BAB II
DASAR TEORI

BAB II
DASAR TEORI

2.1 Informasi dan Sistem Informasi

Informasi adalah data yang telah diolah menjadi bentuk yang lebih berarti dan berguna bagi penerimanya untuk mengambil keputusan masa kini maupun yang akan datang. Kualitas informasi dipengaruhi atau ditentukan oleh hal berikut [3]:

1. Relevan (*relevancy*), informasi harus memberikan manfaat bagi yang membutuhkan, informasi yang disajikan harus mendukung suatu proses bisnis atau pihak yang membutuhkan.
2. Akurat (*accuracy*), informasi harus bebas dari kesalahan dan tidak menyesatkan serta harus jelas karena jika tidak akurat akan menimbulkan banyak gangguan yang dapat merubah atau merusak informasi.
3. Tepat waktu (*timeliness*), berbagai proses diselesaikan dengan tepat waktu, tidak boleh terlambat karena dapat berakibat fatal pada pengambilan keputusan.
4. Lengkap (*complete*), informasi harus diberikan secara jelas, lengkap atau detail, dan mutakhir sesuai dengan yang diinginkan atau dibutuhkan

Sistem informasi adalah kumpulan perangkat keras dan perangkat lunak yang dirancang untuk mentransformasikan data ke dalam bentuk informasi yang berguna. Sistem informasi secara umum terdiri dari sekumpulan komponen berbasis komputer dan manual yang dibuat untuk menghimpun, menyimpan, dan mengelola data serta menyediakan informasi keluaran kepada pemakai [4]. Sehingga dapat disimpulkan bahwa sistem informasi adalah suatu sistem yang dibuat oleh manusia untuk kebutuhan pengolahan data yang mendukung fungsi operasi organisasi yang bersifat manajerial untuk dapat menyediakan informasi berupa laporan-laporan yang diperlukan.

2.2 Pemrograman Berorientasi Objek

Konsep pendekatan berorientasi objek atau juga dikenal sebagai pemrograman berorientasi objek (*object-oriented programming*) adalah pendekatan pemrograman yang menggunakan objek (*object*) dan kelas (*class*). Konsep ini terkait dengan cara pandang *programmer* dalam melakukan analisa sistem dan masalah pemrograman. *Object Oriented Programming* (OOP) adalah sebuah strategi dalam melakukan pembangunan perangkat lunak yang mengelompokkan data dan operasi sebagai sekumpulan objek [5].

Pemrograman berorientasi objek adalah salah satu strategi pembangunan perangkat lunak yang mengoperasikan perangkat lunak sebagai kumpulan objek yang berisi data dan operasi yang dilakukan. Berikut ini adalah beberapa konsep dasar yang harus dipahami tentang metodologi berorientasi objek [6]:

1. Kelas (Class) adalah kumpulan objek dengan karakteristik yang sama. Sebuah kelas akan mempunyai sifat (*atribut*), kelakuan (operasi/metode), hubungan (*relationship*) dan arti.
2. Objek (*object*) adalah suatu entitas yang mampu menyimpan informasi (status). Objek mewakili benda, manusia, organisasi, dan hal-hal yang bersifat abstrak.
3. Atribut (*attribute*) variabel global yang dimiliki sebuah kelas. Atribut dapat berupa nilai atau elemen data yang dimiliki oleh objek dalam kelas objek.
4. Abstraksi (*abstraction*) merupakan prinsip untuk merepresentasikan dunia nyata yang kompleks menjadi satu bentuk model yang sederhana.
5. Enkapsulasi (*encapsulation*) yaitu pembungkus atribut data dan operasi yang dimiliki objek untuk menyembunyikan implementasi dan objek sehingga objek lain tidak mengetahui cara kerjanya.

Pemrograman berorientasi objek menjadi salah satu metode pembuatan perangkat lunak yang menggunakan data dan operasi sebagai suatu objek. Struktur pemrograman berorientasi objek meliputi kelas, objek, metode dan atribut. Penggunaan metode ini menjadi solusi bagi programmer yang terkendala dengan penggunaan metodologi lama.

2.3 Software Development Life Cycle (SDLC) Model Prototype

SDLC (*Software Development Life Cycle*) atau bisa disebut juga *System Development Life Cycle* adalah proses mengembangkan atau mengubah suatu sistem perangkat lunak dengan menggunakan model dan metodologi yang digunakan untuk mengembangkan sistem-sistem perangkat lunak sebelumnya. SDLC terdiri dari tahap perencanaan (*planning*), analisis (*analysis*), desain (*design*), implementasi (*implementation*), uji coba (*testing*), dan pengelolaan (*maintenance*) [6].

Terdapat beberapa model dalam SDLC, salah satunya yaitu *prototype*. Penggunaan model *prototype* ini membuat pengembang dan pengguna dapat saling berinteraksi selama proses pembuatan suatu produk. *Prototyping* adalah proses merancang sebuah *prototype*, yaitu sebuah model produk yang mungkin belum memiliki semua fitur produk sesungguhnya namun sudah memiliki fitur utama dari produk tersebut. *Prototyping* perangkat lunak adalah salah satu metode siklus hidup sistem yang didasarkan pada konsep model kerja (*working model*). Tujuannya untuk mengembangkan model menjadi sistem *final*, dimana sistem akan dikembangkan lebih cepat dari pada metode tradisional dan biaya menjadi lebih rendah. *Prototype* biasanya digunakan untuk keperluan *testing* atau uji coba sebelum berlanjut ke fase pembuatan produk sesungguhnya [7].

Model *prototype* cocok diterapkan untuk menggali spesifikasi kebutuhan pengguna secara lebih detail, tetapi beresiko tinggi terhadap membengkaknya biaya dan waktu proyek. Mengantisipasi hal tersebut, pengembang harus mempunyai perencanaan yang matang, berkomitmen membangun komunikasi yang teratur dengan pengguna, melakukan identifikasi risiko serta melakukan evaluasi secara berkala.

2.4 Unified Modeling Language (UML)

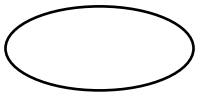
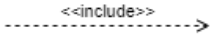
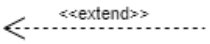
Unified Modeling Language (UML) adalah bahasa pemodelan untuk sistem atau perangkat lunak yang berparadigma berorientasi objek. Pemodelan sesungguhnya digunakan untuk penyederhanaan permasalahan-permasalahan yang kompleks sehingga lebih mudah dipelajari dan dipahami [8]. UML menjadi standar dalam industri untuk visualisasi, merancang dan mendokumentasikan sistem perangkat lunak. UML dapat digunakan untuk membuat model semua jenis aplikasi piranti lunak. UML menggunakan *class* dan o*peration* dalam konsep dasarnya sehingga lebih cocok untuk penulisan dalam bahasa-bahasa berorientasi objek. Seperti bahasa lainnya, UML mendefinisikan notasi dan *syntax*. Notasi UML merupakan sekumpulan bentuk khusus untuk menggambarkan makna tertentu, dan UML *syntax* mendefinisikan bagaimana bentuk-bentuk tersebut dapat dikombinasikan [9].

UML mencakup banyak diagram diantaranya yaitu diagram kelas (*class diagram*), *use case diagram*, *activity diagram*, dan *sequence diagram*. Setiap diagram memiliki peranan masing-masing dan mempresentasikan hal yang berbeda untuk mempermudah pemahaman, pengembangan dan pemeliharaan sistem.

2.4.1 *Use Case Diagram*

Use Case Diagram adalah rangkaian atau uraian sekelompok aktivitas yang saling terkait dan membentuk sistem secara teratur yang dilakukan atau diawasi oleh seorang aktor. Menurut Satzinger, pengertian *use case* adalah rangkaian tindakan yang dilakukan oleh sistem, aktor mewakili user atau sistem lain yang berinteraksi dengan sistem yang dimodelkan. Berikut ini simbol-simbol yang digunakan untuk membuat *use case diagram* [10] :

Tabel 2. 1 Simbol-simbol Use case

No	Nama	SIMBOL	PENJELASAN
1.	*Actor*		*Actor* merupakan peran user sistem ketika berinteraksi dengan *use case*. *Actor* biasanya digambarkan sebagai manusia, perangkat lunak, atau sistem lain yang terlibat dalam skenario yang digambarkan dalam *use case* sistem.
2.	*Association*		*Association* menggambarkan interaksi aktor dan *use case*.
3.	*Use Case*		*Use Case* merupakan seperangkat aktivitas atau tindakan yang dilakukan oleh sebuah sistem.
4.	Sistem		Sistem merupakan entitas atau perangkat lunak yang akan dikembangkan dan menjadi tempat seluruh aktivitas sistem yang sedang berjalan.
5.	*Include*		*Include* menggambarkan bahwa sebuah *use case* merupakan fungsionalitas atau bagian dari *use case* lainnya.
6.	*Extend*		*Extend* menggambarkan hubungan antar *use case* dimana sebuah *use case* merupakan fungsionalitas *use case* lainnya apabila kondisi tertentu terpenuhi.

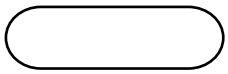
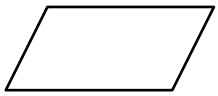
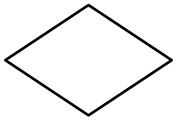
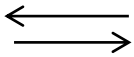
Suatu diagram *use case* dapat terdiri dari satu aktor atau lebih yang mempunyai ikatan dengan sistem. Tiap aktor yang dibuat akan terlihat status serta aksesnya sesuai dengan fungsinya di dalam sebuah sistem atau aplikasi. *Use case* berguna untuk menceritakan rangkaian tindakan (*use case*) yang harus atau bisa dilakukan oleh sistem dalam kerja sama dengan satu atau lebih orang yang menggunakan eksternal sistem (*actor*) [5].

Use case menggambarkan hubungan interaksi antara sistem dengan pengguna sistem. Penggunaan *use case* mempermudah dalam memahami aktivitas yang dapat dilakukan oleh masing-masing aktor sehingga pengembang lebih mudah dalam menentukan hak akses dalam sistem tersebut. *Use case* sangat penting sehingga menjadi diagram wajib yang harus ada dalam pengembangan sebuah sistem.

2.5 Flowchart

Flowchart adalah penggambaran secara grafik dari langkah-langkah dan urutan prosedur dari suatu program. Flowchart membantu analis dan programmer untuk memecahkan masalah ke dalam segmen yang lebih kecil dan membantu dalam menganalisis alternatif lain dalam pengoprasian. *Flowchart* berguna untuk mendesain program dan merepresentasikan program. Sebuah flowchart harus memiliki 3 fungsi utama yaitu *Relationship* untuk memberikan gambaran yang efektif, jelas dan ringkas tentang prosedur logika, *Analysis* untuk mempermudah pembaca melihat permasalahan, serta *Communication* untuk menjadi alat bantu dalam mengkomunikasikan logika suatu masalah [11]. Bagan alir terdiri dari simbol-simbol yang mewakili setiap langkah program, sedangkan garis alir (*flow lines*) menunjukan urutan yang akan dikerjakan. Berikut ini simbol *flowchart* [12] :

Tabel 2. 2 Simbol-simbol Flowchart

No	Nama	SIMBOL	FUNGSI
1.	*Terminator*/Terminal		Simbol Terminal, digunakan untuk menyatakan awal atau akhir suatu program.
2.	*Input/Output*		Simbol *Input/Output*, digunakan untuk menunjukkan operasi masukan atau keluaran.
3.	Proses		Simbol Proses, digunakan untuk menggambarkan proses pengolahan data.
4.	*Decision*/Keputusan		Simbol Keputusan, digunakan untuk menyatakan suatu pilihan berdasarkan suatu kondisi tertentu.
5.	Dokumen		Simbol dokumen, menunjukan dokumen yang digunakan untuk *input* dan *output* baik secara manual, mekanik ataupun komputerisasi.
6.	*Manual Operation*		Simbol *Manual Operation*, digunakan untuk menggambarkan langkah-langkah yang tidak dilakukan oleh komputer.
7.	*Flow*/Arah aliran		Simbol Arah aliran, digunakan untuk menunjukkan arah aliran proses.

Flowchart berfungsi untuk memberi gambaran jalannya sebuah program atau sistem dari satu proses ke proses berikutnya. Penggunaan *flowchart* akan membuat proses dari sebuah program lebih jelas, ringkas dan meminimalisir kesalahan dalam penafsiran. *Flowchart* sistem menunjukan kontrol sebuah sistem secara fisik dan menjelaskan hal yang dikerjakan oleh sistem. Sedangkan *flowchart* program menunjukan kontrol sebuah program dalam sebuah sistem, berisi rincian langkah-langkah dari proses program.

2.6 Basis Data

Basis data adalah kumpulan informasi yang disimpan di dalam komputer secara sistematik sehingga dapat diperiksa menggunakan suatu program komputer untuk memperoleh informasi dari basis data tersebut. Perangkat lunak yang digunakan untuk mengelola dan memanggil *query* basis data disebut dengan sistem manajemen basis data (*Database Management System*, DBMS). Basis data menyediakan *Data Definition Language* untuk menentukan skema basis data dan *Data Manipulation Language* untuk mengekspresikan permintaan dan pembaruan basis data[13].

Prinsip kerja dan tujuan basis data sama dengan prinsip kerja dan tujuan dari lemari arsip. Tujuan utama dari basis data adalah mengatur data sehingga diperoleh kemudahan, kecepatan dan ketepatan dalam memanggil kembali data yang diinginkan. Manfaat dari basis data bagi pengguna adalah [14]:

1. Kemudahan dan kecepatan, sistem basis data memudahkan dalam memilah data menjadi satu kelompok secara berurutan sehingga dapat mempercepat proses pencarian.
2. Multi-user, basis data memberikan kemudahan akses bagi satu atau beberapa pengguna dalam waktu yang bersamaan.
3. Keamanan data, sistem basis data akan mengamankan data yang ada dalam sistem tersebut melalui penggunaan password, sehingga hanya pihak diizinkan yang bisa mengakses data.
4. Penghematan biaya perangkat, satu basis data terpusat membuat perusahaan tidak lagi membutuhkan hard disk di setiap komputer.
5. Kontrol data terpusat, satu server terpusat untuk melakukan penyimpanan data memudahkan banyak pengguna di cabang lain untuk mengakses data tersebut.

Basis data merupakan sekumpulan informasi yang disimpan dalam komputer sehingga dapat dilakukan pemeriksaan, pengolahan, atau manipulasi secara sistematis menggunakan program komputer. Penggunaan basis data membuat pengguna dapat menyimpan informasi dalam media lain dan mengakses kembali ketika diperlukan. Dalam proses manipulasi data, basis data memerlukan peranan dari *Database Management System.*

2.6.1 Data Definition Language (DDL)

DDL adalah perintah-perintah yang biasa digunakan oleh *Database Administrator* (DBA) untuk menentukan detail implementasi dari skema basis data yang biasanya disembunyikan dari pengguna. DDL dapat digunakan untuk membuat tabel baru, membuat indeks, mengubah tabel dan menentukan struktur tabel. Perintah dasar yang termasuk ke dalam DDL yaitu [15]:

1. *Create*, digunakan untuk membuat database, tabel, *view* dan kolom.
2. *Alter*, digunakan untuk mengubah struktur tabel yang telah dibuat seperti mengganti nama tabel, menambah, mengubah, dan menghapus kolom pada tabel.
3. *Rename*, digunakan untuk mengubah nama objek.
4. *Drop*, digunakan untuk menghapus *database* dan menghapus tabel.

Data Definition Language (DDL) adalah struktur basis data yang menggambarkan desain basis data secara keseluruhan. DDL merupakan bahasa dasar dalam tahap awal penerapan basis data di MySQL yang digunakan untuk membuat dan menghapus basis data, membuat, mengubah dan menghapus tabel serta view. DDL memperbolehkan pengguna untuk mendeskripsikan struktur basis data misalnya merinci tipe dan batasan data yang akan disimpan dalam basis data serta menentukan kunci field dan relasinya [16].

2.6.2 Data Manipulation Language (DML)

DML adalah bahasa yang memungkinkan pengguna untuk mengakses atau memanipulasi data sebagaimana diatur oleh model data yang sesuai. DML juga digunakan untuk memasukkan, merubah, dan menghapus data di dalam sebuah tabel [15]. DML adalah bentuk bahasa basis data yang digunakan untuk melakukan manipulasi atau pengelolaan data yang ada dalam basis data atau tabel. DML merupakan bahasa yang memperbolehkan pengguna untuk mengakses atau memanipulasi data yang ada di basis data. DML mengacu pada kumpulan perintah-perintah yang bisa digunakan untuk melakukan manipulasi data seperti menyimpan data ke dalam tabel, mengubah data, menghapus data dan menampilkan data kembali. Manipulasi data meliputi perintah berikut [16]:

1. *Insert* berfungsi untuk memasukkan atau menyisipkan penambahan data baru ke tabel dalam basis data.
2. *Update* untuk mengubah atau memperbarui data lama dengan data baru pada basis data.
3. *Select* berfungsi untuk menampilkan atau memilih data dari satu tabel atau beberapa tabel dalam relasi pada basis data.
4. *Delete* berfungsi untuk menghapus data dari tabel.

2.6.3 MySQL

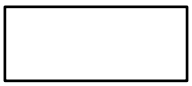
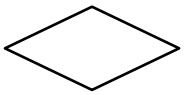
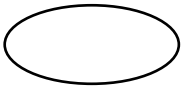
MySQL adalah sistem manajemen basis data relasional (RDBMS) *open source* yang populer dan banyak digunakan di seluruh dunia. MySQL menyediakan fitur-fitur lengkap dan cepat, seperti dukungan untuk bahasa pemrograman, keamanan yang kuat, skalabilitas yang tinggi, dan dukungan untuk berbagai platform. MySQL juga memiliki kemampuan untuk mengelola jumlah data yang sangat besar dan dapat diintegrasikan dengan berbagai aplikasi dan teknologi [17].

MySQL adalah sebuah implementasi dari sistem manajemen basis data relasional (RDBMS) yang didistribusikan secara gratis. Setiap pengguna dapat secara bebas menggunakan MySQL, namun dengan batasan perangkat lunak tersebut tidak boleh dijadikan produk turunan yang bersifat komersial. MySQL sebenarnya merupakan turunan salah satu konsep utama dalam basis data yang telah ada sebelumnya yaitu SQL (Structured Query Language) [18]. MySQL menjadi salah satu database terpopuler karena memiliki banyak kelebihan yaitu bersifat *open-source*, memiliki keamanan yang terjamin, *multi-user*, struktur tabel fleksibel dan mendukung berbagai macam data.

2.6.4 Entity Relationship Diagram (ERD)

Entity Relationship Diagram (ERD) merupakan teknik yang digunakan untuk memodelkan kebutuhan data dari suatu sistem. ERD menyediakan cara untuk mendeskripsikan perancangan basis data pada perangkat. ERD merupakan suatu model untuk menjelaskan hubungan antardata dalam basis data berdasarkan objek-objek dasar data yang mempunyai hubungan antar relasi. Secara sederhana pengertian ERD adalah sebuah konsep yang mendeskripsikan hubungan penyimpanan (*database*) dan sekumpulan objek yang disebut sebagai entitas dan hubungan atau relasi antar objek-objek tersebut. ERD berfungsi untuk memodelkan struktur data dan hubungan antar data, untuk menggambarkannya digunakan beberapa simbol [19] :

Tabel 2. 3 Sombol-simbol ERD

NO	NAMA	NOTASI	KETERANGAN
1.	*Entity*		*Entity*, data inti yang harus disimpan datanya agar dapat diakses oleh aplikasi komputer.
2.	*Relationship*		*Relationship*, hubungan yang terjadi antara satu atau lebih entitas.
3.	*Attribut*		*Attribut*, kolom atau *field* data yang disimpan dalam suatu entitas.
4.	*Key Attribut*		*Key Attribut*, kolom yang disimpan dalam suatu entitas dan digunakan sebagai kunci akses *record* yang diinginkan, biasanya berupa field id.
5.	*Derived attribute*		*Derived attribute*, suatu atribut yang nilainya dihitung atau berdasarkan atribut lain. Atribut ini dapat disimpan atau tidak disimpan di database.

Dalam ERD, hubungan atau relasi terdiri dari sejumlah entitas yang disebut dengan derajat relasi. Derajat relasi maksimum disebut dengan kardinalitas, sedangkan derajat minimum disebut modalitas. Kardinalitas ERD ada dua yaitu [20]:

1. Kardinalitas Entitas, yaitu penentuan banyaknya relasi suatu entitas dengan entitas lainnya.
2. Kardinalitas Atribut, merupakan representasi dari jumlah entitas dalam bentuk alur data yang didefinisikan melalui DFD.

Terdapat tiga jenis relasi yang digunakan dalam ERD, diantaranya yaitu :

1. One to One, pada relasi ini setiap entitas hanya boleh memiliki relasi dengan satu entitas yang lain.
2. One to Many, yaitu hubungan antara satu entitas dengan beberapa entitas dan begitu pula sebaliknya.
3. Many to Many, merupakan hubungan antara beberapa entitas yang memiliki lebih dari satu relasi.

Model ERD sangat berguna dalam memetakan makna dan interaksi perusahaan dari dunia nyata ke dalam skema konseptual. Karena memiliki banyak manfaat, banyak alat desain database menggunakan konsep dari model *Entity Relationship*. Model data ER menggunakan tiga konsep dasar yaitu set entitas, set relasi dan atribut.

2.7 Hypertext Processor (PHP)

PHP digunakan sebagai bahasa *script server-side* dalam pengembangan web yang disisipkan pada dokumen HTML. Penggunaan PHP memungkinkan web dapat dibuat dinamis sehingga *maintenance* situs web tersebut menjadi lebih mudah dan efisien. PHP merupakan *software open-source* yang disebarkan dan dilisensikan secara gratis serta dapat didownload secara bebas dari situs resminya [21]. Keunggulan PHP dibandingkan dengan bahasa pemrograman lainnya diantaranya [22] :

1. Bahasa pemrograman PHP adalah sebuah bahasa skrip yang tidak melakukan sebuah kompilasi dalam penggunaanya.
2. Web server yang mendukung PHP mudah ditemukan dengan konfigurasi yang relatif mudah.
3. Pada sisi pemahaman, PHP adalah bahasa *scripting* yang paling mudah karena memiliki referensi yang banyak.
4. Dalam sisi pengembangan lebih mudah, karena banyaknya *developer* yang siap membantu dalam pengembangan.
5. PHP adalah bahasa *open sources* yang dapat digunakan di berbagai mesin (Linux, Unix,Windows).

Bahasa pemrograman PHP paling banyak digunakan oleh programmer dalam pembuatan website dinamis.hal tersebut karena sudah banyak dokumentasi/tutorial pembahasan yang tersebar di internet yang memudahkan *programmer* pemula. Selain itu, PHP bersifat *open source* sehingga memudahkan dalam penggunaan.

2.8 Framework Laravel

Framework adalah sekumpulan skrip yang dapat membantu *developer/programmer* dalam menangani berbagai masalah dalam pemrograman, seperti *koneksi* ke *database*, pemanggilan variabel, file dan lainnya sehingga pekerjaan *developer* lebih fokus dan lebih cepat dalam membangun aplikasi. *Framework* merupakan komponen pemrograman yang dapat digunakan berulang sehingga *programmer* tidak harus membuat skrip untuk tugas yang sama [23]. Laravel dirilis di bawah lisensi MIT dengan kode sumber yang sudah disediakan oleh Github, Laravel dibangun dengan konsep MVC (*Model-Controller-View*). Laravel juga dilengkapi *command line tool* yang bernama "*Artisan*". Laravel mempunyai kelebihan yaitu [24]:

1. Ekspresif, Laravel adalah *framework* PHP yang ekspresif, sehingga seorang *programmer* diharapkan akan langsung tahu kegunaan dari sintak meskipun belum pernah mempelajari atau mempergunakannya.
2. Simpel, salah satu yang membuat laravel simpel adalah penggunaan *Eloquent ORM*, sehingga ketika ingin mengambil semua data dalam tabel, *programmer* hanya perlu membuat sebuah *class* model.
3. *Accessible*, Laravel dibuat dengan dokumentasi yang lengkap sehingga framework akan lebih mudah untuk digunakan.

Laravel menjadi *framework* PHP paling populer akhir-akhir ini, hal itu karena laravel memiliki template yang mempermudah pekerjaan *web developer*. Laravel juga dilengkapi dengan dokumentasi atau *library* yang lengkap sehingga memudahkan *web developer* pemula. Penggunaan model MVC (*Model View dan Controller*) membuat dokumen lebih terstruktur sehingga memudahkan dalam pencarian file yang digunakan untuk tampilan, konfigurasi database dan logika.

2.9 Website

Website merupakan kumpulan halaman yang digunakan untuk menampilkan informasi teks, gambar diam atau bergerak, animasi, suara, dan gabungan dari semuanya baik yang bersifat statis maupun dinamis. Website juga dapat dikatakan sebagai kumpulan halaman dalam suatu domain yang memuat tentang berbagai informasi agar dapat dibaca dan dilihat oleh pengguna internet melalui sebuah mesin pencari [25]. Website merupakan sejumlah halaman web yang memiliki topik saling berkaitan antara halaman yang satu dengan halaman yang lain. Sebuah website ditempatkan pada sebuah server web yang dapat diakses melalui jaringan seperti internet ataupun jaringan wilayah lokal (LAN) [26]. Website sangat digemari dalam pengimplementasian suatu sistem, hal tersebut karena fleksibilitas yang dimilikinya. Selama perangkat tersambung dengan internet dan dapat mengakses *browser*, website dapat digunakan dimanapun dan kapanpun. Website digunakan dalam hampir semua bidang mulai dari *branding* bisnis, menyajikan konten, sampai promosi produk.

2.10 Penjualan dan Pemesanan

Penjualan merupakan suatu kegiatan transaksi yang dapat dilakukan oleh dua orang atau dua belah pihak dengan menggunakan alat pembayaran yang sah. Penjualan menjadi salah satu sumber pendapatan seseorang atau suatu perusahaan yang melakukan jual beli. Sebuah perusahaan akan mendapat penghasilan yang besar ketika penjualan yang didapat semakin besar dan sebaliknya pula jika penjualan sedikit dan berkurang maka penghasilannya semakin berkurang [27].

Pemesanan adalah suatu aktivitas yang dijalankan oleh konsumen sebelum membeli. Untuk mewujudkan kepuasan konsumen maka perusahaan harus mempunyai sebuah sistem pemesanan yang baik. Surat pemesanan adalah surat yang dibuat oleh seseorang, sekelompok orang atau organisasi untuk melakukan pemesanan barang atau jasa kepada pihak lain guna memenuhi berbagai kebutuhan mereka. Dalam penulisan surat pemesanan dicantumkan nama barang yang dipesan, jumlah atau kuantitas barang yang dipesan, harga per unit barang yang dipesan, dan spesifikasi atau keterangan ringkas mengenai barang yang dipesan [28].

Penjualan merupakan suatu kegiatan yang dilakukan antara pelaku usaha yang dilakukan secara langsung oleh konsumen dan proses penjualan pun bisa berupa barang atau jasa yang menjadi suatu komoditas bagi seorang pembeli dengan harga tertentu. Penjualan dan pemesanan saling berkaitan dan harus dilakukan oleh kedua belah pihak yaitu penjual/perusahaan dan konsumen untuk melakukan transaksi jual beli sebuah barang.

2.11 Metode First Expired First Out (FEFO)

Metode FEFO (*First Expired First Out*) adalah sebuah metode dimana menjual produk dengan jangka waktu kadaluarsa pendek terlebih dahulu kepada pelanggan. Dengan kata lain, pemilik usaha tidak perlu memikirkan kapan produk itu masuk melainkan kapan produk itu akan kadaluarsa. Jadi, ia bisa saja menjual produk yang baru saja masuk [29]. Sebagai contohnya adalah sistem penjualan yang ada di apotek. Pelaku usaha apotek akan menjual produk obat sesuai dengan masa kadaluarsa.

Produk yang memiliki masa kadaluarsa paling dekat akan diletakan pada bagian etalase paling depan. Hal tersebut bertujuan agar calon pembeli mengetahui produk obat tersebut. Metode ini memiliki dua kelebihan yang bisa dijadikan sebagai pertimbangan yaitu metode FEFO diklaim mampu memperkecil angka kerugian dari usaha yang sedang dilangsungkan dan mampu mengurangi terjadinya penyimpanan stok yang sudah terlewat masa kadaluarsanya [30].

Metode FEFO merupakan metode pengelolaan barang dengan mengeluarkan atau menjual barang yang memiliki masa kadaluarsa paling dekat terlebih dahulu. Penggunaan metode FEFO mengharuskan pengusaha melakukan pengurutan produk berdasarkan tanggal kadaluarsanya. Metode ini sering digunakan pada perusahaan retail, farmasi, minuman, makanan dan barang lainnya yang memiliki masa kadaluarsa.

(~~*Halaman ini sengaja dikosongkan*~~)