BAB II

TINJAUAN PUSTAKA

II.1. Penelitian Terdahulu

Menurut Kurniadi (2015) pada penelitian dengan judul penelitian "Penerapan Algoritma Rc4 Untuk Enkripsi Keamanan Data". Keamanan data komputer dapat memberikan sebuah perlindungan terhadap data. teknik perlindungan terhadap hardisk atau flashdisk sehingga data mendapat perlindungan yang efektif. Kriptografi merupakan ilmu dan seni untuk mengamankan pesan. Karya tulis ini membahas tentang ilmu kriptografi, merupakan ilmu pengetahuan yang menggunakan persamaan matemtis untuk melakukan proses enkripsi dan dekripsi data. Enkripsi merupakan proses untuk mengubah plaintek (bisa dimengerti) menjadi chiperteks(tidak bisa dimengerti) dan dekripsi merupakan kebalikan dari enkripsi, yaitu merubah chiperteks menjadi plainteks, yang kemudian akan diimplementasikan ke sebuah bahasa pemrograman yaitu Visual studio 2008 untuk dijadikan sebagai sebuah tools yang berguna untuk mengamakan data. Pada Implementasi kriptografi, Enkripsi dan Dekripsi. Keuntungan dari enkripsi ini adalah keamanannya dan kecepatan dalam melakukan enkripsi dan dekripsi. Adapun algoritma yang digunakan dalam Enkripsi ini adalah algoritma Rc4, yaitu sebuah algoritma yang digunakan untuk melakukan pengacakan pesan dan password.

Penelitian yang dilakukan oleh Rinci Kembanghapsari, S.Si (2012) dengan judul Aplikasi Sistem Pengaman Data Dengan Metode Enkripsi Menggunakan

Algoritma Rc4. Keamanan dan kerahasian data pada jaringan komputer saat ini menjadi isu yang sangat penting dan terus berkembang. Beberapa kasus menyangkut keamanan jaringan komputer saat ini menjadi suatu pekerjaan yang membutuhkan biaya penanganan dan pengamanan yang sedemikian besar. Sistem-sistem vital, seperti sistem pertahan, sistem perbankan, sistem bandara udara dan sistem-sistem yang lain setingkat itu, membutuhkan tingkat keamanan yang sedemikian tinggi. Hal ini lebih di sebabkan karena kemajuan bidang jaringan komputer dengan konsep *open system*-nya sehingga siapapun, di mananpun dan kapanpun, mempunyai kesempatan untuk mengakses kawasan-kawasan vital tersebut. Untuk menjaga keamanan dan kerahasiaan pesan, data, atau informasi dalam suatu jaringan komputer maka di perlukan beberapa enkripsi guna membuat pesan, data, atau informasi agar tidak dapat di baca atau di mengerti oleh sembarang orang, kecuali untuk penerima yang berhak.Hal tersebut di atas yang mendasari perancangan suatu " Aplikasi Sistem Pengaman Data Dengan Metode Enkripsi Menggunakan Algoritma Rc4 ".

II.2. Landasan Teori

Untuk mendukung keberhasilan penelitian ini, penyusun melakukan pendekatan teoritis melalui beberapa literatur yang berhubungan dengan penelitian yang dilakukan. Beberapa tinjauan pustaka pada penelitian ini yaitu:

II.2.1. 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.2.2. Sistem Keamanan

Keamanan sebuah informasi merupakan suatu hal yang harus diperhatikan. Masalah tersebut penting karena jika sebuah informasi dapat di akses oleh orang yang tidak berhak atau tidak bertanggung jawab, maka keakuratan informasi tersebut akan diragukan, bahkan akan menjadi sebuah informasi yang menyesatkan.

Ada banyak cara mengamankan data atau informasi pada sebuah sistem. Pada umumnya pengamanan data dapat dikategorikan menjadi dua jenis, yaitu : penecegahan (*presentif*) dan pengobatan (*recovery*). Pencegahan dilakukan supaya data tidak rusak, hilang dan dicuri, sementara pengobatan dilakukan apabila data sudah terkena virus, sistem terkena worm, dan lubang keamanan sudah *diexploitasi*. Sistem keamanan informasi (*information security*) memiliki empat tujuan yang sangat mendasar adalah :

1. Kerahasiaan (*Confidentiality*). Informasi pada sistem komputer terjamin kerahasiaannya, hanya dapat diakses oleh pihak-pihak yang diotorisasi, keutuhan serta konsistensi data pada sistem tersebut tetap terjaga. Sehingga upaya orang-orang yang ingin mencuri informasi tersebut akan sia-sia.
2. Ketersediaan (*Availability*). Menjamin pengguna yang sah untuk selalu dapat mengakses informasi dan sumberdaya yang diotorisasi. Untuk memastikan bahwa orang-orang yang memang berhak untuk mengakses informasi yang memang menjadi haknya.
3. Integritas (*Integrity*) Menjamin konsistensi dan menjamin data tersebut sesuai dengan aslinya, sehingga upaya orang lain yang berusaha merubah data akan segera dapat diketahui.
4. Penggunaan yang sah (*Legitimate Use*). Menjamin kepastian bahwa sumberdaya tidak dapat digunakan oleh orang yang tidak berhak (Paryati ; 2012 : 379).

II.2.3. *Google Docs*

Google Documents berasal dari dua produk terpisah, *Writely* dan *Google Spreadsheets. Writely* adalah berbasis web pengolah kata dibuat oleh perusahaan perangkat lunak *Upstartle* dan diluncurkan pada Agustus 2005. Spreadsheet, diluncurkan sebagai Google Labs Spreadsheets pada tanggal 6 Juni 2006, berasal dari akuisisi produk XL2Web oleh 2Web Teknologi . Fitur asli *Writely* sudah termasuk teks kolaboratif mengedit suite dan kontrol akses. Menu, shortcut keyboard, dan kotak dialog yang mirip dengan apa yang pengguna dapat

mengharapkan dalam pengolah kata desktop seperti *Microsoft Word* atau *OpenOffice.org Writer.*

Google Documents Viewer adalah penampil *embeddable* berbasis browser yang hanya membutuhkan URL untuk file yang tersedia *online*. Ini rapi bypasses kebutuhan bagi pengguna untuk memiliki perangkat lunak yang kompatibel pada mesin mereka untuk orang-orang jenis file dan menampilkan dokumen tepat di browser (Vuji Suprihatin ; 2012 : 2)

II.2.4. Algoritma RC4

Algoritma RC4 mengenkripsi dengan mengombinasikannya dengan plainteks dengan menggunakan bit-wise Xor (Exclusive-or). RC4 menggunakan panjang kunci dari 1 sampai 256 byte yang digunakan untuk menginisialisasikan tabel sepanjang 256 byte. Tabel ini digunakan untuk generasi yang berikut dari *pseudo random* yang menggunakan XOR dengan plaintext untuk menghasilkan *ciphertext*. Masing - masing elemen dalam tabel saling ditukarkan minimal sekali. Proses dekripsinya dilakukan dengan cara yang sama (karena Xor merupakan fungsi simetrik) (Okie Setiawan ; 2014 : 113).

Untuk menghasilkan *keystream*, *cipher* menggunakan *state* internal yang meliputi dua bagian :

1. Tahap *key scheduling* dimana *state automaton* diberi nilai awal berdasar kan kunci enkripsi. State yang diberi nilai awal berupa *array* yang merepresentasikan suatu permutasi dengan 256 elemen, jadi hasil dari algoritma KSA adalah permutasi awal. *Array* yang mempunyai 256 elemen

ini (dengan indeks 0 sampai dengan 255) dinamakan S. Berikut adalah algoritma KSA dalam bentuk *pseudo-code* dimana *key* adalah kunci enkripsi dan *keylength* adalah besar kunci enkripsi dalam *bytes* (untuk kunci 128 bit, keylength = 16).

```
for i = 0 to 255
    S [i]     := i
j   := 0
for  i  =  0 to 255
    j   :=  (j + S[i] + key [i mod keylength]) mod 256
    swap(S[i] ,S[j])
```

2. Tahap *pseudo-random generation* dimana state automaton beroperasi dan outputnya menghasilkan *keystream*. Setiap putaran, bagian *keystream* sebesar 1 *byte* (dengan nilai antara 0 sampai dengan 255) dioutput oleh PRGA berdasarkan *state* S. Berikut adalah algoritma PRGA dalam bentuk *pseudo-code*

```
i   :=  0
j   :=  0
loop
    i   :=  (i  +  1)  mod 256
    j   :=  (j  +  S[i])  mod  256
    swap (S[i] ,S[j])  mod  256
    output  S[ (S[i]  +  S[j] )  mod  256]
```

Setelah terbentuk *keystream*, kemudian *keystream* tersebut dimasukkan dalam operasi XOR dengan plaintext yang ada, dengan sebelumnya pesan dipotong-potong terlebih dahulu menjadi *byte-byte* (Okie Setiawan ; 2014 : 113).

Berikut adalah implementasi algoritma Rc4 dengan mode 4 byte (untuk lebih menyederhanakan dalam perhitungan manual) serta untuk kebutuhan sistem

yang sangat terbatas. S-Box dengan panjang 4 byte, dengan S[0]=0, S[1]=1, S[2]=2 dan S[3]=3 sehingga array S menjadi:

0 1 2 3

Inisialisasi 4 byte kunci array, K. Misalkan kunci Ulang kunci sampai memenuhi seluruh adalah 2 5 7 3, sehingga array K berisi 2 5 7 3 dan mencoba untuk mengenkripsikan kata HALO.

Inisialisasi i dan j dengan 0 kemudian dilakukan KSA agar tercipta state-array yang acak. Penjelasan iterasi lebih lanjut dapat dijelaskan sebagai berikut:

Iterasi 1

i = 0

j = (0 + S[0] + K [0 mod 4]) mod 4

= (0 + 0 + 2) mod 4 = 2

Swap (S[0],S[2])

Hasil Array S

2 1 0 3

Iterasi 2

i = 1

j = (2 + S[1] + K [1 mod 4]) mod 4

= (2 + 1 + 5) mod 4 =0

Swap (S[1],S[0])

Hasil Array S

1 2 0 3

Iterasi 3

i = 2

j = (0 + S[2] + K [2 mod 4]) mod 4

= (0 + 0 + 7) mod 4 = 3

Swap (S[2],S[3])

Hasil

1 2 3 0

Iterasi 4

i = 3

j = (3 + S[3] + K [3 mod 4]) mod 4

= (3 + 0 + 3) mod 4 = 2

Swap (S[3],S[2])

Hasil Array S

1 2 0 3

Setelah melakukan KSA, akan dilakukan PRGA. PRGA akan dilakukan sebanyak 4 kali dikarenakan plainteks yang akan dienkripsi berjumlah 4 karakter. Hal ini disebakan karena dibutuhkan 1 kunci dan 1 kali pengoperasian XOR untuk tiap tiap karakter pada plainteks. Berikut adalah tahapan penghasilan kunci enkripsi dengan PRGA.

Array S

1 2 0 3

Inisialisasi

i = 0

j = 0

Iterasi 1

i = (0 + 1) mod 4 = 1

j = (0 + S[1]) mod 4 = (0+ 2) mod

4 = 2

swap (S[1],S[2])

1 0 2 3

K1 = S[(S[1]+S[2]) mod 4] = S[2

mod 4] = 2

K1 = 00000010

Iterasi 2

i = (1 + 1) mod 4 = 2

j = (2 + S[2]) mod 4 = (2+ 2) mod

4 = 0

swap (S[2],S[0])

2 0 1 3

K2 = S[(S[2]+S[0]) mod 4] = S[3

mod 4] = 3

K2 = 00000011

Iterasi 3

i = (2 + 1) mod 4 = 3

j = (0 + S[3]) mod 4 = (0+ 3) mod

4 = 3

swap (S[3],S[3])

1 0 2 3

K3 = S[(S[3]+S[3]) mod 4] = S[6

mod 4] = 2

K3 = 00000010

Iterasi 4

i = (3 + 1) mod 4 = 0

j = (3 + S[0]) mod 4 = (3+ 1) mod

4 = 0

swap (S[0],S[0])

1 0 2 3

K1 = S[(S[0]+S[0]) mod 4] = S[2

mod 4] = 2

K4 = 00000010

Setelah menemukan kunci untuk tiap karakter, makadilakukan operasi XOR antara karakter pada plaintext dengan kunci yang dihasilkan. Berikut adalah tabel ASCII untuk tiap-tiap karakter pada plaintks yang digunakan.

Huruf	Kode ASCII (Binary 8 bit)
H	01001000
A	01000001
L	01001100
O	01001111

Berikut adalah proses pengXORan dari plainteks dengan key yang telah didapat:

H A L O : 01001000 01000001 01001100 01001111

Key : 00000010 00000011 00000010 00000010

Cipherteks : 01001010 01000010 01001110 01001101

Algoritma RC4 (Dekripsi)

Proses dekripsi *ciphertext* menggunakan algoritma Rc4 ini sama untuk proses *key-schedule*-nya. Untuk mendapatkan *plaintext, ciphertext* yang diperoleh di XORkan dengan *pseudo random byte* yang didapat sebelumnya. Maka hasilnya adalah plainteks atau teks asli.

Pesan dikirim dalam bentuk cipherteks sehingga setelah sampai di penerima pesan dapat kembali diubah menjadi plainteks dengan meng-XOR-kan dengan kunci yang sama. Pemrosesan pesan setelah sampai pada penerima dapat dilihat pada dibawah ini. (Okie Setiawan ; 2014 : 113).

Proses XOR pseudo random byte dengan cipherteks pada dekripsi yaitu:

Cipherteks : 01001010 01000010 01001110 01001101

pseudo random byte : 00000010 00000011 00000010 00000010

Plainteks : 01001000 01000001 01001100 01001111

H A L O

II.2.5. Pengertian *Android*

Android adalah sistem operasi untuk telepon seluler yang berbasis Linux. Android menyediakan *platform* terbuka bagi para pengembang untuk membuat aplikasi mereka sendiri. Pada awalnya dikembangkan oleh *Android Inc*, sebuah perusahaan pendatang baru yang membuat perangkat lunak untuk ponsel yang kemudian dibeli oleh *Google Inc*. Untuk pengembangannya, dibentuklah *Open*

Handset Alliance (OHA), konsorsium dari 34 perusahaan perangkat keras, perangkat lunak, dan telekomunikasi termasuk Google, HTC, Intel, Motorola, Qualcomm, T-Mobile, dan Nvidia (M. Ichwan ; 2011 : 15).

II.2.6. 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.

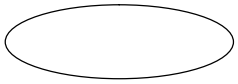
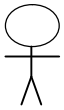
UML merupakan metodologi dalam mengembangkan sistem berorientasi objek dan juga merupakan alat untuk mendukung pengembangan sistem. UML saat ini sangat banyak dipergunakan dalam dunia industri yang merupakan standar bahasa pemodelan umum dalam industri perangkat lunak dan pengembangan sistem.

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

- *Use case* Diagram

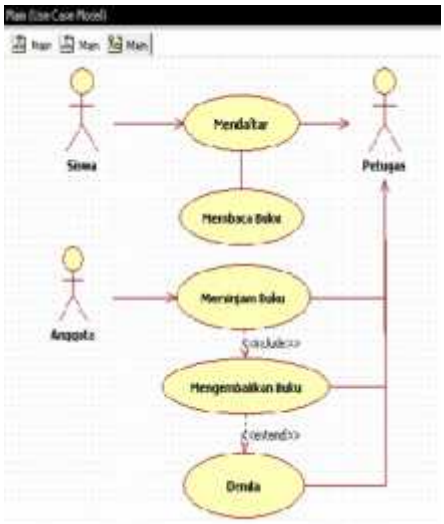
Use case diagram merupakan pemodelan untuk kelakukan (*behavior*) sistem informasi yang akan dibuat. *Use case* mendeskripsikan sebuah interaksi antara satu atau lebih aktor dengan sistem informasi yang akan dibuat. Dapat dikatakan *use case* digunakan untuk mengetahui fungsi apa saja yang ada di dalam sistem informasi dan siapa saja yang berhak menggunakan fungsi-fungsi tersebut. Simbol-simbol yang digunakan dalam *use case* diagram, yaitu :

Tabel II.1. 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 :

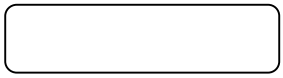
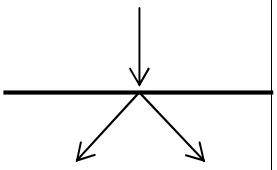


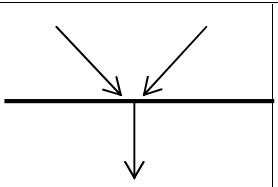
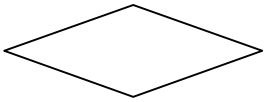
Gambar. II.1. ***Use Case Diagaram***
(Sumber : Windu Gata ; 2013 : 4)

- 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 :

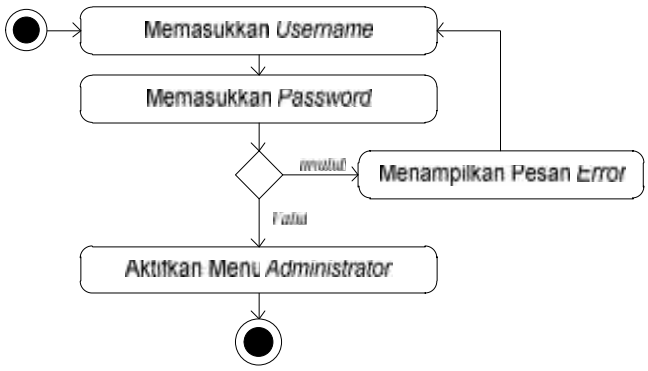
Tabel II.2. Simbol *Activity Diagram*

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 : Windu Gata ; 2013 : 6)

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

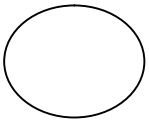
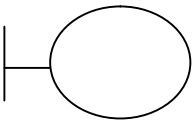
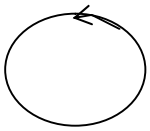
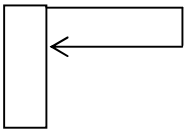


Gambar. II.2. *Activity Diagaram*
(Sumber : Windu Gata ; 2013 : 6)

- Diagram Urutan (*Sequence Diagram*)

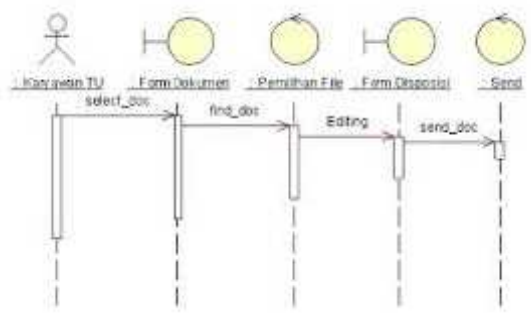
Sequence diagram menggambarkan kelakuan objek pada *use case* dengan mendeskripsikan waktu hidup objek dan pesan yang dikirimkan dan diterima antar objek. Simbol-simbol yang digunakan dalam *sequence diagram*, yaitu :

Tabel II.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 : Windu Gata ; 2013 : 7)

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



Gambar. II.3. *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.

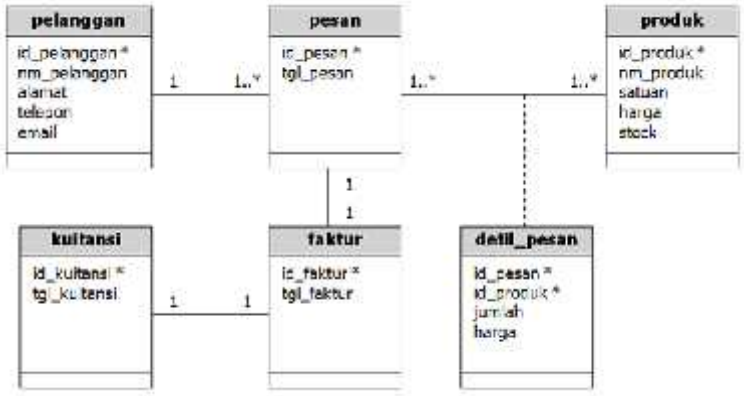
Class diagram juga menunjukkan atribut-atribut dan operasi-operasi dari sebuah kelas dan *constraint* yang berhubungan dengan objek yang dikoneksikan. *Class diagram* secara khas meliputi: Kelas (*Class*), Relasi, *Associations*, *Generalization* dan *Aggregation*, Atribut (*Attributes*), Operasi (*Operations/Method*),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

Tabel II.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 : 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)