BAB II

TINJAUAN PUSTAKA

II.1. Pengertian Aplikasi

Program aplikasi adalah program siap pakai atau program yang direka untuk melaksanakan suatu fungsi bagi pengguna atau aplikasi yang lain. Aplikasi juga diartikan sebagai penggunaan atau penerapan suatu konsep yang menjadi pokok pembahasan atau sebagai program komputer yang dibuat untuk menolong manusia dalam melaksanakan tugas tertentu. Aplikasi software yang dirancang untuk penggunaan praktisi khusus, klasifikasi luas ini dapat dibagi menjadi 2 (dua) yaitu:

a. Aplikasi *software* spesialis, program dengan dokumentasi tergabung yang dirancang untuk menjalankan tugas tertentu.
b. Aplikasi paket, suatu program dengan dokumentasi tergabung yang dirancang untuk jenis masalah tertentu (Rahmatillah ; 2011: 3).

II.2. Pengertian Perancangan

Perancangan sistem adalah penentuan proses data yang diperlukan oleh sistem baru. Jika sistem itu berbasis komputer, rancangan dapat menyertakan spesifikasi jenis peralatan yang digunakan. tahap-tahap perancangan sistem informasi adalah sebagai berikut :

a. Menyiapkan rancangan sistem yang terinci
b. Mengidentifikasi berbagai alternatif konfigurasi sistem

c. Mengevaluasi berbagai alternatif konfigurasi sistem

d. Memilih konfigurasi terbaik e. Menyiapkan usulan penerapan

e. Menyetujui atau menolak penerapan sistem

System design is the specification or construction of a technical , computer based solution for the business requirements identified in a system analysis". Yang diterjemahkan sebagai berikut: "Perancangan sistem adalah spesifikasi atau perwujudan dari solusi teknis berbasiskan komputer untuk kebutuhan bisnis yang diidentifikasikan di sistem analisis". Menurut Romney dan Steinbart (2006, p792): "System design is the process of preparing detail specifications for development of a new information system". Yang diterjemahkan sebagai berikut: "Perancangan sistem adalah suatu proses detail spesifikasi untuk mengembangkan sebuah sistem informasi yang baru". Berdasarkan pendapat-pendapat diatas, dapat disimpulkan bahwa perancangan sistem adalah proses mengimplementasikan hasil-hasil dari analisis sistem ke dalam suatu rancangan sistem yang baru (Henny Hendarti ; 2009 : 158).

Model perancangan sesungguhnya adalah modal objek yang mendeskripsikan realisasi fisik *use case* dengan cara berfokus pada bagaimana spesifikasi-spesifikasi kebutuhan fungsional dan *non*-fungsional, bersama dengan batasan-batasan lain yang berhubungan dengan lingkungan implemenatasi, memiliki imbas langsung pada pertimbangan-pertimbangan pada aktivitas-aktivitas yang dilakukan pada tahap implementasi. Tambahannya, model perancangan sesungguhnya secara langsung bertindak sebagai abstraksi implementasi sistem/perangkat lunak dan dengan sendirinya model perancangan

suatu saat nanti akan menjadi asupan bagi aktivitas-aktivitas selanjutnya yang kelak akan terdefinisi pada tahap implementasi (Adi Nugroho ; 2010 : 212).

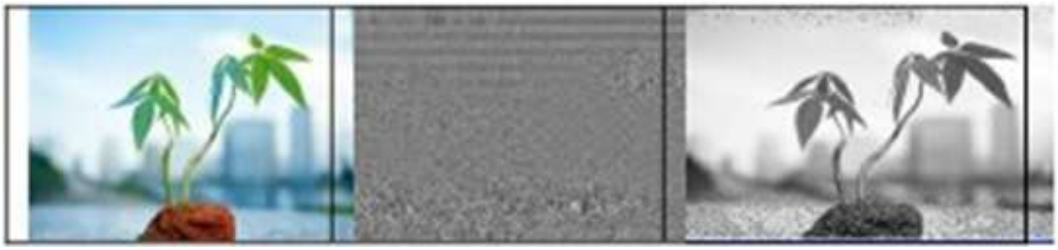
II.3. Kriptografi

Kriptografi berasal dari bahasa Yunani, “kryptós” yang berarti tersembunyi dan “gráphein” yang berarti tulisan. Sehingga kata kriptografi dapat diartikan menjadi “tulisan tersembunyi”. Menurut Request for Comments (RFC), Kriptografi adalah ilmu matematika yang berhubungan dengan transformasi data agar arti dari data tersebut menjadi sulit untuk dipahami (untuk menyembunyikan maknanya), mencegahnya dari perubahan tanpa izin, atau mencegahnya dari penggunaan yang tidak sah. Jika transformasinya dapat dikembalikan, kriptografi juga dapat diartikan sebagai proses mengubah kembali data yang terenkripsi menjadi bentuk yang mudah dipahami. Sehingga, kriptografi juga dapat diartikan sebagai proses untuk melindungi data dalam arti yang luas (Anandia Zelvina ; 2012 : 59).

II.4. Enkripsi File Gambar

Enkripsi gambar adalah cara untuk menyembunyikan informasi gambar asli dan dibuat gambar tersebut tidak tampak seperti gambar aslinya. Menentukan gambar apa yang akan disembunyikan informasinya beserta kunci. Kemudian langkah selanjutnya melakukan proses enkripsi, bagaimana cara memetakan tiap piksel dari gambar tersebut, lalu melakukan permutasi sederhana dari lokasi piksel serta transformasi dari nilai skala abu- abu melalui operasi. Sebuah citra diubah ke bentuk digital agar dapat disimpan dalam memori komputer atau media lain.

Proses mengubah citra ke bentuk digital bisa dilakukan dengan beberapa perangkat, misalnya scanner, kamera digital, dan handycam. Ketika sebuah citra sudah diubah ke dalam bentuk digital (selanjutnya disebut citra digital), bermacam-macam proses pengolahan citra dapat diperlakukan terhadap citra tersebut.Image processing atau sering disebut dengan pengolahan citra digital merupakan suatu proses dari gambar asli menjadi gambar lain yang sesuai dengan keinginan kita. Misal suatu gambar yang kita dapatkan terlalu gelap maka dengan image processing gambar tersebut bisa kita proses sehingga mendapat gambar yang jelas (Tri Hariyono Reiza Hafidz ; 2015 : 1-2).



Gambar II.1. Citra Hasil Enkripsi
(Sumber : Tri Hariyono Reiza Hafidz ; 2015 : 1-2)

II.5. Algoritma Triple Des

3DES (Triple Data Encryption Standard) merupakan suatu algoritma pengembangan dari algoritma DES (Data Encryption Standard). Pada dasarnya algoritma yang digunakan sama, hanya pada 3DES dikembangkan dengan melakukan enkripsi dengan implementasi algoritma DES sebanyak tiga kali. 3DES memiliki tiga buah kunci yang berukuran 168-bit (tiga kali kunci 56-bit dari DES). Pada algoritma 3DES dibagi menjadi tiga tahap, setiap tahapnya merupakan implementasi dari algoritma DES. Tahap pertama, plainteks yang diinputkan dioperasikan dengan kunci eksternal pertama (K1) dan melakukan

proses enkripsi dengan menggunakan algoritma DES. Sehingga menghasilkan pra-cipherteks pertama. Tahap kedua, pra-cipherteks pertama yang dihasilkan pada tahap pertama, kemudian dioperasikan dengan kunci eksternal kedua (K2) dan melakukan proses enkripsi atau proses dekripsi (tergantung cara pengenkripsian yang digunakan) dengan menggunakan algoritma DES. Sehingga menghasilkan prs-cipherteks kedua. Tahap terakhir, pra-cipherteks kedua yang dihasilkan pada tahap kedua, dioperasikan dengan kunci eksternal ketiga (K3) dan melakukan proses enkripsi dengan menggunakan algoritma DES, sehingga menghasilkan cipherteks (Akik Hidayat ; 2014 : 5)

II.6. Pengertian Java

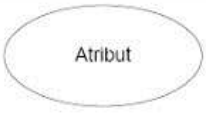
Java adalah bahasa pemrograman yang dapat dijalankan di berbagai jenis komputer dan berbagai sistem operasi termasuk telepon genggam. Java dikembangkan oleh *Sun Microsystem* dan dirilis tahun 1995. Java merupakan suatu teknologi perangkat lunak yang digolongkan *multi platform*. Selain itu, Java juga merupakan suatu *platform* yang memiliki *virtual machine* dan *library* yang diperlukan untuk menulis dan menjalankan suatu program.

Bahasa pemrograman java pertama lahir dari *The Green Project*, yang berjalan selama 18 bulan, dari awal tahun 1991 hingga musim panas 1992. Proyek tersebut belum menggunakan *versi* yang dinamakan Oak. Proyek ini dimotori oleh Patrick Naughton, Mike Sheridan, James Gosling dan Bill Joy, serta Sembilan pemrograman lainnya dari *Sun Microsystem.* Salah satu hasil proyek ini adalah mascot Duke yang dibuat oleh Joe Palrang (Wahana Komputer ; 2010 : 1).

II.7. *Entity Relationship Diagram* (ERD)

Entity Relationship Diagram atau ERD adalah alat pemodelan data utama dan akan membantu mengorganisasi data dalam suatu proyek ke dalam entitas-entitas dan menentukan hubungan antarentitas. Proses memungkinkan analis menghasilkan struktur basisdata yang baik sehingga data dapat disimpan dan diambil secara efisien (Janner Simarmata ; 2010 : 67).

Tabel II.1. Simbol ERD

Notasi	Keterangan
	**Entitas**, adalah suatu objek yang dapat diidentifikasi dalam lingkungan pemakai.
	**Relasi**, menunjukkan adanya hubungan di antara sejumlah entitas yang berbeda.
	**Atribut**, berfungsi mendeskripsikan karakter entitas (atribut yg berfungsi sebagai key diberi garis bawah)
	Garis, sebagai penghubung antara relasi dengan entitas, relasi dan entitas dengan atribut.

(Sumber : Janner Simarmata ; 2010 : 67)

II.8. Kamus Data

Kamus data (data *dictionary*) mencakup definisi-definisi dari data yang disimpan di dalam basis data dan dikendalikan oleh sistem manajemen basis data. Figur 6.5 menunjukkan hanya satu tabel dalam basis data jadwal. Struktur basis

data yang dimuat dalam kamus data adalah kumpulan dari seluruh definisi *field*, definisi tabel, relasi tabel, dan hal-hal lainnya. Nama *field* data, jenis data (seperti teks atau angka atau tanggal), nilai-nilai yang valid untuk data, dan karakteristik-karakteristik lainnya akan disimpan dalam kamus data. Perubahan-perubahan pada struktur data hanya dilakukan satu kali di dalam kamus data, program-program palikasi yang mempergunakan data tidak akan ikut terpengaruh (Raymond McLeod ; 2008 : 171).

II.9. Teknik Normalisasi

Proses normalisasi menyediakan cara sistematis untuk meminimalkan terjadinya kerangkapan data di antara relasi dalam perancangan logikal basis data. Format normalisasi terdiri dari lima bentuk, yaitu:

II.9.1.Bentuk-bentuk Normalisasi

a. Bentuk normal tahap pertama (1" Normal Form)

Suatu tabel dikatakan sudah 1NF jika telah memenuhi ketentuan sebagai berikut:

- Tidak ada atribut mempunyai nilai berulang atau nilai *array*
- Tidak mempunyai baris yang rangkap

Bentuk unnormal mengijinkan nilai-nilai pada suatu atribut dapat berulang.

b. Bentuk normal tahap kedua (2^{nd} normal form)

Relasi dapat dikatakan format normal kedua jika sudah dalam format normal pertama dan diikuti kondisi sebagai berikut:

- *Key* terdiri dari atribut tunggal
- Setiap atribut *non-key* ketergantungan fungsional pada semua *key* atau tidak terjadinya ketergantungan pada *key composite*.

Misalnya tabel UNIV berada dalam normal kedua dengan mengasumsikan DNO sebagai *key*, kecuali CRSE. Jika ditentukan CNO dan SECNO sebagai *key* composite, atribut *nonkey* CNAME tergantung hanya pada CNO, bukan pada SECNO, sehingga CNAME tidak secara ketergantungan fungsional penuh terhadap *key* (CNO, SECNO).

c. Bentuk normal tahap ketiga (3rd normal form)

Relasi dikatakan format normal ketiga jika sudah dalam format normal kedua dan tidak ada ketergantungan transitif di antara atribut. Misalnya tabel STUDNT mempunyai atribut SSNO sebagai *key* (2NF). Ketergantungan transitif terjadi di antara DNO dan COLREG. Saat DNO determinan COLREG tanpa melibatkan *key* SSNO. Contohnya, DNO=’CS’ termasuk COLREG=’Arts/Sc.’ tidak tergantung oleh atribut SSNO, sehingga STUDNT belum termasuk 3NF. Yang menjadi catatan, ketergantungan transitif tidak akan terjadi jika ada ketergantungan fungsional di antara atribut-atribut *non-key* yang melibatkan *key*.

d. Boyce Code Normal Form (BCNF)

BCNF menentukan setiap determinan adalah kunci kandidat (*candidate key*). Misalnya UNIV mempunyai dua determinan yaitu

DNO dan DNAME yang merupakan *kunci kandidat* sehingga termasuk ke dalam BCNF. Di lain pihak CRSLST dalam 3NF tetapi tidak dalam BCNF. Atribut komposisinya (CNO, SECNO, SID, OFRNG) sebagai kunci-kunci kandidat dan tidak ada ketergantungan transitif, sehingga CRSLST termasuk ke dalam 3NF. Namun atribut CNO adalah determinan saat SECNO tergantung penuh secara fungsional terhadap CNO, walaupun CNO bukan kunci kandidat, sehingga CRSLST belum termasuk BCNF.

e. Bentuk Normal Tahap Keempat dan Kelima

Bentuk ini adalah bentuk normal ketiga atau BCNF dengan nilai atribut tidak tergantung pada nilai banyak (*multivalue dependency*). Konsep pada bentuk ini adalah ketergantungan pada gabungan beberapa atribut (*join dependency*) (Haidar Dzacko ; 2007 : 12).

II.10. UML (*Unified Modeling Language*)

Menurut Windu Gata (2013 : 4) 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

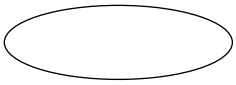
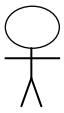
bahasa pemodelan umum dalam industri perangkat lunak dan pengembangan sistem.

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

- *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, yaitu :

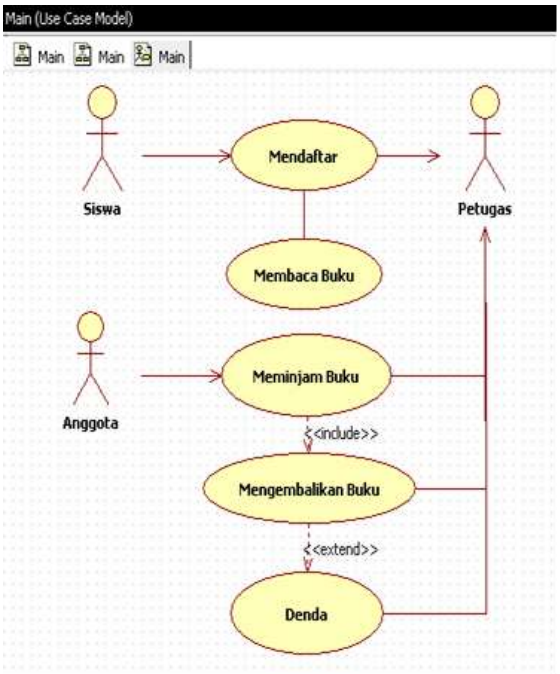
Tabel II.2. Simbol *Use Case*

Gambar	Keterangan
	*Use case* menggambarkan fungsionalitas yang disediakan sistem sebagai unit-unit yang bertukan 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 : Windu Gata ; 2013 : 4)

Contoh dari pembuatan *use case diagram* dapat dilihat pada gambar II.2 berikut :



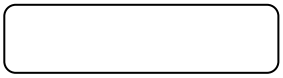
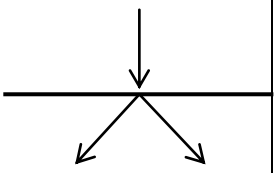
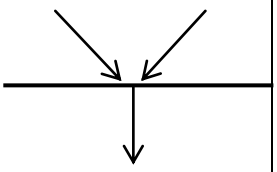
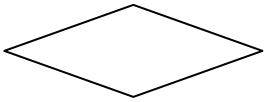
Gambar. II.2. *Use Case Diagaram*

(Sumber : Windu Gata ; 2013 : 4)

- Diagram Aktivitas (*Activity Diagram*)

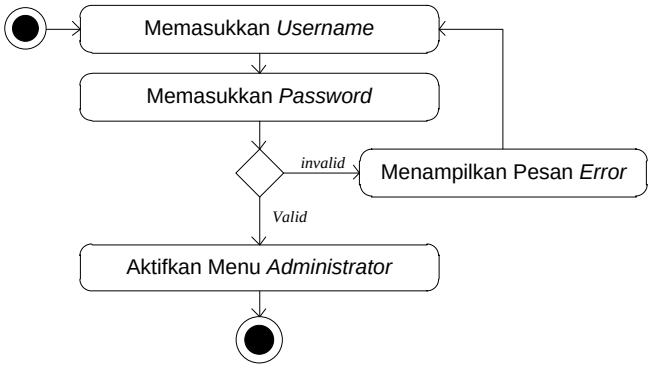
Activity Diagram menggambarkan *workflow* (aliran kerja) atau aktivitas dari sebuah sistem atau proses bisnis. Simbol-simbol yang digunakan dalam *activity diagram*, yaitu :

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

(Sumber : Windu Gata ; 2013 : 6)

Contoh dari pembuatan *activity diagram* dapat dilihat pada gambar II.3 berikut :

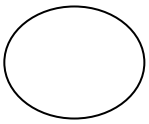
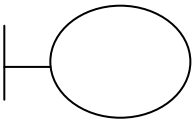
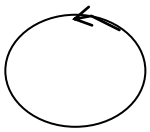


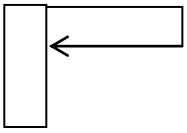
Gambar. II.3. *Activity Diagaram*
(Sumber : Windu Gata ; 2013 : 6)

- 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*, yaitu :

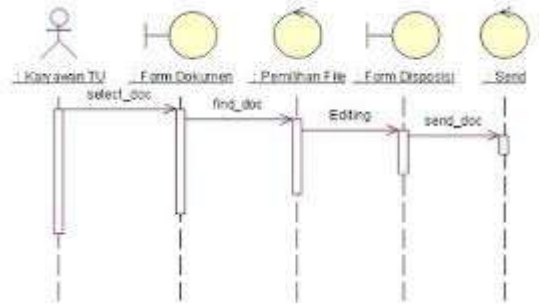
Tabel II.4. Simbol *Sequence Diagram*

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 : Windu Gata ; 2013 : 7)

Contoh dari pembuatan *sequence diagram* dapat dilihat pada gambar II.4 berikut :



Gambar. II.4. *Sequence Diagaram*
(Sumber : Windu Gata ; 2013 : 7)

- *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.

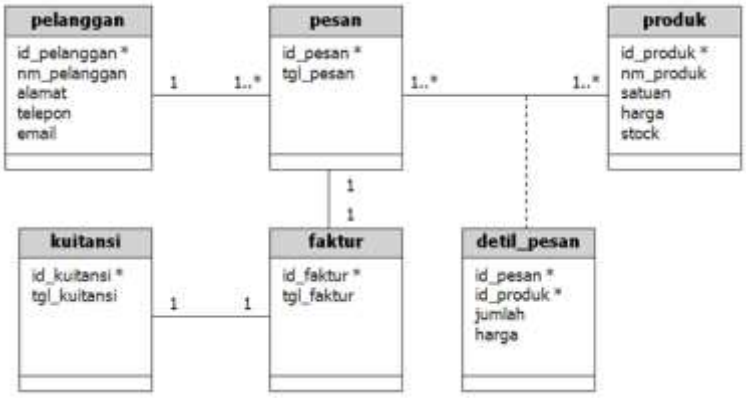
Class diagram juga menunjukkan atribut-atribut dan operasi-operasi dari sebuah kelas dan *constraint* yang berhubungan dengan objek yang dikoneksikan. *Class diagram* secara khas meliputi: Kelas (*Class*), Relasi, *Associations*, *Generalization* dan *Aggregation*, Atribut (*Attributes*), Operasi (*Operations/Method*), *Visibility*, tingkat akses objek eksternal kepada suatu operasi atau atribut. Hubungan antar kelas mempunyai keterangan yang disebut dengan *multiplicity* atau kardinaliti.

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 : Windu Gata ; 2013 : 8)

Contoh dari pembuatan *use case diagram* dapat dilihat pada gambar II.5 berikut :



Gambar. II.5. *Class Diagaram*
(Sumber : Windu Gata ; 2013 : 8)