BAB 2
KAJIAN PUSTAKA DAN DASAR TEORI

2.1 KAJIAN PUSTAKA

Riset Adnan [7] pada tahun 2017 berjudul "Analisis Perbandingan Kinerja *Web Server Apache* dan *NGINX* Menggunakan HTTPerf Pada VPS dengan Sistem Operasi CENTOS", riset ini difokuskan kepada analisa kinerja *web server* serta menggunakan beragam subjek web lalu diterapkan pada VPS memakai sistem operasi CENTOS. Berdasarkan parameter *reply time web server Nginx* lebih cepat saat merespons permintaan kepada subjek *web* statis, gambar, dan *wordpress*. Sedangkan *web server Apache* lebih dominan saat segi *bandwidth* kepada subjek web statis, gambar, PHP, serta toko online.

Penelitian Irza, dkk [8] pada tahun 2017 berjudul "Analisis Perbandingan Kinerja *Web Server Apache* dan *Nginx* Menggunakan HTTPerf Pada Portal Berita (Studi Kasus beritalinux.com)," riset tersebut bertujuan untuk mengobservasi kerja *web server Apache* dan *Nginx* saat digunakan dalam Portal Berita dengan memakai alat *benchmark* HTTPerf. HTTPerf sendiri berfokus kepada penyajian peralatan yang kuat serta berkinerja tinggi yang memungkinkan pengembangan besar serta kecil. Hasil pengujian didapatkan pada respon dan *connecting* data yang direquest oleh *client* dari *server web server Nginx* lebih dominan dari *Apache*. Dari hasil tersebut disarankan kepada admin *website* beritalinux.com untuk menggunakan *web Nginx server* untuk kinerja situs *website* yang lebih baik.

Pada tahun 2019 Chandra, dkk [9] melaksanakan riset dengan judul "Analisis Performasi Antara *Apache* dan *Nginx Web Server* Dalam Menangani *Client Request*". Dalam riset ini diuji kinerja *web server* berlandaskan kesanggupannya saat menjalani *client request* dengan memakai aplikasi *Apache Bench* berat beban dimulai dari 100 pengguna hingga 1.000.000 pengguna. Perolehan uji dari *web server Apache* dan *Nginx* ini dinantikan bisa memberi arahan dalam pilihan *web server* yang nantinya

dipakai saat pembangunan sebuah *website* yang nantinya dipakai bila *website* tersebut akan melayani beragam *client request*.

Penelitian Fandy, dkk [6] pada tahun 2022 melakukan penelitian yang berjudul "Pengujian Kinerja *Web Server* Atas Penyedia Layanan *Elastic Cloud Compute* (EC2) Pada *Amazon Web Services* (AWS)". Pada penelitian tersebut menguji kinerja *web server* telah didirikan pada *elastic cloud compute* di *Amazon Web Services*, beragam instrumen uji yakni *throughput*, *response* time, *latency* serta *resource utilization* menggunakan aplikasi *performance testing* dari *Apache Jmeter*. Berlandaskan perolehan uji kepada *Amazon Web Services* dengan ragam uji *stress testing* serta jenis uji HTTP *Request*, *web server* yang didirikan kepada layanan *Elastic Cloud Compute* mempunyai performa atau kinerja *web server* lebih baik dari penyedia pelayanan lainnya.

Tabel 2.1 menggambarkan perbandingan berlandaskan penguraian kajian pustaka pada riset sebelumnya telah relevan pada riset yang nantinya dilaksanakan.

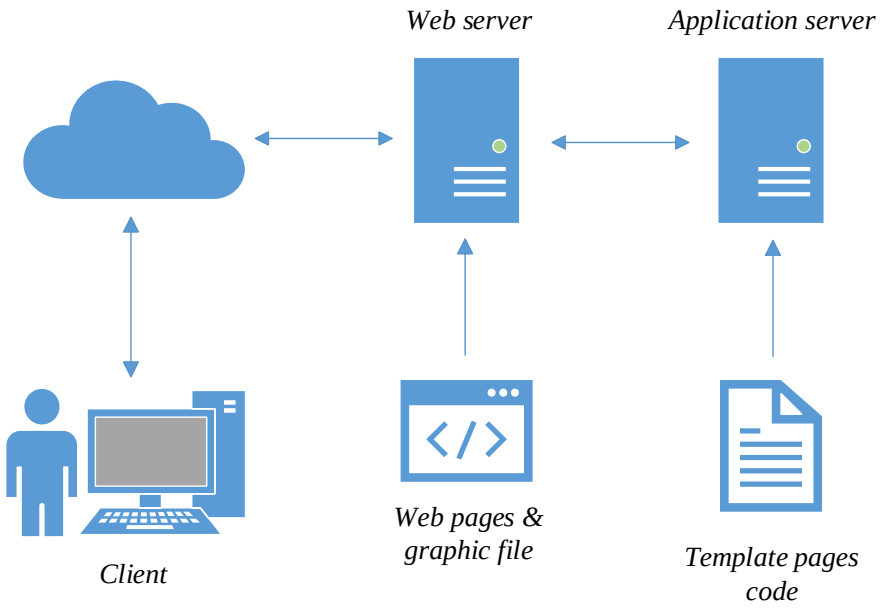
Tabel 2.1 Perbandingan Jurnal Referensi

Tahun	**Penelitian Oleh**	***Nginx & Apache***	***Amazon Web Services***	***Apache Jmeter***	**Parameter QoS**			
					Throughput	***Delay & Jitter***	***Packet Loss***	***Response Time***
2017	Fariq Adnan	√			√			√
2017	Intan Ferina Irza, dkk	√			√		√	
2019	Albert Yakobus Chandra, dkk	√						
2022	Fandy, dkk		√	√	√			√
2023	Logi Yosica Fulqi	√	√	√	√	√	√	√

2.2 DASAR TEORI

2.2.1 Pengertian dan Perancangan Jaringan

Di bawah ini ialah bentuk topologi jaringan web server yang biasanya dipakai oleh *client*.



Gambar 2.1 Perancangan Jaringan

Dalam gambar 2.1 merupakan sebuah susunan jaringan *web server* yang umumnya dipakai dalam berinteraksi antar pengguna melalui internet. Awalnya *client* melakukan akses apabila ingin menggunakan *web server* baik dalam *Apache*, *Nginx*, maupun *litespeed* dengan lewat suatu *router* yang terkoneksi terhadap internet. Melalui *router* ini *client* mampu me-*request* pada *web server* dalam membuat sebuat *portal*. *server* nantinya merespon permintaan klien guna mampu masuk kedalam sebuah *website* yang mereka minta.

Jaringan komputer ialah suatu aturan meliputi dari dua/lebih komputer telah memiliki hubungan satu dengan lainnya melalui media yang terhubung nantinya bisa bertukar data melalui *platform* maupun beragam *hardware*. Definisi jaringan komputer itu biasanya didefinisikan apabila terkumpulnya beberapa terminal komunikasi meliputi dari 2 buah perangkat/lebih yang saling berhubungan.

Tujuan dibentuknya jaringan komputer ini ialah supaya data yang dimiliki oleh yang mengirim bisa diperoleh pada yang menerima secara

efisien serta cepat. Adanya jaringan ini mengizinkan penggunanya mampu berinteraksi dengan yang lain dengan mudah. Dan juga nantinya peranan jaringan komputer ini begitu diperlukan dalam memadukan data dari komputer *client* hingga didapati sebuah data yang valid.

2.2.2 *Web Server*

Web server ialah sebuah perangkat lunak yang merupakan tulang akar dari *world wide web* (www). *Web server* menanti minat dari *client* nantinya memakai browser seperti *netscape navigator*, *Mozilla*, serta program *browser* lain. Apabila terdapat *request* dari *browser*, nantinya *web server* akan menjalankan keinginan tersebut dan memunculkan perolehan yang merupakan data yang diminta lagi ke *browser*. Data tersebut memiliki acuan yang ditetapkan yang biasanya bernama SGML/standar general markup *languange*. Data format ini nantinya disajikan *browser* berimbang pada fungsi dari *browser* ini.

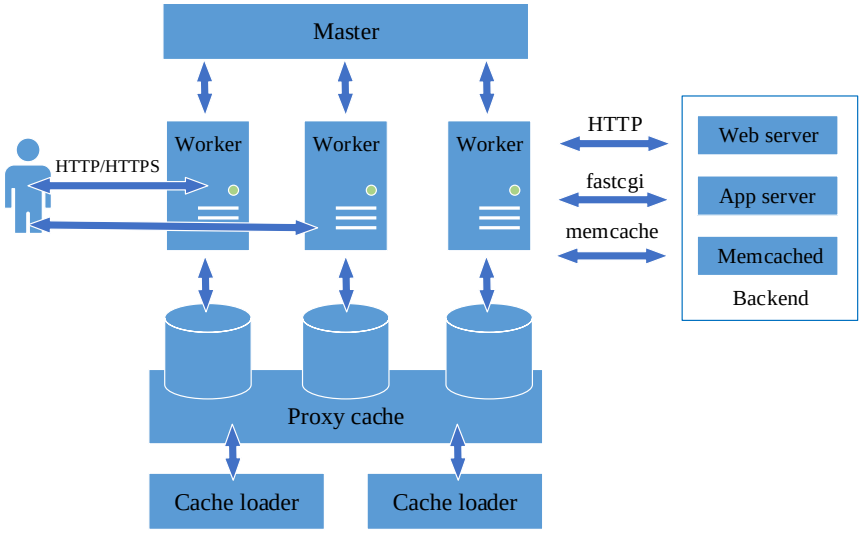
Web server juga dikatakan sebagai aplikasi yang menyajikan layanan data untuk request HTTP (*Hypertext transfer protocol*) sudah diinginkan pelanggan lewat *browser*. *Web server* bermanfaat dalam menjadi wadah aplikasi *web* dan sebagai penerima permintaan dari klien [10]. Dikatakan saat ini terdapat berbagai macam *web server* dengan performa dan skill yang beragam seperti *Apache*, *Nginx*, *light http*, dan lain sebagainya. Akan tetapi untuk *web server* telah peneliti bahas pada riset tersebut hanya membahas dua *web server* yaitu *Nginx* serta *Apache*.

A. *Nginx*

Nginx atau pada umumnya dalam khalayak biasanya menyebutnya dengan *Engine* X merupakan sebuah *open source web server*. Nginx dibuat oleh Igor Sysoev dan dirilis ke publik pada bulan Oktober 2004, *Nginx* adalah sebuah jawaban untuk mengatasi permasalahan yang ada pada saat itu yaitu permasalahan performa *web server* jika memiliki koneksi aktif lebih dari 10.000 koneksi secara bersamaan. Nginx menawarkan penggunaan memori yang lebih rendah dibandingkan web server lainnya . *Nginx* memberikan tawaran berupa fitur yakni *reverse proxy*, IPv6, *load balancing*, FastCGI *support*, *web*

sockets, *handling static files*, TLS/SSL dan juga pemakaian penyimpanan yang sedikit daripada *web server* lainnya. Meskipun dikatakan merupakan *web server* yang masih baru akan tetapi penggunanya seterusnya menjalani kenaikan beberapa tahun.

Dibandingkan *web server* lainnya yakni *Apache*, *Nginx* dikatakan lebih konstan serta irit *resource*, hingga banyaknya *web master* yang mempunyai klasifikasi *server* belum begitu besar memilih memakai *Nginx* dibanding *web server* free yang lain. Lain halnya sistem operasional *Linux* yakni *Centos*, *Red Hat*, *Debian* serta *Ubuntu*, *Nginx* juga didukung beragam jenis dari sistem operasi *Windows* serta *Solaris*.



Gambar 2.2 Arsitektur *Open Source Nginx*

Pada gambar 2.2 dijelaskan bahwa *Nginx* memakai sistem asinkron serta *event driven* yang dimana *software* ini nantinya memproses permintaan hanya dalam satu dasar pemanfaatan CPU saja, dan tidak membuat permintaan itu menjadi sebuah prosedur. *Nginx* memiliki peraturan *worker* process disaat memproses keseluruhan *thread* yang dimana terdapat didalamnya *worker connection* dengan bentuk lebih kecil yang memiliki tugas guna memproses permintaan *thread* didalam *worker process* [11].

B. *Apache*

Apache web server ialah *Open source* yang dibuat serta diatur oleh *Apache*.org yang merupakan susunan *web server* teratas telah

dikenal serta cukup ramai penggunanya sekarang. Dikatakan *Apache* setidaknya memiliki kuasa berkisar 40 persen total seluruh pasaran *web server* di internasional, dengan banyak portal yang melebih 330 juta portal[12]. Dalam hal ini dikatakan juga yang mana menaruh *Apache* menjadi *web server* telah begitu dipertimbangkan walau dibedakan oleh *web server* yang bertarif.

Walaupun mulanya *Apache* cuma dirancang dalam hal peruntukkan prosedur di lingkup unix, akan tetapi hal ini berkembang. *Apache* juga dikatakan mendukung sistem *operasi windows*. Terdapat dorongan dalam beragam sistem operasi yang sifatnya tidak bayar serta pemakaian yang ringan akhirnya menjadi daya tarik bagi *Apache* dimata banyaknya *web master* guna memakai *web server* buatan *Apache software foundation* tersebut.

Server HTTP *Apache* merupakan *server web* bisa dipakai dibanyak sistem operasi yang bermanfaat guna menjalani serta berfungsi guna portal. *Protocol* yang dipakai guna memfasilitasi *web*/www tersebut memakai HTTP. Apache menyediakan berbagai macam modul *multiprocessing* (biasa di sebut MPMs) yang menentukan bagaimana permintaan klien ditangani. Pada dasarnya, hal ini memungkinkan administrator untuk menukar koneksi penanganan arsitektur dengan mudah.

1. Mpm_*prefork*

   Modul *prefork* ini membuat *child* proses dan tiap *child* proses hanya mempumyai satu *thread* dalam waktu yang sama (*non-threaded*). Modul MPM *prefork* menurut *Apache* cukup stabil dalam melakukan proses, karena tiap *request* ditangani satu *child* proses yang terisolasi dari yang lainnya.

   Efek sampingnya adalah penggunaan dana yang lebih besar pada *thread* yang terjadi bersama sama (*concurrent request*), jika trafik *web* sangat besar penggunaan modul ini akan membuat proses lebih lama, solusinya bisa melakukan perubahan pada *Max Request Worker* sesuai dengan kemampuan sumber daya *server*. Perhatikan

juga mod_php yang merupakan *non-thread safe library* hanya bekerja dengan MPM *Prefork*, jika membutuhkan mod_php gunakan MPM_*Prefork*.

2. Mpm_*worker*

Modul MPM *Worker* ini dapat mengubah proses *Apache* menjadi multi proses. Modul MPM Worker tidak seperti modul MPM *Prefork*, tiap *child* proses modul ini bisa menampung banyak *thread*. Membuat proses yang terjadi bersama sama (*concurrent request*) bisa ditangani dengan cepat. Modul ini direkomendasikan untuk menangani web dengan trafik yang cukup besar. Modul MPM *worker* juga menggunakan sumber daya RAM yang lebih sedikit. MPM *worker* hanya disupport pada Apache 2.4.x untuk versi di bawahnya modul ini bersifat eksperimen. Untuk cek versi *Apache* bisa menggunakan *command* http -V *grep version*.

3. Mpm-*event*

Modul MPM *Event* harus digunakan pada *Apache* versi 2.4 keatas, karena modul tersebut akan membuat peforma *server* lebih efisien dalam penggunaan sumber daya seperti RAM dan lebih cepat dalam melakukan proses. MPM *event* mempunyai kemampuan yang lebih luas lagi dari modul MPM *Worker*, setiap *child* proses mempunyai banyak *thread* secara bersama sama (*concurent*) dan tiap *thread* bisa melakukan *multi* proses.

C. Perbandingan *Web Server Nginx* dan *Apache*

Peneliti merangkum perbedaan diantara *web server Nginx* serta *Apache* yang tersajikan pada tabel 2.2.

Tabel 2.2 Perbandingan *Web Server*

Keterangan	Kelebihan	Kekurangan
Nginx	a. Memiliki kesanggupan saat mengatur sebuah trafik yang besar. b. Mempunyai skalabilitas yang baik. c. Dapat diganti *Hardware Load Balancer*.	a. Peng-update-an relatif lama berbeda dengan web *server* lain. b. *Fast* CGI belum berfungsi baik. c. Kadang kala terdapat permasalahan di.htaccess tapi ini

Keterangan	Kelebihan	Kekurangan
	d. memiliki *tool* yang banyak fungsi. e. Tersedia berbagai macam dokumentasi lengkap.	bergantung dari konfigurasi *web* masternya tersebut d. Pemakai tidak sebanyak *Apache* / IIS.
Apache	a. Kompatibel dengan *WordPress*. b. Lintas platform. c. Ada komunitas besar dan instruksi yang lengkap. d. Menggunakan lisensi gratis yang bersifat *open source*. e. Software yang handal serta konstan. f. Konfigurasi yang ringan serta tidak sulit g. *Patch* yang berkaitan dengan keamanan wajib diperbarui.	a. Keamanan rapuh karena beragam konfigurasi belum berkembang dengan sempurna b. Jika banyak pengunjung masuk, akan ada gangguan atau masalah dengan kinerja *website*.

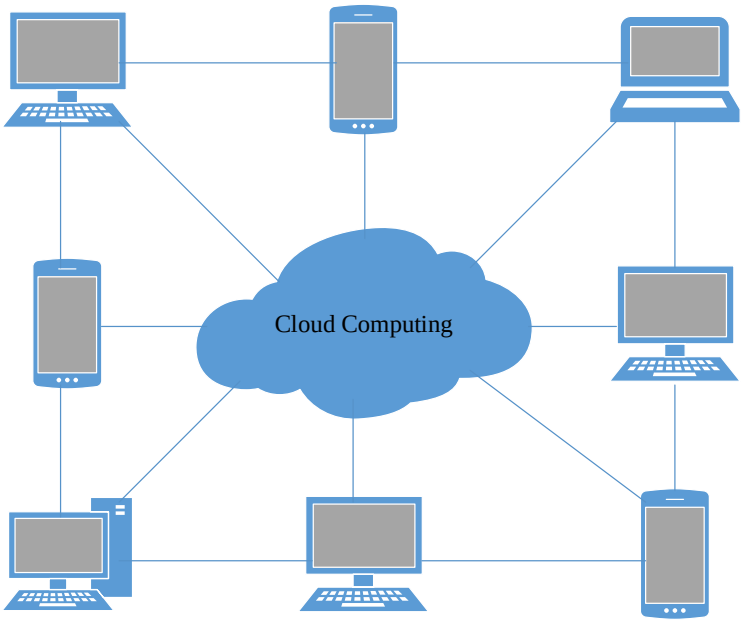
2.2.3 *Cloud Computing*

A. Pengertian *Cloud Computing*

Sebuah model *cloud computing* memungkinkan akses jaringan yang mudah dan bisa diakses kapan saja dan di mana saja ke seperangkat peralatan komputasi (yakni jaringan, *server*, harddisk, aplikasi, serta layanan). *Cloud computing* adalah teknologi informasi telah memungkinkan orang menggunakan jaringan atau internet untuk mengakses software, informasi, serta platform serta sumber daya bagi komputer lainnya yang membutuhkannya. Kata "*cloud*" serta "*computing*". "*cloud*" diartikan internet lalu "*computing*" berarti prosedur komputasi.

Dikarenakan internet sendiri diumpamakan menjadi awan (*cloud*) besar (biasanya digambarkan sebagai "awan" pada skema jaringan) yang berisikan bermacam komputer yang saling berhubungan, konsep *cloud computing* biasanya dianggap bagaikan internet. *Cloud computing* datang sebagai evolusi yang menunjukkan perpaduan aplikasi dan teknologi yang lebih dinamis. Jika ada perubahan besar, itu akan

berdampak pada hampir semua aspek komputasi. Konsep *cloud computing* dijelaskan pada gambar 2.3.



Gambar 2.3 Konsep *Cloud Computing*

Pada dasarnya, salah satu manfaat *cloud computing* adalah kemampuan untuk memberikan skalabilitas. Skalabilitas adalah proses meningkatkan kapasitas *cloud computing* belum menggunakan alat tambahan seperti hardisk dikarenakan *cloud computing* sudah tersedia layanan/sumber daya guna melakukannya.

Inovasi berawal dari *cloud computing* sejak 1960-an, saat John McCarthy, ahli komputasi MIT yang disebut menjadi pelopor intelegensi tiruan, mengutarakan visinya bahwasannya "suatu saat komputasi bisa menjadi infrastruktur luas, sama halnya dengan listrik serta telepon". Tapi sejak 1995 lah, Larry Ellison, pelopor *Oracle*, membuat terobosan "*Network Computing*" menjadi pergerakkan guna menuntut dominasi *Microsoft* saat itu unggul dalam *desktop computing* oleh Windows 95-nya.

Larry Ellison menjajakan inovasi bahwasannya *user* tidak membutuhkan beragam *software*, diawali dari Sistem Operasi serta beragam *software* lainnya, dicoba pada PC *Desktop* mereka. PC *Desktop* dapat tergantikan oleh sebuah terminal yang spontan berhubungan pada *server* yang terdapat *environment* serta berisikan beragam keperluan

software yang siap digunakan oleh pemakainya. Disebutkan juga ada empat model pengembangan cloud yaitu sebagai berikut:

1. *Public Cloud*

   Jenis *Cloud* ini dikhususkan bagi umum oleh penyediaan fasilitas.

2. *Private Cloud*

   ialah infrastruktur fasilitas *Cloud*, yang dijalankan sebagai sebuah Lembaga terkait. Infrastruktur *Cloud* dapat diatur dari Lembaga/pihak ke-3. Tempatnya dapat on-*site*/off-*site*. Namun Lembaga berskala besar saja yang dapat mempunyai dan mengatur *private Cloud* ini.

3. *Community Cloud*

   Pada bentuk ini, sebuah infrastruktur *Cloud* dipakai secara simultan oleh beragam lembaga yang mempunyai kesamaan keperluan, contoh dari segi yang dituju/tingkatan rasa aman yang diperlukan, dll..

4. *Hybrid Cloud*

   Dalam ragam ini , infrastruktur *Cloud* yang ada ialah pencampuran dari dua/lebih infrastruktur *Cloud* (*private*, *community*, atau *public*). namun secara entitas mereka bisa berdiri mandiri, namun terhubung dalam teknologi/mekanisme yang mungkin terjadi portabilitas data serta perangkat antar *Cloud* itu. Misalnya, prosedur *loadbalancing* yang antar *Cloud*, hingga pengalokasian kemampuan dalam mempertahankan pada level yang Optimal.

B. Karakteristik *Cloud Computing*

Lima karakterisrik yang wajib terpenuhi sebuah sistem untuk dapat diikut sertakan pada rumpun *Cloud Computing*, yaitu sebagai berikut:

1. *On Demand Self Service*

   Pemberli berkemungkinan secara spontan “memesan” sumber daya yang diperlukannya, contoh *processor time* dan berkapasitas menyimpan lewat *control panel* elektronis yang telah tersedia, hingga tidak memerlukan interaksi antar personil

customer service bila memungkinkan bertambah/berkurang sumberdaya komputasi yang dibutuhkan

2. *Broadband Network Access*

   Layanan yang ada berkaitan lewat jaringan pita lebar, yang utama agar bisa membuka dengan optimal lewat jaringan internet, baik memakai thin client, thick client maupun sarana lainnya contohnya: *smartphone*.

3. *Resource pooling*

   Kesediaan layanan *cloud*, dalam pemberian layanan lewat sumberdaya telah diklasifikasikan di suatu tempat atau beragam tempat pusat data yang meliputi beragam server dengan prosedur multi-tenant. Prosedur multi-tenant ini berkemungkinan sebanyak sumberdaya komputasi ini dipakai secara simultan oleh beberapa *user*, yang mana sumberdaya itu dapat berupa fisik/virtual, sehingga bisa ditempatkan secara dinamis guna kebutuhan pemakai sesuai permintaan.

4. Elastis (*Rapid elasticity*)

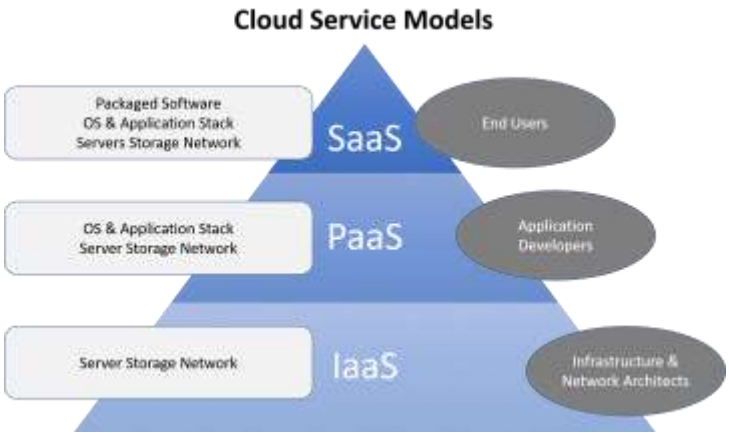
   ruang komputasi yang tersedia bisa secara elastis serta mudah diadakan baik itu saat berbentuk penambahan/pengurangan kapasitas yang dibutuhkan. Untuk pembeli itu sendiri, dengan keahlian ini beranggapan bahwa kapasitas yang disediakan unlimited besarannya serta dapat "dibeli" kapan saja dengan jumlah berapa saja.

5. *Measured Service*

   Sumber daya *cloud* yang ada wajib dikelola serta dioptimalkan pemakaiannya saat pengukuran sistem yang dapat diukur pemakaian sumberdaya komputasi yang dipakai(ruang menyimpan, *memory*, *processor*, lebar pita, aktivitas *user*, dan lainnya). Berikut ini keseluruhan sumberdaya yang dipakai secara transparan dihitung sehingga menjadi landasan pemakai guna menuntaskan biaya pemakaian layanan.

C. Model Layanan *Cloud Computing*

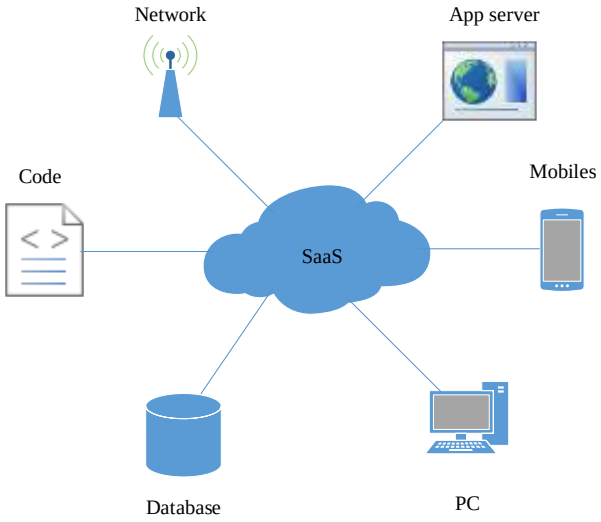
Berdasarkan macam layanan *cloud computing* dibagi menjadi 3 model layanan seperti pada gambar 2.4 [13].



Gambar 2.4 Model *Cloud Computing*

1. *Software as a Service*

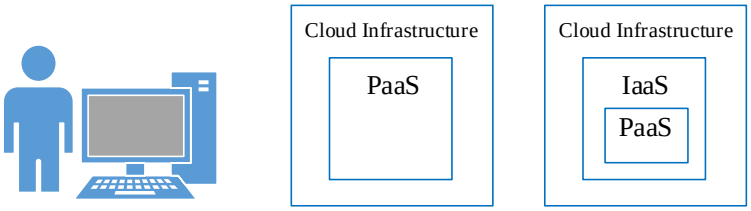
Software as a Service (SaaS) tersedia fasilitas meliputi aplikasi yang bisa dipakai pelanggan yang sesuai dengan konstruksi *cloud*. Contoh kesediaan layanan SaaS ialah *gmail*, *google docs*, office 365 serta *SalesForce*. Konsepsi bentuk fasilitas Saas dijelaskan pada gambar 2.5.



Gambar 2.5 Konsep Model Layanan SaaS

2. *Platform as a Service*

Platform as s Service (PaaS) Tersedia fitur (Bahasa pemrograman, *Tools*, *Web server*, *database*) yang bermanfaat guna perkembangan aplikasi yang sejalan dengan konstruksi *cloud* serta outputnya bisa bermanfaat bagi pelanggan. Contoh kesediaan fasilitas PaaS, *OpenShift*, *PHPCloud*, *AppFog*, *Heroku* serta *GoogleApp Engine*. Konstruksi layanan PaaS diterangkan pada gambar 2.6.



Gambar 2.6 Arsitektur Model Layanan PaaS

3. *Infrastructure as a Service*

Infrastructure as a Service (IaaS) kesediaan sumber daya memproses, *storage*, kapasitas jaringan, serta sumber daya komputasi lainnya. contohnya penyediaan layanan *Amazon* EC2 dan Telkom*Cloud*. prinsip kerja IaaS diterangkan pada gambar 2.7.

Gambar 2.7 Konsep Kerja IaaS

D. Penyedia layanan *cloud computing*

1. *Amazon Web Service*

Dengan menjadi pemasok fasilitas *cloud* yang terkendali, *Amazon Web Services* (AWS) menjajakan bantuan komputasi, ruang

simpanan *database*, "jaringan pengiriman konten", serta konten yang ditambah agar mempermudah banyak badan usaha kembangkan serta menjalani aplikasi mereka sebaik mungkin [14]. AWS memberi beragam produk yang mempermudah pendirian usaha *Infrastructure as a service*, merupakan sebutan yang pas untuk AWS. Definisi lainnya terdapat pula *Software as a service* dan *Platform as a service*.

beragam produk AWS sudah di susun agar mempermudah pemakaian , seperti *Elastic Compute Cloud* dan EC2. EC2 pada dasarnya adalah *server virtual*, serta keringanan yang dijajakan oleh EC2 terlihat ketika sedang membandingkan pada *server* konvensional, kita perlu mempelajari dengan detail klasifikasi *server* yang diharapkan. seperti *support*, *spare part*, serta penyimpanan di *data center*, yang merupakan prosedur yang sulit. Bila disiapkan begitu banyaknya server meskipun keperluan komputasi belum besar, kita akan menyiakan banyaknya anggaran. Namun, bila kita siapkan sedikitnya *server* meskipun keperluan komputasi sangat besar, bisa mengganggu operasi usaha

Beragam keunggulan AWS terkait menu "*pay as you go*", yang berarti bayar saat digunakan. Model *pricing* per jam dipakai guna macam *deployment* on-*demand*, seperti AWS EC2. Selanjutnya, Kinesis Streams memiliki *shard hour*, PUT *Payload Unit*, serta pengumpulan data yang diperluas. Kita juga memiliki layanan yang kuat, seperti kinesis *streams*, dengan biaya yang cukup murah, dan kita dapat menuntaskan permasalahan yang sama dengan waktu sedikit tanpa perlu mengawasi aplikasi streaming seperti *Apache Kafka*. Ini berarti kita dapat menuntaskan menu dengan cepat, lebih fokus pada pengguna dan bisnis, dan tidak perlu memikirkan "*scaling*". Ada dua jenis AWS yang ada pada *web amazon* ialah *amazon elastic container service* dan *amazon fargate* yang dikemukakan sebagai berikut.

a. *Amazon Elastic Container Service*

Amazon Elastic Container Service (*Amazon* ECS) ialah fasilitas orkestrasi kontainer yang sepenuhnya dikelola [15]. ECS dipakai langganan seperti *Duolingo*, *Samsung*, GE, dan *Cookpad*

guna pemakaian di aplikasi sensitif dan tugas penting dikarenakan rasa aman, handal serta skalabilitasnya. Untuk beberapa alasan, ECS adalah pilihan yang bagus guna menggerakan kontainer. Pertama, pengguna bisa menggerakan klaster ECS memakai AWS *Fargate*, yang termasuk komputasi kontainer tidak dengan *server*. *Fargate* mengurangi penyediaan dan pengelolaan *server*, dimungkinkan Anda menetapkan dan menuntaskan ke sumber daya per aplikasi, dan menaikkan rasa aman dengan memisahkan aplikasi berdasarkan desain. Kedua, ECS dipakai beragam kalangan pada *Amazon* guna mendorong fasilitas seperti *Amazon SageMaker*, AWS *Batch*, *Amazon Lex*, dan mesin pencarian *Amazon.com*, yang menunjukkan bahwa ECS pengujian di uji secara keseluruhan untuk keamanan, keandalan, dan ketersediaan[15].

lalu, dikarenakan ECS telah jadi tombak utama, fasilitas utama *Amazon*, ECS bisa berintegrasi natif dengan fasilitas lainnya seperti *Amazon Route* 53, *Secrets Manager*, AWS *Identity* and *Access Management* (IAM), dan *Amazon CloudWatch*, yang memberikan Anda wawasan luas saat mengimplementasikan serta skala kontainer Anda. Selain itu, ECS bisa dengan mudah berintegrasi dengan fasilitas AWS lainnya guna terdorong ke kapabilitas baru ke ECS. Misalnya, ECS terintegrasi dengan AWS *App Mesh*, yang memakai mesh layanan, serta memberi fitur observabilitas, kontrol lalu lintas, serta keamanan yang kaya ke aplikasi Anda. ECS juga memberi fleksibilitas kepada aplikasi Anda dimungkinkan pemakaian campuran *Amazon* EC2 dan AWS *Fargate* dengan pilihan anggaran *Spot* dengan sesuai minatnya. Dibandingkan dengan instans peluncuran EC2, ECS telah berkembang dengan lancar saat peluncurannya dan sekarang peluncurannya lima kali lebih banyak kontainer setiap jam [15].

b. *Amazon Fargate*

AWS *Fargate* adalah mesin komputasi tidak ada *server* guna *container* yang bermanfaat apda *Amazon Elastic Container Service*

(ECS) dan *Amazon Elastic Kubernetes Service* (EKS) [16]. *Fargate* memudahkan berfokus kepada pembentukan aplikasi. *Fargate* mengurangi adanya kesediaan serta mengolah *server*, kemungkinan Anda menetapkan dan menuntaskan sumber daya per aplikasi, dan menaikkan rasa akan lewat isolasi aplikasi berdasarkan desain.

Fargate mengalokasikan jumlah komputasi yang tepat, menghilangkan kebutuhan instans, dan meningkatkan skala kapasitas klaster. Anda hanya membayar sumber daya yang diperlukan untuk menjalankan container Anda, sehingga Anda tidak perlu membayar *server* tambahan untuk sumber daya. *Fargate* menjalankan setiap *pod* atau tugas di dalam kernelnya sendiri, yang memberikan lingkungan komputasi terpisah untuk *pod* dan tugas. Ini memungkinkan aplikasi memiliki beban kerja yang terpisah dan peningkatan keamanan yang sesuai dengan desain mereka. Pelanggan seperti *Vanguard*, *Accenture*, *Foursquare*, dan *Ancestry* memilih untuk menggunakan *Fargate* untuk aplikasi utama mereka[16].

2. *Alibaba Cloud*

Alibaba cloud adalah penyedia komputansi awan dengan layanan yang dirancang untuk berbagai ekosistem e-*commerce*. Layanan yang disediakan terkait penyediaan pengolahan data dan analisis data, layanan *web*, *cloud storage*, *big data*, dan *Internet of Things* (IoT). Pengguna Alibaba *cloud* yang menggunakan layanan *database cloud* secara otomatis menyimpan semua informasi dan data yang dikirim ke *database Alibaba cloud* [17].

Dalam menyediakan platform cloud yang berkualitas, Alibaba cloud meluncurkan berbagai fitur yang bisa dinikmati oleh pengguna, berikut adalah beberapa fitur yang disediakan oleh *alibaba cloud*:

a. Polar DB

Polar DB adalah *database* relasional yang kompatibel dan dirancang khusus untuk cloud computing yang menggabungkan ketersediaan dan kinerja enterprise basis data tradisional dengan

menggunakan biaya yang lebih murah. Teknologi ini menjanjikan skalabilitas, solusi penyimpanan yang disediakan oleh *Alibaba cloud* bisa di upgrade sampai 100TB. Dengan seperti itu, pelangga akan lebih mudah mengatur perkembangan data dalam skala yang besar.

b. *Alibaba Log Service*

Fitur *Alibaba log service* biasa dikenal dengan sebutan SLS, sebuah fitur yang memungkinakan suatu perusahaan untuk mengumpulkan *log* secara otomatis dari seluruh layanan aplikasi pada bagian dalam maupun luar *cloud*. SLS memungkinkan proses penandaan *log* data dari *context derived*. Fitur ini juga bisa digunakan untuk pengambilan keputusan yang dilakukan secara *real time* dan digunakan untuk menganalisis hal yang sedang ramai dibicarakan, membuat peringatan, serta visualisasi.

3. *Google Cloud*

Google cloud memiliki banyak layanan yang fungsinya sebagai *cloud computing* dan dapat digunakan untuk merancang bangun infrastruktru *server* yang memiliki tingkat reliabilitas dan *availability* yang tinggi, dimana pada setiap produk akan memiliki fungsi dan kelebihan yang berbeda pada masing-masing fitur. Beberapa layanan produk yang disediakan oleh google cloud sebagai pendukung untuk merancang bangun infrastruktur *server* adalah *compute engine*, *cloud* SQL, *cloud storage*, dan *container registry*. Dimana produk tersebut memiliki beberapa metode dan fitur yang bisa diterapkan untuk membangun infrastruktur high availability pada *server* seperti fitur *autohealing*, *multiple zones*, *load balancing*, *autoscaling*, *automatic*, *updating*, dan *failover* [18].

a. *Google compute engine*

Kelebihan dari fitur *google compute engin*e adalah pengaturan komputer secara *real time* dengan dukungan jaringan yang sangat kuat. *Google compute engine* merupakan fitur yang memungkinkan adanya mesin *virtual* instan untuk melakukan

hosting terhadap beban kerja para pengguna. Layanan ini biasa disebut sebagai *infrastructure as a service* atau IaaS.

b. *Google app engine*

Fitur google app engine memberikan kesempatan pengguna melakukan pengembangan terhadap layanan milik mereka sendiri menggunakan kerangka kerja milik *Google* yang dikenal sebagai *platform as a service* atau PaaS. Pada fitur ini terdapat *software development* kit atau SDK yang bisa dengan mudah digunakan oleh pengguna untuk membangun sebuah aplikasi yang bisa berjalan di *App Engine*.

c. *Google cloud storage*

Fitur *google cloud storage* mampu memberikan kebebasan bagi para pengguna untuk melakukan penyimpanan data pada *server Google*. Bahkan secara keseluruhan jenis data bisa disimpan dengan baik menggunakan layanan ini. Penyimpanan *database* dapat tersimpan dalam skala penyimpanan yang lebih besar pada platform unggulan seperti *cloud datastore, cloud* SQL, dan *cloud bigtable google*.

d. *Google container engine*

Google container engine adalah fitur yang bisa dimanfaatkan untuk peningkat produktivitas bagi para pengguna. Layanan yang diberikan fitur ini dapat memberikan kemudahan terutama pada saat membangun aplikasi dan menyebarkannya melalui layanan *cloud*.

E. Perbandingan layanan *Cloud Computing*

Tabel 2.3 menjelaskan tentang perbandingan layanan *cloud* computing [19].

Tabel 2.3 Perbandingan Layanan *Cloud Computing*

Jenis Layanan	Kelebihan	Kekurangan
Amazon Web Services (AWS)	a. Menyediakan layanan gratis selama satu tahun yang bisa diakses oleh pengguna.	a. Harga layanan AWS mahal.

Jenis Layanan	Kelebihan	Kekurangan
	b. Sering mengadakan seminar atau webinar secara resmi. c. AWS menyediakan cakupan pilihan area yang cukup luas, yaitu bisa memilih 14 region. d. Banyak fitur layanan AWS yang bisa dipakai untuk membangun produk atau startup sesuai kebutuhan layanan.	
Alibaba Cloud	a. Memiliki penyimpanan yang cukup variatif, dari blok, objek, hingga *file*. b. Jumlah CPU VM lebih tinggi. c. Penyediaan harga yang lebih baik.	a. Cakupan *Alibaba* sempit. b. Layanan pelanggan yang belum efisien c. Fungsional yang tidak transparan.
Google Cloud Platform (GCP)	a. Infrastruktur *server* lebih simple, cocok untuk pemula. b. Memiliki kecepatan *loading* yang bagus. c. Akses GCP lebuh fleksibel bagi pengguna.	d. Tidak mendukung sending email, jika pengguna membutuhkan itu maka perlu mengeluarkan budget lagi.

2.2.4 Aplikasi Pengujian dan Monitoring jaringan

A. *Apache Jmeter*

Ada berbagi alat yang digunakan untuk melakukan *load testing* (pengujian beban) dari sebuah *website*, Menurut Rabiya Abbas *Apache JMeter* merupakan alat terbaik dalam menganalisis kinerja dari sebuah *website* [20]. *Apache JMeter* adalah perangkat lunak sumber terbuka yang sepenuhnya menggunakan bahasa pemrograman *Java*. Perangkat lunak ini dirancang khusus untuk melakukan pengujian perilaku fungsional dan mengukur kinerja. Meskipun awalnya dikembangkan untuk pengujian aplikasi *web*, *Apache JMeter* telah diperluas untuk mendukung berbagai jenis pengujian lainnya. Sebagai aplikasi *open source* yang berbasis *Java*, *Apache JMeter* dapat digunakan oleh QA Engineer untuk melakukan pengujian beban atau stres pada aplikasi *web*, aplikasi FTP, dan pengujian *server database*. Dengan menggunakan *Apache JMeter*, pengguna dapat mensimulasikan beban yang berat pada *server*, sekelompok *server*, jaringan, atau objek lainnya untuk menguji kekuatan sistem atau menganalisis kinerja secara menyeluruh dari berbagai jenis beban [6].

B. *Wireshark*

Wireshark adalah sebuah aplikasi yang digunakan sebagai alat untuk menganalisis paket data yang sedang berlangsung dalam jaringan. Wireshark juga dikenal sebagai pengamat paket jaringan yang berfungsi untuk menampilkan informasi lengkap dari setiap paket dan menangkap paket-paket yang dikirim dan diterima. Pengamat paket jaringan dapat digunakan untuk memantau berbagai jenis jaringan, baik yang menggunakan kabel maupun yang bersifat nirkabel. Dengan menggunakan *Wireshark*, administrator jaringan dapat lebih mudah mengawasi jaringan karena data yang ditangkap oleh *Wireshark* dapat disimpan dan dibuka kembali untuk dianalisis [21].

2.2.5 Pengujian Kinerja

Web Performance Test adalah serangkaian proses pengujian untuk mengukur kinerja perangkat lunak aplikasi *web server* [22]. *Load testing* adalah suatu metode dalam uji kinerja yang digunakan untuk mengukur bagaimana respons sistem terhadap berbagai tingkat beban atau bekerja dalam berbagai kondisi beban. Dalam pengujian ini, sistem atau perangkat lunak diuji dengan menghadapkan pada jumlah pengguna atau permintaan yang beragam secara bersamaan. Tujuannya adalah untuk mengidentifikasi batas kemampuan sistem, menemukan titik-titik lemah, dan memastikan bahwa aplikasi atau situs web tetap berkinerja baik meskipun dihadapkan pada situasi penggunaan yang padat [23].

Bagian dari pengujian kinerja adalah pengujian kapasitas sistem untuk menangani permintaan atau permintaan. Tujuan dari pengujian kapasitas dan kinerja biasanya adalah untuk mengetahui respons waktu atau *latency*, *troughput*, penggunaan sumber daya, dan *workload* [24]. Ada beberapa parameter yang diukur dalam sebuah pengujian kinerja *web server* yaitu sebagai berikut.

A. *Throughput*

Jumlah *bit* atau data yang sukses diterima melalui sistem atau media komunikasi dalam selang waktu pengamatan tertentu disebut *throughput*. Faktor utama *throughput* adalah ketersediaan *bandwitch*

yang cukup untuk menjalankan aplikasi. Ini menentukan jumlah trafik yang dapat diterima suatu aplikasi saat melalui jaringan, sehingga *throughput* yang lebih besar akan lebih baik [25]. Rumus perhitungan *throughput* ditunjukkan dengan persamaan 1.

$$Throughput = \frac{Paket\ data\ diterima\ (bytes)}{Lama\ pengiriman\ (\text{sec})} \quad (1)$$

Tabel 2.4 menunjukkan pengelompokan standarisasi parameter *throughput* berdasarkan standarisasi TIPHON.

Tabel 2.4 Tabel Kategori Nilai *Throughput* TIPHON

Kategori	*Troughput*	Indeks
Sangat Bagus	>2,1 Mbps	4
Bagus	1200 kbps – 2,1 Mbps	3
Cukup	700-1200 kbps 2	2
Kurang Bagus	338-700 kbps 1	1
Buruk	0-338kbps	0

B. *Delay*

Tempo yang dibutuhkan paket data untuk bergerak dari *client* ke *server* atau sebaliknya disebut *delay*. Parameter *delay* adalah faktor yang mempengaruhi jarak fisik antara dua titik dan waktu proses yang lebih lama. Rumus perhitungan *delay* ditunjukkan dengan persamaan 2.

$$Delay = \frac{Total\ delay\ (ms)}{Total\ paket\ yang\ diterima} \quad (2)$$

Tabel 2.5 menunjukkan pengelompokan standarisasi parameter *delay* berdasarkan standarisasi TIPHON.

Tabel 2.5 Tabel Kategori Nilai *Delay* TIPHON

Kategori	*Delay*	Indeks
Sangat Bagus	0-150 ms	4
Bagus	150-300 ms	3
Sedang	300-450 ms	2
Buruk	>450 ms	1

C. *Jitter*

Jitter adalah variasi kedatangan paket yang disebabkan oleh panjang antrian, waktu pengolahan data, dan waktu penghimpunan ulang paket di akhir perjalanan *jitter*. Rumus perhitungan *jitter* ditunjukkan dengan persamaan 3.

$$Jitter = \frac{Total\ variasi\ delay}{Total\ paket\ yang\ diterima} \quad (3)$$

Tabel 2.6 menunjukkan pengelompokan standarisasi parameter *jitter* berdasarkan standarisasi TIPHON.

Tabel 2.6 Tabel Kategori Nilai *Jitter* TIPHON

Kategori	***Jitter***	**Indeks**
Sangat Bagus	0 ms	4
Bagus	0-75 ms	3
Sedang	75-125 ms	2
Buruk	125-225ms	1

D. *Packet Loss*

Packet loss adalah parameter yang menggambarkan suatu kondisi yang menunjukkan jumlah total paket yang hilang. Paket yang hilang dapat terjadi karena *collision* dan *congestion* pada jaringan. *Packet loss* merupakan kegagalan transmisi paket data mencapai tujuannya yang disebabkan oleh beberapa kemungkinan, salah satunya adalah karena terjadinya *overload* trafik didalam jaringan. Rumus perhitungan *packet loss* ditunjukkan dengan persamaan 4.

$$Packet\ loss = \frac{(Paket\ data\ dikirim - paket\ data\ diterima)x100\%}{Total\ paket\ yang\ dikirim} \quad (4)$$

Tabel 2.7 menunjukkan pengelompokan standarisasi parameter *packet loss* berdasarkan standarisasi TIPHON.

Tabel 2.7 Tabel Kategori Nilai *Packet Loss* TIPHON

Kategori	*Packet loss*	Indeks
Sangat Bagus	0%	4
Bagus	3%	3
Cukup	15%	2
Buruk	25%	1

E. *Response time*

Response time juga dikenal sebagai waktu tanggap. *Response time* adalah waktu tanggap yang diberikan oleh UI ketika pengguna mengajukan permintaan ke *server*. Secara umum, waktu respons yang baik adalah yang tercepat [25].