BAB II
TINJAUAN PUSTAKA

II.1. Penelitian Terkait

Untuk mendukung keberhasilan penelitian ini, penulis melakukan pendekatan teoritis melalui beberapa literatur yang berhubungan dengan penelitian yang dilakukan. Beberapa uraian penelitian terdahulu yang menjadi acuan dalam penelitian ini yaitu:

Penelitian terkait yang dilakukan oleh Raja Nasrul Fuad, Haikal Nando Winata, dkk (2017) yang berjudul **"*Aplikasi Keamanan File Audio Wav (Waveform) Dengan Terapan Algoritma Rsa.*** Perangkat lunak (*software*) dapat melakukan penyandian data audio dengan menerapkan algoritma RSA dan struktur data audio WAV. Ukuran berkas audio WAV menjadi bertambah besar setelah dilakukan enkripsi menggunakan algoritma RSA berdasarkan besar kunci yang digunakan.

Penelitian terkait lain yang dilakukan oleh Heri Santoso, M. Fakhriza (2018) dengan judul **"*Perancangan Aplikasi Keamanan File Audio Format Wav (Waveform) Menggunakan Algoritma Rsa*".** Perangkat lunak (*software*) dapat melakukan penyandian data audio dengan menerapkan algoritma RSA dan struktur data audio WAV. Ukuran berkas audio WAV menjadi bertambah besar setelah dilakukan enkripsi menggunakan algoritma RSA berdasarkan besar kunci yang digunakan.

Penelitian terkait lain yang dilakukan oleh Muhammad Arif Damanik

(2017) dengan judul ***"Penerapan Algoritma Triple Transposition Vigenere Cipher Dan Algoritma Rsa Dalam Keamanan Login"*** Masalah keamanan berada diurutan kedua atau bahkan di urutan terakhir dalam daftar hal-hal yang dianggap penting. Apabila mengganggu kinerja sistem, seringkali keamanan dikurangi atau ditiadakan. Jika berbicara mengenai masalah keamanan yang berkaitan dengan penggunaan komputer, maka sulit memisahkannya dengan proses *login*. *Login* bertujuan untuk memberikan layanan keamanan pada sistem. Saat melakukan *login* pengguna akan memasukkan password dimana *password* tersebut bersifat privasi dan rahasia. Oleh karena itu, masalah keamanan menjadi masalah yang sangat penting mengingat internet merupakan jaringan publik yang saling terhubung dalam suatu jaringan dan akan sangat berbahaya jika *password* yang dimasukkan user tersebut tidak dienkripsi sebelum dikirim ke server melalui jaringan. *Triple Transposition Vigènere Cipher* adalah metode enkripsi dengan cara mengulang teknik *Vigènere Cipher* yang setiap *plaintext* nya dilakukan transposisi terlebih dahulu sebanyak tiga kali dengan menggunakan kunci yang tiap kuncinya harus berbeda satu dengan yang lainnya.

Penelitian terkait lain yang dilakukan oleh Prasetyo (2018) dengan judul ***"Perancangan Aplikasi Pengamanan File Teks dengan Skema Hybrid Menggunakan Algoritma Enigma dan Algoritma RSA"*** Perkembangan teknologi informasi yang semakin pesat memberi pengaruh yang besar, Salah satunya adalah kriptografi dimana pesan disamarkan menjadi pesan tersandi. Dalam proses penyandian, penyandian yang digunakan adalah Enigma dan RSA (Rivest Shamir Adleman). Dimana Enigma sebagai proses penyandian simetris pada pesan dan

RSA (*Rivest Shamir Adleman*) sebagai proses penyandian asimetris pada kunci sehingga terjadinya Kriptografi *Hybrid*. Kriptografi *Hybrid* sebenarnya metode yang sangat aman dikarenakan memakai dua jenis algoritma yaitu algoritma simetris dan asimetris, untuk algoritma simetris *Ciphertext* yang dihasilkan akan sama panjang dengan *Plaintext* yang di enkripsi, lain halnya dengan algoritma asimetris yang berfungsi hanya untuk proses enkripsi kunci pada penelitian ini, karena jika algoritma asimetris dipakai untuk enkripsi maka *Ciphertext* yang dihasilkan akan lebih besar jumlah datanya.

Penelitian terkait lain yang dilakukan oleh Iskandar Muda Siregar (2019) dengan judul **"*Penerapan Algoritma Affine Cipher Dan Algoritma Coloumnar Transposition Dalam Keamanan Teks".*** Dengan adanya proses enkripsi dan dekripsi pada keamanan teks dengan menggunakan algoritma *affine cipher* dan *algoritma coloumnar transposition*, teks dapat diamankan dari orang – orang yang tidak berkepentingan. Proses pengaman teks dengan mengkombinasikan algoritma *affine cipher* dan algoritma *coloumnar transposition* bertujuan agar mempersulit analisis sandi. Perancangan aplikasi dengan visual basic studio 2008 bertujuan untuk mengimplementasikan hasil enkripsi dan dekripsi untuk keamanan teks.

Berdasarkan hasil penelitian yang terdahulu, maka dibuatlah kesimpulan untuk merancang sebuah aplikasi untuk menerapkan sebuah algoritma yang belum pernah digunakan sebelumnya untuk melakukan enkripsi dan dekripsi terhadap data *absensi karyawan* yaitu *Algoritma RSA dan Affine Cipher*. Sehingga pada penulisan skripsi ini dibuatlah sebuah judul **"*Penerapan Algoritma RSA dan Affine Cipher***

***dalam Keamanan Data Absensi karyawan*"**. Berdasarkan judul tersebut nantinya akan dihasilkan sebuah aplikasi untuk melakukan enkripsi dan dekripsi terhadap data absensi karyawan.

II.2. Landasan Teori

II.2.1. Aplikasi

Aplikasi adalah satu unit perangkat lunak yang dibuat untuk membantu melayani kebutuhan seseorang untuk melakukan aktivitas. Dengan adanya aplikasi maka kebutuhan akan pelayanan sebuah aktivitas menjadi lebih baik. Dimana setiap pekerjaan dapat dilakukan dengan mudah kalau menggunakan sebuah aplikasi (Agusdi Syafrizal: 2018).

II.2.2. Kriptografi

Kriptografi adalah suatu ilmu yang mempelajari bagaimana cara menjaga agar data atau pesan tetap aman saat dikirimkan, dari pengirim ke penerima tanpa mengalami gangguan dari pihak ketiga. *Kriptografi* adalah ilmu pengetahuan dan seni menjaga *message* agar tetap aman. Tujuan penerapan *kriptografi* adalah untuk membuat sesuatu yang tersembunyi, dapat suatu pesan rahasia berupa teks, suara, gambar dan video. Di dalam sistem *kriptografi* terdapat 5 bagian yaitu

1. *Plaintext* adalah pesan atau data dalam bentuk aslinya teks yang dapat terbaca. *Plaintext* adalah masukan bagi algoritma enkripsi.
2. *Secret Key* adalah masukan bagi algoritma enkripsi merupakan nilai yang bebas terhadap teks asli dan menentukan hasil keluaran algoritma enkripsi.
3. *Chipertext* adalah keluaran algoritma enkripsi. *Ciphertext* dapat dianggap

sebagai pesan tersembunyi yang akan terlihat acak.

4. *Algoritma Enkripsi* memiliki 2 masukan teks asli dan kunci rahasia. *Algoritma enkripsi* melakukan transpormasi terhadap teks asli sehingga menghasilkan teks sandi.
5. *Algoritma Dekripsi* memiliki 2 masukan yaitu teks sandi dan kunci rahasia. *Algoritma dekripsi* memulihkan kembali teks sandi menjadi teks asli bila kunci rahasia *algoritma enkripsi* sama dengan *algoritma dekripsi* (Guntur Tri Wibowo, dkk: 2015).

II.2.3. Metode Algoritma RSA (*Rivest Shamir Adleman*)

RSA merupakan salah satu dari *Public Key Cryptosystem* yang sangat sering digunakan untuk memberikan kerahasiaan terhadap keaslian suatu data digital. Keamanan enkripsi dan dekripsi data model ini terletak pada kesulitan untuk memfaktorkan modulus n yang sangat besar. Dalam kriptografi, RSA adalah algoritma untuk enkripsi kunci publik. Algoritma ini adalah algoritma pertama yang diketahui paling cocok untuk menandai *(signing)* dan untuk enkripsi dan salah satu penemuan besar pertaman dalam kriptografi kunci publik. RSA masih digunakan secara luas dalam protokol-protokol perdagangan elektronik dan dipercayai sangat aman karena diberikan kunci-kunci yang cukup panjang dan penerapan-penerapannya yang sangat mutakhir.

Algoritma pembentukan kunci :

1. Tentukan p dan q bernilai dua bilangan prima besar, acak dan dirahasiakan, $p \neq q$, p dan q memiliki ukuran yang sama.

2. Hitung n = p x q, dan hitung ϕ(n) = (p – 1) x (q – 1), bilangan integer n disebut (RSA) modulus.
3. Tentukan e bilangan prima acak yang memiliki syarat : 1 < e < 1(n), GCD(e, i(n)) = 1, disebut e relatif prima terhadap i(n), bilangan integer n disebut (RSA) enciphering component, sehingga menghasilkan Dd (Ee(m)) = Ee(Dd(c)) ≡ md mod n. (Indra Gunawan, 2018;125).

II.2.4. *Affine cipher*

Affine cipher adalah perluasan dari *caesar cipher*, yang menggalikan plainteks dengan sebuah nilai dan menambahkannya dengan sebuah pergeseran. Secara matematis enkripsi plainteks P menghasilkan cipherteks C dinyatakan dengan fungsi kongruen

C=((a x P) +k)mod 26.....(1)

Dimana 26 adalah jumlah alphabet, persamaan 1 digunakan pada proses enkripsi. Proses dekripsi menggunakan persamaan 2 di bawah ini :

P=a-1(Ci-k) mod 26.....(2)

a adalah bilangan bulat yang harus relatif prima dengan26. Dengan kata lain great common divisior gcd(a,26) harus sama dengan 1. (Rinaldi Munir, 2016).

a. Enkripsi

Encrypt atau enkripsi merupakan sebuah teknik yang dilakukan mengacak data asli menjadi kode rahasia sehingga menyulitkan orang yang tidak berkepentingan untuk mengakses dan mengetahui data yang asli.

b. Dekripsi

Decryption atau dekripsi adalah kebalikan dari enkripsi, dimana berfungsi

untuk mendeskripsikan data yang telah dienkripsi sehingga data yang telah menjadi kode rahasia diubah kembali menjadi data biasa atau aslinya (Arisantoso, dkk; 2017).

Data adalah sesuatu yang belum mempunyai arti bagi penerimanya dan masih memerlukan adanya suatu pengolahan. Data bisa berwujud suatu keadaan, gambar, suara, huruf, angka, matematika, bahasa ataupun simbol-simbol lainnya yang bisa kita gunakan sebagai bahan untuk melihat lingkungan,obyek, kejadian ataupun suatu konsep (Eka Ikusnady; 2017).

II.2.5. *WEB*

Web merupakan salah satu sumber daya internet yang berkembang pesat. Pendistribusian informasi web dilakukan melalui pendekatan hyperlink, yang memungkinkan suatu teks, gambar, ataupun objek yang lain menjadi acuan untuk membuka halaman-halaman yang lain. Melalui pendekatan ini, seseorang dapat memperoleh informasi dengan beranjak dari satu halaman ke halaman lain. (Abdul Kadir, 2017).

II.2.6. HTML

HyperText Markup Language (HTML) adalah sebuah bahasa *mark up* yang digunakan untuk membuat sebuah halaman *web*, menampilkan berbagai informasi di dalam sebuah Penjelajah *web* Internet dan formating *hypertext* sederhana yang ditulis kedalam berkas format ASCII agar dapat menghasilkan tampilan wujud yang terintegerasi. Dengan kata lain, berkas yang dibuat dalam perangkat lunak pengolah kata dan disimpan kedalam format ASCII normal sehingga menjadi *home page*

dengan perintah-perintah HTML. (Abdul Kadir, 2017).

II.2.7. PHP

PHP adalah bahasa yang dirancang secara khusus untuk penggunaan pada *Web*. PHP adalah *tool* untuk pembuatan halaman web dinamis. Pada awalnya PHP merupakan kependekan dari Personal *Home Page* (Situs Personal). PHP pertama kali dibuat oleh Rasmus Lerdorf pada tahun 1995. Pada waktu itu PHP masih bernama FI (*Form Interpreted*), yang wujudnya berupa sekumpulan *script* yang digunakan untuk mengolah data form dari *web*. Saat ini PHP adalah singkatan dari PHP: *Hypertext Preprocessor*, sebuah kepanjangan rekursif, yakni permainan kata dimana kepanjangannya terdiri dari singkatan itu sendiri: PHP: *Hypertext Preprocessor* (Ahmad Lutfi, 2017 ; 105).

II.2.8. *Database*

Database atau biasa disebut basis data merupakan kumpulan data yang saling berhubungan. Data tersebut biasanya terdapat dalam tabel-tabel yang saling berhubungan satu sama lain, dengan menggunakan *field*/kolom pada tiap tabel yang ada (Agus Prayitno dan Yulia Safitri, 2017; 2).

II.2.9 SQL

SQL adalah bahasa standard untuk melakukan berbagai operasi data pada *database*, diantaranya mendefinisikan tabel, menampilkan data dengan kriteria tertentu, menambahkan data hingga menghapus data tertentu. Penggunaan SQL pada beberapa bahasa pemrograman secara umum relatif sama. (Agus Prayitno dan

Yulia Safitri, 2017; 2).

II.2.10. *MySQL*

MySQL adalah salah satu aplikasi DBMS (*Database Management System*) yang sudah sangat banyak digunakan oleh para pemrogram aplikasi *web*. Dalam sistem database tak relasional, semua informasi disimpan pada satu bidang luas, yang kadangkala data di dalamnya sangat sulit dan melelahkan untuk diakses. Tetapi *MySQL* merupakan sebuah sistem database relasional, sehingga dapat mengelompokkan informasi ke dalam tabel-tabel atau grup-grup informasi yang berkaitan. Setiap tabel memuat bidang-bidang yang terpisah, yang mempresentasikan setiap bit informasi. MySQL menggunakan indeks untuk mempercepat proses pencarian terhadap baris informasi tertentu. *MySQL* memerlukan sedikitnya satu indeks pada tiap tabel. Biasanya akan menggunakan suatu *primary key* atau pengenal unik untuk membantu penjejakan data (Ahmad Lutfi, 2017; 106).

II.2.11. UML (*Unified Modeling Language*)

UML (*Unified Modeling Language*) digunakan sebagai suatu cara untuk mengkomunikasikan idenya kepada para pemrogram serta calon pengguna sistem/perangkat lunak. Dengan adanyan bahasa yang bersifat standar, komunikasi perancang dengan pemrogram (komunikasi antar anggota kelompok pengembang) serta calon pengguna diharapkan menjadi mulus, adapun pengertian UML menurut para ahli dapat dipaparkan sebagai berikut:

Menurut Shofwan Hanief & Dian Pramana (2018 : 166) menyatakan bahwa:

"*Unified Modelling Language* (UML) adalah sebuah "bahasa" yang telah menjadi standar dalam industri untuk visualisas, merancang dan mendokumentasikan sistem piranti lunak".

Menurut Eng RH. Sianipar (2017 :75) menyatakan bahwa:"UML merupakan singkatan *Unifed Modelling Language*, yang telah menjadi notasi popular untuk mempresentasikan perancangan atas sebuah program berorientasi objek".

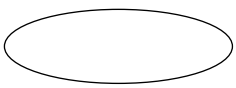
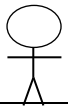
Alat bantu yang digunakan dalam perancangan berorientasi objek berbasiskan UML adalah sebagai berikut:

1. *Use case* Diagram

Menurut Sri Mulyani (2017 : 245) menyatakan bahwa *Use Case Diagram* yaitu *diagram* yang menggambarkan dan merepresentasikan aktor, *use cas*e dan *dependencies* suatu proyek dimana tujuan dari diagram ini adalah menjelaskan konsep hubungan antara sistem dengan dunia luar.

Jadi, dapat disimpulkan *Use case* adalah langkah – langkah atau urutan kegiatan yang dilakukan actor dan sistem informasi yang akan dibuat. Secara singkat, *Use case* digunakan untuk mengetahui fungsi apa saja yang ada didalam sebuah sistem informasi dan siapa saja yang berhak menggunakan.

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 : Sri Mulyani : 2016: 245)

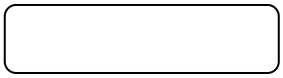
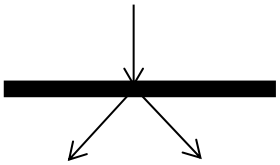
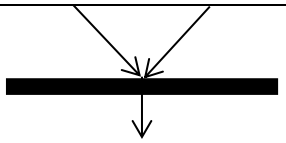
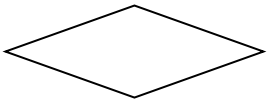
2. Diagram Aktivitas (*Activity Diagram*)

Activity diagram menggambarkan rangkaian aliran dari aktivitas yang dibentuk dalam suatu operasi sehingga dapat juga digunakan untuk aktifitas lainnya seperti *use case* atau interaksi.

Menurut Sri Mulyani (2016 : 249) : "*Activity diagram* adalah diagram UML yang digunakan untuk menggambarkan alur aktivitas dari suatu proses" Menurut Muhammad Muslihudin dan Oktafianto (2016 :63) mengungkapkan: "Diagram aktivitas adalah tipe khusus dari diagram status yang memperlihatkan aliran dari suatu aktivitas ke aktivitas lainnya dalam suatu sistem".

Simbol-simbol yang digunakan dalam *activity diagram* dapat dilihat pada tabel II.2:

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 : Sri Mulyani : 2017: 249)

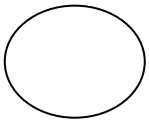
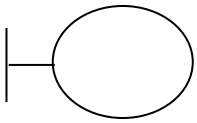
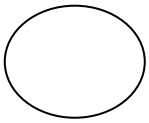
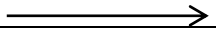
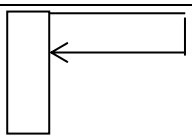
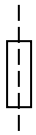
3. Diagram Urutan (*Sequence Diagram*)

Sequence diagram menggambarkan kelakuan objek pada *use case* dengan mendeskripsikan waktu hidup objek dan pesan yang dikirimkan dan diterima antar objek.

Menurut (Irmayani & Susyatih, 2017), *Sequence Diagram* menggambarkan bagaimana sistem merespon kegiatan user. *Sequence Diagram* yang dibuat yaitu berhubungan langsung dengna kegiatan utama dari sistem anggaran pendapatan dan belanja desa berbasis objek. Simbol-simbol yang

digunakan dalam *sequence diagram* dapat dilihat pada tabel II.3:

Tabel II.3. 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 sistem 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 : Irmayani & Susyati, 2017)

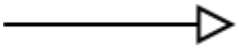
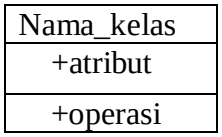
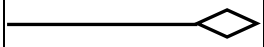
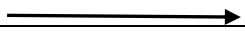
4. Diagram Kelas (*Class Diagram)*

Diagram kelas (*Class Diagram*) adalah diagram yang digunakan untuk menampilkan beberapa kelas serta paket-paket yang ada dalam sistem atau perangkat lunak yang sedang kita kembangkan. Dan berikut ini merupakan penjelasan mengenai *class diagram.*

Menurut Sri Mulyani (2017 : 247) menyatakan bahwa : "*Class Diagram* adalah diagram yang digunakan untuk mempresentasikan kelas, komponen-

komponen kelas dan hubungan antara masing-masing kelas" Simbol *class diagram* dan *multiplicity class diagram* dapat dilihat pada Tabel II.4 dan Tabel II.5:

Tabel II.4. Simbol *Class Diagram*

Gambar	Keterangan
	*Generalization*, untuk menghubungkan antar kelas dengan arti umum-khusus. Jadi jika ada kelas dengan arti umum-khusus. Jadi jika ada kelas bermakna umum dan kelas bermakna khusus dapat menggunakan simbol ini.
	*Class*, untuk sebuah kelas pada struktur sistem. Penulisan tidak boleh menggunakan spasi. Simbol ini memiliki 3 susunan, yaitu kotak pertama adalah nama kelas, kedua atribut dan ketiga operasi.
	*Interface*, untuk simbol *interface* atau dalam bahasa indonesianya antar muka. Konsep yang digunakan pun sama dengan pemrograman berorientasi object (OOP).
	*Association,*digunakan untuk menghubungkan atau merelasikan kelas satu dengan kelas yang lainnya dengan makna umum.
	Directed Association, adalah relasi antar kelas dengan makna kelas yang satu digunakan oleh kelas yang lain.
	*Aggregation*, adalah relasi antar kelas dengan makna semua bagian.
	*Dependency*, adalah relasi antar kelas dengan makna kebergantungan antar kelas.

(Sumber : Sri Mulyani : 2017 : 247)

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 : Sri Mulyani : 2017 : 247)