

**EVALUASI KINERJA WEB SERVER NGINX, LITESPEED, DAN APACHE  
TERHADAP BEBAN KERJA EKSTREM DENGAN METODE STRESS  
TESTING PADA LAYANAN *AMAZON ELASTIC*  
*COMPUTE CLOUD* (AWS EC2)**

**SKRIPSI**

**FAJAR FAKHRI**

**171401062**



**PROGRAM STUDI ILMU KOMPUTER  
FAKULTAS ILMU KOMPUTER DAN TEKNOLOGI INFORMASI  
UNIVERSITAS SUMATERA UTARA  
MEDAN  
2024**

**EVALUASI KINERJA WEB SERVER NGINX, LITESPEED, DAN APACHE  
TERHADAP BEBAN KERJA EKSTREM DENGAN METODE STRESS  
TESTING PADA LAYANAN *AMAZON ELASTIC*  
*COMPUTE CLOUD* (AWS EC2)**

**SKRIPSI**

Diajukan untuk melengkapi tugas dan memenuhi syarat memperoleh ijazah Sarjana  
Ilmu Komputer

**FAJAR FAKHRI**

**171401062**



**PROGRAM STUDI S1 ILMU KOMPUTER  
FAKULTAS ILMU KOMPUTER DAN TEKNOLOGI INFORMASI  
UNIVERSITAS SUMATERA UTARA  
MEDAN  
2024**

## PERSETUJUAN

Judul : EVALUASI KINERJA WEB SERVER  
NGINX, LITESPEED, DAN APACHE  
TERHADAP BEBAN KERJA EKSTRIM  
DENGAN METODE STRESS TESTING PADA  
LAYANAN AMAZON ELASTIC COMPUTE  
CLOUD (EC2)

Kategori : SKRIPSI

Nama : FAJAR FAKHRI

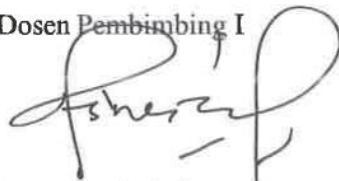
Nomor Induk Mahasiswa : 171401062

Program Studi : SARJANA (S-1) ILMU KOMPUTER

Fakultas : FAKULTAS ILMU KOMPUTER DAN  
TEKNOLOGI INFORMASI UNIVERSITAS  
SUMATERA UTARA

Komisi Pembimbing :

Dosen Pembimbing I



Amer Sharif S.Si., M.Kom  
NIP. 196910212021011001

Dosen Pembimbing II



Handrizal S.Si., M.Comp.Sc  
NIP. 197706132017061001

Diketahui/Disetujui oleh  
Program Studi S1 Ilmu Komputer

Ketua



Dr. Amalia S.T., M.T.  
NIP. 197812212014042001

**PERNYATAAN****EVALUASI KINERJA WEB SERVER NGINX, LITESPEED, DAN  
APACHE TERHADAP BEBAN KERJA EKSTRIM DENGAN METODE  
STRESS TESTING PADA LAYANAN AMAZON ELASTIC COMPUTE  
CLOUD (AWS EC2)****SKRIPSI**

Saya mengakui bahwa Skripsi ini adalah hasil karya saya sendiri, kecuali beberapa kutipan dan ringkasan yang masing-masing telah disebutkan sumbernya.

Medan, Juni 2024



Fajar Fakhri

171401062

## UCAPAN TERIMA KASIH

Penulis mengucapkan rasa syukur kepada Allah SWT atas segala berkat-Nya selama proses penelitian dan penulisan skripsi ini, yang merupakan bagian dari persyaratan untuk menyelesaikan pendidikan Program Studi Sarjana Ilmu Komputer di Fakultas Ilmu Komputer dan Teknologi Informasi Universitas Sumatera Utara, serta meraih gelar sarjana.

Tak lupa penulis juga ingin mengucapkan penghargaan kepada seluruh pihak yang telah memberikan dukungan dalam penyusunan skripsi ini. Pihak-pihak tersebut antara lain:

1. Bapak Prof. Dr. Muryanto Amin, S.Sos., M.Si. Rektor Universitas Sumatera Utara.
2. Ibu Dr. Maya Silvi Lydia B.Sc. M.Sc., Dekan Fakultas Ilmu Komputer dan Teknologi Informasi Universitas Sumatera Utara.
3. Ibu Dr. Amalia ST., M.T., Ketua Program Studi S-1 Ilmu Komputer Universitas Sumatera Utara.
4. Bapak Amer Sharif S.Si., M.Kom Dosen Pembimbing I yang telah memberikan bimbingannya kepada penulis selama proses penyusunan skripsi.
5. Bapak Handrizal S.Si., M.Comp.Sc Dosen Pembimbing II yang bersedia memberikan bimbingannya kepada penulis selama proses penyusunan skripsi.
6. Seluruh dosen Program Studi S-1 Ilmu Komputer Fakultas Ilmu Komputer dan Teknologi Informasi Universitas Sumatera Utara.
7. Para staf pegawai Fakultas Ilmu Komputer dan Teknologi Informasi Universitas Sumatera Utara.
8. Dan seluruh pihak yang banyak membantu dan tidak dapat disebutkan namanya satu-persatu
9. Keluarga serta orang tua peneliti, dan sahabat yang selalu memberi doa, serta motivasi kepada penulis dalam menyelesaikan skripsi ini.

Penulis menyadari bahwa dalam skripsi ini mungkin memiliki kekurangan dan kesalahan. Namun demikian, penulis berharap agar skripsi ini tetap bermanfaat setelah dibaca dan mungkin dapat digunakan sebagai referensi bagi penelitian dimasa mendatang.

Medan, Juni 2024

Penulis,

A handwritten signature in black ink, appearing to read 'Fajar Fakhri', with a stylized, cursive script.

Fajar Fakhri

## ABSTRAK

### EVALUASI KINERJA WEB SERVER NGINX, LITESPEED, DAN APACHE TERHADAP BEBAN KERJA EKSTREM DENGAN METODE STRESS TESTING PADA LAYANAN AMAZON ELASTIC COMPUTE CLOUD (AWS EC2)

Kehadiran situs web dan web aplikasi menjadi kunci dalam menyediakan layanan dan informasi kepada pengguna. Semakin besar dan kompleks sebuah situs web atau web aplikasi, maka semakin besar juga beban kerja yang harus ditangani oleh web server yang menghostingnya. Untuk menghadapi tantangan ini, penggunaan server web yang tepat menjadi sangat penting. Pada penelitian ini, penulis melakukan evaluasi kinerja web server dengan melakukan metode *Stress Testing* pada 3 web server populer yaitu *Nginx*, *Litespeed*, dan *Apache*, dengan menggunakan layanan *Amazon Elastic Compute Cloud* (AWS EC2). *Stress Testing* ialah bentuk pengujian intensif yang dilakukan secara sengaja untuk mengetahui bagaimana kemampuan kinerja dari suatu elemen dalam menghadapi beban kerja yang ekstrem. Ketiga web server ini akan menangani situasi beban kerja ekstrem dan penulis akan melakukan evaluasi dari hasil *Stress Testing* pada parameter seperti *response time*, *error*, *throughput*, *data sent*, dan *data received*. Hasil dari evaluasi didapatkan bahwa, *LiteSpeed* memiliki tingkat *error request* yang paling rendah dari ketiga server dan *Apache* memiliki data *Sent* dan *Received* paling rendah yang bagus untuk optimisasi *bandwidth* dari ketiga web server. Evaluasi ketiga web server ini akan membantu kita dalam memilih web server yang tepat dan sesuai dengan kebutuhan yang kita perlukan dalam membangun sebuah situs web dan web aplikasi.

**Kata Kunci** : Stress Testing, Web Server, *Amazon Elastic Compute Cloud*.

## **ABSTRACT**

### **EVALUATION OF THE PERFORMANCE OF NGINX, LITESPEED, AND APACHE WEB SERVERS UNDER EXTREME WORKLOADS USING STRESS TESTING METHODS ON AMAZON ELASTIC COMPUTE CLOUD (AWS EC2)**

The presence of websites and web applications is key in providing services and information to users. The larger and more complex a website or web application is, the greater the workload that must be handled by the web server hosting it. To tackle this challenge, choosing the right web server is crucial. In this study, the author evaluates web server performance by conducting Stress Testing on three popular web servers: Nginx, Litespeed, and Apache while using the Amazon Elastic Compute Cloud (AWS EC2) service. Stress Testing is an intensive testing method deliberately carried out to determine how well an element can perform under extreme workload conditions. These three web servers will handle extreme workload situations, and the author will evaluate the Stress Testing results based on parameters such as response time, errors, throughput, data sent, and data received. The evaluation results show that LiteSpeed has the lowest error request rate among the three servers, and Apache has the lowest data sent and received, which is beneficial for bandwidth optimization. The evaluation of these three web servers will help us choose the right web server that meets our needs in building a website and web application.

**Keywords** : Stress Testing, Web Server, Amazon Elastic Compute Cloud.



## DAFTAR ISI

PERSETUJUAN .....	i
PERNYATAAN .....	ii
UCAPAN TERIMA KASIH.....	iii
ABSTRAK.....	v
ABSTRACT.....	vi
DAFTAR ISI.....	vii
BAB 1 .....	1
1.1. Latar Belakang .....	1
1.2. Rumusan Masalah.....	2
1.3. Batasan Masalah .....	2
1.4. Tujuan Penelitian .....	3
1.5. Manfaat Penelitian .....	3
1.6. Metodologi Penelitian.....	4
1.7. Sistematika Penulisan .....	5
1.8. Penelitian yang Relevan.....	5
BAB II.....	7
2.1. Topologi Jaringan.....	7
2.2. Web Server.....	7
2.3. Nginx.....	8
2.4. Apache.....	9
2.5. Litespeed .....	9
2.6. Apache Jmeter .....	10
2.7. Cloud Computing .....	10
2.8. Amazon Web Service.....	12
2.9. Amazon Elastic Compute Cloud (EC2) .....	12
2.10. Stress Testing.....	13
2.11. Beban Kerja .....	15
BAB III .....	17
3.1. Tahapan Penelitian .....	17
3.2. Analisis Masalah .....	17
3.3. Perancangan Sistem.....	18

3.3.1.	Alat dan Bahan Pengujian.....	19
3.3.2.	Skema Pengujian Sistem.....	20
3.3.3.	Parameter Kinerja .....	24
BAB IV .....		25
4.1.	Launch Instance.....	25
4.2.	Pemasangan Web Server Nginx .....	31
4.3.	Pemasangan Web Server Apache.....	34
4.4.	Pemasangan Web Server LiteSpeed.....	35
4.5.	Pengaturan Apache JMeter.....	38
4.6.	Percobaan Kinerja Server.....	40
4.7.	Pembahasan.....	44
BAB V .....		46
5.1.	Kesimpulan.....	46
5.2.	Saran.....	46
DAFTAR PUSTAKA .....		47

**DAFTAR TABEL**

	Halaman
Tabel 4.4. Perbandingan Parameter Error .....	41
Tabel 4.5. Perbandingan Parameter Throughput .....	42
Tabel 4.6. Perbandingan Parameter Data Recieved.....	42
Tabel 4.7. Perbandingan Parameter Data Send.....	43

## DAFTAR GAMBAR

	Halaman
Gambar 2.1. Perancangan Jaringan.....	6
Gambar 2.2. Apache.....	8
Gambar 1.1. Survei Netcraft Web Server Bulan Febuari 2024 .....	8
Gambar 2.3. Prosedur Stress Testing.....	12
Gambar 3.1. Struktur Penelitian.....	16
Gambar 3.2. Topologi Jaringan.....	18
Gambar 3.3. Tahapan Percobaan Pada Apache web Server .....	20
Gambar 3.4. Tahapan Percobaan Pada Nginx Web Server .....	21
Gambar 3.5. Tahapan Percobaan Pada Litespeed Web Server .....	22
Gambar 4.1. EC2 Dashboard .....	24
Gambar 4.2. Launch Instance AWS EC2 .....	25
Gambar 4.3. Instance Nginx Web Server .....	25
Gambar 4.4. Instance Apache Web Server .....	26
Gambar 4.5. Instance LiteSpeed Web Server .....	26
Gambar 4.6. Sistem Operasi Web Server .....	27
Gambar 4.7. Jenis Instance AWS EC2 .....	27
Gambar 4.8. Generate new Key Pair.....	28
Gambar 4.9. Pengaturan Network & Firewall .....	29
Gambar 4.10. Pengaturan Penyimpanan Server.....	29
Gambar 4.11. Status Launch Instance.....	30
Gambar 4.12. Remote Server dengan AWS Console .....	31
Gambar 4.13. Instalasi Nginx Web Server .....	31
Gambar 4.14. Status Nginx Aktif .....	32
Gambar 4.15. Status Nginx Aktif Pada Browser .....	32
Gambar 4.16. Instalasi Apache Web Server .....	33
Gambar 4.17. Status Apache Web Server Aktif .....	34
Gambar 4.18. Status Apache Web Server Aktif Pada Browser.....	34
Gambar 4.19. Instalasi OpenLiteSpeed Web Server .....	35
Gambar 4.20. Status Aktif OpenLiteSpeed Web Server.....	35
Gambar 4.21. Konfigurasi username dan password Litespeed.....	36
Gambar 4.22. Konfigurasi port Litespeed.....	36
Gambar 4.23. Pengaturan Apache JMeter 700 Pengguna dalam 1 Second.....	38
Gambar 4.24. Pengaturan Apache JMeter 700 Pengguna dalam 5 Second.....	38
Gambar 4.25. Pengaturan Apache JMeter 900 Pengguna dalam 1 Second.....	39
Gambar 4.26. Pengaturan Apache JMeter 900 Pengguna dalam 5 Second.....	39
Gambar 4.27. Pengaturan Apache JMeter 1100 Pengguna dalam 1 Second.....	40
Gambar 4.28. Pengaturan Apache JMeter 1100 Pengguna dalam 5 Second.....	40
Gambar 4.29. Halaman Percobaan.....	41
Gambar 4.30. Hasil Grafik response time 1 Second.....	43
Gambar 4.31. Hasil Grafik response time 5 Second.....	44

## BAB 1

### PENDAHULUAN

#### 1.1. Latar Belakang

Dalam era digital yang mengalami pertumbuhan dengan cepat, kehadiran situs web dan aplikasi web menjadi kunci dalam menyediakan layanan dan informasi kepada pengguna. Kinerja situs web memiliki peran penting dalam menentukan pengalaman pengguna, kepuasan, dan kepercayaan pengguna terhadap layanan tersebut. Semakin besar dan kompleks sebuah situs web atau aplikasi web, semakin besar juga beban kerja yang harus ditangani oleh *web server* yang menghostingnya.

Untuk menghadapi tantangan ini, penggunaan *web server* yang tepat sangat penting. Dalam konteks ini, *web server* populer seperti Nginx, LiteSpeed, dan Apache memainkan peran yang signifikan dalam mendukung aplikasi web modern. Kinerja *web server* ini memiliki dampak langsung terhadap respon cepat, tingkat ketersediaan, dan skalabilitas aplikasi web.

Namun, dalam situasi beban kerja ekstrim, kemampuan sebuah *web server* untuk menjaga kinerja dan ketersediaannya menjadi sangat penting. Beban kerja ekstrim dapat muncul akibat lonjakan lalu lintas yang tidak terduga, situasi tertentu seperti acara penjualan besar-besaran online, atau serangan DDoS (*Distributed Denial of Service*). Dalam konteks ini, metode pengujian stres menjadi krusial untuk mengevaluasi kemampuan web server dalam menghadapi beban kerja yang tidak biasa.

*Amazon Elastic Compute Cloud* (AWS EC2) adalah platform hosting yang populer dan banyak digunakan untuk menyediakan infrastruktur cloud yang elastis dan dapat diandalkan untuk menjalankan aplikasi web. Penggunaan AWS EC2 sebagai platform hosting menawarkan fleksibilitas dan skalabilitas dalam menghadapi berbagai tipe beban kerja.

Dari survei yang dilakukan oleh Netcraft pada bulan Februari Apache dan Nginx masih menjadi *web server* yang paling banyak digunakan, dengan Apache yang memiliki pengguna hingga 20% di dunia, lalu Nginx dengan pengguna hingga 18% di dunia. Bagi para pengguna *web server* berbasis *open source*, Apache dan Nginx selalu menjadi

pilihan utama, namun selain kedua *web server* tersebut ada beberapa *web server* lainnya yang berbasis *open source*, salah satunya adalah Litespeed. Berdasarkan survei yang sama, Litespeed memiliki pengguna sebanyak 6% di dunia dan angka tersebut cukup tinggi. (Netcraft, 2024)

Oleh karena itu saya menggunakan ketiga *web server* tersebut yaitu Apache, Nginx, dan Litespeed. Ketiga *web server* ini juga memiliki kelebihan masing-masing yang membuat *web server* ini ramai untuk digunakan, seperti Apache yang memiliki segudang fitur seperti .htaccess, SSL, HTTP/2, PHP, dan masih banyak lagi. Lalu untuk Nginx terkenal dengan kecepatan menyajikan data yang stabil, serta kemudahan pada konfigurasinya. Sedangkan untuk Litespeed yang lebih cepat dalam mengolah PHP dan keamanan yang lebih baik. (Hostinger, 2024)

Dalam penelitian ini, penulis melakukan evaluasi kinerja tiga *web server* populer, yaitu Nginx, LiteSpeed, dan Apache, dengan memanfaatkan metode *Stress Testing* pada layanan AWS EC2 dengan parameter yaitu *Response time*, *Throughput*, *Error*, *Data Sent*, dan *Data Received*. Penelitian ini bertujuan untuk memahami bagaimana ketiga *web server* ini menangani situasi beban kerja ekstrim dan bagaimana perbandingan ketiga *web server* ini. Hasil penelitian ini akan memberikan wawasan berharga dalam pemilihan dan penggunaan *web server* yang sesuai dengan kebutuhan aplikasi web yang beragam.

## 1.2. Rumusan Masalah

Permasalahan yang dibahas pada penelitian ini adalah dengan jumlah pengguna internet yang semakin besar menyebabkan permintaan pada *web server* juga semakin tinggi dan beban kerja server yang juga semakin berat, sehingga dibutuhkan evaluasi pada *web server* dalam hal ini, yaitu Apache, Nginx, dan Litespeed untuk mengetahui kinerja ketiga *web server* dengan menggunakan metode *Stress Testing* pada beban kerja ekstrim, dan melakukan perbandingan terhadap ketiga *web server* tersebut.

## 1.3. Batasan Masalah

Batasan Masalah diperuntukkan agar dapat terhindar dari perluasan cakupan dari apa yang akan dibahas pada penelitian ini sehingga tidak akan terjadi penyimpangan dari pokok permasalahan. Batasan-batasan dalam penelitian ini meliputi :

1. Aplikasi yang digunakan untuk *Stress Testing* adalah Apache JMeter.
2. Dalam pengujian ini, parameter yang diukur meliputi waktu respons, tingkat kesalahan, throughput, pengiriman data, dan penerimaan data.
3. Web server yang digunakan adalah Apache, Nginx dan Litespeed.
4. *Platform* yang digunakan adalah AWS EC2

#### **1.4. Tujuan Penelitian**

Tujuan penelitian ini adalah untuk mengenali dan menyediakan informasi tentang kinerja web server Nginx, Apache, dan Litespeed. Parameter yang akan diuji dalam penelitian ini adalah indikator yang relevan untuk menilai kinerja sebuah web server. Untuk *error request*, *response time*, *send and data received* semakin kecil angka pengujiannya, maka semakin baik kinerjanya. Lalu untuk *throughput* semakin besar angka pengujiannya, maka semakin baik kinerjanya. Selain itu tujuan penelitian ini juga dapat menjadi sebuah acuan untuk memilih sebuah web server yang nantinya digunakan untuk menghosting sebuah aplikasi web.

#### **1.5. Manfaat Penelitian**

Manfaat dari penelitian ini adalah sebagai berikut.

1. Memberikan bantuan kepada pengelola layanan dalam memilih server yang sesuai dengan menyajikan informasi tentang kelebihan dan kekurangan dari setiap server.
2. Dengan pemahaman yang lebih mendalam tentang faktor-faktor yang memengaruhi performa *web server*, pengelola layanan dapat mengoptimalkan konfigurasi untuk mencapai performa yang lebih baik lagi, dalam menghadapi situasi beban kerja yang tinggi atau lonjakan lalu lintas.
3. Dalam situasi beban kerja yang ekstrim penelitian ini dapat membantu pengelola layanan untuk merancang sebuah strategi untuk menjaga ketersediaan layanan. Hal ini dapat mengurangi resiko gangguan layanan yang dapat berdampak pada bisnis dan reputasi
4. Selain untuk pengelola layanan, penelitian ini juga bermanfaat untuk pengembang aplikasi. Pengembang aplikasi dapat mengambil sebuah keputusan yang lebih terinformasi berdasarkan nilai penelitian ini. Mereka dapat melakukan penyesuaian

atau perubahan yang diperlukan guna meningkatkan kinerja dan ketersediaan aplikasi web yang akan dibangun.

### 1.6. Metodologi Penelitian

Metode penelitian yang dilakukan dalam penelitian ini adalah:

#### 1. Studi Pustaka

Dalam penelitian ini penulis mengumpulkan informasi yang berhubungan dengan web server, *stress test*, Nginx, Litespeed, Apache dan topik lain yang dibahas pada penelitian berdasarkan buku, internet, maupun sumber referensi terpercaya lainnya.

#### 2. Analisis Kebutuhan

Dalam evaluasi kinerja web server Nginx, Litespeed dan Apache terhadap beban kerja ekstrim dengan metode stress testing pada layanan Amazon EC2 diperlukannya akun *Amazon Web Services (AWS)* dan keperluan lainnya.

#### 3. Perancangan Server

Pada tahapan perancangan server akan dilakukan perancangan pada masing-masing *instance* yang ada pada layanan Amazon EC2. Pada setiap instance akan di install web server yang berbeda, yaitu Nginx, LiteSpeed dan Apache, serta menggunakan konfigurasi standar dari masing-masing web server.

#### 4. Instalasi Server

Pembuatan sistem atau persiapan penginstallan web server disetiap instance yang berbeda serta melakukan konfigurasi dan penginstallan paket standar seperti php dan wordpress yang berguna sebagai media web uji coba dalam melakukan stress testing.

#### 5. Pengujian Server

Pada tahapan ini akan dilakukan pengujian dengan Aplikasi JMeter yang akan menguji setiap parameter yang sudah dijelaskan sebelumnya.



## 1.7.Sistematika Penulisan

Struktur penulisan dalam skripsi ini terdiri dari lima bagian pokok, yang dijabarkan sebagai berikut.

### **BAB I PENDAHULUAN**

Latar belakang, rumusan masalah, batasan masalah, tujuan dan manfaat penelitian, metodologi penelitian, dan sistematika penulisan.

### **BAB II LANDASAN TEORI**

Bab ini memaparkan berbagai teori yang berhubungan dengan Amazon Cloud Compute, Nginx, Apache dan Litespeed.

### **BAB III ANALISIS DAN PERANCANGAN SISTEM**

Bab ini memaparkan analisis dan perancangan server yang akan di bangun pada layanan Amazon Cloud Compute (AWS EC2).

### **BAB IV IMPLEMENTASI DAN PENGUJIAN**

Bab ini memaparkan bagaimana hasil implementasi server jika menggunakan web server Apache, Nginx ataupun Litespeed, serta hasil pengujian stress test menggunakan JMeter.

### **BAB V KESIMPULAN DAN SARAN**

Bab ini memaparkan kesimpulan yang didapat dari pengujian yang telah dilakukan serta saran yang ditujukan untuk penelitian berikutnya.

## 1.8.Penelitian yang Relevan

Penelitian lain yang berkaitan dengan penelitian ini yaitu :

1. Andhica dan Irwan , (2017) melakukan penelitian yang berjudul “Performa Kinerja Web Server Berbasis *Ubuntu Linux dan Turnkey Linux*”. Penelitian ini menguji bagaimana performa server web berdasarkan beberapa paramater yaitu dari *request route*, *reply time*, dan *throughput*. Aplikasi Httpref dipakai untuk melihat performa dari masing-masing web server.
2. Candra, (2019) melakukan penelitian yang berjudul “Analisis Performansi Antara Apache dan Nginx *Web Server* Dalam Menangani Client Request”. Penelitian ini melakukan pengujian kinerja web server berdasarkan kemampuannya dalam menangani permintaan dari klien menggunakan aplikasi Apache Bench, dengan beban mulai dari 100 pengguna hingga 1.000.000 pengguna. Hasil dari pengujian

pada web server Apache dan Nginx diharapkan dapat memberikan panduan dalam pemilihan web server yang akan digunakan untuk membangun aplikasi web, terutama jika aplikasi tersebut akan menerima banyak permintaan dari klien.

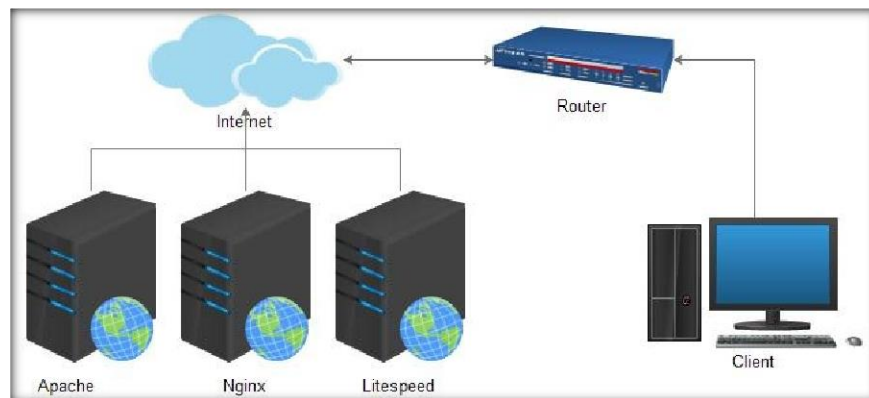
3. Busran & Ridwan, (2022) melakukan penelitian yang berjudul “Analisis Perbandingan Performa Apache *Web Server* dan Nginx Menggunakan Apache JMeter”. Penelitian ini menguji kinerja web server dengan parameter *response time* dan *throughput* dalam menangani permintaan oleh klien menggunakan JMeter pada beban *request* 100, 1000 dan 10000 pengguna. Dari hasil percobaan pada server web Apache dan Nginx, kesimpulannya adalah bahwa Nginx merespons klien lebih efisien, dengan waktu respons yang lebih cepat daripada web server Apache.

## BAB II

### LANDASAN TEORI

#### 2.1. Topologi Jaringan

Terdapat beberapa topologi jaringan, salah satunya adalah topologi jaringan web server yang biasanya dipakai oleh *client*, dapat dilihat pada **Gambar 2.1.** di bawah ini



**Gambar 2.1.** Topologi Jaringan

Gambar di atas menunjukkan contoh desain jaringan web server yang memungkinkan interaksi antara klien dan server melalui internet. Langkah awal yang diambil oleh klien saat ingin mengakses web server adalah melakukan koneksi melalui sebuah router yang terhubung ke internet. Dengan router, klien dapat melakukan permintaan atau request pada web server untuk mengakses suatu situs web. Server kemudian merespon permintaan dari klien sehingga klien dapat mengakses situs web yang diminta.

#### 2.2. Web Server

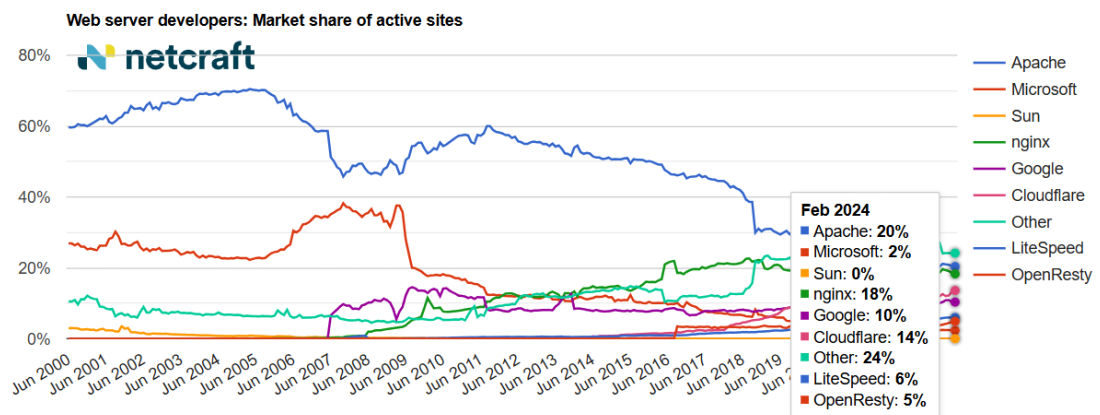
Web server merupakan perangkat lunak yang bertugas untuk menerima *request* HTTP (*HyperText Transfer Protocol*) atau HTTPS (*HTTP Secure*) yang dikirim oleh klien melalui peramban web, dan merespon permintaan tersebut dengan menyajikan halaman web dalam bentuk dokumen HTML (*HyperText Transfer Protocol*) atau format

lainnya. Web server berperan sebagai tempat penyimpanan aplikasi web dan sebagai titik penerimaan permintaan dari klien. (Indra Warman & Zahni, 2013).

Saat mengambil halaman website, *client* akan memasukkan URL (*Uniform Resource Locator*) yang merupakan alamat unik pada web server. URL tersebut akan diterjemahkan oleh DNS (*Domain Name Service*) menjadi *IP address* yang akan mengarahkan kita pada web server. *Client* menggunakan Browser untuk mengirim permintaan ke server, dimana permintaan tersebut nantinya akan diproses oleh web server. Sebelum memproses permintaan HTTP, server web juga akan melakukan pemeriksaan keamanan. Pada web server, permintaan HTTP akan ditangani oleh server HTTP. Web server kemudian akan mengirimkan respon HTTP kepada *Client* dalam bentuk HTML dan browser akan memprosesnya menjadi halaman situs web.

### 2.3. Nginx

Nginx merupakan web server open source yang dibuat oleh Igor Sysoev dan pertama kali dirilis bulan Oktober 2004. Nginx memiliki memori yang lebih kecil dibandingkan server web lainnya.



**Gambar 2.2.** Survei Netcraft Web Server Bulan Febuari 2024

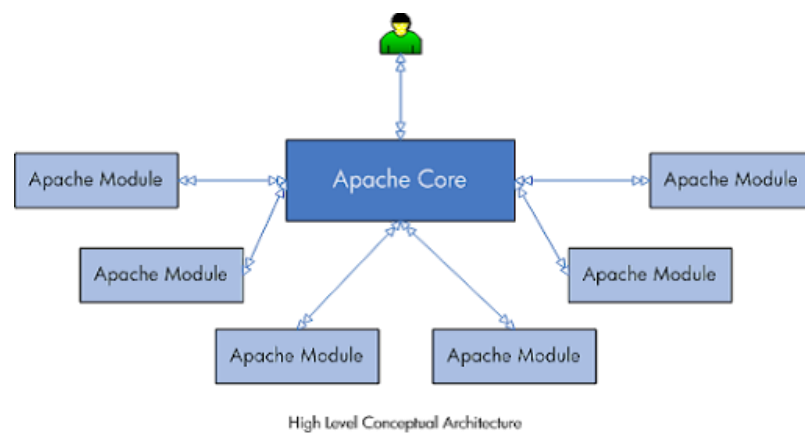
Dapat dilihat pada Gambar 2.2. merupakan survei yang dilakukan oleh Netcraft pada Febuari 2024, *web server* Nginx memiliki pengguna sebanyak 18% di dunia dan menempati posisi kedua untuk *web server* terbanyak digunakan.

Nginx dirancang berdasarkan arsitektur berbasis event (event-based architecture/EBA), di mana komponen-komponennya saling berinteraksi secara khusus menggunakan notifikasi event. Setelahnya, event-antrian diarahkan untuk diproses oleh pengendali event. Proses pengelolaan event berjalan secara berulang-ulang. Setelah

event dikerjakan, event tersebut dihapus dari antrian dan akan diteruskan ke proses selanjutnya

#### 2.4. Apache

Apache adalah perangkat lunak terbuka yang dikembangkan dan dikelola oleh Apache Software Foundation (ASF). Struktur Apache terdiri dari dua komponen utama, yaitu Apache Core dan Apache Module. Apache Core merupakan inti dari server web Apache, sedangkan Apache Module memperluas fungsionalitas inti Apache Core dengan menambahkan berbagai fitur tambahan. (Irza et al., 2017).



**Gambar 2.3.** Apache

Apache HTTP Server, diciptakan dengan tujuan untuk menjadi server HTTP terbuka yang dapat berjalan pada sistem operasi yang modern seperti UNIX dan Windows. Fokus utamanya adalah memberikan keamanan, efisiensi, dan skalabilitas kepada pengembang aplikasi, serta memastikan kompatibilitas dengan standar HTTP. Apache menjadi *web server* yang paling banyak digunakan mencapai 20% pengguna di seluruh dunia yang dapat dilihat pada Gambar 2.2.

#### 2.5. Litespeed

Litespeed adalah sebuah teknologi server web yang memiliki kemampuan untuk meningkatkan kecepatan akses server hingga 50 kali lebih cepat daripada server konvensional. OpenLiteSpeed, yang dikembangkan oleh LiteSpeed Technologies, menyajikan kinerja tinggi, kecepatan, ringan, stabilitas, dan kehandalan dalam mengelola beban kerja skala besar. (Technologies, n.d., 2021).

Litespeed memiliki beberapa fitur yaitu kompatibilitas dengan Apache, cepat dan stabil, kontrol panel yang terintegrasi, mempunyai dukungan HTTP/2, dan peningkatan

anti-DDoS. Dengan fitur tersebut, Litespeed memiliki keunggulan dari pada web server Apache.

## 2.6. Apache Jmeter

Apache Jmeter adalah sebuah aplikasi yang berjalan 100% Java. Jmeter dirancang untuk melakukan uji beban terhadap kinerja dan perilaku fungsional. Jmeter awalnya hanya dirancang untuk menguji web aplikasi, namun setelah itu berkembang untuk menguji fungsi lainnya.

JMeter merupakan alat yang digunakan untuk melakukan pengujian performa pada sumber daya yang statis maupun dinamis, serta aplikasi web dinamis. Alat ini memungkinkan simulasi beban tinggi pada server, grup server, jaringan, atau objek untuk menguji keandalan atau meninjau performa secara menyeluruh di berbagai jenis beban.

## 2.7. Cloud Computing

*Cloud computing* adalah bidang ilmu komputer yang sedang berkembang dan membawa sektor Teknologi Informasi ke Tingkat baru. *Cloud Computing* merupakan hasil pengembangan dari berbagai teknologi gabungan seperti komputasi terdistribusi, komputasi utilitas, virtualisasi, dan sebagainya untuk menyediakan sumber daya dan layanan TI melalui internet dengan modal pembayaran sesuai dengan pengguna. (Haris et al., 2018)

*Cloud Computing* adalah arsitektur berbasis internet yang menciptakan lingkungan komputasi untuk menyediakan ketersediaan, skalabilitas, dan fleksibilitas infrastruktur komputer. Ini dapat disimpulkan sebagai layanan komputasi yang menawarkan komputasi sebagai utilitas untuk memenuhi Kebutuhan pengguna dengan biaya yang sangat rendah berdasarkan pembayaran sesuai Kebutuhan. *Cloud Computing* menyediakan aplikasi, perangkat keras, dan perangkat lunak sebagai layanan berdasarkan permintaan.

Berdasarkan *National Institute of Standard Technology* (NIST) terdapat 3 (Tiga) model layanan yang disediakan oleh *Cloud Computing*, yaitu :

### **Model 1 — SaaS (Software as a Service).**

*Software as a Service* adalah model pengiriman perangkat lunak yang digunakan oleh pelanggan *Cloud* dengan model pembayaran sesuai pengguna. Ini dihosting oleh penyedia layanan dan tersedia untuk pelanggannya kapan saja melalui internet. Ini adalah

arsitektur *multitenant* yang berarti ribuan pelanggan dapat mengaksesnya secara bersamaan. Sebagai contoh, Gmail adalah SaaS terbaik di mana pengguna hanya membutuhkan browser dan internet untuk mengakses aplikasi tersebut, Mendukung banyak pengguna secara bersamaan

### **Model 2 — PaaS (Platform as a Service)**

*Platform as a Service* adalah model yang menawarkan peluncuran aplikasi dengan mengurangi biaya pembelian dan pemeliharaan perangkat keras dan perangkat lunak. Ini digunakan oleh pengembang untuk mengembangkan aplikasi baru. Layanan PaaS mencakup desain aplikasi, pengembangan, pengujian, peluncuran, dan hosting. Sebagai contoh, Google App Engine, merupakan layanan dari Google yang menawarkan *client* untuk menjalankan aplikasi mereka di infrastruktur Google.

### **Model 3 — IaaS (Infrastructure as a Service)**

*Infrastructure as a Service* adalah arsitektur yang menyediakan infrastruktur melalui internet. Ini memungkinkan aksesibilitas sistem operasi dan perangkat penyimpanan dengan basis pembayaran sesuai penggunaan. Sebagai contoh, Amazon Elastic Compute Cloud (EC2)

Berdasarkan NIST terdapat 4 (Tiga) model implementasi yang disediakan oleh *Cloud Computing*, yaitu :

#### **Model 1 — Public Cloud**

Infrastruktur *Cloud* seperti penyimpanan, aplikasi, dan layanan lainnya tersedia untuk semua orang dan hanya membayar untuk durasi waktu mereka ketika menggunakan layanan tersebut. Namun, *public cloud* kurang aman karena semua aplikasi dan data tersedia untuk semua pengguna.

#### **Model 2 — Private Cloud**

Infrastruktur cloud ini diimplementasikan dalam sebuah organisasi tunggal dan hanya tersedia untuk pengguna terbatas yang merupakan bagian dari organisasi tersebut. Sumber daya dan aplikasinya dikendalikan oleh organisasi itu sendiri. Oleh karena itu, keamanan pada *Private Cloud* jauh lebih baik.

#### **Model 3 — Hybrid Cloud**

Model implementasi ini adalah komposisi lebih dari satu cloud di mana data penting akan disimpan pada *Private Cloud* dan data yang tidak terlalu penting akan disimpan pada *Public Cloud*.

#### **Model 4 — Community Cloud**

*Community Cloud* melibatkan distribusi infrastruktur komputasi di antara organisasi-organisasi dalam komunitas yang sama. Infrastruktur dan sumber daya komputasi dari sebuah *cloud* eksklusif untuk dua atau lebih organisasi yang memiliki privasi, keamanan, dan regulasi yang sama. Hal ini dapat membantu mengurangi biaya modal untuk pengaturannya karena biaya didistribusikan di antara organisasi-organisasi tersebut.

### **2.8. Amazon Web Service**

Amazon Web Service atau biasa disingkat AWS, adalah penyedia layanan *cloud* yang paling komprehensif dan dipakai oleh banyak pengguna di seluruh penjuru dunia. AWS menawarkan lebih dari 200 layanan yang dapat gunakan sesuai dengan kebutuhan pengguna. Terdapat jutaan pengguna AWS termasuk *startup* yang sedang tumbuh menjadi besar, perusahaan besar, dan lembaga pemerintah. Mereka menggunakan AWS untuk menurunkan biaya, menjadi lebih fleksibel, dan dapat berinovasi lebih cepat.

Layanan yang disediakan dan banyak digunakan oleh pengguna AWS adalah Elastic Compute Cloud (EC2). EC2 adalah layanan yang memperbolehkan para pengguna untuk dapat menyewa *virtual computer* yang digunakan untuk menjalankan aplikasi komputer yang mereka butuhkan.

### **2.9. Amazon Elastic Compute Cloud (EC2)**

EC2 merupakan salah satu layanan yang ada pada AWS. EC2 menawarkan platform komputasi terluas, dengan lebih dari 750 jenis *virtual computer* dan pilihan prosesor terbaru, penyimpanan, jaringan, sistem operasi, serta model pembelian untuk membantu pengguna mencocokkan kebutuhan sesuai dengan beban kerja pengguna.

Pengguna layanan EC2 dapat memilih *virtual computer* yang mereka butuhkan dan hanya membayar sebanyak apa yang pengguna pakai. EC2 juga memiliki fitur skalabilitas, dimana EC2 dapat menyesuaikan kebutuhan sesuai dengan beban kerja pengguna, lalu dapat memperbesar dan memperkecil kinerja EC2 secara otomatis sesuai dengan keadaan disaat itu.

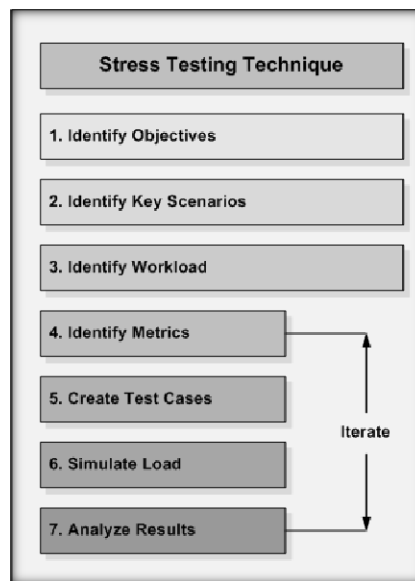


### 2.10. Stress Testing

Pengujian atau *testing* adalah proses menjalankan suatu aplikasi untuk menemukan masalah atau kesalahan. Pengujian kinerja atau *performance testing* dilakukan dengan mengirim permintaan dalam jumlah besar, seperti mengakses sistem dengan banyak pengguna secara bersamaan, untuk mengevaluasi respons dan kinerja aplikasi tersebut di bawah beban yang berat. Tujuan pengujian kinerja untuk mengetahui kemampuan sebuah sistem dalam menangani banyaknya permintaan.

*Stress testing* adalah komponen dari pengujian kinerja yang mengevaluasi ketahanan, ketersediaan, dan kemampuan aplikasi dalam menghadapi kondisi beban kerja yang tinggi.

Berdasarkan petunjuk dari *Performance Testing Guidance for Web Application* (2007) yang dibuat oleh Microsoft berikut adalah prosedur stress testing pada gambar 2.3 dibawah



**Gambar 2.3** Prosedur *stress testing*  
(Microsoft, 2007)

**Langkah 1 — Identifikasi test objectives.**

Menanyakan pada diri sendiri atau orang lain pertanyaan yang dapat membantu mengidentifikasi hasil dari Stress testing. Contoh pertanyaan yang bisa diajukan diantaranya :

1. Bagaimana hasil pengujian terhadap web server Apache, Nginx, dan Litespeed ketika menghadapi beban permintaan yang besar?

### **Langkah 2 — Identifikasi key scenario.**

Untuk mendapatkan hasil maksimal dari stress test, pengujian perlu focus pada perilaku skenario pengguna atau skenario mana yang paling penting bagi keberhasilan aplikasi secara keseluruhan. Untuk mengidentifikasi skenario ini, biasanya dimulai dengan menentukan satu skenario yang ingin anda uji untuk mengidentifikasi potensi masalah kinerja yaitu sebagai berikut :

1. Cobalah untuk menjalankan skenario yang dapat mempengaruhi performa suatu aplikasi, seperti skenario seperti *intensive* HTTP request pada web server.

### **Langkah 3 — Identifikasi workload.**

Beban yang anda terapkan pada skenario tertentu harus memberikan tekanan yang cukup pada sistem samapi melampaui ambang batas sehingga memungkinkan kita mengamati kondisi stres pada sistem tersebut. Salah satu metode untuk menentukan beban untuk suatu sistem ialah dengan meningkatkan beban secara bertahap dan mengamati perilaku aplikasi dalam berbagai kondisi beban. Kuncinya adalah melakukan pengujian secara sistematis dengan berbagai beban kerja hingga menciptakan kegagalan yang signifikan

### **Langkah 4 — Identifikasi metrik.**

Ketika diidentifikasi dan diamati dengan benar, metrik memberikan informasi tentang seberapa baik atau buruk kinerja sistem anda dengan tujuan kinerja anda. Selain itu, metrik dapat membantu anda mengidentifikasi area masalah dan hambatan aplikasi anda.

### **Langkah 5 — Membuat test cases.**

Mengidentifikasi profil beban kerja dan skenario utama umumnya tidak memberikan semua informasi yang diperlukan. Inputan tambahan untuk merancang *stress test* mencakup tujuan kinerja, karakteristik beban kerja, data pengujian, lingkungan pengujian, dan metrik yang teridentifikasi. Setiap desain pengujian harus menyebutkan hasil yang diharapkan dan data yang ingin dikumpulkan, dengan demikian setiap kasus pengujian dapat ditandai dengan “lulus”, “gagal”, atau “tidak meyakinkan” setelah pelaksanaan.

### **Langkah 6 — Simulasikan load.**

Setelah menyelesaikan langkah-langkah sebelumnya hingga tingkat yang sesuai, anda harus siap mensimulasikan beban yang ingin dijalankan. Berikut adalah langkah-langkah *Test execution* :

1. Pastikan bahwa pengujian dan lingkungan pengujian dikonfigurasi untuk pengumpulan metrik.
2. Pastikan bahwa pengujian dan lingkungan pengujian terkonfigurasi untuk pengumpulan metrik.
3. Mulai test dengan mereset sistem.

### **Langkah 7 — Analisa hasil.**

Analisa hasil data yang didapatkan dan bandingkan hasilnya dengan tingkat metrik yang diterima. Jika hasilnya menunjukkan bahwa tingkat kinerja anda perlukan belum tercapai, analisis dan pebaiki penyebab *bottleneck* tersebut.

## **2.11. Beban Kerja**

Pada dasarnya beban kerja adalah program atau aplikasi yang berjalan di komputer. Beban kerja bisa berupa aplikasi sederhana seperti jam alarm atau aplikasi kalkulator pada komputer, atau bisa juga pada skala yang lebih besar seperti aplikasi perusahaan yang kompleks yang *di-hosting* di satu server atau lebih, dengan ribuan *client* atau pengguna yang terhubung dan berinteraksi dengan server aplikasi di seluruh jaringan yang luas.

Beban kerja juga bisa merujuk pada jumlah pekerjaan yang ditangani oleh aplikasi dengan sumber daya komputasi yang ada. Beban kerja aplikasi dapat

dihubungkan dengan jumlah, waktu, dan sumber daya komputasi yang dibutuhkan untuk menyelesaikan pekerjaan yang spesifik atau menghasilkan keluaran dari masukan yang diberikan.

Beban kerja ringan menyelesaikan jumlah tugas yang relatif sedikit atau hanya menggunakan daya komputasi dan sumber daya komputasi yang sedikit, seperti *processor*, *clock cycle*, *storage*, *input/output*, dan sebagainya. Sebaliknya, beban kerja berat membutuhkan jumlah sumber daya komputasi yang jauh lebih besar.

Web server sendiri juga memiliki beban kerja yang terbatas dan bergantung pada beberapa faktor, seperti:

- Jenis HTTP *request*
- Jenis konten web: Statis atau dinamis
- Konten web tersebut tersimpan dalam *cache* atau tidak
- *Hardware* yang menjalankan web server

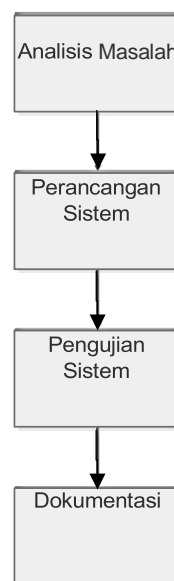
Pada Petunjuk *Performance Testing Guidance for Web Application* yang dibuat oleh Microsoft, untuk mengetahui limitasi beban kerja pada web server, dilakukan *request* terus menerus hingga web server tidak dapat lagi menerima HTTP *request* dan hanya menampilkan halaman *error*. Jumlah dari batas *request* pada web server akan digunakan untuk membuat beban kerja ekstrim pada web server dengan melakukan *request* pada jumlah maksimal untuk web server tersebut dan dilakukan *request* pada jumlah waktu yang berbeda.

## BAB III

### ANALISIS DAN PERANCANGAN SISTEM

#### 3.1. Tahapan Penelitian

Untuk memastikan hasil penelitian sesuai dengan tujuan awal, langkah-langkah yang tepat perlu dilakukan selama proses pengumpulan dan analisis data, berikut adalah gambar skema dari prosedur penelitian yang akan penulis ikuti :



**Gambar 3.1.** Tahapan Penelitian

#### 3.2. Analisis Masalah

Tahapan ini merupakan sebuah tahap dalam melakukan apa saja yang diperlukan oleh sistem yang akan dibangun dan menganalisis apakah sebuah *web server* dapat bekerja dan dapat diteliti sesuai dengan yang penulis harapkan dan dapat memenuhi kebutuhan pengguna.

Dalam penelitian ini, ada beberapa permasalahan terkait layanan *web server*:

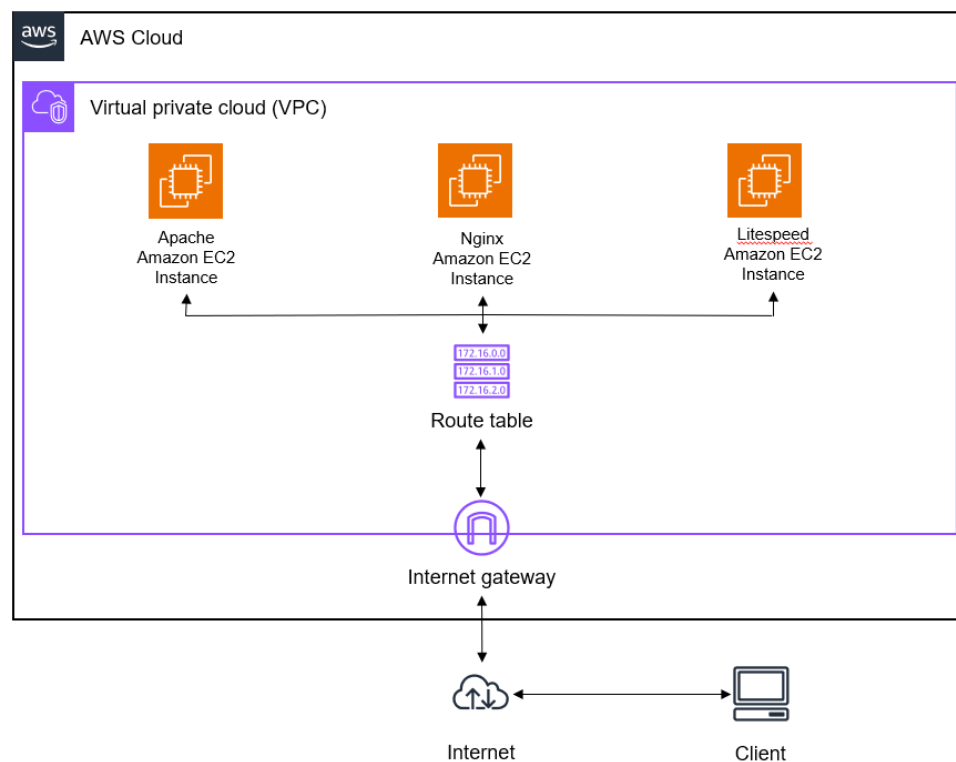
1. Sedikitnya informasi mengenai *web server* yang dapat menahan beban permintaan yang tinggi serta banyaknya jumlah user yang ingin mengunjungi *web server* tersebut.

2. Terdapat kelemahan pada kinerja *web server* yang akhirnya dilakukan *stress test* pada kinerja *web server* Nginx, Apache, dan Litespeed menggunakan Apache JMeter untuk mendapatkan perbandingan pada kinerja ketiga *web server* tersebut.

Dari kedua permasalahan tersebut, akan dilakukan pengujian pada *web server* Litespeed, Apache, dan Nginx dengan parameter yang akan diuji yaitu *response time*, *throughput*, *Sent*, *Received*, dan *Error*.

### 3.3. Perancangan Sistem

Pengujian akan melibatkan penggunaan layanan Amazon *Elastic Compute Cloud* (EC2) serta satu laptop klien dengan sistem operasi Windows 10 yang dilengkapi dengan aplikasi Apache JMeter. Konfigurasi topologi yang akan digunakan selama pengujian dapat dilihat pada Gambar 3.2.



**Gambar 3.2** Rancangan Sistem

Sebelum melakukan proses *stress testing*, dilakukan pembuatan 3 (tiga) *virtual computer* menggunakan layanan Amazon *Elastic Compute Cloud* (EC2) yang akan berada pada jaringan Amazon Web Service (AWS) sendiri yang disebut sebagai *Virtual Private Cloud* (VPC) dan ini dilakukan secara otomatis oleh layanan EC2. *Virtual*

*Computer* yang diluncurkan oleh EC2 disebut sebagai *instance*. Pada jaringan VPC, ketiga *instance* akan dihubungkan ke *route table* atau kurang lebih seperti router yang akan menghubungkan ketiga *instance* ke internet. Dengan begitu *instance* dapat mengakses internet dan kita juga dapat mengakses *instance* tersebut. Masing-masing *instance* akan diinstal dengan *web server* Apache, Nginx, dan Litespeed. Dengan begitu kita dapat melakukan proses *stress testing* dengan aplikasi JMeter.

Adapun Alat dan bahan yang digunakan, Skema Pengujian pada sistem, serta Parameter yang diuji pada penelitian ini sebagai berikut.

### 3.3.1. Alat dan Bahan Pengujian

Berikut adalah rincian mengenai spesifikasi perangkat keras dan perangkat lunak yang digunakan untuk melakukan pengujian :

#### 1. Perangkat Keras (*Hardware*)

Berikut adalah spesifikasi perangkat keras yang diperlukan untuk server atau komputer yang akan digunakan dalam penelitian ini:

- *Virtual Computer Server (AWS Instance t2.micro)*
  - a. *vCpu 1*
  - b. *Memori 1 Gib*
  - c. *Bandwith 5Gbps*
  - d. *Storage 8Gb*

#### 2. Perangkat Lunak (*Software*)

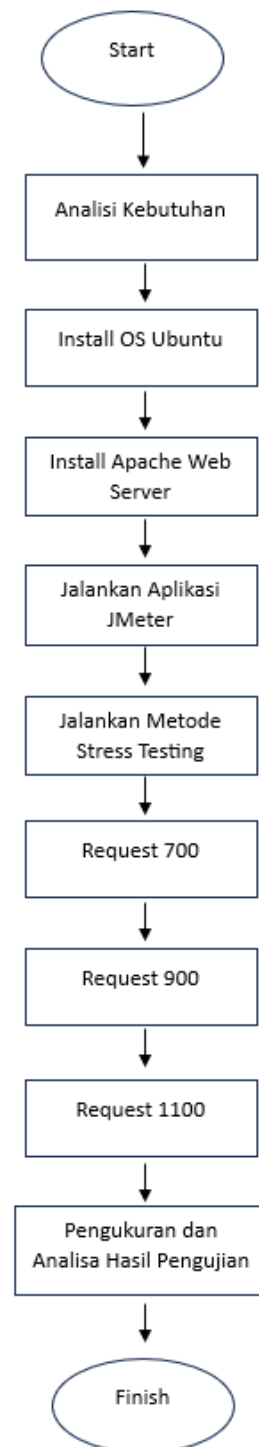
Perangkat lunak yang dipakai dalam membuat pengujian pada penelitian ini adalah :

- Ubuntu Server Versi 22.04
- Windows 10 64bit
- Apache JMeter.
- *Web Server* Nginx
- *Web Server* Apache
- *Web Server* LiteSpeed

### 3.3.2. Skema Pengujian Sistem

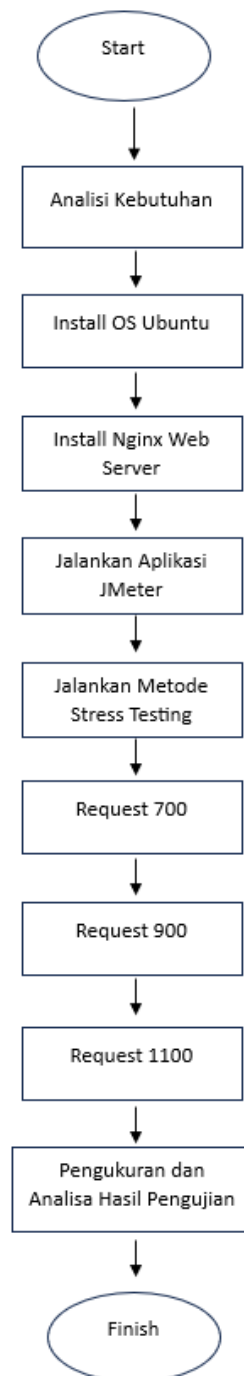
Untuk menganalisa perbandingan pada kinerja *web server* Apache, Nginx, dan Litespeed akan dilakukan dengan metode *stress testing* dan memakai JMeter. Terdapat beberapa tahapan pada pengujian ini yang akan dijadikan prosedur penelitian. Akan dilakukan pengukuran dengan mengambil nilai median dari tiap pengujian pada web server menggunakan halaman statis. Lalu beban yang diberikan pada tiap web server adalah 700, 900, dan 1100 permintaan berdasarkan penelitian Chandra., (2019), tahapan proses ada pada **Gambar 3.3** sampai **Gambar 3.5** dibawah ini.





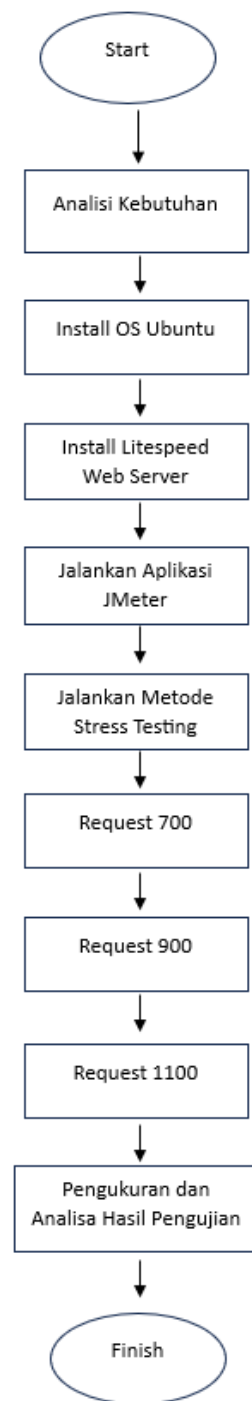
**Gambar 3.3.** Tahapan Percobaan Pada Apache *Web Server*

Pada tahapan Gambar 3.3. dilakukan instalasi sistem operasi ubuntu dan instalasi *web server* Apache. Setelah instalasi selesai, dilakukan proses pengujian dengan menggunakan aplikasi JMeter dan menjalankan proses *stress testing* dengan melakukan *request* HTTP untuk 700, 900, 1100 *users*. Lalu dilakukan pengukuran dan perbandingan dengan *web server* lainnya.



**Gambar 3.4.** Tahapan Percobaan Pada Nginx *Web Server*

Pada tahapan Gambar 3.4. dilakukan instalasi sistem operasi ubuntu dan instalasi *web server* Nginx. Setelah instalasi selesai, dilakukan proses pengujian dengan menggunakan aplikasi JMeter dan menjalankan proses *stress testing* dengan melakukan *request* HTTP untuk 700, 900, 1100 *users*. Lalu dilakukan pengukuran dan perbandingan dengan *web server* lainnya.



**Gambar 3.5** Tahapan Percobaan Pada Litespeed *Web Server*

Pada tahapan Gambar 3.5. dilakukan instalasi sistem operasi ubuntu pada *instance* dan instalasi *web server* Litespeed. Setelah instalasi selesai, dilakukan proses pengujian dengan menggunakan aplikasi JMeter dan menjalankan proses *stress testing* dengan melakukan *request* HTTP untuk 700, 900, 1100 *users*. Lalu dilakukan pengukuran dan perbandingan dengan web server lainnya.

### 3.3.3. Parameter Kinerja

Parameter kinerja merupakan proses evaluasi yang digunakan untuk mengukur kinerja sesuai dengan sasaran dan tujuan yang telah ditetapkan. Hasil evaluasi tersebut kemudian dibandingkan dengan tujuan yang diinginkan untuk melihat efektivitas kinerja dalam mencapai tujuan tersebut. Parameter yang akan digunakan dalam pengujian ini adalah *response time*, *throughput*, *Error*, *data sent*, dan *data received*. Berikut merupakan penjelasan dari parameter tersebut :

1. *Response Time* adalah interval waktu yang diperlukan untuk menyelesaikan permintaan kepada *web server* dan waktu tanggapan *web server* atas permintaan tersebut. Semakin kecil nilainya semakin responsif *web server* tersebut.
2. *Error* mengacu pada situasi dimana permintaan yang diberikan kepada *web server* gagal memenuhi kriteria respon yang diharapkan. Semakin kecil nilainya baik kinerja *web server* tersebut.
3. *Throughput* adalah ukuran seberapa banyak data atau pekerjaan yang diproses dalam jangka waktu tertentu. Pada penelitian ini *throughput* merujuk pada jumlah permintaan yang dapat *web server* tangani per detik.
4. *Data Sent* mengacu pada jumlah total data yang dikirimkan klien ke server selama pengujian kinerja.
5. *Data Received* mengacu pada jumlah total data yang dikirimkan server ke klien selama pengujian kinerja.

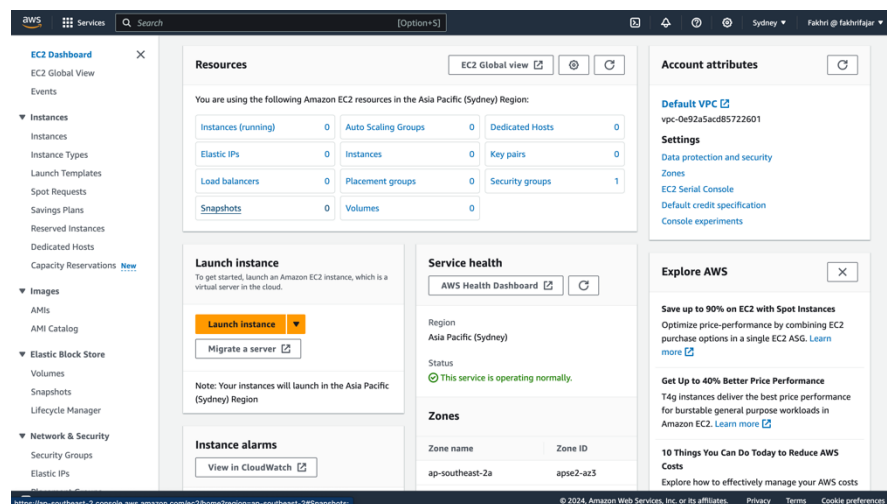
## BAB IV

### IMPLEMENTASI DAN PENGUJIAN

Langkah awal sebelum melakukan pengujian, peneliti menjalankan 3 (tiga) *instance* pada layanan Amazon *Elastic Compute Cloud* (EC2) terlebih dahulu, dan mengkonfigurasinya serta menginstall *web server* di masing-masing *instance*.

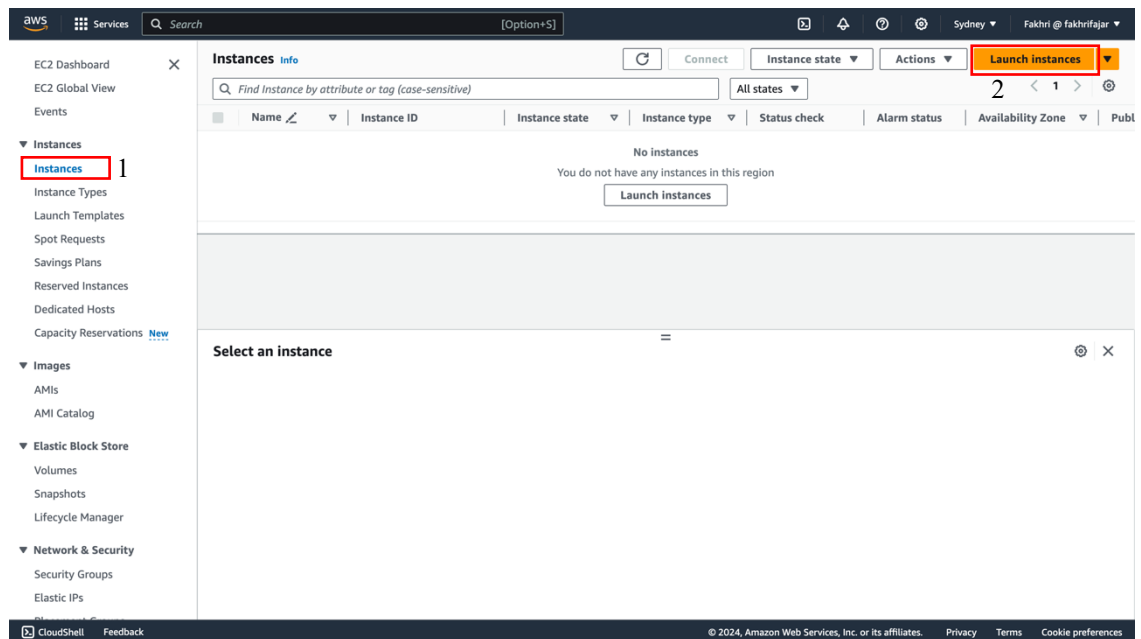
#### 4.1. Launch Instance

Untuk meluncurkan sebuah *instance* pada Amazon *Elastic Computer Cloud* (EC2), peneliti sudah mempersiapkan akun AWS yang sudah dapat dipakai, dan kemudian menuju ke Menu EC2 Dashboard, yang dapat dilihat pada Gambar 4.1.



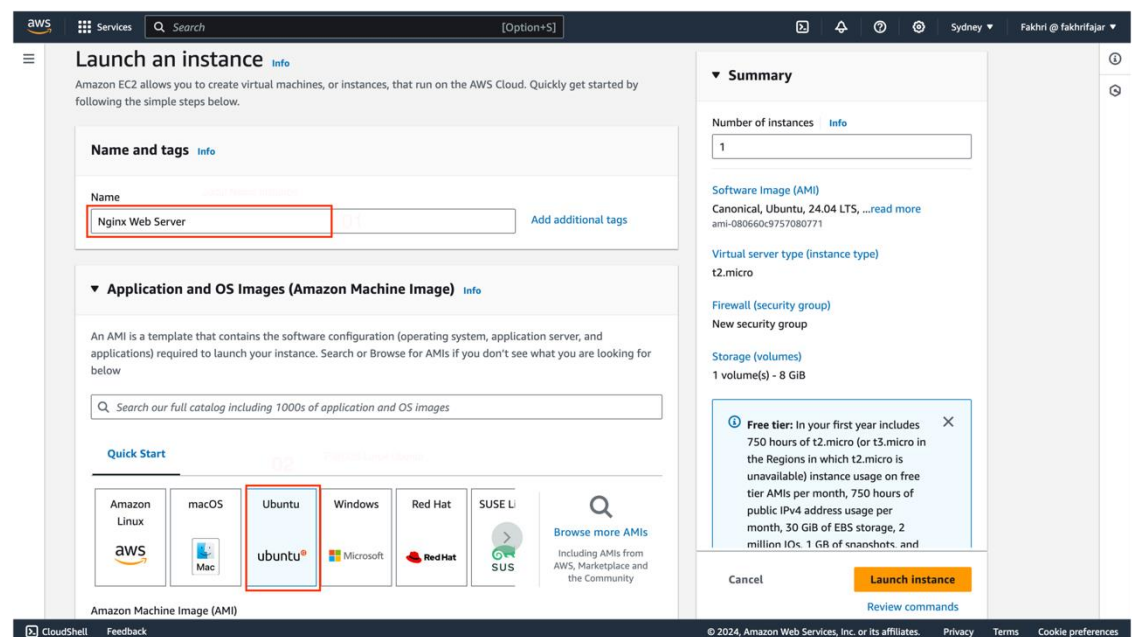
**Gambar 4.1. EC2 Dashboard**

Langkah selanjutnya adalah memilih menu *Launch Instance* untuk membuat *instance* baru, yang dapat dilihat pada Gambar 4.2.

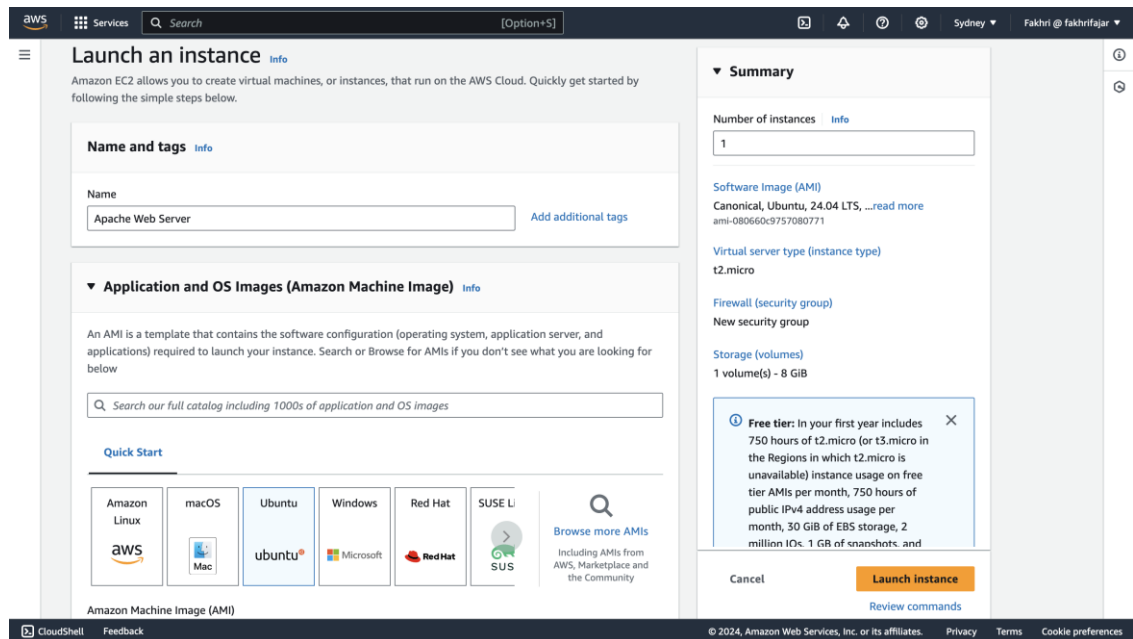


**Gambar 4.2.** Launch Instance AWS EC2

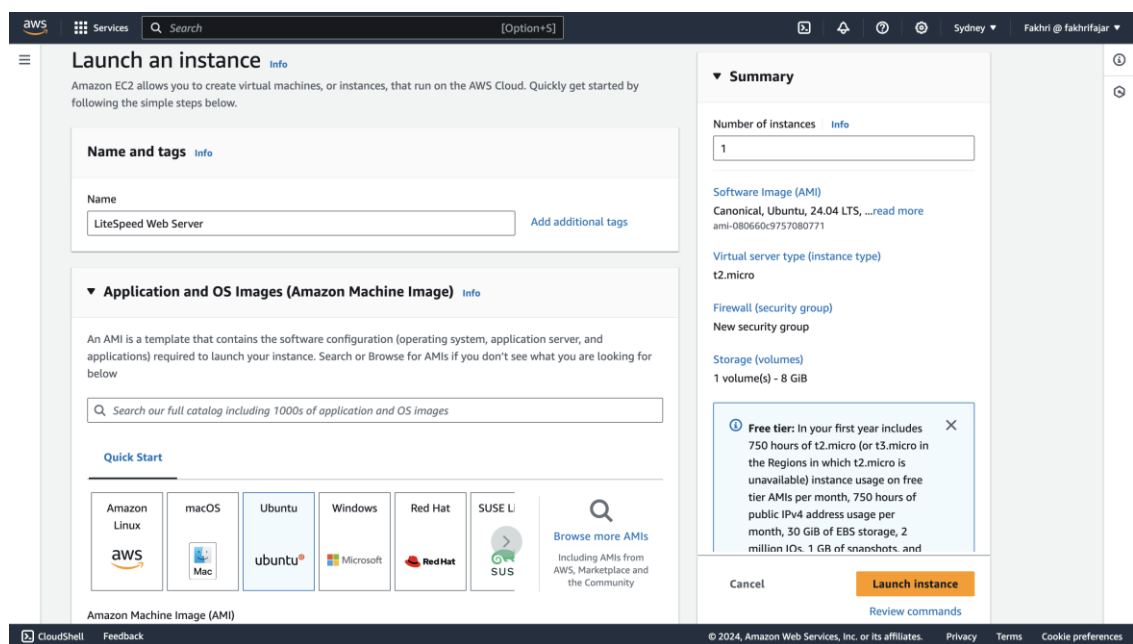
Langkah Selanjutnya adalah melengkapi beberapa *form* yang dibutuhkan untuk membuat *instance* pada AWS EC2, Langkah-langkahnya dapat dilihat mulai dari Gambar 4.3.



**Gambar 4.3.** Instance Nginx Web Server

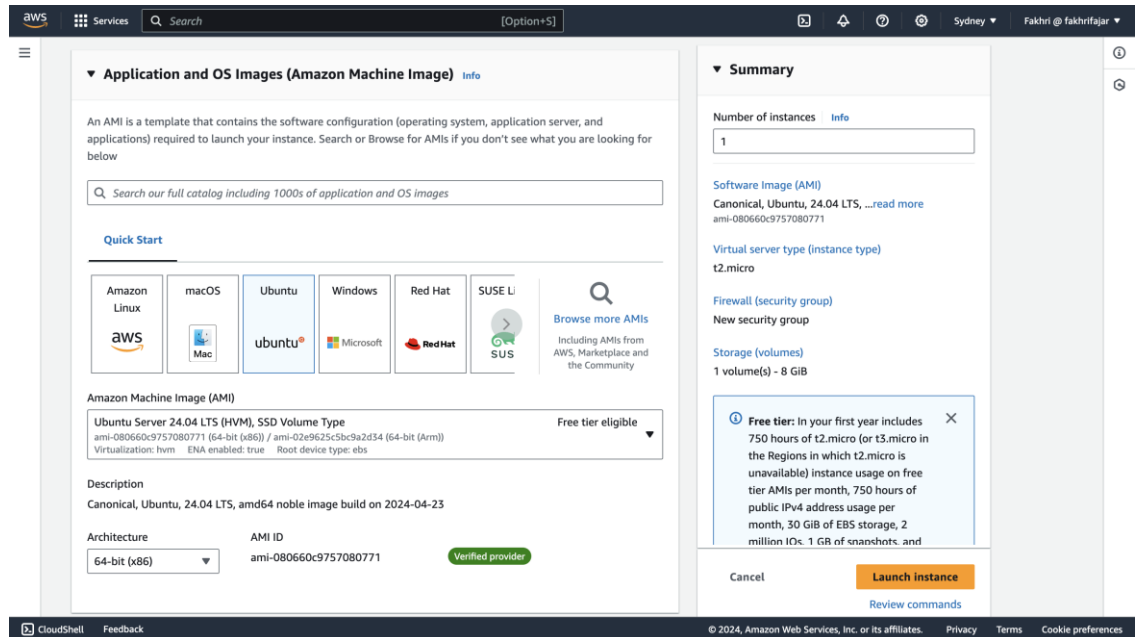


Gambar 4.4. Instance Apache Web Server



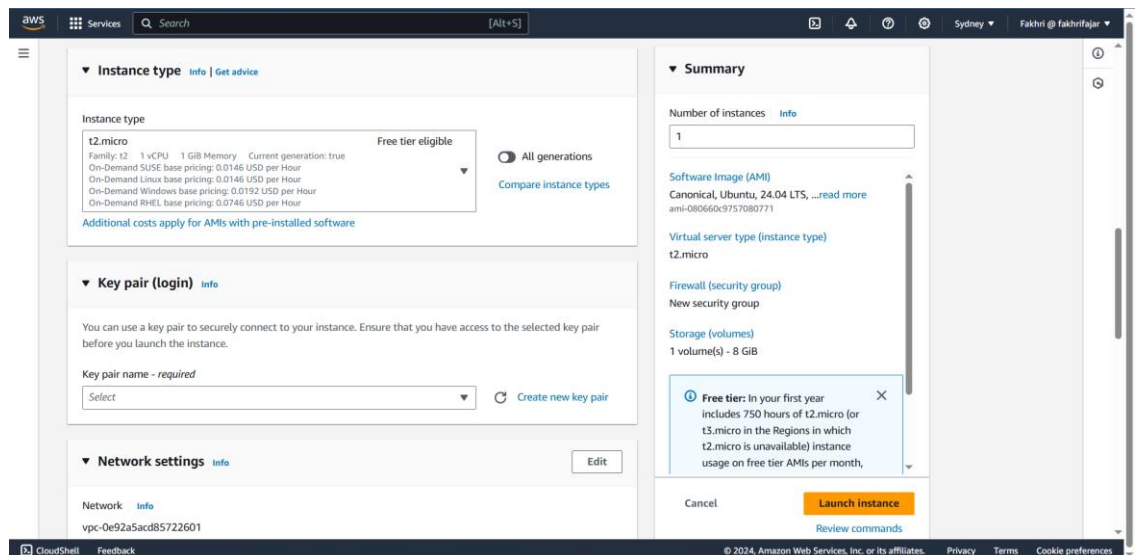
Gambar 4.5. Instance LiteSpeed Web Server

Pada Gambar di atas, untuk Kolom Name, silahkan di isi nama dari *instance* yang akan dibuat, peneliti membuat dengan nama Nginx Web Server untuk instance pertama, Apache Web Server untuk instance kedua dan LiteSpeed Web Server untuk instance ketiga.



**Gambar 4.6.** Sistem Operasi Web Server

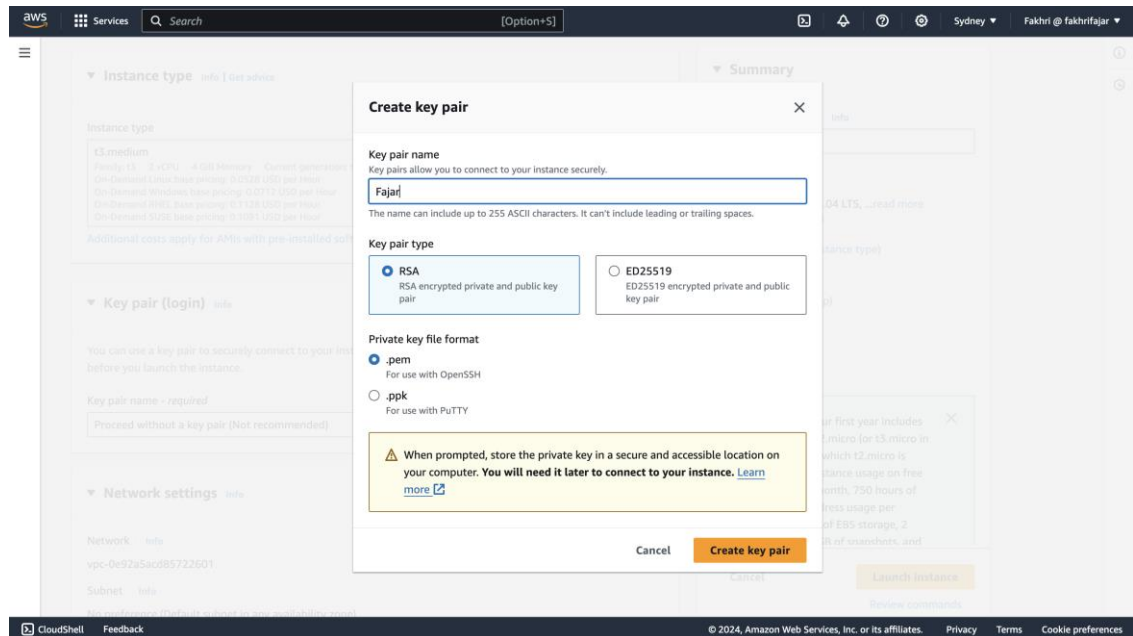
Selanjutnya pada Gambar 4.6. untuk Sistem Operasi (OS) linux yang akan dipakai adalah Ubuntu Server 24.02 LTS dengan arsitektur 64-bit. Setelah memilih sistem operasi, form selanjutnya yang harus di pilih adalah jenis instance yang akan di luncurkan pada AWS EC2, untuk penelitian ini kita akan memilih spesifikasi servernya adalah t2.micro dengan spesifikasi 1vCPU dan 1Gib Memory, untuk detailnya ada pada Gambar 4.7.



**Gambar 4.7.** Jenis Instance AWS EC2

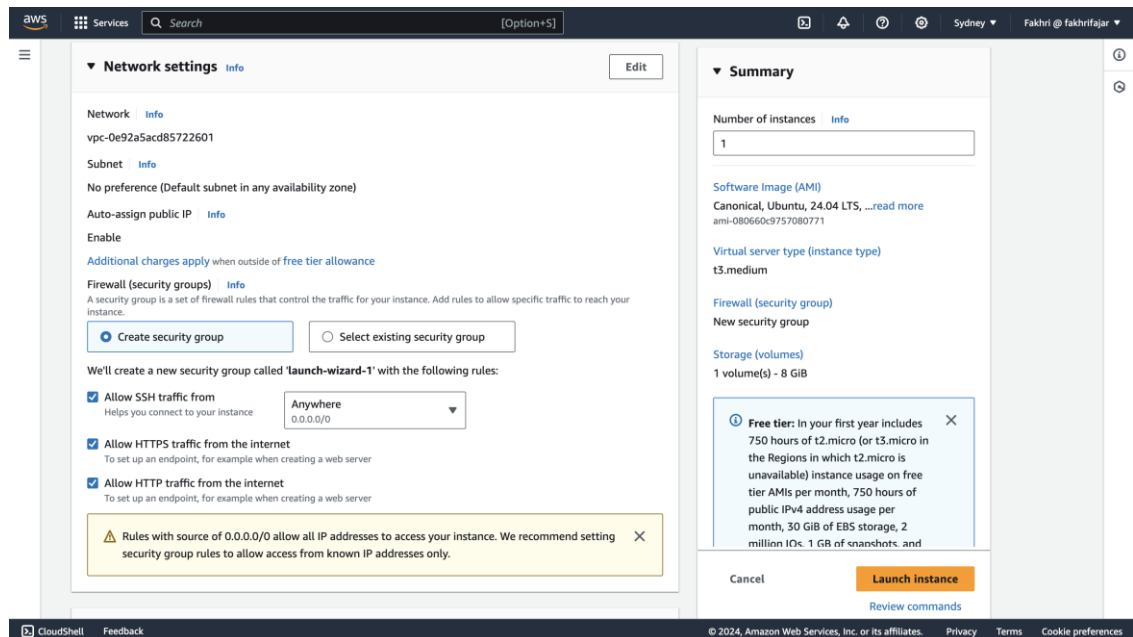


Langkah selanjutnya adalah membuat *key pair (login)*, ini berguna sebagai akses untuk dapat masuk ke dalam terminal server melalui akses SSH di port 22. Jika belum memiliki key pair, maka harus di generate terlebih dahulu, dengan memilih menu *Create New Key Pair*, lalu untuk kolom *key pair name*, isikan dengan nama *key pair*, untuk *key pair type* pilih RSA dan untuk *private key file format* pilih .pem, kemudian klik tombol *Create Key Pair*, untuk detailnya ada pada Gambar 4.8.



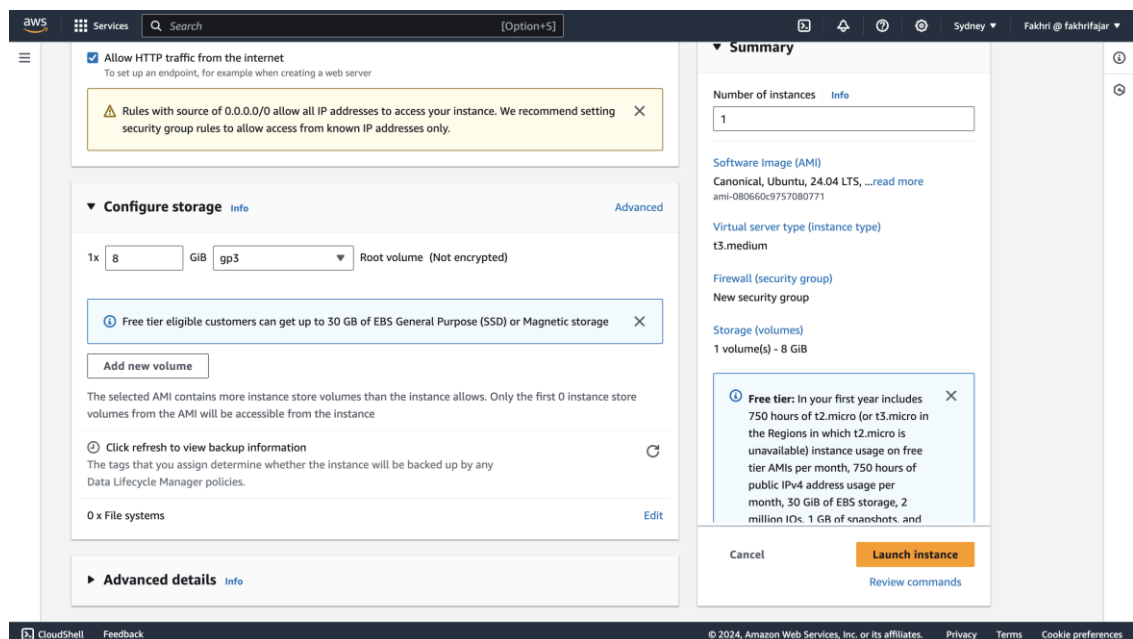
**Gambar 4.8.** Generate new Key Pair

Langkah selanjutnya adalah mengkonfigurasi jaringan dan *firewall* untuk ketiga *instance* yang akan di uji coba, pada bagian *firewall*, direkomendasikan untuk mencentang *Allow SSH Traffic*, *Allow HTTPS Traffic From Internet* dan *Allow HTTP Traffic From Internet* yang memiliki arti, *firewall* akan mengizinkan client atau pengguna dapat mengakses server di port 22 untuk SSH, 80 untuk HTTP dan port 443 untuk HTTPS, untuk detailnya ada pada Gambar 4.9.



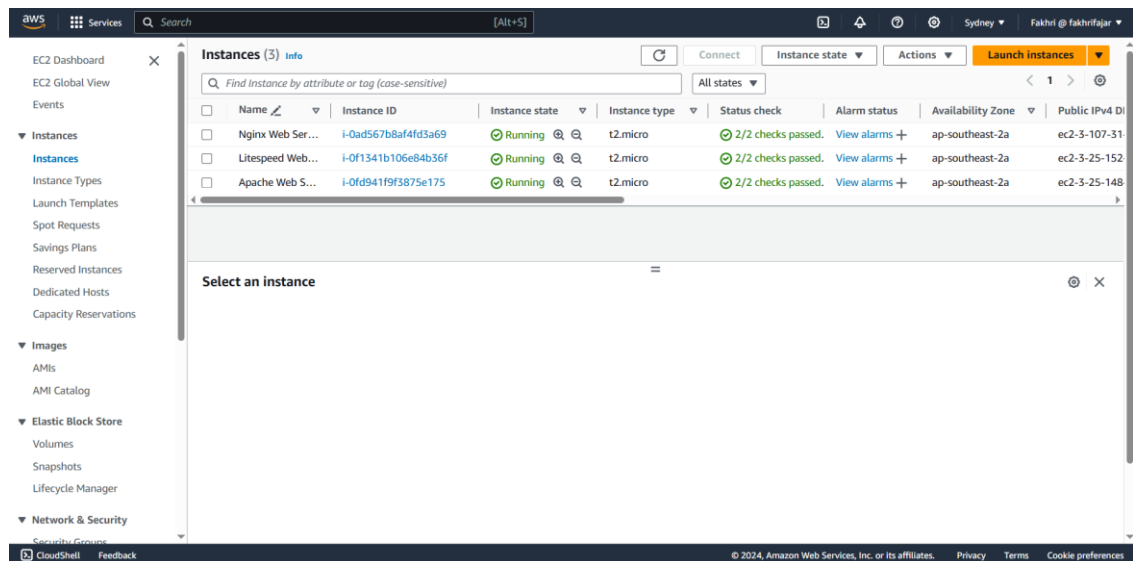
**Gambar 4.9.** Pengaturan Network & Firewall

Setelah itu, Langkah terakhir untuk membuat *instance* di AWS EC2 adalah memilih besaran penyimpanan untuk ketiga web server yang akan diuji coba. Untuk penyimpanannya peneliti membuat sebesar 8Gib saja, karena tidak banyak kebutuhan untuk penelitian ini. Setelah memilih ukuran penyimpanan *instance*, klik *Launch Instance* untuk membuat *instance* yang kita butuhkan, untuk detailnya ada pada Gambar 4.10.



**Gambar 4.10.** Pengaturan Penyimpanan Server

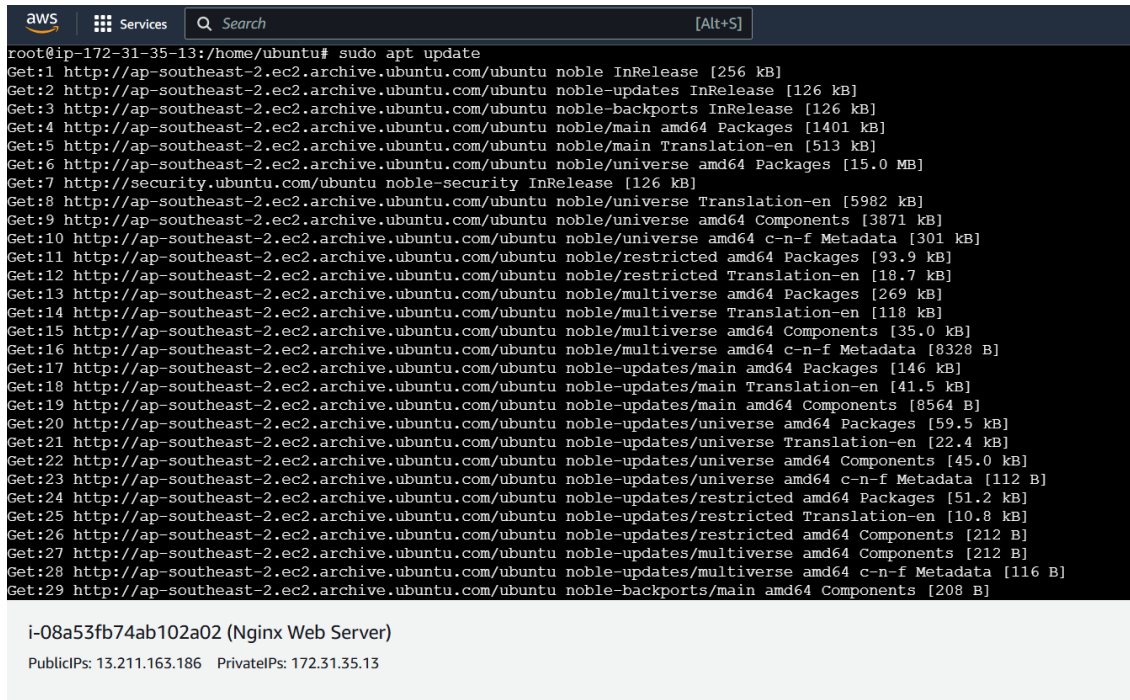
Setelah itu tunggu hingga proses *launch instance* berhasil dan status *instance* pada kondisi *running*. Detail dapat dilihat pada Gambar 4.11.



**Gambar 4.11.** Status Launch Instance

## 4.2. Pemasangan Web Server Nginx

Setelah sudah meluncurkan instance pada AWS EC2, Langkah selanjutnya adalah konfigurasi masing-masing server, untuk tahap pertama, pada instance Nginx Web Server, penulis akan melakukan instalasi web server nginx. Dalam melakukan instalasi ini, penulis menggunakan aplikasi PuTTY atau menggunakan *web console* yang sudah disediakan langsung oleh AWS untuk melakukan *remote web server*. Pada Gambar 4.12. di bawah adalah *remote server* pada *instance* dan melakukan update pada sistem server.



```

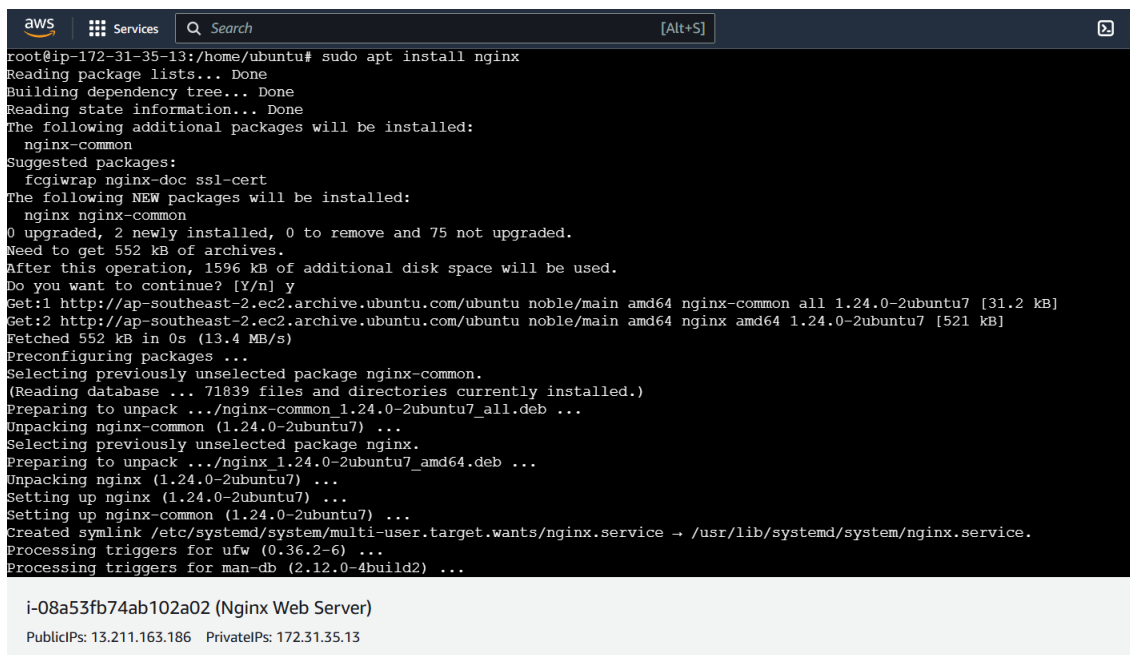
aws Services Search [Alt+S]
root@ip-172-31-35-13:/home/ubuntu# sudo apt update
Get:1 http://ap-southeast-2.ec2.archive.ubuntu.com/ubuntu noble InRelease [256 kB]
Get:2 http://ap-southeast-2.ec2.archive.ubuntu.com/ubuntu noble-updates InRelease [126 kB]
Get:3 http://ap-southeast-2.ec2.archive.ubuntu.com/ubuntu noble-backports InRelease [126 kB]
Get:4 http://ap-southeast-2.ec2.archive.ubuntu.com/ubuntu noble/main amd64 Packages [1401 kB]
Get:5 http://ap-southeast-2.ec2.archive.ubuntu.com/ubuntu noble/main Translation-en [513 kB]
Get:6 http://ap-southeast-2.ec2.archive.ubuntu.com/ubuntu noble/universe amd64 Packages [15.0 MB]
Get:7 http://security.ubuntu.com/ubuntu noble-security InRelease [126 kB]
Get:8 http://ap-southeast-2.ec2.archive.ubuntu.com/ubuntu noble/universe Translation-en [5982 kB]
Get:9 http://ap-southeast-2.ec2.archive.ubuntu.com/ubuntu noble/universe amd64 Components [3871 kB]
Get:10 http://ap-southeast-2.ec2.archive.ubuntu.com/ubuntu noble/universe amd64 c-n-f Metadata [301 kB]
Get:11 http://ap-southeast-2.ec2.archive.ubuntu.com/ubuntu noble/restricted amd64 Packages [93.9 kB]
Get:12 http://ap-southeast-2.ec2.archive.ubuntu.com/ubuntu noble/restricted Translation-en [18.7 kB]
Get:13 http://ap-southeast-2.ec2.archive.ubuntu.com/ubuntu noble/multiverse amd64 Packages [269 kB]
Get:14 http://ap-southeast-2.ec2.archive.ubuntu.com/ubuntu noble/multiverse Translation-en [118 kB]
Get:15 http://ap-southeast-2.ec2.archive.ubuntu.com/ubuntu noble/multiverse amd64 Components [35.0 kB]
Get:16 http://ap-southeast-2.ec2.archive.ubuntu.com/ubuntu noble/multiverse amd64 c-n-f Metadata [8328 B]
Get:17 http://ap-southeast-2.ec2.archive.ubuntu.com/ubuntu noble-updates/main amd64 Packages [146 kB]
Get:18 http://ap-southeast-2.ec2.archive.ubuntu.com/ubuntu noble-updates/main Translation-en [41.5 kB]
Get:19 http://ap-southeast-2.ec2.archive.ubuntu.com/ubuntu noble-updates/main amd64 Components [8564 B]
Get:20 http://ap-southeast-2.ec2.archive.ubuntu.com/ubuntu noble-updates/universe amd64 Packages [59.5 kB]
Get:21 http://ap-southeast-2.ec2.archive.ubuntu.com/ubuntu noble-updates/universe Translation-en [22.4 kB]
Get:22 http://ap-southeast-2.ec2.archive.ubuntu.com/ubuntu noble-updates/universe amd64 Components [45.0 kB]
Get:23 http://ap-southeast-2.ec2.archive.ubuntu.com/ubuntu noble-updates/universe amd64 c-n-f Metadata [112 B]
Get:24 http://ap-southeast-2.ec2.archive.ubuntu.com/ubuntu noble-updates/restricted amd64 Packages [51.2 kB]
Get:25 http://ap-southeast-2.ec2.archive.ubuntu.com/ubuntu noble-updates/restricted Translation-en [10.8 kB]
Get:26 http://ap-southeast-2.ec2.archive.ubuntu.com/ubuntu noble-updates/restricted amd64 Components [212 B]
Get:27 http://ap-southeast-2.ec2.archive.ubuntu.com/ubuntu noble-updates/multiverse amd64 Components [212 B]
Get:28 http://ap-southeast-2.ec2.archive.ubuntu.com/ubuntu noble-updates/multiverse amd64 c-n-f Metadata [116 B]
Get:29 http://ap-southeast-2.ec2.archive.ubuntu.com/ubuntu noble-backports/main amd64 Components [208 B]

i-08a53fb74ab102a02 (Nginx Web Server)
PublicIPs: 13.211.163.186 PrivateIPs: 172.31.35.13

```

**Gambar 4.12.** Remote Server dengan AWS Console

Ketika proses pembaruan sistem server sudah selesai dengan perintah “*sudo apt update*”, Langkah selanjutnya adalah proses instalasi *web server* Nginx dengan perintah “*sudo apt install nginx*”. Untuk proses instalasi dapat dilihat pada Gambar 4.13.



```

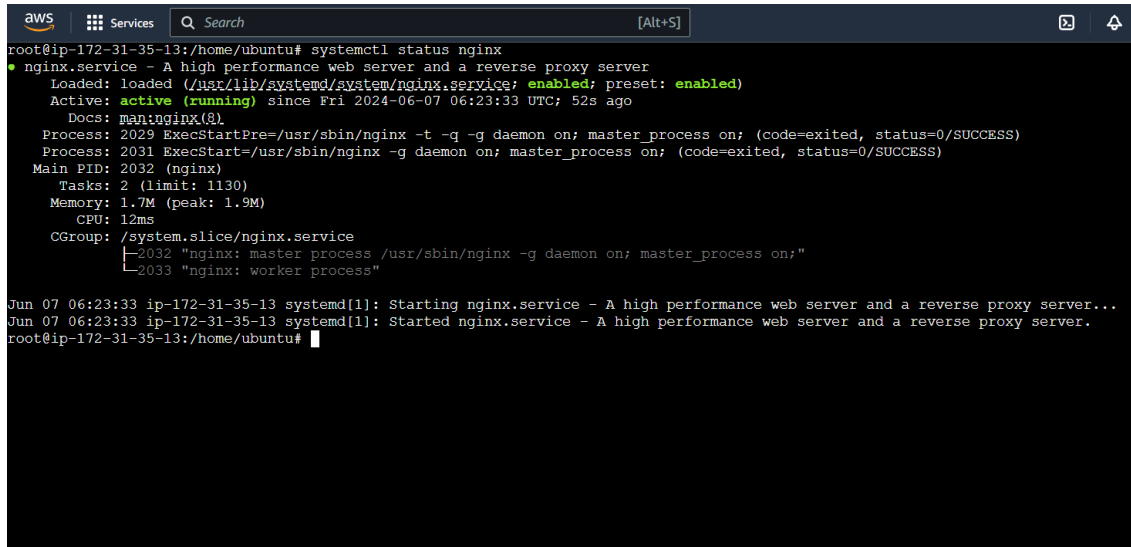
aws Services Search [Alt+S]
root@ip-172-31-35-13:/home/ubuntu# sudo apt install nginx
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following additional packages will be installed:
  nginx-common
Suggested packages:
  fcgiwrap nginx-doc ssl-cert
The following NEW packages will be installed:
  nginx nginx-common
0 upgraded, 2 newly installed, 0 to remove and 75 not upgraded.
Need to get 552 kB of archives.
After this operation, 1596 kB of additional disk space will be used.
Do you want to continue? [Y/n] y
Get:1 http://ap-southeast-2.ec2.archive.ubuntu.com/ubuntu noble/main amd64 nginx-common all 1.24.0-2ubuntu7 [31.2 kB]
Get:2 http://ap-southeast-2.ec2.archive.ubuntu.com/ubuntu noble/main amd64 nginx amd64 1.24.0-2ubuntu7 [521 kB]
Fetched 552 kB in 0s (13.4 MB/s)
Preconfiguring packages ...
Selecting previously unselected package nginx-common.
(Reading database ... 71839 files and directories currently installed.)
Preparing to unpack .../nginx-common_1.24.0-2ubuntu7_all.deb ...
Unpacking nginx-common (1.24.0-2ubuntu7) ...
Selecting previously unselected package nginx.
Preparing to unpack .../nginx_1.24.0-2ubuntu7_amd64.deb ...
Unpacking nginx (1.24.0-2ubuntu7) ...
Setting up nginx (1.24.0-2ubuntu7) ...
Setting up nginx-common (1.24.0-2ubuntu7) ...
Created symlink /etc/systemd/system/multi-user.target.wants/nginx.service → /usr/lib/systemd/system/nginx.service.
Processing triggers for ufw (0.36.2-6) ...
Processing triggers for man-db (2.12.0-4build2) ...

i-08a53fb74ab102a02 (Nginx Web Server)
PublicIPs: 13.211.163.186 PrivateIPs: 172.31.35.13

```

**Gambar 4.13.** Instalasi Nginx Web Server

Setelah instalasi Nginx Web Server, maka tahap selanjutnya adalah memastikan *service* Nginx sudah berjalan pada server AWS EC2, untuk detailnya dapat dilihat pada Gambar 4.14.



```
aws Services Search [Alt+S]
root@ip-172-31-35-13:/home/ubuntu# systemctl status nginx
● nginx.service - A high performance web server and a reverse proxy server
   Loaded: loaded (/usr/lib/systemd/system/nginx.service; enabled; preset: enabled)
   Active: active (running) since Fri 2024-06-07 06:23:33 UTC; 52s ago
     Docs: man:nginx(8)
   Process: 2029 ExecStartPre=/usr/sbin/nginx -t -q -g daemon on; master_process on; (code=exited, status=0/SUCCESS)
   Process: 2031 ExecStart=/usr/sbin/nginx -g daemon on; master_process on; (code=exited, status=0/SUCCESS)
   Main PID: 2032 (nginx)
    Tasks: 2 (limit: 1130)
   Memory: 1.7M (peak: 1.9M)
      CPU: 12ms
   CGroup: /system.slice/nginx.service
           └─2032 "nginx: master process /usr/sbin/nginx -g daemon on; master_process on;"
             └─2033 "nginx: worker process"

Jun 07 06:23:33 ip-172-31-35-13 systemd[1]: Starting nginx.service - A high performance web server and a reverse proxy server...
Jun 07 06:23:33 ip-172-31-35-13 systemd[1]: Started nginx.service - A high performance web server and a reverse proxy server.
root@ip-172-31-35-13:/home/ubuntu#
```

i-08a53fb74ab102a02 (Nginx Web Server)  
PublicIPs: 13.211.163.186 PrivateIPs: 172.31.35.13

**Gambar 4.14.** Status Nginx Aktif

## Welcome to nginx!

If you see this page, the nginx web server is successfully installed and working. Further configuration is required.

For online documentation and support please refer to [nginx.org](https://nginx.org).  
Commercial support is available at [nginx.com](https://nginx.com).

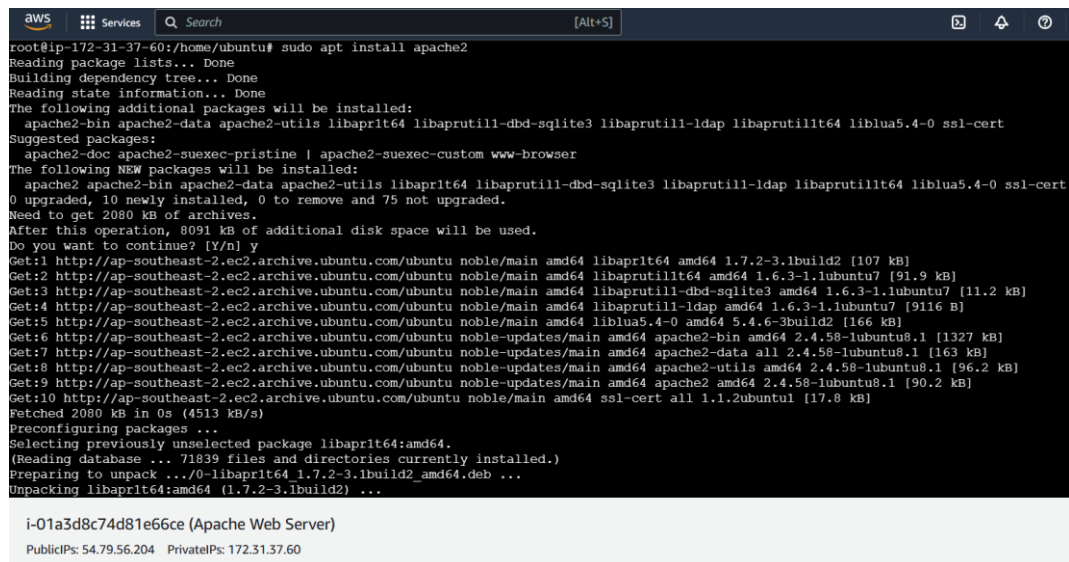
*Thank you for using nginx.*

**Gambar 4.15.** Status Nginx Aktif Pada Browser

Pada Gambar 4.15. kita dapat melihat status aktif *web server* Nginx dengan mengakses *public ip address* pada instance pertama di Browser.

### 4.3. Pemasangan Web Server Apache

Tahap selanjutnya adalah instalasi *Web Server Apache* pada *instance* ke-dua dengan perintah “*sudo apt install apache2*”. Proses dapat dilihat pada Gambar 4.16.



```

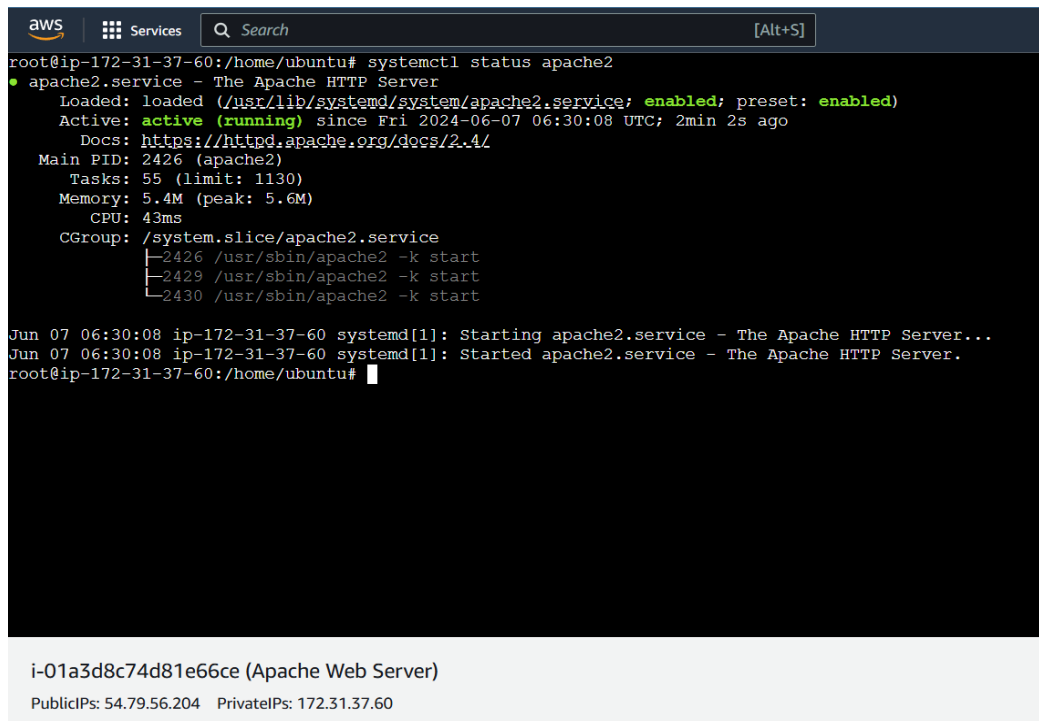
aws | Services | Search | [Alt+S]
root@ip-172-31-37-60:/home/ubuntu# sudo apt install apache2
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following additional packages will be installed:
  apache2-bin apache2-data apache2-utils libapr1t64 libaprutil1-dbd-sqlite3 libaprutil1-ldap libaprutil1t64 liblua5.4-0 ssl-cert
Suggested packages:
  apache2-doc apache2-suexec-pristine | apache2-suexec-custom www-browser
The following NEW packages will be installed:
  apache2 apache2-bin apache2-data apache2-utils libapr1t64 libaprutil1-dbd-sqlite3 libaprutil1-ldap libaprutil1t64 liblua5.4-0 ssl-cert
0 upgraded, 10 newly installed, 0 to remove and 75 not upgraded.
Need to get 2080 kB of archives.
After this operation, 8091 kB of additional disk space will be used.
Do you want to continue? [Y/n] y
Get:1 http://ap-southeast-2.ec2.archive.ubuntu.com/ubuntu noble/main amd64 libapr1t64 amd64 1.7.2-3.1build2 [107 kB]
Get:2 http://ap-southeast-2.ec2.archive.ubuntu.com/ubuntu noble/main amd64 libaprutil1t64 amd64 1.6.3-1.1ubuntu7 [91.9 kB]
Get:3 http://ap-southeast-2.ec2.archive.ubuntu.com/ubuntu noble/main amd64 libaprutil1-dbd-sqlite3 amd64 1.6.3-1.1ubuntu7 [11.2 kB]
Get:4 http://ap-southeast-2.ec2.archive.ubuntu.com/ubuntu noble/main amd64 libaprutil1-ldap amd64 1.6.3-1.1ubuntu7 [9116 B]
Get:5 http://ap-southeast-2.ec2.archive.ubuntu.com/ubuntu noble/main amd64 liblua5.4-0 amd64 5.4.6-3build2 [166 kB]
Get:6 http://ap-southeast-2.ec2.archive.ubuntu.com/ubuntu noble-updates/main amd64 apache2-bin amd64 2.4.58-lubuntu8.1 [1327 kB]
Get:7 http://ap-southeast-2.ec2.archive.ubuntu.com/ubuntu noble-updates/main amd64 apache2-data all 2.4.58-lubuntu8.1 [163 kB]
Get:8 http://ap-southeast-2.ec2.archive.ubuntu.com/ubuntu noble-updates/main amd64 apache2-utils amd64 2.4.58-lubuntu8.1 [96.2 kB]
Get:9 http://ap-southeast-2.ec2.archive.ubuntu.com/ubuntu noble-updates/main amd64 apache2 amd64 2.4.58-lubuntu8.1 [90.2 kB]
Get:10 http://ap-southeast-2.ec2.archive.ubuntu.com/ubuntu noble/main amd64 ssl-cert all 1.1.2ubuntu1 [17.8 kB]
Fetched 2080 kB in 0s (4513 kB/s)
Preconfiguring packages ...
Selecting previously unselected package libapr1t64:amd64.
(Reading database ... 71839 files and directories currently installed.)
Preparing to unpack .../0-libapr1t64_1.7.2-3.1build2_amd64.deb ...
Unpacking libapr1t64:amd64 (1.7.2-3.1build2) ...

i-01a3d8c74d81e66ce (Apache Web Server)
PublicIPs: 54.79.56.204 PrivateIPs: 172.31.37.60

```

**Gambar 4.16.** Instalasi Apache Web Server

Setelah instalasi Apache Web Server, maka tahap selanjutnya adalah memastikan *web server* Apache sudah berjalan pada *instance* ke-dua, untuk detailnya ada pada Gambar 4.17.



```

aws | Services | Search | [Alt+S]
root@ip-172-31-37-60:/home/ubuntu# systemctl status apache2
● apache2.service - The Apache HTTP Server
   Loaded: loaded (/usr/lib/systemd/system/apache2.service; enabled; preset: enabled)
   Active: active (running) since Fri 2024-06-07 06:30:08 UTC; 2min 2s ago
     Docs: https://httpd.apache.org/docs/2.4/
    Main PID: 2426 (apache2)
      Tasks: 55 (limit: 1130)
     Memory: 5.4M (peak: 5.6M)
        CPU: 43ms
    CGroup: /system.slice/apache2.service
            └─2426 /usr/sbin/apache2 -k start
              └─2429 /usr/sbin/apache2 -k start
                └─2430 /usr/sbin/apache2 -k start

Jun 07 06:30:08 ip-172-31-37-60 systemd[1]: Starting apache2.service - The Apache HTTP Server...
Jun 07 06:30:08 ip-172-31-37-60 systemd[1]: Started apache2.service - The Apache HTTP Server.
root@ip-172-31-37-60:/home/ubuntu#

i-01a3d8c74d81e66ce (Apache Web Server)
PublicIPs: 54.79.56.204 PrivateIPs: 172.31.37.60

```

**Gambar 4.17.** Status Apache Web Server Aktif

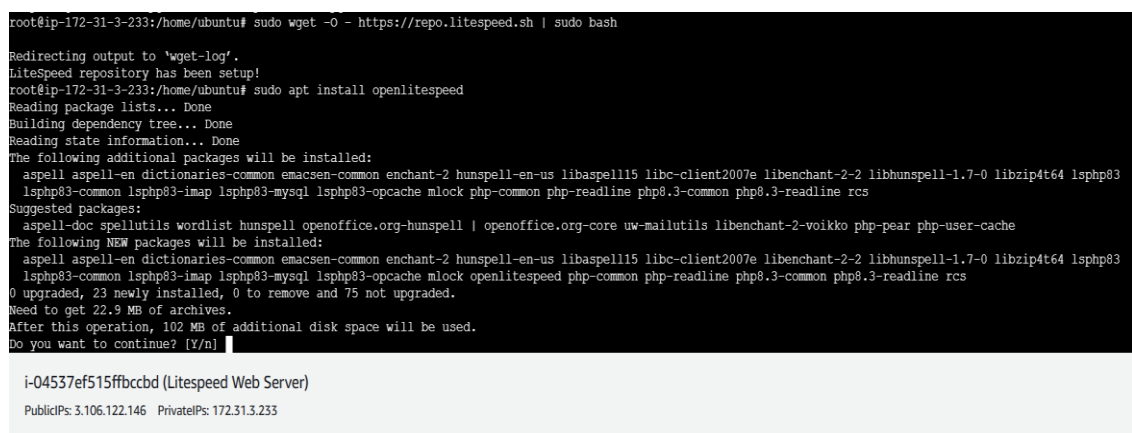


**Gambar 4.18.** Status Apache Web Server Aktif Pada Browser

Pada Gambar 4.18. kita dapat melihat status aktif *web server* Apache dengan mengakses *public ip address* pada instance kedua di Browser.

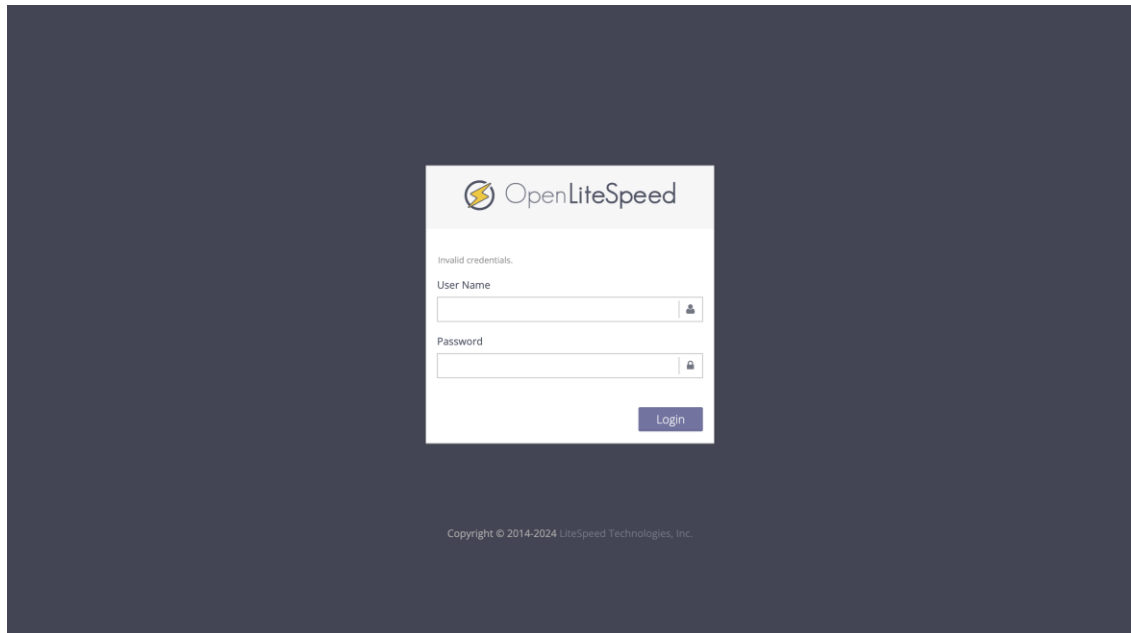
#### 4.4. Pemasangan Web Server LiteSpeed

Langkah selanjutnya adalah installasi web server LiteSpeed di instance ke-tiga dan lanjut mengkonfigurasinya. Perintah yang dipakai “*sudo wget -O - https://repo.litespeed.sh | sudo bash*” tunggu sejenak, lalu masukkan perintah “*sudo apt install openlitespeed*”, proses installasi dapat dilihat pada Gambar 4.19.



**Gambar 4.19.** Installasi OpenLiteSpeed Web Server

Pada Gambar 4.20. dapat dilihat status aktif dari *web server* Litespeed dengan mengakses *public ip address* dari *instance* ke-tiga ditambah dengan “:7080” pada Browser untuk dapat mengakses admin *web server* Litespeed yang dapat dilihat pada Gambar 4.20.



**Gambar 4.20.** Status Aktif OpenLiteSpeed Web Server

Dikarenakan web server Litespeed memiliki halaman admin untuk konfigurasi lebih lanjut, maka kita perlu merubah *username* dan *password* agar dapat melakukan konfigurasi lanjutan, dengan perintah “*sudo /usr/local/lsws/admin/misc/admpass.sh*”. Dapat dilihat pada Gambar 4.21.

```
root@ip-172-31-3-233:/home/ubuntu# sudo /usr/local/lsws/admin/misc/admpass.sh

Please specify the user name of administrator.
This is the user name required to login the administration Web interface.

User name [admin]: admin

Please specify the administrator's password.
This is the password required to login the administration Web interface.

Password:
Retype password:
```

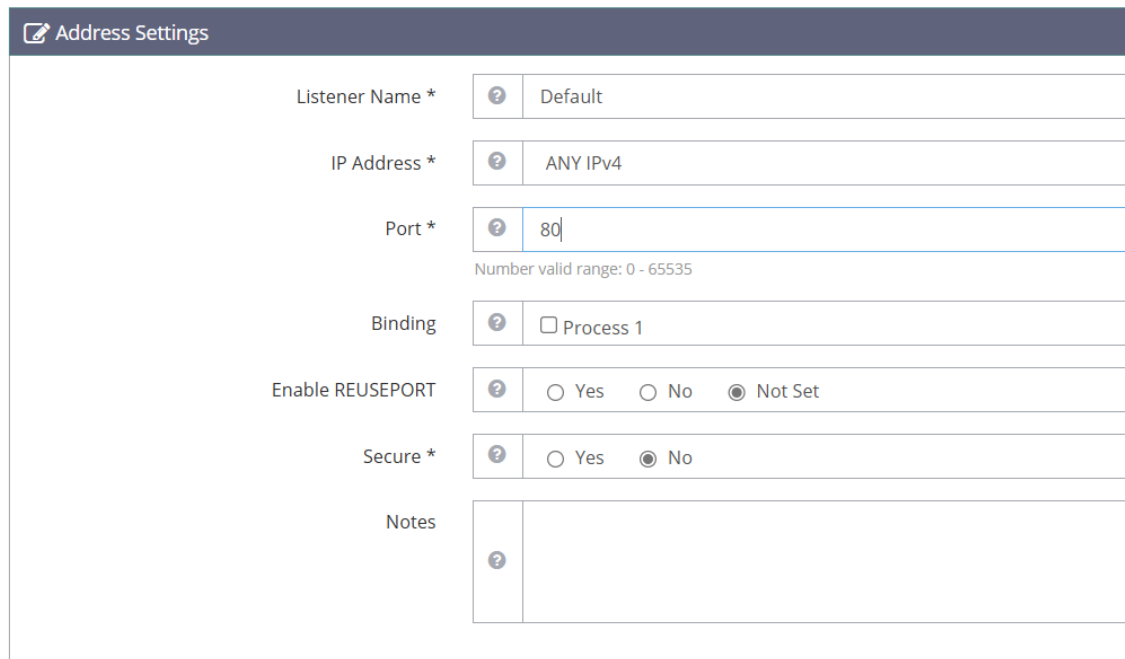
i-04537ef515ffbccbd (Litespeed Web Server)

PublicIPs: 3.106.122.146 PrivateIPs: 172.31.3.233

**Gambar 4.21.** Konfigurasi *username* dan *password* Litespeed



Setelah melakukan konfigurasi untuk *username* dan *password*, akses kembali halaman admin Litespeed. Setelah berhasil masuk ke halaman admin litespeed, konfigurasi selanjutnya adalah melakukan perubahan *port default* pada web server litespeed agar lebih mudah untuk diakses. Setting *port* yaitu dengan masuk ke menu “Listener > Default > Edit > Port : 8088 menjadi 80 > Save”. Dapat dilihat konfigurasi pada Gambar 4.22.



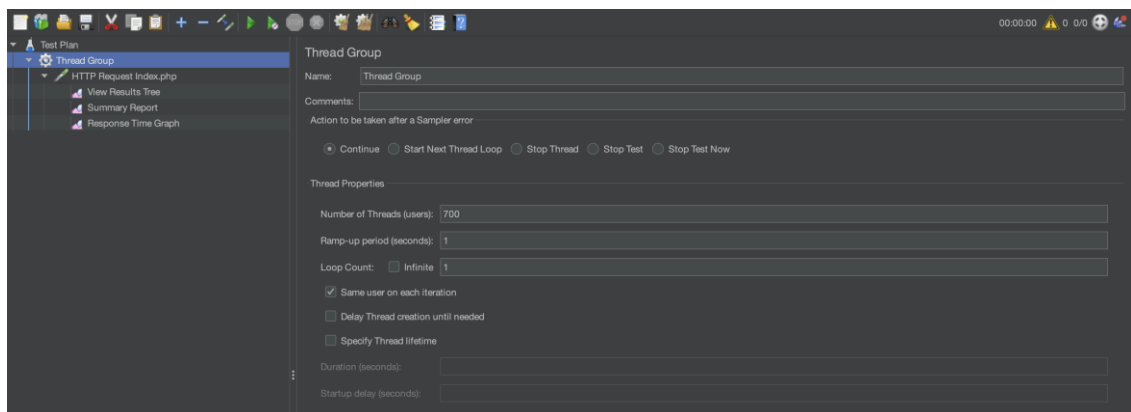
The screenshot displays the 'Address Settings' configuration page for Litespeed. The interface includes a header bar with a pencil icon and the text 'Address Settings'. Below this, there are several configuration fields, each with a help icon (a question mark in a circle) to its left. The fields are: 'Listener Name \*' with the value 'Default'; 'IP Address \*' with the value 'ANY IPv4'; 'Port \*' with the value '80' and a note 'Number valid range: 0 - 65535'; 'Binding' with a checkbox for 'Process 1'; 'Enable REUSEPORT' with three radio buttons: 'Yes', 'No', and 'Not Set' (which is selected); 'Secure \*' with two radio buttons: 'Yes' and 'No' (which is selected); and 'Notes' with a text area and a help icon. The entire form is enclosed in a light gray border.

Listener Name *	? Default
IP Address *	? ANY IPv4
Port *	? 80 Number valid range: 0 - 65535
Binding	? <input type="checkbox"/> Process 1
Enable REUSEPORT	? <input type="radio"/> Yes <input type="radio"/> No <input checked="" type="radio"/> Not Set
Secure *	? <input type="radio"/> Yes <input checked="" type="radio"/> No
Notes	? 

**Gambar 4.22.** Konfigurasi *port* Litespeed

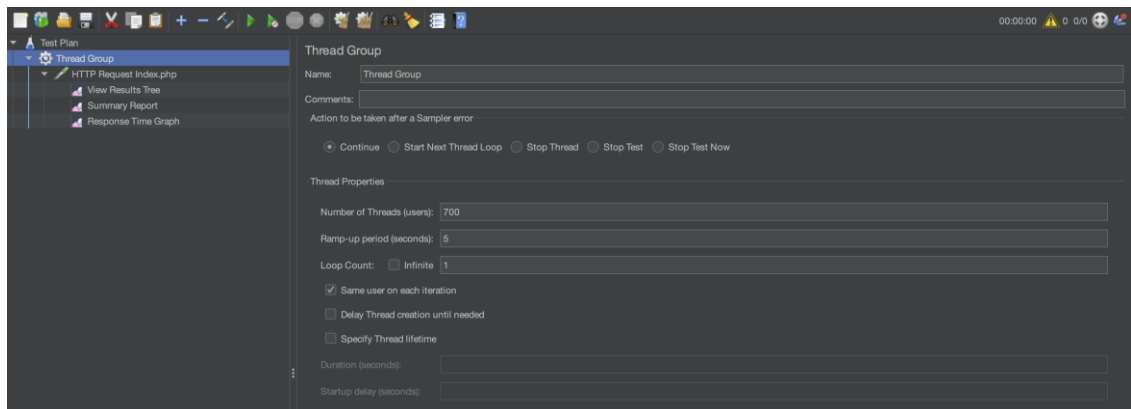
#### 4.5. Pengaturan Apache JMeter

Apache JMeter adalah aplikasi yang akan penulis gunakan untuk melakukan stress testing dan mendapatkan nilai pada parameter *Error*, *response time*, *throughput*, *data sent*, dan *data received*. Pada aplikasi JMeter dilakukan konfigurasi untuk melakukan *stress testing* dengan jumlah pengguna dan durasi waktu yang diperlukan oleh setiap pengguna untuk mengakses web server tersebut. Berikut Gambar 4.23. merupakan konfigurasi pada aplikasi JMeter untuk pengujian 700 dalam interval waktu 1 detik.



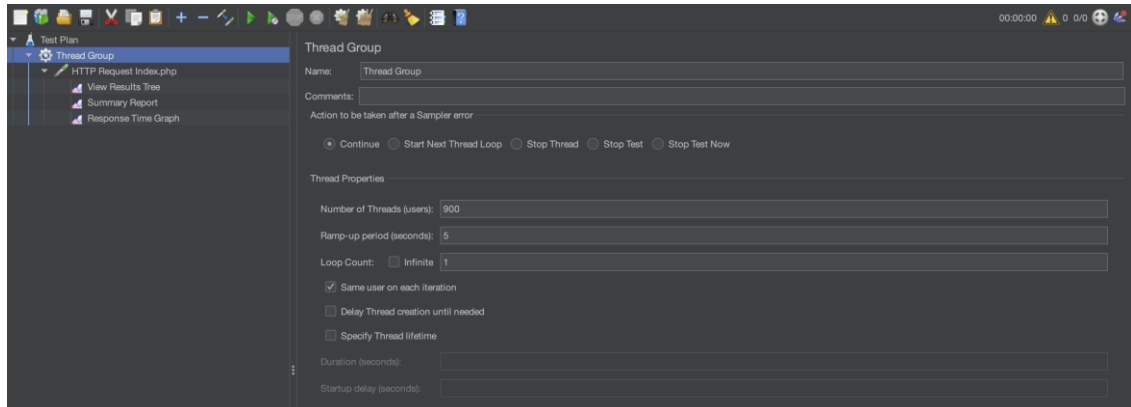
**Gambar 4.23.** Pengaturan JMeter untuk 700 pengguna dalam 1 detik

Lalu dilakukan pengujian untuk jumlah pengguna yang sama, namun dengan penyesuaian pada interval waktu yang berbeda, yaitu menjadi 700 pengguna dengan interval waktu 5 detik, seperti Gambar 4.24.



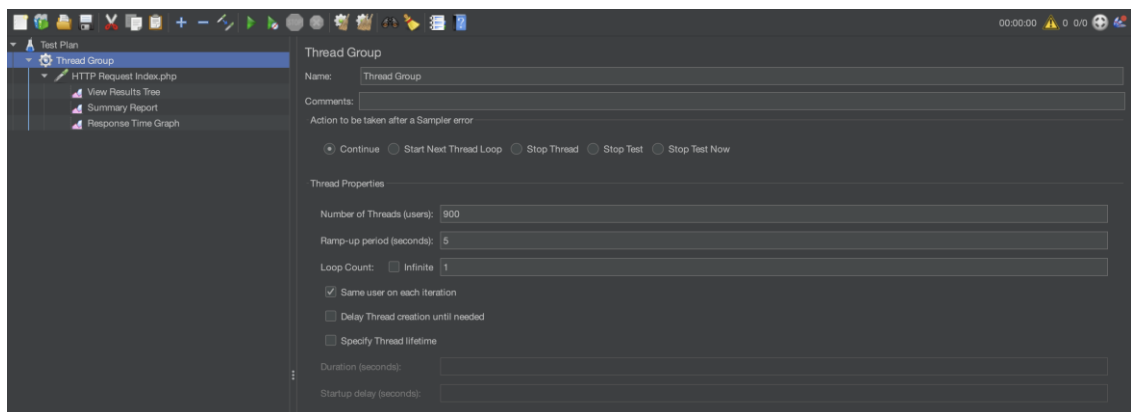
**Gambar 4.24.** Pengaturan JMeter 700 pengguna dalam 5 detik

Setelah melakukan pengujian untuk 700 users dalam interval waktu 1 detik dan 5 detik, Kemudian pengujian diteruskan dengan jumlah pengguna sebanyak 900 dalam interval waktu 1 detik, seperti Gambar 4.25.



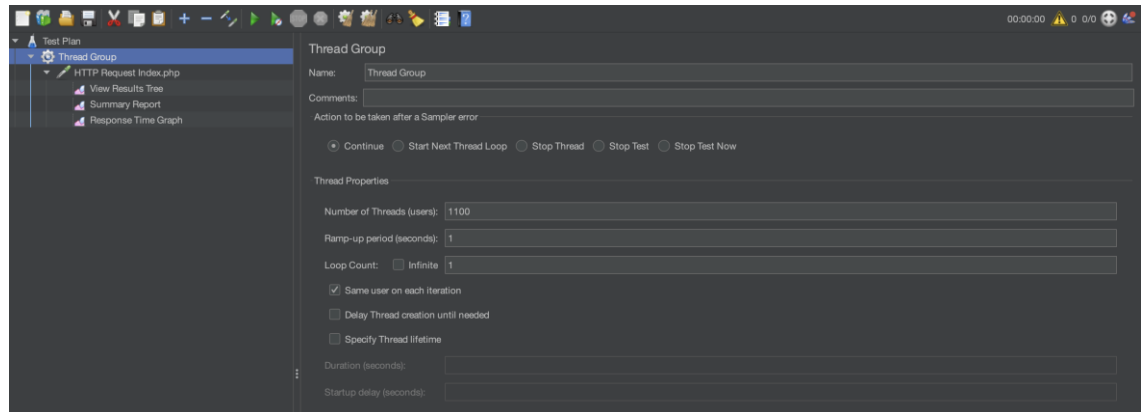
**Gambar 4.25.** Pengaturan Apache JMeter 900 users dalam 1 detik

Lalu dilakukan pengujian untuk jumlah pengguna yang sama, namun dengan penyesuaian pada interval waktu yang berbeda, yaitu menjadi 900 pengguna dengan interval waktu 5 detik, seperti yang tertera pada Gambar 4.26.



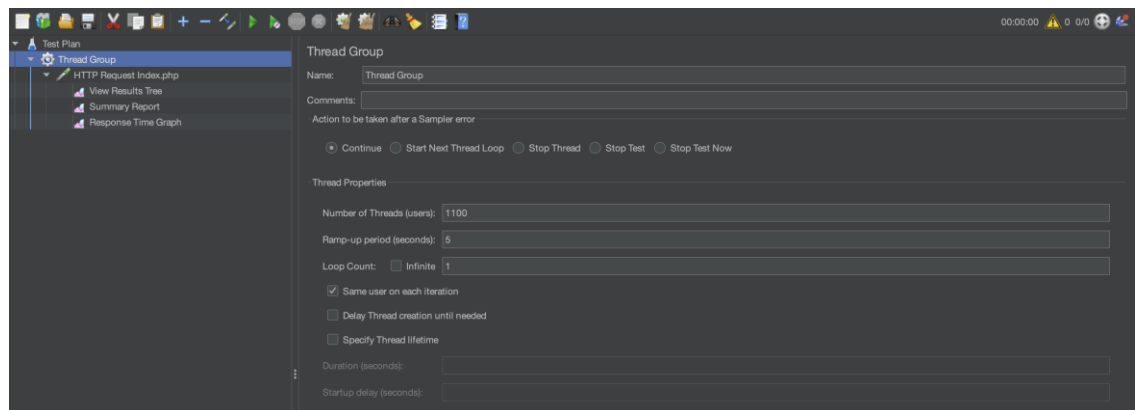
**Gambar 4.26** Pengaturan Apache JMeter 900 users dalam 5 detik

Setelah melakukan pengujian untuk 700 users dalam interval waktu 1 detik dan 5 detik, Kemudian pengujian diteruskan dengan jumlah pengguna sebanyak 1100 dalam interval waktu 1 detik, seperti yang tertera pada Gambar 4.27.



**Gambar 4.27** Pengaturan Apache JMeter 1100 users dalam 1 detik

Lalu dilakukan pengujian untuk jumlah pengguna yang sama, namun dengan penyesuaian pada interval waktu yang berbeda, yaitu menjadi 1100 pengguna dengan interval waktu 5 detik seperti Gambar 4.28.

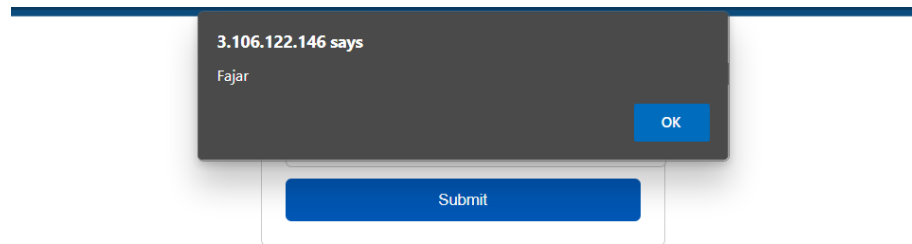


**Gambar 4.28** Pengaturan Apache JMeter 1100 users dalam 5 detik

#### 4.6. Percobaan Kinerja Server

Pengujian kinerja web server dilakukan secara bergantian dengan menggunakan metode *stress testing* pada aplikasi Apache JMeter.

Pengujian server web dilakukan dengan mengakses file web bernama index.html yang memiliki ukuran 1,682 bytes dan berisikan teks “*input your text*” yang sudah dimodifikasi dengan html, css, dan js. File html ini akan berada didalam direktori “/var/www/html” untuk web server Nginx dan Apache. Untuk LiteSpeed direktorinya berada di “/usr/local/lsws/Example/html”. Berikut merupakan tampilan dari web statis yang sudah disiapkan untuk melakukan uji coba yang dilihat pada Gambar 4.29.



**Gambar 4.29** Tampilan Halaman

Setelah melakukan pengujian menggunakan aplikasi JMeter, berikut merupakan seluruh hasil dari tiap pengujian dengan parameter yang sudah ditentukan, Berikut adalah perbandingan dari masing masing parameter pada tiap web server.

**Tabel 4.1.** Perbandingan Parameter *Error (%)*

<b>Jumlah Pengujian</b>	<b>Nginx</b>	<b>Apache</b>	<b>Litespeed</b>
700 users 1 second	0.0	0.0	0.0
700 users 5 second	0.0	0.0	0.0
900 users 1 second	3.4	0.0	0.0
900 users 5 second	0.2	0.0	0.0
1100 users 1 second	5.6	2.13	0.0
1100 users 5 second	0.8	0.71	0.0

Dari Tabel 4.1. dapat dilihat untuk parameter *Error*, *web server* Litespeed lebih unggul dibandingkan dengan *web server* Apache dan Nginx, dengan jumlah 0 persen untuk semua pengujian. Sedangkan Nginx memiliki *Error* pada pengujian untuk 900 user 1 detik sampai pengujian 1100 user untuk 5 detik. Namun untuk *web server* Apache lebih unggul dari Nginx dikarenakan memiliki tingkat *Error* yang lebih sedikit untuk pengujian 1100 user untuk 1 detik dan 5 detik.

**Tabel 4.2.** Perbandingan Parameter *Throughput (request/sec)*

<b>Jumlah Pengujian</b>	<b>Nginx</b>	<b>Apache</b>	<b>Litespeed</b>
700 users 1 second	282.3	285.4	320.1
700 users 5 second	134.8	132.7	169.7
900 users 1 second	318.4	309.9	377.5
900 users 5 second	157.8	155.6	166.3
1100 users 1 second	442.1	444.9	451.5
1100 users 5 second	187.5	187.5	194.7

Dari Tabel 4.2. dapat dilihat bahwa *web server* Litespeed unggul pada parameter *throughput*. Litespeed unggul pada jumlah pengujian 700 users untuk 1 dan 5 detik, lalu pada pengujian 900 users untuk 1 detik Litespeed juga unggul. Namun, untuk pengujian pada jumlah 900 users untuk 5 detik dan 1100 users untuk 1 dan 5 detik, ketiga *web server* memiliki angka yang tidak terlalu berbeda. Ini membuat *web server* Litespeed dapat menerima dan memproses permintaan dari *client* cepat.

**Tabel 4.3.** Perbandingan Parameter *Data Received (KB/sec)*

<b>Jumlah Pengujian</b>	<b>Nginx</b>	<b>Apache</b>	<b>Litespeed</b>
700 users 1 second	598	456.7	582
700 users 5 second	245	220.7	230.9
900 users 1 second	688	504.9	690.0
900 users 5 second	286	242.8	236.3
1100 users 1 second	806	705	784.8
1100 users 5 second	341	331	339.5

Dari Tabel 4.3. dapat dilihat bahwa *web server* Apache unggul pada parameter *Data Received*. Pada pengujian dengan jumlah 700 user, 900 user, 1100 user untuk waktu 1 detik, Apache lebih unggul dari Nginx dan Litespeed. Namun untuk pengujian pada waktu 5 detik, ketiga server tersebut memiliki angka yang berdekatan dan tidak dapat

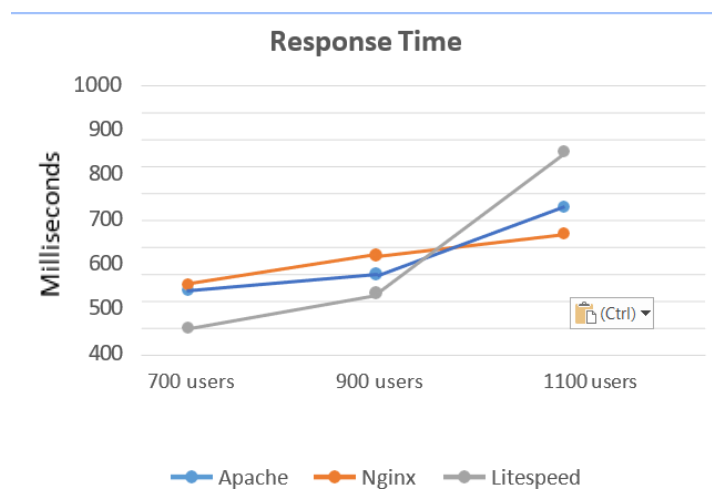
ditentukan *web server* mana yang lebih unggul. Ini membuat *web server* Apache lebih unggul dalam mengirim data kepada *client* dengan lebih optimal.

**Tabel 4.4.** Perbandingan Parameter *Data Sent (KB/sec)*

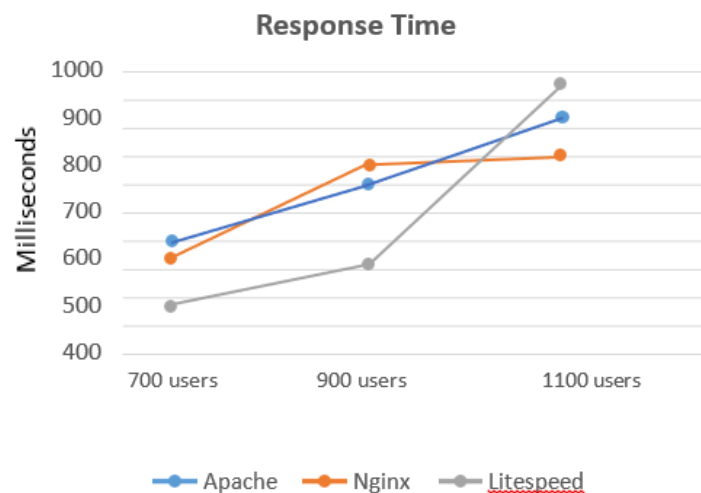
Jumlah Percobaan	Nginx	Apache	Litespeed
700 users 1 second	38.1	36.8	39
700 users 5 second	15.6	15.1	19.3
900 users 1 second	43.9	39.2	36.4
900 users 5 second	18.2	17.7	20.1
1100 users 1 second	51.4	50	25.2
1100 users 5 second	21.3	21	23.5

Dari Table 4.4. dapat dilihat bahwa ketiga *web server* memiliki kinerja yang sama pada parameter *Data Sent*, dikarenakan memiliki angka yang tidak terlalu berbeda antara satu dengan lainnya. Yang membuat ketiga *web server* mengirim permintaan kepada *web server* dengan optimal.

Pada aplikasi JMeter, terdapat fitur untuk menampilkan grafik waktu respons yang menampilkan hasil waktu respons dari *web server* tersebut. Berikut adalah grafik waktu respons yang dapat dilihat pada Gambar 4.30. dan Gambar 4.31.



**Gambar 4.30.** Hasil Grafik *response time 1 Second*



**Gambar 4.31.** Hasil Grafik *response time 5 Second*

Dari kedua grafik diatas dapat kita menemukan bahwa *web server* LiteSpeed memiliki *response time* lebih baik dibandingkan dengan *web server* Apache dan Nginx pada pengujian 700 user dan 900 user untuk waktu 1 dan 5 detik. LiteSpeed lebih cepat dalam menerima dan membalas permintaan dari *client* kepada *web server*, Namun pada pengujian 1100 user untuk waktu 1 dan 5 detik, LiteSpeed memiliki *response time* yang lebih buruk dibandingkan dengan *web server* Apache dan Nginx. Dan Nginx memiliki *response time* yang unggul dibandingkan dengan Nginx dan LiteSpeed pada pengujian 1100 user untuk 1 dan 5 detik.

#### 4.7.Pembahasan

Dari hasil pengujian dengan metode *stress testing* dengan menggunakan aplikasi JMeter pada ketiga *web server* Nginx, Apache dan LiteSpeed maka hasil yang didapatkan adalah :

1. Dalam memilih *web server* untuk menangani permintaan yang besar secara bersamaan, LiteSpeed menjadi pilihan yang baik karena keunggulannya dalam parameter *error*, *throughput*.
2. Dalam memilih *web server* yang dapat mengelola *bandwith data*, *web server* Apache menjadi pilihan yang layak karena keunggulannya dalam parameter *data received*.



3. Dalam memilih *web server* untuk menangani permintaan dengan cepat, Litespeed menjadi pilihan yang baik karena unggul pada parameter *response time*, namun untuk menangani permintaan dengan cepat dan jumlah yang cukup besar, *web server* Apache menjadi pilihan yang lebih baik.

## BAB V

### KESIMPULAN DAN SARAN

#### 5.1. Kesimpulan

Dari hasil analisis pembahasan yang telah dilakukan sebelumnya, Keputusan yang dapat diambil adalah sebagai berikut :

1. Dalam pengujian *Troughput*, *web server* Litespeed lebih unggul dibandingkan dengan *web server* Apache maupun Nginx. Hal ini dikarenakan Litespeed dapat mengelola permintaan pengguna dengan kapasitas bandwidth yang lebih besar.
2. Dalam pengujian *Error*, *litespeed* memiliki tingkat kegagalan yang rendah dibandingkan dengan *web server* Apache dan Nginx, sehingga kinerjanya lebih optimal dalam layanan *web server*.
3. Dalam pengujian *Data Sent*, *web server* Apache menunjukkan performa yang lebih baik daripada *web server* Nginx dan Litespeed. Apache lebih unggul dalam mengelola pengiriman data yang diminta oleh klien, sehingga data dapat diterima dengan baik oleh klien.
4. Dalam pengujian *Data Received*, *web server* Apache lebih unggul dibandingkan dengan Litespeed dan Nginx. *Web server* Apache lebih responsif dalam menanggapi penerimaan dari klien.
5. Dalam pengujian *Response Time*, *web server* Litespeed lebih unggul dibandingkan dengan Apache dan Nginx. Namun, untuk permintaan yang cukup besar, *web server* Apache lebih unggul dari Nginx dan Litespeed.

#### 5.2. Saran

Dari hasil penelitian dan pengujian yang telah dilakukan, pengujian *web server* dapat dikembangkan lebih lanjut dengan menambahkan parameter lain serta menggunakan berbagai alat atau perangkat lunak tambahan. Hal ini bertujuan agar hasil yang diperoleh lebih baik dan dapat digunakan sebagai dasar untuk penelitian selanjutnya.

## DAFTAR PUSTAKA

- Andhica, I. Y., & Irwan, D. (2017). Performa Kinerja Web Server Berbasis Ubuntu Linux Dan Turney Linux. Universitas Islam 45.
- AWS.Amazon.com. (n.d). About Amazon Web Service. Retrieved June 1, 2024. From <https://aws.amazon.com/about-aws>
- Busran, & Ridwan, A., (2020). Analisis Perbandingan Performa Apache Web Server Dan Nginx Menggunakan Apache Jmeter. Institut Teknologi Informasi
- Chandra, A. Y. (2019). Analisis Performansi Antara Apache & Nginx Web Server dalam Menangani Client Request. Universitas Mercu Buana Yogyakarta.
- Guntoro, Kusumo, D. S., & Adiwijaya. D. (2015). Analisis Web Performance dan Load Test Studi Kasus : Topologi Cloud Microsoft Azure Test Rig Pada I banking Bank XYZ. Universitas Telkom.
- Harris, M., & Khan, R. Z. (2018). A Systematic Review on Cloud Computing, Aligarh Mulim University. India.
- Hostinger.com (n.d), Nginx vs Apache – Choosing The Best Web Server. Retrieved March 2, 2024. From <https://hostinger.com/tutorials/nginx-vs-apache-what-to-use>.
- Irza, I. F., Zulhendra, & Efrizon. (2017). Analisis Perbandingan Kinerja Web Server Apache dan Nginx Menggunakan Httpperf Pada Portal Berita (Studi Kasus beritalinux.com). Universitas Negeri Padang
- Jmeter.Apache.com. (n.d), About Jmeter. Retrieved June 5. 2024. From <https://jmeter.apache.org>
- Kisnandar, Rasma Bayu. (2019). Analisis Perbandingan Kineja Web Server Nginx, Apache, dan Lighttpd Dengan Metode Stress Test. STMIK Akakom Yogyakarta
- Meire, J. D., Farre, C., Bansode, P., Barber, S., & Rea, D. (2007) Performance Testing Guidance For Web Application. Microsoft Corporation.
- Netcraft.com (n.d). About Netcraft. Retrieved March 20, 2021, from <https://www.netcraft.com/blog/february-2024-web-server-survey/>
- Putra, Ronaldi. (2020) Analisa Perbandingan Web Server Nginx Dengan Litespeed. Universitas Islam Riau.

- Riswandi, Kasim, & Raharjo, M. F. (2020). Evaluasi Kinerja Web Server Apache menggunakan Protokol HTTP2. Politeknik Negeri Ujung Padang.
- Satwika, I. K. S., & Semadi, K. N. (2020). Perbandingan Performasi Web Server Apache Dan Nginx Dengan Menggunakan IPV6. STIMIK STIKOM Indonesia.
- Technologies, L. (n.d.). Mengenal Litespeed – Web Server Technology. Retrieved March 2, 2024, from <https://help.idcloudhost.com/id/articles/1463024-mengenal-litespeed-web-server-technology>.