BAB II
TINJAUAN PUSTAKA DAN LANDASAN TEORI

BAB II

TINJAUAN PUSTAKA DAN LANDASAN TEORI

2.1 Tinjauan Pustaka

Berdasarkan penelitian yang dilakukan oleh Hidayat [3] sistem informasi ini dibuat karena penjualan penjualan pada Bengkel Yusuf Bekasi masih dilakukan dengan cara konvensional, yaitu pelanggan datang secara langsung ke bengkel, peneliti ingin membantu pemilik bengkel agar jangkauan pasarnya lebih luas dan meningkatkan *income*. Sistem informasi ini dikembangkan untuk membantu karyawan dalam mengelola informasi dan data produk, pelanggan, transaksi dan laporan penjualan. Pengunjung juga dapat mengakses secara *online* informasi produk, pemesanan, dan konfirmasi pembayaran serta mengisi testimoni. Metode yang digunakan pada penelitian ini adalah menganalisa sistem lebih rinci baik proses, prosedur dan fungsi sesuai dengan data yang telah dikumpulkan. Dengan adanya sistem informasi penjualan *online* ini, dapat mempermudah proses pengolahan dan pengelolaan data penjualan produk menjadi lebih akurat dan tepat.

Penelitian dilakukan oleh Solihin [4] sistem informasi ini menghasilkan aplikasi yang dapat memberi kemudahan dalam melakukan transaksi penjualan, pembelian, dan persediaan suku cadang. Penelitian ini didukung oleh beberapa kendala yang ditemukan dalam proses bisnisnya, yaitu pencatatan data transaksi penjualan membutuhkan waktu yang cukup lama karena ditulis secara konvesional sehingga banyak data yang masih tertinggal. Metode yang digunakan untuk pembangunan sistem informasi ini menggunakan model *waterfall*, dengan dimodelkan menggunakan *Flow Map*, diagram konteks, dan *Data Flow Diagram* (DFD). Dengan adanya aplikasi ini, dapat memberi kemudahan akses informasi dan proses transaksi penjualan bagi karyawan untuk melayani dan mendapatkan informasi suku cadang yang dibutuhkan oleh konsumen.

Penelitian lain yang dilakukan oleh Handayani [5] sistem ini memiliki tujuan untuk memberi solusi pemecahan masalah-masalah yang ada dengan merancang sebuah sistem informasi penjualan berbasis *e- commerce*. Sistem informasi ini dibuat karena penjualan pada Bengkel Kun Jakarta hanya menggunakan spanduk sebagai media promosi,

penjualan masih *offline*, serta penyimpanan data dan pencetakan laporan masih berupa catatan dalam bentuk arsip. Metode yang digunakan untuk pembangunan sistem informasi ini menggunakan metode *waterfall* model SDLC, dengan dimodelkan menggunakan *Unified Modeling Language* (UML). Produk aplikasi *e-commerce* dapat dijadikan sebagai media promosi, mempermudah penjualan dan dapat mengolah data dan mencetak laporan secara otomatis. Dengan adanya aplikasi *e-commerce*, pelanggan dapat memesan produk secara *online*, media promosi tidak hanya spanduk, bisa melalui *e-commerce* yang jangkauannya lebih luas, serta *e-commerce* ini mempermudah penyimpanan data dan pencetakan laporan yang dapat disimpan dan di-*retrieve* melalui *e-commerce* kapan pun karyawan mau.

Penelitian lain dilakukan oleh Zailuddin [6] dengan judul "Perancangan Sistem informasi Penjualan Berbasis *Web* (Studi Kasus: Newbiestore)". Sistem ini menghasilkan aplikasi *e-commerce*. Sistem informasi ini dibuat karena penjualan di Newbiestore masih dilakukan secara konvensional, di mana konsumen harus mendatangi bengkel untuk memilih dan membeli produk. Aplikasi yang dibuat dapat memudahkan Newbiestore dalam mengelola proses transaksi penjualan *online*, selain itu Newbiestore akan terlihat lebih profesional dan terpercaya. Metode yang digunakan untuk pembangunan sistem informasi ini menggunakan metode waterfall, dengan dimodelkan menggunakan *Unified Modeling Language* (UML), dan *Data Flow Diagram* (DFD). Dengan adanya aplikasi ini, informasi tentang produk terbaru di Newbiestore dapat terpublikasi dengan baik, dan wilayah pemasaran menjadi lebih luas.

Penelitian lain juga dilakukan oleh Prasetyo [7] sistem informasi ini dibuat karena penjualan pada Bengkel Karunia Ban Pamulang sering mengalami kesulitan untuk mengetahui klasifikasi transaksi, mana yang penjualan suku cadang dan pembayaran jasa perbaikan. Perancangan sistem informasi Penjualan Berbasis *Web* ini dibangun dengan meggunakan bahasa pemograman *PHP* dan *MySQL* sebagai data basenya dengan dimodelkan menggunakan *Unified Modeling Language* (UML). Sistem informasi ini menghasilkan aplikasi yang dapat melakukan transaksi penjualan, pembelian, penambahan stok barang, dan membuat laporan transaksi. Dengan adanya aplikaksi ini, untuk menggantikan sistem yang lama pada Bengkel Karunia Ban.

Sistem yang akan dikembangkan oleh penulis lebih baik, karena sistem ini dapat mempermudah pelanggan dalam melakukan penjualan

barang secara *online* karena pelanggan tidak perlu datang langsung ke bengkel untuk membeli *spare part*, pelanggan dapat mengakses *website* bengkel Rinjani Jaya Motor untuk membeli *spare part*. Mempermudah karyawan dalam mengontrol stok karena karyawan tidak perlu menghitung satu persatu jenis *spare part* yang masih *ready*, karyawan dapat mengontrol stok *spare part* yang masih *ready* melalui *website* bengkel Rinjani Jaya Motor. Mempermudah karyawan dalam membuat laporan penjualan yang tidak lagi dilakukan secara konvesional yaitu karyawan harus mengumpulkan nota penjualan per harinya, kemudian nota tersebut dicatat ulang pada buku, sehingga membutuhkan waktu yang cukup banyak untuk merekap hasil penjualan.

2.2 Landasan Teori

Penelitian ini diperlukan adanya teori-teori yang mendasar untuk menunjang proses penelitian ini. Teori-teori yang digunakan dalam penelitian ini adalah :

2.2.1 Sistem Informasi

Sistem informasi merupakan serangkaian komponen berupa prosedur, data dan teknologi yang digunakan untuk melakukan sebuah proses untuk menghasilkan informasi dalam pengambilan keputusan. Dua kelompok pendekatan didalam mendefiniskan sistem yaitu, yang menekankan pada prosedurnya dan yang menekankan pada komponen. Pendekatan sistem yang lebih menekankan pada prosedur mendefinisikan sistem sebagai berikut. Suatu sistem adalah jaringan kerja dari prosedur-prosedur yang saling berhubungan, berkumpul bersama-sama untuk melakukan suatu kegiatan atau untuk menyelesaikan suatu sasaran yang tertentu. Prosedur didefinisikan oleh Richard F. Neuschel sebagai berikut. Suatu prosedur adalah suatu urutan-urutan operasi klerikal (tulis-menulis), biasanya melibatkan beberapa orang didalam satu atau lebih departemen, yang diterapkan untuk menjamin penanganan yang seragam dari transaksi bisnis yang terjadi. Pendekatan sistem yang lebih menekan pada elemennya mendefinisikan sistem sebagai berikut. Sistem adalah kumpulan elemen-elemen yang beriteraksi untuk mencapai suatu tujuan tertentu.

Sistem mempunyai karakteristik atau sifat-sifat tertentu [8] , yakni antara lain :

1. Komponen sistem *(component)*

Suatu sistem terdiri dari sejumlah komponen yang saling berinteraksi, yang artinya saling bekerjasama membentuk suatu kesatuan.

2. Batasan sistem *(boundary)*

   Batasan sistem merupakan daerah yang membatasi antara suatu sistem dengan sistem yang lainnya atau dengan lingkungan luarnya.

3. Lingkungan luar sistem *(evinronment)*

   Lingkungan luar dari suatu sistem adalah apapun diluar batas dari sistem yang mempengaruhi operasi.

4. Penghubung sistem *(interface)*

   Penghubung merupakan media penghubung antara suatu subsistem dengan subsistem yang lainnya.

5. Masukan sistem *(input)*

   Masukan sistem adalah energy yang dimasukan ke dalam sistem. Sinyal input adalah energi yang diproses untuk mendapatkan keluaran dari sistem.

6. Pengolahan sistem *(process)*

   Suatu sistem harus memiliki suatu bagian pengolah yang akan merubah masukan menjadi keluaran.

7. Keluaran sistem *(output)*

   Keluaran sistem adalah hasil dari energi yang diolah dan di klarifikasikan menjadi keluaran yang bagus.

8. Sasaran sistem *(objectives)*

   Suatu sistem mempunyai tujuan atau sasaran, jika sistem tidak memiliki sasaran maka sistem tidak ada. Suatu sistem dikatakan berhasil bila mengenai sasaran atau tujuannya. Sasaran sangat berpengaruh pada masukan dan keluaran yang dihasilkan.

Informasi dapat diperoleh dari sistem informasi *(information systems)* atau disebut juga dengan *processing systems.* Sistem informasi didefinisikan oleh Robert A. Leitch dan K. Roscoe Davis adalah suatu sistem didalam organisasi yang mempertemukan kebutuhan pengolahan transaksi harian, mendukung operasi, bersifat *managerial* dan kegiatan strategi dari suatu organisasi dan menyediakan pihak luar tertentu dengan laporan-laporan [8].

Robert N. Anthony dan John Dearden menyebutkan keadaan dari sistem dalam hubungannya dengan keberakhirannya dengan istilah

entropy. Informasi yang berguna bagi sistem akan menghindari proses *entropy* yang disebut dengan *negative entropy* atau *negentropy*. Informasi adalah data yang diolah menjadi bentuk yang lebih baik berguna dan lebih berarti bagi yang menerimanya. Sumber dari informasi adalah data. Data merupakan bentuk jamak dari bentuk tunggal data-item. Data adalah kenyataan yang menggambarkan suatu kejadian-kejadian dan kesatuan yang nyata.

2.2.2 Rekayasa Perangkat Lunak

Rekayasa perangkat lunak (*software engineering*) merupakan pembangunan dengan menggunakan prinsip atau konsep rekayasa dengan tujuan menghasilkan perangkat lunak yang bernilai ekonomi yang dipercaya dan bekerja secara efisien menggunakan mesin. Perangkat lunak sering dibuat dan pada akhirnya tidak memenuhi kebutuhan pelanggan atau bahkan karena masalah non-teknis seperti keengganan pemakai perangkat lunak (*user*) untuk mengubah cara kerja dari manual ke otomatis, atau ketidakmampuan *user* menggunakan komputer. Oleh karena itu, rekayasa perangkat lunak dibutuhkan agar perangkat lunak yang dibuat tidak hanya menjadi perangkat lunak yang tidak terpakai.

Rekayasa perangkat lunak lebih fokus pada praktik pengembangan perangkat lunak yang bermanfaat kepada pelanggan (*customer*). Rekayasa perangkat lunak lebih fokus pada bagaimana membuat perangkat lunak yang memenuhi kriteria berikut [2]:

1. Dapat terus dirawat dan dipelihara (*maintainability*).
2. Dapat mengikuti perkembangan teknologi (*dependability*).
3. Dapat mengikuti keinginan pengguna (*robust*).
4. Efektif dan efisien dalam menggunakan energi dan penggunaannya.
5. Dapat memenuhi kebutuhan yang diinginkan (*usability*).

2.2.3 Metode Pengembangan Sistem

Metode yang digunakan yaitu metode *waterfall*. Metode *waterfall* sering juga disebut model sekuensial linear (*sequential linear*) atau alur hidup klasik (*classic life cycle*). Model air terjun menyediakan pendekatan alur hidup perangkat lunak secara sekuensial atau terurut dimulai dari analisis, desain, pengodean, pengujian, dan tahap pendukung (*support*). Metode *waterfall* terdiri dari 6 tahap yaitu [2] :

1) Analisis Kebutuhan Perangkat Lunak

Proses pengumpulan kebutuhan dilakukan secara intensif

untuk menspesifikasikan kebutuhan perangkat lunak agar dapat dipahami seperti apa yang dibutuhkan oleh *user*. Spesifikasi kebutuhan perangkat lunak pada tahap ini perlu di dokumentasikan.

2) Desain

Desain perangkat lunak adalah proses multi langkah yang fokus pada desain pembuatan program perangkat lunak termasuk struktur data, arsitektur perangkat lunak, representasi antar muka dan prosedur pengkodean. Tahap ini mentranslasi kebutuhan perangkat dari tahap analisis kebutuhan ke reprentasi desain agar dapat di implementasikan program pada tahap selanjutnya. Desain perangkat lunak yang dihasilkan pada tahap ini juga perlu di dokumentasikan.

3) Pembuatan Kode Program

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

4) Pengujian

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

5) Pemeliharaan (*maintenance*)

Tidak menutup kemungkinan sebuah perangkat lunak mengalami perubahan ketika sudah dikirimkan ke *user*. Perubahan bisa terjadi karena adanya kesalahan yang muncul dan tidak terdeteksi saat pengujian atau perangkat lunak.

2.2.4 Metode *Black box Testing*

Pengujian *black box* merupakan sebuah metode pengujian software dimana penguji tidak mengetahui internal struktur, desain, dan implementasian dari suatu bagian yang sedang diuji. Pengujian sistem dilakukan untuk melihat apakah sistem yang dibangun sudah sesuai dengan keinginan dari pelanggan dan layak atau tidak digunakan. Tujuan dari metode *black box* untuk mengetahui apakah sistem telah benar menampilkan kesalahan yang ada jika terjadi human error. Pada pengujian *black box* hanya mengambil hasil output melalui data uji dan mengecek fungsionalitas dari *software*. Penguji tidak perlu memiliki pengetahuan tentang Bahasa pemrograman hanya saja menguji pada

tampilan luar (*interface*).

2.2.5 Pemrograman Berorientasi Objek

Pemrograman berorientasi objek adalah suatu strategi pembangunan perangkat lunak yang mengorganisasikan perangkat lunak sebagai kumpulan objek yang berisi data dan operasi yang diberlakukan terhadapnya. Pemrograman berorientasi objek meliputi rangkaian aktivitas analisis berorientasi objek, perancangan berorientasi objek, pemrograman berorientasi objek, dan pengujian berorientasi objek.

Ada banyak cara untuk mengabstraksikan dan memodelkan objek-objek tersebut, mulai dari abstraksi objek, kelas, hubungan antarkelas sampai abstraksi sistem. Saat mengabstraksi dan memodelkan objek, data dan proses-proses yang dipunyai oleh objek akan dienkapsulasi (dibungkus) menjadi satu kesatuan. Setiap komponen dalam metodologi berorientasi objek dapat mewarisi atribut, sifat dan komponen lainnya, dan dapat berinteraksi satu sama lain. Metodologi berorientasi objek memiliki 3 karakteristik utama [2], yaitu :

1. Enkapsulalsi *(encapsulation)*
   Pembungkusan atribut data dan layanan (operasi-operasi) yang dipunyai objek untuk menyembunyikan implementasi dan objek sehingga objek lain tidak mengetahui cara kerjanya.
2. Pewarisan *(inheritance)*
   Mekanisme yang memungkinkan satu objek mewarisi sebagian atau seluruh definisi dan objek lain sebagai bagian dan dirinya.
3. Polimorfisme *(polymorphism)*
   Kemampuan suatu objek untuk digunakan dibanyak tujuan yang berbeda dengan nama yang sama sehingga menghemat baris program.

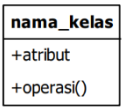
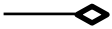
Pada pemrograman berorientasi objek, *UML* salah satu pemodelan yang digunakan untuk pemodelan sistem. *UML (Unified Modelling Language)* adalah salah satu standar bahasa yang banyak digunakan di dunia industri untuk mendefinisikan *requirement*, membuat analisis dan desain, serta menggambarkan arsitektur dalam pemrograman berorientasi objek [2].

1. *Class Diagram*

Diagram kelas atau *class diagram* menggambarkan struktur sistem dari segi pendefinisian kelas-kelas yang akan dibuat untuk membangun sistem. Kelas memiliki apa yang disebut atribut dan metode atau operasi.

Atribut merupakan variabel-variabel yang dimiliki oleh suatu kelas. Operasi atau metode adalah fungsi-fungsi yang dimiliki oleh suatu kelas. Berikut adalah simbol-simbol yang ada pada diagram kelas [2], sesuai pada Tabel 2.1.

Tabel 2. 1 Simbol-Simbol diagram kelas

No.	Simbol	Nama	Keterangan
1.		Kelas	Kelas pada struktur sistem.
No.	**Simbol**	**Nama**	**Keterangan**
2.	 nama_interface	Antarmuka / *interface*	Sama dengan konsep *interface* dalam pemrograman berorientasi objek.
3.		Asosiasi	Relasi antarkelas dengan makna umum, asosiasi biasanya juga disertai dengan *multiplicity*.
4.		Asosiasi berarah	Relasi antarkelas dengan makna kelas yang satu digunakan oleh kelas yang lain, asosiasi biasanya juga disertai dengan *multiplicity*.
5.		Generalisasi	Relasi antarkelas dengan makna generalisasi-spesialisasi (umum khusus).
6.		Agregasi	Relasi antarkelas dengan makna semua-bagian *(whole-part).*

2. *Use case Diagram*

Use case atau diagram *use case* merupakan pemodelan untuk kelakukan *(behavior)* sistem informasi yang akan dibuat. *Use case* mendeskripsikan sebuah interaksi antara satu atau lebih aktor dengan sistem informasi yang akan dibuat. *Use case* digunakan untuk mengetahui fungsi apa saja yang ada di dalan sebuah sistem informasi dan siapa saja yang berhak menggunakan fungsi-fungsi itu.

Syarat penamaan pada *use case* adalah nama didefinisikan sesimpel mungkin dan dapat dipahami. Ada dua hal utama pada *use case*

yaitu pendefinisian apa yang disebur aktor dan *use case.* Aktor merupakan orang, proses, atau sistem lain yang berinteraksi dengan sistem informasi yang akan dibuat diluar sistem informasi yang akan dibuat itu sendiri. Use case merupakan funngsionalitas yang disediakan sistem sebagai unit-unit yang saling bertukar pesan antar unit atau aktor. Berikut adalah simbol–simbol yang ada pada diagram *use case* [2], sesuai pada Tabel 2.2.

Tabel 2. 2 Simbol-Simbol diagram *use case*

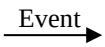
No.	Simbol	Nama	Keterangan
1.		*Use case*	Fungsionalitas yang disediakan sistem sebagai unit-unit yang saling bertukar pesan antar unit atau aktor.
2.		Aktor	Orang, proses, atau sistem lain yang berinteraksi dengan sistem informasi yang akan dibuat diluar sistem informasi itu sendiri.
3.		Asosiasi	Komunikasi antara aktor dan *use case* yang berpartisipasi pada *use case* atau *use case* memiliki interaksi dengan aktor.
4.		*Include*	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.
5.		*Extend*	Relasi *use case* tambahan ke sebuah *use case* dimana *use case* yang ditambahkan dapat berdiri sendiri walau tanpa *use case* tambahan itu.
6.		Generalisasi	Hubungan generalisasi dan spesialisasi (umum-khusus) antara dua buah *use case* dimana fungsi yang satu adalah fungsi yang lebih umum dari lainnya.

3. *State Machine Diagram*

State Machine Diagram atau *statechart diagram* atau dalam

bahasa Indonesia disebut diagram mesin status atau sering juga disebut diagram status digunakan untuk menggambarkan perubahan status atau transisi status dari sebuah mesin atau sistem atau objek. Diagram status digunakan untuk interaksi di dalam sebuah objek. Perubahan tersebut digambarkan dalam suatu graf berarah. *State Machine Diagram* merupakan pengembangan dari diagram *Finite State Automata* dengan penambahan beberapa fitur dan konsep baru. *State Machine Diagram* cocok digunakan untuk menggambarkan alur interaksi pengguna dengan sistem. Berikut ini adalah komponen-komponen dasar yang ada dalam *State Machine Diagram*, [2], sesuai pada Tabel 2.3.

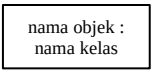
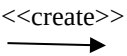
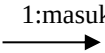
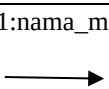
Tabel 2. 3 Komponen-komponen *State Machine Diagram*

No.	Simbol	Nama	Keterangan
1.		*Start*	Keadaan awal pada saat sistem mulai hidup.
2.		*End*	Keadaan akhir dari daur hidup suatu sistem.
3.		*Event*	Kegiatan yang menyebabkan berubahnya status mesin.
4.		*State*	Keadaan sistem dalam waktu tertentu.

4. *Sequence Diagram*

Diagram sequence menggambarkan kelakuan objek pada *use case* dengan mendeskripsikan waktu hidup objek dan *message* yang dikirimkan dan diterima antar objek. Oleh karena itu, untuk menggambar diagram sekuen maka harus diketahui objek-objek yang terlibat dalam sebuah *use case* beserta metode-metode yang dimiliki kelas yang diinstansiasi menjadi objek itu. Membuat diagram sekuen juga dibutuhkan untuk melihat skenario yang ada pada *use case*. Berikut adalah simbol-simbol yang ada pada diagram sekuen [2], sesuai pada Tabel 2.4.

Tabel 2. 4 Simbol-Simbol diagram sequence

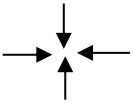
No.	Simbol	Nama	Keterangan
1.		*LifeLine*	Menyatakan kehidupan suatu objek.
2.		Aktor	Orang, proses, atau sistem lain yang berinteraksi dengan sistem informasi yang akan dibuat diluar sistem informasi itu sendiri.
3.		Objek	Menyatakan objek yang berinteraksi pesan.
4.		Waktu aktif	Menyatakan objek dalam keadaan aktif dan berinteraksi, semua yang terhubung dengan waktu aktif ini adalah sebuah tahapan yang dilakukan di dalamnya.
5.		Pesan tipe *call*	Menyatakan suatu objek memanggil operasi/metode yang ada pada objek lain atau dirinya sendiri.
6.		Pesan tipe *return*	Menyatakan bahwa suatu objek yang telah menjalankan suatu operasi atau metode menghasilkan suatu kembalian ke objek tertentu.
7.		Pesan tipe *create*	Menyatakan suatu objek membuat objek yang lain, arah panah mengarah pada objek yang dibuat.
8.		Pesan tipe *send*	Menyatakan bahwa suatu objek mengirimkan data/masukan/informasi ke objek lainnya, arah panah

No.	Simbol	Nama	Keterangan
			mengarah pada objek yang dikirimi.
9.	<<destroy>> X	Pesan tipe *destroy*	Menyatakan suatu objek yamg mengakhiri hidup objek yang lain.

2.2.6 Flowchart

Flowchart adalah adalah bagian-bagian yang mempunyai arus yang menggambarkan langkah-langkah penyelesaian suatu masalah. *Flowchart* merupakan cara penyajian dari suatu algoritma. Berikut adalah simbol-simbol yang ada pada *flowchart* [2], seperti pada Tabel 2.5.

Tabel 2. 5 Simbol-Simbol *Flowchart*

No.	Simbol	Nama	Keterangan
1.		*Terminal*	Simbol untuk memulai dan mengakhiri suatu program.
2.		*Input – Output*	Simbol untuk memasukkan data maupun menunjukan hasil dari suatu *process*.
3.		*Process Symbol*	Simbol yang menunjukkan pengolahan yang dilakukan oleh komputer.
4.		*Flow*	Merupakan prosedur yang dapat dilakukan dari atas ke bawah, bawah ke atas, kiri ke kanan, dan kanan ke kiri.
5.		*Decision*	Suatu kondisi yang akan menghasilkan beberapa kemungkinan jawaban/pilihan.
6.		*Connector*	Merupakan suatu prosedur akan masuk dan keluar dalam lembar yang sama.

No.	Simbol	Nama	Keterangan
7.		*Off line connector*	Merupakan simbol masuk dan keluarnya suatu prosedur dalam lembar kertas lain.
8.		*Document*	Merupakan simbol untuk data yang terbentuk informasi.
9.		*Predifed process*	Simbol untuk menyatakan kumpulan langkah proses ditulis sebagai prosedur.
10.		*Manual Operation*	Menunjukkan pengolahan yang tidak dilakukan oleh komputer/pc.
11.		*Display*	Simbol untuk *output* yang ditunjukan untuk device seperti printer, plotter.
12.		*Storage*	Merupakan simbol untuk menyimpan data.
13.		*Manual Input*	Untuk memasukkan data secara manual.

2.2.7 Basis Data

Basis data *(database)* adalah kumpulan informasi yang diatur agar mudah dicari. Dalam arti umum basis data adalah sekumpulan data yang diproses dengan bantuan computer yang memungkinkan data dapat diakses dengan mudah dan tepat, yang dapat digambarkan sebagai aktivitas dari satu atau lebih organisasi yang berelasi [9].

Skema menggambarkan objek yang diwakili suatu basis data, dan hubungan diantara objek tersebut. Ada banyak cara untuk mengorganisasi skema atau memodelkan struktur basis data ini dikenal sebagai model basis data atau model data. Model yang umum digunakan sekarang adalah model relasional yang menurut istilah mewakili semua informasi dalam bentuk tabel-tabel yang saling berhubungan dimana setiap tabel terdiri dari baris dan kolom. Dalam model ini, hubungan antar tabel diwakili dengan menggunakan nilai yang sama antar tabel. Model yang lain seperti model hierarkis dan model jaringan menggunakan cara yang lebih cepat eksplisit untuk mewakili hubungan antar tabel.

A. *Database Management System* (DBMS)

DBMS adalah perangkat lunak yang digunakan untuk mengelola dan mengontrol pengaksesan *database* [9].
Kelebihan menggunakan DBMS :

a. Mengontrol duplikasi data.
b. Menyediakan integrasi data.
c. Mengatur keamanan data.

Dengan menggunakan perangkat lunak ini pengelolaan data lebih mudah dilakukan. Selain itu, perangkat lunak ini juga menyediakan berbagai piranti yang berguna misalnya piranti yang memudahkan dalam membuat berbagai laporan. Bahasa yang dipakai dalam DBMS diantaranya adalah sebagai berikut [9] :

1. *Data Definition Language* (DDL)

Hasil dari perintah DDL adalah suatu set dari tabel yang disimpan dalam file khusus yang disebut data *dictionary/directory*. Fungsi DDL digunakan untuk membuat tabel, membuat key atau indeks, membuat relasi antar tabel. Beberapa *statement* atau sintaks yang sering dijumpai adalah sebagai berikut :

a. *Create tabel,* bertugas untuk membuat tabel.
b. *Create index,* bertugas untuk membuat suatu index tabel.
c. *Drop tabel,* bertugas untuk menghapus suatu tabel.
d. *Drop index,* bertuga untuk menghapus suatu index dalam tabel.
e. *Alter tabel,* bertugas untuk merubah struktur suatu tabel.

2. *Data Manipulation Language* (DML)

DML merupakan bahasa yang memperbolehkan pemakai untuk akses atau manipulasi data seperti yang telah diorganisasikan sebelumnya dalam model data yang tepat. Fungsi DML melakukan manipulasi dan pemanggilan data ke suatu *database management system* [9]. Ada dua tipe dari DML, yaitu :

a. Prosedural
   Tipe ini menuntut pemakai untuk menspesifikasikan data apa yang dibutuhkan dan bagaimana mendapatkannya. Contohnya pada *Fox Base*, D *Base* III +, dan lain-lain.
b. Nonprosedural
   Tipe ini menuntut pemakai untuk menspesifikasikan data apa yang dibutuhkan tanpa menspesifikasikan bagaimana mendapatkannya.

Contohnya pada *SQL*, *QBE*, dan lain-lain.

3. *Data Control Language* (DCL)

DCL merupakan kelompok perintah yang berisi untuk mengendalikan pengaksesan data. DCL digunakan untuk menangani masalah keamanan dalam *database server*. Pengelompokan dalam perintah DCL ini adalah *Grant* dan *Revoke*.

*B. Entity Relationship Diagram (*ERD*)*

ERD merupakan suatu cara untuk menjelaskan kepada para pemakai tentang hubungan antar data dala basis data secara *logic* dengan presepsi bahwa *real world* terdiri dari objek-objek dasar yang saling berhubungan dengan cara menvisualisasikan ke dalam bentuk simbol-simbol grafis [2]. Pada dasarnya ada tiga simbol yang digunakan, yaitu :

1. *Entitiy*

   Entity merupakan objek yang mewakili sesuatu yang nyata dan dapat dibedakan dari sesuatu yang lain. Simbol dari *entity* ini biasanya digambarkan dengan persegi panjang.

2. *Atribut*

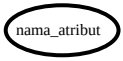
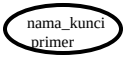
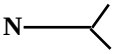
   Setiap entitas pasti mempunyai atribut yang berfungsi untuk mendeskripsikan karakteristik dari entitas tersebut. Isi dari atribut mempunyai sesuatu yang dapat mengidentifikasikan isi elemen satu dengan yang lain. Gambar atribut biasanya digambarkan dengan elips.

3. Hubungan/relasi

   Hubungan antara sejumlah entitas yang berasal dari himpunan entitas yang berbeda. Berikut simbol-simbol yang digunakan dalam ERD [2], sesuai pada Tabel 2.6.

Tabel 2. 6 Simbol-Simbol *ERD* (*Entity Relationship Diagram*)

No.	Simbol	Nama	Keterangan
1.		*Entity*	Data inti yang akan disimpan.
2.		Relasi	Relasi yang menghubungkan antar entitas.

No.	Simbol	Nama	Keterangan
3.		Atribut	Kolom data yang butuh disimpan dalam suatu entitas.
4.		Atribut kunci primer	Kolom data yang butuh disimpan dalam suatu entitas data digunakan sebagai kunci akses *record* yang diinginkan.
5.		Atribut multinilai	Kolom data yang butuh disimpan dalam suatu entitas yang dapat memiliki nilai lebih dari satu.
6.		Asosiasi	Penghubung antar relasi dan entitas dimana di kedua ujungnya memiliki *multiplicity* kemungkinan jumlah pemakaian.

Relasi antar tabel sebagai berikut [9]:

a) Hubungan *One-to-One* adalah dengan menggabungkan ke entitas yang membutuhkan atau ke entitas yang mempunyai jumlah atribut yang lebih sedikit.
b) Hubungan *One-to-Many* adalah dengan menggabungkan ke arah entitas berderajat banyak *(many)*, sehingga penggabungannya tidak perlu melihat jumlah atribut.
c) Hubungan *Many-to-Many* adalah suatu relasi atau hubungan akan membentuk relasi atau tabel tersendiri atau relasi baru.