

Implementasi Infrastruktur Server Berbasis *Cloud Computing* Untuk *Web Service Berbasis Teknologi Google Cloud Platform*

Nopi Ramsari^{1*}, Arif Ginanjar¹

¹Universitas Nurtanio Bandung, Bandung

*Email korespondensi : nopiramsarihatta@gmail.com

Received: Feb 21, 2022; Accepted Mar 05, 2022; Published Mar 08, 2022

Abstrak. Ketika bisnis atau startup terus berkembang, maka kebutuhan server menjadi suatu kebutuhan dalam melengkapi kebutuhan bisnis di era digital. Server menjadi bagian penting untuk meningkatkan produktivitas, efisiensi, dan penyedia layanan pada bisnis. Karena server sebagai mesin penyedia layanan, tentunya server dituntut untuk senantiasa mampu menyediakan layanan tanpa mengalami kendala pada infrastruktur server web service. Oleh karena itu perlu dirancang dan dibangun Infrastruktur server yang baik sehingga dapat menghasilkan suatu sistem dengan tingkat reliabilitas dan availability tinggi atau dikenal dengan sebutan *high availability server*. Dalam membangun Infrastruktur server tentunya membutuhkan biaya yang tidak sedikit, oleh karena itu salah satu solusi dalam merancang dan membangun sebuah infrastruktur server yang *high availability server* adalah dengan menggunakan Teknologi *cloud computing*. *Cloud Computing* merupakan model pemberian layanan teknologi informasi untuk pengguna secara fleksibel dengan server virtual, skalabilitas besar, dan manajemen layanan. Layanan teknologi informasi ini dapat digunakan oleh organisasi untuk mempermudah menjalankan proses bisnisnya, dengan menggunakan *Cloud Computing*, kita dapat mengakses data atau program dimana saja, kapan saja, dan dengan perangkat apapun. Adapun salah satu penyedia layanan dari *cloud computing* ini adalah Google Cloud Platform (GCP). Infrastruktur server yang akan dibangun pada penelitian ini menggunakan produk Compute Engine, Cloud SQL, Cloud Storage, dan Container Registry yang ada pada Teknologi Google Cloud Platform (GCP) dan menerapkan fitur *autohealing*, *multiple zones*, *load balancing*, *autoscaling*, *automatic updating*, dan *failover* sebagai metode untuk menjaga tingkat reliabilitas dan availability dari infrastruktur server yang akan dirancang dan dibangun. Infrastruktur server yang dibangun diharapkan mampu menghasilkan suatu sistem dengan tingkat reliabilitas dan availability tinggi hingga menghasilkan sistem yang mampu mendekati zero downtime.

Kata kunci: Infrastruktur, server, cloud computing, layanan, Google Cloud Platform

1. Pendahuluan

Kebutuhan server sudah menjadi kebutuhan utama bagi hampir semua perusahaan maupun pengguna pada umumnya terutama perusahaan sebagai penyedia layanan Teknologi Informasi (TI). Beberapa perusahaan penyedia layanan TI seperti *e-commerce* atau perusahaan pengembang game, server digunakan sebagai penyedia layanan Program dan data. Oleh karena itu dibutuhkan server yang handal, yang tentunya membutuhkan biaya yang tidak sedikit, belum lagi server kadang kala mengalami masalah teknis baik pada software maupun hardware. Kendala tersebut dapat menyebabkan server menjadi down, sehingga tidak dapat diakses dan beroperasi sebagai mana mestinya. Penyebab umum yang mengakibatkan server mengalami down karena server karena perancangan dan pembangunan infrastruktur server yang kurang

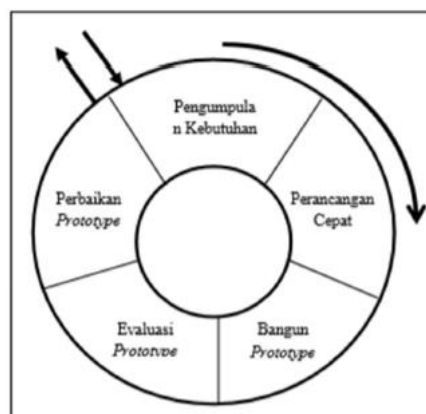
baik, seperti misalnya server akan mengalami overload ketika user yang mengakses server melebihi kapasitas atau jika infrastruktur server tidak di maintenance ada kalanya server akan mengalami crash baik secara hardware maupun software. Oleh karena itu agar kinerja dan ketersediaan server tetap terjaga, maka diperlukan perancangan dan pembangunan infrastruktur server yang baik supaya dapat menghasilkan suatu sistem dengan tingkat reliabilitas dan availability tinggi. Salah satu solusinya dengan menggunakan Teknologi cloud computing.

Teknologi Cloud Computing merupakan paradigma baru dalam penyampaian layanan komputasi. Cloud Computing memiliki banyak keunggulan dibandingkan dengan sistem konvensional [1]. Cloud Computing atau komputasi awan adalah gabungan pemanfaatan teknologi komputer ('komputasi') dan pengembangan berbasis Internet Cloud Storage adalah metafora dari internet, sebagaimana media penyimpanan yang sering digambarkan pada diagram jaringan komputer, Cloud Storage dalam Cloud Computing juga merupakan abstraksi dari infrastruktur kompleks yang disembunyikannya kapabilitas yang terkait teknologi informasi disajikan sebagai suatu layanan (service) sehingga pengguna dapat mengaksesnya lewat internet tanpa mengetahui apa yang ada didalamnya [2]. Cloud Computing merupakan model yang memungkinkan untuk mengakses jaringan dari manapun secara nyaman, cepat, dan sesuai dengan permintaan/kebutuhan kepada suatu kumpulan sumber daya komputasi yang dirilis dengan upaya manajemen interaksi penyedia layanan [3]. Cloud Computing menerapkan satu metode komputasi, yaitu Cloud computing disebut sebagai Teknologi Internet baru yang menyediakan infrastruktur fleksibel, efisien dan bermacam-macam aplikasi untuk bisnis, dimana kapabilitas terkait Teknologi informasi disajikan sebagai suatu layanan (as a service), sehingga pengguna dapat mengaksesnya melalui internet tanpa mengetahui apa yang ada didalamnya beserta kendali terhadap infrastruktur Teknologi yang membantunya [4]. Ada banyak service provider yang menyediakan layanan cloud computing, salah satunya adalah Google dengan teknologinya yang bernama Google Cloud Platform[5].

Google Cloud Platform (GCP) memiliki banyak layanan produk yang fungsinya sebagai cloud computing dan dapat digunakan untuk merancang bangun infrastruktur server yang memiliki tingkat reliabilitas dan availability tinggi, dimana setiap produk yang ada memiliki fungsi dan keunggulan yang berbeda pada setiap fiturnya[6]. Adapun beberapa layanan produk GCP yang dapat digunakan untuk merancang bangun infrastruktur server tersebut adalah Compute Engine, Cloud SQL, Cloud Storage, dan Container Registry [7]. Dimana produk – produk GCP tersebut memiliki beberapa fitur dan metode yang dapat diterapkan dalam membangun infrastruktur high availability server seperti fitur autohealing, multiple zones, load balancing, autoscaling, automatic updating, dan failover [8]. Pembangunan infrastruktur server dengan menggunakan service product dari Teknologi GCP ini diharapkan mampu menghasilkan suatu sistem dengan tingkat reliabilitas dan availability tinggi hingga menghasilkan sistem yang mampu mendekati zero downtime [9].

2. Metode Penelitian

Metode yang digunakan dalam penelitian ini adalah Metode *Prototyping*. Adapun tahapan – tahapan yang ada dalam metode *Prototyping* adalah, sebagai berikut :



Gambar 1. Tahapan Metode Prototype

1. Pengumpulan Kebutuhan
Pada tahap ini dilakukan proses pengumpulan data mengenai infrastruktur server, *cloud computing* dan produk – produk *Google Cloud Platform* yang akan digunakan dalam perancang infrastruktur server yang akan dibangun.
2. Perancangan Cepat
Setelah informasi – informasi telah didapat langkah selanjutnya adalah membangun rancangan sistem dengan membuat desain *prototype* sistem, *prototype* ini akan digunakan sebagai acuan dalam rancangan sistem.
3. Bangun *Prototype*
Prototype dibangun dengan mengacu pada rancangan yang telah dibuat sebelumnya.
4. Evaluasi *Prototype*
Pada tahap ini dilakukan pengujian, apakah *prototype* yang dibangun sudah sesuai dengan yang diharapkan. Apabila *prototype* yang dibangun tidak sesuai maka dilakukan perbaikan *prototype*.
5. Perbaikan *Prototype*
Melakukan perbaikan terhadap *prototype* dari hasil evaluasi.

3. Hasil dan Pembahasan

Proses pembangunan infrastruktur server berbasis cloud computing dengan menggunakan teknologi Cloud Computing dilakukan oleh penulis menggunakan dua tahapan yaitu tahapan analisa dan perancangan serta tahapan implementasi.

3.1 Analisa Kebutuhan Sistem

3.1.1 Kebutuhan Fungsional Service

Pada tabel kebutuhan fungsional service dapat dijelaskan kebutuhan serta persyaratan layanan (service) atau software yang dapat dijalankan dalam infrastruktur server yang akan dibangun nantinya. Tabel kebutuhan fungsional service dapat dilihat pada tabel di bawah ini.

Tabel 1. Kebutuhan Fungsional Service

No.	Kebutuhan Fungsional	Penjelasan
1	Menyediakan layanan (service) berbasis Web Service	Layanan (service) atau software yang dibuat dan dipasangkan pada infrastruktur server adalah software yang berbasis Web Service.
2	Web Service bersifat stateless	Layanan web (Web Service) yang dibuat dan dipasangkan pada infrastruktur server bersifat stateless tidak stateful
3	Web Service dibuild kedalam Docker image, sehingga tidak tergantung bahasa pemrograman	Layanan web (Web Service) dibuild menggunakan Teknologi Docker dan dijadikan Docker image. Ini berfungsi agar Layanan web (Web Service) yang dibuat dapat menggunakan bahasa pemrograman apapun.
4	Web Service bersifat multi-platform menggunakan teknologi Docker container	Layanan web (Web Service) dijalankan menggunakan Teknologi Docker container. Ini berfungsi agar Layanan web (Web Service) yang dibuat dapat dijalankan dalam Web Server yang multi-platform.
5	Web Service menggunakan port yang dapat diakses public	Layanan web (Web Service) berjalan pada satu port tertentu, dimana port tersebut dapat dipakai sebagai public access (untuk user mengakses service tersebut).

6	Web Service menggunakan database SQL	Disediakan database berjenis SQL, jika layanan web (Web Service) memerlukan database untuk penyimpanan data.
7	Web Service memerlukan storage assets	Disediakan Storage Server yang terpisah dari Web Server, jika layanan web (Web Service) memerlukan storage untuk menyimpan assets seperti gambar, video, file, dll.
8	Bisa diintegrasikan dengan 3rd Party Service	Jika Layanan web (Web Service) yang dibuat memerlukan service lain (seperti database NoSQL, Mail Service, dll.) diluar infastruktur yang dibangun, dapat menggunakan 3 rd Party Service.

3.1.2 Kebutuhan Fungsional Infrastruktur

Kebutuhan fungsional infrastruktur berfungsi untuk menunjang service dan server agar menjadi sebuah infrastuktur sistem yang memenuhi nilai availability, scalability dan reliability tinggi[10]. Tabel kebutuhan fungsional infrastruktur dapat dilihat pada tabel di bawah ini.

Tabel 2. Kebutuhan fungsional Infrastruktur

No.	Kebutuhan Fungsional	Penjelasan
1	Web Service dapat berjalan di Multiple Server	Layanan web (Web Service) yang terpasang, dapat berjalan pada banyak server dan dapat diakses seluruhnya oleh user.
2	Web Server memiliki fitur auto-update	Ketika layanan web (Web Service) memerlukan update ke versi terbaru, Web Server dapat melakukan pembaharuan otomatis tanpa mengalami downtime.
3	Web Server dan Database Server memiliki fitur auto recovery	Ketika Web Server dan Database Server mengalami kerusakan baik secara service maupun hardware, dapat melakukan perbaikan dengan sendirinya.
4	Database Server memiliki fitur auto backup	Database Server dapat melakukan otomatis backup data, dan dapat di-restore kapanpun ketika dibutuhkan.
5	Database, Storage, dan Container Registry Server memiliki fitur auto increment capacity	Server dapat meningkatkan kapasitas disk-nya secara otomatis agar data, assets, dan build image dapat disimpan terus menerus tanpa khawatir kehabisan disk capacity.
6	Infrastruktur server tersedia di beberapa zona	Infrastruktur server tersedia di beberapa zona, untuk menjaga ketersediaan ketika terjadi kerusakan di zona tertentu.
7	Infrastruktur server mudah dalam melakukan scaling up dan scaling down	Mudah untuk mengatur kapasitas infrastruktuktur server, ketika mengalami load tinggi maupun rendah karna peningkatan dan penurunan jumlah user.

3.2 Perancangan Kebutuhan Sistem

3.2.1 Pemilihan Layanan dari Google Cloud Platform

Adapun dalam penelitian ini, layanan (*service*) *Google Cloud Platform* yang dipilih untuk memenuhi kebutuhan analisa sistem sebelumnya, dapat dijelaskan pada tabel dibawah.

Tabel 3. Spesifikasi dari Layanan yang dipilih

No	Service yang digunakan	Spesifikasi yang dapat memenuhi kebutuhan sistem
----	------------------------	--

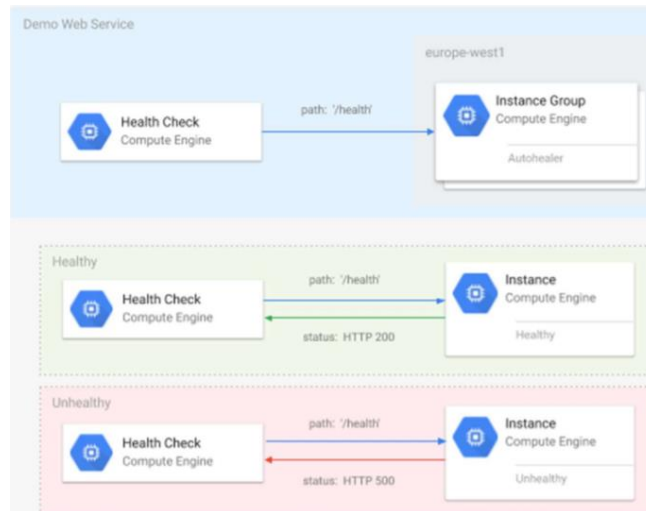
1	Compute Engine	<ul style="list-style-type: none"> a) Untuk memenuhi kebutuhan <i>Web Server</i>. b) Dapat menggunakan fitur <i>deployment</i> menggunakan <i>container image</i>. c) Dapat menggunakan fitur <i>Managed instance groups</i> (MIGs) yang cocok untuk <i>Web Service stateless</i>. d) <i>Managed instance groups</i> (MIGs) dapat menggunakan macam – macam metode untuk menjaga <i>availability</i>, <i>scalability</i> dan <i>reliability</i>. e) Spesifikasi <i>server</i> seperti CPU dan <i>Memory</i> dapat ditentukan sesuai kebutuhan, dan dapat dirubah kembali kapanpun.
2	Cloud SQL	<ul style="list-style-type: none"> a) Untuk memenuhi kebutuhan <i>Database Server</i>. Tersedia berbagai macam <i>Database Engine</i> bertipe <i>SQL</i> (MySQL, PostgreSQL, SQL Server). b) Dapat menggunakan fitur <i>automatic storage increases</i>. c) Dapat menggunakan fitur <i>automatic data backups</i>. d) Dapat menggunakan metode untuk menjaga <i>availability</i> dari <i>server</i>. e) Layaknya <i>Compute Engine</i> spesifikasi <i>server</i> seperti CPU dan <i>Memory</i> dapat ditentukan sesuai kebutuhan, dan dapat dirubah kembali kapanpun
3	Cloud Storage	<ul style="list-style-type: none"> a) Untuk memenuhi kebutuhan <i>Storage Server</i>. b) Terkelola sepenuhnya oleh <i>Google Cloud Platform</i>. c) <i>Availability SLA</i> yang ditawarkan 99.95%. d) Tidak ada batasan dari maksimum penggunaan <i>storage</i>. e) Penyimpanan data dapat diatur di <i>multiple zone</i> ataupun <i>multiple region</i>.
4	Container Registry	<ul style="list-style-type: none"> a) Untuk memenuhi kebutuhan <i>Container Registry Server</i>. b) Terkelola sepenuhnya oleh <i>Google Cloud Platform</i>. c) <i>Availability SLA</i> yang ditawarkan 99.95%. d) Tidak ada batasan dari maksimum penggunaan <i>storage</i>. e) Penyimpanan <i>container image</i> tersebar di <i>multiple Region</i> f) Fitur <i>tagging</i> yang berfungsi untuk mengontrol <i>versioning</i> dari aplikasi <i>Web Service</i> yang disimpan di <i>Container Registry Server</i>.

3.2.2 Perancangan Metode High Availability Server

Adapun pada perancangan infrastruktur server ini, melihat dari service – service yang sudah dipilih pada Tabel 3, maka metode – metode high availability server yang dapat diterapkan adalah sebagai berikut :

- a. Autohealing
Autohealing adalah metode yang bertugas melakukan perbaikan secara otomatis terhadap server atau aplikasi yang berjalan pada server, jika server atau aplikasinya mengalami crash atau stop responding[11]. Autohealing berkerja dengan menggunakan health check, dimana health check

secara berkala melakukan request ke server menggunakan protokol yang ditentukan, seperti HTTP, HTTPS, SSL, atau TCP.



Gambar 2. Health Check Instances Group

b. Load Balancer

Load balancing merupakan proses distribusi beban kerja antar berbagai sumber daya pada sistem apapun, sehingga setiap sumber daya mendapatkan beban tugas yang sama setiap waktu. Tujuannya adalah untuk menyediakan teknik yang mampu menyeimbangkan request dari pengguna sehingga menjadi solusi untuk aplikasi yang lebih cepat [S. Ray And A. De Sarkar,2012].Layaknya autohealing, load balancer juga bekerja menggunakan health check, dimana bila ada salah satu server yang terhubung dengannya dan ditandai sebagai unhealthy server maka load balancer tidak akan mengirimkan traffic dari user ke server tersebut.



Gambar 3. Load Balancer Methods

c. Multiple Zones

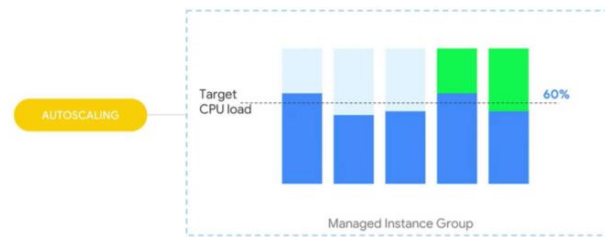
Multiple Zones berfungsi untuk menjaga server dari kesalahan zona yang mungkin terjadi di Google Cloud Platform, dimana jika suatu saat terjadi kesalahan pada salah satu zona, masih dapat tersedia beberapa zona yang lainnya.



Gambar 4. Multiple Zone Methods

d. Autoscaling

Autoscaling bekerja jika server mengalami load yang lebih tinggi atau lebih rendah dari matrik yang telah ditentukan, dimana autoscaling akan otomatis menambahkan atau mengurangi jumlah server untuk dapat menangani masalah tersebut .



Gambar 5. Autoscaling Methods

e. Auto-updating

Auto-updating adalah metode high availability server yang berfungsi untuk melakukan update aplikasi pada server dengan cara yang aman dan tanpa perlu mengalami downtime.

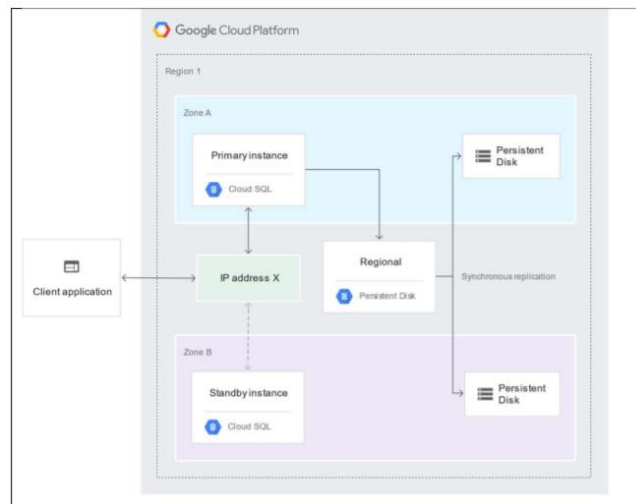
Terdapat beberapa cara yang umum dilakukan dalam melakukan auto-updating, diantaranya adalah canary update dan rolling update. Adapun pada kasus pembuatan infrastruktur server ini cara yang akan dipakai untuk menerapkan Auto-updating adalah dengan menggunakan cara rolling update. Dimana, cara rolling update bekerja dengan cara menambahkan server baru terlebih dahulu, menunggu hingga server baru dalam kondisi healthy, memindahkan traffic dari server lama ke server baru, dan yang terakhir adalah melakukan terminate pada server lamanya.



Gambar 6. Auto-updating Methods

f. Failover

Metode failover ini diterapkan pada Database Server, dimana sesuai dengan yang telah dijelaskan sebelumnya cara kerja dari failover yaitu jika server yang dikonfigurasi high availability menjadi tidak responsif, secara otomatis server akan beralih ke menyajikan data dari server yang standby.



Gambar 7. Failover Methods

3.2.3 Perancangan Topologi Infrastruktur

Perancangan topologi infrastruktur merupakan tahapan selanjutnya yang harus dilakukan setelah ditentukannya layanan (service) Google Cloud Platform yang akan digunakan, serta perancangan dari metode – metode high availability server yang akan diterapkan[12]. Pada perancangan topologi ini, setiap service yang dipilih dan metode yang akan diterapkan disusun sehingga menjadi sebuah desain dari infrastruktur server yang akan dibangun.

3.3 Implementasi Sistem

Implementasi sistem pada penelitian ini bertujuan untuk melakukan pembangunan infrastruktur server sesuai dengan apa yang sudah dirancang sebelumnya, pada tahapan ini penulis melakukan implementasi database server, implementasi storage server, implementasi container registry server dan implementasi web server .

3.3.1 Implementasi Database Server

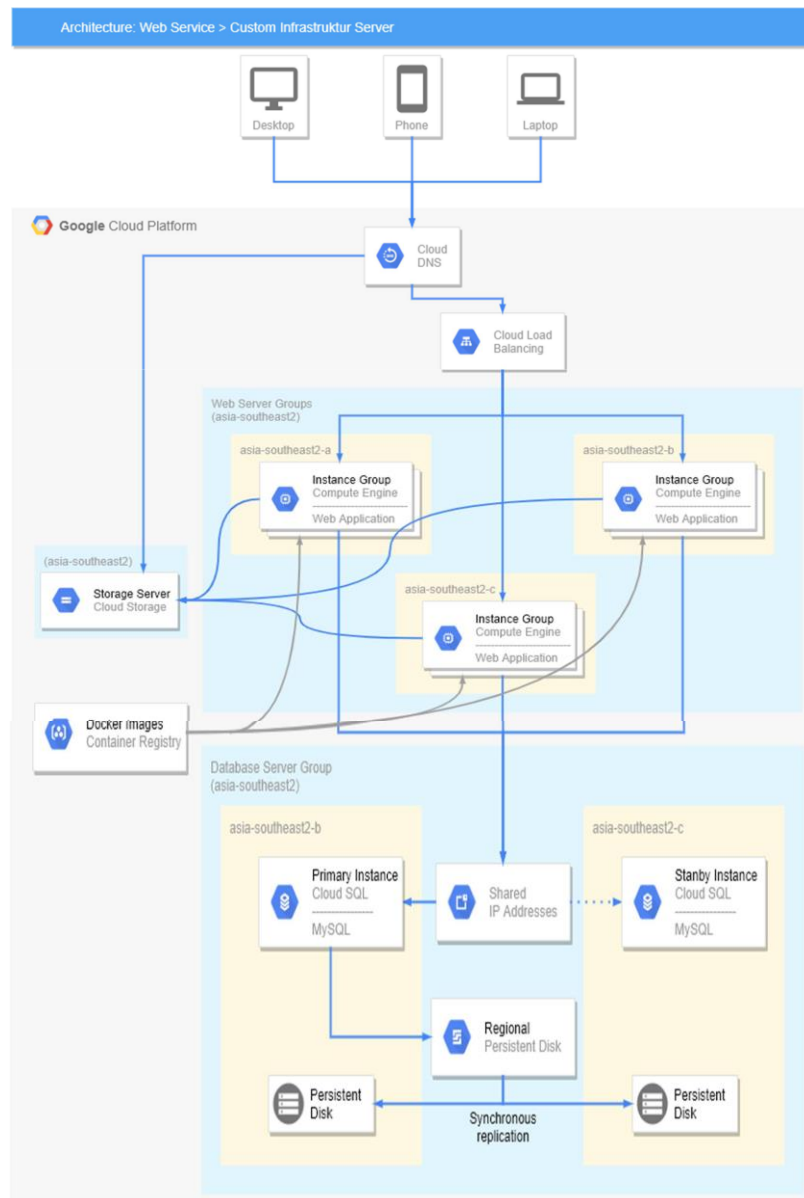
Pada saat melakukan pembangunan Database Server pada layanan (service) Cloud SQL terdapat beberapa hal yang harus dikonfigurasi, diantaranya terdiri dari :

- Memilih engine DBMS
- Mengkonfigurasi instance info
- Database connectivity
- Machine type
- Mengaktifkan metode failover
- Menambahkan MySQL flags
- Tambahkan database user

3.3.2 Implementasi Storage Server

Adapun, pada saat melakukan pembangunan Storage Server pada layanan (service) Cloud Storage terdapat beberapa hal yang harus dikonfigurasi, diantaranya terdiri dari :

- Bucket Name
- Lokasi Storage Server
- Membuat akses untuk user
- Membuat akses untuk Web Server

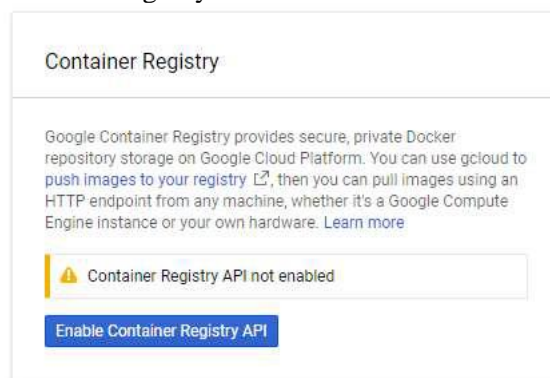


Gambar 8.Topologi Infrastruktur Server

3.3.3 Implementasi Container Registry Server

Untuk melakukan pembangunan Container Registry Server pada layanan (service) Google Continaer Registry terdapat beberapa hal yang harus dikonfigurasi, diantaranya terdiri dari:

- Mengaktifkan layanan Container Registry



Gambar 9. Mengaktifkan Layanan Container Registry

- b) Membuat Gitlab CI CD script untuk build dan push docker Images Adapun hasil Container Registry Server yang menggunakan layanan (service) dari Google Container Registry dan sudah dapat menyimpan hasil build docker image seperti gambar dibawah ini:

19235528-migs-app-sample/release/migs-app-sample
gcr.io/unnur-migsappsample/19235528-migs-app-sample/release/migs-app-sample

Filter by name or tag		Columns	
<input type="checkbox"/> Name	Tags	Created	Uploaded
<input type="checkbox"/> 9fcc474ac78c	1.0, 1.0.82, latest	Oct 2, 2020	Oct 2, 2020
<input type="checkbox"/> 2d21f7e922c4	2.0, 2.0.81	Oct 2, 2020	Oct 2, 2020
<input type="checkbox"/> b16a0b86c22	1.0.80	Sep 12, 2020	Sep 12, 2020
<input type="checkbox"/> b8b8e9f9ef06	1.0.78	Sep 11, 2020	Sep 11, 2020
<input type="checkbox"/> 5968a7a2c21d	1.0.77	Sep 11, 2020	Sep 11, 2020
<input type="checkbox"/> 9efc21b55af0	1.0.73	Sep 11, 2020	Sep 11, 2020

Gambar 10. Hasil Container Registry Server

3.3.4 Implementasi Web Server

Server terakhir yang perlu diimplementasikan agar infrastruktur server yang dirancang dapat dibangun adalah Web Server, ini merupakan server yang cukup rumit dalam melakukan konfigurasinya. Adapun, hal – hal yang perlu dikonfigurasi untuk melakukan pembangunan Web Server pada layanan (service) Google Compute Engine, diantaranya terdiri dari :

- a) Membuat Gitlab CI CD script untuk membuat instance template
b) Mengkonfigurasi informasi grup Web Server

Name ⓘ
Name is permanent

Description (Optional)

Gambar 11. MIGs Information Name

- c) Mengkonfigurasi metode multiple zone

Location
To ensure higher availability, select a multiple zone location for an instance group.
[Learn more](#)

☐ Single zone
☒ Multiple zones
Only managed instance groups can exist in multiple zones.

Region ⓘ
Region is permanent

Zones
☒ asia-southeast2-a
☒ asia-southeast2-b
☒ asia-southeast2-c

Gambar 12. Konfigurasi Multiple Zone

- d) Mengkonfigurasi metode autoscaling

Autoscaling
Use autoscaling to allow automatic resizing of this instance group for periods of high and low load. [Autoscaling groups of instances](#)

Autoscaling mode
Autoscale

Autoscaling metrics
Use metrics to determine when to autoscale the group.
[Autoscaling policy and target utilization](#)

CPU utilization: 60% (default)

+ Add new metric

Cool down period
Specify how long it takes for your app to initialize from boot time until it is ready to serve.
[Cool down period](#)

120 seconds

Minimum number of instances 3 **Maximum number of instances** 8

Gambar 13. Konfigurasi Metode Autoscaling

e) Mengkonfigurasi health check auto healing

Name
Name is permanent
migsappsample-prod-logic-group-health

Description (Optional)

Protocol HTTP **Port** 80

Proxy protocol NONE

Request path /health

Response (Optional) OK

Host HTTP header (Optional)
Example: myapp.example.com (leave blank to use external IP address)

Less

Health criteria
Define how health is determined: how often to check, how long to wait for a response, and how many successful or failed attempts are decisive

Check interval 10 seconds **Timeout** 5 seconds

Healthy threshold 2 consecutive successes **Unhealthy threshold** 3 consecutive failures

Gambar 14. Health Check Autohealing

f) Melakukan konfigurasi load balancer backend

The 'New backend' configuration window includes the following fields:

- Instance group:** migsappsample-prod-logic-group (asia-southeast2, multi-zone)
- Port numbers:** 80
- Balancing mode:** Utilization (selected), Rate
- Maximum backend utilization:** 80%
- Maximum RPS (Optional):** Max total RPS. Leave blank for unlimited. RPS: per instance
- Capacity:** 100%

Gambar 15. Konfigurasi Load Balancer Backend

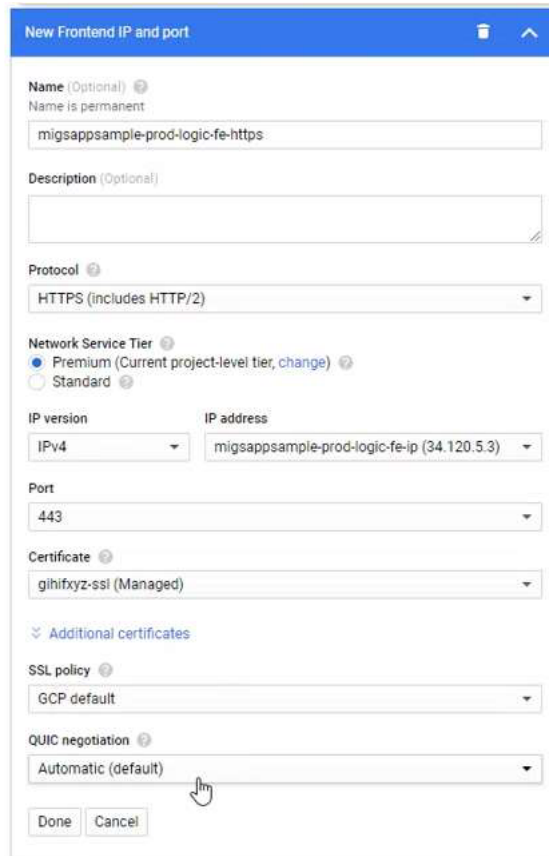
g) Melakukan konfigurasi health check pada load balancer

The 'Health check' configuration window includes the following fields:

- Name:** migsappsample-prod-logic-lb-health
- Description (Optional):**
- Protocol:** HTTP
- Port:** 80
- Proxy protocol:** NONE
- Request path:** /health
- Response (Optional):** OK
- Host HTTP header (Optional):** Example: myapp.example.com (leave blank to use external IP address)
- Less**
- Health criteria:**
 - Define how health is determined: how often to check, how long to wait for a response, and how many successful or failed attempts are decisive
 - Check interval:** 5 seconds
 - Timeout:** 3 seconds
 - Healthy threshold:** 2 consecutive successes
 - Unhealthy threshold:** 3 consecutive failures

Gambar 16. Konfigurasi Health Check Load balancer

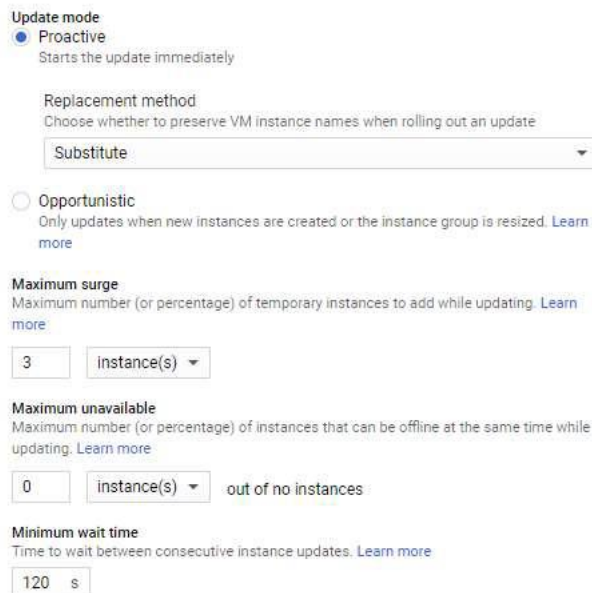
h) Melakukan konfigurasi load balancer frontend



The screenshot shows the 'New Frontend IP and port' configuration window. The 'Name' field is 'migsappsample-prod-logic-fe-https'. The 'Protocol' is 'HTTPS (includes HTTP/2)'. The 'Network Service Tier' is 'Premium'. The 'IP version' is 'IPv4' and the 'IP address' is 'migsappsample-prod-logic-fe-ip (34.120.5.3)'. The 'Port' is '443'. The 'Certificate' is 'gihifxyz-ssl (Managed)'. The 'SSL policy' is 'GCP default'. The 'QUIC negotiation' is 'Automatic (default)'. The 'Done' and 'Cancel' buttons are at the bottom.

Gambar 17. Konfigurasi Load Balancer Frontend

- i) Melakukan konfigurasi rolling update



The screenshot shows the 'Update mode' configuration window. The 'Update mode' is 'Proactive'. The 'Replacement method' is 'Substitute'. The 'Maximum surge' is '3 instance(s)'. The 'Maximum unavailable' is '0 instance(s)'. The 'Minimum wait time' is '120 s'.

Gambar 18. Konfigurasi Rolling Update

4. Kesimpulan

Perancangan infrastruktur server untuk Web Service dengan tingkat reliabilitas dan availability tinggi menggunakan Teknologi Google Cloud Platform berhasil diimplementasikan serta Metode high availability server yang diterapkan berupa autohealing, multiple zones, load balancing, autoscaling,

automatic updating, dan failover telah berhasil diimplementasikan sehingga dapat menjaga realibilitas dan availability dari infrastruktur server yang dibangun.

5. Daftar Pustaka

- [1] Mamuaya G, ... L W-I J of and 2021 U 2021 Cloud Computing-Based Electric Payment System Application Design *Int. J. Inf. Technol. Educ.* **1** 88–93
- [2] ZULKARNAIN D 2020 *RANCANG BANGUN SERVER CLOUD PADA JURUSAN TEKNIK KOMPUTER POLITEKNIK NEGERI SRIWIJAYA* (Politeknik Negeri Palembang)
- [3] Manalu A S, Siregar I M, Panjaitan N J, Sugara H, Komputer T, Indonesia P B, Industri T, Malang U N and Indonesia P B 2021 RANCANG BANGUN INFRASTRUKTUR CLOUD COMPUTING DENGAN *J. TEKINKOM* **4** 303–11
- [4] AA M Y F, Udin M N and ... 2020 The Plastic Hunter Sensoric: APLIKASI PENDETEKSI SAMPAH PLASTIK BERBASIS KECERADASAN BUATAN *Lomba Karya Tulis ...* 135–54
- [5] Sukarno A 2021 *Implementasi Google Cloud Storage Integrasi Dengan Firebase Untuk Aplikasi Facecare (Identifikasi Jerawat) Berbasis Android* (INSTITUT TEKNOLOGI TELKOM PURWOKERTO)
- [6] Taufik and Restyani B 2019 *Teknologi Agrosistem Pada Tanaman Hidroponik Berbasis Microcontroller Unit (Studi Kasus Tabuga Hidroponik Bandung)* (Universitas Nurtanio Bandung)
- [7] Sunaryo S, Tedyana A and Kasmawi K 2017 Rancang Bangun Server Cloud Computing Di Politeknik Negeri Bengkalis *INOVTEK Polbeng - Seri Inform.* **2** 33
- [8] Soumya Ray 2012 Execution Analysis of Load Balancing Algorithms in Cloud Computing Environment *Int. J. Cloud Comput. Serv. Archit.* **2** 1–13
- [9] Apriliana L, Darusalam U D and Nathasia N D 2018 Clustering Server Pada Cloud Computing Berbasis Proxmox VE Menggunakan Metode High Availability *JOINTECS (Journal Inf. Technol. Comput. Sci.* **3**
- [10] Moch. Irfan, S.T. M K 2014 *Sistem Informasi Manajemen*
- [11] Awal A T 2020 Cloud Computing Dan Strategi Ti Modern 1–7
- [12] Challita S, Zalila F, Gourdin C and Merle P 2018 A precise model for Google cloud platform *Proc. - 2018 IEEE Int. Conf. Cloud Eng. IC2E 2018* 177–83