BAB II
TINJAUAN PUSTAKA DAN DASAR TEORI

2.1 Tinjauan Pustaka

Penelitian tentang sistem pengolahan data sebelumnya telah dilakukan oleh Agus Heryanto, Hilmi Fuad dan Dani Dananggi (2014) dengan judul "Rancang Bangun Sistem Informasi Inventory Barang Berbasis Web Studi Kasus di PT.Infinetworks Global Jakarta". Penelitian ini menghasilkan sistem yang dapat digunakan untuk mempermudah dalam mencatat dan mengolah data seluruh barang, mempermudah dalam pencarian barang, memperkecil resiko kehilangan data, dan mempermudah serta mempercepat dalam penyusunan laporan [6].

Penelitian lain juga dilakukan oleh Rina Agustina (2015) dengan judul "Sistem Informasi Penjualan (Studi Kasus Di Counter Ketro)". Sistem ini dapat digunakan untuk mempermudah proses pencarian dan mengupdate suatu data untuk sebuah informasi yang lebih akurat dan lebih tepat waktu [7].

Penelitian selanjutnya dengan judul "Rancang Bangun Sistem Informasi Persediaan Barang Berbasis Web dengan Metode FAST (Framework for The Applications)" dilakukan oleh Ani Oktarini Sari dan Elan Nuari (2017). Sistem ini digunakan untuk mengefektifkan pengolahan data barang yang keluar masuk, mengefisienkan pencarian data, memantau dengan baik persediaan barang/stock barang beserta file data barang yang masuk dan keluar dan dapat diakses ketika dibutuhkan, dan memroses pelaporan menjadi lebih baik karena dapat diakses dan dicetak langsung [8].

Selanjutnya penelitian yang dilakukan oleh Nono Sudarsono dan Sukardi (2015). Penelitian ini berjudul "Sistem Informasi Inventory Berbasis Web di PT Autotech Indonesia". Sistem informasi ini dapat memberikan kemudahan kepada *user* dalam melakukan pengolahan data dan melakukan pengecekan stok barang yang ada. Sistem ini juga dapat mempercepat pembuatan laporan stok barang tanpa merekap ulang data [9].

Ada pula penelitian yang dilakukan oleh Sujono, Melati Suti Mayasari dan Koloniawan (2019) dengan judul "Prototipe Aplikasi Simpan Pinjam Pada Koperasi Darma Karya Pangkalpinang Babel". Aplikasi ini dibangun menggunakan model pengembanagn perangkat

lunak *prototype*. Aplikasi ini mampu menyimpan data simpan pinjam pada koperasi dengan rapi [5].

Dari hasil penelitian lain yang telah ada sebelumnya, maka penulis bermaksud untuk mengembangkan suatu sistem informasi persediaan barang dengan studi kasus di *Counter* Langgeng Jaya Cell. Yang membedakan sistem ini dengan sistem yang telah ada sebelumnya adalah sistem ini juga digunakan untuk mendata siapa saja penggunanya dan dapat melihat status *online/offline* setiap pengguna. Selain itu, sistem ini dibuat berbasis web, menggunakan bahasa pemrograman PHP dengan *framework* Laravel dan metode pengembangan perangkat lunak *prototype*. Di mana sistem ini mencakup beberapa bagian sistem-sistem yang sudah ada sebelumnya.

Dengan dasar sistem yang berbasis web, maka informasi bisa diakses dengan lebih mudah dan dapat menyesuaikan perangkat yang sedang digunakan. Selain itu mampu memberikan informasi persediaan barang, mengelola proses pendataan barang masuk dan barang keluar dan memberikan laporan hasil penjualan yang berupa data elektronik pada perangkat komunikasi elektronik seperti komputer, *smartphone* dan perangkat lainnya secara tepat serta dapat diakses kapanpun dan di manapun sesuai kebutuhan.

2.2 Dasar Teori

2.2.1 Sistem Informasi

Sistem informasi merupakan hasil kerja satu atau beberapa manusia yang dilakukan secara terstruktur dan sistematis dalam memproduksi sistem pengolah data yang *output*nya berupa informasi yang bermanfaat bagi penerimanya. Dengan kata lain, sistem informasi adalah sistem yang dapat memberikan informasi yang dibutuhkan oleh pengguna sedemikian rupa sehigga pengguna dapat merasakan manfaat yang diberikan oleh sistem tersebut [1].

2.2.2 Persediaan

Persediaan merupakan sejumlah barang yang disediakan oleh perusahaan dagang dengan tujuan untuk dijual [10].

2.2.3 Metode Pencatatan Persediaan

Metode pencatatan persediaan yang digunakan dalam sistem ini adalah metode pencatatan perpetual. Metode pencatatan perpetual merupakan pencatatan persediaan yang dilaksanakan setiap waktu, baik persediaan yang masuk maupun persediaan yang keluar. Jadi, dalam metode ini setiap kali terdapat transaksi pembelian barang maka secara otomatis jumlah persediaan barang akan bertambah. Begitu pula sebaliknya, jika terdapat transaksi penjualan, secrara otomatis jumlah persediaan barang akan berkurang [11].

2.2.4 Stock Opname Dalam Persediaan Barang

Stock Opname adalah kegiatan menghitung jumlah barang dagang yang terdapat di gudang untuk dijual. Tujuan dilakukannya stock opname ini adalah untuk mengetahui kecocokan antara catatan persediaan barang dengan jumlah barang yang sebenarnya. Jika terjadi selisih antara stock opname dengan catatan persediaan barang, maka ada dua kemungkinan, yaitu ada transaksi jual beli yang belum tercatat atau terjadi kecurangan dalam melakukan persediaan [11].

2.2.5 Perangakat Lunak (S*oftware*)

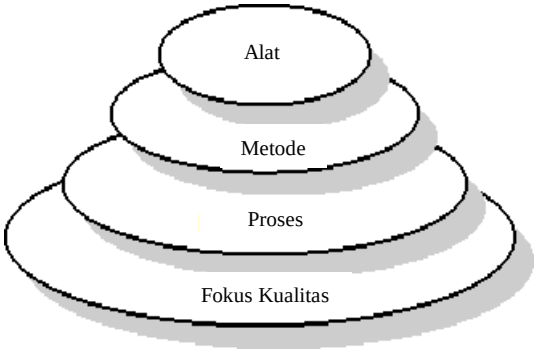
Perangkat lunak adalah program dengan sekumpulan perintah yang dimengerti oleh komputer dan digunakan untuk memproses informasi yang dibutuhkan oleh pengguna [12].

2.2.6 Rekayasa Perangkat Lunak (RPL)

Rekayasa perangkat lunak adalah satu bidang yang mendalami cara pengembangan perangkat lunak mulai dari tahap pembuatan hingga perawatan sistem saat sistem sudah dalam tahap penggunaan [13].

2.2.7 Lapisan Dalam RPL

Secara umum lapisan Rekayasa Perangkat Lunak dapat dibagi menjadi 4, yaitu Alat, Metode, Proses dan Fokus Kualitas [2].



Gambar 2. 1 Lapisan pada RPL

a. Alat merupakan perangkat yang digunakan dalam proses pembuatan sistem.

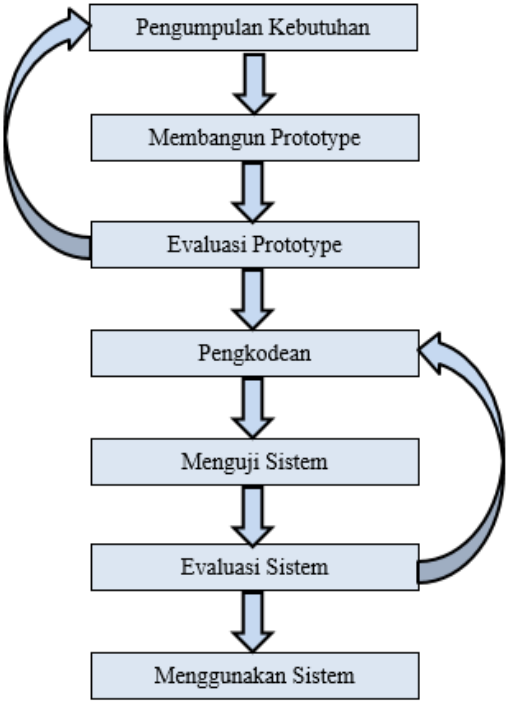
b. Metode, merupakan cara yang digunakan dalam proses pembuatan perangkat lunak.

c. Proses merupakan tahapan yang dilakukan *programmer* dalam pembuatan perangkat lunak

d. Kualitas, merupakan dasar yang menopang alat, metode dan proses dalam RPL. *Programmer* harus tau betul mengenai pandangan sistem seperti apa yang akan dibuat.

2.2.8 Proses Perangkat Lunak (*Software)*

Terdapat beberapa model proses perangkat lunak yang umum digunakan, salah satunya adalah model Prototype [5]. Urutan aktivitas di dalam model ini ditunjukkan pada Gambar 2.2.



Gambar 2. 2 Tahapan Model *Prototype*

Uraian penjelasan dalam tahapan Model *Prototype* adalah sebagai berikut :

1. Pengumpulan Kebutuhan

Pengembang dan pelanggan bersama-sama menentukan garis besar sistem yang akan dibuat dan melakukan analisis kebutuhan sistem.

Ketika kebutuhan-kebutuhan sistem dirasa sudah terpenuhi, selanjutnya pengembang membangun *prototype* sistem.

2. Membangun Prototype

Membuat rancangan sementara sebagai gambaran sistem yang akan dibuat kemudian disajikan kepada pelanggan.

3. Evaluasi Prototype

Evaluasi dilakukan oleh pelanggan untuk menentukan sesuai atau tidaknya *prototype* yang dibuat dengan hasil analisis kebutuhan sistem dan/atau keinginan pelanggan. Jika belum sesuai, maka akan kembali ke tahap pengumpulan kebutuhan, pembangunan prototype hingga evaluasi sebelum memasuki tahap pengkodean.

4. Pengkodean

Prototype yang sudah disepakati oleh pengembang dan pelanggan selanjutnya dibangun menjadi sebuah sistem menggunakan bahasa pemrograman yang sesuai.

5. Menguji Sistem

Setelah sistem selesai dibangun, dilakukan proses pengujian sistem. Pengujian ini dilakukan dengan *Black Box*. Pengujian *Black Box* dilakukan dengan menguji alur kerja sistem bagi *user* tanpa menilik susunan program-program di dalamnya.

6. Evaluasi Sistem

Sistem yang telah diuji kemudian akan dievaluasi oleh pelanggan apakah sistem sudah berjalan semestinya dan sesuai dengan prototype yang dibuat. Jika belum, maka akan dilakukan lagi proses pengkodean, pengujian sistem dan evaluasi sistem hingga sesuai dengan yang diinginkan pelanggan.

7. Menggunakan Sistem

Sistem yang telah selesai dibuat dan sesuai dengan keinginan pelanggan siap digunakan.

2.2.9 Unified Modeling Language (UML)

UML (*Unified Modeling Language)* adalah bahasa standar dalam dunia industri yang digunakan untuk merancang, menganalisis, mendesain dan menggambarkan arsitektur perangkat lunak yang akan dibuat dalam pemrograman berorientasi objek [1].

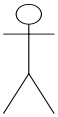
a. *Use Case Diagram*

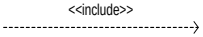
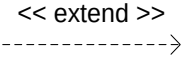
Use Case diagram merupakan uraian yang menggambarkan aktivitas yang dilakukan oleh setiap aktor pada sistem informasi yang

akan dibuat [3].

Simbol-simbol yang sering digunakan dalam *Use Case Diagram* beserta dengan fungsinya dapat dilihat pada Tabel 2.1

Tabel 2. 1 Simbol-Simbol *Use Case Diagram*

No	Simbol	Nama	Keterangan
1.		*Actor*	Orang proses atau sistem lain yang berinteraksi dengan sistem informasi yang akan dibuat diluar sistem informasi yang akan dibuat itu sendiri.
2.		*Use Case*	Fungsionalitas yang disediakan sistem sebagai unit-unit yang saling bertukar pesan antar unit atau aktor; biasanya dinyatakan menggunakan kata kerja di awali di awal frase nama nama *use case*.
3.		*Interaction*	Digunakan untuk menunjukkan baik aliran pesan atau informasi antar obyek maupun hubungan antar obyek. Biasanya *interaction* dilengkapi dengan *teks* bernama operation *signature* yang tersusun dari nama operasi, parameter yang dikirim dan tipe parameter yang dikembalikan.

No	Simbol	Nama	Keterangan
4.		*Dependency*	Merupakan relasi yang menunjukkan bahwa perubahan pada salah satu elemen memberi pengaruh pada elemen lain. Elemen yang ada di bagian tanda panah adalah elemen yang tergantung pada elemen yang ada di bagian tanpa tanda panah.
5.		*Include*	Menunjukkan bahwa suatu bagian dari elemen memicu eksekusi bagian dari elemen lain (yang ada pada tanda panah).
6.		*Extend*	Menunjukkan bahwa suatu bagian dari elemen bisa disisipkan ke dalam elemen yang lain.
7.		*Association*	Menggambarkan navigasi antar *class*, berapa banyak obyek lain yang bisa berhubungan dengan satu obyek, dan apakah suatu *class* menjadi bagian dari *class* lainnya.
8.		*Generalization*	Menunjukkan hubungan antara elemen yang lebih umum ke elemen yang lebih spesifik. Dengan *generalization*, *class*

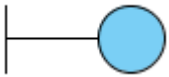
No	Simbol	Nama	Keterangan
			yang lebih spesifik (*subclass*) akan menurunkan atribut dan operasi dari *class* yang lebih umum (*superclass*). Notasi ini digunakan pada konsep *inheritance*.

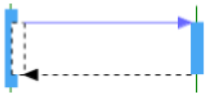
b. *Sequence Diagram*

Sequence Diagram atau diagram urutan menggambarkan kegiatan objek pada *use case* dengan mendeskripsikan interaksi antar objek yang berguna untuk mengetahui pesan yang dikirimkan dan diterima setiap objek [14].

Simbol-simbol yang sering digunakan dalam *sequence diagram* beserta dengan fungsinya dapat dilihat pada Tabel 2.2.

Tabel 2. 2 Simbol-simbol *Sequence Diagram*

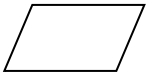
No	Simbol	Nama	Keterangan
1.		*Object Lifeline*	Menggambarkan *object* apa saja yang terlibat.
2.		*Actor*	Mengambarkan *actor* yang terlibat.
3.		*Lifeline Control*	Digunakan untuk menggambarkan sebuah *form.*
4.		*Lifeline Entity*	Digunakan untuk menggambarkan hubungan kegiatan yang akan dilakukan.
5.		*Lifeline Control*	Digunakan untuk menghubungkan *boundary* dengan tabel.

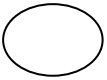
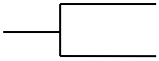
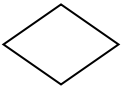
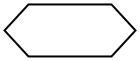
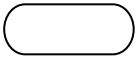
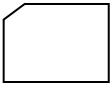
No	Simbol	Nama	Keterangan
6.		*Time Active*	Menyatakan objek dalam keadaan aktif dan berinteraksi pesan.
7.		*Synchron ous*	Relasi ini digunakan untuk memanggil operasi atau *method* yang dimiliki oleh suatu objek. *Synchronous* mengharuskan kita menyelesaikan 1 proses baru kemudian memanggil proses berikutnya.
8.		*Asynchro nous*	Relasi ini digunakan untuk memanggil operasi atau *method* yang dimiliki oleh suatu objek. *Asynchronous* memberikan kita fasilitas untuk menjalankan proses lain ketika proses sebelumnya belum selesai.

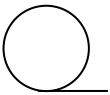
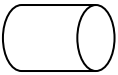
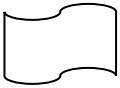
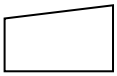
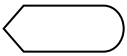
c. *Flowchart*

Flowchart atau bagan alir adalah bagan yang menunjukkan aliran suatu program atau prosedur sebuah sistem [15]. Berikut simbol-simbol dalam *Flowchart* beserta masin-masing fungsinya dapat dilihat pada Tabel 2.3.

Tabel 2. 3 Simbol-simbol *Flowchart*

No	Simbol	Nama	Keterangan
1.		*Input/Output*	Mempresentasikan *input* data atau *output* data yang diproses atau informasi.
2.		Proses	Mempresentasikan operasi

3.		Penghubung	Keluar ke atau masuk dari bagian lain, khususnya halaman yang sama.
4.		Anak Panah	Mempresentasikan alur kerja.
5.		Penjelasan	Digunakan untuk komentar tambahan.
6.		Keputusan	Ke putusan dalam program.
7.		*Predefined Process*	Rincian operasi berada di tempat lain.
8.		*Preparation*	Pemberian harga awal.
9.		*Terminal Port*	Awal/akhir *Flowchart* .
10.		*Punched Card*	*Input/Output* yang menggunakan kartu berlubang.
11.		Dokumen	*Input/Output* dalam format yang dicetak.

12.		*Magnetic Tape*	*Input/Output* yang menggunakan pita *magnetic.*
13.		*Magnetic Disk*	*Input/Output* yang menggunakan *disk magnetic*.
14.		*Magnetic Drum*	*Input/Output* yang menggunakan drum *magnetic*.
15.		*On-line Storage*	*Input/Output* yang menggunakan penyimpanan akses langsung.
16.		*Punched Tape*	*Input/Output* yang menggunakan pita kertas berlubang.
17.		*Manual Input*	*Input* yang dimasukkan secara manual dari *keyboard.*
18.		*Display*	*Output* yang ditampilkan pada terminal.
19.		*Manual Operation*	Operasi manual.
20.		*Communicatio n Link*	Transmisi data melalui *channel* komunikasi, seperti telepon
21.		*Off-line Storage*	Penyimpanan yang tidak dapat diakses oleh komputer secara langsung

2.2.8 Website

Website merupakan halaman situs sistem informasi yang dapat diakses secara luas. Website muncul karena perkembangan teknologi informasi dan komunikasi yang mendasari terciptanya suatu jaringan

antar komputer yang saling berkaitan. Jaringan tersebut dikenal sebagai internet [3].

Menurut Nugroho dalam Aprisa (2015) menjelaskan bahwa Website atau situs dapat diartikan sebagai kumpulan halaman-halaman yang berasal dari file-file berisi bahasa pemrograman yang saling berhubungan digunakan untuk menampilkan informasi, gambar bergerak dan tidak bergerak, suara dan atau gabungan dari semuanya itu baik yang bersifat statis maupun dinamis [14].

Singkatnya, *website* merupakan halaman-halaman situs berisi informasi yang terdapat di internet, dibuat dari bahasa pemrograman dan diakses melalui jaringan internet itu sendiri.

2.2.9 PHP

PHP (*Hypertext Preprocessor*) adalah bahasa pemograman yang memiliki kemampuan untuk memproses data sesuai permintaan yang digunakan untuk membuat aplikasi berbasis website atau bahasa pemograman berjenis *server-side* [1], yakni bahasa pemrograman yang letak *source code* nya terdapat di *server* dan diproses di *server* [9].

2.2.10 *Framework* Laravel [16]

Framework adalah wadah atau kerangka kerja dari sebuah website yang akan dibangun. *Framework* juga memberikan struktur yang baik dalam prgram yang dibuat karena *framework* memiliki *library* atau fungsi yang siap untuk digunakan. Dengan demikian, proses membangun sebuah *website* akan menjadi lebih mudah, waktu yang digunakan menjadi lebih singkat dan lebih mudah saat melakukan perbaikan.

Laravel merupakan *framework* berbasis PHP yang sifatnya *open source*, dan menggunakan konsep MVC (*model – view – controller*). Laravel berada di bawah lisensi MIT, dengan menggunakan GitHub sebagai sarana berbagi kode.

Berikut adalah dasar-dasar laravel :

a. Artisan

Artisan adalah *command line* atau perintah yang dijalankan melalui terminal. Salah satu fungsi dari php artisan yaitu php artisan serve yang berfungsi untuk membuka website yang telah dibuat.

b. Routing

Routing adalah proses membuat rute yang bertujuan agar item/data yang dinginkan dapat sampai ke *user* sesuai permintaan. Dengan menggunakan routing dapat ditentukan halaman halaman yang akan muncul ketika dibuka oleh *user*.

c. Controller

Controller merupakan salah satu bagian dari seluruh fungsional web dibuat, yang bertujuan untuk mengambil permintaan, menginisialisasi, memanggil model unutk dikirimkan ke view.

d. View (*Blade templating*)

Blade adalah *template engine* bawaan dari laravel. Salah satu fungsinya adalah penggunaan *layout*, agar tampilan yang berulang seperti *header, footer, sidebar, navbar* dan lain-lainnya tidak perlu dibuat berulang kali untuk setiap *view* dan terhindar dari inkonsistensi tampilan.

e. Middleware

Middleware adalah penengah Antara request yang masuk dengan controller yang dituju. Dengan kata lain, *middleware* berfungsi sebagai pencegah agar *request* yang masuk tidak diproses terlebih dahulu sebelum masuk ke *controller* dengan tujuan untuk melakukan verifikasi terhadap setiap *request* yang masuk, seperti *login*.

f. Session

Session adalah sebuah cara yang digunakan untuk penyimpanan pada *server* dan penyimpanan tersebut digunakan pada beberapa halaman termasuk halaman itu sendiri.

g. Migration

Migration adalah sebuah fitur yang ada dalam laravel , dan merupakan *Control Version System* untuk *database*. Dengan menggunakan migration, penulis bisa membuat tabel di *database* dengan lebih mudah dan cepat.

h. Model

Model merupakan salah satu dari bagian MVC yang bertugas berhubungan langsung dengan database. Bisa dikatakan juga bahwa Model adalah penghubung setiap alur program yang berhubungan dengan data. Nantinya, model yang sudah terhubung ke *database* akan digunakan/dipanggil *via Controller* sebagaimana konsep MVC itu berjalan.

2.2.11 Basis Data(*Database*)

Menurut Asrianda dalam Urva, Gellysa (2008) *Database* adalah sekumpulan tabel-tabel yang saling berelasi, relasi tersebut bisa ditunjukkan dengan kunci dari tiap tabel yang ada. Biasanya, satu *database* digunakan di dalam satu lingkup sebuah organisasi. *Database* juga merupakan sekumpulan data yang pada umumnya menggambarkan aktivitas-aktivitas sebuah sistem yang digunakan dalam sebuah organisasi. Menurut Nugroho, Yuliandri Priyo (2012) Sistem *database* merupakan sistem komputer yang digunakan untuk menyimpan dan mengelola data tersebut [14].

DBMS (*Database Management System*) adalah perangkat lunak untuk mengendalikan pembuatan, pemeliharaan, pengolahan, dan penggunaan database dalam skala yang besar. DBMS juga dirancang untuk memudahkan memanipulasi data [17].

Adapun bahasa dalam *Database Management System* (DBMS) [18] sebagai berikut :

1. *Data Definition Language* (DDL)

DDL merupakan perintah untuk membuat atau mengubah struktur objek database seperti *create, alter, drop, truncate dan rename.*

2. *Data Manipulation Language* (DML)
   DML merupakan perintah untuk melakukan manipulasi data seperti *insert, delete, update dan merge*.
3. *Data Control Language* (DCL)
   DCL merupakan perintah untuk mengatur hak akses *user* dalam *database*, seperti *grand dan revoke*.

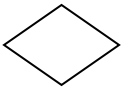
MySQL adalah *Relational Database Management System* (RDBMS) yang sebenarnya merupakan turunan SQL (*Structured Query Language*). SQL adalah sebuah konsep pengoperasian *database*, terutama untuk pemilihan atau seleksi dan pemasukan data yang memungkinkan pengolahan data dikerjakan dengan mudah secara otomatis. Keadaan suatu sistem database (DBMS) dapat diketahui dari kerja *optimizer*-nya dalam melakukan proses perintah-perintah SQL yang dibuat oleh *user* maupun program-program aplikasinya. Sebagai *database server*, MySQL dapat dikatakan lebih unggul dibandingkan *database server* lainnya dalam Query data [17].

Query adalah perintah untuk mengakses dan menampilkan suatu data dari *database* yang diambil dari tabel-tabel yang ada dalam *database* sesuai keinginan [17].

Entity Relationship Model (ERM) merupakan suatu model data yang dikembangkan berdasarkan objek dan digunakan untuk menjelaskan hubungan antar data dalam basis data kepada pengguna secara *logic*. ERM digambarkan dalam bentuk diagram yang disebut *Entity Relationship Diagram* (ERD). ERD digunakan dalam pemodelan sistem yang basis datanya akan dikembangkan serta membantu perancang/analis sistem pada saat melakukan analisis dan perancangan basis data karena model ini dapat menunjukkan macam data yang dibutuhkan dan kerelasian antar data di dalamnya [18]. Singkatnya, ERD merupakan sebuah susunan yang terdiri dari himpunan entitas dan himpunan relasi yang dilengkapi dengan atribut-atribut untuk merepresentasikan seluruh fakta hasil tinjauan sehingga hasilnya dapat digambarkan dengan lebih sistematis [19].

Simbol-simbol dan keterangan dari setiap simbol dalam ERD dapat dilihat pada tabel 2.4.

Tabel 2. 4 Simbol-simbol ERD

No	Simbol	Nama	Keterangan
1.		Himpunan *Entity*	Sesuatu apa saja yang di dalam sistem, nyata maupun abstrak dimana data tersimpan atau dimana terdapat data.
2.		Atribut	Sifat atau karakteristik dari tiap-tiap entitas maupun tiap *relationship*.
3.		Himpunan Relasi	Kumpulan semua relasi diantara entitas-entitas yang terdapat dalam himpunan entitas.
4.		*Link*	Penghubung antara himpunan relasi dengan himpunan entitas dan himpunan entitas dengan atribut.

Elemen-elemen *Entity Relationship Diagram* (ERD) [20] :

a. Entitas (*Entity*)
Digunakan untuk menunjukkan sekumpulan orang, objek, tempat, konsep dan sebagainya yang menunjukkan tempat penyimpanan data.

b. Atribut
Digunakan untuk menunjukkan karakteristik dari setiap entitas maupun hubungan.

c. Relasi
Digunakan untuk menunjukkan relasi/hubungan antar entitas.

d. *Cardinality*/Kardinalitas
Kardinalitas menunjukkan tingkat hubungan yang terjadi, dilihat dari segi kejadian atau banyak tidaknya hubungan antar entitas tersebut. Ada tiga tingkat hubungan, yaitu :

1) Satu ke satu (one to one atau 1:1) Tingkat hubungan dinyatakan satu ke satu jika suatu kejadian pada entitas pertama hanya mempunyai satu hubungan dengan satu kejadian pada entitas kedua. Demikian juga sebaliknya, satu kejadian pada entitas yang kedua hanya bisa mempunyai satu kejadian pada entitas yang pertama.

2) Satu ke banyak (one to many atau 1:M) Tingkat hubungan satu kebanyak (1:M) adalah sama dengan banyak ke satu (M:1), tergantung dari arah mana hubungan-hubungan tersebut dilihat. Untuk satu kejadian pada entitas yang pertama dapat mempunyai banyak hubungan dengan kejadian pada entitas yang kedua. Sebaliknya satu kejadian pada entitas yang kedua hanya bisa mempunyai satu hubungan dengan satu kejadian pada entitas yang pertama.

Banyak ke banyak (many to many atau M:N) Tingkat hubungan banyak ke banyak terjadi jika tiap kejadian pada sebuah entitas akan mempunyai banyak hubungan dengan kejadian pada entitas lainnya. Baik dilihat dari sisi entitas yang pertama maupun dilihat dari sisi entitas yang kedua.