan Client untuk protokol jaringan, khususnya SSH (Secure Shell), Telnet, rlogin, dan serangkaian protokol lainnya. PuTTY sangat populer dalam lingkungan sistem operasi Windows dan digunakan untuk berinteraksi dengan server atau perangkat jaringan melalui koneksi jarak jauh. Dalam praktik CI/CD, PuTTY berperan sebagai antarmuka untuk mengakses dan mengelola server dari proses CI/CD berlangsung secara jarak jauh.

Fungsi utama PuTTY adalah memberikan akses remote yang aman melalui protokol SSH, memungkinkan eksekusi perintah di terminal server jarak jauh, sehingga dapat mengelola konfigurasi, menjalankan script, atau memonitor proses CI/CD tanpa harus berada di lokasi fisik server. Selain itu, PuTTY juga menyediakan utilitas seperti PuTTYgen untuk manajemen kunci SSH, yang digunakan untuk mengamankan koneksi dan otentikasi antara server dan Client dalam konteks CI/CD.

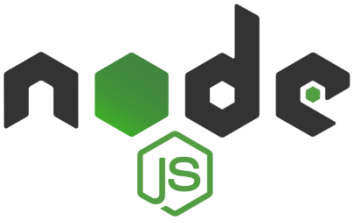


Gambar 2. 7 Logo PuTTY

2.1.7 Node.js

Dalam proses CI/CD, Node.js digunakan sebagai base image dalam pembuatan *container* menggunakan Docker. Sebagai base image, Node.js menyediakan lingkungan runtime yang dibutuhkan untuk menjalankan aplikasi berbasis JavaScript atau Node.js di dalam *container*.

Digunakannya Node.js sebagai base image, dapat berfungsi untuk membuat *Container* yang berisi aplikasi Node.js beserta semua dependensinya. *Container* ini menjadi unit yang portabel dan dapat dijalankan di berbagai lingkungan yang mendukung Docker. Selain itu, penggunaan Node.js sebagai base image dapat digunakan dalam langkah-langkah build dan deploy dalam proses CI/CD. Hal ini dapat menjaga konsistensi lingkungan runtime di seluruh siklus pengembangan perangkat lunak. Logo dari Node.js dapat dilihat pada Gambar 2. 8 Logo Node.js.



Gambar 2. 8 Logo Node.js

2.1.8 SonarQube

Sonarqube merupakan salah satu tool SAST. Sonarqube adalah otomasi review *source code* program untuk mendeteksi bugs, vulnerabilities, dan code smells Sonarqube dapat diintegrasikan dengan CI/CD tools seperti Jenkins, CircleCI, AWS CodeBuild, Azure DevOps, Atlassian Bamboo, atau Travis CI (Shama & Chandra, 2021).

SonarQube menjadi platform analisis kualitas code yang dapat membantu dalam memelihara kualitas perangkat lunak. SonarQube dapat memberikan informasi mengenai kualitas code berdasarkan berbagai faktor seperti kompleksitas, kejelasan, dan kepatuhan terhadap standar. Informasi yang diberikan oleh SonarQube tersebut akan ditampilkan melalui visualisasi, sehingga dapat membantu tim untuk fokus terhadap area perbaikan. Platform ini juga sudah terintegrasi dengan beberapa alat CI seperti Jenkins, GitHub Actions, GitLab CI sehingga akan melakukan analisis otomatis setiap kali ada perubahan pada code. Logo SonarQube dapat terlihat pada Gambar 2. 9 Logo SonarQube.

Gambar 2. 9 Logo SonarQube

2.1.9 Trivy

Trivy merupakan tools pemindai keamanan khusus untuk untuk Docker *container* dengan memfokuskan deteksi terhadap kerentanan keamanan pada *Docker Image* dan berbagai dependensinya. Trivy dirancang untuk memberikan hasil analisis yang cepat dan diintegrasikan dalam alur kerja CI/CD sehingga dapat mendeteksi kerentanan sejak dini.

Trivy dapat diintegrasikan langsung ke dalam alur kerja CI/CD, dengan memberikan kecepatan dan otomatisasi dalam mendeteksi kerentanan seiring dengan proses pengembangan perangkat lunak. Integrasi Trivy dalam CI/CD

memastikan bahwa setiap perubahan *source code* yang dijalankan dalam *container* telah melewati pemindaian keamanan, meminimalkan risiko keamanan sejak tahap awal pengembangan. Sehingga SonarQube dan Trivy akan saling berhubungan, dimana SonarQube berfokus pada analisis kualitas code, sementara Trivy lebih berfokus pada keamanan Docker *Container*. Dengan kombinasi penggunaan keduanya dapat memberikan keamanan dan kualitas yang lebih kuat pada proses pengembangan perangkat lunak. Logo dari Trivy dapat dilihat pada Gambar 2. 10 Logo Trivy.

Gambar 2. 10 Logo Trivy

2.1.10 cAdvisor

cAdvisor merupakan singkatan dari *Container* Advisor, tool ini berfungsi untuk memonitor sumber daya kinerja suatu *container* dalam sebuah lingkungan Docker. cAdvisor dapat memberikan informasi mengenai penggunaan CPU, memori, penyimpanan, jaringan, dan performa dari setiap *container* yang berjalan. Data dan informasi dari hasil pemantauan dapat digunakan untuk analisis tren jangka panjang oleh tim sehingga dapat membantu perbaikan berkelanjutan.

Manfaat dari penggunaan cAdvisor adalah kemampuannya untuk memberikan pemahaman mendalam tentang kinerja *container* tanpa memerlukan konfigurasi yang rumit. Dengan informasi yang akurat dan real-time yang disediakan oleh cAdvisor, PT Cloufina Nata Karya dapat melakukan pemantauan proaktif dan meningkatkan efisiensi penggunaan sumber daya secara keseluruhan dalam lingkungan Docker. Logo dari tool cAdvisor dapat dilihat pada Gambar 2. 11 Logo cAdvisor.

Gambar 2. 11 Logo cAdvisor

2.1.11 Prometheus

Prometheus adalah perangkat lunak pemantauan dan peringatan sistem yang bersifat opensource awalnya dibuat di SoundCloud. Sejak dimulai pada 2012, banyak perusahaan dan organisasi yang telah mengadopsi Prometheus dan memiliki banyak komunitas pengembang dan pengguna yang sangat aktif. Sekarang proyek open source mandiri dan dikelola secara independen dari perusahaan mana pun. Prometheus mengumpulkan metrik dari data resource, baik secara langsung atau melalui gateway push prometheus menggunakan metrik untuk pekerjaan yang berjangka pendek. Metrik mengumpulkan data yang diambil dari exporter yang telah di install dan dapat digunakan untuk memberi peringatan (Rahman et al., 2020).

Prometheus terintegrasi dengan lingkungan berbasis *container* sehingga mendukung pengumpulan metrik dari layanan-layanan yang berjalan di dalam *Container*. Data yang dikumpulkan oleh Prometheus berasal dari cAdvisor yang kemudian akan ditampilkan oleh Grafana. Logo Prometheus digambarkan seperi ujung obor dan dapat dilihat pada Gambar 2. 12 Logo Prometheus.

Gambar 2. 12 Logo Prometheus

2.1.12 Grafana

Grafana adalah perangkat lunak visualisasi dan analitik yang bersifat open source. Grafana memungkinkan untuk memvisualisasikan, mengingatkan, dan menjelajahi metrik disimpan. Grafana juga memiliki dashboard template yang bisa digunakan untuk mengumpulkan variabel data yang digunakan. Dalam paparan ini dijelaskan bahwa Grafana sangat support dalam visualisasi data dalam bentuk time series. Grafana digunakan untuk menampilkan status service yang berjalan pada aplikasi maupun server yang digunakan. Beberapa data source yang didukung dari Grafana, antara lain : Graphite, InfluxDB, OpenTSDB, Prometheus, Elasticsearch, dan CloudWatch (Rahman et al., 2020).

Dalam Proyek Akhir ini, data yang ditampilkan oleh Grafana berasal dari Prometheus. Grafana bertujuan untuk menyajikan data melalui dashboard yang interaktif sehingga mudah dimengerti dan memberikan alat yang kuat dalam memantau, menganalisis, dan berinteraksi dengan data, sehingga dapat membantu tim dalam mengambil keputusan. Logo dari tool Grafana dapat dilihat pada Gambar 2. 13 Logo Grafana dibawah ini.

Gambar 2. 13 Logo Grafana

2.3 Tinjauan Pustaka

Adapun beberapa penelitian terdahulu yang dijadikan acuan untuk penyusunan Proyek Akhir ini ditunjukkan pada Tabel 2. 1 Tabel Tinjauan Pustaka:

Tabel 2. 1 Tabel Tinjauan Pustaka

Peneliti	Judul Penelitian	Hasil
Danur Wijayanto, Arizona Firdonsyah, Faisal Dharma Adhinata	Implementasi Continuous Integration/Continuous Delivery Menggunakan Process Manager2 (Studi Kasus: SIAKAD Akademi Keperawatan Bina Insan)	Hasil dari penelitian ini yaitu penerapan CI CD pada Sistem Informasi Akademik (SIAKAD) Akademi Keperawatan Bina Insan tidak menggunakan Docker *Container* sebagai wadah aplikasi, namun aplikasi dijalankan langsung menggunakan PM2. Dengan menggunakan PM2, repository atau tempat penyimpanan tambahan untuk menyimpan *Docker Image* sudah tidak diperlukan lagi. Selain itu, jika PM2 diterapkan pada lingkungan komputer dengan sumber daya terbatas, maka PM2 memiliki performa yang lebih baik daripada Docker (Wijayanto & Adhinata, 2021).
Andrian Alpery, Muhammad Arif Fadhly Ridha	Implementasi CI/CD Dalam Pengembangan Aplikasi Web Menggunakan Docker Dan Jenkins	Dalam Penelitian ini menghasilkan Pengembangan aplikasi web yang menggunakan framework CodeIgniter dengan metode CI/CD. Dalam proses CI/CD penelitian ini menggunakan Docker dan Jenkins, dimana deployment dilakukan sebanyak 10x. Hasil dari deployment tersebut merupakan hasil rata-rata waktu yang dibutuhkan jenkins dalam melakukan proses deployment jenkins pipeline adalah 1 menit 58 detik, tingkat keberhasilan jenkins dalam melakukan proses deployment adalah sebesar 90%, dan jenkins mendeteksi adanya bugs dengan total 78 bug (Alpery & Ridha, 2021).
Jaeni, Nicko Aji S, Arid Dwi L.	Implementasi Continuous Integration/Continuous	Dalam Penelitian ini didapatkan hasil bahwa proses deployment

	Delivery (CI/CD) Pada Performance Testing DevOps	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. Berbeda dengan proses tradisional yang sedari awal sudah bergantung dengan campur tangan manusia dari proses testing sampai proses upload ke server. Selain itu CI/CD memberikan hasil yang lebih teliti dengan penemuan bug pada pengujian tersebut (Laksito, 2022).
Ahmad Farid, Indra Gita Anugrah.	Implementasi CI/CD Pipeline Pada Framework Androbase Menggunakan Jenkins (Studi Kasus: PT. Andromedia)	Hasil dari penelitian ini adalah melakukan deploy aplikasi di PT. Andromedia, dimana aplikasi tersebut dibangun menggunakan framework androbase. Dengan mengintegrasikan aplikasi androbase menggunakan CI/CD pipeline dapat mempermudah proses deployment aplikasi pada PT. Andromedia. Dalam proses deploy aplikasi, penelitian ini menggunakan 4 stage pipeline Jenkins yaitu Stage Pull, Stage Build, Stage Test, dan Stage Deploy (Farid & Anugrah, 2021)
Salma Alfinda	Implementasi CI/CD Menggunakan Jenkins pada Pengembangan Website Company Profile	Hasil penelitian ini adalah menerapkan praktik Continuous Integration/Continuous Deployment (CI/CD) pada website Company Profile dengan tujuan memperoleh website yang lebih efisien dan memiliki tingkat kesalahan (bug) yang minimal saat diimplementasikan secara langsung. Hal ini dilakukan dengan mengintegrasikan beberapa perangkat (tools) dengan Jenkins sebagai platform

		utama. Beberapa aspek yang dievaluasi dari website termasuk minimisasi bug, penggunaan sumber daya pada sistem operasi, dan efisiensi penggunaan jaringan internet.