

# ANALISIS PERBANDINGAN PERFORMA RESTFUL API ANTARA EXPRESS.JS DENGAN LARAVEL *FRAMEWORK* DENGAN JMETER

Wira Hadinata<sup>1\*</sup>, Lilis Stianingsih<sup>2</sup>

<sup>1,2</sup>Institut Teknologi dan Bisnis Bina Sarana Global; Jl. Aria Santika No.43, Margasari, Kec. Karawaci, Kota Tangerang, Banten 15113; (021) 5522727

---

## Riwayat artikel:

Received: 22 November 2022

Accepted: 29 Desember 2023

Published: 1 Januari 2024

## Keywords:

Restful API;  
Performa Rest API;  
Express.js;  
Laravel *Framework*.

## Correspondent Email:

wira@global.ac.id

---

**Abstrak.** Application Programming Interface (API) merupakan sebuah antarmuka yang mampu mengintegrasikan data dan menghubungkan suatu aplikasi yang berjalan di banyak platform sehingga dapat saling terhubung, salah satu implementasi API tersebut adalah RESTful API. Pemilihan teknologi dalam pembuatan RESTful API sangatlah penting karena dapat mempengaruhi kinerja server. Kerangka kerja Laravel dan Express.js adalah dua teknologi backend yang digunakan untuk membuat RESTful API. Untuk mengetahui performa masing-masing *framework*, perlu dilakukan pengujian dengan menggunakan metode *performance test* untuk mengetahui respon time serta penggunaan CPU dan memori. Hasil pengujian menunjukkan bahwa Express.js mempunyai waktu respon rata-rata 48,68 ms lebih cepat dibandingkan *framework* Laravel yang memperoleh waktu respon rata-rata 635,17 ms, sedangkan untuk penggunaan CPU *framework* Express.js 17% dan *framework* Laravel 20%. Untuk penggunaan memori *framework* Express.js 63% lebih sedikit dari *framework* Laravel 72%. Jadi Express.js cocok untuk sistem yang diakses oleh banyak pengguna dan ditempatkan pada server dengan spesifikasi tinggi. Sedangkan *framework* Laravel cocok diterapkan pada sistem dengan akses pengguna yang lebih sedikit dan dapat ditempatkan pada server dengan spesifikasi lebih tinggi.

**Abstract.** Application Programming Interface (API) is an interface that is capable of integrating data and connecting applications running on many platforms so that they can be connected to each other, one implementation of this API is the RESTful API. The choice of technology in creating a RESTful API is very important because it can affect server performance. Laravel *framework* and Express.js are two backend technologies used to create RESTful APIs. To find out the performance of each *framework*, it is necessary to test it using the *performance test* method to determine response time and CPU and memory usage. The test results show that Express.js has an average response time of 48.68 ms, which is faster than the Laravel *framework* which has an average response time of 635.17 ms, while the CPU usage for the Express.js *framework* is 17% and the Laravel *framework* is 20%. The Express.js *framework* uses 63% less memory than the Laravel *framework*, 72%. So Express.js is suitable for systems that are accessed by many users and placed on servers with high specifications. Meanwhile, the Laravel *framework* is suitable for applications with less user access and can be placed on servers with higher specifications.

---

## 1. PENDAHULUAN

Saat ini perkembangan teknologi, khususnya di bidang aplikasi mengalami perkembangan yang signifikan. Terdapat hubungan linear secara signifikan antara perkembangan teknologi dengan komunikasi serta pola pikir manusia [1], salah satu aspek perkembangannya seperti di mana dahulu aplikasi hanya berbasis monolith, sekarang arsitektur aplikasi sudah berbasis *microservices* dan juga *nanoservices*. Mereka yang mengadopsi arsitektur *microservices* melaporkan tingkat keberhasilan yang tinggi[2]. Perkembangan arsitektur ini menjawab tantangan bisnis manusia di era zaman sekarang. Salah satu implementasi arsitektur *microservices* adalah perkembangan aplikasi berbasis mobile, karena pangsa pasar sangat luas, dan bisa menangani segala kebutuhan teknologi para pelaku bisnis.

Tentunya teknologi aplikasi khususnya mobile, sangat membutuhkan kinerja yang baik[3], dan para pelaku bisnis pun harus mengikuti tren ini, supaya bisnis tetap berkesinambungan. Dengan adanya aplikasi mobile dengan berbagai platform diperlukan adanya integrasi data agar tidak terjadi duplikasi data di sebuah aplikasi yang berjalan di berbagai platform yang berbeda.

Application Programming Interface (API) sendiri merupakan sebuah *interface* yang mampu untuk mengintegrasikan data dan menghubungkan sebuah aplikasi yang berjalan di banyak platform sehingga dapat saling terhubung satu sama lain. Selain dapat bertukar data di berbagai platform yang berbeda, API juga dapat mempercepat proses development dengan menyediakan function secara terpisah sehingga developer tidak perlu membuat fitur yang serupa [4].

Representational State Transfer (REST) adalah sebuah arsitektur metodekomunikasi yang menggunakan protokol HTTP untuk pertukaran data dimana metode ini sering diterapkan dalam pengembangan aplikasi. RESTful salah satu arsitektur API yang cukup populer. Dalam pembuatan RESTful API banyak sekali bahasa pemrograman dan *framework* yang bisa digunakan. Pemilihan teknologi dalam pengembangan RESTful API sangat penting karena dapat mempengaruhi performa pada server baik secara *response time*, *cpu usage* maupun *memory usage*. Maka dari itu dalam pengembangan RESTful API perlu

memilih bahasa pemrograman dan *framework* yang tepat sehingga server RESTful API dapat menangani *request* dari *client* tanpa simpul yang berpartisipasi melacak keadaan sesi sebelumnya [5].

Bahasa dan *framework* pemrograman banyak sekali yang dapat digunakan dalam membangun RESTful API, namun ada beberapa bahasa pemrograman yang populer dan banyak digunakan. Salah satu *framework* yang banyak digunakan oleh programmer adalah *framework* laravel. Menurut Ambriani dan Nurhidayat [6], Laravel adalah salah satu *framework* berbasis PHP yang sifatnya sumber terbuka, dan menggunakan konsep *model – view – controller*. Laravel berada di bawah lisensi MIT *License* yang menggunakan Github sebagai tempat berbagi *code*.

Bahasa pemrograman yang cukup populer dan banyak digunakan adalah Javascript. Berdasarkan penelitian dari Minokaura dkk, menyatakan bahwa hampir seluruh situs web Jaringan Dokumentasi dan Informasi Hukum (JDIH) kabupaten dan kota yang berada di Pulau Jawa menggunakan jQuery sebagai *framework* Javascript yang digunakan[7]. Javascript sendiri pada sisi *server backend* dijalankan oleh sebuah *platform* yang bernama Node.js dan memiliki *framework* Express.js. Pada penelitian Nasution [8], didapatkan bahwa Performa Express.JS pada web Ubaform dengan API adalah 877 ms, *read* data 313ms, *update* data 628 ms dan *delete* data 752 ms. Sehingga disimpulkan bahwa kecepatan yang didapat dari implementasi Express JS dengan data web Ubaform bahkan tidak sampai menghabiskan waktu 1 detik.

Berdasarkan uraian diatas, memilih sebuah teknologi atau *framework* dalam sebuah pengembangan aplikasi sangatlah penting khususnya dalam mengembangkan sebuah RESTful API karena dapat mempengaruhi kinerja dari aplikasi tersebut. Oleh karena itu penelitian ini bertujuan mengidentifikasi cara membandingkan kinerja *framework* Laravel dengan Express.js serta menjelaskan hasil analisis dari perbandingan kinerja *framework* Laravel dengan Express.js menggunakan metode *performance testing*.

Objek penelitian akan menggunakan data karyawan yang disediakan oleh MySQL, data tersebut merupakan sample data yang dikembangkan oleh Giuseppe Maxia dan

Patrick Crews pada tahun 2008 yang bertujuan melakukan pengujian database pada server. Database ini nantinya akan dipanggil menggunakan *framework* Laravel dan Express.js dan memberikan sebuah *response* yang akan ditampilkan kepada user serta mengukur seberapa baik kinerja dari kedua *framework* tersebut.

## 2. TINJAUAN PUSTAKA

### 2.1. Restful API

Restful API merupakan arsitektur untuk penerapan web *service* dalam menerapkan konsep peralihan antar negara [9]. Negara disini dapat dilustrasikan sebagai peramban yang meminta halaman web, pada sisi *server* akan mengirimkan keadaan halaman web saat ini ke peramban. Dengan REST API memungkinkan berbagai sistem untuk dapat berkomunikasi dan mengirim atau menerima data dengan cara yang cukup sederhana. Didalam RESTful API terdapat REST *client* yang dapat mengakses data atau *resource* pada REST *server* dimana setiap *resource* akan dibedakan berdasarkan dari global ID atau URI (*Universal Resource Identifiers*). Hal ini membuat RESTful API sangat cocok diterapkan pada aplikasi yang terintegrasi dengan ponsel pintar.[10]

### 2.2. Node.js

Doglio [11] dalam bukunya mengemukakan, bahwa berikut beberapa perusahaan besar yang menggunakan teknologi node.js dalam produksinya:

- PayPal: Menggunakan Node.js untuk mendukung aplikasi webnya.
- eBay: Menggunakan Node.js terutama karena manfaat yang diberikan I/O asinkronous.
- LinkedIn: Seluruh tumpukan seluler *back-end* dilakukan di Node.js. Dua alasan untuk menggunakannya adalah skala dan kinerja yang diperoleh dari tumpukan sebelumnya.
- Netflix: Menggunakan Node.js di beberapa layanan; sering menulis tentang pengalaman menggunakan Node.js di blog teknologinya di <http://techblog.netflix.com>.
- Yahoo!: Menggunakan Node.js pada beberapa produk, seperti Flickr, My Yahoo!, dan halaman beranda).

### 2.3. Laravel Framework

Menurut penelitian dari Desma Aipina dan Harry Witriyono [12], bahwa Laravel merupakan sebuah kerangka kerja pemrograman yang berbasis *open source* yang dipakai oleh banyak *developer* dari seluruh dunia. Laravel juga menjadi salah satu kerangka kerja yang dapat membantu *developer* aplikasi untuk memaksimalkan penggunaan bahasa pemrograman PHP di dalam proses pengembangan website. Selain dari itu, Laravel juga memiliki beberapa fitur yang menjadi unggulan, seperti *template engine*, *routing*, dan *modularity*. Kerangka kerja Laravel mempunyai kelebihan dalam struktur file dan koding dibandingkan dengan php *native* biasa.

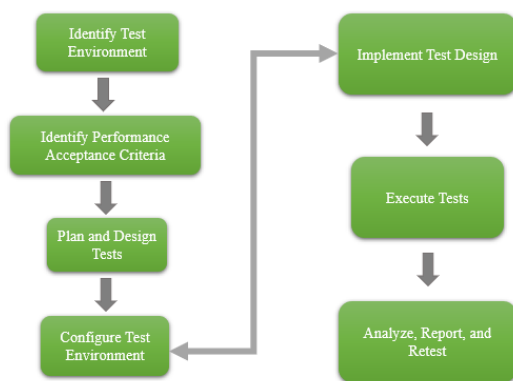
### 2.4. JMeter

Menurut Harijanto dan Ariyanto [13] dalam penelitiannya mengemukakan bahwa pengujian web *server* dengan menggunakan *software* aplikasi berstandar seperti Jmeter diperlukan dalam rangka menghasilkan produk web server yang tidak hanya berkualitas tetapi juga aplikasi dengan biaya perawatan yang rendah (Jmeter bersifat sumber terbuka). Selain itu juga memastikan perangkat lunak atau aplikasi yang kita kembangkan efektif.

## 3. METODE PENELITIAN

Metode yang penulis lakukan pada penelitian adalah menggunakan metode *performance testing*. *Performance testing* adalah suatu proses menjalankan aplikasi dengan mensimulasi *virtual user* menggunakan sebuah tools seolah olah aplikasi sedang berjalan dan di akses oleh *user* yang sebenarnya untuk mengetahui sistem berjalan dengan baik dan memiliki kinerja yang baik [4].

Pengujian aplikasi terdiri dari empat elemen: pengujian fungsional, pengujian kompatibilitas, pengujian kegunaan, dan pengujian kinerja. Jika standar kualitas perangkat lunak dibandingkan dengan standar ISO 25010, aplikasi harus diuji fungsionalitas, kompatibilitas, kegunaan, dan kinerjanya [14]. *Performance testing* akan dilakukan dalam tujuh tahapan seperti pada gambar 3.1.

Gambar 3.1 Tahapan *Performance Testing*

Berikut detail penjelasan tahapan *performance testing* pada gambar 3.1:

### 3.1. Identify Test Environment

Dalam pengujian RESTful API diperlukan identifikasi *test environment* yang akan digunakan dalam memonitoring setiap request yang akan diberikan kepada masing masing *endpoint* di setiap *framework*. Adapun *test environment* yang akan digunakan berupa *server*, *tools* dan *framework* itu sendiri

### 3.2. Identify Performance Acceptance Criteria

Untuk melakukan identifikasi kriteria pengujian perlu disesuaikan dengan kasus penelitian. Seperti yang sudah dijelaskan sebelumnya, pada penelitian ini akan menggunakan data karyawan yang berjumlah 1.500 baris data seperti pada gambar 3.2. Semua data itu diperlukan untuk sebuah sistem yang memerlukan detail setiap karyawan serta melakukan manipulasi pada data tersebut. Dari kasus tersebut terlihat bahwa sistem harus memiliki kecepatan dalam memproses data karena sistem akan diakses oleh banyak *user* yang akan melihat datanya pada sistem.

```

mysql> select count(*) from employees;
+-----+
| count(*) |
+-----+
|      1500 |
+-----+
1 row in set (0.02 sec)

mysql>
  
```

Gambar 3.2 Jumlah Data Karyawan

### 3.3. Plan and Design Test

Pengujian untuk kedua *framework* akan dilakukan dengan mengacu setiap kemungkinan

penggunaan oleh user, yaitu melihat, menambahkan, merubah serta menghapus setiap data.

### 3.4. Configure Test Environment

Pada tahap ini dilakukan konfigurasi pada *environment* yang sudah diidentifikasi pada tahap pertama sehingga setiap *environment* sudah siap digunakan untuk pengujian. Konfigurasi tersebut meliputi instalasi JMeter, konfigurasi *server* dan *framework* pada kedua *environment*.

### 3.5. Implement Test Design

Pada tahap ini *environment* yang sudah disiapkan akan di implementasikan sesuai dengan rencana pengujian yang sudah ditetapkan pada tahap sebelumnya. JMeter yang sudah terpasang pada komputer *client* peneliti, akan dilakukan pengaturan sesuai dengan rencana pengujian.

### 3.6. Execute Tests

Pada tahap ini dilakukan eksekusi pengujian serta memonitoring setiap hasil pengujian dari kedua *framework* baik Laravel maupun Express.js sehingga diperoleh data yang akan digunakan pada tahap selanjutnya. Untuk menjalankan rencana pengujian yang sudah disimpan perlu menuliskan script pada *command prompt*. Script untuk menjalankan sebuah rencana pengujian terbagi menjadi tiga perintah yaitu:

- `-n -t [test_plan_file]` berfungsi untuk mengeksekusi file test pada *directory* yang sudah ditentukan.
- `-l [file_result]` berfungsi untuk mengenerate file csv yang didalamnya berisikan hasil pengujian.
- `-e -o [report_folder]` berfungsi untuk menyimpan file *dashboard* hasil dari pengujian agar mudah dalam menganalisa hasil.

### 3.7. Analyze, report, and Retest

Pada tahapan ini, dilakukan analisa dan perbandingan dari *framework* Laravel dengan Framework Express.Js yang sudah dilakukan pengujian dengan tujuan untuk mengetahui kinerja dari setiap *framework* sehingga diperoleh sebuah hasil yang menyatakan *framework* mana yang lebih optimal secara kinerja.

## 4. HASIL DAN PEMBAHASAN

### 4.1. Identify Test Environment

Pada penelitian ini server yang akan digunakan adalah dua buah *virtual machine* dari layanan *virtual private server* Domainsia. Setiap *framework* akan diletakkan di dua server yang berbeda dengan spesifikasi yang sama dengan tujuan agar setiap *framework* dapat menangani setiap *request* tanpa adanya *sharing resource* antara *framework* Laravel dengan Express.js.

Untuk *tools* yang akan digunakan peneliti dalam memonitoring setiap hasil pengujian adalah JMeter. JMeter akan memberikan keluaran berupa *response time* serta *error rate* dari setiap permintaan terhadap masing masing *endpoint*, selain itu untuk *monitoring resource* peneliti akan menggunakan data *plugin* yang terdapat pada JMeter yaitu *PerfMon Metrics Collector* sehingga dapat mengetahui penggunaan CPU dan memori ketika *server* menangani setiap *request*. Selain *server* dan *tools*, untuk penggunaan *framework* akan menggunakan *framework* Laravel dengan menggunakan versi 10.35.0 serta *framework* Express.js versi 4.17 yang dijalankan menggunakan Node.js versi 16.15, untuk spesifikasi dari *server* yang akan digunakan dapat terlihat pada tabel 1.

Tabel 1 Spesifikasi test Environment

Server	
Cloud Service	Domainsia
CPU	1 Core
Memory	1 GB
HDD	20 GB
Sistem Operasi	Ubuntu Server 22.04
Web Server	Apache
Database	
DBMS	MySQL
Versi	8.0.35
Tools	
Aplikasi	Apache Jmeter
Versi	5.6.2
Framework	
Framework 1	Laravel v 10.35.0
Framework 2	Express.js

### 4.2. Identify Performance Acceptance Criteria

Berdasarkan kebutuhan sistem tersebut, seperti yang terlihat pada tabel 2 salah satu *performance objective* pada penelitian ini adalah *response time*, sehingga untuk melakukan pengujian peneliti menetapkan untuk *response time* tidak melebihi lima detik ketika diakses oleh seribu user untuk kedua *framework*. Selain *response time*, karena pada penelitian ini terdapat perbandingan *framework* maka dari itu perlu juga diketahui penggunaan sumber daya dari masing masing *framework* sehingga *performance objective* lainnya adalah melihat sumber daya yang digunakan yaitu CPU dan memori. Pada penelitian ini untuk kriteria penggunaan sumber daya yaitu dibawah dari 75% ketika kedua *framework* menangani banyak permintaan.

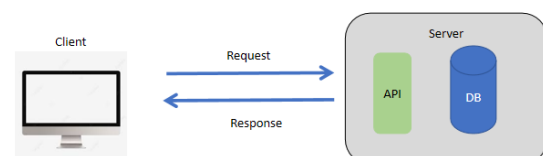
Tabel 2 Kriteria Pengujian

Performance Objective	Criteria
Response Time	< 5 detik ketika 120 user akses
Penggunaan CPU	< 75 %
Penggunaan Memory	< 75 %

### 4.3. Plan and Design Test

Rencana pengujian yang akan dilaksanakan peneliti, adalah dengan cara melakukan permintaan dari *client* menggunakan JMeter ke setiap *endpoint* API yang sudah disediakan *framework* yang sudah diletakkan pada masing masing *server* menggunakan HTTP Method (GET, POST, PUT, DELETE).

Pada gambar 2 dijelaskan bagaimana alur dari pengujian yaitu request dilakukan secara bersamaan dengan penambahan jumlah user dari 100 sampai 1500 user dengan interval setiap pengujian adalah lima menit.



Gambar 4.1 Alur pengujian

#### 4.4. Configure Test Environment

Berikut *capture* terkait konfigurasi status apache di server 1 pada gambar 4.1 dan server 2 pada gambar 4.2 setelah dikonfigurasi.

```

root@belajar1:~# apache2ctl configtest
AH00558: apache2: Could not reliably determine the server's fully qualified domain name, using 127.0.0.1. Set the 'ServerName' directive globally to suppress this message
Syntax OK
root@belajar1:~# service apache2 start
root@belajar1:~# systemctl status apache2.service
● apache2.service - The Apache HTTP Server
   Loaded: loaded (/lib/systemd/system/apache2.service; enabled; vendor preset: enabled)
   Active: active (running) since Mon 2023-12-04 03:06:29 UTC; 2min 46s ago
     Docs: https://httpd.apache.org/docs/2.4/
   Main PID: 5026 (apache2)
     Tasks: 55 (limit: 1131)
    Memory: 5.2M
   CGroup: /system.slice/apache2.service
           └─5026 /usr/sbin/apache2 -k start
             └─5029 /usr/sbin/apache2 -k start
               └─5030 /usr/sbin/apache2 -k start

Dec 04 03:06:29 belajar1 systemd[1]: Starting The Apache HTTP Server...
Dec 04 03:06:29 belajar1 apache2[5026]: AH00558: apache2: Could not reliably determine the server's fully qualified domain name, using 127.0.0.1. Set the 'ServerName' directive globally to suppress this message
Dec 04 03:06:29 belajar1 systemd[1]: Started The Apache HTTP Server.
lines 1-15/15 (END)

```

Gambar 4.2 Apache di Server 1

```

root@belajar2:~# apache2ctl configtest
AH00558: apache2: Could not reliably determine the server's fully qualified domain name, using 127.0.0.1. Set the 'ServerName' directive globally to suppress this message
Syntax OK
root@belajar2:~# service apache2 start
root@belajar2:~# systemctl status apache2.service
● apache2.service - The Apache HTTP Server
   Loaded: loaded (/lib/systemd/system/apache2.service; enabled; vendor preset: enabled)
   Active: active (running) since Mon 2023-12-04 03:06:15 UTC; 4min 22s ago
     Docs: https://httpd.apache.org/docs/2.4/
   Main PID: 5025 (apache2)
     Tasks: 55 (limit: 1131)
    Memory: 5.2M
   CGroup: /system.slice/apache2.service
           └─5025 /usr/sbin/apache2 -k start
             └─5027 /usr/sbin/apache2 -k start
               └─5028 /usr/sbin/apache2 -k start

Dec 04 03:06:15 belajar2 systemd[1]: Starting The Apache HTTP Server...
Dec 04 03:06:15 belajar2 apache2[5024]: AH00558: apache2: Could not reliably determine the server's fully qualified domain name, using 127.0.0.1. Set the 'ServerName' directive globally to suppress this message
Dec 04 03:06:15 belajar2 systemd[1]: Started The Apache HTTP Server.
lines 1-15/15 (END)

```

Gambar 4.3 Apache di Server 2

Gambar 4.3 merupakan *capture* versi Mysql pada server 1.

```

root@belajar1:~# mysql --version
/usr/sbin/mysqld Ver 8.0.35-0ubuntu0.20.04.1 for Linux on x86_64 ((Ubuntu))
root@belajar1:~#

```

Gambar 4.4 Mysql di Server 1

Gambar 4.5 merupakan *capture* versi Mysql di server 2

```

root@belajar2:~# mysql --version
/usr/sbin/mysqld Ver 8.0.35-0ubuntu0.20.04.1 for Linux on x86_64 ((Ubuntu))
root@belajar2:~#

```

Gambar 4.5 Mysql di Server 2

Aplikasi node.js di server 2 ditunjukkan oleh gambar 4.6.

```

root@localhost:~/node-rest-crud-api# node app.js
Node app is running on port 3000

```

Gambar 4.6 Aplikasi node.js di Server 2

Aplikasi Laravel pada server 1 ditunjukkan oleh gambar 4.7.

```

root@localhost:~/test_api_apps# php artisan serve --host=0.0.0.0 --port=8080
[INFO] Server running on [http://0.0.0.0:8080].
Press Ctrl+C to stop the server

```

Gambar 4.7 Aplikasi Laravel di Server 1

#### 4.5. Implement Test Design

Untuk pembuatan rencana pengujian pada JMeter, peneliti membuatnya sama untuk kedua *framework*, yang beda hanya IP *public* dari masing-masing *server* saja. Pembuatan rencana pengujian pada JMeter mengacu pada *endpoint* API yang akan menangani setiap permintaan, *endpoint* API tersebut dapat terlihat pada tabel 3.

Tabel 3 Endpoint API

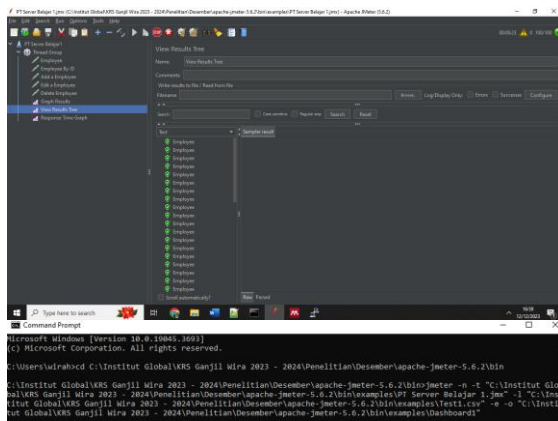
HTTP Method	URL	Keterangan
GET	/api/employee	Menampilkan semua data karyawan
GET	/api/employee/{id}	Menampilkan data karyawan sesuai id
POST	/api/employee	Menambahkan data karyawan
PUT	/api/employee/{id}	Mengubah data karyawan
DELETE	/api/employee/{id}	Menghapus data karyawan

#### 4.6. Execute Tests

Pada tahap ini dilakukan eksekusi pengujian serta memonitoring setiap hasil pengujian dari kedua *framework* baik Laravel maupun Express.js sehingga diperoleh data yang akan digunakan pada tahap selanjutnya. Untuk menjalankan rencana pengujian yang sudah disimpan perlu menuliskan *script* pada command prompt. *Script* untuk menjalankan sebuah rencana pengujian terbagi menjadi tiga perintah yaitu:

- `-n -t [test_plan_file]` berfungsi untuk mengeksekusi file *test* pada folder yang sudah ditentukan
- `-l [file_result]` berfungsi untuk menghasilkan file csv yang didalamnya berisikan hasil pengujian.
- `-e -o [report_folder]` berfungsi untuk menyimpan berkas *dashboard* hasil dari pengujian agar mudah dalam menganalisa hasil pengujian.





#### 4.7. Analyze, report, and Retest

Untuk hasil pengujian *Framework* Laravel terlihat pada tabel 4 dengan nilai rata-rata *response time* sebesar 635,17 ms. Sedangkan untuk rata-rata penggunaan CPU adalah 20% dan rata-rata penggunaan memori adalah 72%.

Tabel 4 Hasil Pengujian *Framework* Laravel

User	Respon Time (ms)	CPU Usage	Memory Usage
10	1242.75	12%	79%
20	375.68	15%	80%
30	399.38	27%	44%
40	51.83	14%	80%
50	741.02	40%	80%
60	129.69	3%	80%
70	1161.58	21%	80%
80	817.97	34%	17%
90	231.49	26%	79%
100	998.80	1%	81%
110	1136.74	16%	80%
120	335.16	25%	80%
<b>Average</b>	<b>635,17</b>	<b>20%</b>	<b>72%</b>

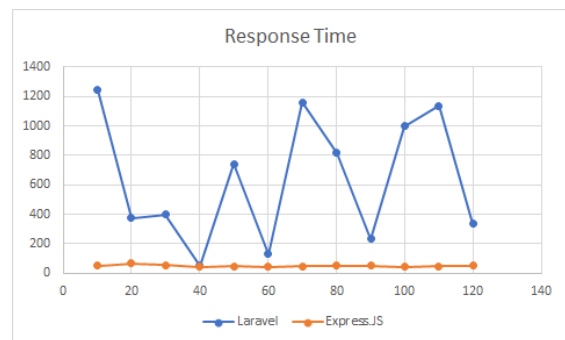
Selanjutnya untuk hasil dari pengujian *framework* Express.js dapat terlihat pada tabel 5 dengan nilai rata-rata *response time* sebesar 48,68 ms. Sedangkan untuk rata-rata penggunaan CPU adalah 17% dan rata-rata penggunaan memori adalah 63%.

Tabel 5 Hasil Pengujian *Framework* Express.js

User	Respon Time (ms)	CPU Usage	Memory Usage
10	48	0.25%	64%
20	66	0.20%	61%
30	52.60	0.22%	59%

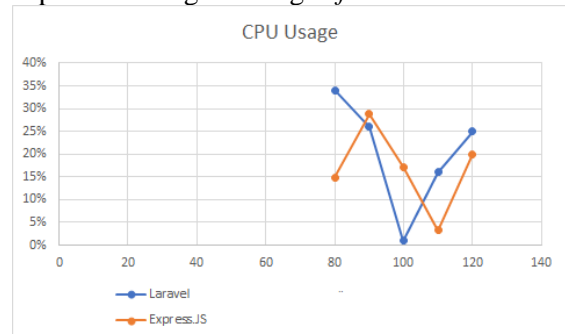
40	43.20	0.22%	61%
50	47	0.40%	61%
60	42.80	0.35%	63%
70	45.60	0.13%	63%
80	49.20	14,84%	63%
90	49.80	28,83%	64%
100	40.60	17,24%	64%
110	47.60	3,30%	64%
120	51.80	20%	64%
<b>Average</b>	<b>48.68</b>	<b>17%</b>	<b>63%</b>

Berdasarkan pengujian yang sudah dilakukan maka dapat diperoleh hasil rata rata *response time*, penggunaan CPU dan penggunaan memori dari masing masing *framework* baik Laravel maupun Express.js. Untuk menganalisisnya peneliti menguraikan hasil pengujian tersebut dalam beberapa grafik.

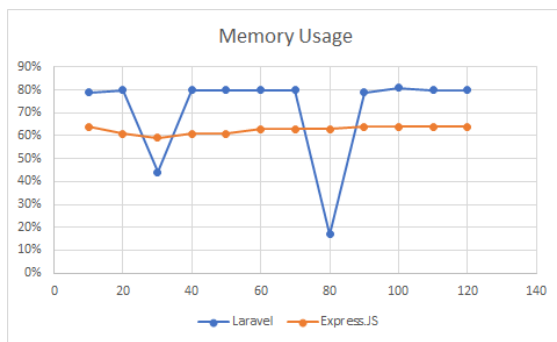


Gambar 4.8 Grafik *Response Time*

Hasil pengujian pertama adalah mengukur *response time* dari masing masing *framework*, dapat terlihat pada gambar 4 *framework* Express.js memiliki *response time* yang lebih cepat dibandingkan dengan *framework* Laravel.



Gambar 4.9 Grafik *CPU Usage*



Gambar 4.10 Grafik Memory Usage

Untuk hasil pengujian selanjutnya adalah pengujian penggunaan *resource* dari setiap *framework*, *resource* tersebut meliputi penggunaan CPU dan memori pada server. Pada grafik yang tertera pada gambar 4.9 dan gambar 1.10 terlihat bahwa *framework* Laravel menggunakan *resource* yang lebih besar dibanding *framework* Express.js. Sehingga dari semua pengujian yang sudah dilakukan diperoleh rata rata *response time*, penggunaan CPU dan memori seperti terlihat pada tabel 6.

Tabel 6 hasil rata-rata pengujian kedua *framework*

Framework	Response Time (ms)	CPU Usage	Memory Usage (%)
Laravel	635.17	20%	72%
Express.js	48.68	17%	63%

Dari hasil pengujian tersebut rata rata *response time* dari *framework* Express.js adalah 48,68 ms lebih cepat dibandingkan *framework* Laravel yang memperoleh rata rata *response time* 635,17 ms. Untuk konsumsi *resource* dengan rata rata penggunaan CPU 17% dan penggunaan memori 63% lebih sedikit dibandingkan dengan penggunaan CPU *framework* Laravel yang memiliki rata rata 20% dan penggunaan memori sebesar 72%.

Dari hasil tersebut dapat terlihat bahwa kinerja *framework* Express.js lebih cepat dibandingkan dengan *framework* Laravel, itu dibuktikan dengan *response time* yang lebih cepat. Sehingga *framework* Express.js yang berbasis Javascript ini sangat cocok diimplementasikan kepada sistem yang diakses oleh banyak pengguna. Sedangkan untuk *framework* Laravel sendiri, meskipun memiliki *response time* yang lebih lama dibandingkan dengan *framework* Express.js tapi *framework* Laravel memiliki penggunaan *resource* yang

tidak jauh lebih besar dari *framework* Express.js, sehingga *framework* Laravel dapat diimplementasikan untuk sistem yang tidak terlalu banyak user yang mengakses dan dapat diletakkan di *server* yang memiliki spesifikasi yang lebih tinggi.

## 5. KESIMPULAN

Berdasarkan hasil penelitian yang telah dilakukan peneliti dapat diambil kesimpulan sebagai berikut:

- Dengan menggunakan metode *performance testing*, sebuah teknologi dapat diuji kinerjanya sehingga diperoleh sebuah hasil pengujian yang dapat digunakan sebagai pembandingan dengan kinerja teknologi lainnya. Teknologi yang dapat diukur kinerjanya salah satunya adalah *framework* dari bahasa pemrograman PHP dan Javascript yaitu *framework* Laravel dan *framework* Express.js. Dengan menggunakan metode *performance testing* yang memiliki tujuh tahapan yaitu *Identify the Test Environment*, *Identify Performance Acceptance Criteria*, *Plan and Design Tests*, *Configure Test Environment*, *Implement Test Design*, *Execute Tests*, dan *Analyze, Report, and Retest*. Kedua *framework* dapat dilakukan perbandingan kinerjanya dengan melihat rata rata *response time* serta penggunaan CPU dan memori.
- Hasil pengujian kinerja pada *framework* Laravel dengan *framework* Express.js menunjukkan bahwa rata rata *response time* dari *framework* Express.js adalah 48,68 ms lebih cepat dibandingkan *framework* Laravel yang memperoleh rata-rata *response time* 635,17 ms. *Framework* Express.js mengonsumsi *resource* lebih sedikit dengan rata-rata penggunaan CPU 17% dan penggunaan memori 63% lebih kecil sedikit dibandingkan dengan penggunaan CPU *framework* Laravel yang memiliki rata rata 20% dan penggunaan memori sebesar 72%. Sehingga dari hasil tersebut diperoleh bahwa *framework* Express.js memiliki kinerja lebih cepat dalam menangani setiap *request* dan cocok diimplementasikan untuk sistem yang diakses oleh banyak pengguna dan diletakkan pada server dengan spesifikasi yang tinggi. Sedangkan untuk *framework* Laravel dapat diimplementasikan untuk sistem dengan pengguna tidak terlalu



banyak dan dapat diletakkan pada server dengan spesifikasi yang lebih tinggi.

Berdasarkan hasil penelitian yang telah dilakukan, terdapat beberapa saran yang dapat disampaikan untuk kepentingan penelitian selanjutnya, yaitu:

1. Untuk penelitian selanjutnya disarankan untuk mencoba *framework backend* lainnya yang dapat digunakan dalam pembuatan RESTful API dan meningkatkan lagi jumlah pengaksesan dalam proses pengujian serta menambah parameter pengujian.
2. Disarankan untuk menggunakan data lain dalam penggunaan database serta menggunakan DBMS berbeda sehingga dapat diperoleh manakah database yang sesuai untuk setiap *framework*.
3. Diharapkan hasil penelitian ini dapat menjadi pertimbangan *developer* aplikasi dalam mengembangkan sebuah sistem khususnya dalam pengembangan RESTful API agar dapat diperoleh sebuah sistem yang dapat memuaskan user baik secara kecepatan response maupun penggunaan resource.

## UCAPAN TERIMA KASIH

Terima kasih penulis ucapkan kepada rekan-rekan yang membantu terlaksananya penelitian ini. Serta dukungan dari keluarga tercinta.

## DAFTAR PUSTAKA

- [1] I. S. Sari and S. Sueb, "Hubungan Antara Perkembangan Teknologi Informasi Dan Komunikasi Terhadap Perubahan Pola Pikir Masyarakat Di Desa Kampung Sawah Kabupaten Bangkalan Madura," *J. Psikol. Jambi*, vol. 5, no. 1, pp. 12–20, 2019, doi: 10.22437/jpj.v6ijuli.11741.
- [2] L. M. Alchuluq and F. Nurzaman, "Analisis Pada Arsitektur Microservice Untuk Layanan Bisnis Toko Online," *Tekinfo J. Bid. Tek. Ind. dan Tek. Inform.*, vol. 22, no. 2, pp. 61–68, 2021, doi: 10.37817/tekinfo.v22i2.1761.
- [3] V. Tetiana, D. Sulistiyo Kusumo, and M. Andrian, "Analisis Pengaruh Pola Arsitektur Model View View Model (MVVM) terhadap Kinerja Aplikasi Mobile dengan Menerapkan Application Programming Interface (API) Covid 19," *e-Proceeding Eng.*, vol. 10, no. 3, pp. 3388–3400, 2023.
- [4] L. Mulana, K. Prihandani, A. Rizal, U. Singaperbanga, and K. Abstract, "Analisis Perbandingan Kinerja Framework Codeigniter Dengan Express.Js Pada Server RESTful Api," *J. Ilm. Wahana Pendidik.*, vol. 8, no. 16, pp. 316–326, 2022, [Online]. Available: <https://doi.org/10.5281/zenodo.7067707>
- [5] R. Choirudin and A. Adil, "Implementasi Rest Api Web Service dalam Membangun Aplikasi Multiplatform untuk Usaha Jasa," *MATRIK J. Manajemen, Tek. Inform. dan Rekayasa Komput.*, vol. 18, no. 2, pp. 284–293, 2019, doi: 10.30812/matrik.v18i2.407.
- [6] D. Ambriani and A. I. Nurhidayat, "Rancang Bangun Repository Publikasi Ilmiah Dosen Berbasis Web Menggunakan Framework Laravel," *J. Manaj. Inform.*, vol. 10, no. 01, pp. 58–66, 2020.
- [7] M. F. Minokaura, A. Afisina, A. Rahman, E. Alfa, M. A. Rizal, and N. A. Rakhmawati, "Studi Empiris Penggunaan Framework Pada Website Jaringan Dokumentasi Dan Informasi Hukum Daerah Pulau Jawa," *J. Ilmu Komun. UHO J. Penelit. Kaji. Ilmu Komun. dan Inf.*, vol. 5, no. 2, pp. 93–101, 2020.
- [8] Nasution, "Implementasi Mongo Db, Express Js, React Js Dan Node Js (Mern) Pada Pengembangan Aplikasi Formulir, Kuis, Dan Survei Online," *Informatics Eng.*, pp. 1–160, 2021, [Online]. Available: <https://dspace.uui.ac.id/handle/123456789/38607>
- [9] X. J. Hong, H. Sik Yang, and Y. H. Kim, "Performance Analysis of RESTful API and RabbitMQ for Microservice Web Application," *9th Int. Conf. Inf. Commun. Technol. Conver. ICTC 2018*, pp. 257–259, 2018, doi: 10.1109/ICTC.2018.8539409.
- [10] I. Ahmad, E. Suwarni, R. I. Borman, Asmawati, F. Rossi, and Y. Jusman, "Implementation of RESTful API Web Services Architecture in Takeaway Application Development," *2021 1st Int. Conf. Electron. Electr. Eng. Intell. Syst. ICE3IS 2021*, no. July, pp. 132–137, 2021, doi: 10.1109/ICE3IS54102.2021.9649679.
- [11] F. Doglio, *REST API Development with Node.js: Manage and Understand the Full Capabilities of Successful REST Development, Second Edition*. 2018. doi: 10.1007/978-1-4842-3715-1.
- [12] D. Aipina and H. Witriyono, "Pemanfaatan Framework Laravel dan Framework Bootstrap Pada Pembangunan Aplikasi Penjualan Hijab Berbasis Web," *J. Media Infotama*, vol. 18, no. 1, p. 2022, 2022.

- [13] B. Harijanto and Y. Ariyanto, "Pengujian Kinerja Web Server Polinema Menggunakan Alat Uji Jmeter," *J. Tek. Ilmu Dan Apl.*, vol. 2, no. 1, pp. 1–4, 2021, doi: 10.33795/jtia.v2i1.45.
- [14] M. Hendayun, A. Ginanjar, and Y. Ihsan, "Analysis of Application Performance Testing Using Load Testing and Stress Testing Methods in Api Service," *J. Sisfotek Glob.*, vol. 13, no. 1, p. 28, 2023, doi: 10.38101/sisfotek.v13i1.2656.