Jurnal Ilmiah Wahana Pendidikan, September 2022, 8 (16), 316-326
DOI: https://doi.org/10.5281/zenodo.7067707
p-ISSN: 2622-8327 e-ISSN: 2089-5364
Accredited by Directorate General of Strengthening for Research and Development
Available online at https://jurnal.peneliti.net/index.php/JIWP

Analisis Perbandingan Kinerja *Framework* Codeigniter Dengan Express.Js Pada *Server* RESTful Api

Luky Mulana[1], Kamal Prihandani[2], Adhi Rizal[3]

[1,2,3]Universitas Singaperbanga Karawang

Received: 13 Agustus 2022
Revised: 18 Agustus 2022
Accepted: 24 Agustus 2022

Abstract

Application Programming Interface (API) is an interface that is able to integrate data and connect an application that runs on many platforms so that it can be connected to each other, one of the API implementations is the RESTful API. The choice of technology in making a RESTful API is very important because it can affect the performance of the server. The Codeigniter and Express.js frameworks are two backend technologies that are used to create RESTful APIs. To determine the performance of each framework, it is necessary to test using the performance testing method to determine the response time and CPU and memory usage. The results of the test show that Express,js has an average response time of 438,64 ms faster than the Codeigniter framework which obtains an average response time of 551,72 ms, while for CPU and memory usage, the Express.js framework consumes more than the Codeigniter framework. . So Express.js is suitable for systems that are accessed by many users and are placed on servers with high specifications. While the Codeigniter framework is suitable to be implemented for systems with fewer user access and can be placed on servers with low specifications.

Keywords: *Codeigniter, Express.js, Performance Testing, RESTful API*

(*) Corresponding Author: luky.mulana18200@student.unsika.ac.id

How to Cite: Mulana, L., Prihandani, K., & Rizal, A. (2022). Analisis Perbandingan Kinerja Framework Codeigniter Dengan Express.Js Pada Server RESTful Api. *Jurnal Ilmiah Wahana Pendidikan*, *8*(16), 316-326. https://doi.org/10.5281/zenodo.7067707

PENDAHULUAN

Di era sekarang ini mulai banyaknya aplikasi yang tersedia dalam berbagai *platform* sehingga diperlukan adanya intergrasi data agar tidak terjadi duplikasi data di sebuah aplikasi yang berjalan di banyak *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.

Representional State Transfer (REST) adalah sebuah arsitektur metode komunikasi 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 pemograman dan *framework* yang tepat sehingga *server* RESTful API dapat

menangani *request* dari client dengan baik (Winardi, Abdillah, Hermanto, & Widiyanto, n.d.).

Banyak sekali bahasa pemograman dan *framework* yang dapat digunakan dalam membangun RESTful API, namun ada beberapa bahasa pemograman yang cukup populer dan banyak digunakan. Menurut Aini Rakhmawati, Haris, Hermansyah, & Furqon (2018) yang telah melakukan survei pada penggunaan web *framework* diperoleh bahwa dari 548 website pemerintahan di Indonesia, 73% tidak menggunakan *framework* sedangkan 24% menggunakan *framework* Codeigniter dan 3% menggunakan *framework* Laravel. Dari hasil survei tersebut diperoleh bahwa *framework* Codeigniter merupakan *framework* dari bahasa pemograman PHP yang paling digunakan di Indonesia khususnya pada ruang lingkup pemerintahan.

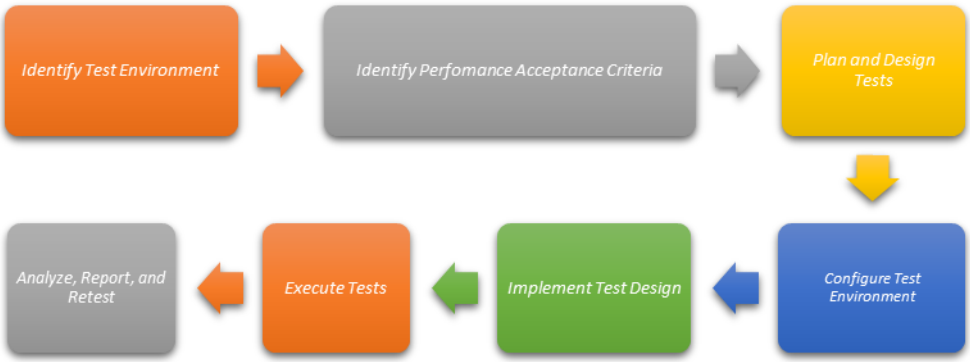
Selain bahasa pemograman PHP yang memiliki *framework* Codeigniter, bahasa pemograman yang cukup populer dan banyak digunakan adalah Javascript. Berdasarkan data dari Stack Overflow Developer Survey pada tahun 2021 bahwa dari 83.052 responden, 64.96% memilih Javascript sebagai bahasa pemograman yang sering digunakan. Javascript sendiri pada sisi *server backend* dijalankan oleh sebuah *platform* bernama Node.js dan memiliki *framework* Express.js. Pada survei yang sama yaitu Stack Overflow Developer Survey 2021, Express.js menempati peringkat ketiga dengan perolehan 23.82% dari 67.593 responden.

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* Codeigniter dengan Express.js serta menjelaskan hasil analisis dari perbandingan kinerja *framework* Codeigniter dengan Express.js menggunakan metode *perfomanace 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* Codeigniter dan Express.js dan memberikan sebuah *response* yang akan ditampilkan kepada *user* serta mengukur seberapa baik kinerja dari kedua *framework* tersebut.

METODOLOGI PENELITIAN

Metode yang dilakukan 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* sebenarnya untuk mengetahui *system* berjalan dengan baik dan memilki kinerja yang baik (H. Sarojadevi, 2011). Tujuan utama *performance testing* adalah menguji *scalability, availability* dan kinerja baik dari sisi perangkat keras maupun perangkat lunak. Performance testing akan dilakukan dalam tujuh tahapan seperti pada gambar 1.



Gambar 1. Rancangan Penelitian

1. *Identify the 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. Pada penelitian ini *server* yang akan digunakan adalah dua buah *virtual machine* dari layanan *cloud computing* Microsoft Azure. Setiap *framework* akan diletakan di dua *server* yang berbeda dengan spesifikasi yang sama dengan tujuan agar setiap *framework* dapat menangani setiap *request* tanpa adanya *sharing resource* antara *framework* Codeigniter dengan Express.js.

Untuk *tools* yang akan digunakan dalam memonitoring setiap hasil pengujian adalah JMeter. JMeter akan memberikan *output* berupa *response time* serta *error rate* dari setiap *request* terhadap masing masing *endpoint* selain itu untuk monitoring *resource* peneliti akan menggunakan data *plugin* yang terdapat pada JMeter yaitu PerfMon sehingga dapat mengetahui penggunaan CPU dan memori ketika *server* menangani setiap *request*.

Selain *server* dan *tools*, untuk penggunaan *framework* akan menggunakan *framework* Codeigniter dengan menggunakan versi 3.1 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 Enviroment*

Server	
Cloud Service	Microsoft Azure Cloud
CPU	2 Core
Memory	16 GB
HDD	30 GB
Sistem Operasi	Ubuntu Server 20.04 LTS
Web Server	Apache

Database	
DBMS	MySQL
Versi	10.2
Tools	
Aplikasi	Apache JMeter
Versi	5.4.3
Framework	
Framework 1	Codeigniter v3.1
Framework 2	Express.js v4.17

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 10000 baris data. 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 karyawan yang akan melihat datanya pada sistem.

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 *request*.

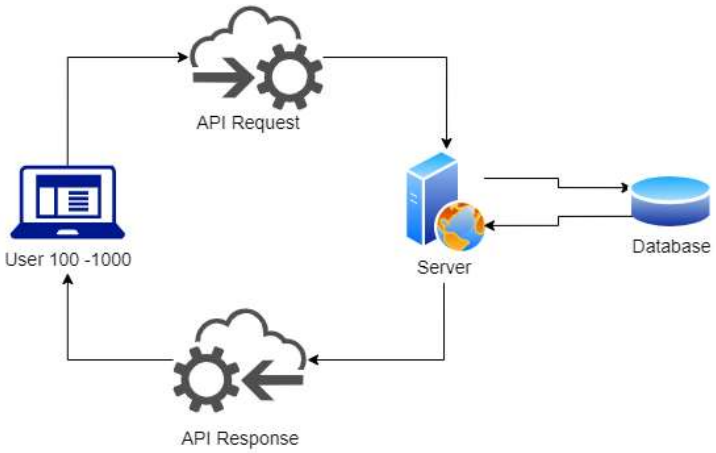
Tabel 2. Kriteria Pengujian

Performance Objective	***Criteria***
Response Time	< 5 detik ketika 1000 *user* akses
Penggunaan CPU	< 75%
Penggunaan Memori	< 75%

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. Berdasarkan objek penelitian, rencana pengujian akan dilaksanakan dengan cara melakukan *request* dari *client* menggunakan JMeter ke setiap *endpoint* yang sudah disediakan *framework* yang sudah diletakkan pada masing masing *server* menggunakan HTTP Method (GET, POST, PUT, DELETE).

Pada gambar 2 dijelaskan bagaiman alur dari pengujian yaitu *request* dilakukan secara bersamaan dengan pertambahan jumlah user dari 100 sampai 1000 *user* dengan interval setiap pengujian adalah lima menit.



Gambar 2. Alur Pengujian

4. *Configure Test Environment*

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

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 terinstall pada komputer *client* akan dilakukan pengaturan sesuai dengan rencana pengujian. Untuk pembuatan *test plan* pada JMeter akan sama untuk kedua *framework* hanya dibedakan *public* IP dari masing masing *server*. Pembuatan test plan pada JMeter mengacu pada endpoint yang akan menangani setiap *request*, *endpoint* tersebut dapat terlihat pada tabel 3.

Tabel 3. *Endpoint* API

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

6. ***Execute Test***

Pada tahap ini dilakukan eksekusi pengujian serta memonitoring setiap hasil pengujian dari kedua *framework* baik Codeigniter maupun Express.js sehingga diperoleh data yang akan digunakan pada tahap selanjutnya. Untuk menjalankan *test plan* yang sudah disimpan perlu menuliskan *script* pada command prompt. Script untuk menjalankan sebuah test plan 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.

Gambar 3. *Script* eksekusi *test plan*

7. ***Analyze Report and Retest***

Pada tahap ini dilakukan analisa dan perbandingan dari kedua *framework* 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.

HASIL DAN PEMBAHASAN

Adapun hasil penelitian setelah semua *test plan* sudah dijalankan maka akan diperoleh hasil untuk setiap *framework*. Untuk hasil pengujian untuk *framework* Codeigniter dapat terlihat pada tabel 4.

Tabel 4. Hasil pengujian *framework* Codeigniter

User	Response Time (ms)	CPU Usage	Memory Usage
100	554,47	1,18%	2,25%
200	535,27	1,32%	2,25%
300	533,84	1,31%	2,30%
400	530,04	1,27%	2,32%
500	579,06	1,43%	2,32%
600	565,95	1,31%	2,34%
700	529,15	1,47%	2,31%
800	589,04	1,45%	2,34%
900	539,28	1,26%	2,34%

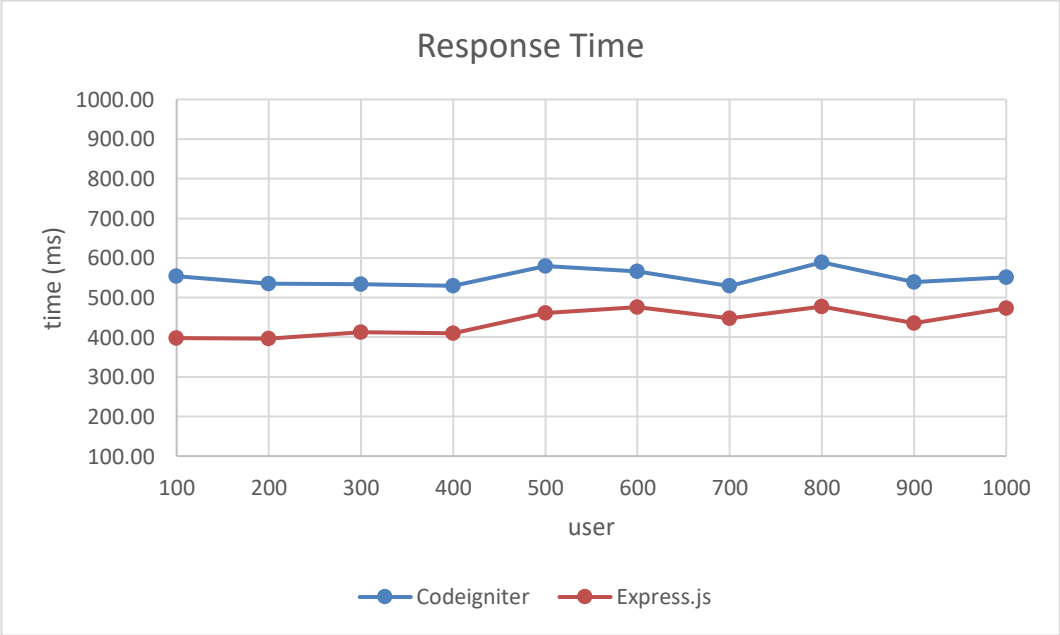
1000	551,08	1,38%	2,35%
Average	**550,72**	**1,34%**	**2,31%**

Selanjutnya untuk hasil dari pengujian *framework* Express.js dapat terlihat pada tabel 5 dengan nilai rata-rata *response time* sebesar 438,64 ms. Sedangkan untuk rata-rata penggunaan CPU adalah 1,95% dan rata-rata penggunaan memori adalah 2,74%.

Tabel 5. Hasil pengujian *framework* Express.js

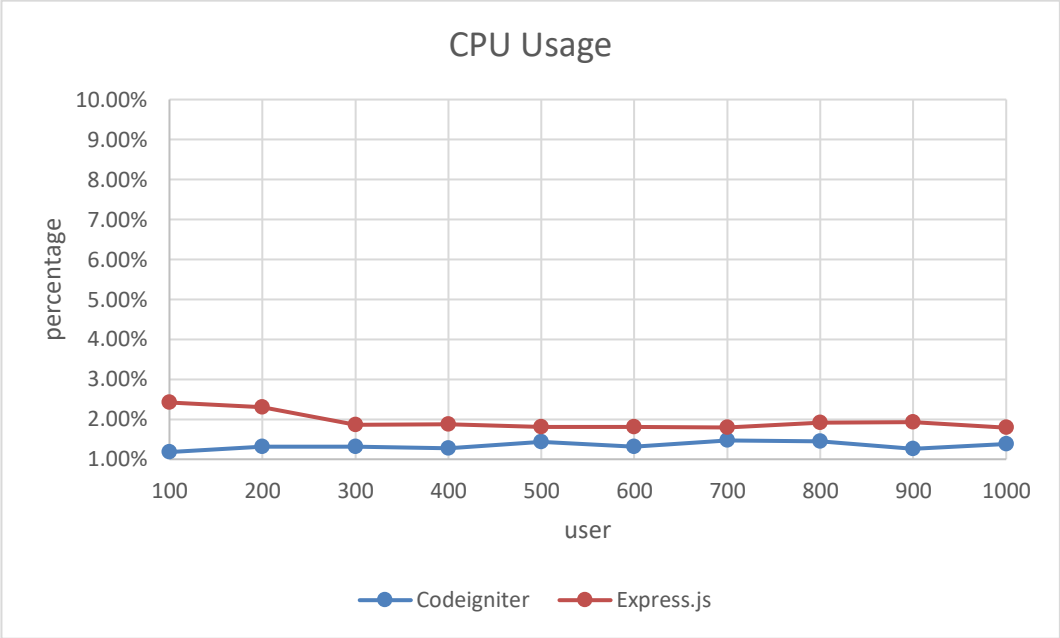
User	Response Time (ms)	CPU Usage	Memory Usage
100	397,84	2,42%	2,67%
200	396,36	2,30%	2,69%
300	412,91	1,86%	2,72%
400	409,34	1,87%	2,76%
500	460,64	1,81%	2,74%
600	476,14	1,81%	2,74%
700	447,03	1,80%	2,75%
800	477,36	1,91%	2,76%
900	436,03	1,93%	2,77%
1000	472,78	1,79%	2,76%
Average	**438,64**	**1,95%**	**2,74%**

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

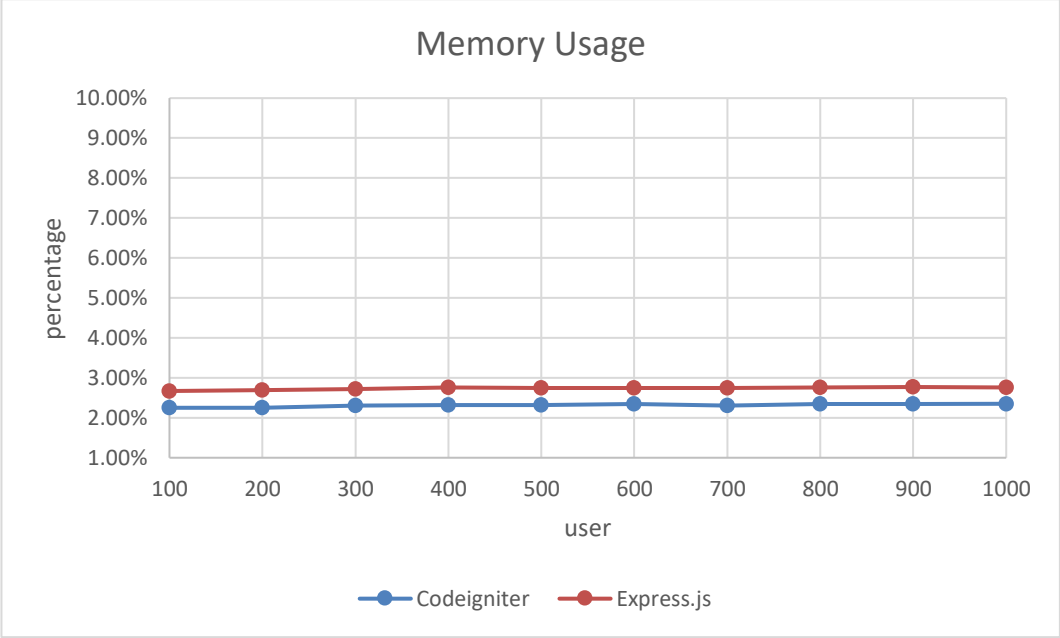


Gambar 4. Grafik *response time*

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



Gambar 5. Grafik penggunaan CPU



Gambar 6. Grafik penggunaan memori

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 5 dan gambar 6 terlihat bahwa *framework* Express.js menggunakan *resource* yang lebih besar dibanding *framework* Codeigniter.

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
Codeigniter	550,72	1,34%	2,31%
Express.js	438,64	1,95%	2,74%

Dari hasil pengujian tersebut rata rata *response time* dari *framework* Express,js adalah 438,64 ms lebih cepat dibandingkan *framework* Codeigniter yang memperoleh rata rata *response time* 550,72 ms. Tetapi disisi lain *framework* Express.js mengonsumsi *resource* lebih besar dengan rata rata penggunaan CPU 1,95% dan penggunaan memori 2,74% lebih besar sedikit dibandingkan dengan penggunaan CPU *framework* Codeigniter yang memiliki rata rata 1,34% dan penggunaan memori sebesar 2,31%.

Dari hasil tersebut dapat terlihat bahwa kinerja *framework* Express.js lebih cepat dibandingkan dengan *framework* Codeigniter, 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 *user*. Namun disisi lain penggunaan *resource* yang lebih tinggi dibanding *framework* Codeigniter membuat *framework* Express.js harus diletakkan di *server* yang memiliki spesifikasi cukup tinggi. Sedangkan untuk *framework* Codeigniter sendiri, meskipun memiliki *response time* yang lebih lama dibandingkan dengan *framework* Express.js tapi *framework* Codeigniter memiki penggunaan *resource* yang lebih kecil, sehingga *framework* Codeigniter dapat diimplementasikan untuk sistem yang tidak terlalu banyak *user* yang mengakses dan dapat diletakkan di *server* yang memiliki spesifikasi yang tidak terlalu tinggi.

KESIMPULAN

Berdasarkan hasil penelitian yang telah dilakukan dapat diambil kesimpulan sebagai berikut.

1. Dengan menggunakan metode *performance testing*, sebuah teknologi dapat diuji kinerjanya sehingga diperoleh sebuah hasil pengujian yang dapat digunakan sebagai pembanding dengan kinerja teknologi lainnya. Teknologi yang dapat diukur kinerjanya salah satunya adalah *framework* dari bahasa pemograman PHP dan Javascript yaitu *framework* Codeigniter dan *framework* Express.js. Dengan menggunakan metode *performance testing* yang memiliki tujuh tahapan yaitu *Identify the Test Environment, Identify Perfomance 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.
2. Hasil pengujian kinerja pada *framework* Codeigniter dengan *framework* Express.js menunjukan bahwa rata rata *response time* dari *framework* Express,js adalah 420,72 ms lebih cepat dibandingkan *framework* Codeigniter yang memperoleh rata rata *response time* 555,90 ms. Tetapi disisi

lain *framework* Express.js mengonsumsi *resource* lebih besar dengan rata rata penggunaan CPU 1,95% dan penggunaan memori 2,74% lebih besar sedikit dibandingkan dengan penggunaan CPU *framework* Codeigniter yang memiliki rata rata 1,34% dan penggunaan memori sebesar 2,31%. 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 *user* dan diletakkan pada *server* dengan spesifikasi yang tinggi. Sedangkan untuk *framework* Codeigniter dapat diimplementasikan untuk sistem dengan *user* tidak terlalu banyak dan dapat diletakkan pada *server* dengan spesifikasi yang tidak terlalu tinggi.

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

1. Untuk penelitian selanjutnya disarankan untuk mencoba *framework backend* lainnya yang dapat diguanakan dalam pembuatan RESTful API dan meningkatkan lagi jumlah pengaksesan dalam proeses 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* dalam mengembanganken sebuah sistem khususnya dalam pengembangan RESTful API agar dapat diperoleh sebuah sistem yang dapat memuaskan *user* baik secara kecepatan *response* maupun penggunaan *resource*.

DAFTAR PUSTAKA

Adani, M. R. (2020, August 7). *Framework Adalah: Pengertian, Fungsi Dan Jenis Untuk web dev*. Sekawan Media | Software House & System Integrator Indonesia. Retrieved March 18, 2022, from https://www.sekawanmedia.co.id/blog/pengertian-framework/

Aini Rakhmawati, N., Haris, S., Hermansyah, D., & Furqon, A. (2018). *A survey of Web Technologies used in Indonesia Local Governments* (Vol. 07).

Ariffudin, M. (2021, December 20). *Mengenal Express.js: Pengertian, Cara Kerja, Keunggulan, tutorial*. Niagahoster Blog. Retrieved March 20, 2022, from https://www.niagahoster.co.id/blog/express-js-adalah/

Belajar cepat framework codelgniter Untuk Pemula. (n.d.). Retrieved March 20, 2022, from https://idcloudhost.com/wp-content/uploads/2017/08/Panduan-Belajar-Cepat-Framework-Codeigniter-untuk-Pemula-IDCloudHost-232x300.pdf

Chatterjee, S. (2020). A comparative study on SOAP and RESTful web services. *International Research Journal of Engineering and Technology*. www.irjet.net

Dhalla, H. K. (2021). A Performance Comparison of RESTful Applications Implemented in Spring Boot Java and MS.NET Core. *Journal of Physics: Conference Series*, *1933*(1). IOP Publishing Ltd. https://doi.org/10.1088/1742-6596/1933/1/012041

Decker, A., & Sarojadevi, H. (2011). *1.Performance Testing Methodologies and Tools*. www.iiste.org

Doglio, F. (2018). REST API Development with Node.js. In *REST API Development with Node.js*. Apress. https://doi.org/10.1007/978-1-4842-3715-1

Fielding, R. T. (2000). REST: architectural styles and the design of network-based software architectures. Doctoral dissertation, University of California.

Gilchrist, A. (2015). *API Design and Control for DevOps*.

Halili, E. H. (2008). *Apache JMeter : a practical beginner's guide to automated testing and performance measurement for your websites*. Packt Pub.

Karlsson, O. (2021). A Performance comparison Between ASP.NET Core and Express.js for creating Web APIs (Dissertation). Retrieved from http://urn.kb.se/resolve?urn=urn:nbn:se:hj:diva-54286

Khan, R. B. (2016). *Comparative Study of Performance Testing Tools: Apache JMeter and HP LoadRunner*. www.bth.se

Lawi, A., Panggabean, B. L. E., & Yoshida, T. (2021). Evaluating graphql and rest api services performance in a massive and intensive accessible information system. *Computers*, *10*(11). https://doi.org/10.3390/computers10110138

Matam, S., & Jain, J. (2017). Pro Apache JMeter: web application performance testing. Apress.

Meier, J. D., Farre, C., Prashant, B., Scott, B., & Dennis Rea. (2007). *Performance Testing Guidance for Web Applications*

Molyneaux, I. (2014). The art of application performance testing: from strategy to tools. " O'Reilly Media, Inc.".

Rick, B. L. (2016). *Express.js: Guide Book on Web framework for Node.js*.

Rompis, A. C., & Aji, R. F. (2018). Perbandingan Performa Kinerja Node. js, PHP, dan Python dalam Aplikasi REST. CogITo SMart Journal, 4(1), 171-187.

Santoro, Mattia., Vaccari, Lorenzino., Mavridis, Dimitrios., Smith, Robin., Posada, Monica., Gattwinkel, Dietmar., & European Commission. Joint Research Centre. (n.d.). *Web Application Programming Interfaces (APIs) : general-purpose standards, terms and European Commission initiatives.*

Setiawan, R. (2021, December 15). *APA ITU framework? developer Wajib Tahu*. Dicoding Blog. Retrieved March 18, 2022, from https://www.dicoding.com/blog/apa-itu-framework/

Stack Overflow Developer Survey 2021. (2021). Stack Overflow. https://insights.stackoverflow.com/survey/2021#technology-most-popular-technologies

Warvante, M. S. (2007). IJARCCE A Survey on Developing Web Services with SOAP and REST. *International Journal of Advanced Research in Computer and Communication Engineering ISO*. https://doi.org/10.17148/IJARCCE.2017.61020

Winardi, P., Abdillah, M. I., Hermanto, D., & Widiyanto, E. P. (n.d.). Analisis Perbandingan Performa Web Server Berbasis PHP dan Web Server Berbasis Python Twisted. *Julyxxxx*, *x*, *No.x*, 1–5