

BAB II

TINJAUAN PUSTAKA DAN DASAR TEORI

2.1 Tinjauan Pustaka

Dalam penelitian ini penulis mengacu pada beberapa jurnal yang berhubungan dengan perancangan dan implementasi RESTful API, adapun jurnal yang digunakan sebagai referensi utama dalam penelitian ini adalah :

Tabel 2.1 Penelitian terkait

No.	Penulis	Judul	Metode	Masalah	Hasil
1.	(Fitrianto, 2017)	Sistem Informasi Monitoring Dan Evaluasi Tugas Akhir Mahasiswa Studi kasus Universitas PGRI Ronggolawe (UNIROW) Tuban	Sistem informasi berbasis website	Masih kurangnya mahasiswa dalam mendapatkan informasi tentang syarat yang harus dipenuhi untuk mengerjakan skripsi, kurangnya informasi dosen pembimbing mengenai nama mahasiswa yang menjadi bimbingannya, kurangnya monitoring dalam pengerjaan (progres) mahasiswa hal ini akan berdampak memperlambat penyelesaian skripsi.	Sistem informasi berbasis website yang dapat mengkoordinasikan pengajuan skripsi, mempermudah mahasiswa dalam melakukan pengajuan skripsi, dan menyimpan data skripsi secara online.

No.	Penulis	Judul	Metode	Masalah	Hasil
2.	(Bagus dkk., 2020)	Sistem Informasi Tugas Akhir Program Studi Teknik Informatika Universitas Mataram	Pengembangan sistem tugas akhir berbasis website menggunakan metode waterfall dan diagram UML (<i>Unified Modeling Language</i>)	Sistem manual akan menghambat proses bimbingan dan konsultasi antara mahasiswa dan dosen pembimbing ketika dosen pembimbing sedang diluar kota atau memiliki kesibukan lain, selain itu hal ini juga akan menyulitkan dosen pembimbing untuk mengontrol mahasiswa dalam proses pengerjaan skripsi.	Sistem Informasi Tugas Akhir berbasis website dengan menggunakan <i>framework</i> Laravel yang dapat mempercepat dalam penanganan administrasi tugas akhir.
3.	(Heryatno, 2020)	Pengembangan Sistem Informasi UIIPerkuliahan Dengan RESTful API	Teknologi <i>web service</i> untuk fleksibilitas pengembangan aplikasi dan ketangkasan aplikasi dalam mengatasi <i>request</i> dalam jumlah besar.	Aplikasi W-Simak saat ini belum menggunakan <i>web service</i> , menyebabkan setiap perubahan yang dilakukan mengharuskan <i>deploy</i> ulang keseluruhan aplikasi yaitu bagian <i>back end</i> dan <i>front end</i> yang berimplikasi pada waktu <i>deploy</i> aplikasi tersebut. Belum adanya <i>web</i>	RESTful API untuk mengelola nilai mahasiswa, presensi mahasiswa, dan pencetakan berkas perkuliahan, yang kemudian RESTful API tersebut dikonsumsi oleh aplikasi berbasis web.

No.	Penulis	Judul	Metode	Masalah	Hasil
				<i>service</i> juga menyebabkan aplikasi diluar W-Simak harus membuat ulang layanan yang sebenarnya sudah ada. Selain itu ketika aplikasi W-Simak diakses oleh banyak pengguna sekaligus, hal ini akan menyebabkan turunnya performa dan kemampuan aplikasi.	
4.	(Somya & Nathanael, 2019)	Pengembangan Sistem Informasi Pelatihan Berbasis Web Menggunakan Teknologi <i>Web Service</i> Dan <i>Framework</i> Laravel	Penerapan <i>web service</i> dan <i>framework</i> Laravel dalam proses integrasi data.	Untuk mengakses aplikasi Pincher ID pengguna harus menginstal aplikasi melalui <i>Playstore</i> atau <i>App Store</i> , hal tersebut menyebabkan aplikasi Pincher ID tidak dapat terindeks oleh <i>search engine</i> sehingga menghambat pengguna dalam menemukan aplikasi Pincher ID.	Sistem berbasis website yang menerapkan teknologi <i>web service</i> dapat membantu dalam integrasi data secara terpusat, serta dengan menggunakan <i>framework</i> laravel menghasilkan sebuah sistem yang ringan dan performa yang cepat.

No.	Penulis	Judul	Metode	Masalah	Hasil
5.	(Arsana & Adnyana, 2020)	Implementasi Web Service Pada Integrasi Data Kerja Praktik, Seminar Dan Tugas Akhir	Implementasi teknologi RESTful dengan menggunakan format data <i>JSON</i> untuk pertukaran data.	Dengan dikembangkannya dua sistem informasi yang berbeda pada STMIK STIKOM Indonesia yang dikembangkan dengan database yang berbeda, data master tidak dapat digunakan secara bersama-sama karena berbeda <i>platform</i> dan struktur tabel dalam <i>database</i> .	<i>RESTful web Service</i> yang dibangun dengan bahasa pemrograman PHP menggunakan <i>framework</i> Lumen yang digunakan dalam proses integrasi data.
6.	(Sutrisno dkk., 2019)	Perancangan sistem pemasangan iklan online pada aplikasi <i>e-commerce</i> (e-gemanausa) menggunakan metode RESTful API dan <i>framework</i> laravel	RESTful API sebagai metode transfer data dan UML sebagai metode penggambaran sistem	Sistem <i>e-gemanausa</i> yang belum menerapkan metode <i>service oriented</i> menyulitkan dalam proses pengembangan.	Rancangan sistem pemasangan iklan <i>online</i> dengan menggunakan arsitektur <i>microservice</i> .

Dari jurnal pertama yang disusun oleh (Fitrianto , 2017) dengan judul Sistem Informasi Monitoring Dan Evaluasi Tugas Akhir Mahasiswa Studi kasus Universitas PGRI Ronggolawe (UNIROW) Tuban, menghasilkan sebuah website yang dibangun dengan *Yii framework* dan *MySQL DBMS* sehingga dapat memberikan panduan dan menyediakan informasi data mahasiswa dalam proses mengerjakan tugas akhir, serta membantu dosen dalam memonitoring progress pengerjaan tugas akhir mahasiswa.

Kemudian dari jurnal kedua yang ditulis oleh (Bagus dkk., 2020) dengan judul Sistem Informasi Tugas Akhir Program Studi Teknik Informatika Universitas Mataram menghasilkan sebuah sistem berbasis website yang dapat mempercepat dalam proses pelayanan administrasi dalam mengerjakan tugas akhir.

Sedangkan pada jurnal ketiga dengan judul Pengembangan Sistem Informasi UIIPerkuliahan Dengan RESTful API yang disusun oleh (Heryatno, 2020), didapatkan sebuah RESTful API yang dapat digunakan sebagai sumber data atau *resource* saat terdapat aplikasi diluar Sistem Informasi UIIPerkuliahan ingin terhubung pada layanan yang sudah ada pada Sistem Informasi UIIPerkuliahan.

Selain itu (Somya & Nathanael, 2019) dalam jurnalnya yang berjudul Pengembangan Sistem Informasi Pelatihan Berbasis Web Menggunakan Teknologi Web Service Dan Framework Laravel menyatakan dengan penerapan *web service* pada Pincher ID dapat mempermudah proses integrasi data dapat dilakukan secara terpusat.

Kemudian dari jurnal kelima yang berjudul Implementasi Web Service Pada Integrasi Data Kerja Praktik, Seminar Dan Tugas Akhir yang ditulis oleh (Arsana & Adnyana, 2020), dengan mengimplementasi teknologi RESTful menggunakan format data *JSON* dapat dipergunakan untuk mengintegrasikan data menjadi lebih lebih ringan, mudah dibaca dan ditulis.

Dan dari jurnal terakhir dengan judul Perancangan sistem pemasangan iklan online pada aplikasi *e-commerce* (e-gemanusa) menggunakan metode RESTful API dan *framework* Laravel yang ditulis oleh (Sutrisno dkk., 2019)

didapatkan hasil dengan penggunaan metode RESTful API akan mempermudah proses pengembangan sistem dan integrasi sistem dengan berbagai *platform*.

Dari beberapa jurnal tersebut dapat diambil kesimpulan bahwa penggunaan metode *RESTful API* kedepan akan pesat perkembangannya, terutama untuk komunikasi data antara *back-end* dan *front-end*. Sehingga dalam penelitian ini penulis mengajukan judul Perancangan Dan Implementasi RESTful API Pada Sistem Informasi Monitoring Dan Evaluasi Tugas Akhir Mahasiswa. Pada penelitian ini penulis mengembangkan RESTful API dengan format pertukaran data *JSON (JavaScript Object Nation)* untuk mengembalikan data dari *server* ke *client* melalui protokol *HTTP (Hypertext Transfer Protocol)*. Kemudian untuk pengujian, *RESTful API* yang telah dibuat akan diimplementasikan kedalam dua jenis sistem yang berbeda yaitu sistem berbasis *mobile* yang diangun dengan menggunakan *Flutter mobile app SDK (Software Development Kit)* dan sistem berbasis website yang dikembangkan dengan *framework* Laravel. Selain itu dengan membuat dokumentasi API dapat digunakan sebagai panduan pengembang yang akan menggunakan data pada *RESTful API* tersebut.

2.2 Dasar Teori

2.2.1 Sistem Informasi

Sistem informasi merupakan cara untuk mengelola kebutuhan transaksi yang mendukung fungsi manajerial sebuah organisasi untuk mendukung penyediaan data dan informasi kepada pihak lain melalui sebuah sistem. Komponen dari sistem informasi atau yang sering disebut dengan *building block* atau blok bangunan, yaitu (Sutabri, 2012) :

- a. *Input block* : Merupakan representasi data yang diinput ke sistem.
- b. *Model block* : Merupakan gabungan metode matematik, logika, dan prosedur dalam proses manipulasi data input yang akan disimpan dalam *database* sehingga menghasilkan sebuah informasi atau data.
- c. *Output block* : Merupakan *output* data atau informasi yang telah diolah oleh sistem informasi dalam *model block* atau blok model.

- d. *Technologi block* : Merupakan teknologi yang diterapkan dalam *input block*, *model block*, dan *output block* untuk mengendalikan keseluruhan data dan informasi.
- e. *Database block* : Merupakan data yang tersimpan pada komputer yang mempunyai relasi satu sama lain.
- f. *Control block* : Merupakan rancangan yang dibangun untuk mencegah, mengatasi dan mengendalikan segala kemungkinan yang dapat merusak sistem.

2.2.2 PHP

PHP atau *Hypertext Preprocessor* didefinisikan sebagai bahasa pemrograman *open source* yang mana proses kompilasi dan penerjemahan kode berjalan pada sisi *server* kemudian hasil data dikembalikan ke *client* dalam bentuk *HTML (Hypertext Markup Language)* (Jannah, dkk., 2019).

PHP merupakan bahasa pemrograman yang berjalan pada sisi *back-end* yang memiliki kelebihan dari segi performa, portabilitas, dan skalabilitas dalam proses manipulasi *database*, yang mana PHP merupakan bahasa pemrograman *open source* yang di desain khusus untuk pengembangan website (Supaartagorn dkk., 2010).

2.2.3 MySQL

MySQL adalah *server* basis data *open source* yang mudah untuk digunakan dan dapat diandalkan karena memiliki kinerja yang cepat serta berfungsi sebagai *RDBMS (relational database manajemen system)* yang berjalan pada arsitektur *client server* (Turban, 2017).

MySQL diciptakan oleh *programmer* asal Swedia Michael “Monty” Wedius pada tahun 1979, yang merupakan pengembangan dari konsep utama *database* untuk memasukkan, memilih dan menyeleksi data secara otomatis dan mudah oleh banyak pengguna pada waktu yang bersamaan (Amin, 2018).

2.2.4 Database

Simarmata & Paryudi (2006:1) menjelaskan bahwa bahwa (Octavian, 2013):

- a. *Database* adalah sekumpulan data atau informasi yang cocok digunakan oleh sebuah perusahaan (Silberschatz, dkk., 2002).
- b. Basisdata adalah sebuah cara yang digunakan untuk menyimpan data atau informasi (Stephens & Plew, 2000).
- c. *Database* adalah kumpulan dari sumber daya milik sebuah organisasi yang berbasis komputer (McLeod, dkk., 2001).
- d. Dan (Ramakrishnan & Gehrke, 2003) menjelaskan bahwa *database* adalah sekumpulan data yang menjelaskan aktivitas sebuah organisasi atau lebih yang saling berhubungan.

Selain itu basis data diartikan sebagai sekumpulan data yang dapat diolah dan dimanipulasi melalui perangkat lunak yang ada didalam komputer untuk menghasilkan sebuah data dan informasi yang disimpan secara sistematis (Yudhanto & Adi, 2018).

2.2.5 Database Management System (DBMS)

DBMS (*Database Management System*) adalah sebuah paket perangkat lunak seperti *MySQL*, *Microsoft SQL*, *Oracle*, *MS. Access* dan lain-lain yang digunakan untuk memasukkan, mengedit, menghapus, dan mengambil informasi dari *database* secara mudah dan efisien, selain itu *DBMS* memiliki beberapa kelebihan diantaranya (Yanto, 2016) :

1. Penggunaan memori dan penyimpanan yang lebih efisien.
2. Integritas data akan lebih terjamin.
3. Pembuatan antarmuka kedalam data akan lebih mudah.
4. Mempermudah pengelolaan basis data.
5. Flesibilitas dalam sistem keamanan.

2.2.6 Unified Modeling Language (UML)

UML atau *Unified Modeling Language* merupakan standar dalam pembuatan *blue print* sebuah sistem berorientasi objek yang meliputi alur bisnis, kelas, skema basis data, dan komponen lain yang diperlukan dalam merancang sebuah *software* melalui gambar atau grafik (Mubarak, 2019).


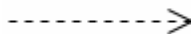
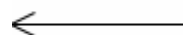
UML menyediakan standar dalam perancangan sebuah model sistem melalui visual dan dokumentasi *software* yang dapat berjalan pada perangkat keras, sistem operasi, dan bahasa pemrograman yang beragam (Dharwiyanti & Wahono, 2003).

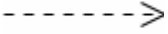





Berikut beberapa jenis diagram yang dapat digunakan dalam perancangan sistem menggunakan *Unified Modeling Language* (Henderi, 2009) :

a. Use Case Diagram (UCD)

Merepresentasikan interaksi “apa” yang dapat dilakukan oleh pengguna atau sistem lain kepada sistem yang dirancang, hal ini sangat membantu dalam proses penyusunan *requerment* aplikasi, merancang skenario pengujian, dan menyiapkan hasil rancangan aplikasi kepada *client*.

Tabel 2.2 Simbol Use Case Diagram (Henderi, 2009)



No	Simbol	Nama	Keterangan
1		<i>Actor</i>	Merupakan <i>user</i> atau sistem lain yang mempunyai hubungan dengan sistem yang dirancang untuk melakukan pekerjaan tertentu.
2		<i>Dependency</i>	Menunjukkan ketergantungan antara suatu elemen dengan elemen lain yang disebut dengan elemen <i>independent</i> atau tidak mandiri.
3		<i>Generalization</i>	Menunjukkan pewarisan spesifikasi sebuah elemen dari elemen lain.





No	Simbol	Nama	Keterangan
4		<i>Include</i>	Menunjukkan penggunaan suatu fungsi yang telah ada pada fungsi <i>use case</i> lain.
5		<i>Extend</i>	Menunjukkan perluasan fungsionalitas yang telah ada pada <i>use case</i> lain.
6		<i>Association</i>	Menunjukkan hubungan elemen antara <i>use case</i> dengan aktor.
7		<i>System</i>	Menunjukkan cakupan paket yang terdapat pada sistem.
8		<i>Use Case</i>	Menggambarkan bagaimana aktor menggunakan sistem.
9		<i>Collaboration</i>	Menunjukkan elemen lain atau aturan yang mempunyai perilaku lebih luas dari jumlah elemennya.

b. Activity Diagram

Activity diagram merupakan model aliran atau kontrol aktivitas ke aktivitas yang lain pada sebuah sistem yang digambarkan secara global, dinamis dan alamiah.

Tabel 2.3 Simbol Activity Diagram (Henderi, 2009)


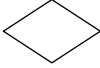
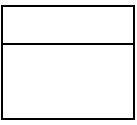

No.	Simbol	Nama	Keterangan
1		<i>Initial Node</i>	Menunjukkan awal dari sebuah aktivitas.
2		<i>Activity</i>	Menunjukkan aktivitas dari setiap kelas yang kemudian dapat diuraikan menjadi aktivitas yang lebih rinci.

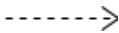

No.	Simbol	Nama	Keterangan
3		<i>Decision</i>	Menunjukkan percabangan untuk pengambilan keputusan.
4		<i>Activity Final Node</i>	Menunjukkan akhir dari sebuah aktivitas yang sedang berjalan.
5		<i>Fork</i>	Menggambarkan suatu aliran yang dapat berubah menjadi beberapa bagian aliran baru.
6		<i>Join</i>	Penggabungan dari beberapa aliran menjadi satu aliran.

c. Class Diagram

Merupakan gambaran strukrur statis yang menunjukkan deskripsi serta logika dari suatu *class* didalam sebuah sistem.

Tabel 2.4 Simbol Class Diagram (Henderi, 2009)

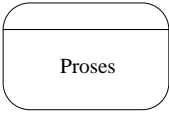
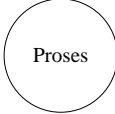
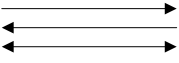

No	Simbol	Nama	Keterangan
1		<i>Generalization</i>	Pembagian struktur data dan perilaku dari objek induk (<i>ancestor</i>) kepada objek anak (<i>descendent</i>).
2		<i>N-Ary Association</i>	Cara untuk menghindar dari asosiasi memiliki objek lebih dari 2.
3		<i>Class</i>	Kumpulan objek yang memiliki operasi dan atribut yang sama.
4		<i>Realization</i>	Aktivitas nyata yang dilakukan oleh sebuah objek.


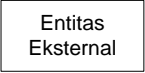


No	Simbol	Nama	Keterangan
5		<i>Dependency</i>	Menunjukkan hubungan yang mempengaruhi elemen lain yang bergantung pada elemen <i>independent</i> saat terjadi perubahan.
6		<i>Association</i>	Menunjukkan hubungan yang terjadi antar objek.

2.2.7 Data Flow Diagram (DFD)

DFD atau *Data Flow Diagram* merupakan gambaran aliran data secara logis dalam suatu sistem. DFD juga digunakan untuk menggambarkan hasil analisis dan rancangan terhadap sistem yang baru. Terdapat dua jenis simbol yang digunakan dalam membuat *Data Flow Diagram* yaitu DeMacro & Yourdon (1979) dan Gane & Sarson (1979) (Weli 2019).

Tabel 2.5 Simbol Data Flow Diagram (Weli 2019)

No	Simbol		Nama	Keterangan
	Gane/Sarson	Yourdon/DeMacro		
1			<i>Process</i>	Menunjukkan bagian dari suatu sistem yang merubah <i>input</i> menjadi <i>output</i> , yang menjelaskan kegiatan yang sedang / akan berjalan.
2			<i>Data Flow</i>	Menggambarkan data yang mengalir dalam sistem yang sedang berjalan.

No	Simbol		Nama	Keterangan
	Gane/Sarson	Yourdon/DeMacro		
3			<i>Terminator</i>	Simbol yang menunjukkan entitas eksternal, yang memiliki kepentingan terhadap sistem .
4			<i>Data Store</i>	Berbagai media untuk melakukan penyimpanan data.

2.2.8 Web Service

Web service adalah sebuah cara sistem untuk berinteraksi dan mengakses data dari sistem lain menggunakan teknologi yang berbeda, sebagai sebuah *Remote Procedure Call* yang dapat mengeksekusi fungsi yang telah didefinisikan oleh aplikasi web melalui sebuah API (*Application Programming Interface*), *web service* memiliki beberapa kelebihan, diantaranya (Sutanta & Mustofa, 2012):

1. Memungkinkan melakukan pertukaran data melalui sistem operasi dan perangkat yang berbeda.
2. Dapat dikembangkan dan diakses menggunakan berbagai bahasa pemrograman.
3. Mengesampingkan jenis DBMS yang digunakan saat ingin terhubung ke *database*.
4. Memudahkan proses pertukaran data.
5. Komponen yang sama pada sebuah aplikasi dapat digunakan secara berulang.

2.2.9 Representational State Transfer (REST)

REST adalah teknis komunikasi yang heterogen untuk aplikasi web, adopsi REST dapat menghasilkan arsitektur yang sederhana, dapat diskalakan, aman, efektif, dan handal. Banyak pengembang yang berhasil membuat API (*Application Programming Interface*) yang sederhana dan kuat pada *RESTful Web service* (Chen dkk., 2017).

REST (*Representational State Transfer*) merupakan strukur yang digunakan untuk mengembangkan *web services* dengan berfokus pada sumber daya dari sebuah sistem, termasuk bagaimana *resources* ditulis kedalam bahasa pemrograman yang berbeda menggunakan protokol HTTP secara tegas dengan cara yang konsisten dengan menetapkan pemetaan operasi CRUD (*create, read, update, delete*), sebagai berikut (Pautasso & Wilde, 2010):

1. Untuk melakukan *create resource* atau mengirimkan data ke server, menggunakan POST.
2. Untuk melakukan *read resource* atau mengambil data dari server, menggunakan GET.
3. Untuk melakukan *update resource* atau mengubah data, menggunakan PUT.
4. Dan untuk melakukan penghapusan data atau *delete resource*, menggunakan DELETE.

2.2.10 Application Programming Interface (API)

API (*Application Programming Interface*) adalah penjelasan layanan-layanan yang tersedia didalam sebuah sistem atau aplikasi melalui sebuah dokumentasi pengembangan perangkat lunak yang berfungsi sebagai panduan kepada pengembang dalam mempelajari dan menggunakan firur yang disediakan (Sutrisno dkk., 2019).

API (*Application Programming Interface*) merupakan uraian *interface* suatu sistem atau aplikasi untuk bertukar data dari sistem satu ke sistem lain (Cited & Data, 2017).

2.2.11 RESTful API

RESTful API merupakan generalisasi *interface* yang telah didokumentasikan menggunakan metode API untuk memudahkan pengembang dalam memahami sistem yang sedang berjalan, serta menjadi panduan dalam penggunaan fitur yang tersedia pada sistem melalui internet menggunakan sistem *web service* yang terdistribusi (Sutrisno dkk., 2019).

Sumber daya atau *resource* merupakan jenis informasi yang dapat diakses dari sebuah aplikasi atau sistem, dapat berupa *object*, *database record*, *algorithm*, atau yang lainnya. Setiap *resource* diidentifikasi oleh sebuah URI (*Universal Resource Identifier*) yang unik dengan menggunakan metode HTTP GET, PUT, POST, DELETE, HEADER, dan OPTIONS, yang akan menghasilkan sebuah data untuk kemudian dikembalikan ke *client* (Chen dkk., 2017).

2.2.12 JavaScript Object Notation (JSON)

JSON atau *JavaScript Object Notation* merupakan struktur pertukaran data yang dikembangkan dengan bahasa pemrograman *JavaScript* serta tidak memiliki ketergantungan dengan bahasa pemrograman lain karena mengaplikasikan bahasa yang sering digunakan dalam pemrograman, selain itu *JSON* memungkinkan proses pertukaran data yang cepat, mudah ditulis dan dibaca karena memiliki format yang sederhana. Berikut adalah kelebihan format data *JSON* dibandingkan dengan *XML* (*eXtensible Markup Language*) (Dawood, 2017):

1. Penulisan format *JSON* lebih mudah dan terstruktur untuk data yang rumit dan kompleks.
2. Untuk data yang sama, ukuran karakter pada format *JSON* relatif lebih sedikit daripada *XML*, sehingga berpengaruh pada kecepatan transfer data.
3. *JSON* menggunakan *function eval()* *JavaScript* untuk menguraikan data, sedangkan *XML* menguraikan data menggunakan *XML HTTP Request*, sehingga penggunaan format *JSON* dirasa lebih sederhana.

2.2.13 Framework

Framework merupakan kumpulan kode yang sering digunakan dalam pembuatan aplikasi yang tersusun rapi pada sebuah folder sehingga *programmer* tidak perlu menuliskan kode program mulai dari awal, karena banyak hal yang telah disediakan oleh *framework* yang sudah siap untuk digunakan (Abdullah, 2017).

Dengan menggunakan *framework* atau kerangka kerja, akan memudahkan programmer dalam menuliskan kode program dengan cukup memanggil *library* atau fungsi yang sudah disediakan oleh *framework*, hal tersebut dapat mengakomodasi *developer* atau *programmer* dalam mengatasi persoalan dalam pemrograman, sehingga *programmer* cukup fokus pada proses membangun aplikasi (Yudhanto & Adi, 2018).

2.2.14 Laravel

Laravel merupakan kerangka kerja PHP *open-source* yang dibuat oleh Taylor Otwell untuk mengemangkan aplikasi web dengan mengikuti pola arsitektur MVC (*model-view-controller*), dengan menyediakan *authentication*, *routing*, *session manager*, *caching*, *IoC container*, *database migration*, serta *unit testing* yang terintegrasi untuk memberi pengembang kemampuan untuk membangun aplikasi yang kompleks dengan mudah (Chen dkk., 2017).

Dengan laravel proses modifikasi *database* dapat dilakukan dengan mudah menggunakan fitur migrasi yang telah disediakan, *migration* laravel juga mendukung beberapa basisdata seperti : *PostgreSQL*, *MySQL*, *SQLITE*, dan *MSSQL*, selain itu di *framework* laravel terdapat *Eloquent* yang dapat digunakan untuk mengimplementasikan *Record* aktif menggunakan standar *Object Oriented Programming* (Luthfi, 2017).

Laravel memiliki beberapa keunggulan dibandingkan dengan *framework* lain, diantaranya (Abdullah, 2017):

1. Banyak fitur pada Laravel yang tidak disediakan oleh *framework* lain.
2. Struktur penulisan pada laravel mudah dipahami oleh programmer pemula sekalipun.
3. Dokumentasi yang lengkap pada setiap versi.

4. Banyak *library* yang mendukung, karena laravel banyak digunakan oleh *programmer*.
5. *Library-library* laravel didukung oleh *composer*.
6. Memudahkan programmer dalam menampilkan data karena memiliki *template engine* sendiri yang disebut dengan *blade*.

2.2.15 Dart

Dart adalah bahasa pemrograman yang memiliki kemiripan dengan dengan bahasa pemrograman lain seperti java dan javascript, karena bahasa pemrograman dart diciptakan oleh google untuk menggantikan bahasa pemrograman javascript yang dirilis pertama kali pada tahun 2011, dart menerapkan konsep *static typing* dimana programmer harus mendefinikan terlebih dahulu variabel yang akan digunakan (Tjandra & Chandra, 2020).

Dart adalah bahasa pemrograman *object oriented* yang menggunakan gaya penulisan bahasa C yang terkompilasi secara opsional kedalam bahasa pemrograman Javascript, sehingga memudahkan *programmer* yang pernah menggunakan bahasa pemrograman java (Suryono & Hardiansah, 2020).

2.2.16 SDK (Software Development Kit)

SDK atau *Software Development Kit* merupakan *API (Application Programming Interface)* yang berfungsi sebagai *emulator* yang digunakan untuk menjalankan aplikasi tanpa harus melakukan *compile* ke dalam format apk dalam mengembangkan aplikasi berbasis android (RIZQI, 2014).

SDK merupakan *Application Programming Interface (API)* yang digunakan dalam melakukan pengujian aplikasi android, selain itu *SDK* merupakan alat bantu yang dibutuhkan dalam mengembangkan aplikasi menggunakan bahasa pemrograman java pada *platform* Android (Mubarak, 2017).

2.2.17 Flutter

Flutter adalah *Software Development Kit* untuk membangun aplikasi mobile android dan iOS yang memiliki kinerja tinggi hanya dengan satu basis code *open source* yang dibuat oleh google, sehingga programmer dapat menghadirkan aplikasi dengan berkinerja tinggi pada berbagai jenis *platform* (Tjandra & Chandra, 2020).

Flutter merupakan sebuah *toolkit/framework* yang dibuat dan dikembangkan oleh google untuk membuat aplikasi *multi-platform* baik mobile, web, ataupun desktop dari sebuah basis code, selain itu flutter juga menawarkan keunggulan yaitu *fast development* (proses pengembangan cepat), *expressive and flexible UI* (menawarkan tampilan yang cantik), serta *native performance* dan fitur *hot reload* yang ditawarkan dapat membantu dalam proses pembuatan *user interface* (Suryono & Hardiansah, 2020).

2.2.18 AVD (*Android Virtual Devices*)

AVD merupakan *emulator* yang dijadikan sebagai tempat pegujian aplikasi berbasis android yang berjalan pada *virtual Machine* (Mubarak, 2017).

AVD (*Android Virtual Devices*) merupakan bagian dari SDK Android yang berjalan pada *virtual machine* sebagai *emulator* untuk menjalankan dan melakukan pengujian aplikasi berbasis android (Yuntoto, 2015).

2.2.19 JDK (*Java Development Kit*)

JDK (*Java Development Kit*) merupakan *software* yang untuk melakukan proses kompilasi kedalam *bytecode* sehingga kode program dapat dipahami untuk dijalankan pada *JRE (Java Runtime Envirotment)*, komputer yang akan mengembangkan aplikasi berbasis java harus menginstal JDK terlebih dulu sebelum memulai membuat aplikasi, tetapi untuk komputer yang akan menjalankan aplikasi tidak wajib menginstal JDK (Mubarak, 2017).

2.2.20 Black Box Testing

Black box testing merupakan teknik pengujian aplikasi yang hanya menguji dari segi fungsionalitas aplikasi untuk mengetahui struktur internal yang tidak sesuai dengan cara kerja sistem (Mubarak, 2017).

Black box testing adalah cara menguji sistem yang hanya fokus pada spesifikasi fungsionalitas untuk menemukan kesalahan atau ketidaksesuaian fungsi, kesalahan dalam tampilan, kesalahan struktur dan akses database, kesalahan terminasi dan inisialisasi, ketidaksesuaian performa aplikasi, dengan tujuan memecahkan masalah-masalah berikut (Mustaqbal dkk., 2015):

1. Kesesuaian fungsi-fungsi yang diuji.
2. Menentukan input yang baik untuk bahan pengujian aplikasi.
3. Tingkat sensitifitas aplikasi dalam menerima sebuah inputan.
4. Banyaknya data yang mampu ditangani oleh sistem.
5. Cara mengisolasi dan membuat kombinasi data.