BAB II

TINJAUAN PUSTAKA DAN DASAR TEORI

2.1. Tinjauan Pustaka

Penelitian yang membandingkan Yii dengan *Codeigniter* untuk developer web dilakukan oleh Darma Aji, Muhammad (2017), hasil penelitian tersebut *framework* memiliki kelebihan dan kekurangan, dimana perbandingan ini menunjukan bahwa Yii *framework* dan *Codeigniter* tidak memliki banyak perbedaan, secara umum hanya ada beberapa perbedaan yang terletak pada aturan-aturan yang berlaku atau yang diterapkan antar *framework*.

Rulli Erinton dkk (2017), membandingkan *Codeigniter* dan *Laravel* menggunakan web *server* Apache, hasil penelitian tersebut menunjukan bahwa aplikasi web yang menggunakan *framework Codeigniter* lebih baik jika dilihat dari sisi performasinya dibandingkan dengan aplikasi web yang menggunakan *framework Laravel*, serta *Laravel* memiliki nilai error tertinggi yakni sebesar 79.7.

Kemudian penelitian serupa dilakukan oleh Hamid, Muhammad Nur (2019), yang membandingkan *framework Codeigniter* dan *Laravel*. Dari hasil penelitian tersebut menunjukan bahwa dari segi performa secara keseluruhan, *Codeigniter* memiliki nilai yang lebih unggul. Namun baik *framework Codeigniter* maupun *framework Laravel* memiliki penerapan yang sama karena ditempatkan pada sisi klien.

Sedangkan dari penelitian dari Rio Renaldo, Hendi Sama (2020), terkait studi perbandingan untuk pengembangan *website* dengan *framework Codeigniter* dan *Laravel*, menunjukan bahwa *framework Codeigniter* memiliki durasi time yang lebih besar daripada *Laravel* dengan selisih sebesar 4,2 ms dan pada nilai rata-rata speed atau kecepatannya, *Codeigniter* masih lebih unggul dengan selisih 40,08 Kbit/s dari *Laravel*.

Perbedaan antara penelitian sebelumnya dengan penelitian yang dilakukan yaitu dari *framework* yang digunakan, pada penelitian kali ini dilakukan analisis perbandingan antara *framework Laravel* versi 9.0 dengan *framework Codeigniter* versi 4.0. Metode perbandingan performa dilakukan dengan mengukur *request per second* dan *response time*. Untuk metode perbandingan cara akses *database* adalah dengan membandingkan serta menganalisa akses *database* untuk tata cara pengaksesan table dalam operasi Create, Read, Update, Delete (CRUD). Perbandingan fitur unggulan dilihat dari keunggulan fitur yang dimiliki.

Tabel 2.1 Perbandingan Tinjauan Pustaka

Penulis	**Objek**	**Masalah**	**Teknologi**	**Hasil**
Darma Aji, Muhammad (2017)	Perbandingan antara *framework* Yii dengan *Codeigniter* untuk developer web	Membandingkan segi koneksi ke database, implementasi CRUD serta penambahan source code untuk mengukur kecepata akses halaman antar *framework*	*Framework* Yii dan *Framework Codeigniter*	Yii *Framework* dan *Codeigniter Framework* tidak berbeda jauh, secara umum perbedaannya hanya terletak pada aturan-aturan yang berlaku atau yang ditetapkan antar *framework*
Ruli Erinton dkk (2017)	Perbandingan antara *Codeigniter* dan *Laravel* menggunakan web *server* Apache	Melakukan pengujian Load Test dan Stress Test dengan menggunaan bantuan software web stress tool.	*Framework Codeigniter* dan *Framework Laravel*	Hasil penelitian tersebut menunjukan bahwa aplikasi web yang menggunakan *framework Codeigniter* lebih baik dari sisi performasiya dibandingkan dengan aplikasi

				web yang menggunakan *framework Laravel*
Hamid, Muhammad Nur (2019)	Perbandingan *Framework Codeigniter* dan *Framework Laravel*	Melakukan perbandingan pada performa, cara akses database dan implemtasi fitur AJAX	*Framework Codeigniter* dan *Framework Laravel*	Hasil penelitian tersebut menunjukan bahwa dari segi performa secara keseluruhan, *Codeigniter* memiliki nilai yang lebih unggul
Rio Renaldo, Hendi Sama (2020)	Studi Komparasi Pengembangan *Website* dengan *Framework Codeigniter* dan *Laravel*	Melakukan pengujian kedua *framework* dengan menggunakan parameter yang menjadi landasan analisis yaitu Data, Time dan Speed	*Framework Codeigniter* dan *Framework Laravel*	Hasil penelitian tersebut menunjukan *framework Codeigniter* memiliki durasi tume yang lebih besar daripada *Laravel* dengan selisih sebesar 4,2 ms dan pada nilai rata-rata speed, *Codeigniter* masih lebih unggul dengan selisih 40,08 Kbit/s dari *Laravel*
Yang diusulkan	Analisis perbandingan antara *Framework*	Melakukan perbandingan kedua *framework* dilihat dari	*Framework Codeigniter* 4 dan	

	Codeigniter dengan *Framework Laravel*	performa aplikasi, koneksi ke database mysql, fitur unggulan dan implementasi CRUD	*Framework Laravel* 9	

2.2. Dasar Teori

2.2.1. Framework

Framework atau dalam bahasa Indonesia "kerangka kerja" adalah kumpulan fungsi-fungsi, prosedur-prosedur, dan kelas-kelas yang telah siap digunakan untuk membantu dan mempercepat pekerjaan seorang programmer. Dengan menggunakan *framework*, programmer tidak perlu membangun fungsi-fungsi dan kelas-kelas dasar dari awal, melainkan dapat memanfaatkan komponen-komponen yang sudah ada dalam *framework*. Hal ini memungkinkan programmer untuk fokus pada logika bisnis yang spesifik dalam pembangunan aplikasi, sehingga meningkatkan produktivitas dan konsistensi dalam pengembangan perangkat lunak (Bin Tahir et al., 2019:56).

Sementara itu menurut Hakim (2010:3), *framework* adalah "sekumpulan atau kumpulan potongan-potongan program yang disusun atau diatur sedemikian rupa, sehingga dapat digunakan untuk membantu membuat aplikasi menjadi lengkap tanpa harus membuat semua kode dari awal." Selanjutnya, Raharjo (2015:2) menyebutkan bahwa *Web Application Framework* (WAF) atau sering disingkat web *framework* adalah kumpulan kode berupa library dan tools yang digabungkan sedemikian rupa menjadi satu *framework* (kerangka kerja) untuk memudahkan dan mempercepat proses pembangunan web. aplikasi".

Berdasarkan pemaparan diatas dapat disimpulkan bahwa, *Framework* adalah kumpulan fungsi-fungsi, prosedur-prosedur, dan kelas-kelas yang telah tersusun sedemikian rupa untuk membantu programmer dalam membangun aplikasi dengan lebih efisien. Dengan memanfaatkan komponen-komponen yang sudah ada dalam *framework*, programmer dapat menghindari pembangunan dari awal dan fokus pada pengembangan logika bisnis yang spesifik. *Framework* membantu meningkatkan produktivitas, konsistensi, dan mempercepat proses pengembangan perangkat lunak dengan menyediakan struktur, aturan, dan alat bantu yang teruji.

Salah satu alasan utama mengapa orang menggunakan *framework* dalam membangun aplikasi adalah kenyamanan yang ditawarkannya. Dengan menggunakan *framework*, struktur aplikasi yang baik, pola desain (*design pattern*), *common function*, dan banyak lagi dapat disediakan secara bawaan. Dengan demikian, pengembang dapat fokus pada proses bisnis yang sedang dihadapi tanpa harus khawatir tentang masalah-masalah lain seperti struktur aplikasi. *Framework* memungkinkan para pengembang untuk lebih efisien dan produktif karena mereka dapat mengambil keuntungan dari fitur-fitur yang sudah ada dan teruji dalam *framework* tersebut.Adapun beberapa alasan menggunakan *framework* menurut Koespradono et al (2014:16) antara lain sebagai berikut:

a. Mempercepat dan mempermudah pengerjaan sebuah aplikasi web.

Salah satu alasan utama menggunakan *framework* dalam pengembangan aplikasi web adalah kemampuannya untuk mempercepat dan mempermudah proses pengerjaan. Dengan menyediakan komponen-komponen yang sudah siap pakai, seperti fungsi-fungsi, prosedur-prosedur, dan kelas-kelas, *framework* menghilangkan kebutuhan untuk membangun semuanya dari awal. Pengembang dapat menghemat waktu dengan menggunakan fitur-fitur yang sudah ada dalam *framework*, seperti manajemen database, autentikasi pengguna, validasi input, dan lainnya. Selain itu, *framework* juga menyediakan struktur yang terdefinisi dengan baik, aturan-aturan, dan pola desain yang dapat diikuti, sehingga memudahkan pengembang dalam mengorganisasi kode dan meminimalkan kesalahan. Dengan demikian, pengembang dapat fokus pada pengembangan logika bisnis khusus aplikasi, meningkatkan produktivitas, dan menghasilkan aplikasi yang lebih konsisten dan berkualitas.

b. Mempermudah proses *maintenance* karena sudah ada pola tertentu dalam sebuah *framework.*

Penggunaan *framework* dalam pengembangan aplikasi web juga mempermudah proses pemeliharaan (*maintenance*) aplikasi. Hal ini dikarenakan *framework* sudah memiliki pola tertentu yang konsisten dan teruji. Dalam sebuah *framework*, seringkali digunakan pola desain yang sudah mapan, seperti *Model View Controller* (MVC) atau pola lainnya. Dengan menggunakan pola-pola ini, struktur aplikasi menjadi lebih terorganisir dan terstruktur, sehingga memudahkan pengembang dalam memahami dan

mengelola kode. Ketika terjadi perubahan atau penambahan fitur, pengembang dapat dengan mudah menemukan tempat yang tepat dalam struktur *framework* untuk melakukan modifikasi atau penyesuaian tanpa harus merombak seluruh kode. Selain itu, dokumentasi dan sumber daya yang melimpah yang tersedia untuk *framework* juga mempermudah proses maintenance, karena pengembang dapat dengan mudah mencari solusi atau referensi saat menghadapi masalah atau perubahan dalam aplikasi. Dengan demikian, penggunaan *framework* dapat meningkatkan efisiensi dalam proses maintenance aplikasi web.

c. Umumnya *framework* menyediakan fasilitas-fasilitas yang umum dipakai sehingga tidak perlu membangun dari awal (misal validasi, *pagination, session*, dll).

Framework umumnya dilengkapi dengan berbagai fasilitas yang sering digunakan dalam pengembangan aplikasi, seperti validasi data, pengaturan sesi (*session*), sistem pengaturan halaman *(pagination),* keamanan, dan banyak lagi. Dengan adanya fasilitas-fasilitas ini, pengembang tidak perlu membangun semuanya dari awal, menghemat waktu dan usaha yang diperlukan. Penggunaan fasilitas yang telah disediakan oleh *framework* juga memberikan keuntungan dalam hal kualitas dan keandalan, karena fasilitas-fasilitas tersebut telah diuji dan digunakan oleh banyak pengembang sebelumnya. Selain itu, *framework* juga menyediakan dokumentasi yang baik untuk penggunaan fasilitas-fasilitas ini, sehingga pengembang dapat dengan mudah memahami dan mengimplementasikannya dalam aplikasi mereka. Secara keseluruhan, adanya fasilitas-fasilitas yang umum digunakan dalam *framework* mempermudah dan mempercepat proses pengembangan aplikasi web.

Berdasarkan pemaparan diatas dapat disimpulkan bahwa *framework* dapat mempercepat dan mempermudah pengerjaan aplikasi dengan menyediakan komponen-komponen yang siap pakai. Fasilitas-fasilitas yang umum digunakan dalam *framework* juga menghilangkan kebutuhan untuk membangun fitur dari awal, sehingga menghemat waktu dan usaha pengembang. Selain itu, *framework* menyediakan pola desain yang teruji dan struktur yang terdefinisi, memudahkan proses pemeliharaan dan penyesuaian aplikasi. Adanya dokumentasi yang baik dalam *framework* juga membantu pengembang dalam memahami dan mengimplementasikan fasilitas-fasilitas yang disediakan. Secara

keseluruhan, penggunaan *framework* meningkatkan produktivitas, kualitas, dan efisiensi dalam pengembangan dan pemeliharaan aplikasi web.

Selanjutnya adapun beberapa keuntungan yang dapat diperoleh dari penggunaan *framework* ialah sebagai berikut (Daqiqil, 2011:1):

1. Menghemat Waktu Pengembangan

   Keuntungan pertama dari penggunaan *framework* dalam pengembangan perangkat lunak adalah kemampuannya untuk menghemat waktu. Dalam *framework*, telah disediakan berbagai komponen dan fitur yang siap pakai, seperti fungsi-fungsi, kelas-kelas, dan modul-modul yang umum digunakan. Dengan menggunakan komponen-komponen ini, pengembang tidak perlu menghabiskan waktu untuk membangun semuanya dari awal. Sebagai contoh, jika aplikasi membutuhkan fitur autentikasi pengguna, *framework* biasanya telah menyediakan sistem autentikasi yang dapat langsung digunakan. Dengan demikian, pengembang dapat fokus pada pengembangan fitur-fitur khusus yang unik untuk aplikasi mereka, menghemat waktu dan usaha yang seharusnya diperlukan untuk membangun komponen dasar dari nol. Dalam dunia yang kompetitif, penghematan waktu pengembangan ini dapat memberikan keunggulan bagi pengembang dalam merilis aplikasi dengan cepat ke pasar.

2. *Refuse of code*

   Keuntungan kedua dari penggunaan *framework* adalah kemampuannya untuk mengurangi penulisan kode yang repetitif. Dalam pengembangan perangkat lunak, seringkali terdapat bagian-bagian kode yang memiliki pola atau logika yang sama di berbagai bagian aplikasi. Dengan menggunakan *framework*, pengembang dapat memanfaatkan fitur-fitur yang sudah tersedia untuk mengatasi masalah ini. *Framework* menyediakan berbagai fungsi dan komponen yang dapat digunakan secara langsung, seperti modul-modul atau library-library yang umum digunakan. Sebagai contoh, jika aplikasi membutuhkan operasi CRUD (*Create, Read, Update, Delete*) pada entitas database, *framework* biasanya menyediakan fitur-fitur untuk melakukan operasi tersebut dengan mudah. Dengan menggunakan fitur-fitur tersebut, pengembang tidak perlu menulis kode dari awal untuk setiap operasi CRUD, sehingga menghemat waktu dan mengurangi jumlah kode yang perlu ditulis. Dengan demikian, penggunaan *framework*

dapat meningkatkan efisiensi dalam pengembangan perangkat lunak dengan mengurangi jumlah kode yang repetitif yang harus ditulis oleh pengembang.

3. Bantuan Komunitas

Keuntungan ketiga menggunakan *framework* adalah adanya bantuan dari komunitas pengembang yang menggunakan *framework* tersebut. Komunitas ini menyediakan platform diskusi, forum, blog, dan sumber daya lainnya yang membantu pengembang dalam berbagi pengetahuan, menyelesaikan masalah, dan mempelajari praktik terbaik dalam penggunaan *framework*. Selain itu, komunitas pengembang juga berperan dalam mengembangkan dan meningkatkan *framework*, sehingga pengguna dapat memanfaatkan pembaruan dan perbaikan terbaru yang dilakukan oleh komunitas. Dengan adanya dukungan komunitas, pengembang dapat meningkatkan efisiensi dan kualitas dalam pengembangan aplikasi menggunakan *framework*.

4. Kumpulan *Best Practice*

Keuntungan keempat menggunakan *framework* adalah adanya kumpulan *best practice* (praktik terbaik) yang disertakan dalam *framework* tersebut. *Framework* umumnya dibangun berdasarkan pengalaman dan pengetahuan terbaik dalam pengembangan perangkat lunak. Mereka menerapkan pola desain, arsitektur, dan metodologi yang telah terbukti efektif dalam membangun aplikasi yang stabil, skalabel, dan mudah dipelihara. Dengan menggunakan *framework*, pengembang dapat mengadopsi praktik terbaik ini secara langsung, tanpa perlu mencari tahu atau menguji sendiri metode yang paling efisien. Praktik terbaik yang disertakan dalam *framework* mencakup berbagai aspek seperti struktur proyek, manajemen dependensi, penggunaan desain berorientasi objek, pengujian unit, dokumentasi, dan banyak lagi. Dengan mengikuti praktik terbaik yang telah ditetapkan dalam *framework*, pengembang dapat membangun aplikasi dengan kualitas yang lebih baik, meningkatkan efisiensi pengembangan, dan meminimalisir budget.

Berdasarkan pemaparan yang dipaparkan diatas dapat disimpulkan bahwa *framework* merupakan kumpulan fungsi-fungsi, prosedur-prosedur, dan kelas-kelas yang telah tersusun sedemikian rupa untuk membantu pengembang dalam membangun aplikasi dengan lebih efisien. Dengan memanfaatkan komponen-komponen yang sudah

ada dalam *framework*, pengembang dapat menghemat waktu, mengurangi penulisan kode yang repetitif, dan meningkatkan efisiensi dalam pengembangan perangkat lunak. Selain itu, *framework* juga memberikan manfaat melalui dukungan komunitas pengembang dan adanya kumpulan praktik terbaik yang telah teruji. Dengan mengadopsi praktik terbaik dalam penggunaan *framework*, pengembang dapat membangun aplikasi dengan kualitas yang lebih baik dan meningkatkan produktivitas dalam pengembangan perangkat lunak. Dengan demikian, penggunaan *framework* menjadi pilihan yang baik bagi pengembang untuk mempercepat, memudahkan, dan meningkatkan kualitas pengembangan aplikasi. Adapun dalam pengembangan perangkat lunak, terdapat dua *framework* yang populer yaitu *CodeIgniter* dan *Laravel*. *CodeIgniter* adalah sebuah *framework* PHP yang ringan dan mudah dipelajari, sementara *Laravel* merupakan *framework* PHP yang kuat dan komprehensif. Kedua *framework* ini menyediakan berbagai fitur dan kemudahan dalam membangun aplikasi web dengan efisiensi dan kualitas yang tinggi. Untuk lebih jelasnya akan dijelaskan sebagai berikut.

2.2.1.1. Framework Codeigniter

Menurut (Betha, 2012:1), *Codeigniter* adalah *framework* pengembangan aplikasi suatu kerangka untuk bekerja atau membuat program dengan menggunakan PHP yang lebih sistematis. Pemogram tidak perlu membuat program dari awal (*from scratch*), karena CI menyediakan sekumpulan *library* yang banyak diperlukan untuk menyelesaikan pekerjaan yang umum, dengan menggunakan antarmuka dan struktur logika yang sederhana untuk mengakses librarinya.

Sementara itu menurut Basuki dikutip dari (Anggraini et al., 2020:66), *framework codeigniter* adalah sebuah *framework* PHP yang dapat membantu mempercepat pengembang dalam pengembangan aplikasi web berbasis PHP. Berdasarkan pemaparan diatas dapat disimpulkan bahwa *CodeIgniter* adalah sebuah *framework* pengembangan aplikasi yang menggunakan PHP. *Framework* ini memberikan kemudahan bagi pengembang dengan menyediakan sekumpulan library yang umum diperlukan dalam pengembangan aplikasi. Dengan menggunakan *CodeIgniter*, pengembang tidak perlu membangun program dari awal, karena *framework* ini telah menyediakan struktur logika

yang sederhana dan antarmuka yang memudahkan akses ke *library-library* yang tersedia. *CodeIgniter* membantu meningkatkan efisiensi dan kecepatan dalam pengembangan aplikasi web berbasis PHP.

Codeigniter dirancang oleh Rick Ellis, pendiri EllisLab (www.ellislab.com) pada tahun 2006, dia menciptakan *Web Application Framework* (WAF) yang dikenal sebagai *CodeIgniter* dengan tujuan untuk memudahkan pengembang web membuat aplikasi yang dijalankan di internet. *Codeigniter* adalah kumpulan kode yang diatur sebagai *framework* dan hadir dalam bentuk *library* dan *tools*. Kumpulan kode ini terintegrasi bersama sedemikian rupa sehingga menjadi *framework*. Sejak 2014 dan berlanjut hingga saat ini, EllisLab telah menyerahkan hak kepemilikan *CodeIgniter* kepada *British Columbia Institute of Technology* (BCIT), yang akan bertanggung jawab atas pengembangan lanjutannya. *Codeigniter* memberi pengembang PHP berbagai fitur dan fasilitas yang memungkinkan mereka membuat aplikasi web dengan cepat dan mudah. Struktur *Codeigniter* tidak terlalu rumit, dan lebih fleksibel (Setiadi, 2018:1).

Dengan ini, pengembang tetap memiliki kebebasan untuk menulis bagian kode tertentu dalam aplikasi menggunakan metode konvensional (tanpa *framework*) meskipun memiliki opsi untuk menggunakan *Codeigniter* secara keseluruhan atau hanya sebagian dari *framework*. Pengembang web memiliki opsi untuk menggunakan *Codeigniter* secara keseluruhan atau hanya sebagian saja.

CodeIgniter mengadopsi pola atau arsitektur desain *Model View Controller* (MVC) yang memisahkan secara jelas bagian kode yang menangani proses bisnis dari bagian kode yang menangani tampilan. Hal ini memungkinkan pengembang web untuk bekerja secara terpisah dan berkolaborasi dalam pembuatan aplikasi berbasis web secara tim. Dengan pola ini, pengembang web dapat fokus pada tanggung jawab masing-masing, misalnya, pengembang model dapat fokus pada akses dan manipulasi data, pengembang view dapat fokus pada tampilan antarmuka pengguna, dan pengembang controller dapat mengatur alur logika bisnis (Setiadi, 2018:2).

Selain pola MVC, *CodeIgniter* juga menawarkan struktur aplikasi yang terstruktur dengan baik. Struktur folder yang jelas memungkinkan pengembang untuk dengan mudah menemukan dan mengelola file-file terkait, seperti *model, view, controller*,

konfigurasi, dan sumber daya lainnya. Ini membantu meningkatkan efisiensi pengembangan dan memastikan konsistensi dalam pengaturan proyek Setiadi, 2018:4).

Dalam *CodeIgniter*, *controller* berperan sebagai penghubung antara permintaan pengguna dan logika bisnis aplikasi. *Controller* mengelola alur kerja aplikasi dan mengarahkan permintaan HTTP ke tindakan yang sesuai. Model, di sisi lain, bertanggung jawab untuk mengelola akses dan manipulasi data. Dengan adanya model, pengembang dapat dengan mudah berinteraksi dengan database atau sumber data lainnya. Sedangkan *view* bertanggung jawab untuk menampilkan data kepada pengguna dalam bentuk antarmuka pengguna yang menarik.

Dengan mengadopsi pola dan struktur aplikasi yang terorganisir ini, *CodeIgniter* membantu mempercepat pengembangan aplikasi web. Setiap anggota tim pengembang dapat fokus pada tugas mereka masing-masing tanpa mengganggu aspek lain dari situs. Hal ini juga memungkinkan pengembangan aplikasi yang lebih efisien dan penyelesaian yang lebih cepat.

Adapun fitur-fitur dari *CodeIgniter* (Sukamdani et al., 2012:19-20):

1. *Model View Controller Based*: *CodeIgniter* mengadopsi pola desain *Model View Controller* (MVC) yang memisahkan logika bisnis (*Model*), tampilan (*View*), dan pengendalian (*Controller*) dalam aplikasi. Ini membantu pengembang dalam memisahkan tanggung jawab dan meningkatkan keterbacaan kode.
2. *Extremely Lightweight*: *CodeIgniter* dirancang dengan struktur yang ringan dan sederhana, sehingga memudahkan pengembang dalam mempelajari, mengelola, dan mengimplementasikan proyek tanpa kompleksitas yang berlebihan.
3. *Full Featured Database Classes with Support for Several Platforms*: *CodeIgniter* menyediakan kelas-kelas database yang lengkap dengan dukungan untuk berbagai platform database. Ini memungkinkan pengembang untuk dengan mudah berinteraksi dengan database tanpa harus menulis kueri SQL mentah.

4. *Active Record Database Support*: *CodeIgniter* menyediakan dukungan *Active Record*, yang mempermudah pengembang dalam membangun dan mengeksekusi kueri database dengan sintaks yang sederhana dan intuitif.
5. *Form and Validation*: *CodeIgniter* memiliki fitur built-in untuk pembuatan form dan validasi data. Pengembang dapat dengan mudah membuat *form* HTML dan melakukan validasi data yang diterima dari pengguna sebelum diproses lebih lanjut.
6. *Security and XSS Filtering*: *CodeIgniter* memiliki fitur keamanan terintegrasi, termasuk filter *cross-site scripting* (XSS) yang membantu melindungi aplikasi dari serangan keamanan umum.
7. *Session Management*: *CodeIgniter* menyediakan fitur pengelolaan sesi yang memungkinkan pengembang untuk mengelola data sesi pengguna dengan mudah.
8. *Email Sending Class*: *CodeIgniter* menyediakan kelas untuk pengiriman email, termasuk dukungan untuk lampiran, *email* HTML/teks, dan protokol yang berbeda seperti *Sedmail*, SMTP, dan *Mail*.
9. *Pagination*: *CodeIgniter* memiliki fitur untuk mempermudah pembuatan tautan halaman (*pagination*) pada data yang panjang, sehingga memudahkan navigasi dan tampilan data secara tersegmentasi.
10. *Data Encryption*: *CodeIgniter* menyediakan fungsi enkripsi data yang memudahkan pengembang dalam melindungi data sensitif dalam aplikasi.
11. *Full Page Caching*: *CodeIgniter* mendukung caching halaman penuh, yang memungkinkan pengembang untuk menyimpan versi yang sudah di-generate dari halaman dalam *cache*, meningkatkan kinerja dan responsivitas aplikasi.
12. *Application Profiling*: *CodeIgniter* menyediakan alat bantu untuk analisis kinerja aplikasi, yang membantu pengembang dalam memonitor dan menganalisis waktu eksekusi dan penggunaan sumber daya.
13. *Template Engine Class*: *CodeIgniter* menyediakan kelas mesin template, yang memudahkan pengembang dalam memisahkan tampilan dari logika bisnis dan menciptakan tampilan yang konsisten.

14. *Search engine friendly URLs*: *CodeIgniter* memiliki dukungan untuk URL yang ramah dengan mesin pencari. Ini memungkinkan pengembang untuk membuat URL yang bersih, deskriptif, dan mudah dibaca oleh mesin pencari, sehingga meningkatkan visibilitas dan optimasi SEO (*Search Engine Optimazation)*.
15. *Large libarary of helper function*: *CodeIgniter* menyediakan berbagai fungsi pembantu (*helper functions*) yang siap pakai. Fungsi-fungsi ini membantu pengembang dalam tugas-tugas umum seperti manipulasi string, pemformatan tanggal, pengiriman email, manipulasi array, dan banyak lagi.

Tentunya fitur-fitur tersebut menjadikan *CodeIgniter* sebagai *framework* yang sangat berguna dan populer dalam pengembangan aplikasi web dengan PHP. Dengan fitur-fitur yang lengkap dan dukungan untuk pola desain MVC, *CodeIgniter* memungkinkan pengembang untuk membangun aplikasi web yang efisien, aman, dan mudah dikelola.

Selanjutnya adapun kelebihan ataupun keunggulan yang ditawarkan oleh *Codeigniter* ialah sebagai berikut (Setiadi, 2018:3):

1. *Codeigniter* adalah *framework* PHP yang bersifat *open-source*.
2. *Codeigniter* memiliki ukuran yang kecil dibandingkan dengan *framework* lain. Setelah proses intalasi, *framework Codeigniter* hanya berukuran kurang lebih 2 MB (tanpa dokumentasi atau jika *user*_guide dihapus).
3. Aplikasi yang dibuat menggunakan *codeigniter* bisa bejalan cepat.
4. *Codeigniter* menggunakan pola desain *Model-View-Controller* (MVC) sehingga satu file tidak terlalu berisi banyak kode. Hal ini menjadikan kode lebih mudah dibaca, dipahami, dan dipelihara di kemudian hari.
5. *Codeigniter* dapat diperluas sesuai dengan kebutuhan.
6. *Codeigniter* terdokumentasi dengan baik informasi tentang pustaka (*Library*) dan fungsi yang disediakan oleh *codeigniter* dapat diperoleh melalui dokumentasi yang disertakan di dalam paket distribusinya.
7. *Codeigniter* memiliki *library* dan *helper* yang lengkap.

8. *Codeigniter* memiliki *security* yang handal seperti *xss filtering*, *session encryption*, dan lain-lain.
9. *Codeigniter* mengizinkan pengembang web menggunakan *library* atau *helper* yang tidak disediakan oleh *codeigniter* seperti: *Google Map API*, *Facebook API*, *fpdf*, dan lain sebagainya.
10. *Codeigniter* bersifat tidak kaku. sehingga memberikan kebebasan kepada pengembang web untuk mengembangkan aplikasi berbasis web bahkan tanpa *framework*.
11. *Codeigniter* memiliki komunitas yang besar dan tersebar di seluruh dunia, sehingga memudahkan para pengembang web untuk memecahkan permasalahan (*problem solving*) yang dihadapi para pengembang web di saat mengembangkan aplikasi berbasis web.
12. *Codeigniter* mendukung banyak RDBMS (*Relational Database Management System*) seperti MySQL, MySQLi, SQL *Server*, Oracle, Maria DB, PostgreSQL, SQLite, dan lain sebagainya.
13. *Codeigniter* pada dasarnya menganut *Clean URL* dan mendukung SEO (*Search Engine Optimazation*).

Adapun selanjutnya kekurangan *Codeigniter* ialah sebagai berikut (Fauzan & Roza, 2019:84):

1. *CodeIgniter* tidak ditujukan untuk membuat web dengan skala besar

   CodeIgniter dirancang untuk menjadi *framework* yang ringan dan sederhana, sehingga tidak memiliki dukungan yang kuat untuk mengatasi proyek-proyek dengan skala besar. Jika ingin mengembangkan aplikasi yang membutuhkan skala yang sangat besar dan kompleks, mungkin *CodeIgniter* bukanlah pilihan yang tepat karena kemampuannya yang terbatas dalam menangani kompleksitas tersebut.

2. *Library* yang terbatas

   Meskipun *CodeIgniter* menyediakan kumpulan *library* yang cukup lengkap untuk menangani tugas-tugas umum, namun mencari *plugin* tambahan yang terverifikasi secara resmi bisa menjadi tantangan. Karena *CodeIgniter* tidak menyediakan repositori resmi untuk plugin tambahan, pengembang harus mencari secara eksternal melalui

sumber-sumber pihak ketiga. Hal ini dapat mengakibatkan keterbatasan dalam menemukan *plugin* yang terverifikasi dan terpercaya, sehingga pengembangan aplikasi dapat terhambat.

3. Belum adanya editor khusus *CodeIgniter*

Salah satu kekurangan *CodeIgniter* adalah ketiadaan editor khusus yang didedikasikan untuk pengembangan dengan *framework* ini. Ini berarti ketika seseorang ingin membuat proyek baru atau mengelola modul dalam *CodeIgniter*, dia harus berpindah folder dan menggunakan editor umum seperti *Sublime Text, Visual Studio Code*, atau IDE lainnya. Kekurangan ini dapat mempengaruhi efisiensi pengembangan dan pengelolaan proyek karena pengembang harus bergantung pada alat eksternal untuk fitur-fitur seperti auto-complete, debugging, dan pengaturan proyek yang spesifik untuk *CodeIgniter*.

Berdasarkan pemaparan di atas, dapat disimpulkan bahwa *CodeIgniter* adalah sebuah *framework* pengembangan aplikasi web berbasis PHP yang ringan, sederhana, dan efisien. *CodeIgniter* menyediakan sekumpulan *library* dan fitur lengkap yang memudahkan pengembang dalam mengembangkan aplikasi web dengan cepat dan mudah. Dengan mengadopsi pola desain *Model View Controller* (MVC) dan struktur aplikasi yang terorganisir, *CodeIgniter* membantu memisahkan tanggung jawab dalam pengembangan aplikasi dan meningkatkan efisiensi tim. *CodeIgniter* juga memiliki keamanan yang handal, dukungan untuk berbagai platform database, dan fitur-fitur penting seperti form dan validasi, manajemen sesi, pengiriman *email*, dan *caching* halaman penuh. Namun, *CodeIgniter* memiliki keterbatasan dalam mengatasi proyek dengan skala besar dan keterbatasan dalam jumlah library yang terverifikasi resmi. Meskipun demikian, dengan kelebihan-kelebihan yang ditawarkan, *CodeIgniter* tetap menjadi pilihan yang populer dan berguna dalam pengembangan aplikasi web dengan PHP. Adapun contoh *website* yang menggunakan CI antara lain yaitu: *Casio Computer, The Mail Guardian, McClatchy* dan *Buffer*.

2.2.1.2. Framework Laravel

Laravel adalah sebuah *Framework* PHP yang dikembangkan dengan fokus pada kesederhanaan dan fleksibilitas desainnya. *Framework* ini tersedia dengan lisensi MIT dan sumber kode dapat diakses melalui repositori GitHub. Seperti kebanyakan *framework* PHP lainnya, *Laravel* mengadopsi pola desain *Model View Controller* (MVC). Salah satu fitur unik yang dimiliki oleh *Laravel* adalah adanya command *line tool* bernama "Artisan" yang memudahkan dalam pengelolaan paket (*bundle*) dan instalasi paket tambahan (Husada, 2019:29).

Laravel sendiri merupakan sebuah *framework* PHP yang pertama kali dirilis pada tahun 2011 oleh Taylor Otwell, seorang pengembang web dari Kanada. Dirancang sebagai alternatif yang lebih sederhana dan elegan, *Laravel* mendapatkan popularitas yang signifikan dengan peluncuran versi 4 pada tahun 2013. Versi tersebut membawa inovasi dengan memperkenalkan konsep "*Composer*" untuk manajemen dependensi, serta menghadirkan fitur-fitur yang mendukung pengembangan web yang efisien dan mudah. Sejak itu, *Laravel* terus berkembang dengan peluncuran versi-versi baru yang menghadirkan peningkatan fitur, keamanan, dan performa, menjadikannya salah satu *framework* PHP yang paling populer dan digunakan secara luas dalam industri pengembangan web (Yudhanto & Prasetyo, 2018:17).

Adapun berikut fitur-fitur dari *Laravel* (Yudhanto & Prasetyo, 2018:47-48):

1. *Blade Template Engine*: *Blade* adalah mesin template yang kuat dalam *Laravel*. Ia menyediakan sintaks yang sederhana dan ekspresif untuk membuat tampilan yang dinamis dan *reusable*.
2. *Routing*: *Laravel* menyediakan sistem routing yang fleksibel untuk mengarahkan permintaan HTTP ke kode yang sesuai di aplikasi. Hal ini memudahkan dalam mengatur rute URL dan menangani permintaan dari pengguna.
3. *Modularity*: *Laravel* menerapkan pola arsitektur yang modular, yang memungkinkan pengembangan aplikasi yang terstruktur dan mudah dikelola. Pengguna dapat membagi kode ke dalam modul-modul yang terpisah untuk meningkatkan keterbacaan dan kemudahan pemeliharaan.

4. *Testability*: *Laravel* memiliki kerangka kerja pengujian yang terintegrasi, yang memudahkan dalam melakukan pengujian unit dan fungsional pada aplikasi. Ini membantu memastikan kehandalan dan kualitas kode.
5. *Query Builder and ORM*: *Laravel* menyediakan *Query Builder* yang mudah digunakan untuk membangun kueri database secara programatik. Selain itu, *Laravel* juga memiliki ORM (*Object Relational Mapping*) yang disebut *Eloquent*, yang memudahkan pengguna dalam berinteraksi dengan database menggunakan model dan relasi.
6. *Authentication*: *Laravel* menyediakan sistem otentikasi yang siap pakai. Pengguna dapat dengan mudah mengimplementasikan fitur otentikasi dan otorisasi ke dalam aplikasi mereka dengan menggunakan metode yang telah disediakan.
7. *Schema Builder*: *Laravel* menyediakan *Schema Builder* untuk mengelola struktur *database*. Pengguna dapat membuat dan mengubah tabel serta indeks dengan menggunakan sintaks yang sederhana.
8. *Configuration Management Features*: *Laravel* memiliki fitur manajemen konfigurasi yang kuat. Pengguna dapat dengan mudah mengelola konfigurasi aplikasi dengan menggunakan file *environment*, file konfigurasi, atau variabel lingkungan.
9. *E-mail Class*: *Laravel* menyediakan kelas untuk mengirim email dengan mudah dan efisien. Pengguna dapat mengatur pengiriman email dengan berbagai protokol dan template yang dapat disesuaikan.
10. *Redis*: *Laravel* mendukung integrasi dengan *Redis*, sebuah database penyimpanan data berkinerja tinggi. Pengguna dapat dengan mudah menggunakan *Redis* untuk *cache*, sesi, dan antrian pekerjaan.
11. *Event and Command Bus*: *Laravel* menyediakan sistem *event* dan *command* bus yang memungkinkan pengguna untuk menerapkan pola publikasi-subskripsi (*publish-subscribe*) dan pemrosesan perintah secara terpusat dalam aplikasi mereka.
12. *Eloquent ORM*: *Eloquent* adalah ORM bawaan *Laravel* yang menyederhanakan interaksi dengan *database*. Ia menyediakan berbagai fitur

seperti pengelolaan relasi, pengambilan data yang lancar, dan penanganan operasi database dengan cara yang mudah dipahami.

Selanjutnya adapun kelebihan ataupun keunggulan yang ditawarkan oleh *Laravel* ialah sebagai berikut (Abdulloh, 2022:11):

1. *Laravel* memiliki banyak fitur yang tidak dimiliki oleh *framework* lain, seperti : *Arisan, migrations, blade* dan sebagainya.
2. *Laravel* merupakan *framework* PHP yang ekspresif. Artinya sintaks pada *laravel* menggunakan bahasa yang mudah dimengerti sehingga programmer pemula sekalipun akan mudah paham kegunaan suatu sintaks walaupun belum mempelajarinya.
3. *Laravel* memiliki dokumentasi yang cukup lengkap, bahkan setiap versinya memiliki dokumentasi tersendiri mulai dari cara instalasi hingga penggunaan fitur-fiturnya.
4. *Laravel* digunakan oleh banyak *programmer*, sehingga banyak *library* yang mendukung *laravel* yang diciptakan para *programmer* pencinta *laravel*.
5. *Laravel* didukung oleh *composer*, sehingga *library-library laravel* dapat didapatkan dengan mudah dari internet menggunakan *composer*. *Composer* sendiri merupakan *dependency management PHP* yang membantu seorang *programmer* untuk mendapatkan *library* yang dia pakai dan menginstalnya dari internet.
6. *Laravel* memiliki template *engine* tersendiri yang diberi nama *blade* yang memudahkan seorang programmer menampilkan data pada template HTML.

Selain keunggulan yang telah disebutkan diatas, berikut adalah beberapa keunggulan *Laravel* lainnya:

1. Sistem *Routing* yang Kuat

Laravel menyediakan sistem *routing* yang kuat dan fleksibel. Seorang *programmer* dapat dengan mudah menentukan rute-rute aplikasinya dan mengarahkannya ke tindakan-tindakan yang sesuai. Sistem *routing* ini memungkinkan *programmer* untuk mengatur URL dengan mudah dan membuat pengembangan aplikasi web menjadi lebih terstruktur.

2. Sistem *Caching* yang Efektif

   Laravel menyediakan fitur *caching* yang kuat untuk meningkatkan performa aplikasi seorang *programmer*. Dengan menggunakan *cache,* seorang *programmer* dapat menyimpan hasil *query database* atau hasil komputasi yang mahal secara sementara dalam memori, sehingga mengurangi waktu akses dan meningkatkan kecepatan respons aplikasi.

3. Sistem Otentikasi dan Otorisasi yang Terintegrasi

   Laravel menyediakan sistem otentikasi yang lengkap dan mudah digunakan. Seorang *programmer* dapat dengan cepat mengimplementasikan otentikasi pengguna, mengelola autentikasi, mengatur hak akses, dan masih banyak lagi. *Laravel* juga menyediakan fitur otorisasi yang kuat dengan menggunakan *gate* dan *policies*.

4. Migrasi *Database* yang Mudah

   Laravel menyediakan fitur migrasi *database* yang memungkinkan seornag *programmer* untuk secara mudah mengelola perubahan skema databasenya. Seorang *programmer* dapat menggunakan kode PHP untuk membuat dan mengubah tabel, indeks, dan kunci asing. Dengan migrasi database, maka akan dapat mengatur dan melacak perubahan skema database secara efisien.

5. *Community Support* yang Aktif

   Laravel memiliki komunitas pengguna yang besar dan aktif. Seoranf *programmer* dapat dengan mudah menemukan *tutorial*, forum diskusi, dan paket-paket tambahan yang dibuat oleh komunitas *Laravel*. *Community support* yang aktif dapat membantu dalam menyelesaikan masalah, berbagi pengetahuan, dan memperluas kemampuan *Laravel*.

Disamping kelebihan *Laravel* praktis juga mempunyai kekurangan, adapun berikut kekurangan dari *Laravel* (Mahfud & Asmunin, 2021:3)*:*

1. *Laravel update* terlalu cepat, hal tersebut akan menyusahkan pengembang yang menggunakan *Laravel* pada versi sebelumnya karena pada setiap *update* mempunyai perbedaan struktur file maupun folder namun resiko lain jika tidak *update* maka lambat laun aplikasi akan ketinggalan zaman.
2. Aplikasi yang jalan di browser tidak langsung *update style* cssnya setelah cssnya diubah. Perlu dilakukan *clear cache* terlebih dahulu untuk merubahnya.
3. Fitur yang terlalu luas, *Laravel* mempunyai fitur yang begitu banyak sehingga membingungkan pengembang yang baru belajar *Laravel*, karena saking canggihnya *Laravel* banyak fitur yang jarang digunakan namun ada pada *Laravel*.
4. Belajar untuk pemula sangat susah sebab untuk pemula akan susah masuk logikanya disarankan untuk belajar *Codeigniter* dahulu sebelum *Laravel*.
5. Penulisan *script* pada *Laravel* tergolong lebih rumit dibanding *framework* lainnya.
6. Perbedaan lingkungan Windows dan Linux, jika *Laravel* diupload di *server* Linux maka perlu dijalankan composer update di terminal Linux, yang menjadi masalah adalah jika tidak mempunyai akses ke terminalnya misalkan hosting membeli dilayanan hosting di mana bisa saja satu *server* berbarengan dengan yang lain, maka perlu menghubungi pihak penyedia hosting untuk melakukan composer update.
7. *Laravel* tidak mendukung *implicit routing*, hal ini membuat pengembang yang menggunakan *Laravel* memerlukan waktu yang lebih untuk mendaftarkan model binding kedalam *library route service provider*.

Dapat disimpulkan bahwa, *Laravel* adalah *framework* PHP yang populer berkat kesederhanaan dan fleksibilitas desainnya. Dengan menggunakan pola desain MVC, *Laravel* menyediakan fitur-fitur seperti *Blade Template Engine*, sistem *routing* yang kuat, modularitas, kemudahan dalam pengujian, *ORM Eloquent*, dan masih banyak lagi. Dukungan komunitas yang aktif dan dokumentasi yang lengkap membuat *Laravel* menjadi pilihan utama para pengembang web untuk membangun aplikasi web yang efisien dan terstruktur. Adapun contoh *website* yang menggunakan *Laravel* antara lain seperti: *My Rank*, *Asgard CMS* dan *World Walking*.

2.2.1.3. Perbedaan *Framework Codeigniter* dan *Framework Laravel*

Adapun berikut adalah kunci perbedaan dari *codeigniter* dan *laravel* (Habibi et al., 2019:15-17):

1. Dukungan ORM:
    - *Laravel*: Mendukung ORM menggunakan *Eloquent* ORM yang kuat. *Eloquent* menyediakan cara yang mudah dan intuitif untuk berinteraksi dengan *database* menggunakan objek dan model.
    - *CodeIgniter*: Tidak memiliki ORM bawaan, tetapi bisa menggunakan pendekatan *Active Record Pattern* untuk berinteraksi dengan *database*. Menggunakan fungsi-fungsi database yang telah disediakan oleh *CodeIgniter* untuk melakukan operasi CRUD.
2. Pemrosesan *Database*:
    - *Laravel*: Menyediakan *Query Builder* yang kuat untuk membangun dan mengeksekusi *query database* dengan sintaks yang ekspresif dan mudah dipahami. *Laravel* juga mendukung bahasa kueri SQL mentah jika diperlukan.
    - *CodeIgniter*: Menyediakan *Query Builder* yang sederhana untuk membangun dan mengeksekusi query database. *Query Builder CodeIgniter* memiliki sintaks yang lebih sederhana dan terbatas dibandingkan dengan *Laravel*.
3. Pengelolaan Modul:
    - *Laravel*: Memiliki fitur pengelolaan paket (*package*) menggunakan *Composer*, yang memudahkan instalasi dan pengaturan paket-paket pihak ketiga dalam proyek *Laravel. Laravel* juga mendukung konsep modularitas dengan menggunakan *namespace* dan *autoloading*.

- *CodeIgniter*: Tidak memiliki fitur pengelolaan modul (*package*) bawaan. Pengelolaan modul harus dilakukan secara manual atau dengan menggunakan *library/library* pihak ketiga.

4. Migrasi *Database*:
    - *Laravel*: Memiliki fitur migrasi *database* bawaan yang memungkinkan pengembang mengatur dan melacak perubahan skema *database* secara terstruktur. Migrasi *database Laravel* menggunakan file PHP sebagai skrip migrasi yang dapat dijalankan dan di-*rollback*.
    - *CodeIgniter*: Tidak memiliki fitur migrasi *database* bawaan. Migrasi *database* dalam *CodeIgniter* dapat dilakukan secara manual dengan menggunakan skrip SQL atau dengan menggunakan *library/library* pihak ketiga.
5. REST APIs:
    - *Laravel*: Memiliki dukungan bawaan untuk membangun REST APIs dengan menggunakan fitur *seperti routing*, *middleware*, dan kontroler. *Laravel* menyediakan cara yang mudah untuk mengkonfigurasi rute-rute API dan menangani permintaan HTTP dengan metode yang sesuai.
    - *CodeIgniter*: Tidak memiliki dukungan bawaan untuk membangun REST APIs. Namun, *CodeIgniter* dapat digunakan untuk membangun REST APIs dengan mengatur rute-rute yang sesuai dan menangani permintaan HTTP dengan menggunakan kontroler dan fungsi-fungsi database yang disediakan.

2.2.2. Site24x7.com

Site24x7.com adalah solusi pemantauan yang diciptakan oleh gabungan keahlian IT dari ManageEngine dan Zoho. Solusi ini dirancang untuk membantu IT dan DevOps dalam memantau aplikasi skala cloud, seperti situs web, *server*, aplikasi, jaringan, dan layanan cloud lainnya. Platform ini menawarkan 16 jenis pemantauan yang mencakup metrik, jejak, dan log, yang dapat diakses melalui satu konsol. Salah satu fitur

yang diberikan adalah pemantauan situs web. Dalam skripsi ini, penulis akan menggunakan Site24x7 sebagai alat untuk menguji performa dengan menggunakan parameter *response time*, *throughput*, dan page load time. *Response time* mengukur waktu yang dibutuhkan oleh sistem untuk merespons permintaan. *Throughput* mengacu pada jumlah permintaan yang berhasil diproses dalam satuan waktu tertentu. *Page load time* mengukur waktu yang dibutuhkan untuk memuat seluruh konten halaman web.

2.2.1.1. Response Time

Response time, atau waktu respons, mengacu pada waktu yang diperlukan oleh sistem atau aplikasi untuk merespons permintaan yang diterima. (Laaziri et al., 2019:865). Selanjutnya menurut Site24x7.com, *response time* merupakan waktu yang diperoleh dari hasil penjumlahan waktu pencarian DNS, waktu koneksi, waktu handshake SSL, dan waktu yang dibutuhkan untuk mengunduh seluruh konten HTML. Namun itu tidak termasuk mendapatkan gambar dan sumber daya lainnya yang dimuat dalam halaman HTML.

Dapat disimpulkan bahwa *response time* adalah waktu yang dibutuhkan oleh sistem atau aplikasi untuk merespons permintaan yang diterima. Hal ini mencakup waktu pencarian DNS, waktu koneksi, waktu handshake SSL, dan waktu yang diperlukan untuk mengunduh konten HTML. Pada pengujian performa, *response time* digunakan untuk mengukur kecepatan respons suatu aplikasi atau layanan terhadap permintaan yang diberikan. Semakin rendah *response time*, semakin cepat respons yang diberikan oleh sistem. *Response time* dapat dipengaruhi oleh faktor-faktor seperti kecepatan *server*, pengolahan data, koneksi jaringan, dan optimisasi aplikasi. Monitoring dan memperbaiki *response time* yang lambat merupakan langkah penting dalam meningkatkan performa suatu sistem atau aplikasi.

2.2.1.2. Throughput

Throughput adalah jumlah data per satuan waktu yang dikirim untuk suatu terminal tertentu di dalam sebuah jaringan, dari suatu titik jaringan, atau dari suatu titik ke titik jaringan yang lain.(Aswariza et al., 2017:72). Selanjutnya menurut Site24x7.com, *Throughput* adalah hasil perhitungan yang didapatkan dengan membagi ukuran konten (dalam satuan KB) dengan waktu respons (dalam satuan detik) selama

periode waktu tertentu. Data ini memberikan informasi penting dalam menganalisis kecepatan koneksi dan penggunaan bandwidth.

Berdasarkan definisi diatas dapat disimpulkan bahwa, *throughput* merupakan ukuran jumlah data yang dikirim atau diterima dalam satuan waktu di dalam sebuah jaringan. Melalui perhitungan yang dilakukan dengan membagi ukuran konten dengan waktu respons, *throughput* memberikan informasi penting tentang kecepatan koneksi dan penggunaan bandwidth. Data *throughput* membantu dalam menganalisis performa jaringan, mengoptimalkan penggunaan bandwidth, dan memastikan efisiensi dalam pengiriman dan penerimaan data.

2.2.3. Page Load Time

Page load time adalah waktu yang dibutuhkan sebuah web untuk menampilkan halaman sebuah halaman web sampai selesai (Sirajuddin et al., 2012:51). Selanjutnya menurut situs resmi site24x7.com, Page Load Time atau waktu muat halaman adalah ukuran waktu yang dibutuhkan untuk memuat seluruh konten sebuah halaman web dari saat permintaan dikirimkan hingga halaman tersebut sepenuhnya ditampilkan di browser pengguna. Waktu muat halaman mencakup waktu untuk mengunduh semua sumber daya halaman, seperti HTML, CSS, JavaScript, gambar, dan file multimedia lainnya.

Berdasarkan definisi diatas dapat disimpulkan bahwa, *Page Load Time* adalah waktu yang dibutuhkan oleh sebuah halaman web untuk sepenuhnya ditampilkan di browser pengguna setelah permintaan dikirimkan. Hal ini melibatkan proses mengunduh semua sumber daya halaman, termasuk HTML, CSS, JavaScript, gambar, dan file multimedia lainnya. Page Load Time menjadi penting karena mempengaruhi pengalaman pengguna dalam mengakses halaman web. Semakin cepat waktu muat halaman, semakin baik pengalaman pengguna dan kemungkinan lebih tinggi bagi pengguna untuk tetap berinteraksi dengan halaman tersebut. Sebaliknya, jika waktu muat halaman terlalu lama, pengguna dapat mengalami ketidaknyamanan, kehilangan minat, dan bahkan meninggalkan halaman tersebut. Oleh karena itu, mengoptimalkan dan memantau Page Load Time menjadi faktor penting dalam menghadirkan pengalaman pengguna yang baik dan meningkatkan kinerja sebuah situs web.

2.2.4. Apache Benchmark

Apache Bench adalah sebuah *tool* dari *Apache organization* yang digunakan untuk mengukur performansi pada *Hypertext Transfer Protocol* (HTTP) *web server*. *Tool* ini digunakan untuk menghitung berapa banyak *request per second* yang dapat di layani oleh *web server* yang digunakan.(Chandra, 2019:49).

Apache Benchmark adalah sebuah perangkat lunak pengujian kinerja yang digunakan untuk mengukur kemampuan dan kinerja sebuah *server* web dengan mengirimkan serangkaian permintaan HTTP ke *server* tersebut dan mencatat responsenya. Dengan *Apache Benchmark*, pengguna dapat menguji seberapa banyak permintaan yang dapat ditangani oleh *server* dalam periode waktu tertentu, serta mengukur waktu respons *server* terhadap permintaan tersebut (Laaziri et al., 2019:865).

Berdasarkan definisi diatas dapat disimpulkan bahwa *Apache Benchmark* adalah sebuah **tool** dari Apache yang digunakan untuk mengukur performansi *server* web dengan mengirimkan permintaan HTTP dan mencatat responsenya. Tool ini membantu pengguna menghitung berapa banyak *request per second* yang dapat ditangani oleh web *server* yang sedang diuji. Dengan *Apache Benchmark*, pengguna dapat melakukan pengujian kinerja *server* web untuk menganalisis seberapa efisien dan responsif *server* tersebut dalam menangani beban permintaan. Selanjutnya Pada skripsi ini parameter yang akan digunakan pada pengujian performa menggunakan tool *Apache Benchmark* adalah *request per second*.

2.2.4.1. *Request Per Second*

RPS (*Request per second*) artinya berapa banyak *request* yang dapat diproses per detik. Semakin tinggi angkanya maka semakin efisien *framework* tersebut (Shiddieq & Aqmarina, 2013:2). Selanjutnya, menurut situs resmi Apache *request per second* adalah jumlah *request* per detik, dimana nilai ini merupakan hasil pembagian jumlah permintaan dengan total waktu yang dibutuhkan.

Jadi dapat disimpulkan bahwa *request per second* merupakan kemampuan sebuah *website* dalam menyelesaikan sejumlah *request* dalam waktu satu detik