BAB II

TINJAUAN PUSTAKA

II.1. Penelitian Terkait

Berdasarkan penelitian yang dilakukan oleh Sulaksono dan Darsono (2015) mengenai sistem pakar penentuan penyakit gagal jantung menggunakan metode *Naïve Bayes Classifier* dari hasil uji coba sebanyak 3 kali dengan menggunakan data training yang berbeda dihasilkan akurasi tertinggi pada uji coba yang ke 1 dengan akurasi 86% yang tediri dari 134 data training dan 66 data uji coba. Sehingga disimpulkan bahwa semakin banyak data training yang diuji coba semakin akurat hasilnya.

Berdasarkan penelitian yang dilakukan oleh Nurlelah dan Wajhillah (2016) mengenai penerapan *Naïve Bayes* untuk diagnosa penyakit diare usia balita pada sistem pakar berbasis website dari hasil uji coba aplikasi dapat membantu para tenaga medis dan juga orang tua dalam mengetahui lebih dini penyakit diare pada anak balita tanpa harus berkonsultasi secara langsung dengan pakar atau tenaga medis.

II.2. Sistem

Sistem adalah suatu kesatuan prosedur atau komponen yang saling berkaitan satu dengan yang lainnya bekerja bersama-sama sesuai dengan aturan yang diterapkan sehingga membentuk suatu tujuan yang sama, dimana dalam sebuah sistem bila terjadi satu bagian saja yang tidak bekerja atau rusak maka suatu

tujuan bisa terjadi kesalahan hasilnya atau outputnya. Sistem merupakan kumpulan elemen yang saling berkaitan yang bertanggung jawab memproses masukan (*input)* sehingga menghasilkan keluaran (*output).* (Luckyana Puspita Sari, 2013).

Ada beberapa karakteristik yang membentuk sebuah system yaitu:

1. Komponen Sistem

   Suatu sistem terdiri dari sejumlah komponen yang saling berinteraksi, yang bekerja sama membentuk satu kesatuan. Komponen-komponen tersebut dapat berupa suatu bentuk subsistem atau bagian-bagian dari sistem.

2. Batasan Sistem

   Batas sistem merupakan daerah yang membatasi antara sistem dengan sistem lainnya atau sistem dengan lingkungan luarnya. Batasan sistem ini memungkinkan suatu sistem dipandang sebagai satu kesatuan, karena dengan batas system ini fungsi dan tugas dari subsistem yang satu dengan lainnya berbeda tetapi tetap saling berinteraksi. Batas suatu system menunjukkan ruang lingkup (*scope*) dari system tersebut.

3. Lingkungan Luar Sistem

   Environment merupakan segala sesuatu diluar batas sistem yang memengaruhi operasi dari suatu sistem. Lingkungan luar system ini dapat bersifat menguntungkan atau merugikan.Lingkungan luar yang menguntungkan harus dipelihara dan dijaga agar tidak hilang pegaruhnya, sedangkan lingkungan luar yang merugikan harus dimusnahkan atau dikendalikan agar tidak mengganggu operasi sistem.

4. Penghubung Sistem

   Merupakan media penghubung antara satu subsistem dengan subsistem yang lainnya untuk membentuk satu kesatuan sehingga sumber-sumber daya mengalir dari subsistem yang satu kesubsistem yang lainnya. Dengan kata lain, *output* dari suatu subsistem akan menjadi *input* dari subsistem yang lainnya.

5. Masukan Sistem

   Merupakan energi yang dimasukan ke dalam sistem. Masukan dapat berupa perawatan (*maintenance input*) adalah energi yang dimasukkan supaya system tersebut dapat beroperasi. Masukan sinyal (*signal input*) adalah energi yang diproses untuk didapatkan keluaran.

6. Keluaran Sistem

   Merupakan hasil dari energi yang diolah oleh system, meliputi *output* yang berguna.

   Setiap system pasti mempunyai tujuan ataupun sasaran yang memengaruhi *input* yang dibutuhkan dan *output* yang dihasilkan. Dengan kata lain, suatu sistem akan dikatakan berhasil kalau pengoperasian sistem itu mengenai sasaran atau tujuannya. Jika sistem tidak mempunyai sasaran, maka operasi sistem tidak aka nada gunanya. (Sulindawati dan Muhammad Fathoni, 2010).

Sistem merupakan suatu bentuk integrasi antara satu komponen dengan komponen lain karena sistem memiliki sasaran yang berbeda setiap kasus yang terjadi yang ada di dalam sistem tersebut. Oleh karena itu sistem dapat diklasifikasikan dari beberapa sudut pandangannya antara lain :

a. Sistem Abstrak (*Abstract System*)

   Sistem yang berupa pemikiran atau ide-ide yang tidak tampak secara fisik.

b. Sistem Fisik (*Physical System*)

   Merupakan system yang ada secara fisik sehingga setiap makhluk dapat melihatnya.

c. Sistem Alamiah (*Natural System*)

   Sistem yang terjadi melalui proses alam, dalam artian tidak dibuat, seperti system tata surya, system galaxy, system reproduksi, dan lain-lain.

d. Sistem Buatan Manusia (*Human Made System*)

   Sistem yang dirancang oleh manusia. Sistem buatan manusia yang melibatkan interaksi manusia dengan mesin disebut *Human Machine System.*

e. Sistem Tertentu (*Deterministic System*)

   Beroperasi dengan tingkah laku yang sudah dapat diprediksi. Interaksi bagian-bagiannya dapat dideteksi dengan pasti sehingga keluaran dari system dapat diramalkan.

f. Sistem Tak Tentu (*Probabilistic* System)

   Sistem yang kondisi masa depannya tidak dapat diprediksi karena mengandung unsur probabilitas.

g. Sistem Tertutup (*Closed System*)

Sistem yang tidak berhubungan dan tidak terpengaruh dengan system luarnya. Sistem ini bekerja secara otomatis tanpa adanya turut campur tangan dari pihak luarnya. Secara teori, system tersebut ada, tetapi kenyataannya tidak ada system yang benar-benar tertutup, yang ada hanyalah *relatively closed system* (secara relative tertutup, tidak benar-benar tertutup).

h. Sistem Terbuka (*Open System*)

Sistem yang berhubungan dan terpengaruh dengan lingkungan luarnya. Lebih spesifik dikenal juga yang disebut dengan system terotomasi, yang merupakan bagian dari system buatan manusia dan berinteraksi dengan control oleh satu atau lebih computer sebagai bagian dari system yang digunakan dalam masyarakat modern. (Sulindawati dan Muhammad Fathoni, 2010).

II.3. Informasi

Informasi adalah data yang diolah menjasi suatu bentuk yang berarti bagi penerimanya dan bermanfaat dalam pengambilan keputusan pada saat sekarang atau yang akan datang. (Sulindawati dan Muhammad Fathoni, 2010)

II.3.1. Sistem Informasi

Suatu sistem adalah suatu sistem di dalam organnisasi yang merupakan kombinasi dari orang – orang, fasilitas, teknologi, media, prosedur-prosedur, dan pengendalian yang ditujukan untuk mendapatkan jalur kombinasi yang penting. (Sulindawati dan Muhammad Fathoni, 2010)

II.3.2. Data

Data adalah bahan mentah dari kejadian – kejadian nyata yang dirangkum kemudian diolah sehingga data mentah tersebut menjadi informasi. (Akhwan Sanoto S.Kom MSI, 2013)

II.4. Sistem Pakar

Sistem Pakar adalah sistem berbasis komputer yang menggunakan pengetahuan, fakta dan teknik penalaran dalam memecahkan masalah yang biasanya hanya dapat di pecahkan oleh seorang pakar dalam bidang tersebut. (Muhammad Silmi dkk, Vol. 4 No. 7)

Sistem pakar tidak lepas dari elemen manusia yang terkait di dalamnya. Personil yang terkait dengan sistem pakar ada 4 yaitu :

1. Pakar (*expert*)
2. Pembangun pengetahuan (*knowledge engineer*)
3. Pembangunan sistem (*system engineer*)
4. Pemakai (*user*)

Sistem pakar merupakan cabang dari *artificial intelegency (AI)* yang sudah lama karena sistem ini telah mulai dikembangkan pada pertengan tahun 1960 sistem pakar yang muncul pertama kali adalah jeneral *purpose problem solper (GPS)* yang dikembangkan oleh Newl dan Simon.

1. *Knowledge Base* (Basis Pengetahuan)

Basis pengetahuan mengandung oengetahuan untuk pemahaman, formulasi, dan penyelesaian masalah. Komponen sistem pakar disusun atas dua elemen dasar, yaitu fakta dan aturan. Fakta merupakan informasi tentang objek dalam

area permasalahan tertentu, sedangkan aturan merupakan informasi tentang cara bagaimana memperoleh fakta baru dari fakta yang telah diketahui.

2. *Inference Engine* (Mesin Inferensi)

   Mesin inferensi merupakan otak dari sebuah sistem pakar dan dikenal juga dengan sebutan *control structure* (struktur kontrol) atau *rule interpreter* (dalam sistem pakar berbasis kaidah). Komponen ini mengandung mekanisme pola pikir dan penalaran yang digunakan oleh pakar dalam menyelesaikan suatu masalah. Mesin inferensi disini adalah *processor* pada sistem pakar yang mencocokkan bagian kondisi dari *rule* yang tersimpan di dalam *knowledge base* dengan fakta yang tersimpan di *working memory*.

3. *Working Memory*

   Berguna untuk menyimpan fakta yang dihasilkan oleh *inference engine* dengan penambahan parameter berupa derajat kepercayaan atau dapat juga dikatakan sebagai global *database* dari fakta yang digunakan oleh *rule-rule* yang ada.

4. *Explanation Facility*

   Menyediakan kebenaran dari solusi yang dihasilkan kepada *user* (*reasoning chain*).

5. *Knowledge Acquistion Facility*

   Meliputi proses pengumpulan, pemindahan dan perubahan dari kemampuan pemecahan masalah seorang pakar atau sumber pengetahuan terdokumentasi ke program *computer*, yang bertujuan untuk memperbaiki atau mengembangkan basis pengetahuan.

6. *User Interface*

   Mekanisme untuk memberi kesempatan kepada *user* dan sistem pakar untuk berkomunikasi. Antar muka menerima informasi dari pemakai dan mengubahnya ke dalam bentuk yang dapat diterima oleh sistem. Selain itu antarmuka menerima informasi dari sistem dan menyajikannya ke dalam bentuk yang dapat dimengerti oleh pemakai.

Sistem pakar umumnya dirancang untuk memenuhi beberapa karakteristik umum berikut ini :

1. Kinerja sangat baik (*high performance*). Sistem harus mampu memberikan respon berupa saran (*advice*) dengan tingkat kualitas yang sama dengan seorang pakar atau melebihinya.
2. Waktu respon yang baik (*adequate respon time*). Sistem juga harus mampu bekerja dalam waktu yang sama baiknya (*reasonable*) atau lebih cepat dibandingkan dengan seorang pakar dalam menghasilkan keputusan. Hal ini sangat penting terutama pada sistem waktu nyata (*real-time*).
3. Dapat diandalkan (*good reliability*). Sistem harus dapat diandalkan dan tidak mudah rusak / *crash*.
4. Dapat dipahami (*understandable*). Sistem harus mampu menjelaskan langkah-langkah penalaran yang dilakukannya seperti seorang pakar.
5. Fleksibel (*flexibility*). Sistem harus menyediakan mekanisme untuk menambah, mengubah, dan menghapus pengetahuan.

II.5. *Vaginitis*

Vaginitis merupakan infeksi pada vagina yang disebabkan oleh berbagai parasit atau jamur. Infeksi ini sebagian besar terjadi karena hubungan seksual. Tipe *Vaginitis* yang sering dijumpai adalah *Vaginitis kandidiasis* dan *trikomonalis vaginalis*. *Vaginitis* dapat menyebabkan infertilitas karena berpotensi terjadi infeksi lanjut pada portio, serviks, endometrium bahkan sampai ke tuba yang dapat menyebabkan gangguan pergerakan dan penyumbatan pada tuba sebagai organ reproduksi vital untuk terjadinya konsepsi. Disfungsi seksual yang mencegah penetrasi penis, atau lingkungan vagina yang sangat asam, yang secara nyata dapat mengurangi daya hidup sperma. (Trinawati, 2015 : 169).

II.6. *Naive Bayes*

Naive Bayes merupakan pendekatan statistik untuk melakukan inferensi induksi pada persoalan klasifikasi. Metode ini menggunakan probabilitas bersyarat sebagai dasarnya. (Chomaya dan Ardian, 2013 : 3).

Saat diagnosa dimulai, sistem akan melakukan kalkulasi bayes sekaligus dimana satu sisi perhitungan dilakukan terhadap bobot probabilitas yang sudah direkam dari Pakar dan disimpan di basis pengetahuan. Secara umum *Naive Bayes* dengan E kejadian dan Hipotesis H dapat dituliskan dalam bentuk :

$$p(H|E) = \frac{p(E|H) * p(H)}{P(E)}$$

Dengan :

P(H | E) = probabilitas hipotesis H jika diberikan *evidence* E

P(E |H) = probabilitas munculnya *evidence* apapun

P(E) = probabilitas *evidence* E

Naive Bayes dapat dikembangkan jika setelah dilakukan pengujian terhadap hipotesis kemudian muncul lebih dari satu *evidence.*

$$P(H \mid E, e) = P(H \mid E) \frac{P(e \mid E, H)}{P(e \mid E)}$$

Keterangan :

e : *evidence* lama

E : *evidence* baru

P(H | E,e) : *probabilitas hipotesis H* benar jika muncul *evidence* baru *E* dari *evidence* baru *E* dari *evidence* lama *e*.

P(H | E) : *probabilitas hipotesis H* benar jika diberikan evidence *E*.

P(e | E,H) : kaitan antar *e* dan *E* jika *hipotesis H* benar.

P(e | E) : kaitan antara *e* dan *E* tanpa memandang *hipotesis* apapun.

II.7. Normalisasi

Normalisasi merupakan parameter digunakan untuk menghindari duplikasi terhadap tabel dalam basis data dan juga merupakan proses mendekomposisikan sebuah tabel yang masih memiliki beberapa anomali atau ketidakwajaran sehingga menghasilkan tabel yang lebih sederhana dan struktur yang bagus, yaitu sebuah tabel yang tidak memiliki *data redundancy* dan memungkinkan *user* untuk melakukan *insert*, *delete*, dan *update* pada baris (*record*) tanpa menyebabkan inkonsistensi data. (Triyono, 2012 : 19).

1. *First Normal Form (1 NF)*

   Sudah tidak ada *repeating group* yaitu pengulangan yang terjadi pada beberapa atribut atau kolom dalam sebuah tabel, dan juga setiap atribut harus bernilai tunggal. Atribut *multivalued, composite, derive* tidak tunggal. Setiap nilai dari atribut hanya mempunyai nilai tunggal.

2. *Second Normal Form (2 NF)*

   Untuk menjadikan tabel normal tingkat ke 2 maka sudah 1NF dan setiap atribut yang bukan *primary key* sepenuhnya secara *fungsional* tergantung pada semua atribut pembentuk *primary key*.

3. *Third Normal Form (3 NF)*

   Tabel sudah 2NF dan tidak memiliki *transitive dependencies*, *Transitive dependency* adalah ketika ada atribut yang secara tidak langsung tergantung pada *primary key* dan atribut tersebut juga tergantung pada atribut lain yang bukan *primary key*.

4. *Boyce-codd Normal Form (BCNF)*

   Tabel dalam BCNF jika sudah 3NF dan semua *determinants* adalah *candidate keys*. Perbedaan 3NF dan BCNF adalah untuk *functional dependency* A □ B, 3NF memperbolehkan ketergantungan ada dalam relasi jika B adalah *Primary Key* dan A bukan merupakan *candidate key*. Sedangkan BCNF menuntut untuk ketergantungan tetap ada dalam relasi, A harus menjadi *candidate key*.

5. *Fourth Normal Form* (4 NF)

   Relasi berada pada bentuk normal keempat apabila memenuhi syarat BCNF dan tidak mempunyai *multivalue dependency.*

6. *Fifth Normal Form* (5 NF)

   Tabel bentuk normal kelima sering disebut PJNF (*Projection Join Normal Form*), penyebutan PJNF karena untuk suatu relasi akan berbentuk normal kelima jika tabel tersebut dapat dipecah atau diproyeksikan menjadi beberapa tabel dan dari proyeksi-proyeksi itu dapat disusun kembali (*join*) menjadi tabel yang sama dengan keadaan semula. Jika penyusunan ini tidak mungkin dilakukan dikatakan pada relasi itu terdapat *join dependencies* dan dikatakan bersifat *lossy join*. (Triyono, 2012 : 20).

II.8. *Hypertext Preprocessor* (PHP)

Hypertext Preprocessor (PHP) yang merupakan bahasa pemrograman berbasis web yang memiliki kemampuan untuk memproses data dinamis. PHP dikatakan sebagai sebuah *server-side embedded script language* artinya sintaks-sintaks dan perintah yang kita berikan akan sepenuhnya dijalankan oleh *server* tetapi disertakan pada halaman HTML biasa. (Kusumawaty, 2012 : 3).

Hypertext Preprocessor (PHP) yang merupakan bahasa pemrograman berbasis web yang memiliki kemampuan untuk memproses data dinamis. PHP dikatakan sebagai sebuah *server-side embedded script language* artinya semua sintaks dan perintah yang ditulis sepenuhnya dijalankan oleh *server* tetapi disertakan pada halaman HTML biasa. (Nugraha, dkk, 2014 : 175).

PHP adalah sebuah bahasa pemrograman *scripting* untuk membuat halaman *web* yang dinamis. PHP dikatakan sebagai sebuah *server-side embedded script language* artinya sintak-sintak dan perintah yang kita berikan akan

sepenuhnya dijalankan oleh *server* tetapi disertakan pada halaman HTML yang seperti biasa. Aplikasi-aplikasi yang dibangun oleh PHP pada umumnya akan memberikan hasil pada tampilan *web browser*, tetapi prosesnya secara keseluruhan dijalankan di *server*. (Inayah, dkk, 2015 : 5).

II.9. MySQL

MySQL adalah *Relation Database Management System* (RDBMS) yang didistribusikan secara gratis di bawah lisensi GPL (*General Public License).* MySQL merupakan turunan dari salah satu konsep utama dalam *database* sejak lama, yaitu SQL (*Structure Query Language)*. SQL merupakan salah satu konsep pengoperasian *database,* terutama sebagai seleksi dan pemasukan data, yang memungkinkan pengoperasian datanya dikerjakan dengan mudah secara otomatis. (Inayah, dkk, 2015 : 5).

II.10. *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.

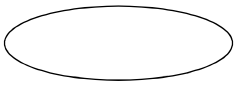
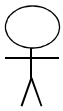
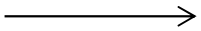
UML merupakan metodologi dalam mengembangkan sistem berorientasi objek dan juga merupakan alat untuk mendukung pengembangan sistem. UML saat ini sangat banyak dipergunakan dalam dunia industri yang merupakan standar bahasa pemodelan umum dalam industri perangkat lunak dan pengembangan sistem. (Gellysa Urva dan Helmi Fauzi Siregar, 2015, Hal : 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.1 dibawah ini:

Tabel II.1. 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,

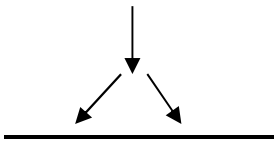
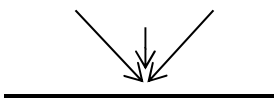
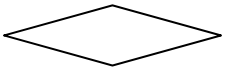
	contohnya adalah pemanggilan sebuah fungsi program.
<----------	*Extend*, merupakan perluasan dari *use case* lain jika kondisi atau syarat terpenuhi.

(Sumber : Gellysa Urva dan Helmi Fauzi Siregar; 2015, Hal : 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.2. Simbol *Activity Diagram*

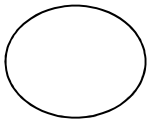
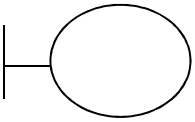
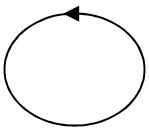
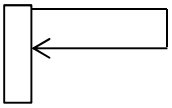
Gambar	Keterangan
	*Start point*, diletakkan pada pojok kiri atas dan merupakan awal aktifitas.
	*End point*, akhir aktifitas.
	*Activites*, menggambarkan suatu proses/kegiatan bisnis.
	*Fork* (Percabangan), digunakan untuk menunjukkan kegiatan yang dilakukan secara parallel atau untuk menggabungkan 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, Hal : 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.3 dibawah ini :

Tabel II.3. 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, Hal : 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.4 dibawah ini :

Tabel II.4. *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, Hal : 95)