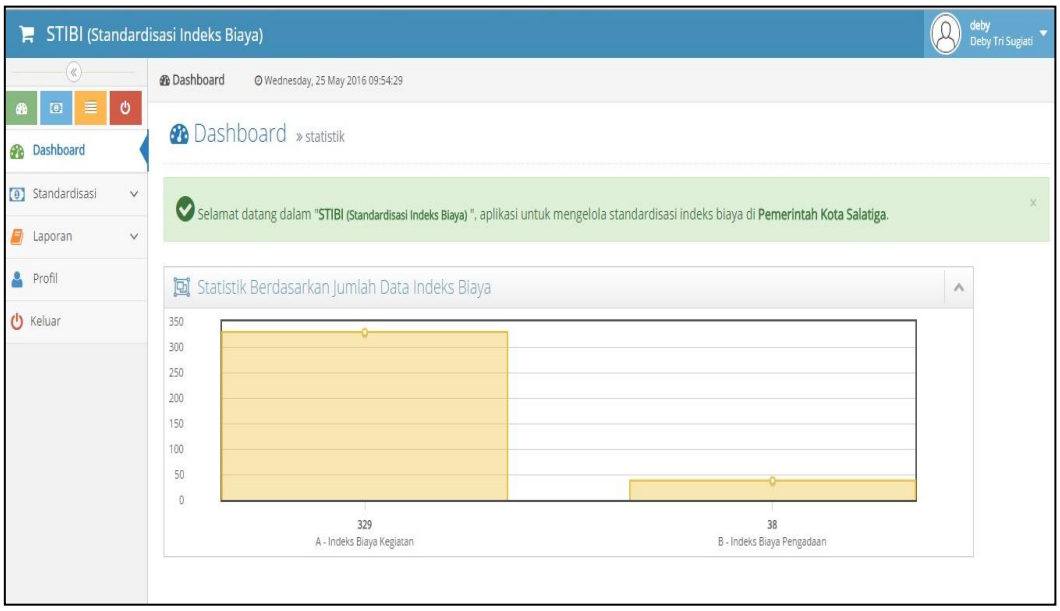
BAB II
LANDASAN TEORI

2.1 Tinjauan Pustaka

Berikut ini Tinjauan Pustaka dari Sistem Informasi Standardisasi Indeks Biaya (STIBI) Pemerintah Kota Salatiga. Hal-hal yang dijelaskan yaitu :

2.1.1 Sistem Informasi Standardisasi Indeks Biaya (STIBI) Pemerintah Kota Salatiga

STIBI Pemerintah Kota Salatiga merupakan salah satu implementasi perkembangan teknologi informasi. *Website* Kota Salatiga adalah *website* yang menampilkan semua informasi dan pemberitaan juga kegiatan Pemerintah Kota Salatiga. dimana STIBI termasuk salah satu komponen di dalamnya. Berikut beberapa contoh tampilan STIBI, terdapat dalam Gambar 2.1 yang memperlihatkan tampilan awal pada *User* Operator dan *Admin/ Dashboard* yang dapat muncul ketika sudah melakukan *Login* berisi komponen-komponen utama STIBI.



Gambar 2.1. Halaman Tampilan Awal Operator dan *Admin* beserta komponen-komponen utama

Pada Gambar 2.2 menggambarkan tampilan awal *User* Publik atau Perangkat Daerah, berfungsi sebagai pencarian data kebutuhan Perangkat Daerah akan Standardisasi Indeks Biaya.

Gambar 2.2 Halaman tampilan awal publik

Berikut adalah spesifikasi dari perancangan STIBI Pemerintah Kota Salatiga adalah sebagai berikut :

1) Berbasiskan *web* sehingga bisa diakses dimana saja dan kapan saja.
2) Tampilan aplikasi yang mudah dikenal dan digunakan, karena seperti windows.
3) Lebih Handal , ringan dan mudah untuk dipelajari oleh *User* .
4) Tingkat keamanan aplikasi yang terjamin.
5) Memiliki fitur-fitur yang lengkap sebagai berikut :
   a) Indeks Biaya Kegiatan
   b) Indeks Biaya Pengadaan
   c) Indeks Honorarium
   d) Fungsi-fungsi pencarian terpetakan dalam aplikasi
   e) Profil *User*
6) Spek Teknis : *Server* (*Server* version: 10.1.13-MariaDB, Apache 2.4.17, PHP version: 5.6.23) , *Client* (*Browser Chrome* dan *Modzilla*).

2.1.2 Kajian Pustaka

Pemanfaatan teknologi informasi dan komunikasi dalam proses pemerintahan (*e-government*) dapat meningkatkan efisiensi, efektifitas, transparansi, dan akuntabilitas penyelenggaraan pemerintahan. Untuk menerapkan

e-government tersebut, diperlukan komponen-komponen pendukung yang dapat meningkatkan percepatan implementasinya. Salah satu komponen tersebut adalah STIBI.

Badie Uddin dan Rafika Yuni (2015: 83-88) melakukan penelitian tentang pengembangan Sistem Informasi pengelolaan arsip digital berbasis *Web* yang disebut dengan SIPAD, yang merupakan studi kasus pada Bagian Kepegawaian Politeknik TEDC Bandung. Pada penelitian tersebut penyimpanan arsip dalam bentuk fisik yang disimpan dalam rak *(filling cabinet)* membutuhkan ruang penyimpanan yang luas sehingga sulit menemukan kembali dokumen saat dibutuhkan. Dengan adanya SIPAD ini dapat membantu dalam proses pencarian untuk mendapatkan arsip dokumen yang sesuai dengan kebutuhan informasi dengan lebih cepat.

Budi Haryanto, dkk (2015: 109-115) melakukan penelitian dalam studi kasus di Pemerintah Kota Salatiga tentang perancangan sistem e-Office Pemerintah Daerah, menyatakan bahwa e-Office memberikan beberapa keuntungan yaitu *paperless* (mengurangi penggunaan kertas), *fast delivery* (efisien dalam distribusi surat dari segi waktu, tenaga, dan biaya), *easy tracking* (mempermudah penelusuran surat), dan *web based* (berbasis *web* sehingga dapat membuat disposisi dan laporan tindak lanjut menggunakan berbagai perangkat yang terhubung internet).

Agus Dwiyanto (2014: 202) dalam bukunya *Mewujudkan Good Governance Melalui Pelayanan Publik* menjelaskan bahwa mempersiapkan teknologi yang dapat mendukung implementasi kegiatan birokrasi. Tersedianya *Information Technology* (IT), yang dapat digunakan untuk mendukung implementasi kegiatan birokrasi memberikan peluang untuk berhasil lebih besar. Selain itu penggunaan teknologi dalam jangka panjang juga akan mengurangi biaya yang harus dikeluarkan oleh Pemerintah, dimana dengan adanya fasilitas internet Pemerintah Daerah dapat memfasilitasi berbagai kegiatan seperti pemberian informasi, keluhan, diskusi publik dan survey masyarakat.

2.2 Kerangka Pemikiran

Berikut adalah kerangka pemikiran yang dijalankan dalam penelitian ini :

1) Latar belakang masalah

   Tahapan paling awal, yakni menelusuri latar belakang kenapa masalah yang akan diangkat menjadi penting untuk dipilih.

2) Rumusan masalah

   Penyimpulan latar belakang masalah menjadi suatu rumusan masalah yang akan diangkat untuk menjadi bahan penelitian.

3) Penguasaan dasar menggunakan Laravel, PHP dan MariaDB

   Tahap untuk mempelajari dasar-dasar Laravel, PHP dan MariaDB agar lebih menguasai program – program yang akan digunakan untuk membangun sistem.

4) Pengumpulan Data

   Pengumpulan data dilakukan baik dengan tanya – jawab (*interview*), survey lapangan,hasil rekomendasi, usulan kebutuhan Perangkat Daerah.

5) Observasi aplikasi Laravel, PHP dan MariaDB

   Merupakan tahap pengamatan sampel – sampel aplikasi yang telah ada, jurnal, buku, maupun karya ilmiah untuk kajian yang dapat dijadikan referensi untuk pembangunan sistem.

6) Analisis dan perancangan aplikasi terstruktur

7) Implementasi Sistem Informasi Standardisasi Indeks Biaya (STIBI) Kota Salatiga

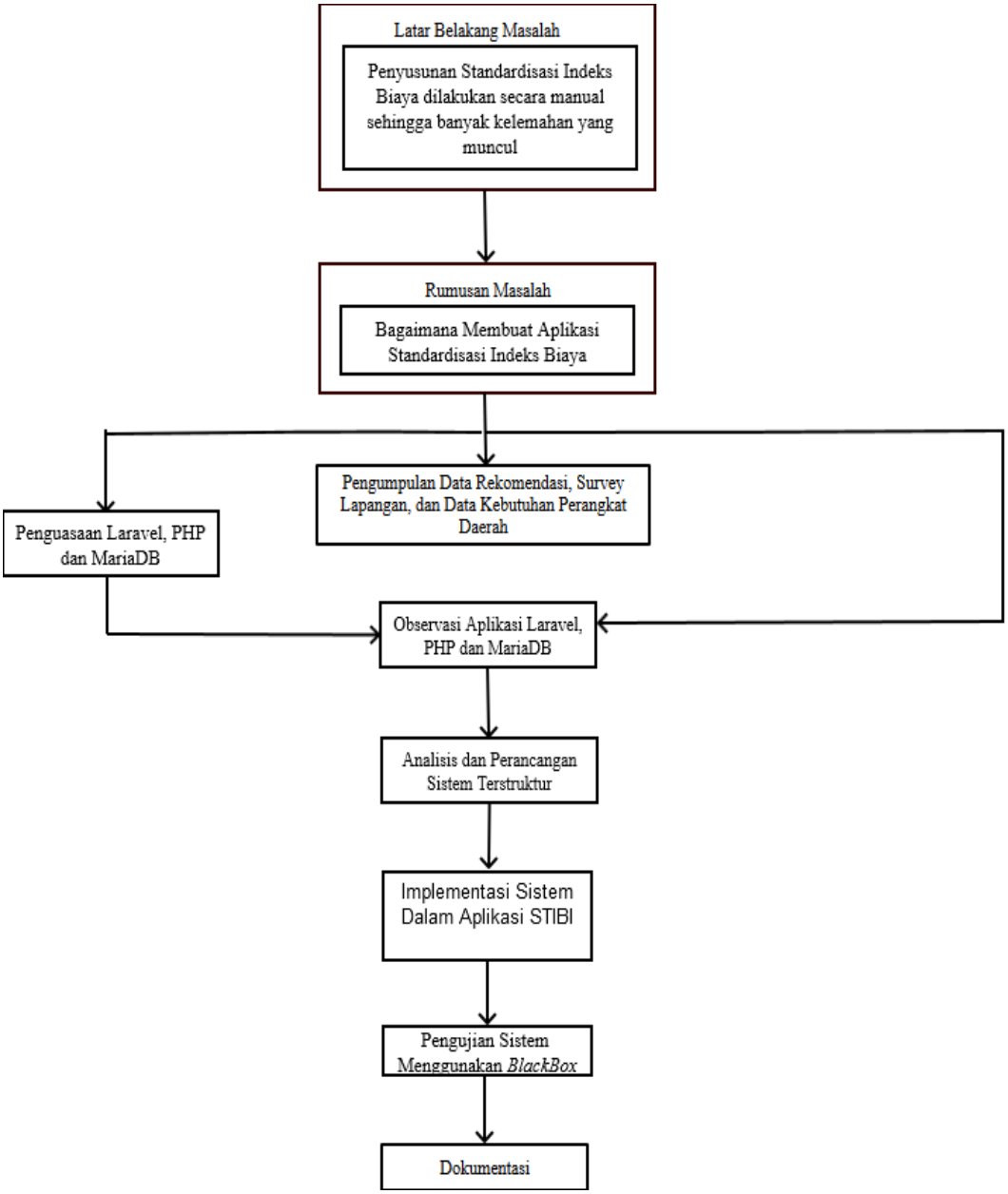
8) Pengujian Sistem

   Pengujian sistem akan dilakukan pada beberapa komputer untuk mengetahui jika ada kesalahan dan kekurangan pada sistem.

9) Dokumentasi

   Tahapan terakhir, yakni tahap pendokumentasian seluruh poses penyususnan tugas akhir ke dalam laporan.

Kerangka pemikiran dalam penelitian ini tergambar pada Gambar 2.3.



Gambar 2.3. Kerangka Pemikiran

2.3 Teori Pendukung

Berikut beberapa teori pendukung dari perancangan STIBI Kota Salatiga.

2.3.1 Standardisasi

Standardisasi Indeks Biaya berdasarkan Peraturan Walikota Salatiga Nomor 38 tahun 2014 tentang Standardisasi Indeks Biaya Tahun Anggaran 2015 adalah patokan harga tertinggi belum termasuk pajak untuk menentukan besaran harga barang sesuai jenis, spesifikasi, dan kualitas dalam 1 (satu) periode tertentu.

Dalam hal ini Standardisasi memfasilitasi kebutuhan Perangkat Daerah akan Pengadaan barang, Perjalanan Dinas/ Kegiatan dan Honorarium, sesuai dengan keadaan yang sebenarnya berpedoman Peraturan Menteri Keuangan serta berasaskan:

1) kepatutan adalah tindakan atau suatu sikap yang dilakukan dengan wajar dan proporsional.
2) ekonomis merupakan hasil pendataan dengan kualitas dan kuantitas tertentu pada tingkat harga yang terendah.
3) manfaat adalah pengelolaan keuangan daerah diutamakan untuk pemenuhan kebutuhan

Standardisasi Indeks Biaya dari tahun ke tahun penyusunannya dicetak menjadi Buku Standardisasi Indeks Biaya yang menjadi Pedoman Perangkat Daerah dalam menyusun rencana anggaran kegiatan.

2.3.2 Sistem Informasi

Abdul Kadir (2003) mendefinisikan sistem melalui dua kelompok pendekatan yaitu yang menekankan pada prosedurnya dan yang menekankan pada komponen atau elemennya. Berikut definisi sistem berdasarkan pendekatan sistem prosedur sebagaimana diungkapkan oleh Abdul Kadir dalam bukunya Pengenalan Sistem Informasi:

"Sistem adalah suatu jaringan kerja dari prosedur-prosedur yang saling berhubungan, berkumpul bersama-sama untuk melakukan suatu kegiatan atau untuk menyelesaikan suatu sasaran tertentu."

Sedangkan berdasarkan pendekatan sistem elemen atau komponennya, Abdul Kadir dalam bukunya Pengenalan Sistem Informasi mendefiniskan sistem sebagai:

"Sistem adalah kumpulan dari elemen-elemen yang berinteraksi untuk mencapai suatu tujuan tertentu."

Dari definisi diatas terihat bahwa sistem mempunyai karakteristik atau sifat-sifat yang tertentu, yaitu :

1) Komponen-komponen sistem (*Components*)

2) Batas sistem (*Boundary*)
3) Lingkungan luar sistem (*Environment*)
4) Penghubung sistem (*Interface*)
5) Masukan sistem (*Input*)
6) Keluaran sistem (*Output*)
7) Pengolah sistem (*Process*)
8) Sasaran sistem (*Goal*)

Menurut Abdul Kadir (2002 : 54) ada beberapa elemen yang membentuk sebuah sistem, yaitu Tujuan, Masukan, Keluaran, Proses, Mekanisme pengendalian dan umpan balik dan Batasan.

Informasi adalah sebuah istilah yang tidak tepat dalam pemakaiannya secara umum. Informasi dapat mengenai data mentah, data tersusun, kapasitas sebuah saluran komunikasi, dan lain sebagainya (Tata Sutabri, 2005:23-24).

Informasi adalah data yang telah diklasifikasikan atau diolah atau diinterprestasi untuk digunakan dalam proses pengambilan keputusan. Bila tidak ada pilihan atau keputusan, maka informasi menjadi tidak diperlukan. Teori informasi lebih tepat disebut teori matematis, komunikasi juga memberikan beberapa pandangan yang berguna bagi sistem informasi manajemen.

Sistem informasi adalah suatu sistem didalam suatu organisasi yang mempertemukan kebutuhan pengolahan transaksi harian yang mendukung fungsi operasi organisasi yang bersifat manajerial dengan kegiatan strategi dari suatu organisasi untuk dapat menyediakan kepada pihak luar tertentu dengan laporan-laporan yang diperlukan (Tata Sutabri, 2005 : 42).

Pengertian sistem informasi menurut Abdul Kadir (2003) Sistem Informasi adalah ***suatu sistem di dalam suatu organisasi yang mempertemukan kebutuhan pengolahan transaksi harian, mendukung operasi, bersifat managerial dan kegiatan strategi dari suatu organisasi dan menyediakan pihak luar tertentu dengan laporan-laporan yang diperlukan.*** Komponen-komponen sistem informasi adalah :

1) *Hardware*, terdiri dari komputer, printer dan jaringan.

2) *Software*, merupakan kumpulan dari perintah atau fungsi yang ditulis dengan aturan tertentu untuk memerintahkan komputer melaksanakan tugas tertentu.
3) Data, merupakan komponen dasar dari informasi yang akan diproses lebih lanjut untuk menghasilakan informasi.
4) Prosedur, dokumentasi prosedur atau proses sistem, buku penuntun operasional (aplikasi) dan teknis.
5) Manusia, yang terlibat dalam komponen manusia adalah seperti operator, pemimpin sistem informasi, dan sebagainya.

Kegiatan yang dilakukan dalam sebuah sistem informasi adalah :

1) *Input*, menggambarkan suatu kegiatan untuk menyediakan data untuk diproses.
2) Proses, menggambarkan bagaimana suatu data diproses untuk menghasilkan suatu informasi yang bernilai tambah.
3) *Output*, menggambarkan suatu kegiatan untuk menghasilkan laporan dari proses di atas tersebut.
4) Penyimpanan, menggambarkan suatu kegiatan untuk memelihara dan menyimpan data.

Menurut George M. Scott, perancangan sistem dapat didefinisikan sebagai berikut : "Menentukan bagaimana suatu sistem akan menyelesaikan apa yang mesti diselesaikan. Tahap ini menyangkut mengkonfigurasi dan komponen-kompoenen perangkat keras dan lunak dari suatu sistem sehingga setelah instalasi dari sistem akan benar-benar memuaskan rancangan bangunan yang telah ditetapkan pada akhir tahap analsis sistem".

Perancangan sistem berarti menyusun sistem yang digunakan maksud-maksud tertentu dalam mencapai tujuan, biasanya sistem tersebut memepunyai kelebuhan diantaranya lebih efisien, akurat, tepat waktu dan relevan.

Perancangan sistem dibuat adalah dengan maksud atau tujuan adalah :

1. Untuk lebih memahami alur sebuah sistem
2. Memenuhi kebutuhan kepada pemakai sistem
3. Memberikan gambaran yang jelas dan rancang bangun yang lengkap kepada *programmer* dan ahli teknik lainnya yang terlibat di dalamnya.

2.3.3. *Unified Modelling Language* (UML)

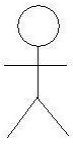
UML (*Unified Modelling Language*) merupakan bahasa yang telah menjadi standar dalam industri untuk visualisasi, merancang, dan mendokumentasikan sistem piranti lunak. UML menawarkan sebuah standar untuk merancang model sebuah sistem (Shalahuddin & Rosa, 2011).

Berdasarkan pendapat yang dikemukakan di atas dapat ditarik kesimpulan bahwa UML adalah sebuah bahasa yang berdasarkan grafika atau gambar untuk menvisualisasikan, menspesifikasikan, membangun dan pendokumentasian dari sebuah system pengembangan perangkat lunak berbasis Objek (*Object Oriented programming*). Berikut ini jenis-jenis diagram yang digunakan dalam *Unified Modeling Language*:

2.3.3.1. *Use Case Diagram*

Use Case Diagram menggambarkan interaksi antara sistem, system eksternal dan pengguna. Dengan kata lain *Use Case* diagram secara grafis mendeskripsikan siapa yang akan menggunakan system dan dalam cara apa pengguna (*User)* mengharapkan interaksi dengan system itu. *Use Case* secaran aratif digunakan untuk secara tekstual menggambarkan sekuensi langkah-langkah dari setiap interaksi. Berikut simbol-simbol yang digunakan pada *Use Case Diagram* dijelaskan pada Tabel 2.1.

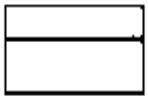
Tabel 2.1. Simbol *Use Case* Diagram

Simbol	Keterangan
	*Actor;* Sebuah peran yang dimainkan oleh seseorang, sistem, atau perangkat yang memiliki saham dalam keberhasilan operasi dari sistem.
	*Use Case*; Untuk mengungkapkan tujuan bahwa sistem harus dicapai.
	Association; Mengidentifikasi interaksi antara aktor dan *Use Case*
	Dependency; Mengidentifikasi hubungan komunikasi antara dua *Use Case.*

2.3.3.2. *Class Diagram*

Menurut Rosa AS dan M Shalahudin (2013), *Class Diagram* atau diagram kelas merupakan suatu diagram yang menggambarkan struktur sistem dari segi pendefinisian *class* yang akan dibuat untuk membangun sistem. *Class* memiliki apa yang disebut nama (dan *stereotype*), atribut dan mengandung metode atau operasi. Atribut merupakan variable-variabel yang dimiliki oleh suatu *class*. Operasi atau metode adalah fungsi-fungsi yang dimiliki oleh suatu *class*. Dan untuk simbol-simbolnya *Class Diagram* kurang lebih digambarkan pada Tabel 2.2.

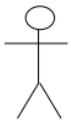
Tabel 2.2. Simbol *Class Diagram*

Simbol	Nama	Keterangan
	*Generalization*	Hubungan di mana objek anak berbagi perilaku dan struktur data dari objek yang di atasnya yaitu objek induk (ancestor)
	*Class*	Himpunan dari objek-objek yang berbagi atribut serta operasi yang sama
	Association	Relasi antar kelas dengan makna umum, asosiasi biasanya juga disertai dengan *multiplicity*.
	*Collaboration*	Interaksi aturan dan elemen lain yang bekerja sama untuk menyediakan perilaku yang lebih besar dari jumlah dan elemennya
	*Realization*	Operasi yang benar-benar dilakukan oleh suatu objek
	Dependency	Hubungan di mana perubahan yang terjadi pada suatu elemen mandiri akan mempengaruhi elemen yang bergantung padanya

2.3.3.3. *Sequence Diagram*

Sequence Diagram menurut Rosa AS dan M Shalahudin (2013), *Sequence Diagram* menjelaskan secara detail urutan proses yang dilakukan dalam sistem untuk mencapai tujuan dari *Use Case*: interaksi yang terjadi antar *class*, operasi apa saja yang terlibat, urutan antar operasi, dan informasi yang diperlukan oleh masing-masing operasi. Pembuatan *Sequence Diagram* merupakan aktivitas yang paling kritikal dari proses desain karena artifak inilah yang menjadi pedoman dalam proses pemrograman nantinya dan berisi *control flow* dari program. Berikut adalah simbol-simbol yang digunakan pada *Sequence Diagram* pada Tabel 2.3.

Tabel 2.3. Simbol *Sequence Diagram*

Simbol	Nama	Keterangan
 Nama Aktor	*Actor*	Orang, proses atau sistem lain yang berinteraksi dengan sistem informasi lain diluar sistem informasi itu sendiri; biasanya dinyatakan menggunakan kata benda di awal frase nama actor
	Lifeline	Menyatakan kehidupan suatu objek
Nama objek	*Objek*	Menyatakan objek yang berinteraksi
Message() 	*Message*	Spesifikasi dari komunikasi antar objek yang memuat informasi tentang aktifitas yang terjadi
	*Boundary*	Digunakan untuk menggambarkan form
	*Control Class*	Digunakan untuk menghubungkan boundary dengan tabel pada database
	*Entity Class*	Digunakan untuk menggambarkan hubungan kegiatan yang akan dilakukan
X	*Pesan tipe destroy*	Menyatakan akhir hidup suatu objek

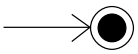
2.3.3.4. *State Chart Diagram*

Statechart diagram menggambarkan transisi dan perubahan keadaan dari satu *state* ke *state* lainnya dalam suatu objek pada sistem sebagai akibat dari stimuli yang diterima.

Pada umumnya *statechart* diagram menggambarkan *class* tertentu pada satu *class* dapat memiliki lebih dari satu *statechart* diagram. Transisi antar *state* umumnya memiliki kondisi *guard* yang merupakan syarat terjadinya transisi yang bersangkutan, dituliskan dalam kurung siklus.

Action atau aktifitas yaitu proses menjalankan atau membuat *state* berubah yang dilakukan sebagai akibat dari *event*. *Event* adalah penyebab terjadinya perubahan tertentu. Berikut ini simbol-simbol beserta fungsi dari statechart diagram tergambar pada Tabel 2.4.

Tabel 2.4. Simbol *State Chart Diagram*

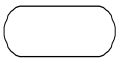
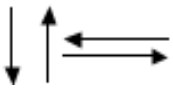
NO	GAMBAR	NAMA	KETERANGAN
1.		*States*	*States* merepresentasikan keadaan *"life"* sebuah objek selama objek tersebut ada.
2.		*Transition*	Suatu anak panah yang tebal merepresentasikan jalan antara state yang berlainan dari sebuah objek.
3.		*Initial Pseudo State*	*Initial pseudo state* digunakan untuk memulai *statechart diagram*.
4.		*Final State*	*Final State* digunakan untuk mengakhiri diagram *statecahart*.

2.3.3.5. *Activity Diagram*

Activity Diagram digunaka nuntuk menggambarkan rangkaian aliran aktivitas baik proses bisnis maupun *Use Case*. *Activity Diagram* dapat juga digunakan untuk memodelkan *action* yang akan dilakukan saat sebuah operasi

dieksekusi, dan memodelkan hasil dari action tersebut.Berikut ini menggambarkan *Activity Diagram* pada Tabel 2.5.

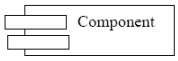
Tabel 2.5. Simbol *Activity* Diagram

NO	GAMBAR	NAMA	KETERANGAN
1.		*Action states*	*Action state* adalah representasi/ gambaran dari aksi yang tidak bisa diganggu oleh aksi yang berasal dari objek-objek. *Action state* digambarkan dalam bentuk empat persegi panjang yang pada sudut-sudutnya melingkar.
2.		*Initial node*	Menunjukkan bagaimana objek dibentuk/ diawali
3.		*Activity Final*	Menunjukkan bagaimana objek bentukan diakhiri
4.		*Decision*	Dugunakan untuk menggambarkan suatu keputusan / tindakan yang harus diambil pada kondisi tertentu
5.		*Action Flow*	*Action* digambarkan dalam bentuk anak panah yang mengilustrasikan relasi antara *action* pada *state*
6.		*Fork Node*	Satu aliran yang pada tahap tertentu berubah menjadi beberapa aliran

2.3.3.6. *Component Diagram*

Component Diagram atau komponen diagram digunakan untuk menggambarkan organisasi dan ketergantungan komponen-komponen software sistem. Komponen diagram dapat digunakan untuk menunjukan bagaimana kode pemrograman dibagi menjadi modul-modul atau komponen. Simbol pada *Component Diagram* dapat dilihat pada Tabel 2.6.

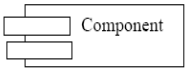
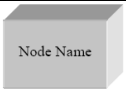
Tabel 2.6 Simbol *Component Diagram*

NO	SIMBOL	NAMA	KETERANGAN
1		*Component*	melambangkan sebuah entitas *software* dalam sebuah sistem.
2		*Dependency*	Sebuah *Dependency* digunakan untuk menotasikan relasi antara dua komponen.

2.3.3.7. *Deployment Diagram*

Deployment Diagram atau diagram pengurai digunakan untuk mendeskripsikan arsitektur fisik dalam istilah ”node” untuk *hardware* dan software dalam sistem. Diagram ini menggambarkan konfigurasi komponen-komponen *software real-time*, prosesor, dan peralatan yang membentuk arsitektur sistem.Simbol pada *Deployment Diagram* dapat dilihat pada Tabel 2.7.

Tabel 2.7 Simbol *Deployment Diagram*

NO	GAMBAR	NAMA	KETERANGAN
1		*Component*	*Component* yang ada diletakkan didalam node untuk memastikan keberadaan posisi mereka
2		*Node*	menggambarkan bagian-bagian *hardware* dalam sebuah sistem.
3		*Association*	menghubungkan dua *node* yang mengindikasikan jalur komunikasi antara element-elemen *hardware*.

2.3.4 *Hypertext Preprocessor* (PHP)

PHP yang merupakan singkatan dari *Hypertext Preprocessor*, merupakan *server-side Programming* , yaitu bahasa pemrograman yang diproses di sisi *server* (Rohi Abdullah, 2015:2).

PHP bekerja di dalam sebuah dokumen HTML *(Hypertext Markup Language)* untuk dapat menghasilkan isi dari sebuah halaman *web* sesuai permintaan. Dengan PHP kita dapat merubah situs kita menjadi sebuah aplikasi berbasis *web*, tidak lagi hanya sekedar sekumpulan halaman statis yang jarang diperbaharui.

Mengapa PHP? Karena PHP bersifat tidak memiliki ketergantungan terhadap berbagai *platform*, jadi PHP dapat dijalankan dalam *platform* apapun, baik itu Unix, Windows ataupun Macintosh.

Kelebihan lain dari PHP adalah kemudahan melakukan pengkodean, karena perintah-perintah PHP mirip dengan perintah-perintah C. Selain itu kemudahan dari PHP adalah dapat dengan mudah dihubungkan dengan aplikasi *database* (melakukan *query*), seperti MySQL dan PostgreSQL.

PHP bersifat *free* (bebas dipakai). Kita tidak perlu membayar apapun untuk menggunakan perangkat lunak ini. Kita dapat mendownload PHP melalui situs resminya yaitu *www.php.net.* Untuk versi Windows, kita dapat memperoleh kode binernya, dan untuk versi Linux, kita mendapatkan kode sumbernya secara lengkap.

2.3.5 *Bootsrap*

Bootstrap adalah sebuah framework CSS yang menyediakan kumpulan komponen-komponen antarmuka dasar pada *web* yang telah dirancang sedemikian rupa untuk digunakan bersama-sama. Selain komponen antarmuka, *Bootstrap* juga menyediakan sarana untuk membangun layout halaman dengan mudah dan rapi, serta modifikasi pada tampilan dasar HTML untuk membuat seluruh halaman *web* yang dikembangkan senada dengan komponen-komponen lainnya. *Bootstrap* dibuat untuk memberikan sekumpulan perangkat yang dapat digunakan untuk membangun *website* sederhana dengan mudah.

File-file yang dibutuhkan untuk menggunakan Bootstrap dengan isi dari masing-masing direktori yaitu:

1) Direktori “css” memiliki empat buah file di dalamnya, yaitu:
    a) bootstrap.css
    b) bootstrap.min.css
    c) bootstrap-responsive.css
    d) bootstrap-responsive.min.css
2) Direktori “img memiliki dua buah file di dalamnya, yaitu:
    a) glyphicons-halflings.png

b) glyphicons-halflings-white.png

3) Direktori “js” memiliki dua buah file di dalamnya, yaitu:
   a) bootstrap.js
   b) bootstrap.min.js

2.3.6 Framework Laravel

Framework adalah komponen pemprograman yang siap *re-use* (bisa digunakan ulang) kapan saja, sehingga programmer tidak harus membuat skrip yang sama untuk tugas yang sama. Misalkan kita ingin membuat halaman-halaman *web* yang menampilkan data dengan paginasi (paging) halaman, framework telah menyediakan fungsi paging tersebut sedangkan programmer cukup menggunakan fungsi tersebut pada saat *coding*, tetapi tentu dengan kaidah-kaidah yang ditetapkan oleh masing-masing *framework* (Akhmad Dharma Kasman, 2015).

Laravel adalah *framework* PHP MVC *(Model View Controller)* yang dikembangkan oleh Taylor Otwell pada tahun 2011 dan sekarang telah mencapai versi 5.2. Banyak sekali fitur-fitur yang sangat membantu kita dalam *framework* laravel ini.

2.3.7 MariaDB

MariaDB adalah sebuah implementasi dari sistem manajemen basis data relasional (RDBMS) yang didistribusikan secara gratis di bawah lisensi GPL (*General Public License*). Setiap pengguna dapat secara bebas menggunakan MariaDB, namun dengan batasan perangkat lunak tersebut tidak boleh dijadikan produk turunan yang bersifat komersial.

MariaDB merupakan versi pengembangan terbuka dan mandiri dari MySQL. Sejak diakuisisinya MySQL oleh Oracle pada September 2010, Monty Program sebagai penulis awal kode sumber MySQL memisahkan diri dari pengembangan dan membuat versi yang lebih mandiri yakni MariaDB

Sebagai *database server* yang memiliki konsep database modern, semua kemampuan MySQL dimiliki pula oleh MariaDB. Keistemewaan MariaDB antara lain:

1) Portabilitas. MariaDB dapat berjalan stabil pada berbagai sistem operasi seperti Windows, Linux, FreeBSD, Mac Os X *Server*, Solaris, Amiga, dan masih banyak lagi.
2) Perangkat lunak sumber terbuka. MariaDB didistribusikan sebagai perangkat lunak sumber terbuka, di bawah lisensi GPL sehingga dapat digunakan secara gratis.
3) *Multi-User* . MariaDB dapat digunakan oleh beberapa pengguna dalam waktu yang bersamaan tanpa mengalami masalah atau konflik.
4) *'Performance tuning'*, MariaDB memiliki kecepatan yang menakjubkan dalam menangani query sederhana, dengan kata lain dapat memproses lebih banyak SQL per satuan waktu.
5) Ragam tipe data. MariaDB memiliki ragam tipe data yang sangat kaya, seperti *signed / unsigned integer, float, double, char, text, date, timestamp*, dan lain-lain.
6) Perintah dan Fungsi. MariaDB memiliki operator dan fungsi secara penuh yang mendukung perintah *Select* dan *Where* dalam perintah (*query*).
7) Keamanan. MariaDB memiliki beberapa lapisan keamanan seperti level *subnetmask*, nama *host*, dan izin akses *User* dengan sistem perizinan yang mendetail serta sandi terenkripsi.
8) Skalabilitas dan Pembatasan. MariaDB mampu menangani basis data dalam skala besar, dengan jumlah rekaman (*records*) lebih dari 50 juta dan 60 ribu tabel serta 5 miliar baris. Selain itu batas indeks yang dapat ditampung mencapai 32 indeks pada tiap tabelnya.
9) Konektivitas. MariaDB dapat melakukan koneksi dengan klien menggunakan protokol TCP/IP, Unix soket (UNIX), atau Named Pipes (NT).

10) Pelokalan Bahasa. MariaDB dapat mendeteksi pesan kesalahan pada klien dengan menggunakan lebih dari dua puluh bahasa. Meski pun demikian, bahasa Indonesia belum termasuk di dalamnya.
11) Antar Muka. MariaDB memiliki antar muka (*interface*) terhadap berbagai aplikasi dan bahasa pemrograman dengan menggunakan fungsi API (*Application Programming Interface*).
12) Klien dan Peralatan. MariaDB dilengkapi dengan berbagai peralatan (*tool*) yang dapat digunakan untuk *Admin*istrasi basis data, dan pada setiap peralatan yang ada disertakan petunjuk online.
13) Struktur tabel. MariaDB memiliki struktur tabel yang lebih fleksibel dalam menangani ALTER TABLE, dibandingkan basis data lainnya semacam PostgreSQL ataupun Oracle.

2.3.8 Pengujian *Black Box*

Pengujian *black box* adalah menguji perangkat lunak dari segi spesifikasi fungsional tanpa menguji desain dan kode program. Pengujian dimaksudkan untuk mengetahui apakah fungsi-fungsi, masukan dan keluaran dari perangkat lunak sesuai dengan *spesifikasi* yang di butuhkan. Pengujian kotak hitam dilakukan dengan membuat kasus uji yang bersifat mencoba semua fungsi dengan memakai perangkat lunak apakah sesuai dengan spesifikasi yang dibutuhkan (Rosa &Shalahuddin, 2013).

Pengertian *Black Box Testing* adalah tipe *testing* yang memperlakukan perangkat lunak yang tidak diketahui kinerja internalnya. Sehingga para *tester* memandang perangkat lunak seperti layaknya sebuah kotak hitam yang tidak penting dilihat isinya, tapi cukup dikenai proses *testing* di bagian luar (Rizky, 2011).

Beberapa keuntungan yang diperoleh dari jenis testing ini antara lain:

1) Anggota tim *tester* tidak harus dari seseorang yang memiliki kemampuan teknis di bidang pemrograman.
2) Kesalahan dari perangkat lunak ataupun bug seringkali ditemukan oleh komponen *tester* yang berasal dari pengguna.

3) Hasil dari *black box testing* dapat memperjelas kontradiksi ataupun keracunan yang mungkin timbul dari eksekusi sebuah perangkat lunak.
4) Proses *testing* dapat dilakukan lebih cepat dibandingkan *white box testing*.

Beberapa teknik testing yang tergolong dalam tipe ini antara lain:

a) *Equivalence Partitioning*

Equivalence Partitioning melakukan pengelompokan data ke dalam grooup tertentu di setiap imputan data, kemudian dibandingkan dengan outpunya.

b) *Boundary Value Analysis*

Boundary Value Analysis merupakan teknik yang sangat umun digunakan pada saat awal sebuah perangkat lunak selesai dikerjakan. Pada teknik ini, dilakukan masukan yang melebihi dari batasan sebuah data, jika perangkat lunak berhasil mengatasi masukan yang salah, maka dapat dikatakan teknik ini telah selesai dilakukan.

c) *Cause Effect Graph*

Cause Effect Graph melakukan proses *testing* yang menghubungkan sebab dari sebuah masukan dan akibatnya pada output yang dihasilkan.

d) *Random Data Selection*

Random Data Selection adalah teknik yang berusaha melakukan proses inputan data yang menggunakan nilai acak. Dari hasil masukan tersebut kemudian dibuat sebuah tabel yang menyatakan validitas dari output yang dihasilkan.

e) *Feature Test*

Feature Test adalah teknik melakukan proses *testing* terhadap spesifikasi dari perangkat lunak yang telah selesai dikerjakan.