BAB II

LANDASAN TEORI

II.1. Sistem

II.1.1. Definisi Sistem

Sistem merupakan kumpulan dari unsur atau elemen-elemen yang saling berkaitan/berinteraksi dan saling memengaruhi dalam melakukan kegiatan bersama untuk mencapai suatu tujuan tertentu.

1. Menurut Jerry Fithgerald, sistem adalah suatu jaringan kerja dari prosedur-prosedur yang saling berhubungan dan berkumpul bersama-sama untuk melakukan suatu kegiatan atau menyelesaikan suatu sasaran tertentu.
2. Menurut Ludwig Von Bartalanfy, sistem merupakan seperangkat unsur yang saling terkait dalam suatu antarrelasi di antara unsur-unsur tersebut dengan lingkungan.
3. Menurut Anatol Raporot, sistem adalah suatu kumpulan kesatuan dan perangkat hubungan satu sama lain.
4. Menurut L. Ackof, sistem adalah setiap kesatuan secara konseptual atau fisik yang terdiri dari bagian-bagian dalam keadaan saling tergantung satu sama lainnya (Asbon Hendra : 2012).

II.1.2. Karakteristik Sistem

1. Komponen (*Component*)

Suatu sistem terdiri dari sejumlah komponen yang saling berinteraksi, bekerja sama membentuk suatu kesatuan. Komponen-komponen sistem dapat berupa suatu subsistem atau bagian-bagian dari sistem. Setiap sistem, tidak peduli betapa pun kecilnya, selalu mengandung komponen-komponen atau subsistem-subsistem. Setiap subsistem mempunyai sifat-sifat dari sistem untuk menjalankan suatu fungsi tertentu dan memengaruhi proses sistem secara keseluruhan. Suatu sistem dapat mempunyai suatu sistem yang lebih besar yang di sebut supra sistem. Sebagai contoh, suatu perusahaan dapat disebut dengan suatu sistem dan industri yang merupakan sistem yang lebih besar dapat disebut dengan supra sistem. Kalau dipandang industri sebagai suatu sistem, maka perusahaan dapat disebut sebagai subsistem. Demikian juga bila perusahaan dipandang sebagai suatu sistem maka sistem akuntansi adalah subsistemnya.

2. Batas Sistem (*Boundary*)

Batas sistem merupakan daerah yang membatasi antara suatu sistem dengan sistem yang lainnya atau dengan lingkungan luarnya. Batas sistem ini memungkinkan suatu sistem dipandang sebagai suatu kesatuan, karena dangan batas sistem ini fungsi dan tugas dari subsistem yang satu dengan lainnya berbeda tetapi saling berinteraksi. Batas suatu sistem menunjukkan ruang lingkup (*scope*) dari sistem tersebut.

3. Lingkungan Luar Sistem (*Environment*)

Environment merupakan segala sesuatu diluar batas sistem yang memengaruhi operasi dari suatu sistem. Lingkungan liar sistem ini dapat bersifat menguntungkan atau merugikan. Lingkungan luar yang menguntungkan harus dipelihara dan dijaga agar tidak hilang pengaruhnya, sedangkan lingkungan luar yang merugikan harus dimusnahkan atau dikendalikan agar tidak mengganggu operasi sistem.

4. Penghubung Sistem (*Interface*)

Penghubung sistem merupakan media penghubung antara suatu subsistem dengan subsistem yang lainnya untuk membentuk suatu kesatuan sehingga sumber-sumber daya mengalir dari subsistem yang satu ke subsistem yang lainnya. Dengan kata lain, *output* dari suatu subsistem akan menjadi *input* dari subsistem yang lainya.

5. Masukan Sistem (*Input*)

Masukan sistem merupakan energi yang dimasukan kedalam sistem. Masukan dapat berupa masukan perawatan (*Maintenance Input*) adalah energi yang dimasukan supaya sistem tersebut dapat beroperasi. Masukan sinyal (*signal input*) adalah energi yang diperoses untuk untuk didapatkan keluaran. Sebagai contoh, di dalam sistem komputer, program adalah *maintenance input* yang digunakan untuk mengoperasikan komputernya dan data adalah *signal input* untuk diolah menjadi informasi.

6. Keluaran Sistem (*Output*)

Keluaran sistem merupakan hasil dari energi yang diolah oleh sistem, meliputi *output* yang berguna, contohnya informasi yang dikeluarkan oleh komputer. Dan *output* yang tidak berguna dikenal sebagai sisa pembuangan, contohnya panas yang dikeluarkan oleh komputer.

7. Pengolahan Sistem (*Process*)

Pengolahan sistem merupakan bagian yang memproses masukan untuk menjadi keluaran yang diinginkan. Contoh CPU pada komputer, bagian produksi yang mengubah bahan baku menjadi barang jadi, serta bagian akuntansi yang mengolah data transaksi menjadi laporan keuangan.

8. Tujuan Sistem (Goal)

Setiap sistem pasti mempunyai tujuan ataupun sasaran yang memengaruhi *input* yang dibutuhkan dan *output* yang dihasilkan. Dengan kata lain, suatu sistem akan dikatakan berhasil kalau pengoperasian sistem itu mengenai sasaran atau tujuannya. Jika sistem tidak mempunyai sasaran, maka operasi sistem tidak akan ada gunanya (Asbon Hendra : 2012).

II.1.3. Klasifikasi Sistem

1. Sistem Abstrak (*Abstract System*)

Sistem yang berupa pemikiran atau ide-ide yang tidak tampak secara fisik. Sebagai contoh, sistem teologia yang merupakan suatu sistem yang menggambarkan hubungan tuhan dengan manusia.

2. Sistem Fisik (*Physical System*)

Sistem fisik merupakan sistem yang ada secara fisik sehingga setiap makhluk dapat melihatnya. Contohnya, sistem komputer, sistem akuntansi, sistem produksi dan lain-lain.

3. Sistem Alamiah (*Natural System*)

Sistem alamiah merupakan sistem yang terjadi melalui proses alam, dalam artian tidak dibuat oleh manusia, seperti sistem tata surya, sistem galaksi, sistem reproduksi dan lain-lain.

4. Sistem Buatan Manusia (*Human Made System*)

Sistem buatan manusia merupakan sistem yang dirancang oleh manusia. Sistem buatan manusia yang melibatkan interaksi manusia dengan mesin disebut *human made system*, contohnya sistem informasi.

5. Sistem Tertentu (*Deterministic System*)

Beroperasi dengan tingkah laku yang sudah dapat diprediksi. Interaksi bagian-bagiannya dapat dideteksi dengan pasti sehingga keluaran dari sistem dapat diramalkan. Contohnya, sistem komputer.

6. Sistem Tak Tentu (*Probalistic System*)

Sistem yang kondisi masa depannya tidak dapat diprediksi karena mengandung unsur probabilitas. Contohnya, sistem manusia.

7. Sistem Tertutup (*Closed System*)

Sistem yang tidak berhubungan dan tidak terpengaruh dengan sistem luarnya. Sistem ini bekerja secara otomatis tanpa adanya turut campur tangan dari pihak luarnya. Secara teori, sistem tersebut ada, tetapi kenyataannya tidak ada

sistem yang benar-benar tertutup, yang ada hanyalah *relatively closed system* (secara relatif tertutup, tidak benar-benar tertutup).

8. Sistem Terbuka (*Open System*)

Sistem yang berhubungan dan terpengaruh dengan lingkungan luarnya. Lebih spesifik dikenal juga yang disebut dengan sistem terotomatis, yang merupakan bagian dari sistem buatan manusia dan berinteraksi dengan kontrol oleh satu atau lebih komputer sebagai bagian dari sistem yang digunakan dalam masyarakat modern (Asbon Hendra : 2012).

II.2. Sistem Pendukung Keputusan / *Decision Support Sistem (DSS)*

II.2.1. Definisi Sistem Pendukung Keputusan

Menurut Alter, Sistem Pendukung Keputusan (SPK) merupakan sistem informasi interaktif yang menyediakan informasi, pemodelan, dan manipulasi data. Sistem itu digunakan untuk membantu pengambilan keputusan dalam situasi yang semi terstruktur dan situasi yang tidak terstruktur, di mana tak seorang pun tahu secara pasti bagaimana keputusan seharusnya dibuat (Asahar Johar T, dkk: 2016).

Sistem pendukung keputusan adalah sebuah himpunan/kumpulan prosedur berbasis model untuk memproses data dan pertimbangan untuk membantu manajemen dalam pembuatan keputusannya (Raymundus Nandy Irawan, dkk : 2012).

Menurut Gorry dan Scott Morton dalam buku Turban (2005 ; 19) Sistem Pendukung Keputusan adalah Sistem berbasis komputer interaktif, yang

membantu para pengambil keputusan untuk menggunakan data dan berbagai model untuk memecahkan masalah-masalah tidak terstruktur (Nana Yulia Fitri, Nurhadi : 2017)..

Moore dan Chang dalam buku Turban (2005 ; 137) mendefenisikan DSS sebagai sistem yang dapat diperluas untuk mampu mendukung analisis data ad hoc dan pemodelan keputusan, berorientasi terhadap perencanaan masa depan, dan digunakan pada interval yang tidak regular dan tak terencana. Suatu sistem yang diperuntukan untuk membantu pembuat keputusan dalam kondisi keputusan yang kurang terstruktur/semi terstruktur Efraim (Nana Yulia Fitri, Nurhadi : 2017)..

Menurut Man dan Watson dalam buku Udo Richard Franz Averweg (2012 ; 16) Sistem Pendukung Keputusan merupakan suatu sistem interaktif, yang membantu pengambil keputusan melalui penggunaan data dan model-model keputusan untuk memecahkan masalah (Nana Yulia Fitri, Nurhadi : 2017)..

Dari berbagai pengertian Sistem Pendukung Keputusan di atas, dapat disimpulkan bahwa Sistem Pendukung Keputusan adalah sebuah sistem yang berbasis komputer yang dapat membantu pengambilan keputusan untuk memecahkan masalah tertentu dengan memanfaatkan data dan model tertentu (Nana Yulia Fitri, Nurhadi : 2017).

II.2.2. Keuntungan Sistem Pendukung Keputusan

Sistem pendukung keputusan dapat memberikan berbagai manfaat atau keuntungan bagi pemakainya, antara lain :

1. Memperluas kemampuan pengambilan keputusan dalam memproses data/informasi bagi pemakainya.
2. Membantu pengambilan keputusan dalam hal penghematan waktu yang dibutuhkan untuk memecahkan masalah terutama berbagai masalah yang sangat kompleks dan tidak terstruktur.
3. Dapat menghasilkan solusi dengan lebih cepat serta hasilnya dapat diandalkan.
4. Walaupun suatu Sistem Pendukung Keputusan, mungkin saja tidak mampu memecahkan masalah yang dihadapi oleh pengambil keputusan, namun dapat menjadi stimulan bagi pengambil keputusan dalam memahami persoalannya, karena sistem pendukung keputusan mampu menyajikan berbagai alternatif.
5. Dapat menyediakan bukti tambahan untuk memberikan bukti tambahan untuk memberikan pembenaran sehingga posisi pengambil keputusan (Nana Yulia Fitri, Nurhadi : 2017).

II.2.3. Komponen Sistem Pendukung Keputusan

Menurut Efraim Turban (2005 ; 143) komponen - komponen dari Sistem Pendukung Keputusan adalah sebagai berikut :

1. Manajemen Data, mencakup database yang mengandung data yang relevan dan diatur oleh sistem yang disebut *Database Management System* (DBMS).
2. Manajemen Model, merupakan paket perangkat lunak yang memasukkan model-model finansial, statistik, ilmu manajemen, atau model kuantitatif yang

lain yang menyediakan kemampuan analisis sistem dan *management software* yang terkait.

3. Antarmuka Pengguna, media interaksi antara sistem dengan pengguna, sehingga pengguna dapat berkomunikasi dan memberikan perintah pada SPK melalui subsistem ini.
4. Subsistem Berbasis Pengetahuan, subsistem yang dapat mendukung subsistem lain atau bertindak sebagai komponen yang berdiri sendiri (Nana Yulia Fitri, Nurhadi : 2017).

II.2.4. Tujuan Sistem Pendukung Keputusan

Tujuan dari Sistem Pendukung Keputusan adalah:

1. Membantu dalam pengambilan keputusan atas masalah yang terstruktur.
2. Memberikan dukungan atas pertimbangan manager dan bukan dimaksudkan untuk mengganti fungsi manager.
3. Meningkatkan efektifitas keputusan yang diambil lebih dari pada perbaikan efesiensinya.
4. Kecepatan komputasi komputer memunggkinkan para pengambil keputusan untuk banyak melakukan komputasi secara cepat (Irsanti Merina, Dita Rizki Amalia : 2015).

II.3. Algoritma *K-Nearest Neighbor* (KNN)

K-Nearset Neighbor (K-NN) adalah suatu model penunjang keputusan yang tidak berbasis aturan, melainkan menggunakan algoritma pembelajaran

terawasi, dimana hasil dari data masukan yang baru diklasifikasi berdasarkan kedekatan dengan data pada kasus lama (fakta) sebagai data pelatihan (Bahar, Nidia Rosmawati : 2014).

K-Nearest neighbor merupakan salah satu metode untuk mengambil keputusan menggunakan pembelajaran terawasi dimana hasil dari data masukan yang baru diklasifikasi berdasarkan terdekat dalam data nilai (Raymundus Nandy Irawan, dkk : 2012)

Kedekatan didefinisikan dalam jarak metrik, seperti jarak *Euclidean*. Jarak *Euclidean* dapat dicari dengan menggunakan persamaan 1 berikut ini:

$$Dxy = \sqrt{\sum_{i=1}^{n}(x_i - y_i)^2} \quad \text{.....[1]}$$

Keterangan:

D : Jarak kedekatan

x : data training

y : data testing

n : jumlah atribut individu antara 1 s.d n

f : fungsi *similitary* atribut i antara kasus X dan kasus Y

r : Atribut individu antara 1 sampai dengan n

Langkah-langkah untuk menghitung metode *K-Nearest Neighbor* antara lain :

1. Menentukan parameter K (jumlah tetangga paling dekat).
2. Menghitung kuadrat jarak *Euclid* (*queri instance*) masing-masing objek terhadap data sampel yang diberikan menggunakan persamaan 1.
3. Kemudian mengurutkan objek-objek tersebut ke dalam kelompok yang mempunyai jarak *Euclid* terkecil.

4. Mengumpulkan kategori Y (Klasifikasi *Nearest Neighbor*)
5. Dengan menggunakan kategori *Nearest Neighbor* yang paling mayoritas maka dapat diprediksi nilai *query instance* yang telah dihitung (Asahar Johar T, dkk : 2016).

II.4. *Unified Modeling Language* (UML)

Unified Modeling Language (UML) adalah kumpulan notasi grafis yang didukung oleh sebuah meta-model tunggal, yang membantu dalam menjelaskan dan merancang sistem perangkat lunak, khususnya sistem perangkat lunak dibangun menggunakan gaya berorientasi objek (Andy Prasetyo Utomo : 2013).

UML terdiri atas banyak elemen-elemen grafis yang digabungkan membentuk diagram. Tujuan representasi elemen-elemen grafis ke dalam diagram adalah untuk menyajikan beragam sudut pandang dari sebuah sistem berdasarkan fungsi masing-masing diagram tersebut. Kumpulan dari beragam sudut pandang inilah yang kita sebut sebuah model(Andy Prasetyo Utomo : 2013).

Menurut Windu Gata, Grace (2013:4), *Unified Modeling Language* (UML) adalah bahasa spesifikasi standar yang dipergunakan untuk mendokumentasikan, menspesifikasikan dan membanngun perangkat lunak. UML merupakan metodologi dalam mengembangkan sistem berorientasi objek dan juga merupakan alat untuk mendukung pengembangan sistem (Ade Hendini : 2016). Adapun beberapa jenis diagram pada UML yang dapat membantu perancangan sistem menurut Yasin adalah sebagai berikut :

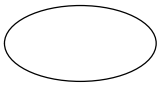
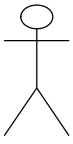
1. *Use Case Diagram*

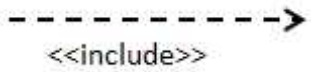
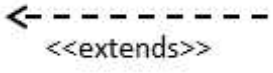
Use case diagram menggambarkan fungsionalitas yang diharapkan dari sebuah sistem. yang ditekankan adalah “apa” yang diperbuat sistem, dan bukan “bagaimana”. Sebuah *use case* merepresentasikan sebuah interaksi antara aktor dengan sistem Seorang/sebuah aktor adalah sebuah entitas manusia atau mesin yang berinteraksi dengan sistem untuk melakukan pekerjaan-pekerjaan tertentu (Andy Prasetyo Utomo : 2013).

Use case diagram merupakan pemodelan untuk kelakuakn (*behavior*) sistem informasi yang akan dibuat. *Use case* digunakan untuk mengetahui fungsi apa saja yang ada di dalam sistem informasi dan siapa saja yang berhak menggunakan fungsi-fungsi tersebut (Ade Hendini : 2016).

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

Tabel II.2. Simbol-Simbol *Use Case Diagram*

Simbol	**Keterangan**
Use Case 	*Use Case* menggambarkan fungsionalitas yang disediakan sistem sebagai unit-unit yang bertukar pesan antar unit dengan aktir, yang dinyatakan dengan menggunakan kata kerja
Aktor / *Actor* 	*Actor* atau Aktor adalah Abstraction dari orang atau sistem yang lain yang mengaktifkan fungsi dari target sistem. Untuk mengidentifikasikan aktir, harus ditentukan pembagian tenaga kerja dan tugas-tugas yang berkaitan dengan peran pada konteks target sistem. Orang atau sistem bisa muncul dalam beberapa peran. Perlu dicatat bahwa aktor berinteraksi dengan *Use Case*, tetapi tidak memiliki kontrol terhadap *use case*
Asosiasi / *Association*	Asosiasi antara aktor dan *use case*, digambarkan dengan garis tanpa panah yang mengindikasikan siapa atau apa yang meminta interaksi secara langsung dan bukannya mengindikasikan data.

	Asosiasi antara aktor dan use case yang menggunakan panah terbuka untuk mengindikasikan bila aktor berinteraksi secara pasif dengan sistem
	Include, merupakan di dalam use case lain (required) atau pemanggilan use case oleh use case lain, contohnya adalah pemanggilan sebuah fungsi program
	Extend, merupakan perluasan dari use case lain jika kondisi atau syarat terpenuhi

(Sumber : Ade Hendini : 2016)

2. *Class Diagram*

Class Diagram adalah sebuah spesifikasi yang jika diinstansiasi akan menghasilkan sebuah obyek dan merupakan inti dari pengembangan dan *desain* berorientasi obyek. Sebuah *class* diagram digunakan untuk menunjukkan keberadaan dari kelas dan hubungannya di dalam pandangan *logic* dari sebuah sistem. Sebuah kelas tunggal merepresentasikan sebuah sudut pandang dari struktur kelas dari sebuah sistem. *Class* menggambarkan keadaan atribut/ properti suatu sistem, sekaligus menawarkan layanan untuk memanipulasi keadaan tersebut (metoda/fungsi) (Andy Prasetyo Utomo : 2013).

Class Diagram Merupakan hubungan antar kelas dan penjelasan detail tiap-tiap kelas di dalam model desain dari suatu sistem, juga memperlihatkan aturan-aturan dan tanggung jawab entitas yang menentukan perilaku sistem *Class Diagram* juga menunjukkan atribut-atribut dan operasi-operasi dari sebuah kelas dan *constraint* yang berhubungan dengan objek yang dikoneksikan. *Class Diagram* secara khas meliputi : Kelas (*Class*), Relasi *Assosiations, Generalitation* dan *Aggregation*, attribut (*Attributes*), operasi (*operation/method*) dan *visibility*,

tingkat akses objek eksternal kepada suatu operasi atau attribut. Hubungan antar kelas mempunyai keterangan yang disebut dengan *Multiplicity* atau *Cardinality* (Ade Hendini, vol 4 : 2016).

Berikut ini adalah simbol-simbol yang ada pada class diagram

Tabel II.2. *Multiplicity Class Diagram*

Multiplicity	**Keterangan**
1	Satu dan hanya satu
0..*	Boleh tidak ada atau 1 atau lebih
1..*	1 atau lebih
0..1	Boleh tidak ada, maksimal 1
n..n	Batasan antara. Contoh 2..4 mempunyai arti minimal 2 maksimal4

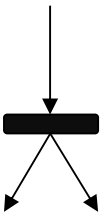
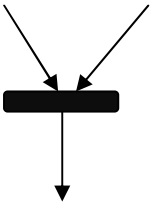
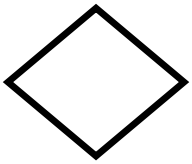
(Sumber : Winda Aprianti, Umi Maliha : 2016)

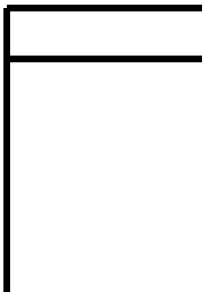
3. *Activity Diagram*

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

Berikut ini adalah simbol-simbol yang ada pada *activity diagram* :

Tabel II.3. Simbol-simbol *Activity Diagram* (Diagram Aktivitas)

Gambar	Keterangan
	*Start Point*, diletakkan pada pojok kiri atas dan merupakan awal aktivitas
	*End Point*, akhir aktivitas
	*Activities*, menggambar kan suatu proses/kegiat an bisnis
	*Fork*/percaban gan, digunakan untuk menunjukkan kegiatan yang dilakukan secara paralel atau untuk menggabung kan dua kegiatan paralel menjadi satu
	*Join* (penggabungan) atau *rake*, digunakan untuk menunjukkan adanya dekomposisi
	*Decision Points*, menggambar kan pilihan untuk pengambilan keputusan, *tru* atau *false*

	*Swimlane*, pembagian *activity diagram* untuk menunjukkan siapa melakukan apa

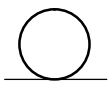
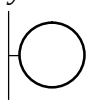
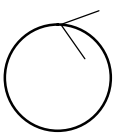
(Sumber : Winda Aprianti, Umi Maliha : 2016)

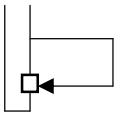
4. *Sequence Diagram*

Sequence Diagram menggambarkan kelakuan objek pada *use case* dengan mendeskripsikan waktu hidup objek dan pesan yang dikirimkan dan diterima antar objek (Ade Hendini : 2016).

Berikut adalah simbol-simbol yang ada pada *Sequence Diagram*:

Tabel II.4. Simbol-Simbol *Sequence Diagram*

Simbol	**Keterangan**
Entity Class 	*Entity Class*, merupakan bagian dari sistem yang berisi kumpulan kelas berupa entitas-entitas yang membentuk gambaran awal sistem dan menjadi landasan untuk menyusun basis data
Boundary Class 	*Boundary Class*, berisi kumpulan kelas yang menjadi interfaces atau interaksi antara satu atau lebih aktor dengan sistem, seperti tampilan form entry dan form cetak
Control class 	*Control class*, suatu objek yang berisi logika aplikasi yang tidak memiliki tanggung jawab kepada entitas, contohnya adalah kalkulasi dan aturan bisnis yang melibatkan berbagai objek

Message	*Message*, simbol mengirim pesan antar class
Recursive 	*Recursive*, menggambarkan pengiriman pesan yang dikirim untuk dirinya sendiri
 *Activation*	*Activation*, mewakili sebuah eksekusi operasi dari objek, panjang kotak ini berbanding lurus dengan durasi aktivasi sebuah operasi
Lifeline	*Lifeline*, garis titik-titik yang terhubung dengan objek, sepanjang lifeline terdapat activation

(Sumber : Ade Hendini : 2016)

II.5. *Hypertext Preprocessor* (PHP)

II.5.1. Definisi *Hypertext Preprocessor* (PHP)

PHP (akronim dari PHP *Hypertext Preprocessor*) yang merupakan bahasa pemrogramman berbasis *web* yang memiliki kemampuan untuk memproses data dinamis. PHP adalah bahasa pemrograman *script server-side* yang didesain untuk pengembangan *web*. Selain itu, PHP juga bisa digunakan sebagai bahasa

pemrograman umum. PHP disebut bahasa pemrograman *server side* karena PHP diproses pada komputer *server* (Asahar Johar T, dkk : 2016).

PHP adalah salah satu bahasa pemrograman yang berjalan dalam sebuah *web server* dan berfungsi sebagai pengolah data pada sebuah *server*. Dengan menggunakan PHP, sebuah *website* akan lebih interaktif dan dinamis. Data yang dikirim oleh pengunjung *website*/komputer *client* akan diolah dan disimpan dalam *database web server* dan dapat ditampilkan kembali apabila diakses (Raymundus Nandy Irawan, dkk : 2012).

II.5.2. Kelebihan *Hypertext Preprocessor* (PHP)

Adapun kelebihan-kelebihan dari PHP yaitu :

1. PHP mudah dibuat dan kecepatan akses tinggi.
2. PHP dapat berjalan dalam *web server* yang berbeda dalam sistem operasi yang berbeda pula.
3. PHP diterbitkan secara gratis.
4. PHP merupaka bahasa yang dapat diletakkan dalam tag HTML.
5. Sistem *database* yang didukung PHP cukup banyak.
6. PHP termasuk *server side programming*.
7. Salah satu fitur yang dapat diandalkan oleh PHP adalah dukungannya terhadap banyak database meskipun dengan kelengkapan yang berbeda-beda (Irsanti Merina TamalaSari dan Dita Rizki Amalia : 2015).

II.6. *My Strukture Query Language (MySQL)*

II.6.1. Definisi *My Strukture Query Language (MySQL)*

MySQL adalah salah satu program yang dapat digunakan sebagai *database* dan merupakan salah satu *software* untuk *database server* yang banyak digunakan. MySQL bersifat *Open Source* dan menggunakan SQL. MySQL bisa dijalankan diberbagai *platform* misalnya *Windows, Linux,* dan lain sebagainya.

Pada MySQL, sebuah *database* mengandung satu atau sejumlah tabel. Tabel terdiri atas sejumlah baris dan setiap baris mengandung satu atau beberapa kolom (Irsanti Merina TamalaSari dan Dita Rizki Amalia : 2015).

Database MYSQL bersifat *open source* dan mampu menangani data yang sangat besar hingga ukuran *Giga Byte*, dengan kemampuan daya tampung data ini maka MySQL sangat cocok digunakan untuk mengcover data pada perusahaan baik yang kecil sampai perusahaan besar (Raymundus Nandy Irawan, dkk : 2012).

II.6.2. Kelebihan My Strukture Query Language (MySQL)

MySQL memiliki beberapa kelebihan , antara lain :

1. MySQL dapat digunakan oleh berbagai *user* dalam waktu yang bersamaan tanpa mengalami masalah.
2. MySQL memiliki kecepatanyang bagus dalam menangani *query* sederhana.
3. MySQL memiliki operator dan fungsi secara penuh dan mendukung perintal *Select* dan *Where* dalam perintah *query*.

4. MySQL memiliki keamanan yang bagus karena beberapa lapisan sekuritas seperti *level subnetmask*, nama *host*, dan izin akses *user* dengan sistem perijinan yang mendetail serta sandi terenkripsi.
5. MySQL dapat melakukan koneksi dengan *client* menggunakan protokol TCP/IP, *Unix soket* (UNIX), atau *Named Pipes* (NT).
6. MySQL dapat berjalan setabil pada berbagai sistem informasi seperti *Windows, Linux, FreeBSD, Mac Os Server, Solaris*, dan masih bnayak lagi.
7. MySQL didistribusikan secara gratis (Irsanti Merina TamalaSari dan Dita Rizki Amalia : 2015).

II.7. Database

Basis data atau *database* adalah kumpulan terintegrasi dari elemen data yang secara logika saling berhubungan. Basis data mencatat dan mengonsolidasikan berbagai catatan yang dahulu disimpan dalam *file-file* terpisah kedalam satu gabungan umum elemen data yang menyediakan data yang menyediakan data untuk banyak aplikasi. Jadi, basis data berisi berbagai elemen data yang mendiskripsikan berbagai entitas dan hubungan antar entitas (Basuki Rahmad, Bambang Eka Purnama : 2013).

Database sering didefinisikan sebagai kumpulan data yang terkait. Secara teknis, yang berada dalam sebuah *database* adalah sekumpulan tabel atau objek lain (*indeks, view* dan lain-lain). Tujuan utama pembuatan *database* adalah untuk memudahkan dalam mengakses data. Data dapat ditambahkan, diubah, dihapus, atau dibaca dengan relatif mudah dan cepat. Saat ini tersedia banyak perangkat

lunak yang ditujukan untuk mengelola *database*. Perangkat lunak seperti itu dinamakan DBMS (*Database Management System*). Berikut ini beberapa contoh perangkat lunak atau DBMS :

1. *Microsoft SQL Server*
2. *Oracle*
3. *MySQL*
4. *PostgreSQL*
5. *Microsoft Access*
6. *Paradox*
7. *Visual FoxPro* (Irsanti Merina TamalaSari dan Dita Rizki Amalia : 2015).

II.8. Normalisasi

Normalisasi adalah suatu teknik untuk mengorganisasikan data kedalam tabel-tabel untuk memenuhi kebutuhan pemakai didalam suatu organisasi (Ganda Yoga Swara, M.Kom, Yunes Pebriadi : 2016).

Tujuan dari normalisasi adalah :

1. Untuk menghilangkan kerangkapan data.
2. Untuk mengurangi kompleksitas.
3. Untuk mempermudah pemodifikasian (Ganda Yoga Swara, M.Kom, Yunes Pebriadi : 2016).

Proses normalisasi antara lain :

1. Data diuraikan dalam bentuk tabel, selanjutnya dianalisis berdasarkan persyaratan tertentu kebeberapa tingkat.

2. Apabila tabel yang diuji belum memenuhi persyaratan tertentu maka tabel tersebut perlu dipecah menjadi beberapa tabel yang lebih sederhana sampai memenuhi data yang optimal (Ganda Yoga Swara, M.Kom, Yunes Pebriadi : 2016).

Bentuk-bentuk dari normalisasi adalah :

1. Bentuk tidak normal (*unformalized form*)

Bentuk ini merupakan bentuk data yang direkam, tidak ada keharusan untuk mengikuti suatu format tertentu, dapat saja data tidak lengkap atau terduplikasi.

2. Bentuk normal pertama (1NF atau *first normal form*)

Bentuk normal pertama mempunyai ciri-ciri yaitu setiap data dibentuk dalam *flat-file* (*file* dasar) dan data dibentuk dalam satu *record* demi satu *record*. Tidak ada set atribut yang berulang-ulang atau atribut yang bernilai ganda.

3. Bentuk normal kedua (2NF atau *second normal form*)

Bentuk normal kedua mempunyai syarat yaitu bentuk data telah memenuhi kriteria bentuk normal pertama, atribut bukan kunci haruslah bergantung secara fungsi pada kunci utama, atau *primery key*, sehingga untuk bentuk normal kedua haruslah sudah ditentukan kunci-kunci *field*. Kunci *field* harus unik dan dapat mewakili atribut lain yang menjadi anggotanya.

4. Bentuk normal ketiga (3NF atau *three normal form*)

Untuk menjadi bentuk normal ketiga maka relasi haruslah dalam bentuk normal kedua dan sama atribut bukan *primer* tidak punya hubungan yang transi,

dengan kata lain setiap atribut bukan kunci haruslah bergantun pada *primery key* secara menyeluruh (Ganda Yoga Swara M.kom, Yunes Pebriadi :2016).

II.9. *Crode Palm Oil* (CPO)

Minyak sawit kasar (*Crude Palm Oil*) merupakan minyak nabati berwarna jingga kemerah-merahan yang diperoleh dari proses ekstraksi daging buah kelapa sawit (*mesocarp*) tanaman *Elais guinensis Jacq*. Minyak sawit kasar terdiri dari *gliserida* yang tersusun oleh serangkaian asam lemak. Komponen utama minyak sawit adalah *trigliserida* dengan sebagian kecil *digliserida* dan *mono gliserida*. Minyak sawit kasar berbentuk semipadat pada suhu kamar. Warna minyak sawit kasar yang berwarna jingga kemerah-merahan disebabkan oleh komponen *minor* yang dimiliki CPO berupa *pigmen karoten* (M. Fajar Wulan D : 2014).