BAB II
TINJAUAN PUSTAKA

2.1 Teori Umum

2.1.1 Pengertian Komputer

Menurut buku Computer Today (Donald H. Sanders), Komputer adalah sistem elektronik untuk memanipulasi data yang cepat dan tepat serta dirancang dan diorganisasikan agar secara otomatis menerima dan menyimpan data *input*, memprosesnya, dan menghasilkan *output* dibawah pengawasan suatu langkah-langkah instruksi program yang tersimpan pada memori.

Robert H. Blissmer (1985), dalam bukunya "Computer Annual", Komputer adalah suatu alat elektronik yang mampu melakukan beberapa tugas sebagai berikut: menerima *input*, memproses *input* tadi sesuai dengan programnnya, menyimpan perintah-perintah dan hasil dari pengolahan dan menyediakan *output* dalam bentuk informasi.

2.1.2 Pengertian Perangkat Lunak (*Software*)

Perangkat lunak atau *software* adalah sebuah perintah program dalam sebuah komputer, yang apabila diberi perintah oleh *user* akan memberikan fungsi dan unjuk kerja seperti yang diharapkan oleh *user*nya. Pernyataan ini menggambarkan bahwa *software* atau perangkat lunak ini berfungsi memberi perintah untuk komputer, agar komputer berfungsi secara optimal, sesuai dengan kemauan *user* atau pengguna yang memberikan perintah. (Roger : 2022,10)

Software adalah sebuah perangkat yang berfungsi sebagai pengatur aktivitas kerja komputer dan seluruh instruksi yang mengarah pada sistem komputer. Kemudian dijelaskan pula bahwa software merupakan perangkat yang menjembati interaksi *user* dengan komputer yang menggunakan bahasa mesin. (Melwin Syafrizal Daulay : 2007, 22)

2.1.3 Pengertian Perangkat Keras (Hardware)

Menurut James O'Brien, hardware adalah semua peralatan fisik yang digunakan dalam pemrosesan informasi, termasuk diantaranya CPU, RAM,

monitor, mouse, keybord, printer, scanner, dan lain-lain. Perangkat keras merupakan media komunikasi yang menghubungkan beberapa jaringan dan memproses paket data sehingga trasmisi data lebih efektif.

Menurut Rainer, hardware adalah perangkat dalam komputer yang berbentuk fisik seperti processor, monitor, keybord, dan printer.Hardware berfungsi untuk menerima data/informasi, memproses dan menampilkan informasi mentah menjadi informasi baru yang berguna.

2.1.4 Pengertian Basis Data (*Database*)

Pengertian *Database* menurut Winarno dan Utomo (2010:142) "*Database* atau biasa disebut basis data merupakan kumpulan data yang saling berhubungan. Data tersebut biasanya terdapat dalam tabel-tabel yang saling berhubungan satu sama lain, dengan menggunakan *field*/kolom pada tiap tabel yang ada".

Menurut Indrajani (2015:70), "basis data adalah kumpulan data yang saling berhubungan secara logis dan didesain untuk mendapatkan data yang dibutuhkan oleh suatu organisasi".

2.1.5 Pengertian Internet

Menurut Oneto dan Sugiarto (2009:1) "Internet adalah jaringan komputer". Ibarat jalan raya, internet dapat dilalui berbagai sarana transportasi, seperti bus, mobil dan motor yang memiliki kegunaan masing-masing.

Menurut Hidayatullah dan Kawistara (2015) "Internet adalah jaringan global yang menghubungkan komputer-komputer seluruh dunia, dengan internet sebuah komputer bisa mengakses data yang terdapat pada komputer lain di benua yang berbeda"

2.1.6 Metode Pengembangan Sistem *Waterfall*

Waterfall adalah model pengembangan perangkat lunak yang paling banyak digunakan. Menurut R. A. Sukamto dan M. Salahuddin (2014) mengemukakan bahwa Model SDLC air terjun (*waterfall*) sering juga disebut model sekuensial linier (*Sequential linier*) atau alur hidup klasik (*classic life cycle*). "Model air terjun menyediakan pendekatan alur hidup perangkat lunak secara sekuensial atau terurut

dimulai dari analisis, desain, pengodean, pengujian, dan tahap pendukup (*support*)". Berikut tahapan dari sistem rekayasa informasi, yaitu:

1. Analisis kebutuhan perangkat lunak

   Proses pengumpulan kebutuhan dilakukan secara intensif untuk menspesifikasikan kebutuhan perangkat lunak agar dapat dipahami perangkat lunak seperti apa yang dibutuhkan oleh user. Spesifikasi kebutuhan perangkat lunak pada tahap ini perlu di dokumentasikan.
2. Desain

   Desain perangkat lunak adalah proses multi langkah yang fokus pada desain pembuatan program perangkat lunak termasuk struktur data, arsitektur perangkat lunak, representasi antarmuka, dan prosedur pengodean. Tahap ini mentranslasi kebutuhan perangkat lunak dari tahap analisis kebutuhan ke representasi desain agar dapat diimplementasikan menjadi program pada tahap selanjutnya. Desain perangkat lunak yang dihasilkan pada tahap ini juga perlu didokumentasikan.
3. Pembuatan Kode Program

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

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

2.2 Teori Judul

2.2.1 Pengertian Aplikasi

Menurut Hasan Abdurahman dan Asep Ririh Riswaya (2014), aplikasi adalah program siap pakai yang dapat digunakan untuk menjalankan perintah-perintah dari pengguna aplikasi tersebut dengan tujuan mendapatkan hasil yang

lebih akurat sesuai dengan tujuan pembuatan aplikasi tersebut, aplikasi mempunyai arti yaitu pemecahan masalah yang menggunakan salah satu teknik pemrosesan data aplikasi yang biasanya berpacu pada sebuah komputansi yang diinginkan atau diharapkan maupun pemrosesan data yang diharapkan.

Pengertian aplikasi menurut Jogiyanto (1999:12) adalah penggunaan dalam suatu komputer, instruksi (*instruction*) atau pernyataan (*statement*) yang disusun sedemikian sehingga komputer dapat memproses *input* menjadi *output*.

2.2.2 Pengertian Jasa

Jasa dapat didefinisikan sebagai berikut (Kotler, 1994 dalam Fandi Tjiptono, 2006:6) : Jasa adalah setiap tindakan atau perbuatan yang dapat ditawarkan oleh suatu pihak lain, yang pada dasarnya bersifat *intangible* (tidak berwujud fisik) dan tidak menghasilkan kepemilikan sesuatu. Produksi jasa bisa berhubungan dengan produk fisik.

2.2.3 Pengertian Ekspedisi

Menurut Bowersox (1987), "Ekspedisi merupakan proses pengaturan strategis pemindahan komponen, material dan barang siap pakai dari pemasok antar fasilitas dalam perusahaan maupun ke konsumen".

Menurut Mulyadi (2013:201), "Ekspedisi merupakan suatu kegiatan mengirim barang dikarenakan adanya penjualan barang dagang".

2.2.4 Pengertian *Website*

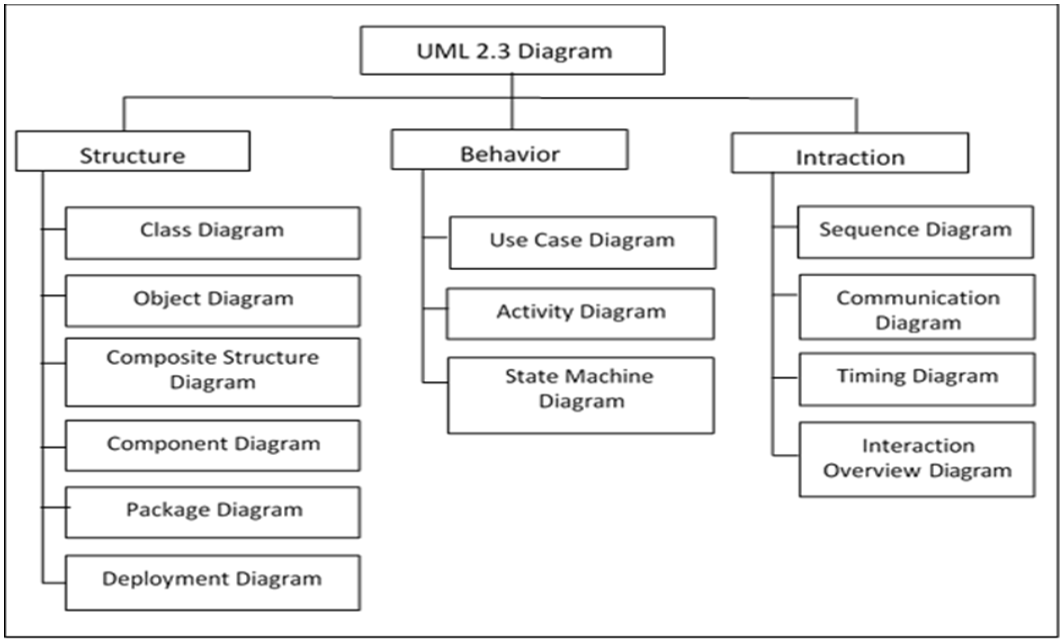
Menurut Yuhefizar (2013:2) pengertian *website* adalah "keseluruhan halaman-halaman *web* yang terdapat dari sebuah domain yang mengandung informasi".

2.3 Teori Khusus

2.3.1 Pengertian UML (*Unified Modelling Language*)

Menurut Sulianta (2017) dalam buku Teknik Perancangan Arsitektur Sistem Infomasi: "*Unified Modelling Language* (UML) merupakan kumpulan diagram-diagram yang sudah memiliki standar untuk membangun perangkat lunak berbasis objek".

Menurut Sukamto dan Shalahuddin (2018:140), “Pada UML 2.3 terdiri dari 13 macam diagram yang dikelompokkan dalam 3 kategori”. Pembagian kategori dan macam-macam diagram menurut Sukamto dan Shalahuddin tersebut dapat dilihat pada gambar dibawah:



Sumber : Sukamto dan Shalahuddin (2018:140)

Gambar 2.1 Macam-macam Diagram UML

Penjelasan singkat dari pembagian kategori pada diagram UML menurut Sukamto dan Shalahuddin (2018:141):

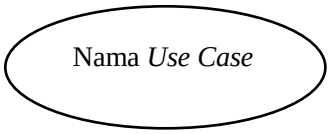
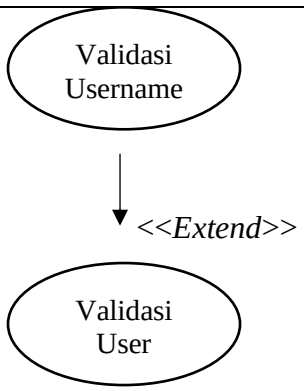
1. *Structure diagrams* yaitu kumpulan diagram yang digunakan untuk menggambarkan suatu struktur statis dari sistem yang dimodelkan.
2. *Behavior diagrams* yaitu kumpulam diagram yang digunakan untuk menggambarkan kelakuan sistem atau rangkaian perubahana yang terjadi pada sebuah sistem.
3. *Interaction diagrams* yaitu kumpulan diagram yang digunakan untuk menggambarkan interaksi sistem dengan sistem lain maupun interaksi antar subsistem pada suatu sistem.

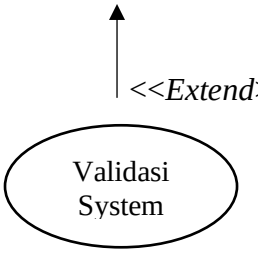
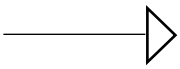
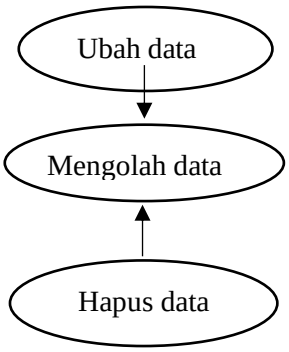
2.3.2 Pengertian *Use Case Diagram*

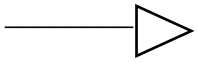
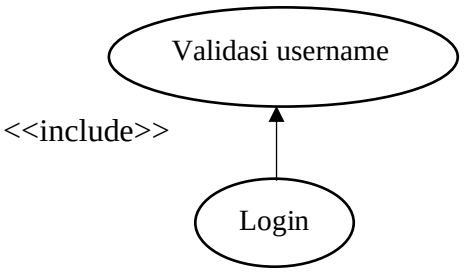
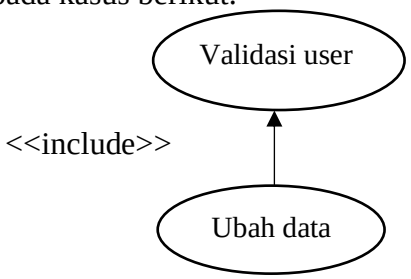
Menurut Pratama (2019b), "Use case diagram adalah gambaran grafis dari beberapa atau semua *actor, use case,* dan interaksi diantaranya yang memperkenalkan suatu sistem".

Sukamto dan Shalahuddin (2018:155), menjelaskan tentang *use case diagram* sebagai berikut: "*Use case* merupakan pemodelan untuk kelakuan (*behavior*) sistem informasi yang akan dibuat. *Use case* mendeskripsikan sebuah interaksi antara satu atau lebih aktor dengan sistem". Adapun simbol-simbol yang digunakan dalam *use case* adalah sebagai berikut:

Tabel 2.1 Simbol-simbol pada *Use Case* Diagram

Simbol	Deskripsi
Use Case 	fungsionalitas yang disediakan sistem sebagai unit-unit yang saling bertukar pesan antar unit atau aktor; biasanya dinyatakan dengan menggunakan kata kerja di awal-awal frase nama *use case*
Aktor / *Actor* 	orang, proses, atau siatem 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
Asosiasi / A*ssociation* _____	komunikasi antar aktor dan *use case* yang berpartisipasi pada *use case*
Ekstensi / *Extend* *<<Extend>>* .....>	

	 arah panah mengarah pada *use case* yang ditambahkan; biasanya *use case* yang menjadi *extend*-nya merupakan jenis yang sama dengan *use case* yang menjadi induknya
Generalisasi / *Generalization* 	hubungan generalisasi dan spesialisasi (umum-khusus) antara dua buah *use case* dimana fungsi yang satu adalah fungsi yang lebih umum dari lainnya, misalnya:  arah panah mengarah pada *use case* yang menjadi generalisasinya (umum)
menggunakan / *include* / *uses*  <<uses>>	relasi tambahan ke sebuah *use case* dimana *use case* yang ditambahkan memerlukan *use case* ini untuk menjalankan fungsinya atau sebagai syarat dijalankan *use case* ini, ada dua sudut pandang yang cukup besar mengenai include di *use case*: • *Include* berarti *use case* yang ditambahkan akan selalu di panggil saat *use case* tambahan dijalankan, misalnya pada kasus berikut:

	 *Include* berarti *use case* yang tambahan akan selalu melakukan pengecekan apakah *use case* yang di tambahkan telah dijalankan sebelum *use case* tambahan dijalankan, misal pada kasus berikut:  kedua interpretasi di atas dapat dianut salah satu atau keduanya tergantung pada pertimbangan dan interpretasi yang dibutuhkan.

Sumber : Sukamto dan Shalahuddin (2018:156-158)

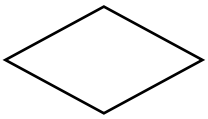
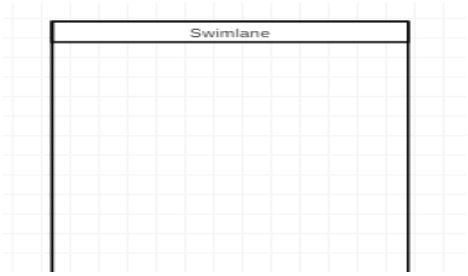
2.3.3 Pengertian *Activity Diagram*

Menurut Hendini (2016) "*Activity Diagram* menggambarkan *workflow* (aliran kerja) atau aktivitas dari sebuah sistem atau proses bisnis".

Menurut Firdaus & Saputra (2018:182) mengatakan bahwa *Activity diagram* menggambarkan aktivitas sistem bukan apa yang dilakukan oleh aktor, jadi aktivitas yang dapat dilakukan oleh sistem.

Tabel 2.2 Simbol-simbol pada *Activity Diagram*

Simbol	Deskripsi
Status awal 	Status awal aktivitas sistem, sebuah diagram aktivitas memiliki sebuah status awal

Aktivitas /*Activity* 	Aktivitas yang dilakukan sistem, aktivitas biasanya diawali dengan kata kerja
Percabangan / *Decision* 	Asosiasi percabangan di mana jika ada pilihan aktivitas lebih dari satu
Penggabungan / *Join* 	Asosiasi penggabungan dimana lebih dari satu aktivitas digabungkan menjadi satu
Status Akhir 	Status akhir yang dilakukan sistem, sebuah diagram aktivitas memiliki sebuah status akhir
Interaksi / *Interaction* 	Alur dari sebauah *activity*
Swimlane  atau 	Memisahkan organisasi bisnis yang bertanggung jawab terhadap aktivitas yang terjadi

Sumber : Sukamto dan Shalahuddin (2018:162-163)

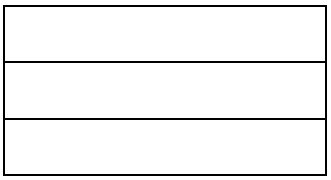
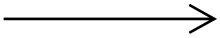
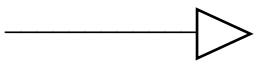
2.3.4 Pengertian *Class Diagram*

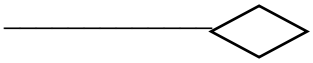
Sukamto dan Shalahuddin (2018:141), menjelaskan tentang *class diagram*

sebagai berikut: *Class Diagram* menggambarkan struktur sistem dari segi pendefinisian kelas-kelas yang akan dibuat untuk membangun sistem. Diagram Kelas dibuat agar pembuat program atau programmer membuat kelas-kelas sesuai rancangan di dalam diagram kelas agar antara dokumentasi perancangan dan perangkat lunak sinkron.

Menurut Firdaus & Saputra (2018:182), *Class diagram* adalah sebuah spesifikasi yang jika diinstansiasi akan menghasilkan sebuah objek dan merupakan inti dari pengembangan dan desain berorientasi objek.

Tabel 2.3 Simbol-simbol *Class Diagram*

Simbol	**Deskripsi**
Kelas / *Class*  .	Kelas pada struktur sistem
Antarmuka / *Interface*  Nama_interface	Sama dengan konsep *interface* dalam pemrograman berorientasi objek
Asosiasi / *Association*	Relasi antarkelas dengan makna umum, asosiasi biasanya juga disertai dengan *multiplicity*
Asosiasi berarah / *Directed association* 	Relasi antar kelas dengan makna kelas yang satu digunakan oleh kelas yang lain, asosiasi biasanya juga disertai dengan *multiplicity*
Generalisasi 	Relasi antar kelas dengan makna generalisasi-spesialisasi (umum khusus)
Kebergantungan / *Dependency*	Relasi antar kelas dengan makna kebergantungan antarmuka

Agregasi / *Aggregation* 	Relasi antar kelas dengan makna semua bagian (*whole part*)

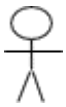
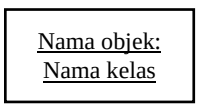
Sumber : Mukhtar, 2019:85

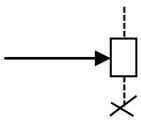
2.3.5 Pengertian *Sequence Diagram*

Hutahaean dan Azhar (2018:21) mengatakan bahwa *Sequence Diagram* menggambarkan kelakuan objek pada *use case* dengan mendeskripsikan waktu hidup objek dan pesan yang dikirimkan dan diterima antar objek.

Menurut Firdaus & Saputra (2018:182) mengemukakan bahwa *Sequence diagram* adalah suatu diagram yang memperlihatkan atau menampilkan interaksi-interaksi antar objek di dalam sistem yang disusun pada sebuah urutan atau rangkaian waktu.

Tabel 2.4 Simbol-simbol *Sequence Diagram*

Simbol	**Deskripsi**
Aktor  Nama_Aktor Tanpa waktu Aktif	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, tetapi aktor belum tentu berupa orang, biasanya dinyatakan menggunakan kata benda diawal frase nama aktor
Garis hidup / *lifeline*	Menyatakan kehidupan suatu objek
Objek  Nama objek: Nama kelas	Menyatakan objek yang berinteraksi pesan
Waktu Aktif 	Menyatakan objek dalam keadaan aktif dan berinteraksi, semua yang terhubung dengan waktu aktif ini adalah sebuah tahapan yang dilakukan didalamnya

Pesan tipe *create*  <<*create*>>	Menyatakan suatu objek membuat objek yang lain, arah panah mengarah pada objek yang dibuat
Pesan tipe call 1 : nama_metode() 	Menyatakan suatu objek memanggil operasi/metode yang ada pada objek lain atau dirinya sendiri
Pesan tipe *send* 1 : masukan 	Menyatakan bahwa suatu objek mengirimkan data/masukan/informasi ke objek lainnya, arah panah mengarah pada objek yang dikirimi
Pesan tipe *return* 1 : keluaran 	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
Pesan tipe *destroy* <<*destroy*>> 	Menyatakan suatu objek mengakhiri hidup objek yang lain, arah panah mengarah pada objek yang diakhiri, sebaiknya jika ada *create* maka ada *destroy*

Sumber : Mukhtar, 2019:84-85

2.3.6 Kamus Data

Menurut Maniah dan Hamidin (2017: 59) “Model berikutnya yang akan dibahas adalah data *dictionary*/ DD (Kamus Data/KD). Kamus Data tidak menggunakan notasi grafis sebagaimana halnya DFD, Kamus Data juga mempunyai fungsi yang sama dalam pemodelan sistem, yaitu sebagai katalog data dan kebutuhan-kebutuhan informasi dari suatu sistem informasi.

Rosa dan Shalahuddin (2018:73),“Kamus Data adalah kumpulan daftar elemen data yang mengalir pada sistem perangkat lunak sehingga masukan (*input*) dan keluaran (*output*) dapat dipahami secara umum (memiliki standar cara penuliasan)”. Kamus data memiliki beberapa simbol sebagai berikut.

Tabel 2.5 Simbol-simbol pada Kamus Data

Simbol	Deskripsi
=	Disusun atau terdiri dari
+	Dan
[|]	Baik..atau...
{}^{n}	N kali diulang/bernilai banyak
()	Data opsional
..	Batas komentar
@	Identifikasi atribut kunci

Sumber : Rosa dan Shalahuddin. 2018:74

2.4 Teori Program

2.4.1 Pengertian Visual Studio Code

Visual Studio Code merupakan sebuah aplikasi *editor code open source* yang dikembangkan oleh Microsoft untuk sistem operasi Windows, Linux, dan MacOS. Visual Code memudahkan dalam penulisan *code* yang mendukung beberapa jenis pemrograman, seperti C++, C#, Java, Python, PHP, GO. Visual Code memiliki kemampuan untuk mengidentifikasi jenis bahasa pemrograman yang digunakan dan memberi variasi warna sesuai dengan fungsi dalam rangkaian *code* tersebut. Visual Studio Code juga telah terintegrasi ke Github. Selain itu fitur lainnya adalah kemampuan untuk menambah ekstensi dimana para pengembang dapat menambah ekstensi untuk menambah fitur yang tidak ada di Visual Studio Code.

(Edy Winarno dan Ali Zaki, 2014:102). Visual Studio Code adalah kode editor sumber yang dikembangkan oleh Microsoft untuk Windows, Linux dan macOS. Ini termasuk dukungan untuk *debugging*, kontrol git yang tertanam dan GitHub, penyorotan sintaksis, penyelesaian kode cerdas, *snippet*, dan *refactoring* kode.

2.4.2 Pengertian HTML

Menurut Winarno dan Utomo (2010:66) “HTML singkatan dari *Hypertext Markup Language* dan berguna untuk menampilkan halaman *web*”.

Menurut Setiawan (2017:16) HTML merupakan sebuah bahasa pemrograman tertekstur yang dikembangkan untuk membuat halaman *website* yang dapat diakses atau ditampilkan menggunaakan *web browser*.

2.4.3 Pengertian CSS

Menurut Winarno dan Utomo (2010:106) menerangkan bahwa “CSS merupakan bahasa pemrograman *web* yang digunakan untuk mengatur *style-style* yang ada di *tag-tag* HTML”.

Jayan (2010:2) mengemukakan bahwa CSS merupakan singkatan dari *Cascading Style Sheet*. Kegunaannya adalah untuk mengatur tampilan dokumen HTML, contohnya seperti pengaturan jarak antar baris, teks, warna dan format border bahkan penampilan file gambar.

2.4.4 Pengertian PHP

Pengertian PHP menurut Anhar (2010:23) “PHP adalah (*PHP Hypertext Preprocessor*) adalah bahasa pemrograman *web* berupa *script* yang dapat diintegrasikan dengan HTML”.

Menurut MADCOMS (2016) “PHP (*Hypertext Preprocessor*) adalah bahasa *script* yang dapat ditanamkan atau disisipkan ke dalam HTML. PHP banyak dipakai untuk membuat program situs *web* dinamis”.

2.4.5 Pengertian MySQL

MySQL adalah sebuah *software database*. MySQL merupakan tipe data relasional yang artinya MySQL menyimpan datanya dalam bentuk tabel-tabel yang saling berhubungan. (Edy Winarno dan Ali Zaki, 2014:102).

Menurut Anhar (2010:21) “”MySQL (*My Structure Query Language*) adalah sebuah perangkat lunak sistem manajemen basis data SQL *Database Management System* atau DBMS dari sekian banyak DBMS seperti Oracle, MS SQL, Postagre SQL dan lainnya.

2.4.6 Pengertian XAMPP

Menurut Wahanana (2009:30) “XAMPP adalah salah satu paket instalasi apache, PHP, dan MySQL secara instant yang dapat digunakan untuk membantu proses instalasi ketiga produk tersebut”.

Menurut Pratama, I Putu Agus Eka (2014: 440) “XAMPP adalah aplikasi *web* server bersifat *instan* (siap saji) yang dapat digunakan baik di sistem operasi Linux maupun di sistem operasi Windows.

2.4.7 Pengertian Laravel

Laravel adalah sebuah *framework web* berbasis PHP yang *open-source* dan tidak berbayar, diciptakan oleh Taylor Otwell dan diperuntukkan untuk pengembangan aplikasi web yang menggunakan pola MVC. Struktrur pola MVC pada laravel sedikit berbeda pada struktur pola MVC pada umumnya. Di laravel terdapat *routing* yang menjembatani antara *request* dari *user* dan *controller*. Jadi *controller* tidak langsung menerima *request* tersebut (Yudanto dkk, 2017).

Laravel adalah *framework* PHP dengan kode terbuka (*open source*) dengan desain MVC (*Model-View-Controller*) yang digunakan untuk membangun aplikasi *website*. *Framework* ini pertama kali dibangun oleh Taylor Otwell pada tanggal 22 Februari 2012 (Abdulloh, 2018).