BAB II

TINJAUAN PUSTAKA

II.1. Penelitian Terdahulu

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 :

1. Pada penelitian yang dilakukan oleh Hengky Mulyono (2013) dengan judul "Implementasi Algoritma One Time Pad Pada Penyimpanan Data Berbasis Web". Salah satu hal yang perlu diperhatikan dalam menjaga keamanan sebuah sistem adalah proses autentikasi. Proses ini dilakukan untuk memastikan bahwa user yang mengakses data maupun informasi pada sistem tersebut adalah user yang memiliki wewenang. Ada beberapa metode untuk melakukan autentikasi, salah satunya dengan menggunakan teknik penyandian data (kriptografi). Kriptografi diterapkan pada data maupun informasi dengan mengkodekan atau menyembunyikan data aslinya sehingga hanya pihak yang memiliki kunci yang dapat mengakses data atau informasi tersebut. Penelitian ini akan mengimplementasikan algoritma One Time Pad (OTP) untuk melakukan penyandian terhadap data dan informasi yang disimpan. Data atau informasi yang disimpan dalam aplikasi akan berbentuk ciphertext sehingga user akan mendapatkan kunci untuk mengakses data atau informasi tersebut. Pembuatan aplikasi ini diharapkan dapat menjaga

kerahasiaan dan keamanan data dengan baik, dimana pihak yang dapat mengakses data atau informasi yang asli hanya pihak yang memiliki kunci

2. Penelitian yang dilakukan oleh Febryaan Christy (2014) dengan judul penelitian "Implementasi Modifikasi Kriptografi One Time Pad (OTP) untuk Pengamanan Data File". Media elektronik dan digital sudah menjadi tren untuk mengirimkan informasi baik yang bersifat umum maupun rahasia. Untuk menjaga keamanan data yang dikirim diperlukan pengamanan khusus. Pengamanan dalam penelitian ini menggunakan algoritma One Time Pad (OTP) yang dimodifikasi dengan melakukan tiga kali proses dan menggunakan bit shifting pada kunci dan plainteks. Kunci pada proses kedua dan ketiga digenerate agar bertambah sebanyak karakter dalam plainteks. Hasil dari penelitian ini adalah teknik kriptografi yang dapat diaplikasikan pada pengamanan file berbasis teks.

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 Keamanan Transmisi data

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 transmisi data atau informasi pada sebuah sistem. Pada umumnya pengamanan transmisi data dapat dikategorikan menjadi dua jenis, yaitu : penecegahan (*presentif*) dan pengobatan (*recovery*). Pencegahan dilakukan supaya transmisi data tidak rusak, hilang dan dicuri, sementara pengobatan dilakukan apabila transmisi 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 transmisi 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 transmisi data tersebut sesuai dengan aslinya, sehingga upaya orang lain yang berusaha merubah transmisi 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.2.2. Websocket

Di dalam websocket.org, WebSocket adalah teknologi yang dirancang untuk menyederhanakan kompleksitas pada komunikasi bi-directional, full-duplex melalui socket Transmission Control Protocol (TCP) tunggal dimana pesan dapat dikirimkan antara klien dan server. WebSocket menggunakan protocol berbasiskan HTTP. Koneksi WebSocket dibentuk dengan mengubah protocol HTTP menjadi protocol WebSocket ketika melakukan handshake antar klien dan server (Ari Pambudi ; 2013 : 113).

II.2.3. Algoritma One-Time Pad

Dalam kriptografi dikenal beberapa algoritma, diantaranya adalah algoritma One Time Pad (OTP).One Time Pad adalah salah satu metode kriptografi dengan algoritma jenis simetri. Ditemukan pada tahun 1917 oleh Major Yoseph Mouborgne dan Gilbert Vernam pada perang dunia ke dua. Metode ini telah diklaim sebagai satu-satunya algoritma kriptografi sempurna yang tidak dapat dipecahkan. Suatu algoritma dikatakan aman, apabila tidak ada cara untuk menemukan plaintext-nya. Sampai saat ini, hanya algoritma One Time Pad (OTP) yang dinyatakan tidak dapat dipecahkan meskipun diberikan sumber daya yang tidak terbatas. Dalam proses enkripsi One Time Pad, cipherteks diperoleh dengan

melakukan penjumlahan modulo 26 dari satu bit plainteks dengan satu bit kunci, seperti terlihat pada Rumus :

$C_i = (P_i + K_i) \text{ mod} 26$

Dimana :

C_i = cipher teks

P_i = plainteks

K_i = kunci

Sedangkan dalam proses dekripsi, untuk mendapatkan kembali plainteks, diperoleh dengan melakukan penjumlahan modulo 26 dari satu bit cipherteks dengan satu bit kunci (Febryan Christy Winaryo ; 2014 : 4):

$$P_i = (C_i - K_i) \bmod 26$$

II.2.4. Netbeans

NetBeans merupakan salah satu proyek *open source* yang disponsori oleh *Sun Microsystem*. Proyek ini bediri pada tahun 2000 dan telah menghasilkan 2 produk, yaitu NetBeanss IDE dan NetBeans Platform. NetBeans IDE merupakan produk yang digunakan untuk melakukan pemrograman baik menulis kode, meng-*compile*, mencari kesalahan dan mendistribusikan program. Sedangkan NetBeans Platform adalah sebuah modul yang merupakan kerangka awal / pondasi dalam bangun aplikasi desktop yang besar. NetBeans juga menyediakan paket yang lengkap dalam pemrograman dari pemrograman standar (aplikasi *desktop*), pemrograman *enterprise*, dan pemrograman perangkat mobile. Saat ini NetBeans telah mencapai versi 6.8 (Wahana Komputer ; 2010 : 15)

II.2.5. *Javascript*

Javascript adalah bahasa pemrograman yang berjalan di sisi klien (klien web / browser, mis:Internet Explorer). Teknologi *Javascript* dibuat dengan tujuan agar dapat memperingan kerja server serta menambah sifat dinamis dan interaktivitas dari sebuah situs HTML. Penggunaan Javascript terutama untuk hal-hal yang tidak bersifat penting atau kritis, seperti pemeriksaan format input, animasi teks, efek kursor mouse, dan aplikasi-aplikasi ringan seperti kalkulator maupun games.

Teknologi Javascript pertama kali diperkenalkan oleh Netscape sejak Netscape 2.0 yang dapat menjalankan Javascript versi 1.0, kemudian Netscape 3.0 menggunakan Javascript versi 1.1 dan Netscape 4 ke atas menggunakan Javascript versi 1.2. Sedangkan untuk Internet Explorer melakukan implementasi script dengan menggunakan standar tersendiri yaitu VBscript serta Jscript yang kompatibel dengan Javascript, sehingga Javascript dapat berjalan pada IE tetapi VBscript tidak dapat berjalan di Netscape.

Penulisan Javascript pada HTML menggunakan tag <SCRIPT>...</SCRIPT> yang dapat ditempatkan pada area <HEAD> ataupun <BODY>. Penempatan tag <SCRIPT> pada area <HEAD> dimaksudkan agar Javascript dijalankan terlebih dahulu sebelum menampilkan halaman HTML, tetapi ada beberapa Javascript yang menggunakan elemen HTML justru harus ditulis pada area <BODY>. (Ek Kian ; 2010 : 2)

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.

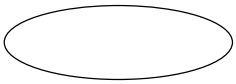
UML merupakan metodologi dalam mengembangkan sistem berorientasi objek dan juga merupakan alat untuk mendukung pengembangan sistem. UML saat ini sangat banyak dipergunakan dalam dunia industri yang merupakan standar bahasa pemodelan umum dalam industri perangkat lunak dan pengembangan sistem.

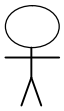
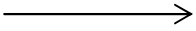
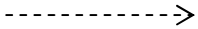
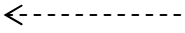
Alat bantu yang digunakan dalam perancangan berorientasi objek berbasiskan UML adalah sebagai berikut :

- *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*

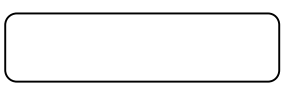
	case.
	Aktor adalah *abstraction* dari orang atau sistem yang lain yang mengaktifkan fungsi dari target sistem. Untuk mengidentifikasikan aktor, harus ditentukan pembagian tenaga kerja dan tugas-tugas yang berkaitan dengan peran pada konteks target sistem. Orang atau sistem bisa muncul dalam beberapa 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.

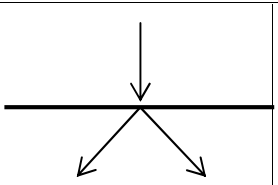
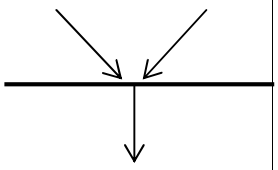
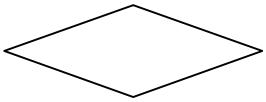
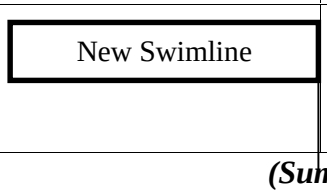
(Sumber : Windu Gata ; 2013 : 4)

- 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.

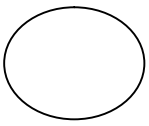
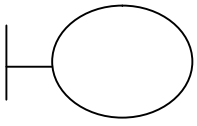
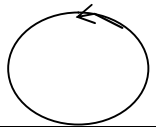
	*Fork* (Percabangan), digunakan untuk menunjukkan kegiatan yang dilakukan secara parallel atau untuk menggabungkan dua kegiatan pararel menjadi satu.
	*Join* (penggabungan) atau rake, digunakan untuk menunjukkan adanya dekomposisi.
	*Decision Points*, menggambarkan pilihan untuk pengambilan keputusan, *true, false.*
	*Swimlane*, pembagian *activity* diagram untuk menunjukkan siapa melakukan apa.

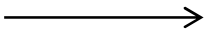
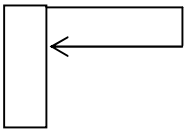
(Sumber : Windu Gata ; 2013 : 6)

- 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)

- *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)