BAB II
LANDASAN TEORI

2.1 Sistem Informasi

Sistem informasi merupakan kombinasi teratur dari orang-orang, perangkat keras (*hardware*), perangkat lunak (*software*), jaringan komunikasi dan sumber daya data yang mengumpulkan, mengubah dan menyebarkan informasi dalam sebuah organisasi. Sistem informasi juga dapat didefinisikan sebagai suatu sistem yang dibuat oleh manusia yang terdiri dari komponen-komponen dalam organisasi untuk menyajikan informasi.

Sistem informasi merupakan sebuah sususan yang terdiri dari beberapa komponen atau elemen. Komponen sistem informasi disebut dengan istilah blok bangunan (*building block*). Komponen sistem informasi tersebut terdiri dari :

1. Blok masukan (*input block*), *input* memiliki data yang masuk kedalam sistem informasi, juga metode-metode untuk menangkap data yang dimasukkan.
2. Blok *Model* (*Model block*), blok ini terdiri dari kombinasi prosedur logika dan matematik yang akan memanipulasi data *input* dan data yang tersimpan di basis data
3. Blok keluaran (*outuput block*), produk dari sistem informasi adalah keluaran yang merupakan informasi berkualitas dan dokumentasi yang berguna untuk semua tingkatan manajemen serta semua pemakai sistem.
4. Blok teknologi (*technology block*), blok teknologi digunakan untuk menerima input, menyimpan, mengakses data, menghasilkan dan mengirimkan keluaran dari sistem secara keseluruhan. Teknologi terdiri dari tiga bagian utama yaitu teknisi (*brainware*), perangkat lunak (*software*) dan perangkat keras (*hardware*).
5. Basis data (*database block*), basis data merupakan kumpulan data yanh saling berhubungan satu dengan yang lainnya, tersimpan di perangkat keras komputer dan digunakan perangkat lunak (*software*) untuk memanipulasi.

Setiap sistem informasi memiliki aktivitas pemrosesan informasi dasar atau pemrosesan data. Aktivitas sistem informasi meliputi: input data, pemrosesan, *output* data, penyimpanan data dan pengendalian yang ada dalam sistem informasi.

a. *Input* sumber daya data, data mengenai transaksi bisnis dan kegiatan lainnya harus ditangkap dan disiapkan untuk pemrosesan melalui aktivitas *input*. Input biasanya berbentuk aktivitas entri data seperti; pencatatan dan pengeditan. Para pemakai akhir biasanya memasukkan data secara langsung kedalam sistem komputer atau mencatat data mengenai transaksi dari beberapa jenis media fisik seperti formulir kertas. Hal ini biasanya meliputi berbagai aktivitas edit untuk memastikan bahwa, data telah dicatat dengan benar. Begitu dimasukkan, data dapat dipindahkan ke dalam media yang dapat dibaca mesin seperti magnetic disk hingga untuk pemrosesan.
b. Pemrosesan menjadi informasi, data biasanya tergantung pada pemrosesan seperti perhitungan, perbandingan, pemilahan, pengklasifikasian dan pengihtisaran. Aktivitas tersebut mengatur, menganalisis, dan memanipulasi data hingga mengubahnya ke dalam sistem informasi juga harus dipelihara terus-menerus.
c. *Output* produk informasi, informasi dalam berbagai bentuk yang dikirim kepada pemakai akhir. Tujuan dari sistem informasi adalah untuk menghasilkan produk informasi yang tepat bagi para pemakai akhir. Produk informasi umum meliputi pesan, laporan, formulir dan gambar yang disediakan melalui tampilan video. Audio, kertas dan multimedia.
d. Penyimpana sumber daya data, penyimpanan adalah sistem dasar informasi. Penyimpanan adalah aktivitas sistem informasi tempat data dan informasi disimpan secara teratur. Hal ini diperlukan untuk memfasilitasi penggunaan di masa mendatang untuk pemrosesan atau penarikan *output* ketika dibutuhkan oleh pemakai sistem.
e. Pengendalian kinerja sistem, aktivitas sisten informasi adalah pengendalian kinerja sistem. Sistem informasi harus menghasilkan umpan balik mengenai aktivitas *input*, proses, *output* dan penyimpanan. Umpan balik ini harus

diawasi dan dievaluasi untuk menetapkan apakah sistem telah dapat memenuhi standar kinerja yang telah ditetapkan. (Yakub, 2012).

2.2 Badan Kependudukan dan Keluarga Berencana Nasional (BKKBN)

Badan Kependudukan dan Keluarga Berencana Nasional (BKKBN) adalah Lembaga Pemerintah Nonkementerian yang berada di bawah dan bertanggung jawab kepada Presiden melalui Menteri Kesehatan. BKKBN mempunyai tugas melaksanakan tugas pemerintah di bidang pengendalian penduduk dan penyelenggaraan keluarga berencana. Dalam melaksanakan tugas sebagaimana dimaksud, BKKBN menyelenggarakan fungsi:

a. Perumusan kebijakan nasional, pemaduan dan sinkronisasi kebijakan di bidang KKB;
b. Penetapan norma, standard prosedur dan kriteria di bidang KKB;
c. Pelaksanaan advokasi dan koordinasi di bidang pengendalian penduduk dan KB;
d. Penyelenggaraan komunikasi, informasi dan edukasi di bidang KKB;
e. Penetapan perkiraan pengendalian penduduk secara nasional;
f. Penyusunan desain Program KKBPK;
g. Pengelolaan tenaga penyuluh KB/petugas lapangan KB (PKB/PLKB);
h. Pengelolaan dan penyediaan alat dan obat kontrasepsi untuk kebutuhan Pasangan Usia Subur (PUS) nasional;
i. Pengelolaan dan pengendalian sistem informasi keluarga;
j. Pemberdayaan dan peningkatan peran serta organisasi kemasyarakatan tingkat nasional dalam pengendalian pelayanan dan pembinaan kesertaan ber-KB dan Kesehatan Reproduksi (KR);
k. Pengembangan desain program pembangunan keluarga melalui pembinaan ketahanan dan kesejahteraan keluarga;
l. Pemberdayaan dan peningkatan peran serta organisasi kemasyarakatan tingkat nasional dalam pembangunan keluarga melalui ketahanan dan kesejahteraan keluarga;
m. Standarisasi pelayanan KB dan sertifikasi tenaga penyuluh KB/Petugas lapangan KB (PKB/PLKB);

n. Penyelenggaraan pemantauan dan evaluasi di bidang pengendalian penduduk dan keluarga berencana;
o. Pembinaan pembimbingan dan fasilitas di bidang KKB;

Selain menyelenggarakan fungsi tersebut, BKKBN juga menyelenggarakan fungsi :

a. Penyelenggaraan pelatihan, penelitian dan pengembangan di bidang KKB;
b. Pembinaan dan koordinasi pelaksanaan tugas administrasi umum di lingkungan BKKBN;
c. Pengelolaan barang milik/kekayaan negara yang menjadi tanggung jawab BKKBN;
d. Pengawasan atas pelaksanaan tugas di lingkungan BKKBN;
e. Penyampaian laporan, saran dan pertimbangan di bidang KKB.

(bkkbn.go.id, 2020)

2.3 Pemilihan Duta Generasi Berencana (GenRe)

Dalam rangka merespon permasalahan remaja saat ini, BKKBN mengembangkan Program Generasi Berencana (GenRe). Program GenRe merupakan wadah untuk mengembangkan karakter bangsa dengan mengajarkan remaja untuk menjauhi pernikahan dini, seks pra-nikah, dan NAPZA guna menjadi remaja tangguh dan dapat berkontribusi dalam pembangunan serta berguna bagi nusa dan bangsa.

Untuk mencapai tujuan diatas, maka didalam program GenRe dikembangkan materi-materi mengenai Kesehatan Reproduksi Remaja, *Life Skill*, Penyiapan Kehidupan Berkeluarga, serta Kependudukan dan Pembangunan Keluarga. Program GenRe dilaksanakan melalui pengembangan Pusat Informasi dan Konseling (PIK) Remaja dengan pendekatan dari, oleh, dan untuk remaja sesuai dengan kecenderungan remaja yang lebih menyukai bercerita tentang permasalahannya dengan teman sebaya. Pada saat ini, PIK Remaja berjumlah kurang lebih 23.579 tersebar di 34 Provinsi yang diharapkan menjadi wadah bagi remaja untuk berkumpul, berbagi cerita, berkreatifitas, dan saling tukar informasi. PIK Remaja dikembangkan melalui jalur pendidikan dan masyarakat. Jalur pendidikan meliputi sekolah, perguruan tinggi, dan pesantren. Di jalur masyarakat diantaranya melalui organisasi kepemudaan,organisasi keagamaan, dan komunitas

remaja. Kedua jalur tersebut merupakan sasaran yang penting untuk mendekati komunitas remaja.

Dalam rangka meningkatkan sosialisasi dan promosi program GenRe, khususnya pengembangan PIK Remaja sebagai sebuah wadah pelayanan informasi dan konseling, maka diperlukan *figure* motivator dari kalangan remaja. *Figure* motivator inilah yang akan menjadi wakil atau Duta GenRe Indonesia Provinsi Papua. Dengan adanya Duta GenRe Indonesia Provinsi Papua, sosialisasi dan promosi program GenRe dilingkungan remaja akan lebih efektif karena komunikasi yang terjalin dilakukan dengan pendekatan dari, oleh, dan untuk remaja sehingga menjadi ramah remaja. Disamping itu, dilingkungan remaja secara umum, ikon Duta GenRe Indonesia Provinsi Papua dirasa memberi nilai lebih dalam hal sosialisasi dan promosi Duta GenRe Indonesia Provinsi Papua. (Papua, 2019)

2.4 Website

Menurut Rohi Abdulloh (2015), *website* atau web adalah sekumpulan halaman yang terdiri dari beberapa laman yang berisi informasi dalam bentuk data digital, baik berupa teks, gambar, video, audio, dan animasi lainnya yang disediakan melalui jalur koneksi internet. Lebih jelasnya, *website* merupakan halaman-halaman yang berisi informasi yang dapat diakses oleh *browser* dan mampu memberikan informasi yang berguna bagi para pengaksesnya. (Sa'ad, 2020)

Website merupakan sebuah media yang memiliki banyak halaman yang saling terhubung (hyperlink), dimana website memiliki fungsi dalam memberikan informasi berupa teks, gambara, video, suara dan animasi atau penggabungan dari semuanya. Website pada saat sekarang ini umumnya telah bersifat dinamis, meskipun dahulu juga ada website yang bersifat statis, namun website statis telah jarang dan bahkan hamper tidak ada lagi ditemukan. Karakteristik utama yang dimiliki oleh website adalah halaman-halaman yang saling terhubung, dan dilengkapi dengan domain sebagai alamat (url) atau World Wide Web (www) dan juga hosting sebagai media yang menyimpan banyak data. (Elgamar, 2020)

2.5 Basis Data

Basis Data terdiri dari kata basis dan data. Basis dapat diartikan sebagai markas atau gudang. Sedangkan data adalah catatan atas kumpulan fakta dunia nyata yang mewakili objek seperti manusia, barang, hewan, konsep, peristiwa dan sebagainya yang diwujudkan dalam bentuk huruf, angka, simbol, gambar, teks, bunyi atau kombinasinya.

Sebagai satu kesatuan istilah, Basis Data (*database*) sendiri dapat didefinisikan dalam sejumlah sudut pandang seperti :

a. Himpunan kelompok data (arsip) yang saling berhubungan yang diorganisasi sedemikian rupa sehingga kelak dapat dimanfaatkan kembali dengan cepat dan mudah.
b. Kumpulan data yang saling berhubungan yang disimpan secara bersama sedemikian rupa dan tanpa pengulangan (redundancy) yang tidak perlu, untuk memenuhi berbagai kebutuhan.
c. Kumpulan *file*/tabel/arsip yang saling berhubungan yang disimpan dalam media penyimpanan elektronis. (Fathansyah, 2015)

Basis data adalah kumpulan *file-file* yang saling berelasi, relasi tersebut biasa ditunjukan dengan kunci dari tiap *file* yang ada. Satu basis data menunjukkan kumpulan data yang dipakai dalam satu lingkup informasi. Dalam satu *file* terdapat record-record yang sejenis, sama besar, sama bentuk, merupakan satu kumpulan entity yang seragam. Satu *record* terdiri dari *fieldfield* yang saling berhubungan untuk menunjukan bahwa field tersebut dalam satu pengertian yang lengkap dan direkam dalam satu record. Suatu sistem manajemen basis data berisi satu koleksi data yang saling berelasi dan satu set program untuk mengakses data tersebut. Jadi sistem manajemen basis data dan set program pengelola untuk menambah data, menghapus data, mengambil data dan membaca data. (Rahmad & Setiady, 2014)

2.5.1 Database Language

Bahasa basis data (*database language*) adalah suatu cara untuk berinteraksi atau berkomunikasi antara pemakai dengan basis data yang diatur dalam bahasa khusus yang ditetapkan oleh perusahaan. *Database language* dipilih menjadi 3 yaitu :

1. Data Definition Language

Data Definition Language atau *Data Description Language* (DDL) adalah bahasa untuk mendefinisikan sebuah data. DDL digunakan untuk membuat, mengubah struktur dan mendefinisikan tipe data dari berbagai objek basis data seperti tabel, indeks, trigger, *View* dan lain-lain. DDL digunakan untuk membuat tabel atau menghapus tabel, membuat *key* atau indeks, membuat relasi antartabel. Contoh perintah DDL diantaranya CREATE, DROP dan ALTER.

a. *CREATE TABLE*, bertugas untuk membuat table.
b. *CREATE INDEX*, bertugas untuk membuat suatu indeks dalam table.
c. *DROP TABLE,* bertugas untuk menghapus suatu table.
d. *ALTER TABLE*, bertugas untuk mengubah struktur suatu table.

2. Data Manipulation Language

Data Manipulation Language (DML) adalah perintah yang digunakan pada basis data untuk mengolah data atau informasi. Perintah DML tersebut diantaranya INSERT, UPDATE, DELETE dan SELECT.Berikut ini adalah penjelasan singkat dari sintak-sintak SQL sebagai berikut :

a. *INSERT,* bertugas untuk menambahkan data ke dalam *database.*
b. *UPDATE,* bertugas untuk mengupdate (mengubah) data dalam suatu tabel pada *database.*
c. *SELECT*, bertugas untuk mengakses data dari suatu tabel dalam *database.*
d. *DELETE*, bertugas untuk menghapus data dari suatu tabel dalam *database.*

3. Data Control Language

Data Control Language (DCL) adalah perintah untuk mengatur hak akses terhadap sebuah basis data. DCL terdiri dari dua perintah yakni GRANT dan REVOKE.

a. *GRANT* :Pernyataan *GRANT* ini digunakan untuk memberikan atau mengizinkan seorang *User* untuk mengakses tabel dalam *database* tertentu

b. *REVOKE* :Pernyataan *REVOKE* ini digunakan untuk mencabut suatu hak akses dalam database tertentu. (Ardeman, Muchalil, & Afdhal, 2017)

2.5.2 Normalisasi

Normalisasi merupakan kondisi dari suatu relasi dipandang dari dependensi fungsinya, serta untuk mendekomposisi relasi dengan anomaly menjadi beberapa relasi yang memiliki ukuran lebih kecil tetapi memiliki struktur yang lebih baik. Normalisasi dapat dicapai melalui proses yang bertahap, masing-masing sesuai dengan bentuk normal tertentu. Bentuk normal adalah kondisi relasi yang dihasilkan dari penerapan aturan sederhana mengenai dependensi fungsional (atau relasi antar-atribut) yang memenuhi persyaratan.

1. Bentuk Tidak Normal

Merupakan kumpulan data yang di rekam tidak ada keharusan dengan mengkuti suatu format tertentu. Pada bentuk tidak normal terdapat repeating group (pengulangan group), sehingga pada kondisi ini data menjadi permasalahan dalam melakukan manipulasi data (insert, update, dan delete) atau biasa di sebut anomali.

Tabel 2. 1 Bentuk Tidak Normal

No. Faktur	Tanggal	Kode Pelanggan	Nama	Kode Barang	Nama Barang	Harga	Qty
F001	12/12/2016	P001	M.Fikri Setiadi	B001	Sampo	12.000,-	1
				B002	Kopi	15.000,-	1
F002	13/12/2016	P002	Jack	B002	Kopi	15.000,-	1
				B003	Teh	7.000,-	2

2. Normalisasi Pertama

Tidak ditemukan atribut jamak, sehingga hanya terdapat nilai tunggal (kemungkinan nol) disetiap pertemuan antara baris dan kolom dalam table Contoh perubahan dari bentuk tidak normal ke bentuk normal pertama:

Tabel 2. 2 Tabel Normalisasi Pertama

No. Faktur	Tanggal	Kode Pelanggan	Nama	Kode Barang	Nama Barang	Harga	Qty
F001	12/12/2016	P001	M.Fikri Setiadi	B001	Sampo	12.000,-	1
F001	12/12/2016	P001	M.Fikri Setiadi	B001	Kopi	15.000,-	1
F002	13/12/2016	P002	Jack	B002	Kopi	15.000,-	1
F002	13/12/2016	P002	Jack	B003	Teh	7.000,-	2

3. Normalisasi Kedua

Setiap dependensi fungsional parsial telah dihilangkan (sehingga semua atribut nonkunci hanya diidentifikasi oleh keseluruhan kunci primer). Contoh perubahan dari bentuk normal pertama ke bentuk normal kedua:

Tabel 2. 3Tabel Normalisasi Kedua

Tabel Transaksi				
No.Faktur	**Tanggal**	**Kode Pelanggan**	**Kode Barang**	**Qty**
F001	12/12/2016	P001	B001	1
F001	12/12/2016	P001	B002	1
F002	13/12/2016	P002	B002	1
F002	13/12/2016	P002	B003	2

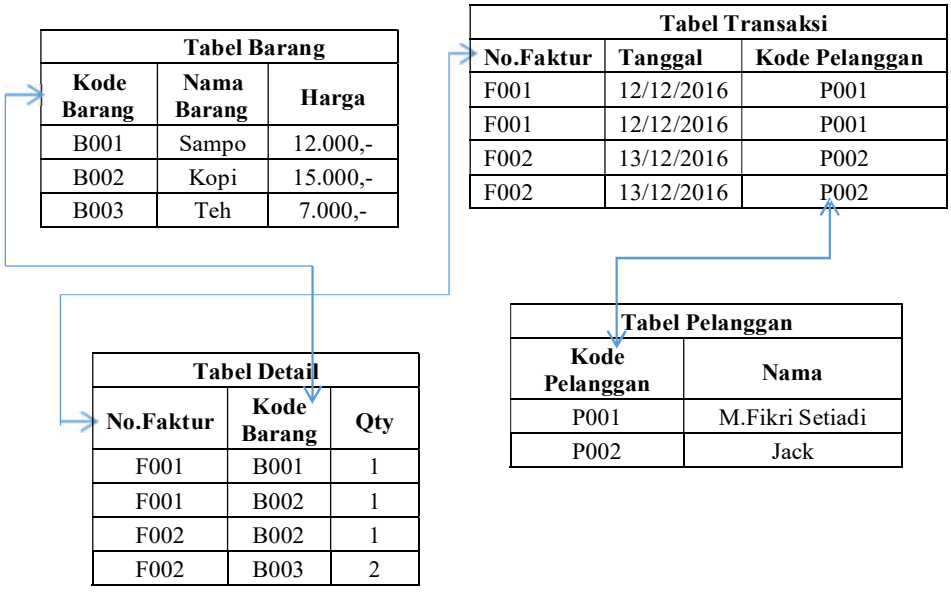
Tabel Pelanggan	
Kode Pelanggan	**Nama**
P001	M. Fikri Setiadi
P002	Jack

Tabel Barang		
Kode Barang	**Nama Barang**	**Harga**
B001	Sampo	12.000,-
B002	Kopi	15.000,-
B003	Teh	7.000,-

4. Normalisasi Ketiga

Setiap dependensi transitif telah dihapus (sehingga semua atribut nonkunci diidentifikasi oleh hanya satu kunci primer). Contoh perubahan dari 2NF ke 3NF:

Tabel 2. 4 Bentuk Normalisasi Ketiga



Tabel Barang

Kode Barang	Nama Barang	Harga
B001	Sampo	12.000,-
B002	Kopi	15.000,-
B003	Teh	7.000,-

Tabel Transaksi

No.Faktur	Tanggal	Kode Pelanggan
F001	12/12/2016	P001
F001	12/12/2016	P001
F002	13/12/2016	P002
F002	13/12/2016	P002

Tabel Detail

No.Faktur	Kode Barang	Qty
F001	B001	1
F001	B002	1
F002	B002	1
F002	B003	2

Tabel Pelanggan

Kode Pelanggan	Nama
P001	M.Fikri Setiadi
P002	Jack

(Reksoatmodjo, 2018)

2.5.3 Relasi Antar Tabel

Relasi adalah hubungan antara tabel yang mempresentasikan hubungan antar objek di dunia nyata. Relasi merupakan hubungan yang terjadi pada suatu tabel dengan lainnya yang mempresentasikan hubungan antar objek di dunia nyata dan berfungsi untuk mengatur mengatur operasi suatu database. Hubungan yang dapat dibentuk dapat mencakup 3 macam hubungan, yaitu one to one, one to many, many to many.

1. One-To-One (1-1)

Mempunyai pengertian "Setiap baris data pada tabel pertama dihubungkan hanya ke satu baris data pada tabel ke dua". Contohnya : relasi antara tabel

mahasiswa dan tabel orang tua. Satu baris mahasiswa hanya berhubungan dengan satu baris orang tua begitu juga sebaliknya.

2. One-To-Many (1-N)

Mempunyai pengertian "Setiap baris data dari tabel pertama dapat dihubungkan ke satu baris atau lebih data pada tabel ke dua". Contohnya : relasi perwalian antara tabel dosen dan tabel mahasiswa. Satu baris dosen atau satu dosen bisa berhubungan dengan satu baris atau lebih mahasiswa.

3. Many-To-Many (N-M)

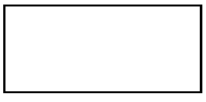
Mempunyai pengertian "Satu baris atau lebih data pada tabel pertama bisa dihubugkan ke satu atau lebih baris data pada tabel ke dua". Artinya ada banyak baris di tabel satu dan tabel dua yang saling berhubungan satu sama lain. Contohnya : relasi antar tabel mahasiswa dan tabel mata kuliah. Satu baris mahasiswa bisa berhubungan dengan banyak baris mata kuliah begitu juga sebaliknya. (Enterprise, 2015)

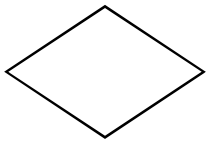
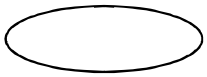
2.5.4 ERD (Entity Relationship Diagram)

ERD adalah tools yang digunakan untuk melakukan pemodelan data secara abstrak dengan tujuan untuk mendeskripsikan atau menggambarkan struktur dari data yang akan digunakan, serta digunakan untuk memodelkan struktur dengan menggambarkan entitas dan hubungan antara entitas (relationship) secara abstrak (konseptual) (Mulyani, 2016).

Simbol-simbol yang digunakan untuk menggambarkan ERD dapat dilihat pada tabel berikut:

Tabel 2. 5Simbol-Simbol ERD

Gambar	Keterangan
 Entitas	Entiti adalah suatu objek yang dapat diidentifikasi dalam lingkungan pemakai, sesuatu yang penting bagi pemakai dalam konteks sistem yang akan dibuat. Sebagai contoh pelanggan, pekerja dan lain-lain. Seandainya A adalah seorang pekerja maka A adalah isi dari pekerja, sedangkan

	jika B adalah seorang pelanggan maka B adalah isi dari pelanggan. Karena itu harus dibedakan antara Entitii sebagai bentuk umum dari deskripsi tertentu dan isi entiti seperti A dan B dalam contoh di atas.Entiti digambarkan dalam bentuk persegi empat.
 Relasi	Relasi (relationship) sebagaimana halnya entiti maka dalam hubunganpun harus dibedakan antara hubungan atau bentuk hubungan antar entiti dengan isi dari hubungan itu sendiri. Misalnya dalam kasus hubungan antara entiti siswa dan entiti mata_kuliah adalah mengikuti, sedangkan isi hubungannya dapat berupa nilai_ujian. Relationship digambarkan dalam bentuk intan / diamonds.
 Atribut	Entiti mempunyai elemen yang disebut atribut, dan berfungsi mendeskripsikan karakter entiti. Misalnya atribut nama pekerja dari entiti pekerja. Setiap ERD bisa terdapat lebih dari satu atribut. Entiti digambarkan dalam bentuk ellips.
 Penghubung	Penghubung antara relasi dan entitas di mana di kedua ujung memiliki kemungkinan jumlah pemakaian. Kemungkinan jumlah pemakaian maksimum keterhubungan antara entitas satu dengan entitas yang lainnya dengan kardinalitas.

(Subandi, 2018)

2.6 *Database Management System* (DBMS)

Database Management System (DBMS) adalah aplikasi yang dipakai unutk mengelola basis data. DBMS biasanya aplikasi menawarkan beberapa kemampuan yang terintergrasi seperti :

1. Membuat, menghapus, menambah dan memodifikasi basis data
2. Pada beberapa DBMS pengelolaanya berbasis *windows* (berbentuk jendela-jendela) sehingga lebih mudah digunakan
3. Tidak semua orang bisa mengakses basis data yang ada sehingga memberikan keamanan bagi data
4. Kemampuan berkomunikasi dengan program aplikasi yang lain. Misalnya dimungkinkan untuk mengakses basis data MySQL menggunakan aplikasi yang dibuat menggunakan HP
5. Kemampuan pengaksesan melalui komunikasi antarkomputer (*client server*)

Kelebihan dari MySQL adalah gratis, handal selalu diupdate dan banyak forum yang memfasilitasi para pengguna jika memiliki kendala. MySQL juga menjadi DBMS yang sering dibunding dengan *web server* sehingga proses instalasinya jadi lebih mudah.

My Structure Query Language (MySQL) atau yang biasa dibaca "*mai-se-kuel*" adalah sebuah program pembuat basis data yang bersifat *open source*. *Open source* memiliki arti bahwa siapa pun boleh menggunakan MySQL dan tidak akan dicekal. Saat kita mendengar *open source*, kita ingat dengan system operasi handal keturunan Unix, yaitu Linux.

MySQL sebenarnya produk yang berjalan pada platform Linux. Karena sifatnya yang *open source*, sehingga dapat dijalankan pada semua platform baik Windows maupun Linux. Selain itu, MySQL merupakan program pengaksesan basis data yang bersifat jaringan sehingga dapat digunakan untuk aplikasi multi*User*. Saat ini, basis data MySQL merupakan salah satu contoh basis data server yang digunakan hampir oleh semua programmer basis data apalagi dalam pemrograman web. Kelebihan lain dari MySQL adalah menggunakan bahasa query standar yang dimiliki *Stucture Query Language* (SQL). SQL adalah suatu

bahasa permintaan yang terstuktur yang telah distandarkan untuk semua program pengakses basis data seperti Oracle, Posgres SQL, dan SQL Server. (Jaya, 2016)

2.6.1 Tipe-Tipe Dalam My SQL

1. Tipe Data Numerik

MySQL menggunakan seluruh tipe data numerik standar ANSI (American National Standards Institute). Berikut ini adalah tipe data numerik yang biasanya digunakan beserta penjelasannya.

Tabel 2. 6 Tipe Data Numerik

Tipe Data	Deskripsi
INT	Nilai integer yang bisa bertanda atau tidak. Jika bertanda, maka rentang yang diperbolehkan adalah -2147483648 sampai 2147483648, sedangkan jika bertanda maka rentangnya dari 0 sampai 4294967295
TINYINT	Nilai integer yang sangat kecil. Rentangnya -128-127 untuk bertanda sedangkan untuk yang tidak bertanda dan 0-255 untuk yang tidak bertanda.
SMALLINT	Nilai integer yang sangat kecil dengan rentang 31768 sampai 32767 untuk yang bertanda sedangkan untuk yang tidak bertanda dari 0-65535
MEDIUMINT	Nilai integer yang sangat kecil dengan rentang -8388608 sampai 8388607 atau 0 sampai 167777215
BIGINT	Integer dengan ukuran besar dengsn rentang -9223372036854775808 sampai 9223372036854775807 atau 0 sampai 18446744073709551615
FLOAT (M,D)	Bilangan pecahan dengan panjang (termasuk jumlah desimal) M dan jumlah desimal D. Presisi desimalnya bisa sampai 24 digit. Defaultnya *float* (10,2). Bilangan *float* selalu bisa bertanda
DOUBLE (M,D)	Bilangan pecahan dengan presisi dua kali lipat. Panjang (termasuk jumlah desimal) M danjumlah desimal D. presisi desimalnya bisa sampai 53 digit. Defaultnya *double* (16,4).

	Bilangan *float* selalu bisa bertanda. Sinonim dari *double* adalah *real*
DECIMAL (M,D)	Bilangan pecahan dan harus didefinisikan M dan D- nya. Setiap desimal membutuhkan tempat 1 *byte*. Sinonim dari *decimal* adalah *numeric*

2. Tipe Data String

Berikut ini adalah tipe data *string* yang paling umum di dalam MySQL

Tabel 2. 7 Tipe Data String

Tipe Data	Deskripsi
CHAR (M)	*String* dengan ukuran tetap. Ukurannya antara 1 sampai 255 karakter. Ukuran ditentukan dengan nilai M. Contoh: *CHAR* (6)
VARCHAR (M)	*String* dengan ukuran bervariasi antara 1 sampai dengan 255 karakter.Contoh: *VARCHAR* (25)
TEXT	*String* dengan ukuran maksimum 65535 karakter. *String* yang tersimpan di dalam *TEXT* dianggap tidak *case sensitive*. Untuk kapasitas yang lebih kecil bisa menggunakan *TINYTEXT* dengan kapasitas maksimal 255 karakter sedangkan untuk kapasitas lebih besar bisa menggunakan *MEDIUMTEXT* (maksimum 16777215 karakter) dan *LONGTEXT* (maksimal 4294967295 karakter
BLOB	*Binary Large Objects* (BLOB) adalah tipe data untuk menyimpan data *binary* dalam jumlah besar. Biasa digunakan untuk menyimpan citra. Untuk penyimpanan data yang lebih kecil bisa menggunakan *TINYBLOB* (maksimal 255 karakter) sedangkan untuk kapasitas yang lebih besar bisa menggunakan *MEDIUMBLOB* (maksimal 16777215 karakter) dan *LONGBLOB* (maksimal 4294967295 karakter)
ENUM	Enumerasi atau sebuah list (daftar). Jadi misalnya anda ingin bahwa sebuah nilai terbatas hanya boleh dengan nilai tertentu saja maka. Anda bisa membuat sebuah daftar, misalnya saja

	nilai itu hanya bisa terdiri dari A-E, maka anda bisa membuatnya menjadi ENUM (‘A’,’B’,’C’,’D’,’E’)

3. Tipe Data Tanggal dan Waktu

Berikut ini adalah tipe data tanggal dan waktu di dalam mysql.

Tabel 2. 8 Tipe Data Tanggal dan Waktu

Tipe Data	Deskripsi
DATE	Adalah tipe data tanggal dengan format YYYY-MM-DD, antara 1000-01-01dan 9999-12-31. Contoh 17 Agustus 1945 akan disimpan sebagai 1945-08-17
DATETIME	Adalah kombinasi tanggal dan waktu dengan format YYYY-MM-DD HH:MM:SS dan rentang data antara 1000- 01-01 00:00:00 sampai dengan 9999-12-31 23:59:59 Contoh : pukul 10:00 pagi pada tanggal 17 agustus 1945 akan disimpan sebagai 1945-08-17 10:00:00
TIMESTAMP	Sebuah penanda waktu antara 1 Januari 1970 tengah malam sampai dengan tahun 2037. Formatnya mirip dengan *DATETIME* tetapi tanpa pembatas di antara angkanya. Contoh pukul 10:00 pagi tanggal 17 Agustus 1945 akan disimpan sebagai 19450817100000
TIME	Menyimpan waktu dalam format HH:MM:SS. Contoh pukul 10:00 akan disimpan menjadi 10:00:00
YEAR (M)	Menyimpan data tahun dalam format 2 atau 4 digit. Jika M diisi dengan nilai 2, maka rentang tahunnya dari 1970-2069 sedangkan jika M diisi dengan nilai 4 maka *YEAR* bisa bernilai 1901 sampai dengan 2155. Default nilai M adalah 4.

2.7 PHP *Framework*

PHP atau yang memiliki kepanjangan *Hypertext PreProcessor*, merupakan suatu bahasa pemrograman yang difungsikan untuk membangun suatu *website* dinamis. PHP menyatu dalam kode HTML, maksudnya beda kondisi. HTML digunakan untuk membangun atau pondasi dari kerangka *layout web*, sedangkan

PHP difungsikan sebagai prosesnya, sehingga dengan adanya PHP, suatu *web* akan sangat mudah di-*maintenance*.

Hypertext Prepocessor (PHP) adalah sebuah bahasa pemrograman di sisi *server*. Ketika kita mengakses sebuah URL, maka peramban web akan melakukan *request* ke sebuah *web server*. Pengembangan bahasa PHP dimulai pada tahun 1994 oleh Rasmus Lerdorf. Dia mengembangkan perkakas yang digunakan sebagai *engine parsing* sebagai penerjemah *interpreter* beberapa *macro*. Pada saat itu engine digunakan untuk pembuatan buku tamu, *counter* dan beberapa *homepage*. Ia menamai *engine parser* tersebut dengan nama PHP/FI. (Magdalena & Rachman, 2017)

2.7.1 Codeigniter

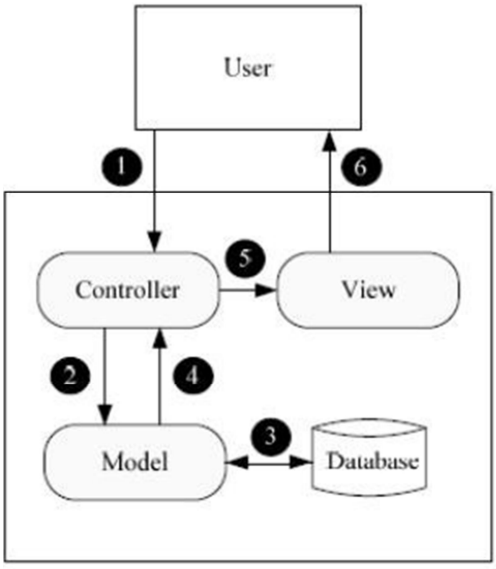
CodeIgniter adalah sebuah *web application framework* yang digunakan untuk membangun aplikasi PHP dinamis yang dibangun menggunakan konsep *Model View Controller. CodeIgniter* menyediakan berbagai macam *library* yang dapat mempermudah dalam pengembangan dan termasuk *framework* tercepat dibandingkan dengan *framework* lainnya. (Erinton & Sanjoyo, 2017)

2.7.2 Pengertian Konsep MVC Pada Web

Model-View-Controller (MVC) adalah pola arsitektur yang memisahkan aplikasi dalam tiga komponen utama Logis: model, *view* dan controller. Masing-masing komponen ini dibangun untuk menangani aspek-aspek tertentu pembangunan aplikasi. MVC adalah salah satu kerangka pembangunan *web* standar industri paling sering digunakan untuk menciptakan proyek yang terukur yang besar dan *extensible*. Istilah MVC ini semakin familiar seiring dengan perkembangan *framework* PHP seperti Yii, Codeigniter, Laravel dan lain - lain. Hampir seluruh *framework* PHP ini menggunakan konsep MVC.

Dengan menggunakan prinsip MVC suatu aplikasi dapat dikembangkan sesuai dengan kemampuan developernya, yaitu *progammer* yang menangani bagian model dan *controller*. Sedangkan desainer yang menangani bagian *view*, sehingga penggunaan arsitektur MVC dapat meningkatkan *maintanability* dan

organisasi kode. Walaupun demikian dibutuhkan komunikasi yang baik antara *programmer* dan desainer dalam.



Gambar 2. 1 Skema Model-View-Controller (MVC)

Pengertian MVC adalah sebuah bentuk pemrograman yang memisahkan berdasarkan logika penanganan tampilan, logika pengontrolan dan logika *Model*. MVC bertujuan supaya pada pengembangan perangkat lunak yang besar mudah untuk dilakukan maintenance. Bagian - bagian dari MVC adalah:

a. Model

   Model adalah bagian kode program yang menangani *query* atau *database*. Jadi isi dari model merupakan bagian (fungsi-fungsi) yang berhubungan langsung dengan *database* untuk memanipulasi data seperti memasukkan data, pembaruan data, hapus data, dan lain-lain, namun tidak dapat berhubungan langsung dengan bagian *view*.

b. View

   View adalah bagian kode program yang mengatur tampilan *website*. Pada aplikasi *web* bagian *View* biasanya berupa *file* template HTML, yang diatur oleh *controller*. Bagian ini tidak memiliki akses langsung terhadap bagian model namun berhubungan langsung dengan *controller*. *View* berfungsi

untuk menerima dan merepresentasikan data kepada pengguna. Jadi bisa di katakan bahwa *View* merupakan halaman *web.*

c. *Controller*

Controller merupakan bagian yang menjembatani *model* dan *view*. *Controller* berisi perintah-perintah yang berfungsi untuk memproses suatu data dan mengirimkannya ke halaman *web*. *Controller* berfungsi untuk menerima *request* dan data dari *user* kemudian menentukan apa yang akan diproses oleh aplikasi. (Pasaribu, 2017)

2.8 XAMPP

Xampp adalah suatu bundel web server yang populer digunakan untuk coba-coba di windows karena kemudahan instalisasinya. Xampp merupakan perangkat lunak bebas yang mendukung banyak sistem operasi, merupakan kompilasi dari beberapa program. Fungsinya adalah sebagai server yang berdiri sendiri (localhost) yang terdiri atas Apache HTTP Server. Mysql database dan penerjemahan bahasa yang ditulis dengan bahasa pemrograman PHP. Nama Xampp merupakan singkatan dari empat sistem operasi yaitu Apache, Mysql, PHP, dan Perl. Program ini tersedia dalam GNU dan bebas, merupakan web server yang mudah digunakan yang dapat melayani tampilan halaman web yang dinamis. (Sarwindah, 2018)

2.9 *Software Development Life Cycle* (SDLC)

2.9.1 Pengertian SDLC

SDLC atau *software development life cycle* atau sering 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 memiliki beberapa tahapan agar menghasilkan perangkat lunak yang berkualitas.

Tahapan-tahapan pada SDLC secara global adalah sebagai berikut :

- Inisiasi (*intiation*) : Tahap ini biasanya ditandai dengan pembuatan proposal proyek perangkat lunak

- Pengembangan konsep sistem (*system concept development*): mendefinisikan lingkup konsep termasuk dokumen lingkup sistem, analis manfaat biaya, manajemen rencana dan pembelajaran kemudahan sistem
- Perencanaan (*planning*) : mengembangakan rencana manajemen proyek dan dokumen perencanaan lainnya. Menyediakan dasar untuk mendapatkan sumber daya (*resource*) yang dibutuhkan untuk memperoleh solusi.
- Analisis kebutuhan (*reqruitment analysis*) : menganalisis kebutuhan pemaai sistem perangkat lunak (*user*) dan mengembangkan kebutuhan *user*. Membuat dokumen kebutuhan fungsional.
- Desain(*design*) : mentransformasikan kebutuhan detail menjadi kebutuhan yang sudah lengkap, dokumen desain sistem fokus pada bagaimana dapat memenuhi fungsi-fungsi yang dibutuhkan.
- Pengembangan (*development*) : mengonversi desain ke sistem informasi yang lengkap termasuk bagaimana memperoleh dan melakukan instalasi lingkungan sistem yang dibutuhkan; membuat basis data dan mempersiapkan berkas atau *file* pengujian, pengodean, pengompilasian, memperbaiki atau membersihkan program; peninjau pengujian.
- Integrasi dan pengujian (*intergration and test*) : mendemonstrasikan sistem perangkat lunak bahwa telah memenuhi kebutuhan yang dispesifikasikan pada dokumen kebutuhan fungsional. Dengan diarahkan oleh staf penjamin kualitas (*quality assurance*) dan *user*. Menghasilkan laporan analisis pengujian.
- Implementasi (*implementation*) : termasuk pada persiapan implementasi, implementasi perangkat lunak pada lingkungan produksi (lingkungan pada *user*) dan menjalankan resolusi dari permasalahan yang teridentifikasi dari fase integrasi dan pengujian.
- Operasi dan pemeliharaan (*operations and maintenance*) : mendeskripsikan pekerjaan untuk mengoperasikan dan memelihara sistem informasi pada lingkungan produksi (lingkungan pada *user*), termasuk impelentasi akhir dan masuk pada proses peninjauan
- Disposisi(*disposition*) : mendeskripsikan aktifitas akhir dari pengembangan sistem dan membangun data yang sebenarnya sesuai dengan aktifitas *user*.

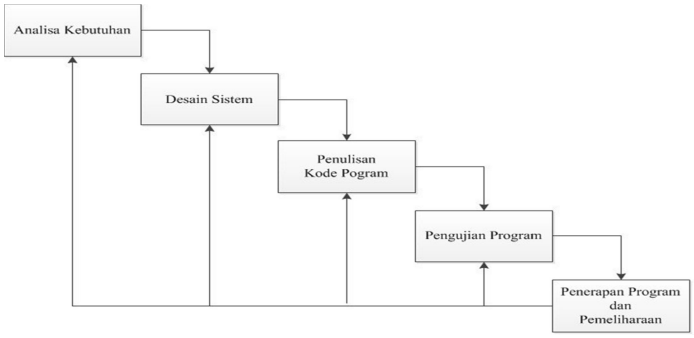
(A.S & Shalahudin, 2018)

2.9.2 *Model* SDLC (Software Development Life Cycle)

SDLC memiliki model dalam penerapan tahapan-tahapan prosesnya. beberapa model dasar akan dibahas, masih banyak model yang muncul dengan memodifikasi model-model SDLC dasar. (A.S & Shalahudin, 2018)

1. Model Waterfall

Adapun Metode Pengembangan Perangkat Lunak menggunakan metode pengembangan waterfall model. Dalam waterfall terdapat beberapa tahapan utama yang menggambarkan aktivitas pengembangan perangkat lunak. Alasan menggunakan metode waterfall karena tahap – tahap dalam pengembangan sistem pada model waterfall terstruktur secara jelas.



Gambar 2. 2 Model Waterfall

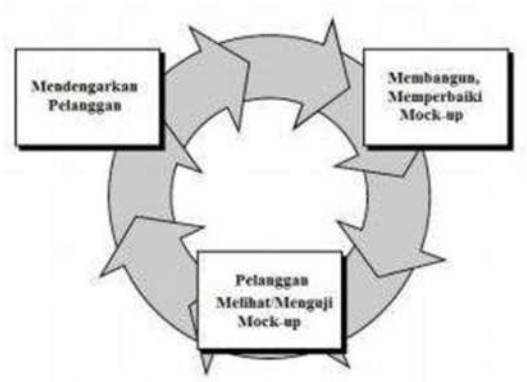
1) Analisis Kebutuhan Pada tahap ini dilakukan pengumpulan kebutuhan secara lengkap kemudian dianalisis dan didefinisikan kebutuhan yang harus dipenuhi oleh program yang akan dibangun. Dalam pengumpulan data kebutuhan dapat dilakukan dengan teknik wawancara, teknik observasi, dan teknik kuisoner.
2) Desain Sistem Proses desain adalah proses multi langkah yang berfokus pada empat atibut yaitu : struktur data, arsitektur perangkat lunak, representasi interface, dan detail prosedural Desain dikerjakan setelah kebutuhan selesai dikumpulkan secara lengkap.
3) Implementasi Pembuatan program atau hasil rancangan ke dalam bahasa pemograman tertentu. Penuliasan kode program sesuai dengan desain yang sudah ditentukan, sehingga menghasilkan aplikasi yang bermanfaat.

4) Pengujian Sistem Melakukan pengujian sistem dengan cara agar sistem valid dan dapat digunakan dengan baik.
5) Pemeliharaan Mengaplikasikan sistem yang sudah terintegrasi dan melakukan perawatan atau perbaikan kalau ada kekeliruan. (A.S & Shalahudin, 2018)

2. Model Prototype

Model *prototype* dapat digunakan untuk menyambungkan ketidakpahaman *user* mengenai hal teknis dan memperjelas spesifikasi kebutuhan yang diinginkan *user* kepada pengembang perangkat lunak.

Model *prototype* dimulai dari mengumpulkan kebutuhan pelanggan terhadap perangkat lunak yang akan dibuat. Lalu dibuatlah program *protoype* agar *user* lebih terbayang dengan apa yang sebenarnya diinginkan. Program *prototype* biasanya merupakan program yang belum jadi. Program ini biasanya menyediakan tampilan dengan simulasi alur perangkat lunak sehingga tampak seperti perangkat lunak yang sudah jadi. Program prototipe ini dievaluasi oleh *user* sampai di temukan spesifikasi yang sesuai dengan keinginan *user*. Berikut adalah gambar dari model *prototype* .



Gambar 2. 3 Model Prototipe

Model prototipe juga memiliki kelemahan sebagai berikut :

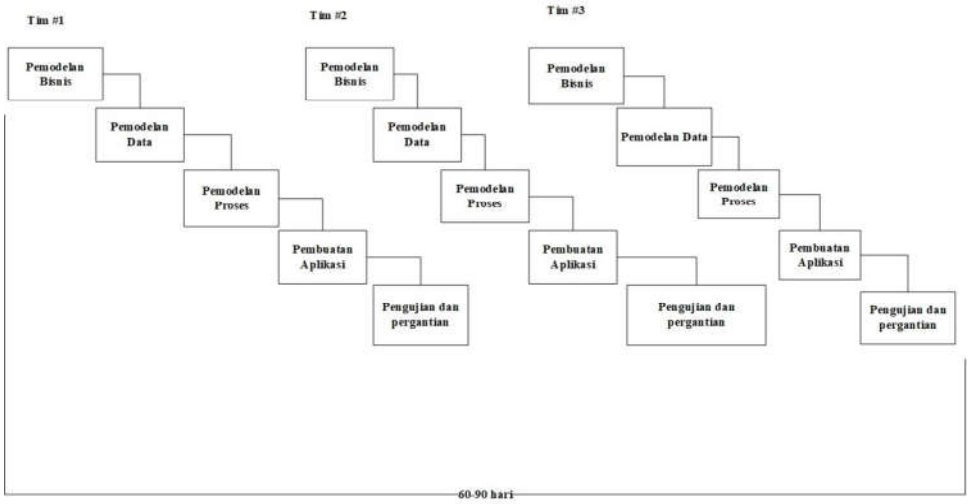
- *User* dapat sering mengubah-ubah atau menambahkan spesifikasi kebutuhan karena menganggap aplikasi sudah dengan cepat dikembangkan.
- Pengembang lebih sering mengambil kompromi denga *user* untuk mendapatkan *prototype* dengan waktu yang cepat sehingga pengembang

lebih sering melakukan segala cara guna menghasilkan protipe untuk didemonstrasikan. (A.S & Shalahudin, 2018)

3. Model Rapid Application Development (RAD)

Rapid Application Deevelopment (RAD) adalah model proses pengembang perangkat lunak yang bersifat inkremental terutama untuk waktu pengerjaan yang pendek. Model RAD adalah adaptasi dari model air terjun versi kecepatan tinggi dengan menggunakan model air terjun untuk pengembang setiap komponen perangkat lunak.

Model RAD membagi tim pengembang menjadi beberapa tim untuk mengerjakan beberapa komponen masing-masing tim pengerjaan dapat dilakukan secara paralel. Berikut adalah gambar dari model RAD.



Gambar 2. 4 Model RAD

- Pemodelan bisnis

  Pemodelan yang dilakukan untuk memodelkan fungsi bisnis untuk mengetahui informasi apa yang terkait proses bisnis, informasi apa saja yang harus dibuat, siapa yang membuat informasi itu, bagaimana alur informasi itu, proses apa saja yang terkait dengan informasi itu.

- Pemodelan data

  Memodelkan data apa saja yang dibutuhkan berdasarkan pemodelan bisnis dan mendefinisikan atribut-atribut nya beserta relasinya dengan data yang lain

- Pemodelan proses

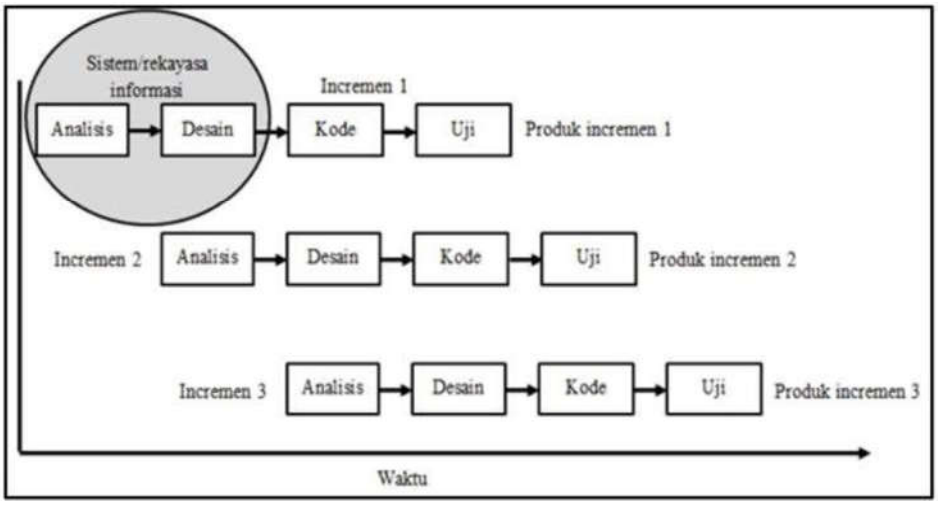
  Mengimplementasikan fungsi bisnis yang sudah didefinisikan terkait dengan pendefinisian data
- Pembuatan aplikasi

  Mengimplementasikan pemodel proses dan data menjadi program. Model RAD sangat menganjurkan pemakaian komponen yang sudah ada jika dimungkinkan.
- Pengujian dan pergantian

  Menguji komponen-komponen yang dibuat. Jika sudah teruji maka tim pengembang komponen dapat beranjak untuk mengembangkan komponen berikutnya. (A.S & Shalahudin, 2018)

4. Model Iteratif

Model iteratif (*iterative models*) mengkombinasikan proses-proses pada model air terjun dan iteratif pada model *prototype*. Model inkremental akan menghasilkan versi-versi perangkat lunak yang sudah mengalami penambahan fungsi untuk setiap pertambahannya. Mode*l* iteratif merumpakan gabungan dari model *waterfall* dan model prototipe. Model ini cocok digunakan pengembang dengan *turnover* staf tinggi. Berikut adalah gambar dari model iteratif :



Gambar 2. 5 Model Iteratif

Model inkremental dibuat untuk mengatasi kelemahan dari model air terjun yang tidak mengakomodasi iterasi dan mengatasi kelemahan dari model

prototipe yang memiliki proses terlalu pendek dan setiap iteratif prosesnya tidak selalu menghasilkan produk. Model inkremental menghasilkan produk/aplikasi untuk tahapan inkremen. (A.S & Shalahudin, 2018)

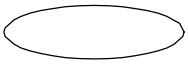
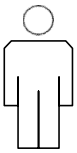
2.10 *Unfied Modelling Language* (UML)

UML (Unified Modeling Language) adalah pendekatan terstruktur memiliki *tool-tool* perancangan yang di kenal secara luas serta menjadi standar umum, seperti *DFD (Data Flow Diagram)*, *ERD (Entity Relationship Diagram)*, bagan terstruktur *(structure chart), diagram* alir *flow chart*. *Unified modeling language* adalah satu kumpulan *diagram*, yang dirancang secara khusus untuk pengembangan berorientasi objek, dan telah menjadi standar industri untuk pemodelan sistem berorientasi objek. (M.Shalahuddin & Sukamto, 2015)

2.10.1 Use Case Diagram

Use case diagram adalah deskripsi fungsi dari sebuah system dari prespektif pengguna. Use case bekerja dengan cara mendeskripsikan tipikal interaksi antara user (pengguna) sebuah sistem dengan sistemnya sendiri melalui sebuah cerita bagaiamana sebuah sistem dipakai.Berikut adalah simbol-simbol dalam *use case diagram.*

Tabel 2. 9 Simbol-simbol dalam use case diagram

Simbol	Keterangan
 Use Case	*Use case* merupakan fungsionalitas yang disediakan sistem sebagai unit-unit yang saling bertukar pesan antara unit atau aktor.
 Actor	Aktor merupakan orang, proses, atau sistem lain yang berinteraksi dengan sistem informasi yang akan dibuat diluar sistem informasi yang akan dibuat itu sendiri. Walaupun simbol dari aktor adalah gambar orang, tapi aktor belum tentu merupakan orang.
	*System* merupakan menspesifikasikan paket yang menampilkan sistem secara terbatas.

<<include>>	*Include* diartikan sebagai relasi *use case* tambahan ke sebuah *use case* di mana *use case* yang ditambahkan memerlukan *use case* ini untuk menjalankan fungsinya atau sebagai syarat dijalankan *use case* ini.
Asosiasi	Asosiasi adalah komunikasi antara aktor dan *use case* yang berpartisipasi pada *use case* atau *use case* memiliki interaksi dengan aktor.

(Munawar, 2018)

2.10.2 Activity Diagram

Activity diagram atau *diagram* aktivitas menggambarkan aspek dinamis dari sistem. Logika procedural, proses bisnis dan aliran kerja suatu bisnis bisa dengan mudah dideskripsikan dalam *activity diagram*. Simbol-simbol dalam *activity diagram* adalah sebagai berikut:

Tabel 2. 10 Simbol-simbol dalam *Activity diagram*

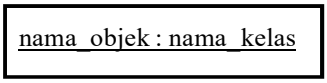
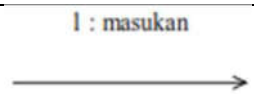
Simbol	Keterangan
Start	*Start*/State Awal Menunjukan dimana aliran kerja dimulai
Activity	Aktivitas menunjukan langkah-langkah dalam sebuah *activity diagram*. Aktivitas bisa terjadi saat memasuki *activity diagram*, meninggalkan *activity diagram*, atau pada event yang spesifik.
Discuss	Percabangan menunjukan suatu keputusan yang mempunyai satu atau lebih transisi dan dua atau lebih transisi sesuai dengan suatu kondisi.
End	*END*/State Akhir menunjukan dimana aliran kerja diakhiri

(Munawar, 2018)

2.10.3 Sequence Diagram

*Sequence diagram*digunakan untuk menggambarkan perilaku pada sebuah scenario. Diagram ini menunjukkan sejumlah contoh obyek dan massage (pesan) yang diletakkan diantara obyek-obyek ini didalam use case. Beberapa simbol dalam *sequence diagram* :

Tabel 2. 11 Simbol-simbol dalam *Sequence Diagram*

Simbol	Keterangan
	Orang, proses atau sistem lain yang berinteraksi dengan sistem informasi dan mendapat manfaat dari *system*
	Garis hidup atau *lifeline* menyatakan kehidupan suatu objek
	Objek menyatakan objek yang berinteraksi pesan
	Pesan tipe *send* menyatakan bahwa suatu objek mengirimkan data/masukan/informasi ke objek lainnya, arah panah mengarah pada objek yang dikirim
	Pesan tipe *return* menyatakan bahwa suatu objek yang telah menjalankan suatu operasi atau metode menghasilkan suatu kembalian ke objek tertentu
	Waktu aktif menyatakan objek dalam keadaan aktif dan berinteraksi
	Pesan tipe *call* menyatakan suatu objek memanggil operasi/metode yang ada pada objek lain ataudirinya sendiri

	Pesan tipe *create* enyatakan suatu objek membuat objek yang lain, arah panah mengarahkan pada objek yang dibuat.

(Munawar, 2018)

2.10.4 Class Diagram

Diagram kelas (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.

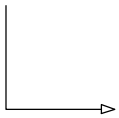
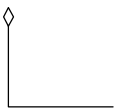
Class Diagram juga menunjukkan atribut-atribut dan operasi-operasi dari sebuah kelas dan *constraint* yang berhubungan dengan objek yang dikoneksikan.

Diagram secara khas meliputi: kelas (*Class*), Relasi *Associations*, *Generalitation* dan *Aggregation*, atribut (*Attributes*), operasi (*operation/method*), dan *visibility*, tingkat akses objek eksternal kepada suatu operasi atau atribut. Hubungan antar kelas mempunyai keterangan yang disebut dengan *Multiplicity* atau *Cardinality*.

Class merupakan implementasi dari sebuah *interface*, yaitu *class* abstrak yang hanya memiliki metoda. *Interface* tidak langsung dinotasikan tetapi harus diimplementasikan dahulu menjadi sebuah *class*. Dengan demikian *interface* mendukung resolusi metoda pada saat *run-time*.

Tabel 2. 12 Simbol Class Diagram

Simbol	Keterangan
	*Class* menggambarkan kelas pada struktur sistem.
	*Association* menggambarkan relasi antar kelas dengan makna umum, asosiasi biasanya disertai dengan *multiplicity*.
	*Directed Association* menggambarka relasi antar kelas dengan makna kelas yang satu digunakan oleh kelas lain, asosiasi biasanya disertai dengan

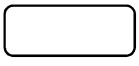
	multiplicity.
	*Generalization* menggambarkan relasi antar kelas dengan makna generalisasi spesialisasi (umum-khusus).
	*Dependency* menggambarkan kebergantungan antar kelas.
	*Aggregation* menggambarkan relasi antar kelas dengan makna semua bagian (*whole part*).

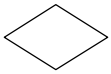
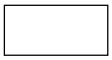
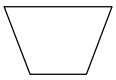
(Munawar, 2018)

2.10.5 Flowmap

Flowmap adalah campuran peta dan flow chart,yang menunjukan pergerakan benda dari satu lokasi ke lokasi lain, seperti jumlah orang dalam migrasi, jumlah barang yang diperdagangkan, atau jumlah paket dalam jaringan. Flowmap menolong analisis dan programmer untuk memecahkan masalah ke dalam segmen-segmen yang lebih kecil dan menolong dalam menganalisis alternaitf-alternatif lain dalam pengoprasian.

Tabel 2. 13 Simbol-simbol dalam Flowmap

Simbol	Keterangan
 Flow line	Flow atau garis aliran merupakan symbol yang digunakan untuk menghubungkan simbol satu dengan yang lain.
 Terminator	Terminator menunjukan awal dan akhir dari program.
 Document	*Document* merupakan simbol yang menyatakan input berasal dari dokumen dalam bentuk kertas atau output dicetak dikertas.

 Database	*Database* merupakan media penyimpanan dari proses entry data dan proses komputasi.
 *Decision*	*Decision* digunakan untuk mewakili operasi perbandingan logika.
 Proses	Proses digunakan untuk pengolahan aritmatika dan pemindahan data.
 Data	*Input/Output* merupakan proses masukan dan keluaran tanpa tergantung jenis peralatannya.
 *Manual Operation*	*Manual operation* merupakan simbol yang menunjukan proses yang dilakukan secara manual.
 *Disk Storage*	*Disk storage* menyatakan input berasal dari disk atau output disimpan ke disk.
 *Connector*	*Connector*, digunakan untuk menunjukkan hubungan arus proses yang terputus masih dalam halaman yang sama.
 *Off page*	*Off page* atau penguhubung halaman lain, digunakan untuk menunjukkan hubungan proses yang terputus masih dalam halaman yang sama.

(Hamidin & Maniah, 2017)

2.11 Pengujian Sistem

2.11.1 Pengujian Sistem *Whitebox*

White Box Testing adalah salah satu cara untuk menguji suatu aplikasi atau *software* dengan cara melihat modul untuk dapat meneliti dan menganalisa kode dari program yang di buat ada yang salah atau tidak. Kalau modul yang telah dan sudah di hasilkan berupa output yang tidak sesuai dengan yang di harapkan maka

akan dikompilasi ulang dan di cek kembali kode-kode tersebut hingga sesuai dengan yang diharapkan.

Kasus yang sering menggunakan *white box testing* akan di uji dengan beberapa tahapan yaitu:

a. Pengujian seluruh keputusan yang menggunakan logikal.
b. Pengujian keseluruh loop yang ada sesuai batasan-batasannya.
c. Pengujian pada struktur data yang sifatnya internal dan yang terjamin validitasnya.

Kelebihan *White Box Test*ing antara lain :

1. Kesalahan Logika

   Menggunakan sintax '*if*' dan sintax pengulangan. Langkah selanjutnya metode *white box testing* ini akan mencari dan mendeteksi segala kondisi yang di percaya tidak sesuai dan mencari kapan suatu proses perulangan di akhiri.

2. Ketidaksesusaian Asumsi

   Menampilkan dan memonitor beberapa asumsi yang diyakini tidak sesuai dengan yang diharapkan atau yang akan diwujudkan, untuk selanjutnya akan dianalisa kembali dan kemudian diperbaiki.

3. Kesalah Pengetikan

   Mendeteksi dan mencaribahasa-bahasa pemograman yang di anggap bersifat *case sensitive.* Kelemahan *white box testing* adalah pada perangkat lunak yang jenisnya besar, metode *white box testing* ini dianggap boros karena melibatkan banyak sumberdaya untuk melakukannya. (M.Sidi & dkk, 2015)

2.11.2 Pengujian Sistem *Blackbox*

Pengujian *black box* bertujuan untuk menemukan kesalahan fungsi pada program. Pengujian dilakukan dengan cara memasukan input tertentu dan melihat hasil yang dapat dari input tersebut. Pengujian *black box*, yang diuji adalah masukan serta keluarannya.

Metode uji coba *black box* memfokuskan pada keperluan fungsional dari *software*. Karna uji coba *black box* memungkinkan pengembangan *software*

untuk membuat himpunan kondisi input yang akan melatih seluruh syarat-syarat fungsional suatu program. Uji coba *black box* bukan merupakan alternatif dari uji coba *white box*, tetapi merupakan pendekatan yang melengkapi untuk menemukan kesalahan lainnya, selain menggunakan metode *white box.*

Black Box Testing berfokus pada spesifikasi fungsional dari perangkat lunak. *Tester* dapat mendefinisikan kumpulan kondisi input dan melakukan pengetesan pada spesifikasi fungsional program. *Black Box Testing* bukanlah solusi alternatif. *White Box Testing* tapi lebih merupakan pelengkap untuk menguji hal-hal yang tidak dicakup oleh *White Box Testing.* Black Box Testing cenderung untuk menemukan hal-hal berikut:

1. Fungsi yang tidak benar atau tidak ada.
2. Kesalahan antarmuka *(interface errors).*
3. Kesalahan pada struktur data dan akses basis data.
4. Kesalahan performansi *(performance errors).*
5. Kesalahan inisialisasi dan terminasi.

(Mustaqbal & H.Rahmadi, 2015)