

BAB II

TINJAUAN PUSTAKA

2.1. Teori Umum

2.1.1. Pengertian Sistem

Menurut Romney dan Steinbart (2015), Sistem (*system*) adalah rangkaian dari dua atau lebih komponen-komponen yang saling berhubungan, yang berinteraksi untuk mencapai suatu tujuan. Sebagian besar sistem terdiri dari subsistem yang lebih kecil yang mendukung sistem yang lebih besar.

2.1.2. Pengertian Informasi

Menurut Romney dan Steinbart (2015), Informasi (*information*) adalah data yang telah dikelola dan diproses untuk memberikan arti dan memperbaiki proses pengambilan keputusan. Sebagaimana perannya, pengguna membuat keputusan yang lebih baik sebagai kuantitas dan kualitas dari peningkatan informasi.

2.1.3. Pengertian Sistem Informasi

Menurut Wikipedia, Sistem Informasi (*Information System*) adalah kombinasi dari teknologi informasi dan aktivitas orang yang menggunakan teknologi itu untuk mendukung operasi dan manajemen. Dalam arti yang sangat luas, istilah sistem informasi yang sering digunakan merujuk kepada interaksi antara orang, proses algoritmik, data, dan teknologi.

2.1.4. Pengertian Anggaran

Menurut Sasongko dan Parulian (2015), Anggaran adalah rencana kegiatan yang akan dijalankan oleh manajemen dalam satu periode yang tertuang secara kuantitatif. Informasi yang dapat diperoleh dari anggaran di antaranya jumlah produk dan harga jualnya untuk tahun depan.

2.1.5. Pengertian Dokumen

Menurut Ensiklopedi Umum, dokumen berarti surat, akte, piagam, surat resmi dan bahan rekaman tertulis atau tercetak yang dapat memberi keterangan.

2.1.6. Pengertian Web

Menurut Abdullah (2015), *Web* dapat diartikan sekumpulan halaman yang terdiri dari beberapa laman yang berisi informasi dalam bentuk data digital baik berupa text, gambar, video, audio, dan animasi lainnya yang disediakan melalui jalur koneksi internet.

2.1.7. Pengertian Sistem Informasi Monitoring Dokumen Pelaksanaan Anggaran Berbasis Web pada Dinas Perhubungan Provinsi Maluku

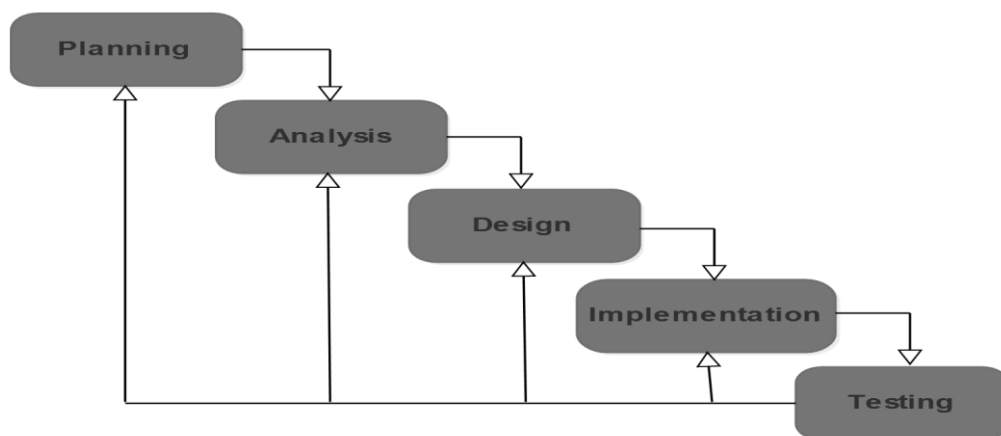
Sistem Informasi Monitoring Dokumen Pelaksanaan Anggaran Berbasis *Web* pada Dinas Perhubungan Provinsi Maluku adalah rangkaian pemrograman yang dipahami oleh komputer yang disusun sehingga menghasilkan sebuah proses untuk melakukan pengolahan data penyelenggara anggaran agar lebih efektif dan efisien dengan menggunakan bahasa pemrograman PHP dan MySQL.

2.2. Teori Khusus

2.2.1. SDLC (*System Development Life Cycle*)

Menurut Rosa A. S. dan M. Shalahuddin (2018), Metode SDLC (*System Development Life Cycle*) adalah proses mengembangkan atau mengubah suatu sistem perangkat lunak dengan menggunakan model-model dan metodologi yang digunakan orang untuk mengembangkan sistem-sistem perangkat lunak sebelumnya.

Menurut Rosa A. S. dan M. Shalahuddin (2018), Model SDLC *Waterfall* atau air terjun menyediakan pendekatan alur hidup perangkat lunak secara sekuensial atau terurut dimulai dari analisis, desain, pengodean, pengujian, dan tahap pendukung (*support*).



(Sumber : Rosa A. S. dan M. Shalahuddin, 2018)

Gambar 2.1 Metode SDLC *Waterfall*

- **Analisis Kebutuhan Perangkat Lunak**

Proses pengumpulan kebutuhan dilakukan secara intensif untuk mespesifikasikan kebutuhan perangkat lunak agar dapat dipahami perangkat lunak seperti apa yang dibutuhkan oleh *user*.

- **Desain**

Proses multi langkah yang fokus pada desain pembuatan program perangkat lunak termasuk struktur data, arsitektur perangkat lunak, representasi antarmuka, dan prosedur pengodean.

- **Pembuatan Kode Program (*coding*)**

Desain harus ditranslasikan ke dalam program perangkat lunak. Hasil dari tahap ini adalah program komputer sesuai dengan desain yang telah dibuat pada tahap desain.

- **Pengujian (*testing*)**

Pengujian fokus pada perangkat lunak secara dari segi logik, fungsional dan memastikan bahwa semua bagian sudah diuji. Hal ini dilakukan untuk meminimalisir kesalahan (*error*) dan memastikan keluaran yang dihasilkan sesuai dengan yang diinginkan.

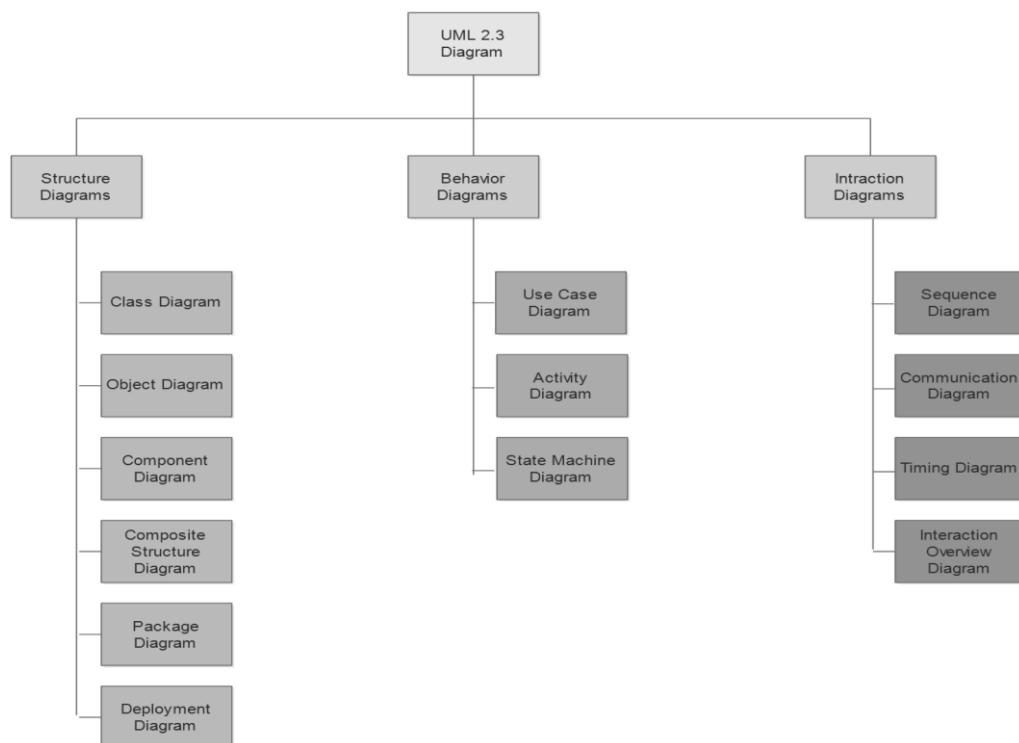
- **Pemeliharaan (*maintenance*)**

Tidak menutup kemungkinan sebuah perangkat lunak mengalami perubahan ketika sudah dikirimkan ke *user*. Perubahan bisa terjadi karena adanya kesalahan yang muncul dan tidak terdeteksi saat pengujian atau perangkat lunak harus beradaptasi dengan lingkungan baru.

2.2.2. UML (*Unified Modeling Language*)

Menurut Rosa A. S. dan M. Shalahuddin (2018), UML (*Unified Modeling Language*) adalah bahasa visual untuk pemodelan dan komunikasi mengenai sebuah sistem dengan menggunakan diagram dan teks pendukung.

2.2.3. Diagram UML



(Sumber : Rosa A. S. dan M. Shalahuddin, 2018)

Gambar 2.2 Diagram UML

Berikut ini penjelasan singkat dari pembagian kategori tersebut.

- *Structure Diagrams* yaitu kumpulan diagram yang digunakan untuk menggambarkan suatu struktur statis dari sistem yang di modelkan.
- *Behavior diagrams* yaitu kumpulan diagram yang digunakan untuk menggambarkan kelakuan sistem atau rangkaian perubahan yang terjadi pada sebuah sistem.

- *Interaction diagrams* yaitu kumpulan diagram yang digunakan untuk menggambarkan interaksi sistem dengan sistem lain maupun interaksi antar subsistem pada suatu sistem.


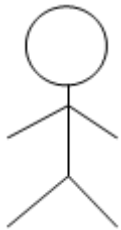
2.2.4. Jenis-jenis Diagram *Unified Modeling Language* (UML)



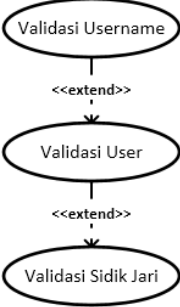
A. *Use Case Diagram*


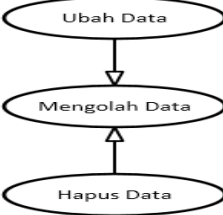
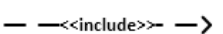
Menurut Rosa A. S. dan M. Shalahuddin (2018), *use case* atau diagram *use case* merupakan pemodelan untuk kelakuan (*behavior*) sistem informasi yang akan dibuat. *Use case* mendeskripsikan sebuah interaksi antara satu atau lebih aktor dengan sistem informasi yang akan dibuat. Secara kasar, *use case* digunakan untuk mengetahui fungsi apa saja yang ada di dalam sebuah sistem informasi dan siapa saja yang berhak menggunakan fungsi-fungsi itu.

Berikut adalah simbol-simbol yang ada pada diagram *use case* :

Tabel 2.1 Simbol-simbol *Use case Diagram*

No	Simbol	Nama	Deskripsi
1		<i>Use Case</i>	Fungsionalitas yang disediakan sistem sebagai unit-unit yang saling bertukar pesan antar unit atau aktor.
2		<i>Aktor/actor</i>	Orang, proses, atau sistem lain yang berinteraksi dengan sistem informasi yang akan dibuat di luar sistem informasi yang akan dibuat itu sendiri, jadi walaupun simbol dari aktor adalah gambar orang, tapi aktor belum tentu

			merupakan orang, biasanya dinyatakan menggunakan kata benda di awal frase nama aktor.
3		<i>Association</i>	Komunikasi antara aktor dan <i>use case</i> yang berpartisipasi pada <i>use case</i> atau <i>use case</i> memiliki interaksi dengan aktor.
4		<i>Extend</i>	<p>Relasi <i>use case</i> tambahan sebuah <i>use case</i> dimana <i>use case</i> yang ditambahkan dapat berdiri sendiri walau tanpa <i>use case</i> tambahan itu, mirip dengan prinsip <i>inheritance</i> pada pemrograman berorientasi objek, biasanya <i>use case</i> tambahan memiliki nama depan yang sama dengan <i>use case</i> yang ditambahkan, misalnya:</p>  <p>Arah panah mengarah pada <i>use case</i> yang ditambahkan, biasanya <i>use case</i> yang menjadi <i>extend</i>-nya merupakan jenis yang sama dengan <i>use case</i> yang menjadi induknya.</p>

5		<i>Generalisasi</i>	<p>Hubungan generalisasi dan spesialisasi (umum-khusus) antara dua buah use case dimana fungsi yang satu adalah fungsi yang lebih umum dari lainnya, misalnya:</p>  <p>arah panah mengarah pada use case yang menjadi generalisasinya (umum)</p>
6		<i>Include</i>	<p>Relasi use case tambahan ke sebuah use case dimana use case yang ditambahkan memerlukan use case ini untuk menjalankan fungsinya atau sebagai syarat dijalankan use case ini</p>






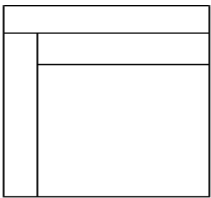
Sumber: A.S Rosa dan Shalahuddin .M (2018).

B. Activity Diagram

Menurut Rosa A. S. dan M. Shalahuddin (2018), diagram aktivitas atau *activity diagram* menggambarkan *workflow* (aliran kerja) atau aktivitas dari sebuah sistem atau proses bisnis atau menu yang ada pada perangkat lunak.

Adapun simbol-simbol yang digunakan dalam *activity diagram* adalah sebagai berikut:

Tabel 2.2 Simbol-simbol *Activity Diagram*


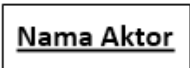
No	Simbol	Nama	Deskripsi
1		<i>Initialstate</i>	Status awal aktivitas sistem, sebuah diagram aktivitas memiliki sebuah status awal.
2		<i>Activity</i>	Aktivitas yang dilakukan sistem, aktivitas biasanya diawali dengan kata kerja.
3		Percabangan/ decision	Asosiasi percabangan dimana jika ada pilihan aktivitas lebih dari satu.
4		<i>Join</i>	Asosiasi penggabungan dimana lebih dari satu aktivitas digabungkan menjadi satu.
5		<i>End State</i>	Status akhir yang dilakukan oleh sistem, sebuah diagram aktivitas memiliki sebuah status akhir.
6		<i>Swimlane</i>	Memisahkan organisasi bisnis yang bertanggung jawab terhadap aktivitas yang terjadi.


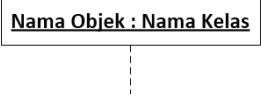

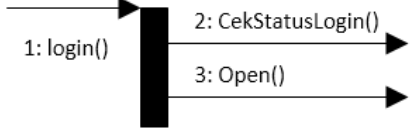
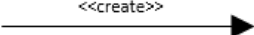
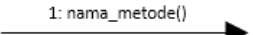
Sumber: Rosa A.S dan Shalahuddin .M (2018)

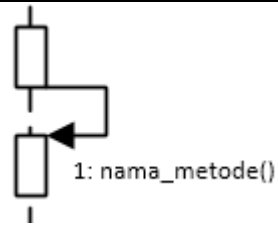
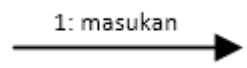
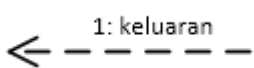
C. *Sequence Diagram*

Menurut Rosa A. S. dan M. Shalahuddin (2018), diagram sekuen menggambarkan kelakuan objek pada *use case* dengan mendeskripsikan waktu hidup objek dengan *message* yang dikirimkan dan diterima antar objek. Oleh karena itu untuk menggambarkan diagram sekuen maka harus diketahui objek-objek yang terlibat dalam sebuah *use case* beserta metode-metode yang dimiliki kelas yang diinstansiasi menjadi objek itu. Membuat diagram sekuen juga dibutuhkan untuk melihat skenario yang ada pada *use case*. Banyaknya diagram sekuen yang harus digambar adalah minimal sebanyak pendefinisian *use case* yang memiliki proses sendiri atau yang penting semua *use case* yang telah didefinisikan interaksi jalannya pesan sudah dicakup dalam diagram sekuen sehingga semakin banyak *use case* yang didefinisikan maka diagram sekuen yang harus dibuat juga semakin banyak. Simbol-simbol yang digunakan dalam *sequence diagram* :

Tabel 2.3 Simbol-simbol *Sequence Diagram*

No	Simbol	Nama	Deskripsi
1	 atau  Tanpa waktu aktif	<i>Actor</i>	Orang, proses, atau sistem lain yang berinteraksi dengan sistem informasi yang akan dibuat diluar sistem informasi yang akan dibuat itu sendiri, jadi walaupun simbol dari aktor

			adalah gambar orang, tapi aktor belum tentu merupakan orang, biasanya dinyatakan dalam menggunakan kata benda diawal frase nama aktor.
2		Garis hidup/ <i>lifeline</i>	Menyatakan kehidupan suatu objek.
3		Objek	Menyatakan objek yang berinteraksi pesan.
4		Waktu aktif	<p>Menyatakan objek dalam keadaan aktif dan berinteraksi, semuanya yang terhubung dengan waktu aktif ini adalah sebuah tahapan yang dilakukan di dalamnya, misalnya:</p>  <p>Maka cekStatusLogin() dan open() dilakukan didalam metode login(). Aktor tidak memiliki waktu aktif.</p>
5		Pesan tipe <i>create</i>	Menyatakan suatu objek membuat objek yang lain, arah panah mengarah pada objek yang dibuat
6		Pesan tipe <i>call</i>	Menyatakan suatu objek memanggil operasi/metode yang ada pada objek lain atau dirinya

			<p>sendiri,</p>  <p>Arah panah mengarah pada objek yang memiliki operasi/metode, karena ini memanggil operasi/metode maka operasi/metode yang dipanggil harus ada pada diagram kelas sesuai dengan kelas objek yang berinteraksi</p>
7		Pesan tipe <i>send</i>	Menyatakan bahwa suatu objek data/masukkan/informasi ke objek lainnya, arah panah mengarah pada objek yang dikirim
8		Pesan tipe return	Menyatakan bahwa suatu objek yang telah menjalankan suatu operasi atau metode menghasilkan suatu kembalian ke objek tertentu, arah panah mengarah pada objek yang menerima kembalian

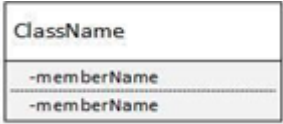


Sumber: Rosa A.S dan Shalahuddin .M (2018)




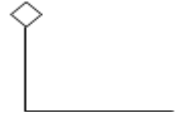
D. Class Diagram

Menurut Rosa A.S dan M. Shalahudin (2018), diagram kelas atau *class diagram* menggambarkan struktur sistem dari segi pendefinisian kelas-kelas yang akan dibuat untuk membangun sistem. Kelas memiliki apa yang disebut atribut dan *method* atau operasi. Berikut penjelasan atribut dan *method* :

1. Atribut merupakan variable-variabel yang dimiliki oleh suatu kelas.
2. Operasi atau *method* adalah fungsi-fungsi yang dimiliki oleh suatu kelas.

Tabel 2.4 Simbol-simbol *Class Diagram*

No	Simbol	Nama	Deskripsi
1		<i>Class</i>	Kelas pada struktur sistem.
2		Antarmuka/ <i>interface</i>	Sama dengan konsep interface dalam pemrograman berorientasi objek
3		<i>Asosiasi/</i> <i>association</i>	Relasi antar kelas dengan makna umum, asosiasi biasanya juga disertai dengan multiplicity.

4		Asosiasi berarah/ <i>directed association</i>	Relasi antar kelas dengan makna kelas yang satu digunakan oleh kelas yang lain, asosiasi biasanya juga disertai dengan multiplicity
5		<i>Generalisasi</i>	Relasi antar kelas dengan makna generalisasi- spesialisasi (umum- khusus).
6		<i>Kebergantungan/ dependensi</i>	Relasi antar kelas dengan makna kebergantungan antar kelas
7		<i>Agrgasi/ aggregation</i>	Relasi antar kelas dengan makna semua-bagian (whole- part)

Sumber: A.S Rosa dan Shalahuddin .M (2018)

2.2.5. Black Box Testing

2.2.5.1. White Box Testing (pengujian kotak putih)

Menurut Rosa A.S (2018), *White Box Testing* yaitu menguji perangkat lunak dari segi desain dan kode program apakah mampu menghasilkan fungsi – fungsi, masukan, dan keluaran yang sesuai dengan spesifikasi kebutuhan. Pengujian kotak putih dilakukan dengan

memeriksa logik dari kode program. Pembuatan kasus logik dari kode program. Pembuatan kasus uji bisa mengikuti standar pengujian dari standar pemrograman yang seharusnya.

2.2.5.2. Pengujian Kotak Hitam (*Black Box Testing*)

Menurut Rosa A.S (2018), *Black box testing* yaitu 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 dibutuhkan. Pengujian kotak hitam dilakukan dengan membuat kasus uji yang bersifat mencoba semua fungsi dengan memakai perangkat lunak apakah sesuai dengan spesifikasi yang dibutuhkan. Kasus uji yang dibuat untuk melakukan pengujian kotak hitam harus dibuat dengan kasus benar dan kasus salah, misalkan untuk proses login maka kasus uji yang dibuat adalah :

1. Jika user memasukan nama pemakai (*username*) dan kata sandi (*password*) yang benar
2. Jika user memasukan nama pemakai (*username*) dan kata sandi (*password*) yang salah, misalnya nama pemakai benar tapi kata sandi salah, atau sebaliknya atau keduanya salah.

2.3. Teori Program

2.3.1. *Xampp*

2.3.1.1. Pengertian *Xampp*

Menurut Madcoms (2016), “Xampp adalah sebuah paket kumpulan software yang terdiri dari *Apache*, *MySQL*, *PhpMyadmin*, *PHP*, *Perl*, *Filezilla*, dan lain-lain.

Menurut Yudho Yudhanto & Helmi Adi Prasetyo (2018), Xampp adalah kompilasi program gratis terfavorit di kalangan developer/programmer yang berguna untuk pengembangan website berbasis PHP dan MySQL. Dengan satu aplikasi ini, kita akan mendapatkan paket software komplet yang bisa dijalankan pada windows sehingga programmer dapat dengan mudah melakukan simulasi pada computer *local* sebelum diunggah ke internet.

2.3.1.2. *PHPMyAdmin*

Menurut Madcoms (2016), “*PhpMyadmin* adalah sebuah aplikasi open source yang berfungsi untuk memudahkan manajemen MySQL”. *PhpMyadmin* dapat dijalankan di banyak OS, selama dapat menjalankan *webserver* dan MySQL.I

Menurut Yudho Yudhanto & Helmi Adi Prasetyo (2018), *PhpMyAdmin* adalah aplikasi web untuk mengelola database MySQL dengan mudah melalui antarmuka (*interface*) garfish. Aplikasi web ini ditulis dengan menggunakan bahasa pemrograman PHP. Sebagaimana aplikasi-aplikasi lain untuk lingkungan web (aplikasi yang dibuka atau dijalankan menggunakan browser), phpMyAdmin juga mengandung unsure HTML/XHTML, CSS dan juga kode Javascript.

2.3.2. MySQL

Menurut Madcoms (2016), MySQL adalah sistem manajemen database *SQL* yang bersifat *Open Source* dan paling populer saat ini. Sistem database MySQL mendukung beberapa fitur seperti *multithreaded*, *multi-user*, dan *SQL database management system* (DBMS).

2.3.3. HTML, CSS dan Javascript

Menurut Achmad Solichin (2016), HTML merupakan singkatan dari *Hypertext Markup Language*. HTML dikembangkan pertama kali oleh Tim Berners-Lee bersamaan dengan protokol HTTP (*Hypertext Transfer Protokol*) pada tahun 1989. Tujuan utama pengembangan HTML adalah untuk menghubungkan satu halaman web hanya berupa teks, tidak seperti sekarang.

Menurut Achmad Solichin (2016), setiap halaman web ditulis dalam bentuk HTML. HTML merupakan bahasa pemrograman web yang memberitahukan perambanan web (web browser) bagaimana menyusun dan menyajikan konten di halaman web. Dengan kata lain, HTML adalah pondasi web. HTML disusun dengan bahasa yang sederhana, sehingga sangat mudah diimplementasikan. Saat ini, HTML dapat menampilkan obyek-obyek seperti teks, tabel, tautan, gambar, audio dan video.

Menurut Achmad Solichin (2016), HTML merupakan bahasa dasar web yang berfungsi untuk menampilkan berbagai komponen web. Sementara itu, untuk mempercantik tampilan web, dikembangkanlah CSS atau *Cascading Style Sheet*. CSS pertama kali diusulkan oleh Hakom Wium Lie pada tahun 1994 dan selanjutnya distandarisasi oleh W3C. CSS memberikan cara yang mudah dan

efisien bagi pemrograman untuk menentukan tata letak halaman web dan mempercantik halaman dengan elemen desain seperti warna, sudut bulat, gradien, dan animasi.

Menurut Achmad Solichin (2016), selain HTML dan CSS, sebuah aplikasi berbasis web tidak dapat dilepaskan dari teknologi Javascript. Pertama kali Javascript dikembangkan oleh Netscape dengan nama awal LiveScript. Fungsi utama dari Javascript adalah untuk menambah fungsionalitas data dari sisi client serta menyajikan komponen web yang lebih interaktif. Javascript makin populer sejak kemunculan konsep AJAX (Asynchronous javascript and XML) yang memungkinkan interaksi antara client dan server lebih elegan dan fleksibel.

2.3.4. PHP

Menurut Madcoms (2016), PHP atau singkatan dari *Hypertext Preprocessor* adalah bahasa script yang dapat ditanamkan atau disisipkan ke dalam HTML. PHP adalah bahasa pemrograman script server-side yang didesain untuk pengembangan web.

Menurut Yudho Yudhanto & Helmi Adi Prasetyo (2018), PHP atau *Hypertext Preprocessor* adalah bahasa pemrograman *script server side* yang sengaja dirancang lebih cenderung untuk membuat dan mengembangkan web. Bahasa pemrograman ini memang dirancang untuk para pengembangan web agar dapat menciptakan suatu halaman web yang bersifat dinamis.

2.3.5. Framework

Menurut Yudho Yudhanto & Helmi Adi Prasetyo (2018), *Framework* adalah kumpulan fungsi (*libraries*) sehingga seorang *programmer* tidak perlu lagi membuat fungsi-fungsi dari awal dan biasanya disebut kumpulan *library*. Programmer cukup memanggil kumpulan *library* atau fungsi yang sudah ada di dalam *framework* yang sudah pasti cara menggunakan fungsi-fungsi itu sudah ditentukan sesuai aturan masing-masing.

2.3.6. CodeIgniter

Menurut Wikipedia, *CodeIgniter* merupakan *framework* yang berupa kerangka kerja PHP dengan model MVC (*Model, View, Controller*) untuk membangun website dinamis dengan menggunakan PHP. *CodeIgniter* memudahkan pengembang web untuk membuat aplikasi web dengan cepat mudah dibandingkan dengan membuatnya dari awal. *CodeIgniter* dirilis pertama kali pada 28 Februari 2006.