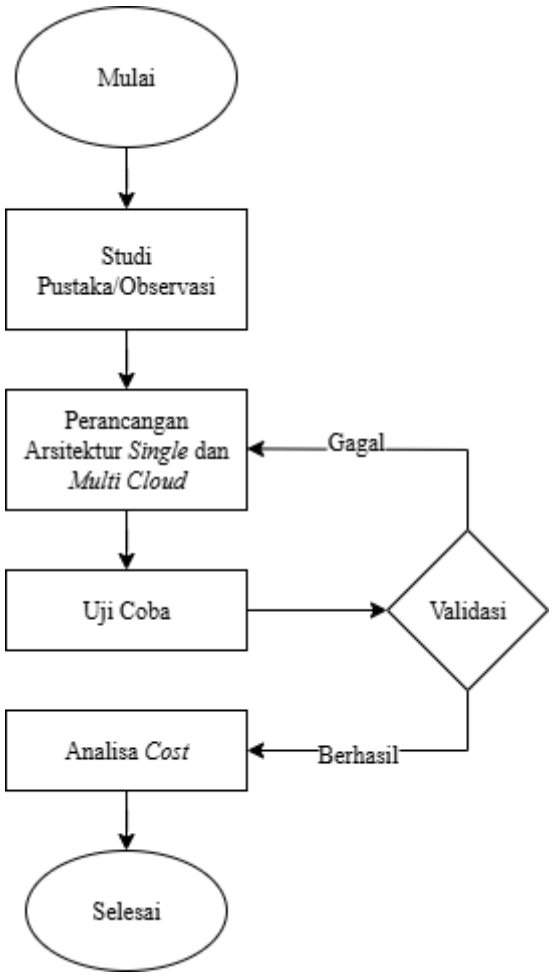
BAB III
METODE PENELITIAN

3.1 Alur Penelitian

Penelitian ini dilakukan melalui beberapa tahapan, dapat dilihat pada gambar 3.1 Alur Penelitian yang dimulai dari analisis dan perancangan infrastruktur *cloud*, pengembangan dan implementasi *web server* menggunakan GCP dan AWS, hingga evaluasi kinerja dan efisiensi biaya pada konfigurasi *single cloud* dan *Multi Cloud*. Hasil pengujian digunakan untuk memberikan rekomendasi implementasi strategi *Multi Cloud* yang optimal dalam pengelolaan biaya aplikasi *website*.



Gambar 3.1 Alur Penelitian

3.2 Karakteristik Objek Penelitian

Objek penelitian ini adalah *website* perusahaan *Food and Beverage* yang membutuhkan infrastruktur *cloud* untuk mendukung kinerja dan efisiensi biaya pengelolaan. *Website* tersebut dipilih karena memiliki karakteristik:

1. Memerlukan aksesibilitas tinggi untuk pengguna.
2. Bergantung pada *database* MySQL untuk pengolahan data transaksi.
3. Memiliki kebutuhan optimasi biaya operasional karena trafik yang bervariasi.
4. Pemilihan *website Food and Beverage* hanya digunakan sebagai studi kasus. Secara umum, hasil penelitian ini dapat diimplementasikan pada berbagai jenis *website* perusahaan lain yang memiliki kebutuhan optimalisasi biaya dan efisiensi pengelolaan infrastruktur *cloud*.

3.2.1 Spesifikasi

Pengujian dilakukan dengan menggunakan konfigurasi spesifikasi yang disesuaikan pada masing masing layanan *cloud*. Pada Google Cloud Platform (GCP), layanan Compute Engine digunakan sebagai *web server* dengan sistem operasi Ubuntu 24.04, 2 vCPU, memori 1 GB, dan penyimpanan 10 GB SSD. Untuk layanan basis data, digunakan Cloud SQL dengan MySQL versi 8.0.41, 1 vCPU, memori sebesar 628.74 MB, dan *storage* 10 GB SSD.

Sementara itu, pada Amazon Web Services (AWS), *web server* dijalankan menggunakan Amazon EC2 dengan spesifikasi 1 vCPU, memori 1 GB, dan sistem operasi Ubuntu 24.04, serta penyimpanan sebesar 10 GB SSD. Layanan basis data menggunakan Amazon RDS dengan MySQL versi 8.0.41, 2 vCPU, memori 1 GB, dan kapasitas penyimpanan sebesar 20 GB.

3.3 Teknik Pengumpulan Data

3.3.1 Studi Literatur

Studi literatur dilakukan untuk memahami konsep dasar *cloud computing* serta strategi *cost management* dalam implementasi sistem berbasis *cloud*. Penelitian ini juga mengkaji berbagai pendekatan *Multi Cloud* dan studi sebelumnya yang membahas perbandingan efisiensi biaya antara penyedia layanan

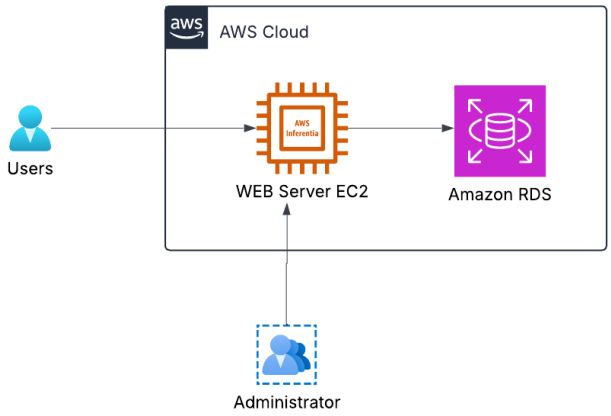
seperti Google Cloud Platform (GCP) dan Amazon Web Services (AWS), sebagai dasar untuk merancang solusi yang efisien dan skalabel.

3.3.2 Perancangan Arsitektur Layanan Cloud

3.3.2.1. Arsitektur Single Cloud

Arsitektur *single cloud* merupakan pendekatan di mana seluruh komponen aplikasi dijalankan dalam satu penyedia layanan *cloud*. Dalam penelitian ini, arsitektur *single cloud* diuji dengan dua *platform* berbeda, yaitu Amazon Web Services (AWS) dan Google Cloud Platform (GCP), untuk membandingkan efisiensi biaya dan kinerja masing masing layanan secara terpisah.

A. Amazon Web Services (AWS)



Gambar 3.2 Arsitektur *single cloud* AWS

Gambar 3.2 menunjukkan arsitektur sistem yang dibangun menggunakan layanan Amazon Web Services (AWS) dalam konfigurasi *single cloud*.

1) *User*

Merupakan entitas klien yang berinteraksi dengan aplikasi melalui *browser*. Setiap permintaan (HTTP/HTTPS *request)* yang dikirim oleh pengguna diarahkan ke *server* aplikasi yang berjalan di AWS.

2) Elastic Compute Cloud (EC2)

Instans virtual yang berfungsi sebagai *web server* utama untuk menjalankan aplikasi. EC2 menerima permintaan dari pengguna, memproses logika aplikasi, berinteraksi dengan *database* (RDS), dan mengembalikan *respons* ke sisi klien.

EC2 dapat dikonfigurasi secara elastis untuk penskalaan vertikal atau horizontal sesuai beban kerja.

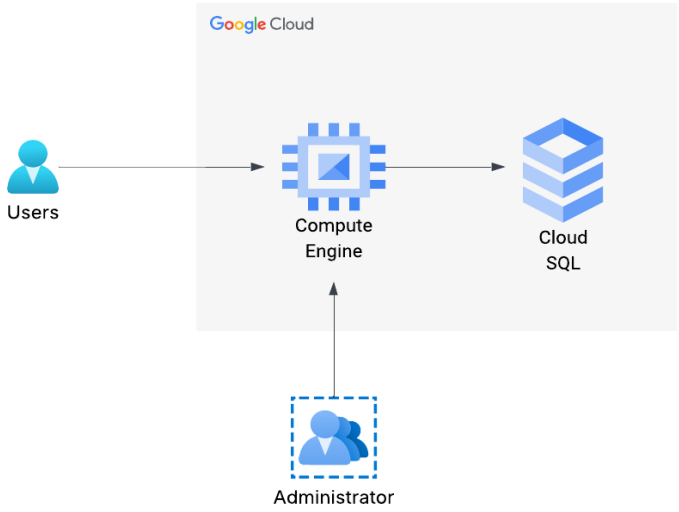
3) Relational Database Service (RDS)

Layanan basis data terkelola berbasis MySQL yang digunakan untuk menyimpan data aplikasi. RDS menangani eksekusi *query*, transaksi, serta penyimpanan data dengan fitur otomatisasi seperti *backup*, *patching*, dan *replikasi* untuk meningkatkan ketersediaan dan keandalan.

4) Administrator

Pihak yang bertanggung jawab atas manajemen infrastruktur AWS, termasuk *deployment instans* EC2, konfigurasi parameter RDS, pengaturan keamanan (IAM, *firewall*/VPC).

B. Google Cloud Platform (GCP)



Gambar 3.3 Arsitektur *single cloud* GCP

Gambar 3.3 menggambarkan arsitektur sistem berbasis Google Cloud Platform (GCP) dengan pendekatan *single cloud*. Seluruh komponen aplikasi dijalankan di lingkungan GCP, mulai dari komputasi hingga manajemen basis data. Arsitektur ini dirancang untuk menangani permintaan pengguna secara efisien dan terintegrasi melalui layanan *cloud native* GCP.

1) *User*

Merupakan klien yang mengakses aplikasi melalui *browser* dan mengirim permintaan (HTTP/HTTPS *request*) ke *server* aplikasi. Interaksi pengguna menjadi titik awal proses komunikasi dalam arsitektur ini.

2) Compute Engine

Layanan infrastruktur sebagai layanan (IaaS) dari GCP yang menyediakan mesin virtual untuk menjalankan aplikasi *web*. *Compute Engine* memproses permintaan pengguna, menjalankan logika aplikasi, dan berkomunikasi dengan *Cloud* SQL untuk mengakses data. *Instans Compute Engine* dapat diskalakan sesuai kebutuhan.

3) *Cloud* SQL

Layanan *database* MySQL terkelola oleh GCP yang berfungsi sebagai penyimpanan data *backend*. *Cloud* SQL menangani *query* secara otomatis dengan fitur seperti *backup, failover*, dan pemeliharaan sistem secara terpusat, serta integrasi penuh dengan Compute Engine.

4) Administrator

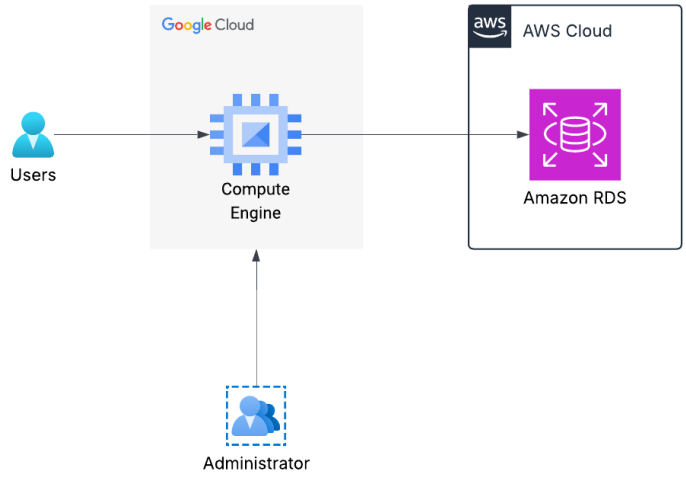
Bertanggung jawab atas pengelolaan sistem secara keseluruhan, termasuk *deployment* aplikasi di Compute Engine, konfigurasi konektivitas dan keamanan antar layanan.

3.3.2.2. Arsitektur Multi Cloud

Arsitektur *Multi Cloud* dalam penelitian ini dirancang dengan menggabungkan layanan dari dua penyedia *cloud*, yaitu Google Cloud Platform (GCP) dan Amazon Web Services (AWS). Pada konfigurasi ini, GCP Compute Engine digunakan sebagai *web server* untuk menjalankan aplikasi, sementara AWS RDS digunakan sebagai layanan *database* MySQL terkelola.

Web server pada Compute Engine memproses permintaan pengguna dan berinteraksi langsung dengan *database* yang di *hosting* di AWS RDS melalui koneksi lintas *cloud.* Pendekatan ini memungkinkan pemanfaatan kekuatan masing masing *platform,* seperti performa komputasi dari GCP dan manajemen *database* canggih dari AWS. Selain itu, konfigurasi ini bertujuan untuk mengoptimalkan

efisiensi biaya dan meningkatkan ketersediaan layanan dengan tidak bergantung pada satu penyedia *single cloud*. Arsitektur sistem ini dapat dilihat pada Gambar 3.4.

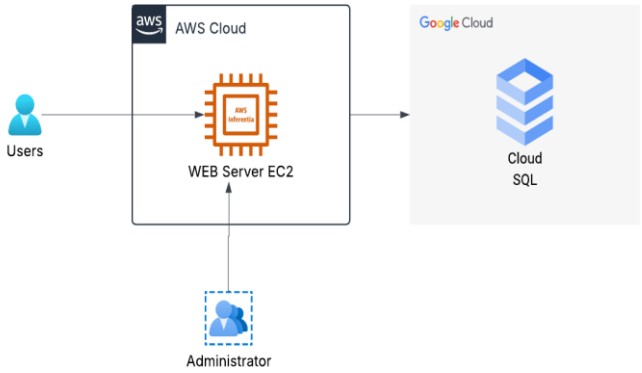


Gambar 3.4 Arsitektur *Multi Cloud* Compute Engine dan AWS RDS

Gambar 3.4 menunjukkan pemrosesan aplikasi yang dijalankan pada Google Compute Engine, sementara penyimpanan data dikelola oleh Amazon RDS sebagai layanan *database* terkelola. Compute Engine berperan sebagai *web server* yang menangani permintaan pengguna, kemudian berkomunikasi dengan Amazon RDS melalui koneksi lintas *cloud* yang telah dikonfigurasi secara aman, mencakup pengaturan *firewall,* autentikasi, serta koneksi jaringan. Dengan arsitektur ini, aplikasi dapat memanfaatkan keunggulan komputasi dari Google Cloud sekaligus mendapatkan keandalan dan skalabilitas pengelolaan basis data dari AWS. Pendekatan *multi cloud* ini juga mendukung strategi optimasi biaya dan fleksibilitas infrastruktur yang lebih baik.

Gambar 3.5 menujukan penyimpanan data, digunakan Google Cloud SQL sebagai layanan *database* MySQL terkelola. *Web server* EC2 berkomunikasi dengan Cloud SQL melalui koneksi lintas *cloud* yang telah dikonfigurasi secara aman, baik dari sisi *firewall*, autentikasi, maupun koneksi jaringan. Arsitektur ini memungkinkan pemanfaatan keunggulan masing masing *platform*, serta

mendukung strategi pengelolaan biaya dan fleksibilitas infrastruktur secara lebih optimal.

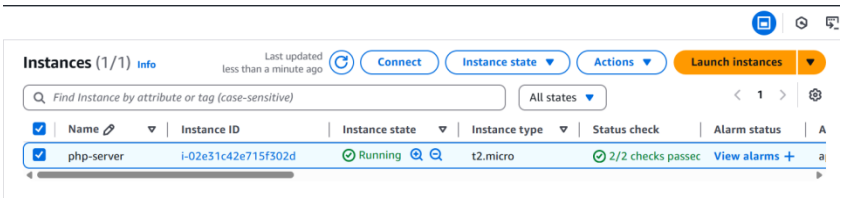


Gambar 3.5 Arsitektur *Multi Cloud* AWS EC2 dan *Cloud* SQL

3.3.3 Uji Coba

Uji coba dilakukan dengan menerapkan sistem pada tiga konfigurasi berbeda, yaitu *single cloud* AWS, *single cloud* GCP, dan *Multi Cloud.* Setiap konfigurasi diuji untuk memastikan keberhasilan *deployment*, konektivitas antara *web server* dan *database,* serta kemampuan aplikasi untuk berjalan secara stabil di masing masing lingkungan *cloud*. Implementasi ini menjadi dasar untuk analisis performa dan efisiensi biaya dari masing masing pendekatan.

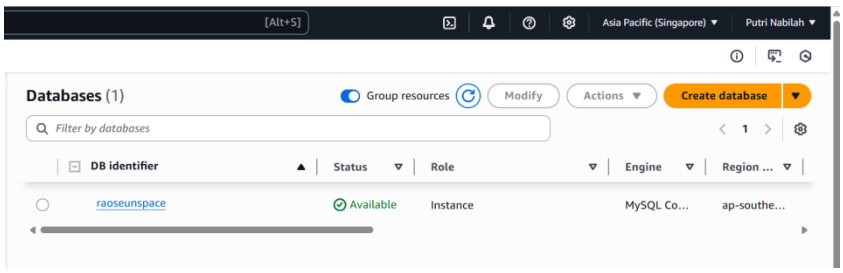
a. *Single cloud* AWS



Gambar 3.6 EC2 berhasil di buat sebagai *web server*

Pada implementasi sistem menggunakan Amazon Web Services (AWS), instans Amazon EC2 berhasil dibuat dan dikonfigurasi sebagai *web server* untuk menjalankan aplikasi berbasis *website*. Proses ini mencakup instalasi sistem operasi, *environment* aplikasi, serta pengaturan akses jaringan. Implementasi ini

ditunjukkan pada Gambar 3.6**,** yang memperlihatkan *instans* EC2 aktif dan siap digunakan.



Gambar 3.7 RDS berhasil di buat sebagai *database* dengan MySQL

Selanjutnya, layanan Amazon RDS berhasil dibuat sebagai *database* menggunakan *engine* MySQL. Konfigurasi meliputi pembuatan *instance database*, pengaturan *endpoint*, serta otorisasi koneksi dari EC2. Koneksi antara EC2 dan RDS telah berhasil dilakukan, memungkinkan aplikasi untuk melakukan *query* ke *database*. Proses ini ditampilkan pada Gambar 3.7.

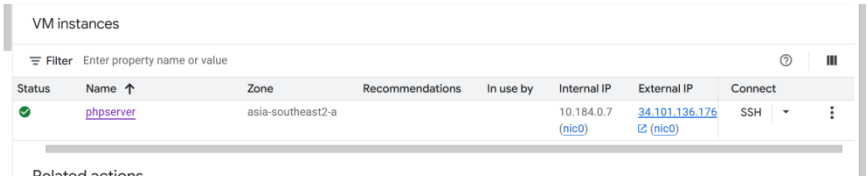
Gambar 3.8 Tampilan *Website* yang Berhasil Di *hosting* di AWS Menggunakan Amazon EC2

Setelah konfigurasi layanan EC2 dan RDS selesai, aplikasi *website* berhasil di *hosting* dan diakses melalui *public* IP *instans* Amazon EC2. Hal ini menandakan bahwa proses *deployment* berjalan dengan baik dan aplikasi dapat merespons permintaan pengguna secara langsung melalui *browser*. *Website* yang sudah aktif menunjukkan bahwa koneksi antara *web server* dan *database* berjalan normal tanpa kendala. Tampilan antarmuka aplikasi yang berhasil diakses ditunjukkan pada Gambar 3.8.

b. *Single cloud* GCP

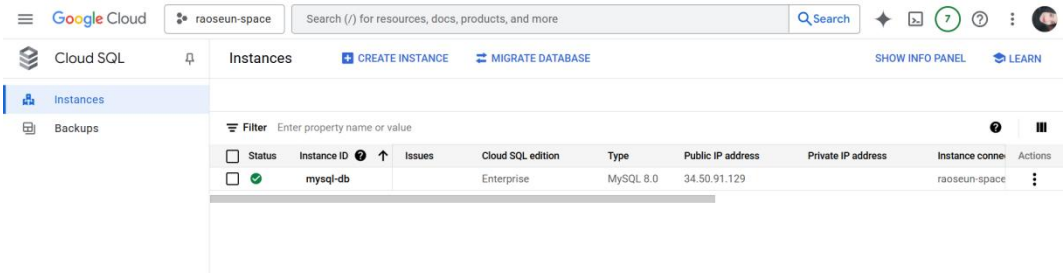
Pada tahap implementasi menggunakan Google Cloud Platform (GCP), *instans* Compute Engine berhasil dibuat dan dikonfigurasi sebagai *web server* yang

menjalankan aplikasi berbasis PHP. Proses instalasi meliputi pengaturan sistem operasi, konfigurasi *web server* (*Apache*/PHP), serta pembukaan akses jaringan untuk memastikan aplikasi dapat diakses secara publik. Proses ini ditampilkan pada Gambar 3.9, yang menunjukkan *instans* Compute Engine aktif dan siap digunakan.



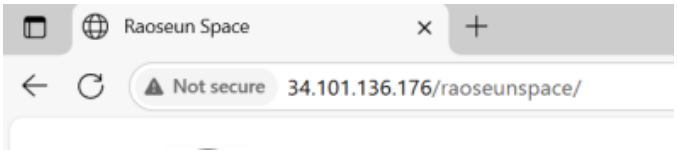
Gambar 3.9 Instans Compute Engine Berhasil Dikonfigurasi sebagai *Web Server*

Selanjutnya, layanan Cloud SQL berhasil dibuat sebagai *database* dengan *engine* MySQL. Konfigurasi meliputi pembuatan *instance database*, pengaturan koneksi eksternal, dan pengelolaan kredensial akses. Implementasi *database* ini ditampilkan pada Gambar 3.10, yang menunjukkan status Cloud SQL dalam kondisi siap digunakan.



Gambar 3.10 Layanan Cloud SQL dengan MySQL Berhasil Dibuat

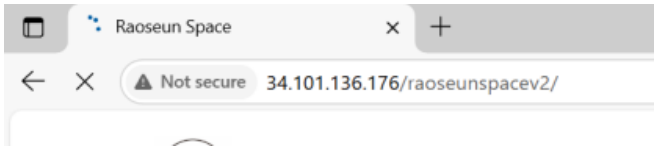
Setelah kedua layanan terhubung, aplikasi berhasil di *hosting* di GCP dan dapat diakses melalui *browser*. Hal ini menandakan bahwa konektivitas antara Compute Engine dan Cloud SQL berjalan dengan baik. Tampilan antarmuka aplikasi yang berhasil diakses secara publik ditunjukkan pada Gambar 3.11.



Gambar 3.11 Tampilan *Website* yang Berhasil Di *hosting* Menggunakan GCP

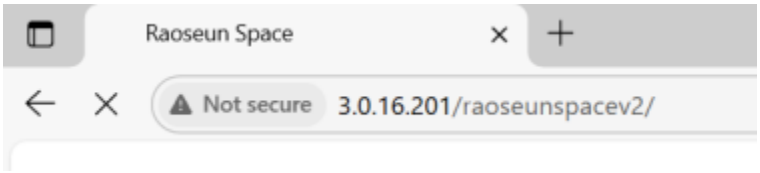
c. Multi Cloud

Implementasi sistem dengan pendekatan *Multi Cloud* dilakukan menggunakan dua konfigurasi lintas *platform*. Konfigurasi pertama menggunakan Google Cloud Platform (GCP) Compute Engine sebagai *web server* dan Amazon RDS sebagai layanan basis data. *Web server* berhasil memproses permintaan aplikasi dan terhubung secara stabil ke *database* RDS, seperti ditunjukkan pada Gambar 3.12**.**



Gambar 3.12 *Multi Cloud*: GCP Compute Engine sebagai *Web Server* dan AWS RDS sebagai *Database*

Konfigurasi kedua dilakukan dengan membalik peran di mana Amazon EC2 digunakan sebagai *web server* dan Cloud SQL dari GCP digunakan sebagai layanan *database* MySQL. Aplikasi berhasil dijalankan dengan baik dan konektivitas lintas *cloud* antara EC2 dan Cloud SQL dapat berfungsi secara lancar sebagaimana ditampilkan pada Gambar 3.13.



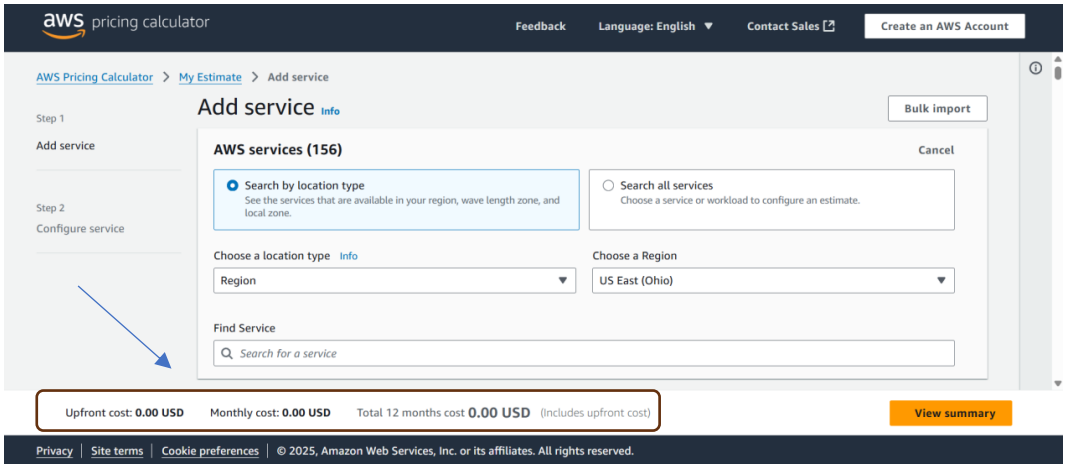
Gambar 3.13 *Multi Cloud:* AWS EC2 sebagai *Web Server* dan Cloud SQL sebagai *Database*

3.4 Teknik Analisis Data

Pada tahap ini dilakukan analisis biaya untuk mengetahui estimasi pengeluaran yang diperlukan dalam penggunaan layanan *cloud* pada masing masing *platform*. Implementasi sistem pada Amazon Web Services (AWS**)** dan Google Cloud Platform (GCP) menggunakan konfigurasi yang masih berada dalam cakupan *Free Tier* dari masing masing penyedia layanan. Artinya, selama masa uji coba awal, layanan seperti Amazon EC2, Amazon RDS, GCP Compute Engine,

dan GCP Cloud SQL tidak dikenakan biaya karena spesifikasi yang digunakan sesuai dengan ketentuan layanan gratis.

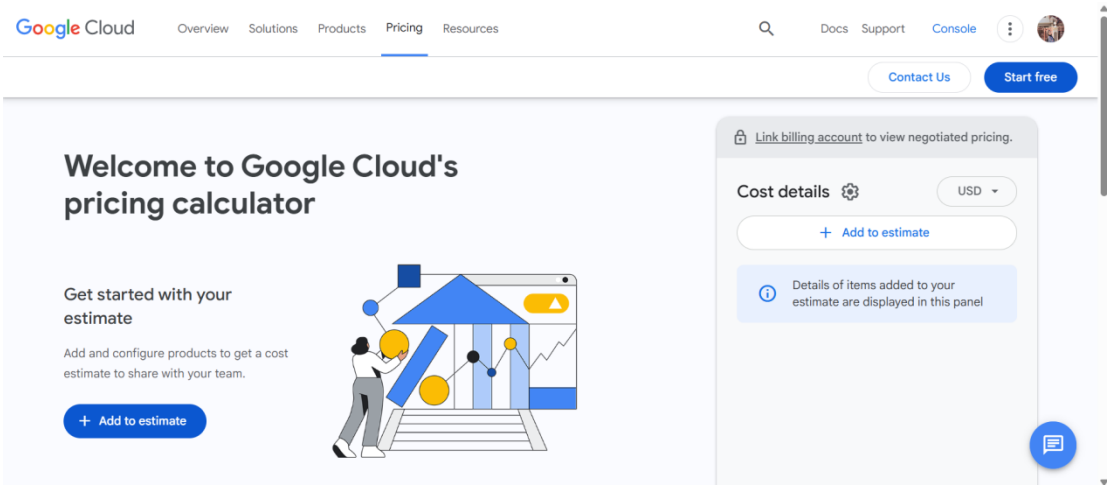
Namun, *Free Tier* hanya berlaku dalam periode dan batas tertentu. Setelah batas pemakaian tersebut terlampaui atau masa layanan gratis berakhir, penggunaan sumber daya akan mulai dikenakan biaya. Oleh karena itu, untuk keperluan evaluasi biaya jangka panjang, dilakukan estimasi menggunakan alat bantu resmi yaitu AWS *Pricing Calculator* dan GCP *Pricing Calculator*. Gambar 3.14 berikut menunjukkan simulasi estimasi biaya menggunakan AWS *Pricing Calculator*.



Gambar 3. 14 Simulasi Estimasi Biaya Menggunakan AWS *Pricing Calculator*

Dari Gambar 3.14 terlihat bahwa estimasi biaya dihitung berdasarkan konfigurasi aktual yang digunakan, meliputi jumlah CPU, kapasitas memori, penyimpanan, serta estimasi durasi pemakaian.

Selanjutnya, Gambar 3.15 memperlihatkan hasil perhitungan biaya menggunakan GCP *Pricing Calculator*.



Gambar 3.15 Simulasi Estimasi Biaya Menggunakan GCP *Pricing Calculator*

Pada gambar ini ditampilkan simulasi biaya layanan GCP dengan konfigurasi yang sama seperti pada AWS, Sehingga dapat dilakukan perbandingan estimasi biaya secara lebih objektif.