*P-ISSN: 2622-6901*
*E-ISSN: 2622-6375*

# PENGUJIAN KINERJA WEB SERVER ELASTIC CLOUD COMPUTE (EC2) FREE TIER PADA AMAZON WEB SERVICE (AWS) MENGGUNAKAN JMETER

**Andi Muhammad Nur Hidayat[1], Adhy Rizaldy[2], Nahrun Hartono[3], Harwalis[4]**
[1]Jurusan Teknik Informatika, Universitas Islam Negeri Alauddin Makassar
[2, 3]Jurusan Sistem Informasi, Universitas Islam Negeri Alauddin Makassar
[4]Jurusan Sistem Informasi, Universitas Tama Jagakarsa
[1, 2, 3]Jl. Sultan Alauddin No.63, Romangpolong, Kec. Somba Opu, Kabupaten Gowa, Sulawesi Selatan - Indonesia
[4]Jl. TB Simatupang No.152 Kec. Jagakarsa, Kota Jakarta Selatan, Daerah Khusus Ibukota Jakarta - Indonesia
**e-mail:** *[1]andi.nurhidayat@uin-alauddin.ac.id, [2]adhy.rizaldy@uin-alauddin.ac.id, [3]nahrunhartono@uin-alauddin.ac.id, [4]harwalis1996@gmail.com

***Abstract***
*The research aims to evaluate the capacity of Amazon Web Service's (AWS) Elastic Cloud Compute (EC2) Free Tier Web Server using the JMeter testing tool. The objective is to ascertain the extent of the server's load-bearing capabilities. Tests were conducted, loading 500 users within a 10-second range and 1000 users within a 1-second range. Despite being a free service, test results showed the server's capability to handle high loads without errors (0% error rate). These findings illustrate the free service's ability to maintain stable performance under significant user loads. The utilization of EC2 Free Tier on AWS demonstrates its capacity to handle heavy tasks without compromising service quality. From about 5 steps in each of two tests, the value for sample time / response time in the initial test of the Amazon Web Service Web server is 59 / ms (millisecond) and the average latency value in the second test, namely 7.213 /ms (millisecond). Both of these gained CPU Utilization from minum, average and maximum with final rate are zero Error. This outcome concludes that the free server possesses robust and reliable capabilities, instilling confidence in an effective and efficient server solution even under high usage levels without additional costs.*

***Keyword:*** *EC2, Komputasi Awan, JMeter, AWS.*

## PENDAHULUAN

Pertumbuhan internet yang cepat saat ini telah menyebabkan peningkatan akses pengguna yang terhubung, yang memengaruhi permintaan akan layanan penyedia seperti *server web*. Teknologi *cloud computing* yang ada sekarang telah memberikan bantuan besar kepada pengelola *web server* dalam manajemen *server*, terutama dalam menghadapi peningkatan akses pengguna. Upaya untuk meningkatkan kinerja *web server* yang mengalami lonjakan akses pengguna dilakukan dengan memungkinkan beberapa *server* untuk menangani beban koneksi yang masuk dan memberikan layanan konten dari *web server* yang sama. Dengan cara ini, kinerja *web server* dapat dipertahankan atau bahkan ditingkatkan.

Teknologi *cloud computing* telah menjadi fondasi yang mengubah lanskap bisnis dan teknologi informasi secara fundamental dalam beberapa dekade terakhir. Sebelumnya, perusahaan harus menginvestasikan banyak waktu dan sumber daya dalam membangun infrastruktur komputasi mereka sendiri. Namun, dengan munculnya layanan *cloud* seperti *AWS*. Paradigma ini berubah. Sekarang, perusahaan dapat menyewa sumber daya secara virtual sesuai kebutuhan, tanpa perlu khawatir tentang perawatan perangkat keras (M. Fransisca, 2019) .

Sebuah *Web Server* yang handal dalam aspek keandalan, kecepatan, dan kinerja merupakan kebutuhan utama untuk menangani semua permintaan pengguna atau pengunjung. Ada tiga opsi server yang umumnya digunakan sebagai *Web Server*, yaitu *Shared Hosting, VPS (Virtual Private Server)*, dan *Cloud Hosting*. *Shared hosting* memungkinkan pengguna untuk

berbagi sumber daya dan kapasitas *server*, menawarkan layanan yang ekonomis namun sering digunakan secara bersama-sama. *VPS* merupakan jenis hosting di mana sumber dayanya berasal dari satu *server* fisik yang terbagi menjadi beberapa *server* virtual yang beroperasi seperti *dedicated server*. Sementara *Cloud hosting* menggunakan sumber daya dari beberapa *server* fisik terpusat yang bekerja secara bersamaan, sering kali dalam bentuk *cluster* (Fandy, 2022) ).

*Amazon Web Services (AWS)* adalah *platform* layanan *cloud computing* yang sangat populer dan luas yang dimiliki oleh Perusahaan raksasa Amazon. Platform ini menyediakan beragam layanan yang mencakup komputasi, penyimpanan data, basis data, analitik, kecerdasan buatan, *Internet of Things (IoT)*, keamanan, dan masih banyak lagi. *Elastic Compute Cloud (EC2)* merupakan salah satu layanan utama yang ditawarkan oleh *Amazon Web Services (AWS),* membawa perubahan besar dalam paradigma infrastruktur teknologi informasi. Sebelum adanya layanan *cloud* seperti *EC2*, organisasi harus mengelola *server* fisik sendiri, yang memerlukan investasi besar dalam infrastruktur, waktu, serta sumber daya untuk pemeliharaan dan pengelolaan. Namun, *EC2* memperkenalkan konsep infrastruktur sebagai layanan *(Infrastructure as a Service/IaaS)*, memungkinkan pengguna untuk menyewa *server* virtual sesuai kebutuhan mereka (A. Amrulla, 2023).

Layanan *EC2* memiliki berbagai jenis instance yang dapat dipilih, yang masing-masing memiliki kapasitas komputasi yang berbeda-beda, mulai dari yang sederhana hingga tingkat performa tinggi. *AWS* memberikan *Free Tier* atau Akses gratis selama 12 bulan untuk menggunakan *EC2* untuk pengguna baru. *Free Tier EC2* menjadi pintu masuk bagi pengguna untuk mulai mengenal layanan *cloud* tanpa biaya awal yang signifikan. *Tier* ini memungkinkan pengguna untuk mengakses sumber daya tertentu secara gratis dalam batasan tertentu, memberi kesempatan bagi pengguna baru untuk menguji layanan, mengembangkan aplikasi, serta memahami kemampuan dan keterbatasan layanan *EC2* tanpa biaya (Amazon, Inc, 2023).

*JMeter* adalah sebuah perangkat lunak yang digunakan untuk melakukan pengujian kinerja atau pengujian beban pada aplikasi web. Ini memungkinkan pengguna untuk menguji seberapa baik sebuah aplikasi atau situs web dapat menangani jumlah pengguna yang besar, seberapa cepat halaman web dapat dimuat, dan bagaimana sistem tersebut bertahan terhadap beban yang tinggi. *JMeter* memungkinkan simulasi berbagai skenario penggunaan untuk mengevaluasi kinerja suatu sistem secara luas (Fadila, A, 2023).

## METODE PENELITIAN

### 1. Tahap Pelaksanaan Penelitian

Proses penelitian dimulai dengan mengidentifikasi permasalahan yang akan menjadi fokus penelitian. Langkah selanjutnya adalah merumuskan permasalahan agar tetap terfokus dan tidak melewati batasan penelitian yang telah ditetapkan. Untuk mendukung dan memperkuat jalannya penelitian, dilakukan tinjauan pustaka yang mencakup sumber-sumber seperti jurnal, artikel, buku, dan situs web yang relevan dengan topik penelitian.

### 2. Pengumpulan Data

Data dikumpulkan guna mendukung penelitian mengenai Pengujian Kinerja *Web Server* pada Layanan *Elastic Cloud Compute (EC2)* di *Amazon Web Services (AWS)*. Studi literatur dilakukan untuk mendapatkan informasi mengenai standar performa yang optimal demi menunjang kesimpulan mengenai efisiensi *Web Server*. Peneliti mengumpulkan serta menganalisis berbagai sumber literatur termasuk buku referensi, jurnal, dan informasi dari internet yang relevan dengan penelitian, khususnya terkait dengan performa yang optimal dari *web server*.

### 3. Perancangan Sistem

Proses pengumpulan data yang diperoleh peneliti akan dikumpulkan dan dianalisis sesuai dengan judul yang diambil oleh peneliti yakni Pengujian Kinerja Web Server *Elastic Cloud Compute (EC2) Free Tier* Pada *Amazon Web Service(AWS).* Pengambilan data akan dilakukan dengan menggunakan aplikasi *Apache JMeter* pada web server yang dihosting di

*P-ISSN: 2622-6901*
*E-ISSN: 2622-6375*

*EC2*. Beberapa tahapan dari proses pengujian akan dijabarkan sebagaimana berikut:

a) Tahapan pertama, membuat akun *AWS* terlebih dahulu di halaman *https://portal.aws.amazon.com/billing/signup#/start/email*. Setelah akun berhasil dibuat, pengguna memilih layanan *EC2* yang ada di halaman utama kemudian terdapat beberapa Langkah -langkah selanjutnya dalam pembuatan *EC2* yaitu:
   1) Pengguna mengatur *Security Groups* terlebih dahulu dengan cara mengisi *security group name* lalu *description* (Wu, 2013). Selanjutnya tambahkan 2 rules untuk lalu lintas masuk(*inbound*) melalui port 5000 dan *port SSH* dengan ketentuan sebagai berikut.
      Ketentuan 1 :
      - Type: Custom TCP
      - Port Range: 5000
      - Source Type: Anywhere-IPv4
      - Description: Application Port

      Ketentuan 2 :
      - Type: SSH
      - Port Range: 22
      - Source Type: Anywhere-IPv4
      - Description: SSH Port
   2) Memilih sistem operasi yang akan digunakan, disini pengguna akan menggunakan sistem operasi Ubuntu Server 22.04 LTS yang berlokasi di Asia Tenggara (singapore).
   3) Memilih type EC2 yang akan digunakan, disini pengguna menggunakan *free tier eligible* (t-family*: t2.micro)* dengan *vCPU* 1x dan memori 1 GB.

b) Tahap kedua, instalasi OpenSSH pada device pengguna untuk menghubungkan server EC2 AWS yang sudah di buat pada device melalui terminal.

c) Tahap ketiga, membangun web server. Peneliti membangun web server untuk aplikasi peminjaman buku menggunakan Node JS v18.13.0 seperti pada penelitian sebelumnya (Sutanto et al., 2021), untuk digunakan sebagai bahan uji coba kinerja *EC2 free tier* bisa dilihat pada *repository* peneliti di link *github* berikut *https://github.com/ekkymannyu/back-end-repo.git*.

## 4. Perancangan Proses Pengujian

Dalam upaya Pengujian Kinerja *Web Server* pada Layanan *Elastic Cloud Compute (EC2)* di *Amazon Web Services (AWS)* beberapa proses yang akan dilakukan adalah:

a) Melakukan implementasi *Web Server* pada *Elastic Cloud Compute (EC2)*
Peneliti akan menerapkan *Web Server Virtual Private Server (VPS)* di *Amazon Web Services (AWS*) yang sering disebut sebagai *Elastic Compute Cloud (EC2)*. Berikut adalah tampilan dari dasboard *Elastic Compute Cloud (EC2)* yang akan digunakan (Kanikathottu, 2020).

b) Melakukan pengujian performa kinerja *Web Server*
Pengujian kinerja *Web Server Virtual Private Server (VPS)* di *Amazon Web Services (AWS)* atau yang dikenal sebagai *Elastic Compute Cloud (EC2)* akan dilaksanakan menggunakan aplikasi perangkat lunak Apache JMeter. Tahapan untuk pengujiannya:
   a. Membuka aplikasi *JMeter*
   b. Beri nama pada tahapan pengujian – *add - Threads (Users) - Thread Group* ;isi data;
   c. Buat sample *http request* dengan cara *add* pada (*Thread Group*) – *Sampler – HTTP Request*; isi data;
   d. Perancangan *Listener* untuk pelaporan hasil dengan *add* pada *(HTTP Request) – Listener* - {pilih jenis laporan; S*ummary Report, Simple Data Writter, View Result in Table*}
   e. Jalankan *Performance Test*

*P-ISSN: 2622-6901*
*E-ISSN: 2622-6375*

**5. Perancangan Pengujian**

Pengujian dilakukan dengan menggunakan bantuan aplikasi *JMeter*. Pengujian ini bertujuan untuk melihat beberapa kondisi diantaranya:

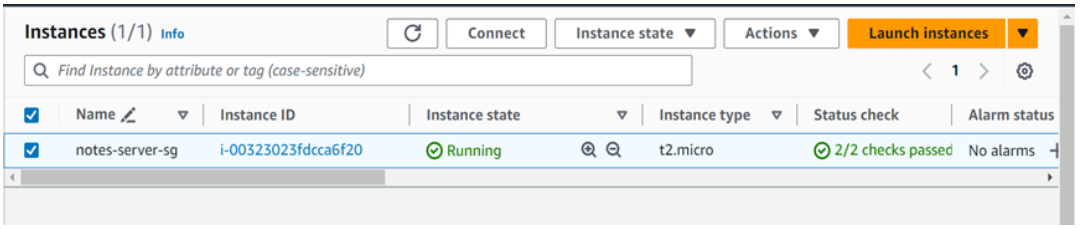
1. *Throughput*, yaitu jumlah permintaan user per menit yang diproses oleh *server* (Prasetyo, 2021).
2. *Sample Time/Respons Time* , yaitu perbedaan antara ketika permintaan dikirim dan waktu respon telah diterima (Pradana, 2022).
3. *Latency* , ini mengacu pada durasi yang dibutuhkan bagi permintaan untuk mencapai *server* dan Kembali ke klien, atau perbedaan antara waktu pengiriman permintaan dan saat respon mulai diterima (Berenberg, A, 2023)

Hasil kinerja tersebut akan dibandingkan dengan data yang telah dikumpulkan dari literatur mengenai standar performa terbaik untuk *Web Server.* Dari situ, peneliti dapat menarik kesimpulan untuk menentukan *Web Server* yang memiliki kinerja terbaik, andal, dan sesuai dengan kebutuhan penggunaan yang diinginkan.

## HASIL DAN PEMBAHASAN

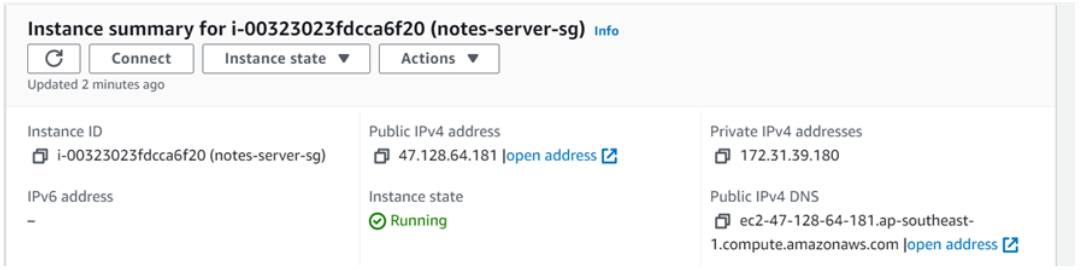
### A. Hasil Pengujian

Pengujian kali ini akan dilakukan sebanyak 2 kali untuk digunakan sebagai bahan perbandingan di sub bab pembahasan. Sebelum dilakukan pengujian peneliti membangun *web server* di *EC2 Amazon Web Service*, lalu kemudian menjalankannya agar bisa di akses oleh aplikasi *JMeter.*




Gambar 1. *Instance EC2*

Membuat dan menjalankan instaces agar web server-nya menyala dan dapat di akses oleh aplikasi *JMeter* dan device peneliti melalui *OpenSSH*. Gambar di bawah menunjukan *IPv4 EC2* yang selanjutnya akan dihubungkan dengan aplikasi *JMeter*.




Gambar 2. *View details EC2*

Menghungkan *Web Server* dengan *OpenSSH client* peneliti, yang kemudian menjalankan server-nya.

*P-ISSN: 2622-6901*
*E-ISSN: 2622-6375*

```
ubuntu@ip-172-31-39-180:~$ cd back-end-repo
ubuntu@ip-172-31-39-180:~/back-end-repo$ npm run start-prod

> submission@1.0.0 start-prod
> NODE_ENV=production node ./src/server.js

server berjalan pada http://0.0.0.0:5000
```

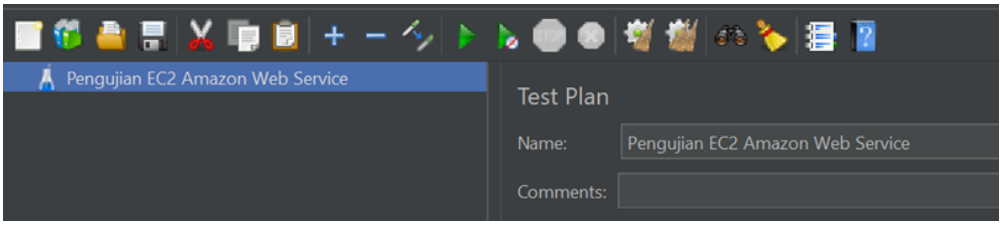
Gambar 3. *Ubuntu server EC2*

Setelah Server dijalankan, maka di bawah ini dilakukan *Web Server Amazon Web Service*.

1. **Pengujian 1 *Web Server Amazon Web Service***
   Pengujian pertama *Load Testing* pada *Web Server Amazon Web Service* dengan *Apache JMeter* dengan format tes sebagai berikut:
   a. Jenis pengujian *HTTP Request*
   b. *Thread / User* 50
   c. *Ramp-Up (in second)* 10.
   d. *Loop Count* 10.
   e. Total *HTTP Request* 500.

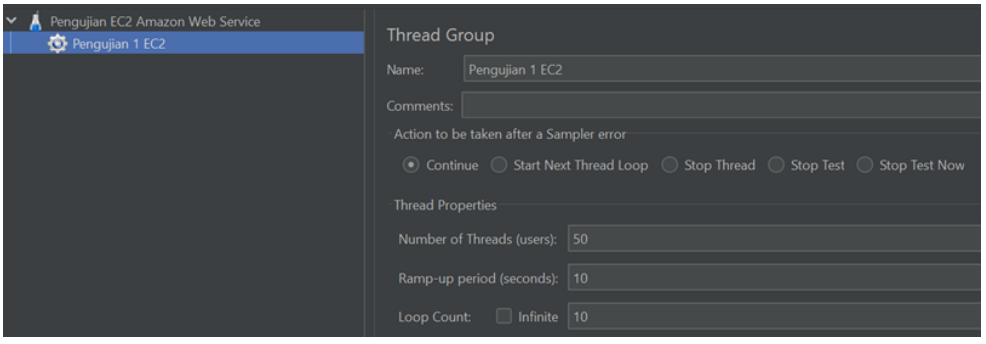
   Berikut langkah Test Plan dari Pengujian pertama Web Server Amazon Web Service:

   Pertama, Pembuatan Test Plan dengan name “Pengujian EC2 Amazon Web Service”




Gambar 4. Test Plan JMeter

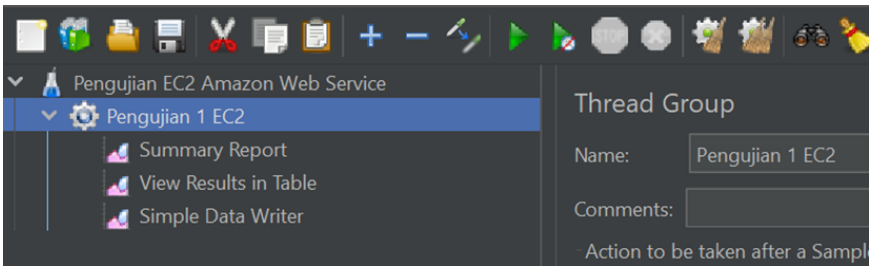
Kedua, membuat Thread Group dengan nama “Pengujian 1 EC2” kemudian mengisi Thread Properties sesuai requirement pengujian.




Gambar 5. Thread Group Pengujian 1 EC2

Ketiga, menambahkan listener untuk melihat hasil disetiap pengujian yang di lakukan pada Theread Group. Listerner yang ditambahkan adalah Summary Report, View Result in Table dan Simple Data Writer.

*P-ISSN: 2622-6901*
*E-ISSN: 2622-6375*




Gambar 6. Listener Pada Thread Group

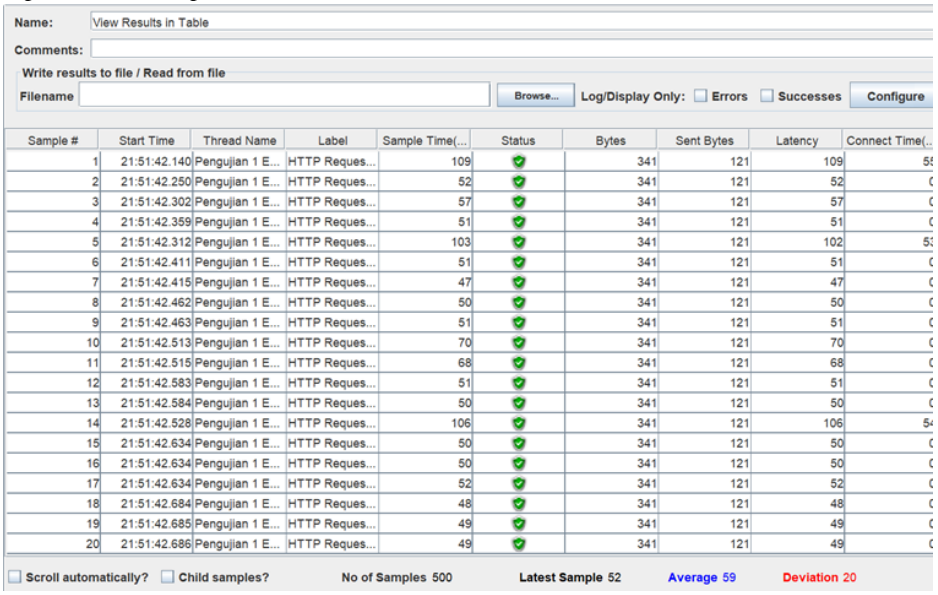
Keempat, menambahkan sampler HTTP Request dengan nama "HTTP Request Pengujian 1"

Kelima, Jalankan test.

Hasil pengujian pertama dari *EC2 Amazon Web Service* adalah sebagi berikut:

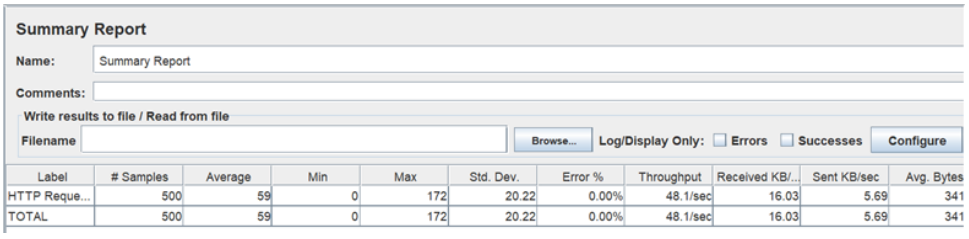
a. *Throughput,* hasil *Througput* dari *sampler HTTP Request* dalam pengujian pertama kali ini adalah 48.1/sec dengan *error samples* 0.0%

*b. Sample Time / Response Time*




| Sample # | Start Time | Thread Name | Label | Sample Time(... | Status | Bytes | Sent Bytes | Latency | Connect Time(... |
|---|---|---|---|---|---|---|---|---|---|
| 1 | 21:51:42.140 | Pengujian 1 E... | HTTP Reques... | 109 | | 341 | 121 | 109 | 55 |
| 2 | 21:51:42.250 | Pengujian 1 E... | HTTP Reques... | 52 | | 341 | 121 | 52 | 0 |
| 3 | 21:51:42.302 | Pengujian 1 E... | HTTP Reques... | 57 | | 341 | 121 | 57 | 0 |
| 4 | 21:51:42.359 | Pengujian 1 E... | HTTP Reques... | 51 | | 341 | 121 | 51 | 0 |
| 5 | 21:51:42.312 | Pengujian 1 E... | HTTP Reques... | 103 | | 341 | 121 | 102 | 53 |
| 6 | 21:51:42.411 | Pengujian 1 E... | HTTP Reques... | 51 | | 341 | 121 | 51 | 0 |
| 7 | 21:51:42.415 | Pengujian 1 E... | HTTP Reques... | 47 | | 341 | 121 | 47 | 0 |
| 8 | 21:51:42.462 | Pengujian 1 E... | HTTP Reques... | 50 | | 341 | 121 | 50 | 0 |
| 9 | 21:51:42.463 | Pengujian 1 E... | HTTP Reques... | 51 | | 341 | 121 | 51 | 0 |
| 10 | 21:51:42.513 | Pengujian 1 E... | HTTP Reques... | 70 | | 341 | 121 | 70 | 0 |
| 11 | 21:51:42.515 | Pengujian 1 E... | HTTP Reques... | 68 | | 341 | 121 | 68 | 0 |
| 12 | 21:51:42.583 | Pengujian 1 E... | HTTP Reques... | 51 | | 341 | 121 | 51 | 0 |
| 13 | 21:51:42.584 | Pengujian 1 E... | HTTP Reques... | 50 | | 341 | 121 | 50 | 0 |
| 14 | 21:51:42.528 | Pengujian 1 E... | HTTP Reques... | 106 | | 341 | 121 | 106 | 54 |
| 15 | 21:51:42.634 | Pengujian 1 E... | HTTP Reques... | 50 | | 341 | 121 | 50 | 0 |
| 16 | 21:51:42.634 | Pengujian 1 E... | HTTP Reques... | 50 | | 341 | 121 | 50 | 0 |
| 17 | 21:51:42.634 | Pengujian 1 E... | HTTP Reques... | 52 | | 341 | 121 | 52 | 0 |
| 18 | 21:51:42.684 | Pengujian 1 E... | HTTP Reques... | 48 | | 341 | 121 | 48 | 0 |
| 19 | 21:51:42.685 | Pengujian 1 E... | HTTP Reques... | 49 | | 341 | 121 | 49 | 0 |
| 20 | 21:51:42.686 | Pengujian 1 E... | HTTP Reques... | 49 | | 341 | 121 | 49 | 0 |

Gambar 7. *Sample Time / Response Time* Pengujian 1 *EC2*




| Label | # Samples | Average | Min | Max | Std. Dev. | Error % | Throughput | Received KB/... | Sent KB/sec | Avg. Bytes |
|---|---|---|---|---|---|---|---|---|---|---|
| HTTP Reque... | 500 | 59 | 0 | 172 | 20.22 | 0.00% | 48.1/sec | 16.03 | 5.69 | 341 |
| TOTAL | 500 | 59 | 0 | 172 | 20.22 | 0.00% | 48.1/sec | 16.03 | 5.69 | 341 |

Gambar 8. Summary Report Pengujian 1 EC2

Di dalam Gambar 7, terdapat hasil dari *sample time / response time*, di mana nilai yang dicatat untuk waktu *sample time / response time* adalah *average*. *Average* ini dihitung dari total nilai *sample time / response time*. Sehingga, nilai untuk *sample time / response time* pada pengujian awal *Web server Amazon Web Service* adalah 59/ms(*milisecond*)

*P-ISSN: 2622-6901*
*E-ISSN: 2622-6375*

c. *Latency*
Untuk menemukan nilai *latency* dari keseluruhan sample harus dihitung secara manual dengan menghitung Rata – rata, dengan rumus :

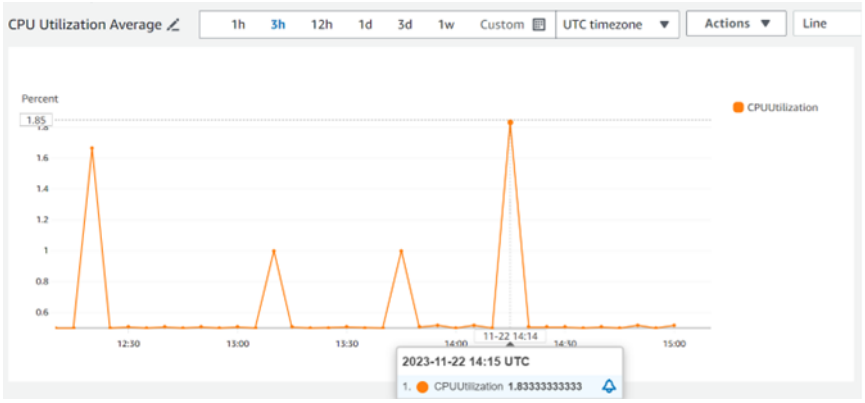
$$Latency = \frac{JumlahSeluruhNilaiLatency}{JumlahSample}$$

Maka nilai rata – rata latency pada pengujian, yaitu:
Pertama adalah $\frac{38.026}{500}$=7.052/ms(*milisecond*)

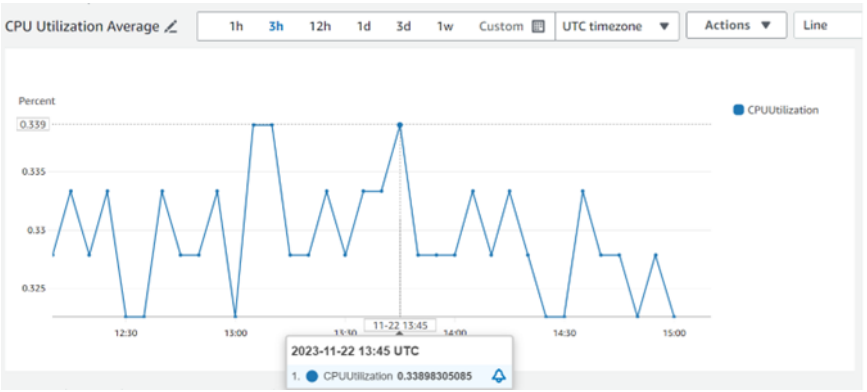
d. *Resource Utilization*
Pengujian dilakukan dari sample 1 mulai berjalan pada pukul 21:51:42, sampai sample nomor 500 berakhir pada pukul 21:51:52.

1. CPU *Utilization Minimum,* menunjukkan nilai *Minimum* (terendah) penggunaan *CPU Utilization* di angka 0,338%

2. CPU *Utilization Maximum*




Gambar 9. Sample Time / Response Time Pengujian 1 EC2

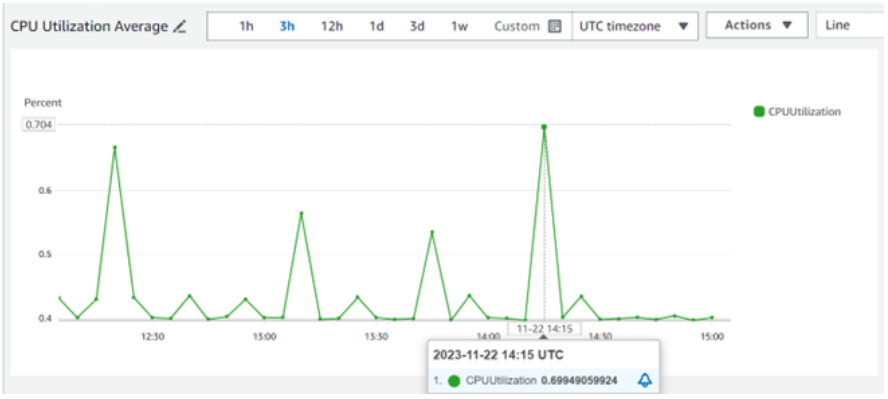



Gambar 10. CPU *Utilization Maximum* Pengujian 1

Menunjukkan nilai *Maximum* (tertinggi) penggunaan *CPU Utilization* di angka 1.83%.

*P-ISSN: 2622-6901*
*E-ISSN: 2622-6375*

3. CPU *Utilization Average*




Gambar 11. *CPU Utilization Average* Pengujian 1

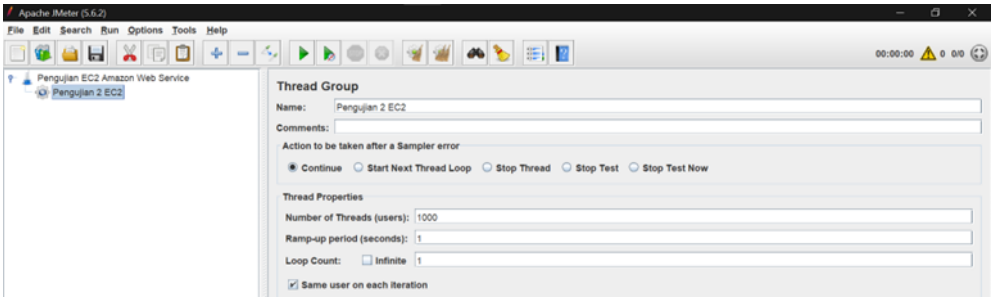
Menunjukan nilai Average (Rata – rata) penggunaan *CPU Utilization* di angka 0.699%

**2. Pengujian 2 *Web server Amazon Web Service***

Pengujian pertama *Load Testing pada Web Server Amazon Web Service* dengan *Apache JMeter* dengan format tes sebagai berikut:

a. Jenis Pengujian *HTTP Request*.
b. *Thread / User* 1000.
c. *Ramp-Up (in Second)* 1.
d. *Loop Count* 1.

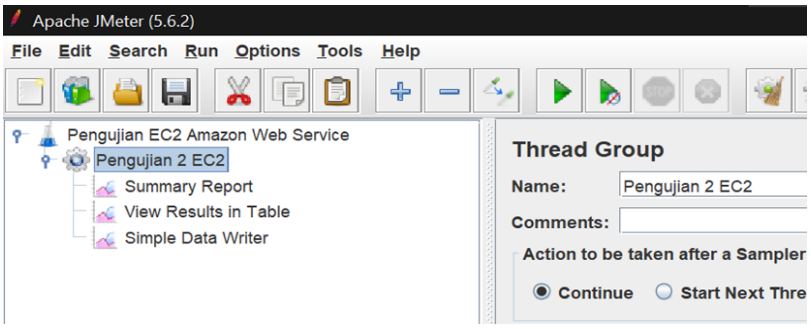
Berikut Langkah Test Plan dari pengujian kedua Web Server Amazon Web Service: Pertama, membuat Thread Group dengan nama “Pegujian 2 EC2” kemudian mengisi Thread Properties sesuai requirement pengujian.




Gambar 12. *Thread Group* Pengujian 2 *EC2*

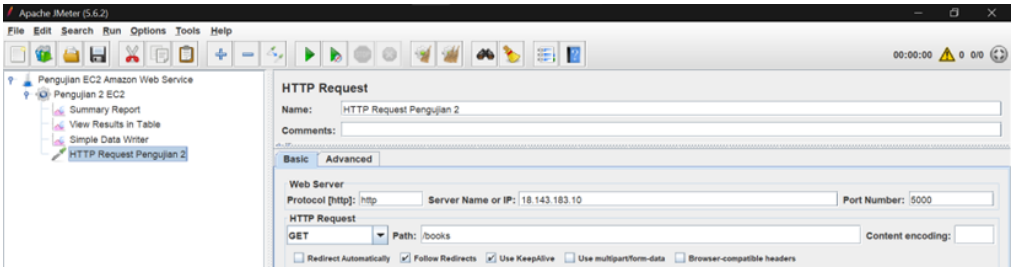
Kedua, menambahkan *listener* untuk melihat hasil disetiap pengujian yang di lakukan pada *Theread Group*. *Listerner* yang ditambahkan adalah *Summary Report, View Result in Table* dan *Simple* Data *Writer*.

*P-ISSN: 2622-6901*
*E-ISSN: 2622-6375*




Gambar 13. *Listener* Pada *Thread Group*

Ketiga, menambahkan sampler *HTTP Request* dengan nama "HTTP Request Pengujian 2"



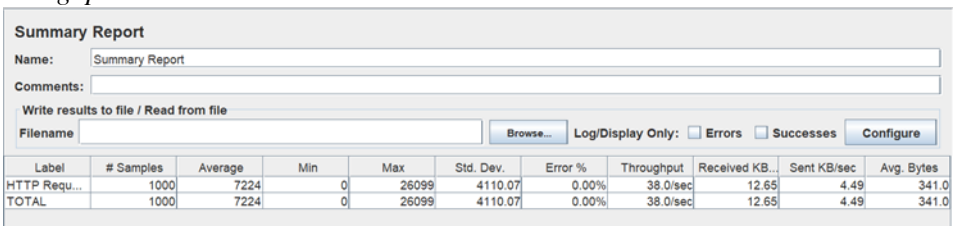

Gambar 14. *HTTP Request* Pengujian 2

Penjelasan, pada Gambar 14 pada kolom Protocol di isi dengan "http", sedangan pada kolom IP diisi dengan *IP Address EC2 AWS*, lalu pada *Port Number* di isi dengan *Port* 5000 sesuai dengan ketentuan *Inbound Rules Security Groups Firewall EC2*, lalu pada *path* di isi dengan "/books" untuk mengakses buku yang ada di *server*.

Ke-empat, Jalankan *Test*.

Hasil pengujian kedua dari EC2 Amazon Web Service adalah sebagai berikut:

*a. Throughput*




| Label | # Samples | Average | Min | Max | Std. Dev. | Error % | Throughput | Received KB... | Sent KB/sec | Avg. Bytes |
|---|---|---|---|---|---|---|---|---|---|---|
| HTTP Requ... | 1000 | 7224 | 0 | 26099 | 4110.07 | 0.00% | 38.0/sec | 12.65 | 4.49 | 341.0 |
| TOTAL | 1000 | 7224 | 0 | 26099 | 4110.07 | 0.00% | 38.0/sec | 12.65 | 4.49 | 341.0 |

Gambar 15. *Summary Report* Pengujian 2 *EC2*

Hasil *Throughput* dari *sampler HTTP Request* dalam pengujian kedua kali ini adalah 38.0/sec dengan *error samples* 0.0%

*b. Sample Time / Response Time*

Di dalam Gambar 20, terdapat hasil dari *sample time / response time*, di mana nilai yang dicatat untuk waktu *sample time / response time* adalah *average*. *Average* ini dihitung dari total nilai sample time / response time. Sehingga, nilai untuk sample time / response time pada pengujian awal *Web server Amazon Web Service* adalah 7224/ms(*milisecond*).

*c. Latency*

Untuk menemukan nilai Latency dari keseluruhan sample harus dihitung secara manual dengan menghitung rata – rata, dengan rumus:

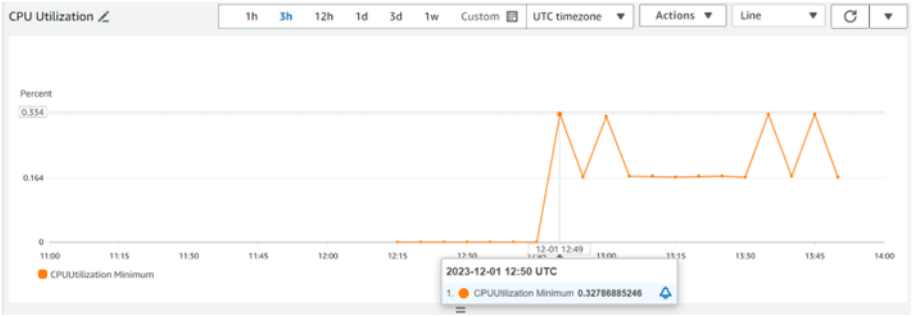
*P-ISSN: 2622-6901*
*E-ISSN: 2622-6375*

$$Latency = \frac{JumlahSeluruhnilaiLatency}{JumlahSample}$$

Maka nilai rata – rata latency pada pengujian kedua, yaitu:
$\frac{7.213.484}{1000}$ =7.213 /ms(*milisecond*)

Pengujian dilakukan dari sample 1 mulai berjalan pada pukul 21:00:34, sampai Sample nomor 1000 berakhir pada pukul 21:00:34.

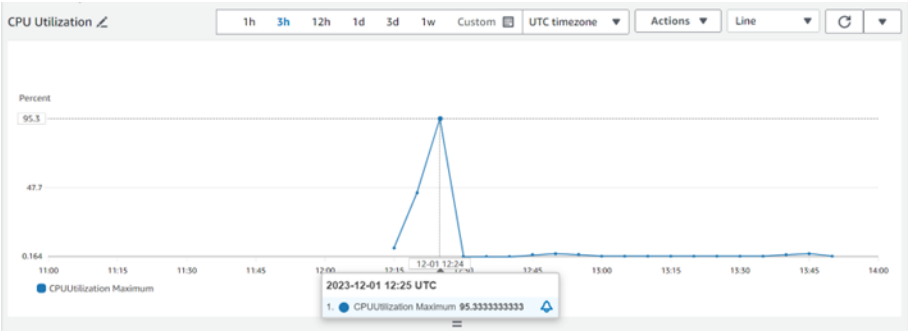
1. *CPU Utilization Minimum*




Gambar 16. *CPU Utilization Minimum* Pengujian 2

Menunjukan *Minimum* (terendah) penggunaan *CPU Utilization* di angka 0,327%.

2. *CPU Utilization Maximum*




Gambar 17. *CPU Utilization Maximum* Pengujian 2

Menunjukan Maximum (tertinggi) penggunaan *CPU Utilization* di angka 95,3%

3. CPU Utilization Average


Gambar 18. *CPU Utilization Average*

*P-ISSN: 2622-6901*
*E-ISSN: 2622-6375*

Menunjukkan *Average* (Rata – rata) penggunaan *CPU Utilization* di angka 36,9%

Pengujian dilakukan pada *Web Server Amazon Web Service* melalui 2 tahap pengujian yang melibatkan penggunaan *thread / user* yang berbeda:

Tabel 1. Hasil Pengujian *Web Server EC2*

| ***Test*** | ***Throughput (second)*** | ***Sample/ Respons Time (Milisecond)*** | ***Latency (Milisecond)*** | ***CPU Utilization*** | | | ***Sample (User)*** | ***Error Sample (%)*** |
|---|---|---|---|---|---|---|---|---|
| | | | | ***Minimal*** | ***Maximum*** | ***Average*** | | |
| **1.** | 48.1 | 59 | 72,062 | 0,338 | 1,82 | 0,699 | 500 | 0 |
| **2.** | 38.0 | 7.224 | 7.213 | 0,327 | 95,3 | 36,9 | 1000 | 0 |

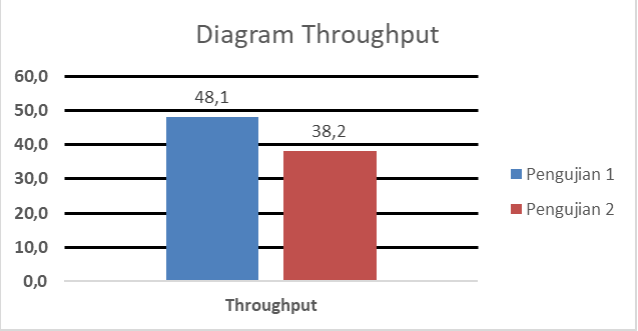
Pada pengujian pertama *sample / user* mengalami *error* sebesar 0%, yang berarti pada percobaan 500 total *sample user* tidak mengalami kegagalan sama sekali. Begitu pula pada pengujian kedua *sample / user* mengalami *error* sebesar 0% , yang berarti pada percobaan 1000 total *sample user* tidak mengalami kegagalan sama sekali.

## B. Pembahasan

Setelah Pengujian pada *Web Server Amazon Web Service*, nilai yang diperoleh akan dianalisis untuk setiap parameter yang diujiakan. Berikut adalah analisis dari hasil pengujian berdasarkan parameter yang telah diuji:

1. *Throughput*

Berikut diagram nilai *throughput* pada kedua pengujian:

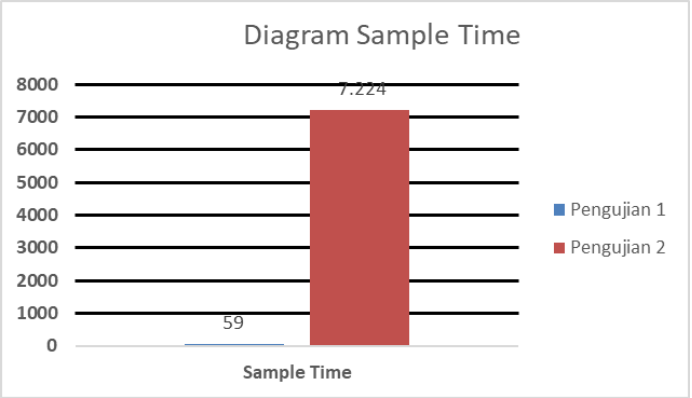



Gambar 19. Diagram *Thoughput*

Pada diagram (Gambar 19) untuk pengujian pertama, *Amazon Web Service* menunjukkan *Throughput* sebesar 48,1/*second*. Sementara pada pengujian kedua, throughput mencapai 38,8/second. oleh karena itu, berdasarkan hasil tersebut, pengujian pertama menunjukan keunggulan yang lebih besar dalam pengujian.

*P-ISSN: 2622-6901*
*E-ISSN: 2622-6375*

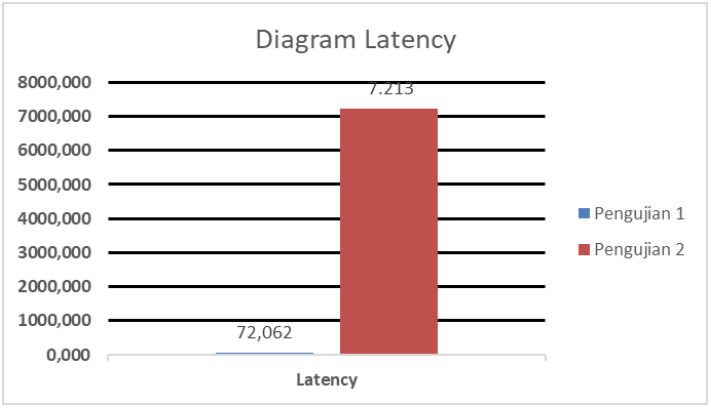
2. *Sample Time / Respons Time*
   Berikut diagram nilai *Sample Time / Response Time* pada kedua pengujian:




Gambar 20. Diagram *Sample Time*

Pada diagram (Gambar 20) untuk pengujian pertama, *Amazon Web Service* menunjukan *Sample Time / Response Time* sebesar 59/*millisecond*, dan pada pengujian *Sample Time / Response Time* kedua menunjukkan sebesar 7.224/*millisecond*, Sehingga dengan hasil pengujian kedua lebih unggul dalam parameter *Sample Time / Response Time.*

3. Latency
   Berikut diagram nilai *Latency* pada kedua pengujian:




Gambar 21. Diagram *Latency*

Pada diagram (Gambar 21) untuk pengujian pertama memiliki nilai 72,062/*millisecond*, sedangkan pada pengujian kedua memiliki nilai 7.213/*millisecond*. Sehingga dengan hasil pengujian kedua lebih unggul dalam parameter *latency.*

## KESIMPULAN

Berdasarkan hasil pengujian pada *EC2 Free Tier* membuktikan bahwa walaupun gratis *server* tetap bisa menanggung beban yang besar berdasarkan pengujian pada *server* dalam hal ini menggunakan pengujian 500 *user* dalam periode 10 detik dan 1000 user dalam periode 1 detik. Dibuktikan dengan kedua pengujian mengalami *error* 0%.

## SARAN

Penulis menyadari bahwa hasil kerja ini jauh dari sempurna. Adapun untuk pengembangan lebih lanjut, beberapa hal dapat dilakukan seperti penerapan load balancing dimana membagi beban berdasarkan banyaknya koneksi yang sedang dilayani oleh server apakah lebih optimal. Atau mengganti alat tes dengan alternatif selain Jmeter, contohnya LoadView by Dotcom-Monitor, atau WebLOAD.

## DAFTAR PUSTAKA

A. Amrullah, A. Nugroho, and Z. Ramadhan, "Perbandingan Kinerja Web Server Pada Penyedia Layanan Cloud Microsoft Azure Dan Amazon Web Services," *J. Inform. Teknol. dan Sains,* vol. 5, no. 1, pp. 92–97, 2023, doi: 10.51401/jinteks.v5i1.2487.

Amazon, "Amazon Web Service." 2023. [Online]. Available: https://aws.amazon.com/id/free/?nc2=h_ql_pr_ft. Diakses pada 18 Desember 2023.

Berenberg, A., & Calder, B. (2023). *Deployment Archetypes for Cloud Applications. ACM Computing Surveys,* 55(3), 1–48. https://doi.org/10.1145/3498336.

Fadila, A., Nasir, M., & Safriadi, S. (2023). Implementasi Sistem Load Balancing Web Server Pada Jaringan public Cloud Computing Menggunakan Least Connection. *Journal of Artificial Intelligence and Software Engineering,* 3(2), 50-55.

Fandy, Rosmasari, and G. M. Putra, "Pengujian Kinerja Web Server Atas Penyedia Layanan Elastic Cloud Compute (EC2) Pada Amazon Web Services (AWS)," *Adopsi Teknol. dan Sist. Inf.*, vol. 1, no. 1, pp. 21–35, 2022, doi: 10.30872/atasi.v1i1.45.

Kanikathottu, H. (2020). *AWS security cookbook: practical solutions for managing security policies, monitoring, auditing, and compliance with AWS*. Packt Publishing Ltd.

M. Fransisca and Y. Yunus, "Validitas Pengembangan Media Pembelajaran Blended Learning Berbasis Cloud Computing Tingkat Sekolah Menengah Atas Kota Padang," *Pros. Semin. Nas. Ris. Inf. Sci.,* vol. 1, no. 3, p. 103, 2019, doi: 10.30645/senaris.v1i0.13.

Pradana, D., Cahyanto, T. A., Umilasari, R., Oktavianto, H., & Zakiyyah, A. M. (2022). Perbandingan Kinerja Web Server Terhadap Performa Integrasi Multi Basis Data. BIOS: *Jurnal Teknologi Informasi dan Rekayasa Komputer*, 3(2), 36-45.

Prasetyo, S. E. (2021). Design and implementation of lightweight virtualization using docker container in distributing web application with experimental methods. *Journal Of Informatics And Telecommunication Engineering,* 4(2), 270-276.

Sutanto, S., Gunawan, W., & Faeshal, F. (2021, February 25). Arsitektur Container Docker Pada Aplikasi Expert Assist Dengan Teknologi Node.Js, Express Framework & Cloud Database NoSql MongoDB Atlas. *Jurnal Sistem Informasi Dan Informatika (Simika*), 4(1), 73–89. https://doi.org/10.47080/simika.v4i1.1189.

Wu, Z., & Madhyastha, H. V. (2013). Understanding the latency benefits of multi-cloud webservice deployments. *ACM SIGCOMM Computer Communication Review*, *43*(2), 13–20. https://doi.org/10.1145/2479957.2479960.