BAB II

TINJAUAN PUSTAKA

II.1. Data Mining

Secara sederhana *data mining* adalah penambangan atau penemuan informasi baru dengan mencari pola atau aturan tertentu dari sejumlah data yang sangat besar (Davies, 2004). *Data mining* juga disebut sebagai serangkaian proses untuk menggali nilai tambah berupa pengetahuan yang selama ini tidak diketahui secara manual dari suatu kumpulan data (Pramudiono, 2007).

Ada istilah lain yang mempunyai makna yang sama dengan data mining yaitu *knowledge-discovery ini database* (*KDD*). Memang data mining atau *KDD* bertujuan untuk memanfaatkan data dalam basis data dengan mengolahnya sehingga menghasilkan informasi baru yang berguna (Eko Prasetyo ; 2011:2).

II.1.1. Proses Data Mining

Secara sistematis, ada tiga langkah utama dalam data mining(Eko Prasetyo ; 2011:7).

1. Eksplorasi/ pemrosesan awal data

   Eksplorasi/ pemrosesan awal data terdiri dari ‘pembersihan’ data, normalisasi data, transformasi data, penanganan data yang salah, reduksi dimensi, pemilihan subset fitur, dan sebagainya (Eko Prasetyo ; 2011:7).

2. Membangun model dan melakukan validasi terhadapnya

   Membangun model dan melakukan validasi terhadapnya berarti melakukan analisis berbagai model dan memilih model dengan kinerja

prediksi yang terbaik. Dalam langkah ini digunakan metode-metode seperti klasifikasi, regresi, analisis cluster, deteksi anomaly, analisis asosiasi, analisis pola sekuensial, dan sebagainya. Dalam beberapa referensi, deteksi anomaly juga masuk dalam langkah eksplorasi. Akan tetapi, deteksi anomaly juga dapat digunakan sebagai algoritma utama, terutama untuk mencari data-data yang spesial (Eko Prasetyo,;2011:7).

3. Penerapan

   Penerapan berarti menerapkan model pada data yang baru untuk menghasilkan perkiraan/ prediksi masalah yang diinvestigasi (Eko Prasetyo ; 2011:7).

II.1.2. Pengelompokan Teknik Data Mining

Data Mining dibagi menjadi beberapa kelompok berdasarkan tugas yang dapat dilakukan yaitu:

1. *Classification*

   Teknik ini dapat memberikan klasifikasi pada data baru dengan memanipulasi data yang ada yang telah diklasifikasi dan dengan menggunakan hasilnya untuk memberikan sejumlah aturan.Salah satu contoh yang mudah dan popular adalah dengan Decision tree yaitu salah satu metode klasifikasi yang paling populer karena mudah untuk diinterpretasi. Decision tree adalah model prediksi menggunakan struktur pohon atau struktur berhirarki(Dennis AprillaC;2013:43).

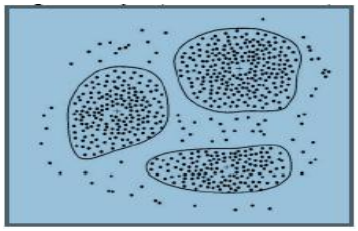
2. *Asosiacion*

   Digunakan untuk mengenali kelakuan dari kejadian-kejadian khusus atau proses dimana hubungan asosiasi muncul pada setiap kejadian. Salah satu contohnya adalah Market Basket Analysis, yaitu salah satu metode asosiasi yang menganalisa kemungkinan pelanggan untuk membeli beberapa item secara bersamaan(Dennis Aprilla C;2013:44).Sebagai contoh dapat berupa berupa studi transaksi di supermarket, misalnya seseorang yang membeli susu bayi juga membeli sabun mandi. Pada kasus ini berarti susu bayi bersama dengan sabun mandi. Karena awalnya berasal dari studi tentang *database* transaksi pelanggan untuk menentukan kebiasaan suatu produk dibeli bersama produk apa, maka aturan asosiasi juga sering dinamakan *market basket analysis.* Aturan asosiasi ingin memberikan informasi tersebut dalam bentuk hubungan "jika-maka". Ada beberapa algoritma yang sudah dikembangkan mengenai aturan asosiasi, namun ada satu algoritma klasik yang sering dipakai yaitu algoritma *apriori* (Yuli Asriningtias, Rodhyah Mardhiyah; 2014:839).

3. *Clustering*

   Tujuan utama dari metode *clustering* adalah pengelompokan sejumlah data/obyek ke dalam *cluster* (*group*) sehingga dalam setiap *cluster* akan berisi data yang semirip mungkin seperti diilustrasikan pada gambar II.1(Yuli Asriningtias, Rodhyah Mardhiyah; 2014:844). Contoh dari pengklusteran dalam bisnis yaitu mendapatkan kelompok-kelompok

konsumen untuk target pemasaran dari satu produk bagi perusahaan yang tidak memiliki dana pemasaran yang besar.



Gambar II.1. ***Clustering***
Sumber :*(Yuli Asriningtias, Rodhyah Mardhiyah;2014:844)*

II.2. Pengertian Aplikasi

Menurut Hendrayudi (2008:194), dalam buku "Visual Basic untuk berbagai keperluan program " Aplikasi adalah program komputer yang dipakai untuk melakukan pekerjaan tertentu. Brdasarkan penegertian diatas penulis menyimpulkan aplikasi adalah suatu pekerjaan atau program yang dirancang dan dihasilkan melalui komputer untuk melakukan pekerjaan tertentu.

II.3. Persediaan

Peranan Persediaan pada dasarrnya mempermudah atau memperlancar jalannya operasi perusahaan yang harus dilakukan secara berturut-turut untuk memproduksi barang-barang serta menyampaikan kepada pelanggan. Persediaan bagi perusahaan, antara lain berguna untuk :

1. Menghilangkan resiko keterlambatan datangnya barang atau bahan-bahan yang dibutuhkan perusahaan.
2. Menumpuk bahan-bahan yang dihasilkan secara musiman sehingga dapat digunakan bila bahan itu tidak ada dalam pasaran.

3. Mempertahankan stabilitas atau kelancaran operasi perusahaan.
4. Memberikan pelayanan kepada pelanggan dengan sebaik-baiknya(Friska Baramuli, Sifrid S. Pangemanan; 2015:54-55).

II.4. Metode *FP-Growth*

Algoritma *FP-Growth* merupakan pengembangan dari algoritma Apriori. Sehingga kekurangan dari algoritma Apriori diperbaiki oleh *FP-Growth. Frequent Pattern Growth (FP-Growth)* adalah salah satu alternatif algoritma yang dapat digunakan untuk menentukan himpunan data yang paling sering muncul *(frequent itemset)* dalam kumpulan data. Pada Algoritma *FP-Growth* menggunakan konsep pembangunan tree dalam pencarian *frequent itemset.* Karakter algoritma *FP-Growth* adalah struktur data yang digunakan adalah tree yang disebut *FP-Tree.* Penggalian itemset yang frequent dengan menggunakan algoritma *FP-Growth akan* dilakukan dengan cara membangkitkan struktur data *tree* atau *FP-Tree.* Algoritma *FP-Growth* merupakan algoritma *Association* Rules yang sering dipakai.

Algoritma apriori menghasilkan kombinasi yang sangat tidak efisien. Algoritma *FP-Growth* ini merupakan salah satu solusi dari algoritma apriori yang memakan waktu yang sangat lama karena harus melakukan pattern matching yang berulang-ulang. Sedangkan dalam *proses algoritma FP-Growth* terdapat banyak kelebihan yang terbukti sangat efisien karena dilakukan pemetaan data atau scan database sebanyak 2 kali untuk membangun struktur. (Fathimah Fatihatul, dkk; 2016:3-4). *Metode FP-Growth dapat* dibagi menjadi 3 tahapan utama yaitu:

1. Tahap pembangkitan conditional pattern base

   Conditional Pattern Base merupakan subdatabase yang berisi prefix path (lintasan prefix) dan suffix pattern (pola akhiran). Pembangkitan conditional pattern base didapatkan melalui FP-tree yang telah dibangun sebelumnya.

2. Tahap pembangkitan conditional *FP-Tree*

   Pada tahap ini, support count dari setiap item pada setiap conditional pattern base dijumlahkan, lalu setiap item yang memiliki jumlah support count lebih besar sama dengan minimum support count akan dibangkitkan dengan conditional FP-tree.

3. Tahap pencarian Frequent itemset

   Apabila Conditional FP-tree merupakan lintasan tunggal (single path), maka didapatkan frequent itemset dengan melakukan kombinasi item untuk setiap conditional *FP-tree*. Jika bukan lintasan tunggal, maka dilakukan pembangkitan *FP-Growth* secara rekursif (Ali Ikhwan, dkk;2015:220).

Contoh data uji coba yang diambil dari data penduduk yang mendapatkan jamkesmas dan kriterianya, seperti terdapat pada tabel di bawah ini:

Tabel II.1. Data Training

No	Nama	Luas Lanta	Jenis Lant	Jenis dind	BAB	S.Air Min	BBM	Pendaptaan	Pendidikan akhir	Aset
1	M F Kholas	12 M^2	keramik	plaster	pribadi	Pdam	lpg	> 2.000.000	SMA	> 1.000.000
2	Supriyanto	8M^2	Plaster	plaster	Pribadi	Sumur	Kayu Bakar	< 600.000	SMP	< 500.000
3	Julaikah	12M^2	Keramik	Tembok P	Pribadi	PDAM	LPG	>2.000.000	SMA	>1.000.000
4	Sri Astutik	16M^2	Keramik	Tembok P	Pribadi	Sumur	LPG	600.000-2.000.000	D3	>1.000.000
5	Satukah	16M^2	Keramik	Tembok P	Pribadi	Sumur	LPG	< 600.000	TIDAK TAMAT SD	500.000-1.000.000
6	Djenab	12M^2	Keramik	Tembok P	Umum	PDAM	Kayu Bakar	< 600.000	SD	< 500.000
7	Vuri Varika	12M^2	Plaster	Tembok T	Umum	Lainnya	LPG	>2.000.000	SMA	>1.000.000
8	Indra	12M^2	Plaster	Tembok T	Umum	Lainnya	LPG	>2.000.000	SMA	>1.000.000
9	Hartatik	18M^2	Keramik	Tembok P	Pribadi	Sumur	LPG	> 600.000	SMA	> 500.000
10	Kemisan	16M^2	Keramik	Tembok P	Pribadi	Sumur	LPG	< 600.000	TIDAK TAMAT SD	500.000-1.000.000

Sumber : (Budanis Dwi Meilani, Azmuri Wahyu Azinar ; 2015:427)

Transformasi Data

Untuk pengolahan data yang sudah diperoleh data tersebut masih belum bisa diproses langsung, oleh karena itu perlu data tersebut di-transformasi untuk mempermudah pemrosesan/pengolahan *data mining.*Atribut yang dipakai adalah luas lantai, jenis lantai, jenis dinding, fasilitas BAB, sumber air minum, bahan bakar masak, sumber pendapatan kepala rumah tangga, pendidikan kepala rumah tangga, asset yang dimiliki. Data dari atribut kemudian digolongkan dalam bentuk kategori untuk menyamakan format data.

Tabel II.2. Hasil Transformasi Data Penduduk

No	Item
1	AB, BC, CD, DB, EB, FB, GC, HC, IC
2	AB, BB, CC,DB, EA, FA, GA, HB, IA
3	AB, BC, CD, DB, EB, FB, GC, HC, IC
4	AC, BC, CD, DB, EA, FB, GB, HD, IC
5	AC, BC, CD, DB, EA, FB, GA, HA, IB
6	AB, BC, CD, DA, EB, FA, GA, HA, IA
7	AB, BB, CC, DA, EC, FB, GC, HC, IC
8	AB, BB, CC, DA, EC, FB, GC, HC, IC
9	AC, BC, CD, DB, EA, FB, GB, HC, IB
10	AC, BC, CD, DB, EA, FB, GA, HA, IB

Sumber : (Budanis Dwi Meilani, Azmuri Wahyu Azinar ; 2015:427)

Proses Data Mining FP-Growth

Penelusuran *database* yang pertama digunakan untuk menghitung nilai support masing-masing item dan memiliki item yang memenuhi nilai minimum support.

Tabel II.3. Frekuensi Kemunculan

Item	frekuensi
FB	8
BC	7
CD	7
DB	7
AB	6
EA	5
HC	5
IC	5
AC	4
GC	4
GA	4
BB	3
CC	3
DA	3
EB	3
HA	3
IB	3
EC	2
FA	2
GB	2
IA	2
HB	1
HD	1

Sumber : (Budanis Dwi Meilani, Azmuri Wahyu Azinar ; 2015:428)

Setelah di peroleh frequent list hapus item yang tidak memenuhi minimum support 50%.

Tabel II.4. Item Yang Memenuhi Minimum *Support*.

Item	frekuensi
FB	8
BC	7
CD	7
DB	7
AB	6
EA	5
HC	5
IC	5

Sumber : (Budanis Dwi Meilani, Azmuri Wahyu Azinar ; 2015:428)

Kemudian urutkan item pada tiap transaksi berdasarkan frekuensi paling tinggi.

Tabel II.5. Urut Frequent List

No	Item
1	FB, BC, CD, DB, AB, HC, IC
2	DB, AB, EA
3	FB, BC, CD, DB, AB, HC, IC
4	FB, BC, CD, DB, EA, IC
5	FB, BC, CD, DB, EA
6	BC, CD, AB
7	FB, AB, HC, IC
8	FB, AB, HC, IC
9	FB, BC, CD, DB, EA, HC
10	FB, BC, CD, DB, EA

Sumber : (Budanis Dwi Meilani, Azmuri Wahyu Azinar ; 2015:428)

Hasil FP-Growth

FB, BC, CD, BD : 6

FB,BC, CD, BD, EA : 4

Hasil FP-GROWTH kemudian dirubah ke bentuk transformasi awal sehingga menjadi sebagai berikut.

LPG, Keramik, Jenis Dinding Lain, Kamar Mandi Pribadi = 6

LPG, Keramik, Jenis Dinding Lain, Kamar Mandi Pribadi, Sumur = 4(Budanis Dwi Meilani, Azmuri Wahyu Azinar ; 2015:429).

II.5. *SQL Server* 2008

SQL Server 2008 adalah sebuah *RDBMS (Relational Database ManagementSystem)* yang sangat powerful dan telahterbukti kekuatannya dalam mengolah data.Dalam versi terbarunya ini, *SQL Server* 2008memiliki banyak fitur yang bisa dihandalkanuntuk meningkatkan performa *database.SQL Server* 2008 memiliki suatu *GUI(Graphic User Interface)* yang kita gunakanuntuk melakukan aktivitas sehari hariberkaitan dengan database,seperti menulis*T-SQL*, melakukan *backup* dan *restoredatabase*, melakukan security databaseterhadap aplikasi, dan sebagainya. Pada *GUI*tersebut kita bisa melakukan settinganterhadap *SQL Serve*r untuk berkerja lebihoptimal.Settingan juga bisa dilakukanmenggunakan script untuk memudahkandeveloper mengubah Setting Opsions pada*SQL Server* 2008(Ruslan ; 2013: 39).

II.6. *Microsoft Visual Basic 2010*

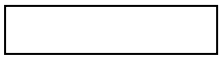
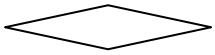
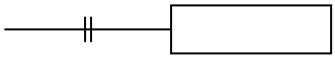
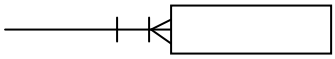
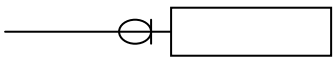
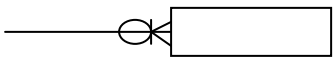
Konsep-konsep dasar dalam Visual Basic yaitu .Net Framework, bahasa yang bekerja dengan Visual Basic adalah inkarnasi dari bahasa Visual Basic yang sangat popular dan telah dilengkapi dengan fitur serta fungsi yang setara dengan bahasa tingkat tinggi lainnya seperti C ++.Visual Basic menyediakan tools dan fitur canggih yang memungkinkan untuk menulis kode, menguji, dan

menjalankanprogram tunggal atau terkadang serangkaingan satu program yang terkait dengan satu aplikasi(Christopher Lee ; 2014:1).

II.7. *Entity Relationship Diagram*

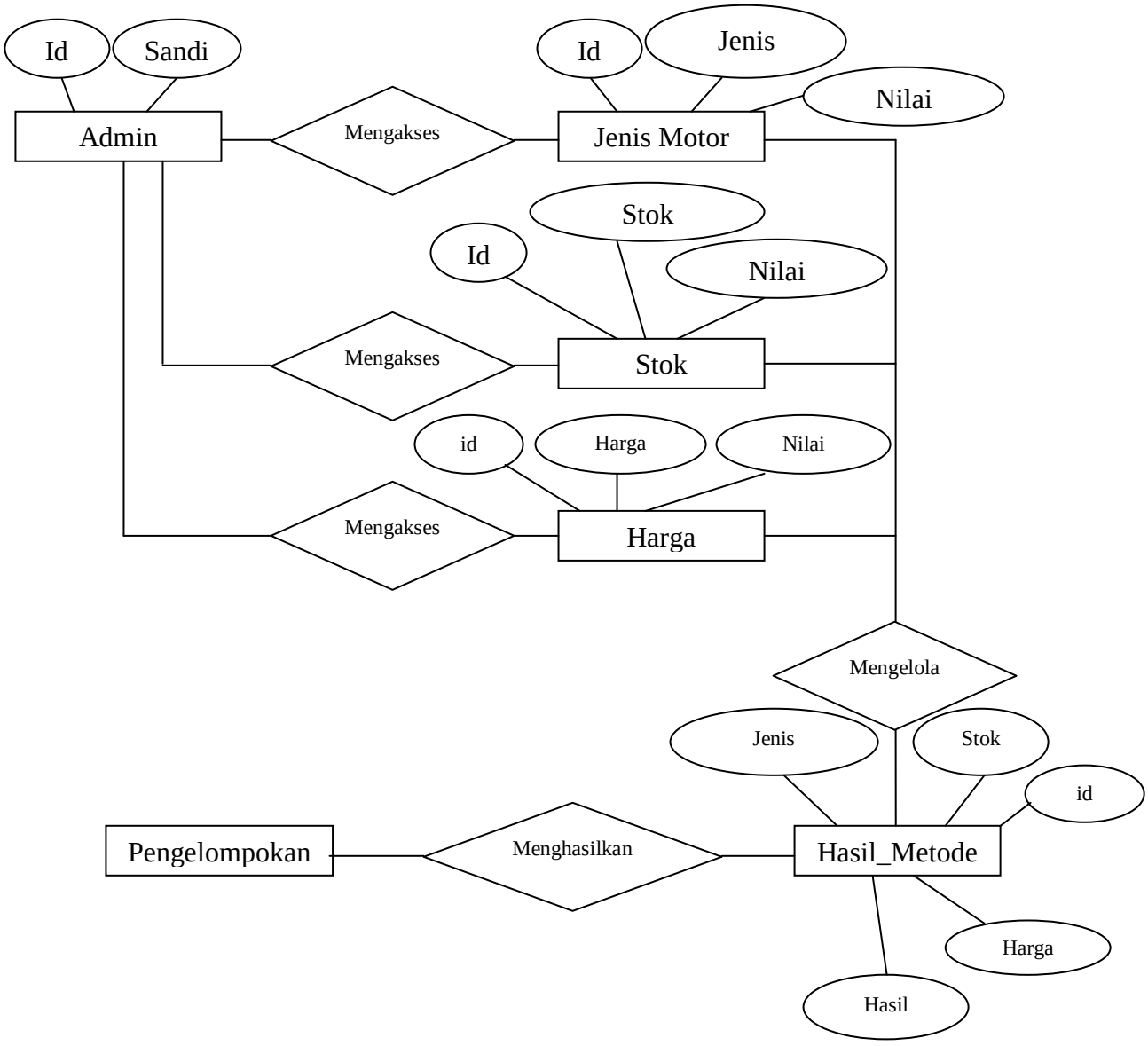
Entity Relationship Diagram (ERD) adalah bagian yang menunjukkan hubungan antara *entity* yang ada dalam sistem.Simbol-simbol yang digunakan dapat dilihat dari tabel II.6(Yuhendra, M.T, Dr. Eng, Riza Eko Yulianto; 2015:70).

Tabel II.6. Simbol Yang DigunakanPada *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, Riza Eko Yulianto; 2015:70)

Berikut adalah contoh penggunaan *Entity Relationship Diagram* (*ERD*) :



Gambar II.2. Contoh *Entity Realitionship Diagram* (*ERD*)

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

II.8. 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 (Adi Nugroho, 2011: 199). 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*)
3. Semua kebergantungan fungsional yang bersifat sebagian (*particial functional dependency*) telah dihilangkan.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.

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: 200).

Berikut ini adalah contoh normalisasi :

Bentuk tidak normal

Tabel II.7. Bentuk Tidak Normal

ID	Admin	Kriteria	Bobot
1	Ucok	Harga	3
2	Wanto	Stok	2

(Sumber :Adi Nugroho; 2011:200)

Bentuk normal pertama

Tabel II.8. Bentuk Normal Pertama

ID	Kriteria	Bobot
1	Harga	3
2	Stok	2

(Sumber :Adi Nugroho; 2011:200)

Bentuk normal kedua

Tabel II.9. Bentuk Normal Kedua

Kriteria	Bobot
Harga	3
Stok	2

(Sumber :Adi Nugroho; 2011:200)

II.9. *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.

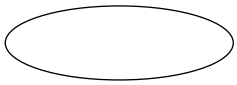
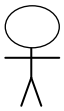
UML merupakan metodologi dalam mengembangkan sistem berorientasi objek dan juga merupakan alat untuk mendukung pengembangan sistem. *UML* saat ini sangat banyak dipergunakan dalam dunia industri yang merupakan standar bahasa pemodelan umum dalam industri perangkat lunak dan pengembangan sistem (Gellysa Urva, 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.10 dibawah ini:

Tabel II.10. Simbol *Use Case*

Gambar	Keterangan
	*Use case* menggambarkan fungsionalitas yang disediakan sistem sebagai unit-unit yang bertukar pesan antar unit dengan aktor, biasanya 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, Helmi Fauzi Siregar ; 2015 : 94)

2. *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.11 dibawah ini:

Tabel II.11. *Multiplicity Class Diagram*

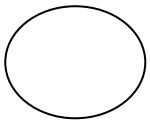
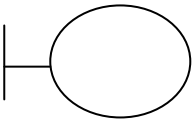
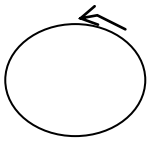
Multiplicity	Penjelasan
1	Satu dan hanya satu
0..*	Boleh tidak ada atau 1 atau lebih
1..*	1 atau lebih
0..1	Boleh tidak ada, maksimal 1
n..n	Batasan antara. Contoh 2..4 mempunyai arti minimal 2 maksimum 4

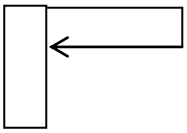
(Sumber:Gellysa Urva, Helmi Fauzi Siregar ; 2015 : 95)

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.12 dibawah ini :

Tabel II.12. Simbol *Sequence Diagram*

Gambar	Keterangan
	*EntityClass,* 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*.

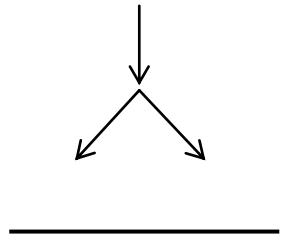
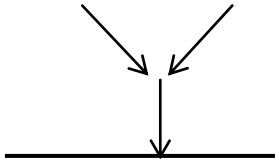
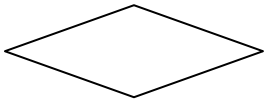
(Sumber : Gellysa Urva dan Helmi Fauzi Siregar; 2015: 95)

4. 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.13 dibawah ini:

Tabel II.13. 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.*
	*Swimlane*, pembagian *activity* diagram untuk menunjukkan siapa melakukan apa.

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