BAB II
LANDASAN TEORI

2.1 Konsep Sistem Informasi

Konsep sistem informasi membahas tentang pengertian dasar sistem informasi, komponen sistem informasi dan sumber daya sistem informasi. Penjelasannya adalah sebagai berikut :

2.1.1 Sistem Informasi

Pengertian dasar dari sistem adalah suatu prosedur-prosedur yang saling berhubungan, dan disusun sesuai dengan skema yang menyeluruh untuk melaksanakan kegiatan atau fungsi dari suatu lembaga yang dihasilkan suatu proses tertentu untuk meyediakan informasi yang layak. Maka dapat disimpulkan sistem adalah suatu kumpulan dari bagian-bagian ataupun jaringan yang saling berhubungan satu dengan yang lainnya dan bekerja sama untuk mencapai satu tujuan. (Tamba, 2017)

Sedangkan informasi dapat didefinisikan sebagai data yang diolah menjadi bentuk yang lebih berguna dan lebih berarti dari yang menerimanya. Sumber dari informasi ialah data. Informasi dapat diperoleh dari processing system. (Fahlapi, 2017)

Kesimpulannya, 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. Adapun pengertian lain sistem informasi adalah suatu sistem di dalam suatu organisasi yang mempertemukan kebutuhan pengolahan data transaksi harian, mendukung operasi, bersifat manajerial dan kegiatan strategi dari suatu organisasi serta menyediakan pihak luar tertentu dengan laporan-laporan yang diperlukan. (Firman, et al., 2016)

Berdasarkan itu sistem informasi adalah suatu sistem yang terdiri dari beberapa komponen, yaitu *software*, *hardware* dan *brainware* yang memproses informasi menjadi sebuah output yang berguna untuk mencapai tujuan tertentu

dan berguna dalam suatu organisasi. Sistem informasi terdiri dari lima sumber daya yang di kenal sebagai komponen sistem informasi, kelima sumber daya tersebut adalah :

1. Manusia

   Manusia mempunyai peranan penting bagi sistem informasi untuk mengoperasikan sistem informasi, dan juga sebagai penggunaan akhir dan pakar sistem informasi.

2. *Hardware*

   Semua peralatan yang digunakan dalam pemrosesan informasi, yang terdiri dari komputer dan media data lainnya.

3. *Software*

   Merupakan semua rangkaian perintah (instruksi) yang digunakan untuk memproses informasi, berupa program dan seluruh prosedur.

4. Data

   Merupakan bahan baku sebagai dasar membentuk sistem informasi dan sebagai dasar sumber daya organisasi.

5. Jaringan

   Merupakan media komunikasi yang menghubungkan komputer, memproses komunikasi dan peralatan lainnya, yang dikendalikan melalui *Software* komunikasi.

Kelima komponen ini memainkan peranan yang sangat penting dalam sistem informasi. Namun, dalam kenyataannya tidak semua sistem informasi mencakup lima komponen tersebut misalnya, sistem informasi pribadi yang mencakup jaringan telekomunikasi. (Tamba, 2017)

2.1.2 Komponen Sistem Informasi

Untuk mendukung lancarnya suatu sistem informasi dibutuhkan beberapa komponen yang fungsinya sangat penting di dalam sistem informasi. Komponen-komponen sistem informasi dapat dijelaskan sebagai berikut :

a. Input

Input disini adalah semua data yang dimasukan ke dalam sistem informasi. Dalam hal ini yang termasuk dalam input adalah dokumen-dokumen, formulir-formulir dan file-file. Dokumen-dokumen tersebut dikumpulkan dan dikonfirmasikan ke suatu bentuk sehingga dapat diterima oleh pengolah yang meliputi:

1) Pencatatan
2) Penyimpanan
3) Pengujian
4) Pengkodean

b. Proses

Proses merupakan kumpulan prosedur yang akan memanipulasi input yang kemudian akan disimpan dalam bagian basis data dan akan diolah menjadi suatu output yang akan digunakan oleh si penerima.

c. Output

Output merupakan semua keluaran atau hasil dari model yang sudah diolah menjadi s12uatu informasi yang berguna dan dapat dipakai penerima. Komponen ini akan berhubungan langsung dengan pemakai sistem informasi dan merupakan tujuan akhir dari pembuatan sistem informasi.

d. Teknologi

Teknologi disini merupakan bagian yang berfungsi untuk memasukan input, mengolah input dan menghasilkan keluaran. Ada 3 bagian dalam teknologi ini yang meliputi perangkat keras, perangkat lunak dan perangkat manusia. Perangkat keras contohnya: *keyboard, mouse* dan lain-lain. perangkat lunak contohnya: program untuk mengolah data dan perangkat manusia contohnya: analisis sistem, programmer, teknisi dan sebagainya.

e. Basis Data

Basis data merupakan kumpulan data-data yang saling berhubungan satu dengan yang lain yang disimpan dalam perangkat keras komputer dan akan diolah menggunakan perangkat lunak.

2.1.3 Sumber Daya Sistem Informasi

Teknologi informasi merupakan perkembangan dari teknologi komputer yang dipadukan dengan teknologi telekomunikasi. Definisi ‘informasi’ itu sendiri adalah ‘hasil’ dari pengolahan data yang secara prinsip mempunyai nilai lebih daripada data mentah.

Teknologi informasi dapat dikatakan sebuah suatu teknologi yang berhubungan dengan pengolahan data menjadi informasi dan proses penyaluran informasi tersebut dalam batas-batas ruang dan waktu.

Sistem informasi merupakan suatu kumpulan dari komponen-komponen dalam suatu organisasi yang berhubungan dengan proses penciptaan dan aliran informasi. Dalam hal ini, teknologi informasi hanya merupakan salah satu komponen kecil saja. Komponen lainnya secara umum adalah proses dan prosedur, struktur organisasi, SDM, model-model untuk analisis, perencanaan, pengendalian dan pembuatan keputusan serta database.

Secara umum sistem informasi merupakan kombinasi dari orang (*people*), perangkat keras (*hardware*), perangkat lunak (*software*), jaringan komunikasi (*communication network*) dan sumber daya yang dihimpun, ditransformasi dan mengalami proses pengaliran dalam suatu organisasi (Andri, 2018).

2.2 Pengolahan Data

Menurut Ladjamudin (2013:9), Pengolahan data adalah masa atau waktu yang digunakan untuk mendeskripsikan perubahan bentuk data menjadi formasi yang memiliki kegunaan.

Menurut Sutarman (2012:4), Pengolahan data adalah proses perhitunganatau transformasi data input menjadi informasi yang mudah dimengerti ataupun sesuai dengan yang diinginkan.

Dari pengertian diatas dapat disimpulkan Pengolahan Data adalah data yang diolah menjadi bentuk yang lebih berarti, dimengerti dan berguna yang berupa informasi.

2.3 Sumbangan Penunjang Pendidikan (SPP)

SPP (Sumbangan Penunjang Pendidikan) adalah iuran atau pembayaran setiap bulan dari siswa yang menjadi kewajiban bagi siswa di sekolah. Pembayaran SPP tersebut diambil berdasarkan kesepakatan rapat Komite sekolah dan orang tua siswa. Pembayaran SPP ditunjukan untuk menunjang peningkatan mutu pendidikan yang terkait dengan sarana dan prasarana kegiatan belajar mengajar (D. Melia, 2012).

2.4 Website

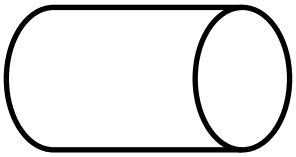
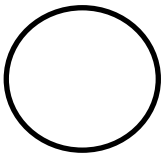
Menurut Medi Suhartanto di dalam (Arief, 2011) *Website* adalah salah satu aplikasi yang berisikan dokumen-dokumen multimedia (teks, gambar, suara, animasi, video) didalamnya yang menggunakan protokol HTTP (*hypertext transfer protocol*) dan untuk mengaksesnya menggunakan perangkat lunak yang disebut *browser*. Beberapa jenis *browser* yang populer saat ini diantaranya: *Internet Explorer* yang diproduksi oleh *Microsoft*, *Mozilla Firefox*, Opera dan Safari yang diproduksi oleh *Apple*, *Browse*r (perambah) adalah aplikasi yang mampu menjalankan dokumen-dokumen *web* dengan cara diterjemahkan.

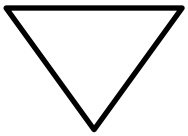
2.5 *Flowmap*

Berdasarkan (Suprianto & Matsea, 2018) *Flowmap* adalah paket perangkat lunak yang: Sebuah lokasi tempat asal aliran dimulai dan lokasi tujuan dimana aliran berakhir. Aliran data itu sendiri misalnya komputer, berbelanja, pengunjung rumah sakit, barang, penggunaan layanan pertanian atau telekomunikasi dan sebagainya.

Flowmap merupakan campuran peta dan *Flowchart*, yang menunjukkan pergerakan benda dari suatu lokasi ke lokasi yang lain. *Flowmap* dapat digunakan untuk menunjukkan gerakan yang hampir segala sesuatu. *Flowmap* dapat menunjukkan hal-hal berupa data yang mengalir, dapat menunjukkan arah aliran data bergerak dan apa saja sumber dan tujuan tersebut, dapat menunjukkan data yang mengalir, yang ditransfer dan dapat memberikan informasi umum yang mengalir dan proses data mengalir. Berikut simbol-simbol dalam *flowmap* ditunjukkan pada tabel berikut:

Tabel 2. 1 *Flowmap*

Simbol	Deskripsi
	***Terminator,*** untuk menyatakan permulaan atau akhir program.
	**Garis Alir,** menunjukkan arah aliran program.
	**Proses Komputerisasi,** menunjukan proses penghitung atau proses pengolahan data.
	**Dokumen,** menunjukan proses *input* atau *output* berupa dokumen.
	***Manual Storage***, berfungsi sebagai tempat penyimpanan manual.
	***Storage Data,*** menunjukan akses langsung perangkat penyimpanan data ke dalam *Harddisk*.
	***Manual Operation,*** menunjukan proses yang dilakukan secara manual.
	***Input Manual,*** menunjukan proses *input* menggunakan *keyboard*.
	***Konektor***, menunjukkan penghubung ke halaman yang masih sama atau ke halaman lain.

	**Arsip,** Menunjukkan pengarispan *file* tanpa menggunakan komputer.

2.6 Metode Pengembangan Sistem (*System Development Life Cycle*)

System Development Life Cycle adalah proses mengembangkan atau mengubah suatu sistem perangkat lunak dengan menggunakan model - model dan metodologi yang digunakan orang untuk mengembangkan sistem-sistem perangkat lunak sebelumnya.Model Pengembangan Sistem Informasi:

1. *Model sekuensial linier (clasic life cycle/waterfall model)*

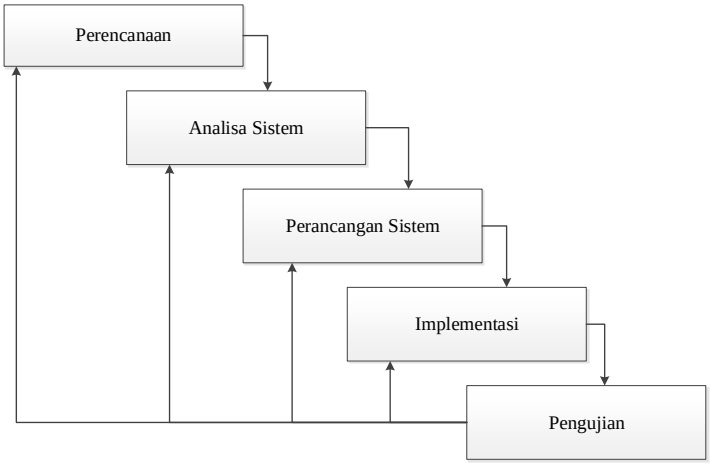
   Terdiri dari tahapan perencanaan sistem (rekayasa sistem), analisa kebutuhan, desain, penulisan program, pengujian dan perawatan sistem.
2. *Model prototype (prototyping model)*

   Dimulai dengan pengumpulan kebutuhan dan perbaikan, desain cepat,pembentukan prototipe, evaluasi pelanggan terhadap prototipe, perbaikan prototipe dan produk akhir.
3. *Rapid Application Development (RAD)*

   Model Dengan kegiatan dimulai pemodelan bisnis, pemodelan data, pemodelan proses, pembangkitan aplikasi dan pengujian (Sagita & Sugiarto, 2016)

2.6.1 Metode Waterfall

Metode Pengembangan sistem yang digunakan adalah metode *Waterfall*. Metode *Waterfall*, metode ini merupakan metode pengembangan sistem yang setiap tahapan pengembangannya dilakukan secara berurutan. Yaitu :



Gambar 2. 1 *Metode Waterfall*

1. Tahap analisis yaitu proses menganalisa data masukan penelitian yang telah dikumpulakan seperti proses binis dan permasalahan yang terjadi. Analisa ini ditujukan untuk mengidentifikasi masalah, menentukan kebutuhan sistem dan lainnya sehingga dapat merumuskan solusi atas permasalahan yang terjadi.
2. Setelah proses analisis kebutuhan sistem ini selesai tahap selanjutnya adalah perancangan sistem yaitu pembuatan model sistem atau design. Pemodelan sistem yang dilakukan yaitu pemodelan proses, pemodelan data, dan pemodelan antarmuka.
3. Tahap ketiga dari model *Waterfall* adalah pengkodean yaitu tahap menerjamahan perancangan sistem kedalam bahasa pemrograman.
4. Pada tahap terakhir yaitu pengujian akan dilakukan secara menyeluruh, dibuatkan terlebih dahulu skenario testing.
5. Apabila ada kesalahan yang terjadi maka dilakukan proses pemeliharaan guna memperbaiki kesalahan. Perbaikan implementasi unit sistem akan dianggap sebagai kebutuhan baru.

Kelebihan dan kekurangan metode *Waterfall* yaitu sifatnya yang kaku terhadap perubahan. Dokumen pengembangan sistem terorganisir, karena setiap tahapan harus terselesaikan dengan lengkap sebelum melangkah ke tahap berikutnya. Jadi setiap tahapan akan mempunyai dokumen tertentu. (Nuryani, et al., 2015)

2.6.2 Metode *Prototype*

Prototype bagi pengembang sistem bertujuan untuk mengumpulkan informasi dari pengguna sehingga pengguna dapat berinteraksi dengan model *prototype* yang dikembangkan, sebab *prototype* menggambarkan versi awal dari sistem untuk kelanjutan sistem sesungguhnya yang lebih besar. *Prototype* dapat diterapkan pada pengembangan sistem kecil maupun besar dengan harapan agar proses pengembangan dapat berjalan dengan baik, tertata serta dapat selesai tepat waktu.

Ketika *protype* terbentuk akan menguntungkan seluruh pihak yang terlibat, bagi pimpinan, pengguna sendiri serta pengembang sistem. Manfaat lainnya dari penggunaan *prototype* adalah :

1. Mewujudkan sistem sesungguhnya dalam sebuah replika sistem yang akan berjalan, menampung masukan dari pengguna untuk kesempurnaan sistem.
2. Pengguna akan lebih siap menerima setiap perubahan sistem yang berkembang sesuai dengan berjalannya *prototype* sampai dengan hasil akhir pengembangan yang akan berjalan nantinya.
3. *Prototype* dapat ditambah maupun dikurangi sesuai berjalannya proses pengembangan. Kemajuan tahap demi tahap dapat diikuti langsung oleh pengguna.
4. Penghematan sumberdaya dan waktu dalam menghasilkan produk yang lebih baik dan tepat guna bagi pengguna.

*Prototypp*e dimulai dengan pengumpulan kebutuhan, melibatkan pengembang dan pengguna sistem untuk menentukan tujuan, fungsi dan kebutuhan operasional sistem. Langkah-langkah dalam *prototyping* adalah sebagai berikut :

1. Pengumpulan Kebutuhan.
2. Proses desain yang cepat.
3. Membangun prototipe.
4. Evaluasi dan perbaikan

Mengumpulkan kebutuhan melibatkan pertemuan antara pengembang dan pelanggan untuk menentukan keseluruhan tujuan dibuatnya perangkat lunak; mengidentifikasi kebutuhan berupa garis besar kebutuhan dasar dari sistem yang akan dibuat.

Desain berfokus pada representasi dari aspek perangkat lunak dari sudut pengguna; ini mencakup *input*, proses dan format *output*. Desain cepat mengarah ke pembangunan prototipe, *prototipe* dievaluasi oleh pengguna dan bagian analis desain dan digunakan untuk menyesuaikan kebutuhan perangkat lunak yang akan dikembangkan. *Prototype* diatur untuk memenuhi kebutuhan pengguna, dan pada saat itu pula pengembang memahami secara lebih jelas dan detil apa yang perlu dilakukannya. Setelah keempat langkah *prototyping* dijalankan, maka langkah selanjutnya adalah pembuatan atau perancangan produk yang sesungguhnya. (Purnomo, 2017)

2.6.3 Metode Rapid Application Development

Rapid Application Development (RAD). RAD merupakan model proses perangkat lunak yang menekankan pada daur pengembangan hidup yang singkat. RAD merupakan versi adaptasi cepat dari model waterfall, dengan menggunakan pendekatan konstruksi komponen. RAD merupakan gabungan dari bermacam-macam teknik terstruktur dengan teknik *prototyping* dan teknik pengembangan *joint application* untuk mempercepat pengembangan sistem/aplikasi. Dari definisi konsep RAD ini, dapat dilihat bahwa pengembangan aplikasi dengan menggunakan metode RAD dapat dilakukan dalam waktu yang relatif lebih cepat. Sesuai dengan metodologi RAD berikut ini adalah tahap-tahap pengembangan aplikasi dari tiap fase pengembangan aplikasi dapat dilihat pada gambar berikut :

Gambar 2. 2 *Workshop Desain RAD*

(Putri & Effendi, 2018)

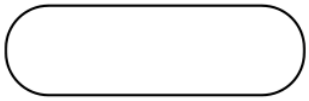
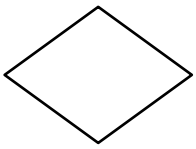
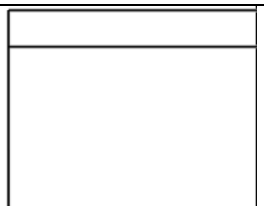
2.7 *Unified Modelling Language* (UML)

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)

2.7.1 *Activity Diagram*

Activity Diagram atau diagram aktivitas menggambarkan *workflow* (aliran kerja) atau aktivitas dari sebuah sistem atau proses bisnis atau menu yang ada pada perangkat lunak. Dimana diagram aktivitas menggambarkan aktivitas sistem bukan apa yang dilakukan aktor, jadi aktivitas yang dapat dilakukan oleh sistem. Berikut adalah simbol-simbol yang ada pada *Activity Diagram* menurut (Rosa dan Shalahuddin, 2015)

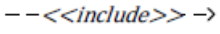
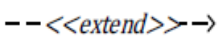
Tabel 2. 2 Simbol *Activity Diagram*

Simbol	Deskripsi
	***Start point***, menyatakan status awal aktivitas sistem.
	***End point***, menyatakan status akhir yang dilakukan sistem.
	***Action,*** menyatakan aktivitas yang dilakukan sistem, biasanya diawali dengan kata kerja.
	***Decision***, menggambar kan pilihan untuk pengambilan keputusan, jika terdapat pilihan lebih dari satu.
	***Join*** **/ Penggabungan,** menyatakan asosiasi penggabungan dimana lebih dari satu aktivitas digabungkan menjadi satu.
	***Swimlane***, pembagian *activity diagram* untuk menunjukkan siapa yang bertanggung jawab terhadap aktivitas yang terjadi.

2.7.2 *Use Case Diagram*

Use case diagram merupakan pemodelan untuk kelakukan (*behavior*) sistem informasi yang akan dibuat. *Use case diagram* mendiskripsikan sebuah interaksi antara satu atau lebih aktor dengan sistem informasi yang akan dibuat. Berikut simbol-simbol yang ada pada *use case diagram* (Rosa dan Shalahuddin, 2015).

Tabel 2. 3 Simbol *Use Case Diagram*

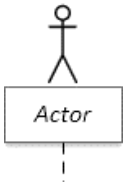
Simbol	Deskripsi
	***Actor***, menggambarkan orang, proses, atau sistem lain diluar sistem (bukan hanya pengguna sistem) yang berinteraksi dengan sistem yang dikembangkan.
	***Use case***, menggambarkan fungsionalitas yang disediakan sistem untuk pertukaran pesan.
	***Include***, menggambarkan relasi *use case* dengan *use case* tambahan dimana *use case* tambahan memerlukan *use case* utama untuk menjalankan fungsinya atau sebagai syarat dijalankannya *use case* tambahan.
	***Extend***, menggambarkan relasi *use case* dengan *use case* turunannya, dimana *use case* utama dapat berdiri sendiri tanpa *use case* tambahan (turunan)
	Association, menggambarkan

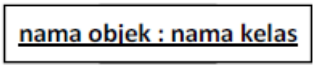
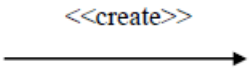
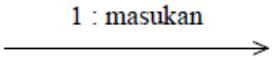
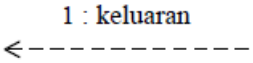
	***Generalization***,menggambarkan hubungan generalisasi dan spesialisasi (umum-khusus)antara dua buah *use case* yang menjadi *extend*-nya merupakan jenis yang sama dengan *use case* utama(yang menjadi induknya).
	relasi komunikasi antara *actor* dengan *use case* yang berpartisipasi atau *use case* memiliki interaksi dengan *actor*.

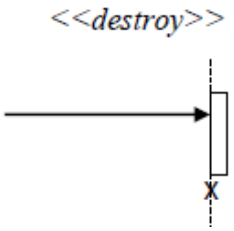
2.6.3. *Sequence Diagram*

Sequence Diagram menggambarkan kelakuan objek pada *use case* dengan mendeskripsikan waktu hidup objek dan *message* yang dikirim dan diterima oleh objek. Simbol-simbol dan fungsi *Sequence Diagram* dapat dilihat pada tabel 2.3 menurut (Rosa dan Shalahuddin, 2015) berikut:

Tabel 2. 4 Simbol *Sequence Diagram*

Simbol	**Deskripsi**
	***Actor***, orang, proses atau sistem lain yang berinteraksi dengan sistem informasi yang akan dibuat itu sendiri.

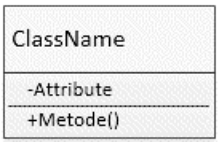
	Lifeline, menyatakan kehidupan suatu objek.
	**Objek**, menyatakan objek yang berinteraksi pesan.
	**Waktu aktif,** menyatakan objek dalam keadaan aktif dan berinteraksi.
	**Pesan Tipe *Create***, Menyatakan suatu objek membuat objek yang lain, arah panah mengarahkan pada objek yang dibuat.
	**Pesan Tipe *Call***, menyatakan suatu objek memanggil operasi/metode yang ada pada objek lain atau dirinya sendiri.
	**Pesan Tipe *Send,*** Menyatakan bahwa suatu objek mengirimkan data/masukan/informasi ke objek lainnya, arah panah mengarah pada objek yang dikirimi.
	**Pesan Tipe *Return,*** menyatakan bahwa suatu objek yang telah menjalankan suatu operasi atau metode menghasilkan suatu kembalian ke objek tertentu
	Pesan Tipe *Destroy,* menyatakan suatu objek mengakhiri hidup objek yang lalu, arah panah mengarah pada

	objek yang diakhiri.

2.7.3 *Class Diagram*

Class Diagram merupakan diagram yang menggambarkan struktur sistem dari segi pendefinisian kelas-kelas yang akan dibuat untuk membangun sistem. Berikut simbol-simbol yang digunakan dalam membuat *Class Diagram* menurut (S & Salahuddin, 2018).

Tabel 2. 5 Simbol ***Class Diagram***

Simbol	**Deskripsi**
	***Class***, menggambarkan kelas pada struktur sistem. Blok-blok pembangun pada pemrograman berorientasi obyek. Terdiri dari 3 bagian. Bagian atas adalah nama dari *class*. Bagian tengah mendefinisikan *property* atau *atribut class*. Bagian akhir mendefinisikan *method-method* dari sebuah *class*.
	Association, menggambarkan relasi antar kelas dengan makna umum, asosiasi biasanya disertai dengan *multiplicity*.
	Directed Association,

	menggambarkan relasi antar kelas dengan makna kelas yang satu digunakan oleh kelas yang lain, asosiasi biasanya disertai dengan *multiplicity*.
	***Generalization***, menggambarkan relasi antar kelas dengan makna generalisasi- spesialisasi (umum-khusus).
	***Dependency***, menggambarkan kebergantungan antarkelas.
	***Aggregation***, menggambarkan relasi antarkelas dengan makna semua bagian (*whole part*).

2.8 Konsep Basis Data

Konsep basis data antara lain:

2.8.1 Basis Data (*Database)*

Database atau basis data merupakan suatu kumpulan data yang terhubung yang disimpan secara bersama-sama pada suatu media, yang diorganisasikan berdasarkan sebuah skema atau struktur tertentu dan terdapat suatu software tertentu yang digunakan untuk manipulasi data. Basis data juga dapat diartikan sebagai tempat untuk sekumpulan berkas data terkomputerisasi, dengan tujuan utama memelihara informasi dan membuat informasi tersebut tersedia saat dibutuhkan.

Guna memudahkan dalam mengakses data, data disusun dalam suatu struktur logis yang menjelaskan bahwa:

1. Kumpulan tabel menyususn basis data.
2. Tabel tersusun atas sejumlah *record*.
3. Sebuah *record* mengandung sejumlah *field*.
4. Sebuah *field* disimpan dalam bentuk kumpulan bit.

Berikut merupakan pengertian istilah di atas:

1. *Field* menyatakan data terkecil yang memiliki makna. Istilah lain untuk *Field* yaitu elemen data, kolom item, dan atribut.
2. *Record* menyatakan kumpulan dari sejumlah elemen data yang saling terikat. Istilah lain yang juga menyatakan *record* yaitu tupel dan baris.
3. Tabel menghimpun sejumlah *record* (Jayanti & Sumiari, 2018).

2.8.2 Relasi Tabel

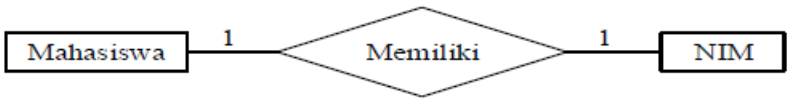
Relasi tabel merupakan hubungan antara beberapa tabel di mana relasi tabel ini dihubungkan dengan *field* yang menjadi *primary key* dan *foreign key*. *Primary key* merupakan *key fields* yang menjadi kunci (identitas) untuk setiap baris (*record*) pada suatu tabel dan tidak bisa diisi dengan data yang sama. Dengan kata lain *primary key* menjadikan tiap *record* memiliki identitas sendiri yang membedakan satu sama lainnya (unik). Sedangkan *Foreign key* merupakan *field* kunci pada suatu tabel yang digunakan pada tabel yang lain sebagai penghubung (relasi) antar tabel. *Foreign key* berguna untuk mendefinisikan kolom-kolom pada suatu tabel yang nilainya mengacu ke tabel yang lain, jadi nilai kolom foreign key nilainya harus diambil dari nilai kolom tabel lain (Mandar, 2017).

Untuk dapat membuat relasi antar tabel minimal mempunyai dua buah tabel yang saling terkait. Relasi tabel dapat berupa relasi *one-to-one*, *one-to-many*, atau *many-to-many*.

1. Relasi *One to One*

   Relasi *one to one* merepresentasikan relasi antara entitas di mana satu data memiliki relasi dengan satu dan hanya satu data saja dalam entititas yang berelasi. Sebagai contoh, setiap mahasiswa hanya memiliki 1 NIM

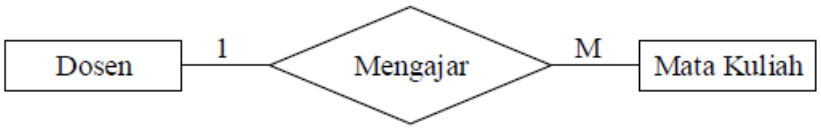
yang tidak bisa dimiliki oleh mahasiswa lain (Enterprise, 2015). Berikut gambarannya :



Gambar 2.1 Relasi *One to One*

2. Relasi *One to Many*

   Relasi *one to many* merupakan relasi yang paling umum digunakan. Pada relasi *one to many*, setiap data pada suatu entitas memiliki relasi dengan nol atau lebih data pada entitas. Sebagai contoh, Setiap dosen dapat mengajar beberapa atau lebih dari satu mata mata kuliah. Berikut gambarannya :



Gambar 2.2 Relasi *One to Many*

3. Relasi *Many to Many*

   Relasi *many to many* merupakan relasi dimana data dalam sebuah entitas memiliki relasi dengan nol atau lebih data pada entitas kedua. Dan pada saat yang bersamaan, setiap data pada entitas kedua memiliki relasi dengan nol atau lebih data pada entitas pertama. Sebagai contoh, beberapa mahasiswa dapat mengontrak beberapa mata kuliah pada setiap semester. Berikut gambarannya:

Gambar 2.3 Relasi *Many to Many*

2.8.3 Normalisasi *Database*

Normalisasi merupakan suatu teknik untuk menghasilkan sekumpulan hubungan dengan properti yang diinginkan, yang memberikan kebutuhan data

terhadap suatu perusahaan. Tujuan dari normalisasi menurut (Jayanti & Sumiari, 2018) adalah sebagai berikut:

1. Meminimalisir jumlah atribut yang diperlukan untuk mendukung kebutuhan data dari suatu lembaga pendidikan.
2. Untuk memperoleh atribut yang bersifat *functional dependencies*.
3. Untuk menghilangkan data yang bersifat *redundancy* pada tiap atribut.

Terdapat beberapa bentuk normalisasi pada database yaitu:

1. *Unnormalized Normal Form* (UNF)

   Unnormalized Normal form (UNF) merupakan sebuah tabel yang mengandung satu atau lebih *repeating group*. Berikut adalah contoh dari bentuk normalisasi UNF ditunjukan pada gambar berikut.

Tabel 2. 6 *Unnormalized Normal Form*

No. Faktur	Tgl	Kode Pelanggan	Nama	Kode Barang	Nama Barang	Harga	Qty
F-001	12/12/2016	P-001	M. Fikri Setiadi	B-001	Shampo	12.000,-	1
				B-002	Kopi	15.000,-	1
F-002	13/12/2016	P-002	Jack	B-002	Kopi	15.000,-	1
				B-003	The	7.000,-	2

2. *First Normal Form* (1NF)

   First Normal Form (1NF) merupakan sebuah relasi dimana setiap potongan baris dan kolom mengandung satu dan mungkin hanya satu nilai, dan proses untuk mengubah tabel UNF ke dalam *First Normal Form* (1NF) adalah dengan cara harus diidentifikasi dan menghilangkan bagian yang mengandung *repeating group* pada tabel. Ada dua pendekatan untuk menghilangkan perulangan kelompok *(repeating group)* dari tabel yang belum dinormalisasikan, yaitu:

   a. Dengan memasukan data kedalam kolom kosong dari baris yang berisi perulangan data. Dengan kata lain, kita mengisi yang kosong

dengan duplikat data yang tidak diulang, yang diinginkan. Pendekatan ini umumnya ditunjuk sebgai *'flattening'* pada tabel.

b. Dengan menempatkan perulangan data, sepanjang dengan sebuah salinan atribut kunci yang asli ke dalam sebuah relasi yang terpisah. Terkadang tabel yang belum dinormalisasi mungkin berisi lebih dari satu perulangan kelompok. Pada kasus seperti ini, pendekatan dapat diulang sampai tidak ada lagi perulangan yang terjadi. Sebuah set relasi dapat berada pada 1NF jika tidak terdapat perulangan kelompok *(repeating group).* Berikut adalah contoh bentuk normalisasi 1NF ditunjukan pada gambar dibawah ini.

Tabel 2. 7 *First Normal Form*

No. Faktur	Tgl	Kode Pelanggan	Nama	Kode Barang	Nama Barang	Harga	Qty
F-001	12/12/2016	P-001	M. Fikri Setiadi	B-001	Shampo	12.000,-	1
F-001	12/12/2016	P-001	M. Fikri Setiadi	B-002	Kopi	15.000,-	1
F-002	13/12/2016	P-002	Jack	B-002	Kopi	15.000,-	1
F-002	13/12/1016	P-002	Jack	B-003	The	7.000,-	2

3. *Second Normal Form* (2NF)

Second Normal Form (2NF) dapat dihasilkan dengan cara melihat apakah ada atribut yang bukan merupakan *primary key* dapat merupakan fungsi dari sebagian primary key *(partial dependence).* Dalam bentuk normal kedua setiap atribut yang bergantung secara parsial harus dipisahkan. Bentuk normal akan diperoleh bila setiap atribut yang bukan merupakan *primary key* dari suatu tabel secara penuh yang merupakan *functional dependence* dari *primary key* itu. Berikut adalah contoh bentuk normalisasi 2NF, ditunjukan pada gambar berikut.

Tabel 2. 8 *Second Normal Form*

Tabel Transaksi

No. Faktur	Tgl	Kode Pelanggan	Kode Barang	Qty
F-001	12/12/2016	P-001	B-001	1
F-001	12/12/2016	P-001	B-002	1
F-002	13/12/2016	P-002	B-002	1
F-002	13/12/1016	P-002	B-003	2

Tabel Barang

Kode Barang	Nama Barang	Harga
B-001	Shampo	12.000,-
B-002	Kopi	15.000,-
B-002	Kopi	15.000,-
B-003	The	7.000,-

4. *Third Normal Form* (3NF)

   Third Normal Form (3NF) akan secara langsung dilakukan pengujian dengan cara melihat apakah terdapat atribut bukan *key* yang bergantung fungsional terhadap atribut yang bukan *key* yang lain atau disebut *(transitive dependence).* Dengan cara yang sama, maka setiap *transitive dependence* harus dipisahkan. *Third Normal Form* (3NF) dapat dikatakan sudah normal apabila anomali yang ada di dalamnya sudah tidak ada. Berikut adalah contoh normalisasi bentuk 3NF ditunjukan pada gambar berikut.

Tabel 2. 9 *Third Normal Form (3NF)*

Tabel Transaksi

No. Faktur	Tgl	Kode Pelanggan
F-001	12/12/2016	P-001
F-001	12/12/2016	P-001
F-002	13/12/2016	P-002
F-002	13/12/1016	P-002

Tabel Detail

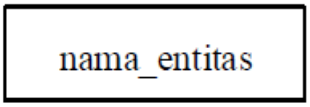
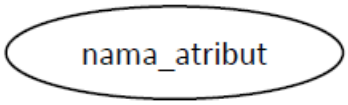
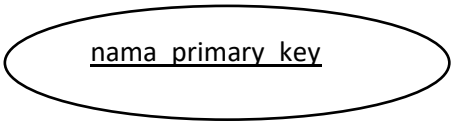
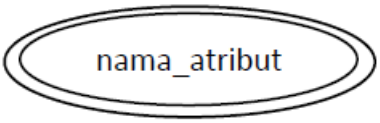
No. Faktur	Kode Barang	Qty
F-001	B-001	1
F-001	B-002	1
F-002	B-002	1
F-002	B-003	2

2.8.4 *Entity Relationship Diagram* (ERD)

Pemodelan awal basis data yang paling banyak digunakan adalah ERD. ERD dikembangkan berdasarkan teori himpunan dalam matematika. 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).

Berikut merupakan simbol-simbol yang digunakan untuk menggambarkan ERD menurut (S. Shalahuddin, 2018):

Tabel 2. 10 Simbol *Entity Relationship Diagram*

Simbol	Deskripsi
	Entitas merupakan data inti yang akan disimpan, sebagai bakal tabel pada basis data yang harus disimpan datanya agar dapat diakses oleh aplikasi komputer.
	*Field* atau kolom data yang perlu disimpan dalam suatu entitas.
	*Field* atau kolom data yang perlu disimpan dalam suatu entitas dan digunakan sebagai kunci akses *record (Primary Key)* yang diinginkan.
	*Field* atau kolom data yang perlu disimpan dalam suatu entitas yang dapat memiliki nilai lebih dari satu.
	Relasi yang menghubungkan antar entitas, biasanya diawali dengan kata kerja.

	Penghubung antar relasi dan entitas di mana kedua ujungnya memiliki *multiplicity* kemungkinan jumlah pemakaian. Kemungkinan jumlah pemakaian maksimum keterhubungan antara entitas satu dengan entitas yang Iainnya disebut dengan kardinalitas.

2.9 MySQL

MySQL adalah sebuah program *database server* yang mampu menerima dan mengirimkan datanya dengan sangat cepat, *multi user* serta menggunakan perintah standar SQL (*Structured Query Language*) (Madcoms, 2011).

Secara sederhana tipe data MySQL dapat dikelompokan menurut beberapa tipe data, berikut menurut (Anggraini, 2017).

1. Tipe Data *Integer*

   Tipe data *integer* merupakan tipe data yang tidak mempunya nilai pecahan (*decimal*). Tipe data *integer* dapat dilihat pada tabel berikut ini.

Tabel 2. 11 Tipe Data *Integer*

Nama	Fungsi	Jangkauan	Ukuran
TinyInt	menyimpan data bilangan bulat positif dan negatif.	-128 s/d 127	1 byte (8 bit).
SmallInt	menyimpan data bilangan bulat positif dan negatif.	-32.768 s/d 32.767	2 byte (16 bit).

Int	menyimpan data bilangan bulat positif dan negatif.	-2.147.483.648 s/d 2.147.483.647	4 byte (32 bit).
BigInt	menyimpan data bilangan bulat positif dan negatif.	± 9,22 x 1018	8 byte (64 it).

2. Tipe Data *Date* dan *Time*

   Tipe data *date* dan *time* merupakan tipe data yang mewakili data tanggal dan waktu. Tipe data *date* dan *time* dapat dilihat pada tabel berikut ini.

Tabel 2. 12 Tipe Data *Date* dan *Time*

Nama	**Fungsi**	**Jangkauan**	**Ukuran**
Date	menyimpan data tanggal.	1000-01-01 s/d 9999-12-31 (YYYY-MM-DD)	3 byte
Time	menyimpan data waktu.	-838:59:59 s/d +838:59:59 (HH:MM:SS)	3 byte
DateTime	menyimpan data tanggal dan waktu.	'1000-01-01 00:00:00' s/d '9999-12-31 23:59:59'	8 byte
Year	menyimpan data tahun dari tanggal.	1900 - 2155	1 byte

3. Tipe Data *String*

Tipe data *string* merupakan tipe data yang bisa digunakan untuk menampung banyak karakter dalam bentuk text, kalimat, atau kumpulan karakter. Tipe data *string* dapat dilihat pada tabel berikut.

Tabel 2. 13 Tipe Data *String*

Nama	**Fungsi**	**Jangkauan**
Char	menyimpan data string ukuran tetap.	0 s/d 255 karakter
Varchar	menyimpan data string ukuran dinamis.	0 s/d 255 karakter (versi 4.1), 0 s/d 65.535
TinyText	menyimpan data text.	0 s/d 255 karakter (versi 4.1), 0 s/d 65.535
Text	menyimpan data text.	0 s/d 65.535
MediumText	menyimpan data text.	0 s/d 224 - 1 karakter
LongText	menyimpan data text.	0 s/d 232 - 1 karakter

4. Tipe Data BLOB (*Biner*)

   Tipe data BLOB merupakan tipe data yang dapat digunakan untuk menampung gambar, musik dan vidio. Tipe data BLOB dapat dilihat pada tabel berikut ini.

Tabel 2. 14 Tipe Data BLOB*(Biner)*

Nama	**Fungsi**	**Jangkauan**
Bit	menyimpan data biner.	64 digit biner
TinyBlob	menyimpan data biner/ gambar ukuran kecil	255 byte
Blob	menyimpan data biner/ gambar	4
MediumBlob	menyimpan data biner/ gambar ukuran sedang	224-1 byte
LongBlob	menyimpan data biner/ gambar ukuran besar	232- 1 byte

2.10 PHP *(Hypertext Preprocessor)*

PHP atau kependekan dari *Hypertext Preprocessor* adalah salah satu bahasa pemrograman *open source* yang sangat cocok atau dikhususkan untuk pengembangan *web* dan dapat ditanamkan pada sebuah skrip HTML. Bahasa PHP dapat dikatakan menggambarkan beberapa bahasa pemrograman seperti C, Java, dan Perl serta mudah untuk dipelajari.

PHP merupakan bahasa *scripting server – side*, dimana pemrosesan datanya dilakukan pada sisi *server*. Sederhananya, *server* yang akan menerjemahkan skrip program, baru kemudian hasilnya akan dikirim kepada *client* yang melakukan permintaan. Adapun pengertian lain PHP adalah akronim dari *Hypertext Preprocessor*, yaitu suatu bahasa pemrograman berbasiskan kode-kode (*script*)

yang digunakan untuk mengolah suatu data dan mengirimkannya kembali ke *web browser* menjadi kode HTML. PHP (atau resminya PHP: *Hypertext Preprocessor*) adalah skrip bersifat *server-side* yang ditambahkan ke dalam HTML (Firman, Wowor, & Najoan, 2016).

2.11 HTML *(Hypertext Markup Language)*

HTML *(Hypertext Markup Languange)* adalah bahasa yang digunakan untuk menulis halaman *web*. HTML merupakan pengembangan dari standar pemformatan dokumen teks, yaitu *Standard Generalized Markup Languange* (SGML). HTML pada dasarnya merupakan dokumen ASCII atau teks biasa yang dirancang untuk tidak tergantung pada suatu sistem operasi tertentu, dibuat oleh Tim Berners-Lee untuk CERN dan dipopulerkan pertama kali oleh *browser Mosaic*. Fungsi utama HTML yaitu memberi perintah pada *browser* untuk melakukan manipulasi tampilan melalui tag-tag yang ditulis dalam HTML (Suryana & Koesheryatin, 2014).

2.12 XAMPP

XAMPP (X(windows/linux) Apache MySQL PHP dan Perl) merupakan paket *server web* PHP dan *database* MySQL yang paling popular dikalangan pengembang *web* dengan menggunakan PHP dan MySQL sebagai databasenya.

XAMPP merupakan paket *server web* PHP dan *database* MySQL yang paling populer di kalangan pengembang *web* dengan menggunakan PHP dan MySQL sebagai basisdatanya (Betha, 2014).

2.13 *Website*

Website adalah salah satu aplikasi yang berisikan dokumen-dokumen multimedia (teks, gambar, suara, animasi, video) didalamnya yang menggunakan protokol HTTP (*hypertext transfer protocol*) dan untuk mengaksesnya menggunakan perangkat lunak yang disebut *browser*. Beberapa jenis *browser* yang populer saat ini diantaranya: *Internet Explorer* yang diproduksi oleh *Microsoft*, *Mozilla Firefox*, Opera dan Safari yang diproduksi oleh *Apple*, *Browse*r (perambah) adalah aplikasi yang mampu menjalankan dokumen-dokumen *web* dengan cara diterjemahkan (Arief, 2012).

2.14 Metode Pengujian Sistem

Metode Pengujian sistem adalah cara-cara pengujian yang digunakan dalam perancangan sistem untuk mendapatkan hasil keluaran yang sesuai. Beberapa metode pengujian sistem antara lain:

2.14.1 Pengujian *Black Box*

Pengujian *black box* bertujuan untuk menemukan kesalahan fungsi pada program. Pengujian dilakukan dengan cara memasukan *input* tertentu dan melihat hasil yang didapat dari *input* tersebut. Pada pengujian *black box* memfokuskan pada keperluan fungsional dari *software*. Karena uji coba *black box* memungkinkan pengembangan *software* untuk membuat himpunan kondisi *input* yang akan melatih seluruh syarat-syarat fungsional suatu program. Uji coba *blavk box* bukan merupakan alternative dari uji coba *white box*, tetapi merupakan pendekatan yang melengkapi untuk menemukan kesalahan lainnya, selain menggunakan metode *white box* (Mustaqbal, Firdaus, & Rahmadi, 2015).

2.14.2 Pengujian *White Box*

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 dibuat terdapat kesalahan atau tidak. Kalau modul yang telah dan sudah di hasilkan berupa *output* yang tidak sesuai dengan yang diharapkan maka akan dikompilasi ulang dan dicek kembali kode-kode tersebut hingga sesuai dengan yang diharapkan.

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

1. Pengujian seluruh keputusan yang menggunakan logikal.
2. Pengujian keseluruh *loop* yang ada sesuai batasan-batasannya.
3. Pengujian pada struktur data yang sifatnya internal dan yang terjamin validitasnya.

Kelebihan *White Box Testing* 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 dipercaya tidak sesuai dan mencari kapan suatu proses perulangan diakhiri.
2. Ketidak Sesuaian 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. Kelasahan Pengetikan. Mendeteksi dan mencari bahasa-bahasa pemograman yang dianggap bersifat *case sensitife*.

Kelemahan *White Box Testing* adalah pada perangkat lunak yang jenisnya besar, metode *white box testing* dianggap boros karena melibatkan banyak sumber daya untuk melakukannya (Mustaqbal, Firdaus, & Rahmadi, 2015).