# BAB II

# TINJAUAN PUSTAKA

## II.1. *Client Server*

Klien-*server* atau *client-server* merupakan sebuah paradigma dalam teknologi informasi yang merujuk kepada cara untuk mendistribusikan aplikasi ke dalam dua pihak: pihak klien dan pihak *server*. Dalam model *clien*/*server*, sebuah aplikasi dibagi menjadi dua bagian yang terpisah, tapi masih merupakan sebuah kesatuan yakni komponen klien dan komponen *server*. Komponen klien juga sering disebut sebagai front-end, sementara komponen *server* disebut sebagai back-end. Komponen klien dari aplikasi tersebut dijalankan dalam sebuah *workstation* dan menerima masukan data dari pengguna. Komponen klien tersebut akan menyiapkan data yang dimasukkan oleh pengguna dengan menggunakan teknologi pemrosesan tertentu dan mengirimkannya kepada komponen *server* yang dijalankan di atas mesin *server*, umumnya dalam bentuk request terhadap beberapa layanan yang dimiliki oleh *server*. Komponen server akan menerima request dari klien, dan langsung memprosesnya dan mengembalikan hasil pemrosesan tersebut kepada klien. Klien pun menerima informasi hasil pemrosesan data yang dilakukan *server* dan menampilkannya kepada pengguna, dengan menggunakan aplikasi yang berinteraksi dengan pengguna. (Imam Chairul Arifin dan Sutariyani ; 2014 : 38).

## II.2. *Android*

*Android* adalah sistem operasi untuk perangkat selular yang berbasis *Linux*. *Android* menyediakan latform terbuka bagi para pengembang buat menciptakan aplikasi mereka sendiri untuk digunakan oleh bermacam peranti bergerak. Awalnya, Google Inc. membeli *Android* Inc., pendatang baru yang membuat peranti lunakuntuk ponsel. Kemudian untuk mengembangkan *Android*, dibentuklah *Open Handset Alliance*, konsorsium dari 34 perusahaan peranti keras, peranti lunak, dan telekomunikasi, termasuk *Google, HTC, Intel, Motorola, Qualcomm, T-Mobile,* dan *Nvidia*. Pada saat perilisan perdana *Android*, 5 November 2007, *Android* bersama *Open Handset Alliance* menyatakan mendukung pengembangan standar terbuka pada perangkat seluler. Di lain pihak, *Google* merilis kode–kode *Android* di bawah lisensi *Apache*, sebuah lisensi perangkat lunak dan standar terbuka perangkat seluler. Di dunia ini terdapat dua jenis distributor sistem operasi. Salah satu *software* pengembang sistem *Android* adalah *eclipse*. Pada penelitian ini peneliti menggunakan *eclipse galileo*. (Andi Sanjaya ; 2014 : 9).

Pada saat perilisan perdana *Android*, 5 November 2007, *Android* bersama *Open Handest Alliance* menyatakan mendukung pengembangan *open source* pada perangkat *mobile*. Di lain pihak, *Google* merilis kode-kode *Android* dibawah lisensi *Apache*, sebuah lisensi perangkat lunak dan *open platform* perangkat seluler. (Nazruddin Safaat H ; 2014 : 2).

### II.2.1 *The Dalvik Virtual Machine* (*DVM*)

Salah satu elemen kunci dari *Android* adalah *Dalvik Virtual Machine* (*DVM*), *Android* berjalan di dalam *Dalvik Virtual Machine* (*DVM*) bukan di *Java Virtual Machine* (*JVM*), Sebenarnya banyak persamaannya dengan *Java Virtual Machine* (*JVM*) seperti *Java ME* (*Java Mobile Edition*), tetapi *Android* menggunakan *Virtual Machine* sendiriyang menurut saya dikustomisasi dan dirancang untuk memastikan bahwa beberapa *feature-feature* berjalan lebih efisien pada perangkat *mobile*. (Nazruddin Safaat H ; 2014 : 2).

### II.2.2 *Android SDK* (*Software Development Kit*)

*Android SDK* adalah *tools API* (*Application Programming Interface*) yang diperlukan untuk mulai mengembangkan aplikasi pada *platform Android* menggunakan bahasa pemrograman *Java*. *Android* merupakan *subset* perangkat lunak untuk ponsel yang meliputi sistem operasi, *middleware* dan aplikasi kunci yang di *release* oleh *Google*. Saat ini disediakan *Android SDK* (*Software Development Kit*) sebagai alat bantu dan *API* untuk mulai mengembangkan aplikasi pada *platform Android* menggunakan bahasa pemrograman *Java*. Sebagai *platform* aplikasi-netral, *Android* member anda kesempatan untuk membuat aplikasi yang kita butuhkan yang bukan merupakan aplikasi bawaan *Handphone/Smartphone*. (Nazruddin Safaat H ; 2014 : 5).

### II.2.3 *ADT* (*Android Development Tools*)

*Android Development Tools* (*ADT*) adalah *plugin* yang didesain untuk *IDE Eclipse* yang memberikan kita kemudahan dalam mengembangkan aplikasi *Android* dengan menggunakan *IDE Eclipse*. Dengan menggunakan *ADT* untuk *Eclipse* akan memudahkan kita dalam membuat aplikasi *project Android*, membuat *GUI* aplikasi, dan menambahkan komponen-komponen yang lainnya, begitu juga kita dapat melakukan *running* aplikasi menggunakan *Android SDK* melalui *eclipse*. Dengan *ADT* kita juga dapat melakukan pembuatan *package Android* (.apk) yang digunakan untuk distribusi aplikasi *Android* yang kita rancang. (Nazruddin Safaat H ; 2014 : 9).

### II.2.4 Versi *Android*

Telepon pertama yang memakai sistem operasi *Android* adalah *HTC Dream*, yang dirilis pada 22 oktober 2008. Pada penghujung tahun 2010 diperkirakan hampir semua vendor seluler didunia menggunakan *Android* sebagai *operating system*. Adapun versi-versi *Android* yang pernah dirilis adalah sebagai berikut :

1. *Android* Versi 1.1

Pada 9 MAret 2009, *Google* merilis *Android* versi 1.1. *Android* versi ini dilengkapi dengan pembaruan estetis pada aplikasi, jam, alarm, *voice search* (pencarian suara), pengiriman pesan dengan *Gmail*, dan pemberitahuan *email*.

2. *Android* Versi 1.5 (*Cupcake*)

Pada pertengahan Mei 2009, *Google* kembali merilis telepon seluler dengan menggunakan *Android* dan *SDK* (*Software Development Kit*) dengan versi 1.5 (*Cupcake*). Terdapat beberapa pembaruan termasuk juga penambahan beberapa fitur dalam seluler versi ini yakni kemampuan merekam dan menonton *video* dengan modus kamera, meng*upload video* ke *youtube* dan gambar ke *picasa* langsung dari telepon, dukungan *Bluetooth A2DP*, kemampuan terhubung secara otomatis ke *headset Bluetooth*, animasi layar, dan *keyboard* pada layar yang dapat disesuaikan dengan sistem.

3. *Android* Versi 1.6 (*Donut*)

*Donut* (versi 1.6) dirilis pada September dengan menampilkan proses pencarian yang lebih baik disbanding sebelumnya, penggunaan baterai indicator dan control *applet VPN*. Fitur lainnya adalah galeri yang memungkinkan pengguna untuk memilih foto yang akan dihapus, kamera, *camcorder* dan galeri yang diintegrasikan, *CDMA/EVDO*, 802.1x, *VPN*, *Gestures*, dan *Text-to-speech engine*, kemampuan *dial* kontak, teknologi *text to change speech* (tidak tersedia pada semua ponsel, pengadaan resolusi *VWGA*).

4. *Android* Versi 2.0/ 2.1 (*Eclair*)

Pada 3 Desember 2009 kembali diluncurkan ponsel *Android* dengan versi 2.0/ 2.1 (*Éclair*), perubahan yang dilakukan adalah pengoptimalan *hardware*, peningkatan *Google Maps* 3.1.2, perubahan *UI* dengan *browser* baru dan dukungan *HTML5*,

daftar kontak yang baru, dukungan *flash* untuk kamera 3,2 *MP*, *digital zoom*, *bluetooth* 2.1.

5. *Android* Versi 2.2 (*Froyo: Frozen Yoghurt*)

Pada bulan mei 2010 *Android* vers 2.2 Rev 1 diluncurkan. *Android* inilah yang sekarang banyak beredar di pasaran, salah satunya adalah dipakai di *Samsung FX tab* yang sudah ada di pasaran. Fitur yang tersedia di *Android* versi ini sudah kompleks diantaranya adalah :

a. Kerangka aplikasi memungkinkan penggunaan dan penghapusan komponen yang tersedia.
b. *Dalvik Virtual Machine* dioptimalkan untuk perangkat *mobile*.
c. Grafik: grafik di 2D dan grafis 3D berdasarkan *libraries OpenGL*.
d. *SQLite*: untuk penyimpanan data.
e. Mendukung media: *audio*, *video*, dan berbagai *format* gambar (MPEG4, H.264, MP3, AAC, AMR, JPG, PNG, GIF).
f. *GSM*, *Bluetooth*, *EDGE*, *3G*, dan *Wifi* (*hardware independent*).
g. Kamera, *Global Positioning System* (*GPS*), kompas, dan *accelerometer* (tergantung *hardware*).

6. *Android* Versi 2.3 (*Gingerbread*)

*Android* versi 2.3 diluncurkan pada desember 2010, hal-hal yang direvisi dari versi sebelumnya adalah kemampuan seperti berikut :

a. *SIP-based VoIP*
b. *Near Field Communications* (*NFC*)

c. *Gyroscope* dan sensor

d. *Multiple cameras support*

e. *Mixable audio effects*

f. *Download Manager*

7. *Android* Versi 3.0 (*Honeycomb*)

Dirilis Februari 2011 sebagai *Android* 3.0 revisi 1 serta *Android* versi 3.0 *revision* 2 telah dirilis pada juli 2011.

8. *Android* Versi 3.1

Dirilis Mei 2011, sedangkan *Android* 3.1 revisi 2 juga dirilis mei 2011, serta *Android* 3.1 revision 3 dirilis pada juli 2011.

9. *Android* Versi 3.2

Dirilis Juli 2011.

10. *Android* Versi 4.0

Dirilis November 2011.

11. *Android* Versi 4.1

12. *Android* Versi 4.2

13. *Android* Versi 4.3

*Android* versi 3.0 ke atas adalah generasi *platform* yang digunakan untuk *tablet pc*. Sementara versi 4.0sudah merupakan *platform* yang bias dipakai di *smartphone* dan *tablet pc*. (Nazruddin Safaat H ; 2014 : 10).

## II.3. *Remote Method Invocation* (*RMI*)

*Remote Method Invocation* (*RMI*) merupakan produk dari *Sun Microsystem* (Sekarang *Oracle*) dan salah satu teknologi sistem yang terdistribusi yang digunakan pada bahasa pemrograman *java*. *RMI* mempermudah *developer* untuk merancang aplikasi terdistribusi dimana *method* dari *remote object* dapat dipanggil melalui *JVM* (*Java Virtual Machine*) lain yang berada pada mesin atau komputer lain. Salah satu perbedaan *RMI* dengan *client server* biasa adalah *RMI* memiliki *live multi thread* tersendiri yang akan otomatis tereksekusi ketika aplikasi *server* berjalan dan memiliki *naming registry* sehingga keamanan terjaga. *RMI* memiliki dukungan paradigm pemrograman berorientasi objek dan memungkinkan aplikasi mengadaptasi teknologi komputasi terdistribusi berorientasi objek dan arsitektur *n-tier*. Dukungan *Distributed Garbage Colecction* pada *RMI* memungkinkan untuk mengumpulkan *remote object* yang tidak lagi oleh *client* maupun *server* sehingga tidak memberatkan memori sistem. (Adinandra Dharmasurya, dkk ; 2012 : 86).

## II.4. Aplikasi

Aplikasi merupakan rangkaian kegiatan atau perintah untuk dieksekusi oleh komputer atau suatu perangkat lunak komputer yang memanfaatkan kemampuan komputer langsung untuk melakukan suatu tugas yang diinginkan pengguna. Beberapa aplikasi yang digabung bersama menjadi suatu paket kadang disebut sebagai suatu paket atau suite aplikasi *(application suite)*. Aplikasi-aplikasi dalam suatu paket biasanya memiliki antar muka pengguna yang memiliki kesamaan

sehingga memudahkan pengguna untuk mempelajari dan menggunakan tiap aplikasi. Sering kali, mereka memiliki kemampuan untuk saling berinteraksi satu sama lain sehingga menguntungkan pengguna. (Ernita Sitohang ; 2013 : 2).

## II.5. Sistem

Sistem adalah sekumpulan elemen yang saling terkait atau terpadu yang dimaksudkan untuk mencapai suatu tujuan. Sebagai gambaran, jika dalam sebuah sistem terdapat elemen yang tidak memberikan manfaat dalam mencapai tujuan yang sama, maka elemen tersebut dapat dipastikan bukanlah bagian dari sistem. (Abdul Kadir ; 2014 : 61).

### II.5.1. Elemen Sistem

Elemen-elemen yang membentuk sebuah sistem yaitu :

a. Tujuan

Setiap sistem memiliki tujuan (*goal*), entah hanya satu atau mungkin banyak. Tujuan inilah yang menjadi pemotivasi yang mengarahkan sistem. Tanpa tujuan, sistem menjadi tidak terarah dan tidak terkendali. Tentu saja, tujuan antara satu sistem dengan sistem lain berbeda-beda. Begitu pula yang berlau pada sistem informasi. Setiap sistem informasi memiliki suatu tujuan, tetapi denga tujuan yang berbeda-beda. Walaupun begitu, tujuan utama yang umum ada tiga macam, yaitu :

1. Untuk mendukung fungsi kepengurusan manajemen,
2. Untuk mendukung pengambilan keputusan manajemen,
3. Untuk mendukung kegiatan operasi perusahaan.

b. Masukan

Masukan (*input*) sistem adalah segala sesuatu yang masuk kedalam sistem dan selanjutnya menjadi bahan untuk diproses. Masukan dapat berupa hal-hal berwujud (tampak secara fisik) maupun yang tidak tampak. Contoh masukan yang berwujud adalah bahan mentah, sedangkan contoh yang tidak berwujud adalah informasi (misalnya permintaan jasa dari pelanggan). Pada sistem informasi, masukan dapat berupa data transaksi, dan data non-transaksi (misalnya, surat pemberitahuan), serta instruksi.

c. Proses

Proses merupakan bagian yang melakukan perubahan atau transformasi dari masukan menjadi keluaran yang berguna, misalnya berupa informasi dan produk, tetapi juga bias berupa hal-hal yang tidak berguna, misalnya saja sisa pembuangan atau limbah. Pada pabrik kimia, proses dapat berupa pemanasan bahan mentah. Pada rumah sakit, proses dapat berupa aktivitas pembedahan pasien. Pada sistem informasi, proses dapat berupa suatu tindakan yang bermacam-macam. Meringkas data, melakukan perhitungan, dan mengurutkan data merupakan beberapa contoh proses.

d. Keluaran

Keluaran (*output*) merupakan hasi dari pemrosesan. Pada sistem informasi, keluaran bias berupa suatu informasi, saran, cetakan laporan, dan sebagainya. (Abdul Kadir ; 2014, 62).

### II.5.2. Informasi

McFadden, dkk. (1999) Mendefenisikan informasi sebagai data yang telah diproses sedemikian rupa sehingga meningkatkan pengetahuan seseorang yang menggunakan data tersebut. Shannon dan Weaver, dua orang insinyur listrik, melakukan pendekatan secara matematis untuk mendefenisikan informasi (Kroenke, 1992). Menurut mereka, informasi adalah "jumlah ketidakpastian yang dikurangi ketika sebuah pesan diterima". Artinya, dengan adanya sistem informasi, tingkat kepastian menjadi meningkat. Menurut Davis (1999), informasi adalah data yang telah diolah menjadi sebuah bentuk yang berarti bagi penerimanya dan bermanfaat dalam pengambilan keputusan saat ini atau saat mendatang. (Abdul Kadir ; 2014 : 45).

## II.6. *Java*

*Java* adalah sebuah bahasa pemrograman yang populer dikalangan para akademisi dan praktisi komputer. *Java* pertama kali dikembangkan untuk memenuhi kebutuhan akan sebuah bahasa komputer yang ditulis satu kali dan dapat dijalankan dibanyak system komputer berbeda tanpa perubahan kode berarti. Pada umumnya, para pakar pemrograman berpendapat bahwa bahasa Java memiliki konsep yang konsisten dengan teori pemrograman objek dan aman untuk digunakan.

*Java* sampai saat ini masih merupakan bahasa pemrograman yang masih sangat di minati dan banyak digunakan oleh para progremer dan software developer untuk mengembangkan berbagai tipe aplikasi, mulai dari aplikasi *console*, aplikasi *desktop*,

game, dan applet (aplikasi yang berjalan di lingkungan *web browser*), sampai ke aplikasi-aplikasi yang berskala *enterprise*. Untuk memenuhi kebutuhan tipe aplikasi yang beragam tersebut, *Java* dikategorikan menjadi tiga edisi, yaitu: *J2SE* (*Java 2 Platform Standart Edition*) untuk membuat aplikasi-aplikasi *desktop* dan *apllet, J2EE* (*Java 2 Platform Enterprise Edition*) untuk membuat aplikasi-aplikasi *multitier* berskala *enterprise*, dan *J2ME* (*Java 2 Platform Micro Edition*) untuk membuat aplikasi-aplikasi yang dapat dijalankan dilingkungan perangkat-perangkat mikro seperti *handphone, PDA* dan *Smartphone*. (Retno Wardhani dan Moh Husnul Yaqin ; 2013 : 474).

## II.7. Basis Data

Basis data dapat didefenisikan sebagai koleksi dari data-data yang terorganisasi sedemikian rupa sehingga data mudah disimpan dan dimanipulasi (diperbarui, dicari, diolah dengan perhitungan-perhitungan tertentu, serta dihapus). Secara teoritis, basis data tidak harus berurusan dengan komputer (misalnya, catatan belanja hari ini yang dibuat oleh seorang ibu rumah tangga juga merupakan basis data dalam bentuk yang sangat sederhana). (Adi Nugroho ; 2011 : 4).

## II.8. Kamus Data

Kamus data merupakan sebuah daftar yang terorganisasi dari elemen data yang berhubungan dengan sistem, dengan definisi yang tegar dan teliti sehingga pemakai dan analis sistem akan memiliki pemahaman yang umum mengenai *input*, *output*, komponen penyimpanan. (Yusi Ardi Binarso ; 2012 : 74).

## II.9. Normalisasi

Normalisasi dapat dipahami sebagai tahapan-tahapan yang masing-masing berhubungan dengan bentuk normal. Bentuk normal adalah keadaan relasi yang dihasilkan dengan menerapkan aturan sederhana berkaitan dengan konsep kebergantungan fungsional pada relasi yang bersangkutan. Kita akan menggambarkannya secara garis besar sebagai berikut :

1. Bentuk Normal Pertama (1NF/ *First Normal Form*)

   Bentuk normal pertama adalah suatu bentuk relasi dimana atribut bernilai banyak (*multivalues attribute*) telah dihilangkan sehingga kita akan menjumpai nilai tunggal (mungkin saja nilai *null*) pada perpotongan setiap baris dan kolom.

2. Bentuk Normal Kedua (2ND/ *Second Normal Form*)

   Semua kebergantungan fungsional yang bersifat sebagian (*particial functional dependency*) telah dihilangkan.

3. Bentuk Normal Ketiga (3RD/ *Thrid Normal Form*)

   Semua kebergantungan transitif (*transitive dependency*) telah dihilangkan.

4. Bentuk Normal *Boyce-Codd* (BCNF/ *Boyce-Codd Normal Form*)

   Semua anomaly yang tersisa dari hasil penyempurnaan kebergantungan fungsional sebelumnya telah dihilangkan.s

5. Bentuk Normal Keempat (4NF/ *Fourth Normal Form*)

   Semua kebergantungan bernilai banyak telah dihilangkan.

6. Bentuk Normal Kelima (5NF/ *Fifth Normal Form*)

   Semua anomaly yang tertinggi telah dihilangkan. (Adi Nugroho, 2011, Hal : 199).

## II.10. *Database MySQL*

*Database MySQL* merupakan aplikasi yang bersifat *daemon* atau menetap dalam memori yang berjalan bersama dengan sistem operasi *Microsoft Windows*. *Interface* utama *MySQL database server* adalah *command line* atau berbasis *DOS* sehingga diperlukan pengetahuan khusus mengenai penggunaan perintah atau *commandi* dalam *command shell MySQL*. (Wahana Komputer ; 2012 : 3).

## II.11. *Hypertext Preprocessor*

*Hypertext Preprocessor* adalah bahasa skrip yang dapat ditanamkan atau disisipkan ke dalam *HTML*. *PHP* banyak dipakai untuk memrogram situs *web* dynamos. *PHP* dapat digunakan untuk membangun sebuah *CMS*. Pada awalnya *PHP* merupakan kependekan dari *Personal Home Page* (situs personal). *PHP* pertama kali dibuat oleh Rasmus Lerdorf pada tahun 1995. Pada waktu Itu *PHP* masih bernama *Form Interpreted* (*FI*), yang wujudnya berupa sekumpulan skrip yang digunakan untuk mengolah data formulir dari web. Selanjutnya Rasmus merilis kode sumber tersebut untuk umum dan menamakannya *PHP/FI*. Dengan perilisan kode sumber ini menjadi sumber terbuka, maka banyak pemrogram yang tertarik untuk ikut mengembangkan *PHP*. (Anisya ; 2013 : 51).

## II.12. *Unified Modeling Language* (*UML*)

Menurut Windu Gata (2013) Hasil pemodelan pada OOAD terdokumentasikan dalam bentuk *Unified Modeling Language* (*UML*). *UML* adalah

bahasa spesifikasi standar yang dipergunakan untuk mendokumentasikan, menspesifikasikan dan membangun perangkat lunak.

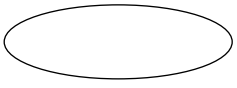
*UML* merupakan metodologi dalam mengembangkan sistem berorientasi objek dan juga merupakan alat untuk mendukung pengembangan sistem. *UML* saat ini sangat banyak dipergunakan dalam dunia industri yang merupakan standar bahasa pemodelan umum dalam industri perangkat lunak dan pengembangan sistem. (Gellysa Urva dan Helmi Fauzi Siregar ; 2015 : 93).

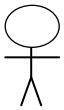
Alat bantu yang digunakan dalam perancangan berorientasi objek berbasiskan *UML* adalah sebagai berikut:

1. *Use case* Diagram

*Use case* diagram 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. Dapat dikatakan *use case* digunakan untuk mengetahui fungsi apa saja yang ada di dalam sistem informasi dan siapa saja yang berhak menggunakan fungsi-fungsi tersebut. Simbol-simbol yang digunakan dalam *use case* diagram dapat dilihat pada tabel II.2 dibawah ini:

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

| Gambar | Keterangan |
|---|---|
|  | *Use case* menggambarkan fungsionalitas yang disediakan sistem sebagai unit-unit yang bertukar pesan antar unit dengan aktor, dan dinyatakan dengan menggunakan kata kerja di awal nama *use case*. |
| | Aktor adalah *abstraction* dari orang atau sistem yang |

| | |
|---|---|
|  | lain yang mengaktifkan fungsi dari target sistem. Untuk mengidentifikasikan aktor, harus ditentukan pembagian tenaga kerja dan tugas-tugas yang berkaitan dengan peran pada konteks target sistem. Orang atau sistem bisa muncul dalam beberapa peran. Perlu dicatat bahwa aktor berinteraksi dengan *use case*, tetapi tidak memiliki *control* terhadap *use case*. |
|  | Asosiasi antara aktor dan *use case*, digambarkan dengan garis tanpa panah yang mengindikasikan siapa atau apa yang meminta interaksi secara langsung dan bukannya mengidikasikan aliran data. |
|  | Asosiasi antara aktor dan *use case* yang menggunakan panah terbuka untuk mengidinkasikan bila aktor berinteraksi secara pasif dengan sistem. |
|  | *Include*, merupakan di dalam *use case* lain (*required*) atau pemanggilan *use case* oleh *use case* lain, contohnya adalah pemanggilan sebuah fungsi program. |
|  | *Extend*, merupakan perluasan dari *use case* lain jika kondisi atau syarat terpenuhi. |

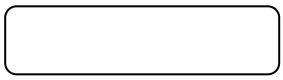
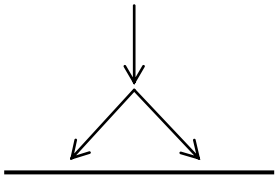
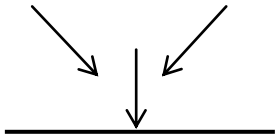
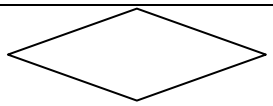
**(Sumber : Gellysa Urva dan Helmi Fauzi Siregar ; 2015 : 94)**

2. Diagram Aktivitas (*Activity Diagram*)

*Activity Diagram* menggambarkan *workflow* (aliran kerja) atau aktivitas dari sebuah sistem atau proses bisnis. Simbol-simbol yang digunakan dalam *activity diagram* dapat dilihat pada tabel II.2 dibawah ini:

**Tabel II.3. Simbol *Activity Diagram***

| Gambar | Keterangan |
|---|---|
|  | *Start point*, diletakkan pada pojok kiri atas dan merupakan awal aktifitas. |
|  | *End point*, akhir aktifitas. |

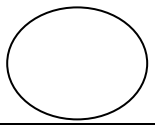
| | |
|---|---|
|  | *Activites*, menggambarkan suatu proses/kegiatan bisnis. |
|  | *Fork* (Percabangan), digunakan untuk menunjukkan kegiatan yang dilakukan secara parallel atau untuk menggabungkan dua kegiatan pararel menjadi satu. |
|  | *Join* (penggabungan) atau rake, digunakan untuk menunjukkan adanya dekomposisi. |
|  | *Decision Points*, menggambarkan pilihan untuk pengambilan keputusan, *true, false.* |
| New Swimline | *Swimlane*, untuk menunjukkan siapa melakukan apa. |

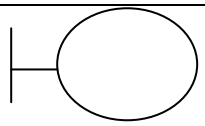
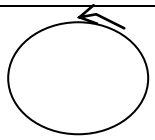
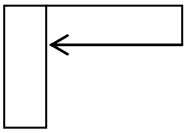
**(Sumber : Gellysa Urva dan Helmi Fauzi Siregar ; 2015 : 94)**

3. Diagram Urutan (*Sequence Diagram*)

*Sequence diagram* menggambarkan kelakuan objek pada *use case* dengan mendeskripsikan waktu hidup objek dan pesan yang dikirimkan dan diterima antar objek. Simbol-simbol yang digunakan dalam *sequence diagram* dapat dilihat pada tabel II.4 dibawah ini :

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

| Gambar | Keterangan |
|---|---|
|  | *Entity Class*, merupakan bagian dari sistem yang berisi kumpulan kelas berupa entitas-entitas yang membentuk gambaran awal sistem dan menjadi landasan untuk menyusun basis data. |

| | |
|---|---|
|  | *Boundary Class*, berisi kumpulan kelas yang menjadi *interface* atau interaksi antara satu atau lebih aktor dengan sistem, seperti tampilan formentry dan *form* cetak. |
|  | *Control class,* suatu objek yang berisi logika aplikasi yang tidak memiliki tanggung jawab kepada entitas, contohnya adalah kalkulasi dan aturan bisnis yang melibatkan berbagai objek. |
|  | *Message*, simbol mengirim pesan antar *class*. |
|  | *Recursive*, menggambarkan pengiriman pesan yang dikirim untuk dirinya sendiri. |
|  | *Activation, activation* mewakili sebuah eksekusi operasi dari objek, panjang kotak ini berbanding lurus dengan durasi aktivitas sebuah operasi. |
|  | *Lifeline,* garis titik-titik yang terhubung dengan objek, sepanjang *lifeline* terdapat *activation.* |

**(Sumber : Gellysa Urva dan Helmi Fauzi Siregar ; 2015 : 95)**

4. *Class Diagram* (Diagram Kelas)

Merupakan hubungan antar kelas dan penjelasan detail tiap-tiap kelas di dalam model desain dari suatu sistem, juga memperlihatkan aturan-aturan dan tanggng jawab entitas yang menentukan perilaku sistem. *Class diagram* juga menunjukkan atribut-atribut dan operasi-operasi dari sebuah kelas dan *constraint* yang berhubungan dengan objek yang dikoneksikan. *Class diagram* secara khas meliputi: Kelas (*Class*), Relasi, *Associations*, *Generalization* dan *Aggregation*, Atribut (*Attributes*), Operasi (*Operations/Method*), *Visibility*, tingkat akses objek eksternal kepada suatu operasi

atau atribut. Hubungan antar kelas mempunyai keterangan yang disebut dengan *multiplicity* atau kardinaliti yang dapat dilihat pada tabel II.5 dibawah ini:

**Tabel II.5. *Multiplicity Class Diagram***

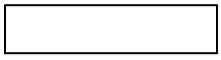
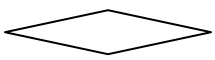
| Multiplicity | Penjelasan |
|---|---|
| 1 | Satu dan hanya satu |
| 0..* | Boleh tidak ada atau 1 atau lebih |
| 1..* | 1 atau lebih |
| 0..1 | Boleh tidak ada, maksimal 1 |
| n..n | Batasan antara. Contoh 2..4 mempunyai arti minimal 2 maksimum 4 |

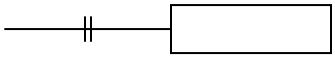
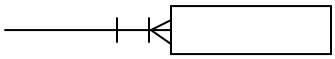
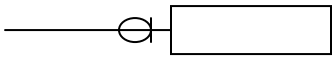
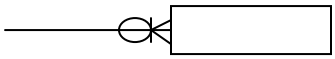
**(Sumber : Gellysa Urva dan Helmi Fauzi Siregar ; 2015 : 95)**

## II.13. *Entity Relationship Diagram*

*Entity Relationship Diagram (ERD)* adalah bagian yang menunjukkan hubungan antara *entity* yang ada dalam sistem. Simbol-simbol yang digunakan dapat dilihat dari tabel II.6. (Yuhendra, M.T, Dr. Eng dan Riza Eko Yulianto, 2015).

**Tabel II.6. Simbol Yang Digunakan Pada *Entity Relationship Diagram (ERD)***

| SIMBOL | KETERANGAN |
|---|---|
|  | *Entity* |
|  | Atribut Dan *Entity* |
|  | Atribut Dan *Entity* Dengan *Key* (Kunci) |
|  | Relasi Atau Aktifitas Antar *Entity* |

| | |
|---|---|
|  | Hubungan Satu Dan Pasti |
|  | Hubungan Banyak Dan Pasti |
|  | Hubungan Satu Tapi Tidak Pasti |
|  | Hubungan Banyak Tapi Tidak Pasti |

**(Sumber : Yuhendra, M.T, Dr. Eng dan Riza Eko Yulianto ; 2015 : 70)**

## II.14. Penjualan

Penjualan adalah pendapatan lazim dalam perusahaan dan merupakan jumlah kotor yang diberikan kepada pelanggan atas barang dan jasa.

Ada beberapa macam transaksi penjualan yaitu :

1. Penjualan Tunai

Adalah penjualan yang bersifat *cash* dan *carry* pada umumnya terjadi secara kontan dan dapat pula terjadi pembayaran selama satu bulan dianggap kontan.

2. Penjualan Kredit

Adalah penjualan dengan tenggang waktu rata-rata diatas satu bulan.

3. Penjualan Tender

Adalah penjualan yang dilaksanakan melalui prosedur tender untuk memenang-kan tender selain harus memenuhi berbagai prosedur.

4. Penjualan Ekspor

Adalah penjualan yang dilaksanakan dengan pihak pembeli luar negeri yang mengimpor barang tersebut.

5. Penjualan Konsinyasi

Adalah penjualan yang dilakukan secara titipan kepada pembeli yang juga sebagai penjual.

6. Penjualan Grosir

Adalah penjualan yang tidak langsung kepada pembeli, tetapi melalui pedagang grosir atau eceran. (Hana Yuliana dan Triandi ; 2013 : 236).

## II.15. Barang Elektronik

Barang Elektronik merupakan barang yang sangat dibutuhkan saat ini, karena barang elektronik sangat membantu manusia dalam melakukan berbagai aktifitas, seperti Televisi yang membantu manusia untuk mendapatkan informasi dan hiburan, dan masih banyak barang elektronik lainnya yang memiliki fungsi yang berbeda-beda untuk mempermudah manusia dalam melakukan berbagai aktifitas. Dari sumber data penjualan Kredit*plus,* menunjukkan permintaan produk elektronik semakin meningkat. (Dewi Kartika Pane ; 2013 : 25).