BAB II

TINJAUAN PUSTAKA DAN DASAR TEORI

2.1 Tinjauan Pustaka

Dalam penelitian ini menggunakan sumber pustaka yang berhubungan dengan metode atau kasus yang diteliti, antara lain :

Tabel 2. 1 Tinjauan Pustaka

No	Penulis	Judul	Tahun	Metode
1	Nurul Khairina Surbakti, Muhammad Arif Fadhly Ridha	Implementasi Kubernetes Cluster Menggunakan KVM	2021	*Stress Testing* pada cluster Kubernetes yang dibangun pada virtualisasi KVM
2	Abriza Mahandis Shama, Dian W. Chandra	Implementasi Static Application Security Testing Menggunakan Jenkins CI/CD	2021	*Security testing* menggunakan Jenkis CI/CD

		Berbasis Docker ContainerPada PT. Emporia Digital Raya		
3	Andrian Alpery, Muhammad Arif Fadhly Ridha	Implementasi CI/CD Dalam Pengembangan Aplikasi Web Menggunakan Docker dan Jenkins	2021	Implementasi CI/CD
4	Lianping Chen	Microservices: Architecting for Continuous Delivery and DevOps	2018	Penerapan arsitektur *microcersices* untuk Continuous Delivery dan DevOps
5	Ahmad Farid, Indra Gita Anugrah	Implementasi CI/CD Pipeline Pada Framework	2021	Implementasi CI/CD menggunakan Jenkins

		Androbase Menggunakan Jenkins (Studi Kasus: PT. Andromedia)		
6	Irvan Maulana	Implementasi Deployment Layanan Website Menggunakan Kubernetes Dengan CI/CD Jenkins	2022	CI/CD dengan kubernetes *cluster*

Nurul Khairina Surbakti dan Muhammad Arif Fadhly Ridha (2021) melakukan pengujian *High-Availability* dan *Stress testing* untuk Kubernetes cluster dengan server konvensional yang menghasilkan apabila node yang sedang menjalankan servicenya gagal atau mati, maka masih terdapat node yang lain dan penggunaan memory lebih banyak di Kubernetes cluster.

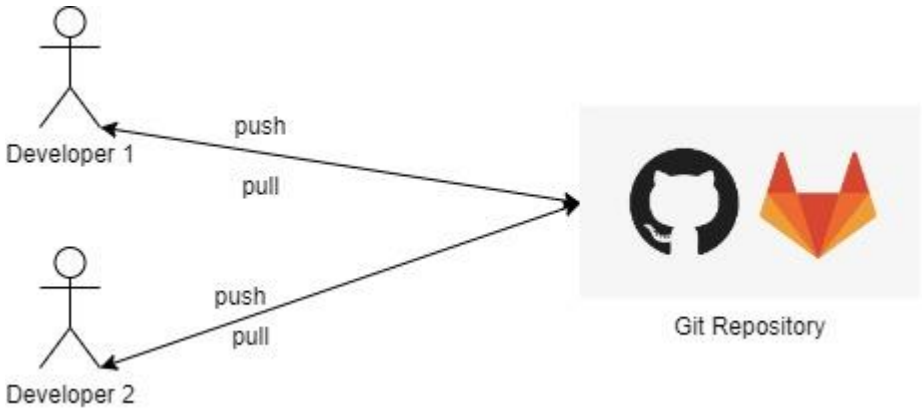
Abriza Mahandis Shama dan Dian W. Chandra (2021) melakukan pengujian keamanan kode program PT. Emporia Digital Raya menggunakan security testing SAST yang terdapat pada Jenkins.

Ahmad Farid dan Indra Gita Anugrah (2021) melakukan penelitian tentang Implementasi CI/CD Pipeline Pada Framework Androbase Menggunakan Jenkins (Studi Kasus: PT. Andromedia) yang menghasilkan proses deployment aplikasi yang dibuat menggunakan framework androbase menjadi lebih lebih cepat karena semua berjalan otomatis.

2.2 Dasar Teori

2.2.1 Git

Git adalah tool *version control system* open-source yang digunakan developer dalam memelihara code mulai dari proyek kecil sampai proyek berskala besar (M. Saiful Mukharom, 2015).



Gambar 2. 1 *Git Repository*

Dengan menggunakan git, developer bisa mengembangkan code nya dengan branch masing-masing, menggabungkan git dengan git repository dapat memudahkan developer borkolaborasi dan manajemen code.

2.2.2 Ubuntu server

Ubuntu merupakan system operasi (OS) open-source gratis yang berbasis kernel Linux (varizal, 2014). Sedangkan “Server adalah sebuah komputer yang menjadi pusat kegiatan suatu jaringan yang dapat memproses satu atau lebih layanan jaringan.” (M. Doss, 1999:1). Singkatnya Ubuntu server adalah system operasi tanpa Graphical User Interface (GUI) yang menjadi pusat kegiatan jaringan.

2.2.3 Amazon Web Services (AWS)

Amazon Web Services (AWS) adalah salah *platform cloud* yang menyediakan berbagai layanan,diantaranya : komputasi, database, *machine learning* dan berbagai layanan lainnya yang sangat berguna dalam perkembangan bisnis. AWS menyediakan masa percobaan selama 12 bulan untuk mencoba layanan yang disediakan AWS secara gratis. Namun pada masa percobaan ini layanan yang bisa digunakan terbatas, misalkan ketika menggunakan layanan *Elastic Compute Cloud* (EC2) hanya bisa menggunakan *instance* T2 *micro*.

2.2.4 Amazon Simple Storage Service (Amazon S3)

Amazon Simple Storage Service (Amazon S3) adalah salah satu layanan penyimpanan objek dari *Amazon Web Services* yang menawarkan skalabilitas, ketersediaan data, keamanan, dan performa terdepan di bidang industri. Lebih mudahnya amazon S3 ini mirip dengan google drive, namun dengan skala dan fitur yang lebih memadai untuk bidang industri.

2.2.5 Amazon Route 53

Amazon Route 53 adalah layanan web Domain Name System (DNS) cloud dengan ketersediaan dan skala yang luas. Amazon route 53 ini berfungsi untuk merutekan lalu lintas aplikasi ke internet dengan menerjemahkan nama domain ke alamat IP. Amazon Route 53 menghubungkan permintaan pengguna secara efektif ke infrastruktur yang berjalan di AWS, seperti *instans Amazon EC2*, penyeimbang beban *Elastic Load Balancing*, atau *bucket Amazon S3* dan juga dapat digunakan untuk merutekan pengguna ke infrastruktur di luar AWS.

2.2.6 CI/CD (Continous Integration/Continous Delivery)

Menurut Andrian Alpery, Muhammad Arif Fadhly Ridha(2021) "Continous Integration/Continous Delivery(Deployment)(CI/CD) adalah metode yang menjembatani antara tim pengembang dan tim operasional dalam proses *build, testing* dan *deployment* aplikasi". CI/CD mempermudah proses perilissan maupun pengembangan aplikasi secara cepat dan terorganisir.

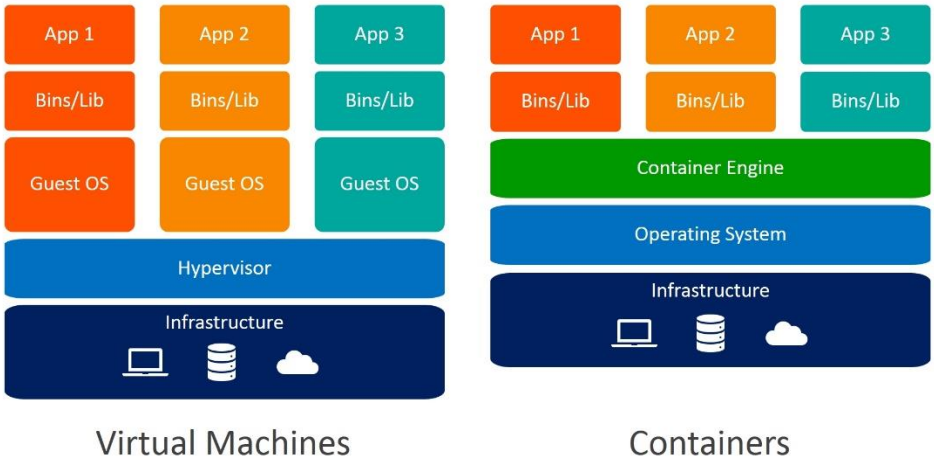
2.2.7 Jenkins

Jenkins adalah tool otomatisasi *open-source* yang ditulis menggunakan Bahasa pemrograman java, jenkins dapat digunakan untuk mengotomatisasi semua jenis tugas yang terkait dengan *build, testing* dan *deployment* perangkat lunak (Jenkins.io diakses pada 12 juli 2022).

2.2.8 Container

*Containe*r adalah virtualisasi pada level sistem operasi dimana tiap proses atau aplikasi yang dijalankan tiap *container* memiliki kernel yang sama. Hal ini

menjadi keuntungan sendiri dibandingkan virtualisasi pada level mesin (*virtual machine*), dimana *virtual machine* membutuhkan kernel sistem operasi yang berbeda beda tiap aplikasi yang dijalankan.(Tanjung P. Kusuma, dkk, 2017).



Gambar 2. 2 Perbedaan *Virtual Machine* Dengan *Container*

Teknologi *container* memungkinkan aplikasi berjalan pada kernel *container* atau terisolasi dari *Hypervisor* dan system operasi sehingga segala proses yang terjadi pada container tidak akan mempengaruhi system operasi yang sedang berjalan, berbeda dengan virtual machine diamna aplikasi berjalan langsung diatas system operasi dan *Hypervisor*.

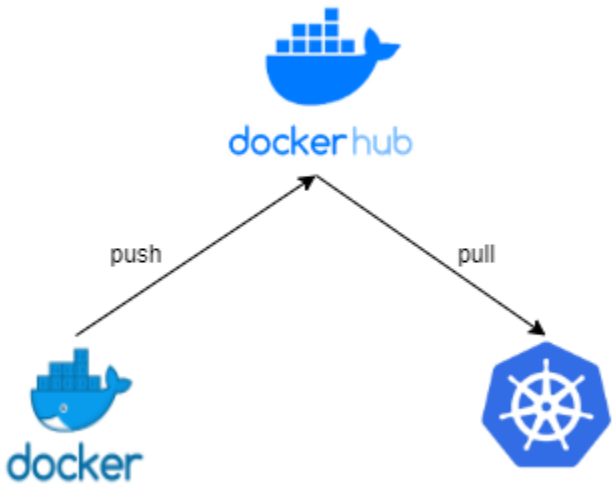
2.2.9 Docker

Docker adalah salah satu *platform* yang dibangun berdasarkan teknologi *container*. Docker merupakan sebuah project *open source* yang menyediakan platform terbuka untuk developer maupun *sysadmin* untuk dapat membangun, mengemas, dan menjalankan aplikasi dimanapun sebagai sebuah wadah(container)

yang ringan. Berbeda denga virtualisasi yang mana aplikasi berjalan diatas hypervisor dan guest OS, docker dapat menjalankan aplikasi langsung tanpa kedua hal tadi. Docker juga dilengkapi dengan fitur *sandbox* yang me njamin pengerjaan pengembang dan sysadmin tidak terganggu. *Sandbox* pada istilah keamanan komputer adalah mekanisme pemisahan aplikasi atau program tanpa mengganggu host (isolasi). (Iin Supiawati, 2018).

2.2.10 DockerHub

Dockerhub adalah *registry images* yang bersifate pribadi (private) yang memiliki akses terbatas ataupun *public* (bisa diakses siapapun). Dockerhub berfungsi untuk menyimpan docker image di internet, sehingga dapat di akses dari manapun. Secara default, mesin yang sudah terpasang docker engine, informasi login akan mengarah ke dockerhub, sehingga tidak diperlukan konfigurasi tmabahan jika akan mengakses dockerhub dari mesin untuk melakukan *push* atau *pull docker image.*



Gambar 2. 3 *Push* dan *Pull Dockerhub*

Images yang telah di-*build* menggunakan docker kemudian dipush ke dockerhub. Setelah *images* tersedia pada repository dockerhub, *images* bisa dipull ke Kubernetes cluster untuk proses deployment.

2.2.11 Kubernetes

Kubernetes adalah *container orchestration* yang berfungsi untuk memastikan seluruh container dalam keadaan berjalan dengan baik dalam memproses *workload* yang berjalan pada mesin fisik atau mesin virtual (M. A. Nugroho, 2018). Untuk menjalankan Kubernetes cluster setidaknya memerlukan 2 *node* baik fisik ataupun virtual, yaitu *master node* dan *worker node*.

Master node menyediakan *control plane* bagi cluster. Node master berperan mengatur seluruh operasi ayng ada pada cluster termasuk mekanisme *scheduler*, proses deteksi serta pemberian respons terhadap *events* yang berlangsung di dalam cluster (contohnya, penjadwalan pod baru apabila jumlah replika yang ada pada *replication controller* tidak terpenuhi). Node master terdiri dari beberapa komponen, antara lain :

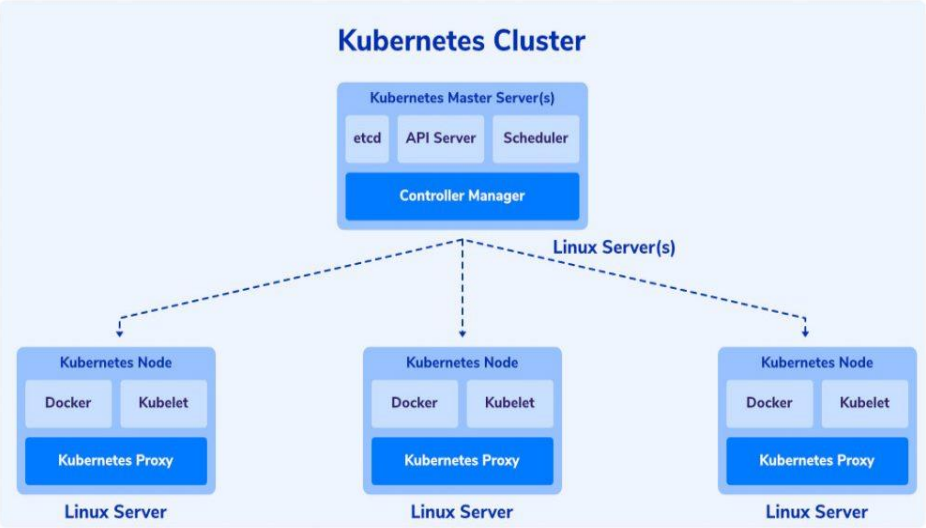
- ***kube-controller-manager*:** bertugas melakukan monitor pada cluster agar sesuai dengan konfigurasi data objek di dalam node.
- ***kube-apiserver*:** sebagai *front-end* dari *control plane* Kubernetes. Komponen ini didesain agar dapat diskalakan secara horizontal.
- ***kube-scheduler*:** menentukan bagaimana events dijadwalkan antara cluster berdasarkan ketersediaan resource, peraturan dari operator, dsb. Bertugas

mengamati Pod baru yang belum ditempatkan di node manapun dan kemudian memilihkan Node di mana Pod baru tersebut akan dijalankan.

- ***etcd*:** storage key value yang digunakan sebagai penyimpanan data klaster Kubernetes.

Worker node berfungsi menjalankan tugas yang diberikan *node master* serta melakukan pemeliharaan terhadap pod dan menyediakan *environment runtime* bagi Kubernetes. *Worker node* terdapat beberapa komponen, yaitu :

- Kubelet : agen yang berjalan pada setiap node dicluster yang bertugas untuk memastikan container berjalan didalam pod.
- Kube-proxy: memelihara aturan-aturan jaringan (*network rules*) serta meneruskan koneksi yang ditujukan pada suatu host.
- Container Runtime : perangkat lunak yang bertanggung jawab dalam menjalankan kontainer. Kubernetes mendukung beberapa runtime, diantaranya adalah: Docker, containerd, cri-o, rktlet dan semua implementasi Kubernetes CRI (Container Runtime Interface).



Gambar 2. 4 Arsitektur Kubernetes

2.2.12 Kops

Kops adalah *tool* yang ditulis menggunakan Bahasa pemrograman go, kelebihan kops adalah memudahkan persiapan pada tahap produksi dengan menejemenisasi kubernetes cluster yang bisa diterapkan pada layanan *cloud Amazon Web Services* dan *Google Cloud Platform*. Dengan hanya menjalankan suatu perintah kops dapat membuat, upgrade dan *maintain* cluster dengan memberikan izin spesifik untuk mengakses *resource* dari layanan *cloud* yang digunakan.

2.2.13 Apache Jmeter

Apache Jmeter adalah testing tool open source berbasis desktop yang ditulis menggunakan bahasa pemrograman java, Apache Jmeter pertama kali dikenalkan oleh Stefano Mazzocchi yang menjadi salah satu pendiri Apache Software Foundation.

Apache Jmeter sangat membantu dalam menganalisis dan menjadi tolak ukur performa aplikasi web yang terdiri dari beberapa service (Narinder Kaur, 2016). Apache jmeter bisa digunakan sebagai salah satu tool untuk test koneksi database, HTTP, web service, TCP, JMS dan lain-lain.