PENGAMANAN KEAMANAN PADA ENKRIPSI DAN DEKRIPSI DENGAN MENGGUNAKAN METODE VERTICAL BIT ROTATION

Disusun dan Diajukan untuk Memenuhi Persyaratan Ujian Akhir Memperoleh
Gelar Sarjana Komputer pada Fakultas Sains dan Teknologi
Universitas Pembangunan Panca Budi
Medan

SKRIPSI

OLEH

NAMA	**: RIO NOVANDI ANWAR**
NPM	**: 1514370223**
PROGRAM STUDI	**: SISTEM KOMPUTER**

FAKULTAS SAINS DAN TEKNOLOGI
UNIVERSITAS PEMBANGUNAN PANCA BUDI
MEDAN
2019

ABSTRAK

RIO NOVANDI ANWAR
Pengamanan Keamanan Pada Enkripsi Dan Dekripsi Dengan Menggunakan Metode Vertical Bit Rotation
2019

Pengamanan informasi sangat penting dilakukan. Hal ini bertujuan agar informasi tersebut tidak dicuri oleh orang yang tidak bertanggung jawab. Dalam menjaga agar informasi tersebut tidak salah digunakan, pengguna harus menggunakan teknik yang disebut dengan kriptografi. Teknik ini akan melindungi informasi agar tidak dapat dipecahkan sehingga informasi yang sebenarnya dapat disalahgunakan. Ada banyak teknik kriptografi yang dapat dilakukan dalam mengamankan data. Teknik kriptografi yang digunakan penulis adalah dengen melakukan pergeseran atau perputaran bit secara vertikal atau vertical bit rotation. Teknik ini sangat mudah dilakukan dan juga ringan sehingga proses enkripsi dan dekripsi dapat dilakukan dengan cepat. Dengan menerapkan teknik vertical bit rotation, kemanaan informasi dapat ditingkatkan.

Kata Kunci: dekripsi, enkripsi, kriptografi, VBR

DAFTAR ISI

DAFTAR GAMBAR

DAFTAR TABEL

BAB I

PENDAHULUAN

1.1 Latar Belakang

Keamanan sistem informasi tidak hanya menangani informasi komputer, tetapi juga melindungi data dan informasi dalam semua bentuknya, seperti percakapan telepon. Penilaian risiko harus dilakukan untuk menentukan informasi apa yang merupakan risiko terbesar. Sebagai contoh, satu sistem mungkin memiliki informasi paling penting di dalamnya dan oleh karena itu akan memerlukan langkah-langkah keamanan lebih untuk menjaga keamanan. Perencanaan kelangsungan keamanan data adalah aspek lain dari seseorang perlu memikirkan keamanan sistem informasi tersebut. Keamanan informasi berarti melindungi informasi dan sistem informasi dari akses, penggunaan, pengungkapan, gangguan, modifikasi, atau perusakan yang tidak sah. Manajemen Keamanan Informasi adalah proses mendefinisikan kontrol keamanan untuk melindungi aset informasi. Tindakan pertama dari program manajemen untuk menerapkan keamanan informasi adalah memiliki program keamanan.

Ada beberapa metode yang dapat dilakukan untuk menghindari pencurian data atau penyalahgunaan informasi. Metode yang dapat digunakan adalah kriptografi dan steganografi. Pada kriptografi, pesan asli akan ditransformasikan ke bentuk yang tidak dapat difahami oleh manusia. Untuk memecahkan problem yang dihasilkan oleh proses enkripsi pada kritografi memerlukan waktu dan cara, tergantung dari kekuatan dan kerumitan algoritma yang digunakan pada

kriptografi tersebut. Kriptografi itu sendiri memiliki algoritma yang sangat banyak. Salah satu diantaranya adalah algoritma *Vertical Bit Rotation*. Algoritma ini bekerja dengan cara memutar posisi bit dari *plaintext* sesuai dengan lebar kolom yang ditentukan. Algoritma ini bekerja dengan cepat karena tidak membutuhkan perhitungan matematika yang rumit.

Untuk menguji perhitungan kebenaran dari algoritma ini, dibutuhkan suatu aplikasi yang dapat bekerja secara cepat dan menghasilkan *ciphertext* dengan benar. Berdasarkan permasalahan yang sudah diungkapkan sebelumnya, maka penulis mencoba untuk memilih judul "**PENGAMANAN KEAMANAN PADA ENKRIPSI DAN DEKRIPSI DENGAN MENGGUNAKAN METODE VERTICAL BIT ROTATION**".

1.2 Rumusan Masalah

Adapun rumusan masalah yang digunakan dalam penulisan tugas akhir ini adalah sebagai berikut:

1. Bagaimana melakukan proses enkripsi dan dekripsi dengan algoritma *Vertical Bit Rotation*?
2. Bagaimana menentukan pergeseran bit secara vertikal pada algoritma *Vertical Bit Rotation*?
3. Bagaimana menentukan kunci enkripsi dan dekripsi pada algoritma *Vertical Bit Rotation*?

1.3 Batasan Masalah

Adapun batasan masalah yang digunakan dalam penulisan tugas akhir ini adalah sebagai berikut:

1. Algoritma *Vertical Bit Rotation* mengunakan pergeseran bit berdasarkan tabel ASCII.
2. Jumlah kolom dibatasi hanya menggunakan lebar 10 karakter.
3. Pesan yang digunakan sebagai *plaintext* adalah pesan berbasis teks.

1.4 Tujuan Penelitian

Adapun tujuan penelitian yang digunakan dalam penulisan tugas akhir ini adalah sebagai berikut:

1. Untuk melakukan proses enkripsi dan dekripsi dengan algoritma *Vertical Bit Rotation.*
2. Untuk menentukan pergeseran bit secara *vertikal*.
3. Untuk menentukan kunci enkripsi dan dekripsi pada algoritma *Vertical Bit Rotation.*

1.5 Manfaat Penelitian

Adapun manfaat penelitian yang digunakan dalam penulisan tugas akhir ini adalah sebagai berikut:

1. Dapat memberikan keamanan pada informasi yang akan dikirim melalui jaringan luas.
2. Informasi yang dikirimkan tidak rentan untuk dipecahkan karena sudah mengalami proses keamanan

BAB II

LANDASAN TEORI

2.1 Pencurian Data

Pencurian data adalah istilah yang digunakan untuk menggambarkan ketika informasi disalin atau diambil secara ilegal dari bisnis atau orang lain. Biasanya, informasi ini adalah informasi pengguna seperti kata sandi, nomor jaminan sosial, informasi kartu kredit, informasi pribadi lainnya, atau informasi rahasia perusahaan lainnya. Karena informasi ini diperoleh secara ilegal, ketika orang yang mencuri informasi ini ditangkap, ia akan dituntut secara hukum sepenuhnya *(Yakub, 201*2).

Pencurian data adalah transfer ilegal atau penyimpanan informasi apa pun yang bersifat rahasia, pribadi, atau finansial, termasuk kata sandi, kode perangkat lunak, atau algoritma, informasi berorientasi proses, atau teknologi eksklusif. Dianggap sebagai pelanggaran keamanan dan privasi yang serius, konsekuensi dari pencurian data bisa sangat parah bagi individu dan bisnis.

Beberapa hal yang sering secara umum termasuk dalam pencurian data adalah sebagai berikut:

1. *Drive USB* - Menggunakan teknik mengisap jempol, informasi dapat dipindahkan ke thumb *drive* atau *drive USB*. Ini dianggap sebagai metode paling mudah pencurian data karena kapasitas penyimpanan perangkat *USB* meningkat seiring waktu dengan biaya yang menurun.

2. *Hard drive* - Informasi besar dapat ditransfer menggunakan hard *drive portabel*
3. Perangkat yang menggunakan kartu memori, *PDA* - *Slurping pod* dimungkinkan dengan perangkat yang menggunakan kartu memori dan *PDA*
4. *Email* - Cara populer lain untuk mengirimkan informasi adalah melalui *email*.
5. Mencetak - Metode lain yang digunakan dalam pencurian data adalah dengan mencetak informasi dan secara ilegal menyimpan atau mendistribusikannya.
6. Berbagi jarak jauh - Menggunakan akses jarak jauh, data dapat ditransfer ke lokasi lain dari mana data dapat didistribusikan.
7. Serangan *malware* - Serangan *malware* berpotensi mengekstraksi informasi sensitif.

Ada beberapa hal yang dapat dilakukan agara pencurian data dapat dicegah, antara lain:

1. Enkripsi informasi rahasia atau informasi pribadi.
2. Sistem manajemen data memiliki langkah-langkah keamanan yang diperlukan untuk memastikan *file* perusahaan tidak dipindahkan atau diakses secara ilegal.
3. Tinjauan berkala pada perangkat dan sistem yang dapat menimbulkan risiko tinggi.

4. Penggunaan jaringan terbatas dalam suatu organisasi.
5. Penggunaan terbatas perangkat yang dapat menyimpan data.
6. Penguncian laptop dan langkah-langkah keamanan biometrik.
7. Melindungi informasi rahasia dan pribadi menggunakan kata sandi.
8. Penggunaan perangkat lunak anti-malware.

Pencurian data adalah tindakan mencuri informasi digital yang disimpan di komputer, server, atau perangkat elektronik dari korban yang tidak dikenal dengan maksud untuk membahayakan privasi atau mendapatkan informasi rahasia. Informasi dapat mencakup apa saja dari informasi keuangan, seperti nomor kartu kredit atau rekening bank, hingga informasi pribadi, seperti nomor jaminan sosial, nomor *SIM*, dan catatan kesehatan. Setelah hanya masalah bisnis besar dan organisasi, pencurian data adalah masalah yang berkembang untuk pengguna komputer sehari-hari *(Stallings, 2013)*.

2.1.1 Bagaimana Pencurian Data Terjadi

Pencurian data terjadi melalui berbagai cara. Paling sering, itu terjadi karena seseorang meretas ke dalam sistem komputer untuk mencuri informasi sensitif, seperti kartu kredit atau informasi pribadi Anda, atau seorang karyawan di perusahaan yang salah menangani informasi tersebut. Dengan dunia yang semakin digital, ratusan bisnis dan organisasi yang berbeda menyimpan informasi pribadi Anda, seperti nomor jaminan sosial, alamat surat, tanggal lahir, dan informasi rekening bank Anda.

Bahkan dengan kemajuan teknologi baru, penjahat *cyber* dapat beradaptasi dan menemukan cara untuk meretas ke dalam sistem untuk mencuri data, terutama perusahaan ritel yang menampung informasi pembayaran. Sebagian besar perusahaan memiliki rencana pelanggaran data, tetapi banyak karyawan tidak tahu mereka ada atau tidak yakin rencana itu akan berhasil. Sangat penting bahwa semua perusahaan yang menangani data sensitif mendidik dan melatih karyawan tentang cara menangani informasi sensitif.

2.1.2 Menghindari Pencurian Data

Pencurian data adalah masalah nyata dan dapat terjadi pada siapa saja. Meskipun tidak ada cara untuk sepenuhnya mencegah pencurian data, ada beberapa langkah yang dapat diambil hari ini untuk membatasi risiko kehilangan data, antara lain:

1. Bayar menggunakan uang tunai, bukan kartu kredit atau debit.
2. Gunakan kartu kredit atau debit dengan teknologi pin-and-chip.
3. Lindungi komputer Anda dari *virus* dan *malware* dengan menginstal, menggunakan, dan memperbarui perangkat lunak *antivirus* dan *anti-spyware* di semua komputer dan perangkat elektronik Anda.
4. Selalu perbarui semua sistem operasi dan program perangkat lunak dengan menginstal pembaruan keamanan, peramban *web*, sistem operasi, dan program perangkat lunak sesegera mungkin setelah tersedia.
5. Jangan buka surel yang meragukan atau lampiran surel karena bisa berupa surel *phising*.

6. Periksa secara teratur laporan kartu kredit Anda dan laporan kredit untuk biaya tidak sah dan jalur kredit baru.
7. Gunakan kata sandi yang kuat dan unik untuk semua situs web yang membutuhkan login. Ubah ini secara rutin, terutama jika kata sandi akun telah dikompromikan dalam pelanggaran data.
8. Gunakan hanya koneksi *Wi-Fi* yang aman.
9. Buang dokumen dengan benar yang berisi informasi sensitif melalui kertas penghancur kertas dan singkirkan semua data dari perangkat elektronik.
10. Amankan jaringan dan koneksi internet melalui firewall dan kata sandi aman.
11. Jika menjalankan bisnis yang menyimpan informasi sensitif, pastikan karyawan terlatih dalam menangani data dan karyawan memahami kebijakan perusahaan terkait dengan berbagi informasi sensitif.

2.2 Algoritma

Algoritma adalah seperangkat instruksi yang dirancang untuk melakukan tugas tertentu. Ini bisa berupa proses sederhana, seperti mengalikan dua angka, atau operasi yang rumit, seperti memutar file video terkompresi. Mesin pencari menggunakan algoritma kepemilikan untuk menampilkan hasil yang paling relevan dari indeks pencarian mereka untuk permintaan tertentu (Hidayat, 2012).

Dalam pemrograman komputer, algoritma sering dibuat sebagai fungsi. Fungsi-fungsi ini berfungsi sebagai program kecil yang dapat dirujuk oleh program yang lebih besar. Misalnya, aplikasi tampilan gambar dapat menyertakan

pustaka fungsi yang masing-masing menggunakan algoritma khusus untuk membuat format file gambar yang berbeda. Program pengeditan gambar dapat berisi algoritma yang dirancang untuk memproses data gambar. Contoh algoritma pemrosesan gambar termasuk pemangkasan, pengubahan ukuran, penajaman, pengaburan, reduksi mata merah, dan peningkatan warna (Firmansyah, 2012).

Dalam banyak kasus, ada beberapa cara untuk melakukan operasi tertentu dalam program perangkat lunak. Oleh karena itu, programmer biasanya berusaha membuat algoritma yang seefisien mungkin. Dengan menggunakan algoritma yang sangat efisien, pengembang dapat memastikan program mereka berjalan secepat mungkin dan menggunakan sumber daya sistem minimal. Tentu saja, tidak semua algoritma diciptakan dengan sempurna untuk pertama kalinya. Oleh karena itu, pengembang sering meningkatkan algoritma yang ada dan memasukkannya dalam pembaruan perangkat lunak di masa mendatang. Ketika Anda melihat versi baru dari program perangkat lunak yang telah "dioptimalkan" atau memiliki "kinerja lebih cepat," sebagian besar berarti versi baru mencakup algoritma yang lebih efisien (Edraw, 2019).

2.2.1 Definisi Algoritma

Sebagai metode yang efektif, suatu algoritma dapat diekspresikan dalam jumlah ruang dan waktu yang terbatas dan dalam bahasa formal yang terdefinisi dengan baik untuk menghitung suatu fungsi. Mulai dari keadaan awal dan input awal (mungkin kosong), instruksi menjelaskan perhitungan yang, ketika dijalankan, berlanjut melalui sejumlah terbatas dari negara berturut-turut yang

terdefinisi dengan baik, akhirnya menghasilkan "*output*" dan berakhir pada kondisi akhir akhir (Sumandri, 2017). Transisi dari satu negara ke negara lain tidak selalu bersifat deterministik; beberapa algoritma, yang dikenal sebagai algoritma acak, memasukkan input acak. Ada empat fitur utama dari algoritma dari definisi:

1. Algoritma bekerja untuk menghasilkan output tertentu.
2. Algoritma bekerja dengan saling terhubung dan berkelanjutan.
3. Algoritma adalah kumpulan dari perintah-perintah kecil untuk mencapai tujuan tertentu.
4. Hasil algoritma akan muncul ketika serangkaian proses sudah terlaksana dengan baik.

Pada prinsipnya, algoritma akan bekerja dengan cara yang masuk akal dan mengerjakan perintah-perintah untuk menghasilkan keluaran yang benar.

2.2.2 Jenis-Jenis Algoritma

Algoritme adalah serangkaian urutan instruksi atau tindakan yang berisi ruang terbatas atau urutan dan yang akan memberikan hasil untuk masalah tertentu dalam jumlah waktu terbatas. Ini adalah pendekatan logis dan matematis untuk memecahkan atau memecahkan masalah menggunakan metode apa pun yang mungkin. Ada banyak jenis algoritme tetapi jenis algoritme yang paling mendasar adalah:

1. Algoritma rekursif

   Ini memecahkan kasus dasar secara langsung dan kemudian berulang dengan input yang lebih sederhana atau lebih mudah setiap kali (Nilai dasar ditetapkan di awal yang diakhiri algoritma). Ini digunakan untuk memecahkan masalah yang dapat dipecah menjadi masalah yang lebih sederhana atau lebih kecil dari jenis yang sama.

2. Algoritma pemrograman dinamis

   Algoritma pemrograman dinamis (juga dikenal sebagai algoritma optimasi dinamis) mengingat hasil masa lalu dan menggunakannya untuk menemukan hasil baru berarti memecahkan masalah kompleks dengan memecahnya menjadi kumpulan subproblem yang lebih sederhana, kemudian menyelesaikan masing-masing subproblem tersebut hanya sekali, dan menyimpannya solusi mereka untuk digunakan di masa depan alih-alih menghitung ulang solusi mereka lagi.

3. Algoritma *Backtracking*

   Bagaimana kalau belajar mengulang menggunakan contoh katakanlah ada suatu masalah "*MONK*" dan akan dibagi menjadi empat masalah yang lebih kecil "M, R, A, A". Mungkin masalahnya solusi dari masalah ini tidak diterima sebagai solusi dari " *MONK* ". Faktanya, tidak tahu yang mana tergantung. Jadi algoritma akan memeriksa masing-masing dari untaian tersebut satu per satu sampai ditemukan solusi untuk "*MONK*". Jadi pada dasarnya algoritma berusaha memecahkan submasalah tetapi

jika tidak mencapai solusi yang diinginkan akan membatalkan apa pun yang telah dilakukan dan mulai dari awal lagi sampai menemukan solusinya.

4. Algoritma *Divide and Conquer*

   Membagi dan menaklukkan terdiri dari dua bagian, pertama-tama, membagi masalah menjadi sub-masalah yang lebih kecil dari jenis yang sama dan menyelesaikannya secara rekursif dan kemudian menggabungkannya untuk membentuk solusi dari masalah asli.

5. Algoritma *Greedy*

   Algoritma *Greedy* adalah algoritma yang memecahkan masalah dengan mengambil solusi optimal di tingkat lokal (tanpa memperhatikan konsekuensi apa pun) dengan harapan menemukan solusi optimal di tingkat global. Algoritma *Greedy* digunakan untuk menemukan solusi optimal tetapi tidak perlu bahwa akan menemukan solusi optimal dengan mengikuti algoritma ini. Seperti ada beberapa masalah di mana solusi optimal tidak ada (saat ini) ini disebut masalah *NP-complete*.

6. Algoritma *Brute Force*

   Algoritma *Brute Force* hanya mencoba semua kemungkinan sampai solusi yang memuaskan ditemukan. Jenis algoritma seperti itu juga digunakan untuk menemukan solusi optimal (terbaik) karena memeriksa semua solusi yang mungkin. Dan juga digunakan untuk menemukan solusi yang memuaskan (bukan yang terbaik), cukup berhenti segera setelah solusi untuk masalah ditemukan.

7. Algoritma acak

   Algoritma acak menggunakan nomor acak setidaknya sekali selama perhitungan untuk membuat keputusan.

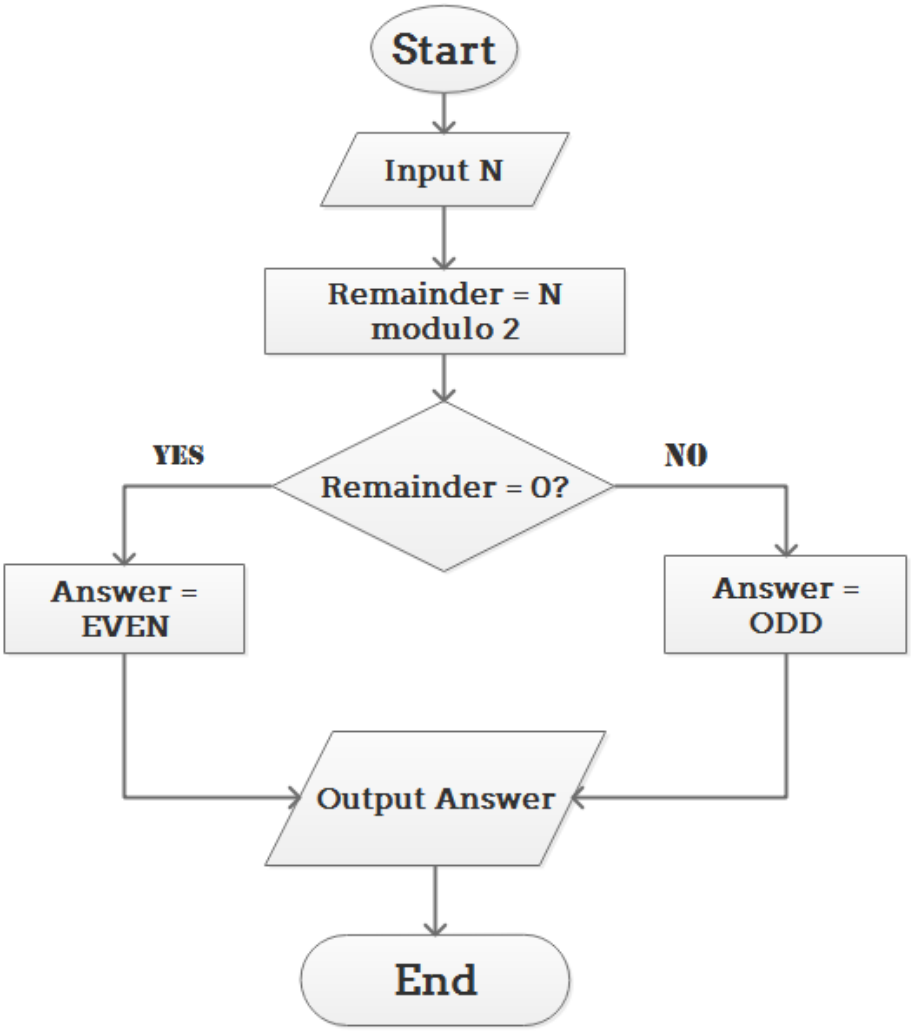
2.2.3 Analisis Kompleksitas Algoritma

Analisis suatu algoritma mengacu pada proses memperoleh estimasi untuk waktu dan ruang yang dibutuhkan untuk menjalankan algoritma. Penting untuk memperkirakan waktu (mis., Jumlah langkah) dan ruang (mis., Jumlah variabel) yang dibutuhkan oleh algoritma. Mengetahui waktu dan ruang yang dibutuhkan oleh algoritma memungkinkan kita untuk membandingkan algoritma yang memecahkan masalah yang sama. Sebagai contoh, jika satu algoritma mengambil n langkah untuk menyelesaikan masalah dan algoritma lainnya mengambil n ^ 2 langkah untuk memecahkan masalah yang sama, kami lebih suka algoritma pertama. Estimasi waktu dan ruang yang diperlukan untuk menjalankan algoritma ini disebut kompleksitas waktu dan ruang dari algoritma.

Waktu yang diperlukan untuk menjalankan suatu algoritma adalah fungsi dari input. Alih-alih berurusan langsung dengan input, parameter digunakan untuk mengkarakterisasi ukuran input. misalnya jika input adalah himpunan yang berisi n elemen, ukuran input n. Ada tiga kasus yang perlu dicatat tentang kompleksitas waktu suatu algoritma karena menentukan kompleksitas waktu yang tepat dari suatu algoritma dalam tugas yang sulit.

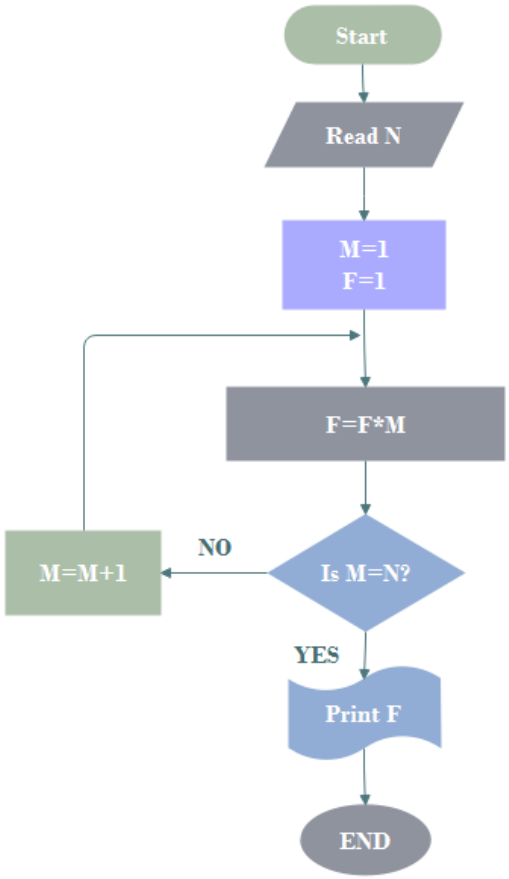
1. Kasus terburuk: f (n) diwakili oleh jumlah maksimum langkah yang diambil pada setiap *instance* ukuran n.

2. Kasing terbaik: f (n) diwakili oleh jumlah minimum langkah yang diambil pada setiap *instance* ukuran n.
3. Kasus rata-rata: f (n) diwakili oleh jumlah rata-rata langkah yang diambil pada setiap contoh ukuran n.



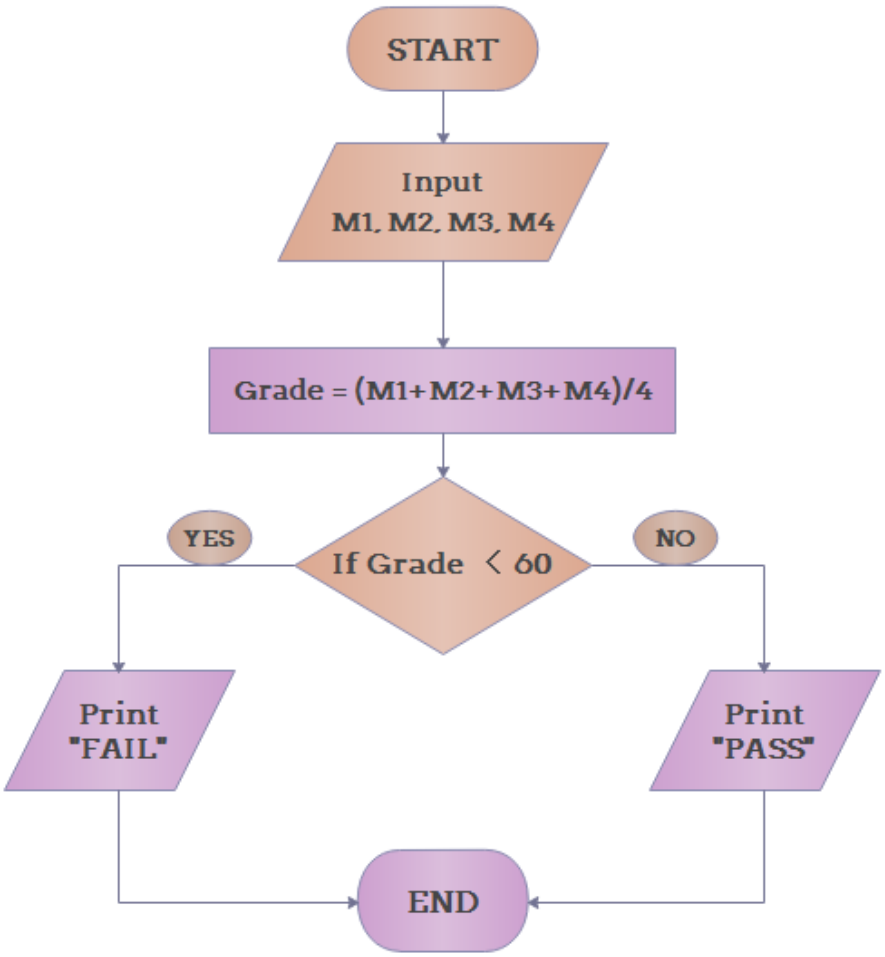
Gambar 2.1 *Flowchart* algoritma bidang matematika
Sumber: (Edraw, 2019)

Berikut ini adalah contoh *flowchart* aplikasi algoritma untuk bidang komputer yaitu menentukan hasil faktorial dari bilangan N.



Gambar 2.2 *Flowchart* algoritma bidang komputer
Sumber: (Edraw, 2019)

Berikut ini adalah contoh flowchart aplikasi algoritma untuk bidang pendidikan formal atau sekolah yaitu menentukan kelulusan mahasiswa.



Gambar 2.3 *Flowchart* algoritma bidang pendidikan
Sumber: (Edraw, 2019)

Gambar-gambar sebelumnya adalah diagram penggunaan algoritma pada beberapa bidan. Contoh-contoh tersebut memberikan demonstrasi yang jelas dari aplikasi algoritma dalam matematika, pemrograman komputer dan pendidikan.

2.3 Kriptografi

Kriptografi adalah ilmu dan seni untuk menjaga kerahasiaan pesan dengan cara menyandikannya ke dalam bentuk yang tidak dapat dimengerti lagi maknanya. Dalam ilmu kriptografi, terdapat dua buah proses yaitu melakukan enkripsi dan dekripsi (Munir, 2006). Pesan yang akan dienkripsi disebut sebagai *plaintext* (teks biasa). Disebut demikian karena informasi ini dengan mudah dapat dibaca dan dipahami oleh siapa saja. Algoritma yang dipakai untuk mengenkripsi dan mendekripsi sebuah *plaintext* melibatkan penggunaan suatu bentuk kunci. Pesan *plaintext* yang telah dienkripsi (atau dikodekan) dikenal sebagai *ciphertext* (teks sandi). Di dalam kriptografi kita akan sering menemukan berbagai istilah atau terminologi. Beberapa istilah yang harus diketahui yaitu :

1. Pesan, *Plainteks*, dan *Cipherteks*

   Pesan (*message*) adalah data atau informasi yang dapat dibaca dan dimengerti maknanya. Nama lain untuk pesan adalah (*plaintext*) atau teks jelas (*cleartext*).

2. Pengirim dan Penerima

   Komunikasi data melibatkan pertukaran pesan antara dua entitas. Pengirim (*sender*) adalah entitas yang mengirim pesan kepada entitas lainnya. Penerima (*receiver*) adalah entitas yang menerima pesan.

   Enkripsi dan dekripsi

3. Proses menyandikan plainteks menjadi *cipherteks* disebut enkripsi (*encryption*) atau *enciphering* (standard nama menurut ISO 7498-2). Sedangkan proses mengembalikan *cipherteks* menjadi *plainteks* semula

disebut dekripsi (*decryption*) atau deciphering (standard nama menurut ISO 7498-2).

4. Cipher dan kunci

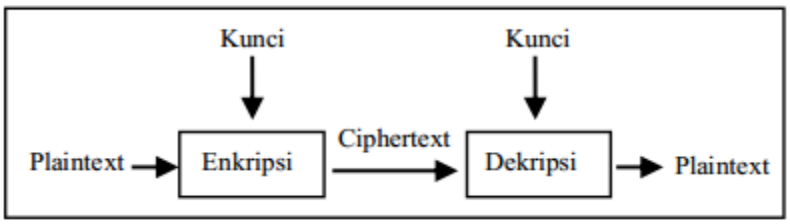
   Algoritma kriptografi disebut juga *cipher*, yaitu aturan untuk enkripsi dan dekripsi, atau fungsi matematika yang digunakan untuk enkripsi dan dekripsi. Beberapa cipher memerlukan algoritma yang berbeda untuk enkripsi dan dekripsi. Konsep matematis yang mendasari algoritma kriptografi adalah relasi antara dua buah himpunan yang berisi elemen-elemen *plainteks* dan himpunan yang berisi *cipherteks*. Enkripsi dan dekripsi merupakan fungsi yang memetakan elemen-elemen antara dua himpunan tersebut. Misalkan P menyatakan *plainteks* dan C menyatakan *cipherteks*, maka :

   $E(P) = C$ ▶ fungsi enkripsi E memetakan P ke C

   $D(C) = P$ ▶ fungsi dekripsi D memetakan C ke P

   Karena proses enkripsi kemudian dekripsi mengembalikan pesan ke pesan asal, maka persamaan $D(E(P)) = P$ harus benar. Kriptografi mengatasi masalah keamanan data dengan menggunakan kunci, yang dalam hal ini Algoritma tidak dirahasiakan lagi, tetapi kunci harus tetap dijaga kerahasiaannya. Kunci (*key*) adalah parameter yang digunakan untuk transformasi enkripsi dan dekripsi. Kunci biasanya berupa *string* atau deretan bilangan. Dengan menggunakan kunci K, maka fungsi enkripsi

dan dekripsi dapat ditulis sebagai skema yang dijelaskan pada Gambar 2.4.



Gambar 2.4 Skema enkripsi dan dekripsi dengan menggunakan kunci
Sumber: (Edraw, 2019)

2.3.1 Sejarah Kriptografi

Kriptografi mempunyai sejarah yang sangat panjang. Kriptografi sudah digunakan 4000 tahun yang lalu, diperkenalkan oleh orang-orang Mesir lewat *hieroglyph.* Jenis tulisan ini bukanlah bentuk standar untuk menulis pesan (Ariyus, 2008).

Gambar 2.5 Tulisan yang menunjukan *Heiroglyph*
Sumber: (Ariyus, 2008)

Dikisahkan, pada Zaman Romawi Kuno, Pada Suatu saat *Jelius Caesar* ingin mengirimkan pesan rahasia kepada seseorang jendral di medan perang. Pesan tersebut dikirimkan melalui kurir. Karena pesan tersebut mengandung rahasia, *jelius Cesar* kemudian memikirkan bagaimana mengatasinya. Ia kemudian mengacak pesan tersbut hingga menjadi suatu pesan yang tidak dapat dipahami oleh siapapun terkecuali oleh jendralnya saja. Tentu pengirim ingin mengenkripsi suatu pesan menggunakan kode *Vigenere* dengan kunci *WARTHOG.*

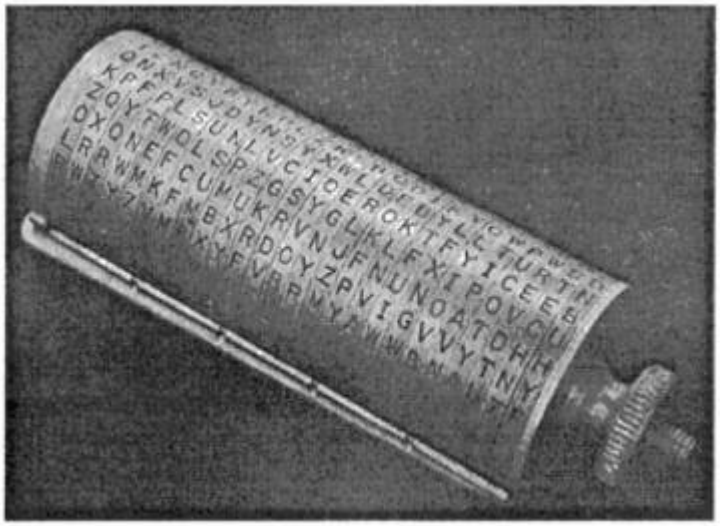
Untuk mengenkripsi huruf pertama, pengirim memutar *W* di potongan silindris dalam hingga berdampingan dengan *a* di silindris luar. Kemudian cari huruf teks-kode di potongan silindris dalam yang cocok dengan huruf teks-asli yang diinginkan di potongan silindris luar.

Selanjutnya mengirim mengenkripsi huruf kedua dengan memutar *A* di potongan silindris dalam hingga berdampingan dengan *a* di potongan silindris luar. Setelah itu cari huruf teks-kode di potongan silindris dalam yang cocok dengan huruf teks-asli yang dinginkan di potongan silindris luar.

Untuk seterusnya, ulang proses untuk huruf R,T,H,O,G dan kemudian ulang lagi dari W hingga seluruh pesan telah terenkripsi (EDUCBA, 2017).

Gambar 2.6 Roda Kaisar
Sumber: (Ariyus, 2008)

Bentuk roda kode sejak versi Jefferson hingga M94 terdiri dari sejumlah potongan silindris yang tersusun di suatu sumbu besi. Setiap potongan silindris memiliki susunan alphabet secara acak di bagaian luar. Potongan-potongan silindris tersebut menjadi dalam mengenkripsi dan mendeskripsi pesan dari pihak penerima dan pengirim. Setiap potongan silindris dapat diputar untuk menyusun alphabet menjadi teks kode ataupun menjadi teks asli. Untuk mengenkripsi suatu pesan, pengirim M94 memiliki bentuk yang hampir sama dengan roda kode jaferson. Bedanya, M94 terbuat dari alumunium. Untuk potongan ke 17, susunan alfabetnya berupa ARMYOFTHEU. Susunan ARMY OFTHEUS menunjukan "Army Orgin of thw M94". M94 memiliki 100 pilihan potongan silindris, walau mungkin yang dipilih untuk digunakan untuk suatu kunci hanya 25 buah. Hal tersebut dapat memperbanyak kemungkinan solusi dari roda kunci.



Gambar 2.7 M-94
Sumber: (Ariyus, 2008)

2.3.2 Kriptografi *Simetris*

Kriptografi Simetri (Kriptografi Kunci-*Privat*) Pada sistem kriptografi kunci-simetri, kunci untuk enkripsi sama dengan kunci untuk dekripsi, oleh karena itulah dinamakan kriptografi simetri. Keamanan sistem kriptografi simetri terletak pada kerahasiaan kuncinya. Ada banyak algoritma kriptografi modern yang termasuk ke dalam sistem kriptografi simetri, diantaranya adalah :

1. *DES (Data Encryption Standard),*
2. *Blowfish,*
3. *Twofish,*
4. *Triple-DES,*
5. *IDEA,*
6. *Serpent,*
7. *AES (Advanced Encryption* Standard).

Algoritma kriptografi (*cipher*) simetri dapat dikelompokkan menjadi dua kategori, yaitu:

1. *Cipher* aliran (stream *cipher*)

   Algoritma kriptografi beroperasi pada plainteks/cipherteks dalam bentuk bit tunggal, yang dalam hal ini rangkaian bit dienkripsikan/didekripsikan bit per bit.

2. *Cipher* blok *(block cipher*)

   Algoritma kriptografi beroperasi pada *plainteks/cipherteks* dalam bentuk blok bit, yang dalam hal ini rangkaian bit dibagi menjadi blok-blok bit yang panjangnya sudah ditentukan sebelumnya.

2.4 Cipher Substitusi

Dalam kriptografi, *cypher* substitusi adalah metode enkripsi dengan mana unit *plaintext* diganti dengan *ciphertext*, sesuai dengan sistem tetap; "unit" dapat berupa huruf tunggal (yang paling umum), pasangan huruf, kembar tiga huruf, campuran di atas, dan sebagainya. Penerima menguraikan teks dengan melakukan substitusi terbalik. *Cipher* substitusi dapat dibandingkan dengan *cipher* transposisi. Dalam sandi transposisi, satuan *plaintext* disusun ulang dalam urutan yang berbeda dan biasanya cukup kompleks, tetapi satuan itu sendiri tidak berubah. Sebaliknya, dalam *cipher* subtitusi, unit *plaintext* dipertahankan dalam urutan yang sama dalam *ciphertext*, tetapi unit itu sendiri diubah (Wagner, 2003).

Ada sejumlah jenis cipher substitusi yang berbeda. Jika sandi beroperasi pada huruf tunggal, itu disebut sandi substitusi sederhana; sandi yang beroperasi

pada kelompok huruf yang lebih besar disebut *poligrafi*. *Cipher monoalphabetic* menggunakan substitusi tetap atas seluruh pesan, sedangkan *cipher polyalphabetic* menggunakan sejumlah substitusi pada posisi yang berbeda dalam pesan, di mana unit dari *plaintext* dipetakan ke salah satu dari beberapa kemungkinan dalam *ciphertext* dan sebaliknya (Weerasinghe, 2013).

2.1 Vertical Bit Rotation

Vertical Bit Rotation (VBR) adalah salah satu dari ilmu kriptografi yang bekerja dengan cara menggeser posisi bit pada karakter secara *vertikal.* Tiap karakter akan dikonversikan ke bentuk biner sehingga memiliki delapan buah bit. Ada sebuah kunci yang digunakan sebagai penentu berapa jarak pergeseran bit tersebut. Rotasi dapat dilakukan secara ke atas atau ke bawah. Nilai maksimal yang dapat dilakukan adalah sebesar 7 dan nilai minimal pergeseran adalah sebesar 1.

PT	KUNCI	CT
0	3	0
1		0
0		1
0		0
0		1
0		0
0		0
1		0

Gambar 2.8 Skema pergeseran bit pada *VBR*
Sumber: (Mollin, 2001)

Gambar 2.4 adalah skema pertukan bit pada tiap karakter. Setiap bit akan digeser posisinya hingga berada di posisi di atas atau di bawah dari posisi bit sebelumnya. Jarak pergeseran ditentukan sesuai dengan nilai kunci. Hasil *ciphertext* diperoleh dengan cara mengubah posisi bit-bit pada karakter *plaintext* dengan jarak yang ditentukan. Pergeseran dapat dilakukan ke arah atas atau ke arah bawah sesuai dengan kunci yang sudah ditentukan.

2.5 Unified Modelling Language

Unified Modeling Language adalah Metodologi kolaborasi antara metoda-metoda Booch, OMT (*Object Modeling Technique*), serta OOSE (*Object Oriented Software Enggineering*) dan beberapa metoda lainnya, merupakan metodologi yang paling sering digunakan saat ini untuk analisa dan perancangan sistem dengan metodologi berorientasi objek mengadaptasi maraknya penggunaan bahasa “pemrograman berorientasi objek” (OOP) (Wasserkrug et al., 2009).

Beberapa literature menyebutkan bahwa UML menyediakan sembilan jenis diagram, yang lain menyebutkan delapan karena ada beberapa diagram yang digabung, misalnya diagram komunikasi, diagram urutan dan diagram pewaktuan digabung menjadi diagram interaksi (Sukmawati & Priyadi, 2019).

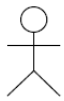
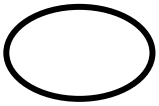
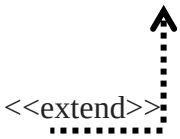
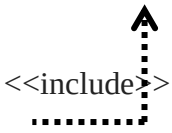
2.5.1 Use Case Diagram

Use case diagram adalah abstraksi dari interaksi antara sistem dan aktor. *Use case diagram* bekerja dengan cara mendeskripsikan tipe interaksi antara user sebuah sistem dengan sistemnya sendiri melalui sebuah cerita bagaimana sebuah sistem dipakai. *Use case diagram* berguna dalam tiga hal:

1. Menjelaskan fasilitas yang ada (*requirement*).
2. Komunikasi dengan klien.
3. Membuat *test* dari kasus-kasus secara umum.

Adapun simbol-simbol dalam *Use Case Diagram* dapat dilihat pada tabel 2.1.

Tabel 2.1 Elemen-Elemen *Use Case*

SIMBOL	NAMA	KETERANGAN
	*Actor*	Menspesifikan himpunan peran yang pengguna mainkan ketika berinteraksi dengan *use case*
	*Use Case*	Deskripsi urutan aksi-sistem yang menghasilkan suatu hasil yang terukur
	*System*	Menspesifikasian paket yang menampilkan sistem secara terbatas
	*Association*	Simbol yang menghubungkan antara objek satu dengan objek lainnya
	*Dependency*	Hubungan dimana perubahan yang terjadi pada suatu elemen mandiri (*independent*) akan mempengaruhi elemen yang bergantung padanya elemen yang tidak mandiri
	*Generalization*	Hubungan dimana objek anak (*descendent*) berbagi perilaku dan struktur data dari objek yang ada di atasnya objek induk (*ancestor*)
	*Extend*	Menspesifikasikan bahwa *use case* target memperluas perilaku dari *use case* sumber pada suatu titik yang diberikan
	*Include*	Menspesifikasikan bahwa *use case* sumber secara *eksplisit*

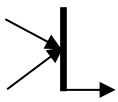
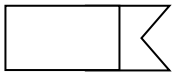
Sumber: (Kurniawan, 2018)

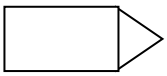
2.5.2 Activity Diagram

Activity diagram menyediakan analis dengan kemampuan untuk memodelkan proses dalam suatu sistem informasi. *Activity* diagram dapat digunakan untuk alur kerja model, *use case individual*, atau logika keputusan yang terkandung dalam metode individual. *Activity* diagram juga menyediakan pendekatan untuk proses pemodelan paralel (Ladjamudin, 2005).

Pada dasarnya, diagram aktivitas canggih dan merupakan diagram aliran data yang terbaru. Secara teknis, diagram aktivitas menggabungkan ide-ide proses pemodelan dengan teknik yang berbeda termasuk model cara, statecharts. *Activity* diagram mempunyai beberapa elemen dalam memodelkan sebuah sistem, yaitu:

Tabel 2.2 Elemen-Elemen *Activity Diagram*

SIMBOL	NAMA	KETERANGAN
	*Action State*	Menandakan sebuah aktivitas
	*Initial State*	Titik awal untuk memulai suatu aktivitas
	*Final State*	Titik akhir untuk mengakhiri aktivitas
	*Decision*	Pilihan untuk mengambil keputusan
	*Flow Final*	Untuk mengakhiri suatu aliran
	Transition	Menunjukan aktifitas selanjutnya setelah aktivitas sebelumnya
	Synchronization	Dibagi menjadi 2 yaitu fork dan join: Fork digunakan.untuk memecah behaviour menjadi activity atau action yang paralel, sedangkan join untuk menggabungkan kembali activity atau action yang parallel
	Swimlane	Untuk melakukan partisi atau pembagian
	Signal Accept State	Tanda penerimaan

	Signal Send State	Tanda pengiriman

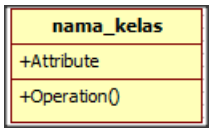
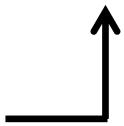
Sumber: (Kurniawan, 2018)

2.5.3 Class Diagram

Tujuan utama dari *class diagram* adalah untuk menciptakan sebuah kosa kata yang digunakan oleh analis dan pengguna. *Class diagram* biasanya merupakan hal-hal, ide-ide atau konsep yang terkandung dalam aplikasi. Misalnya, jika sedang membangun sebuah aplikasi penggajian, diagram kelas mungkin akan berisi kelas yang mewakili hal-hal seperti karyawan, cek, dan pendaftaran gaji.

Class diagram juga akan menggambarkan hubungan antara kelas. Berikut komponen-komponen yang ada pada *class* diagram.

Tabel 2.3 Elemen-Elemen *Class Diagram*

SIMBOL	NAMA	KETERANGAN
	*Class*	Kelas pada struktur sistem
	*Association*	Relasi antar kelas dengan makna umum, asosiasi biasanya juga di sertai dengan multiplicity
	*Directed Association*	Relasi antar kelas dengan makna kelas yang satu digunakan oleh kelas yang lain, asosiasi berarah biasanya juga disertai dengan multiplicity
	*Dependency*	Relasi antar kelas dengan makna kebergantungan antar kelas.

Sumber: : (Kurniawan, 2018)

2.5.4 Sequence Diagram

Sequence diagram menjelaskan interaksi objek yang disusun berdasarkan urutan waktu. Secara mudahnya *sequence* diagram adalah gambaran tahap demi tahap yang seharusnya dilakukan untuk menghasilkan sesuatu sesuai dengan *use case diagram*. Berikut komponen-komponen yang ada pada *sequence* diagram.

Tabel 2.4 Elemen-Elemen *Sequence Diagram*

SIMBOL	NAMA	KETERANGAN
	Objek	Menggambarkan objek/orang yang berintraksi di dalam sistem
	Stimulus	Menggambarkan pengiriman pesan
	Self Stimulus	Menyatakan suatu objek mengirimkan pesan untuk menjalankan oprasi yang ada pada objek lain.

Sumber: (Kurniawan, 2018)

2.5.5 Flowchart

Flowchart adalah jenis diagram yang mewakili alur kerja atau proses. Diagram alir juga dapat didefinisikan sebagai representasi diagram dari suatu algoritma, pendekatan langkah demi langkah untuk menyelesaikan suatu tugas. Diagram alur menunjukkan langkah-langkah sebagai kotak dari berbagai jenis, dan urutannya dengan menghubungkan kotak-kotak dengan panah. Representasi diagram ini menggambarkan model solusi untuk masalah yang diberikan. *Flowchart* digunakan dalam menganalisis, merancang, mendokumentasikan, atau mengelola suatu proses atau program di berbagai bidang. *Flowchart* digunakan dalam mendesain dan mendokumentasikan proses atau program sederhana. Seperti jenis diagram lainnya, diagram membantu memvisualisasikan apa yang sedang terjadi dan dengan demikian membantu memahami suatu proses, dan mungkin juga menemukan fitur yang kurang jelas dalam proses tersebut, seperti kekurangan dan hambatan. Ada berbagai jenis diagram alur: masing-masing jenis

memiliki set kotak dan notasi sendiri. Dua jenis kotak yang paling umum dalam diagram alur adalah:

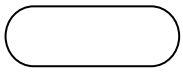
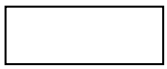
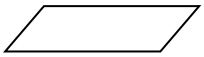
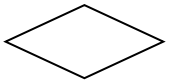
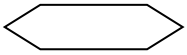
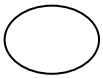
1. Langkah pemrosesan, biasanya disebut aktivitas, dan dilambangkan sebagai kotak persegi panjang.
2. Sebuah keputusan, biasanya dilambangkan sebagai berlian.

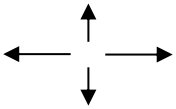
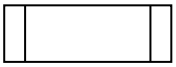
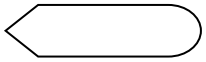
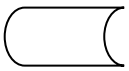
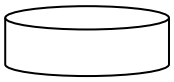
Diagram alir digambarkan sebagai "lintas fungsional" ketika bagan dibagi menjadi bagian vertikal atau horizontal yang berbeda, untuk menggambarkan kontrol unit organisasi yang berbeda. Simbol yang muncul di bagian tertentu berada dalam kendali unit organisasi itu. *Flowchart* lintas fungsional memungkinkan penulis untuk menemukan tanggung jawab untuk melakukan suatu tindakan atau membuat keputusan dengan benar, dan untuk menunjukkan tanggung jawab masing-masing unit organisasi untuk bagian-bagian berbeda dari satu proses tunggal (Nakatsu, 2009).

Struktur grafik yang mendasari diagram alur adalah grafik aliran, yang mengabstraksi jenis simpul, isinya, dan informasi tambahan lainnya. Diagram alir menggambarkan aspek-aspek tertentu dari proses dan biasanya dilengkapi dengan jenis diagram lainnya. Misalnya, Kaoru Ishikawa, mendefinisikan diagram alir sebagai salah satu dari tujuh alat dasar kendali mutu, di sebelah histogram, diagram Pareto, lembar periksa, diagram kontrol, diagram sebab-akibat, dan diagram sebaran. Demikian pula, di UML, notasi pemodelan konsep standar yang digunakan dalam pengembangan perangkat lunak, diagram aktivitas, yang merupakan jenis diagram alur, hanyalah salah satu dari banyak jenis diagram yang berbeda.

Diagram *Nassi-Shneiderman* dan *Drakon-chart* adalah notasi alternatif untuk aliran proses. Nama alternatif umum termasuk diagram alir, diagram alur proses, diagram alur fungsional, peta proses, diagram proses, diagram proses fungsional, model proses bisnis, model proses, diagram alir proses, diagram alur kerja, diagram alir bisnis. Istilah "diagram alur" dan "diagram alir" digunakan secara bergantian. Struktur grafik yang mendasari diagram alur adalah grafik aliran, yang mengabstraksi jenis simpul, isinya, dan informasi tambahan lainnya. Adapun simbol-simbol *flowchart* lihat pada tabel sebagai berikut:

Tabel 2.5 Simbol-simbol *Flowchart*

NO	SIMBOL	FUNGSI
1.		Terminal, untuk memulai atau mengakhiri suatu program
2.		Proses, suatu simbol yang menunjukkan setiap pengolahan yang dilakukan.
3.		Input-Output, untuk memasukkan menunjukkan hasil dari suatu proses
4.		Decision, suatu kondisi yang akan menghasilkan beberapa kemung-kinan jawaban atau pilihan
5.		Preparation, suatu symbol yang menyediakan tempat pengolahan
6.		Connector, suatu prosedur penghubung yang akan masuk atau keluar melalui symbol ini dalam lembar yang sama

7.		Off-Page Connector, merupakan symbol masuk atau keluarannya suatu prosedur pada lembaran kertas lainnya
8.		Arus/Flow, dari pada prosedur yang dapat dilakukan atas ke bawah dari bawah ke atas, ke atas dari kiri ke kanan ataupun dari kanan ke kiri
9.		Predefined Process, untuk menyatakan sekumpulan langkah proses yang ditulis sebagai prosedur
10.		Simbol untuk output, yang ditunjukkan ke suatu device, seperti printer, dan sebagainya
11		Penyimpanan file secara sementara
12		Menunjukkan input / Output Hardisk (media penyimpanan)

Sumber: (Kurniawan, 2018)

2.6 Visual Basic

Visual Basic (VB) adalah bahasa pemrograman yang digerakkan oleh peristiwa dan lingkungan dari Microsoft yang menyediakan antarmuka pengguna grafis (GUI) yang memungkinkan programmer untuk memodifikasi kode hanya dengan menyeret dan menjatuhkan objek dan menentukan perilaku dan penampilan mereka. VB berasal dari bahasa pemrograman *BASIC* dan dianggap *event-driven* dan berorientasi objek. VB dimaksudkan agar mudah dipelajari dan

cepat untuk menulis kode; Akibatnya, kadang-kadang disebut sistem pengembangan aplikasi cepat (RAD) dan digunakan untuk prototipe aplikasi yang nantinya akan ditulis dalam bahasa yang lebih sulit tetapi efisien (Lee, 2014).

Versi terakhir VB, *Visual Basic* 6, dirilis pada tahun 1998, tetapi sejak itu telah digantikan oleh VB. *NET, Visual Basic for Applications* (VBA) dan *Visual Studio .NET. VBA* dan *Visual Studio* adalah dua kerangka kerja yang paling umum digunakan saat ini. VB adalah alat pengembangan berbasis *GUI* yang menawarkan RAD lebih cepat daripada kebanyakan bahasa pemrograman lainnya. VB juga memiliki fitur sintaksis yang lebih mudah daripada bahasa lain, lingkungan visual yang mudah dipahami dan konektivitas basis data yang tinggi.

2.6.1 Visual Basic.NET

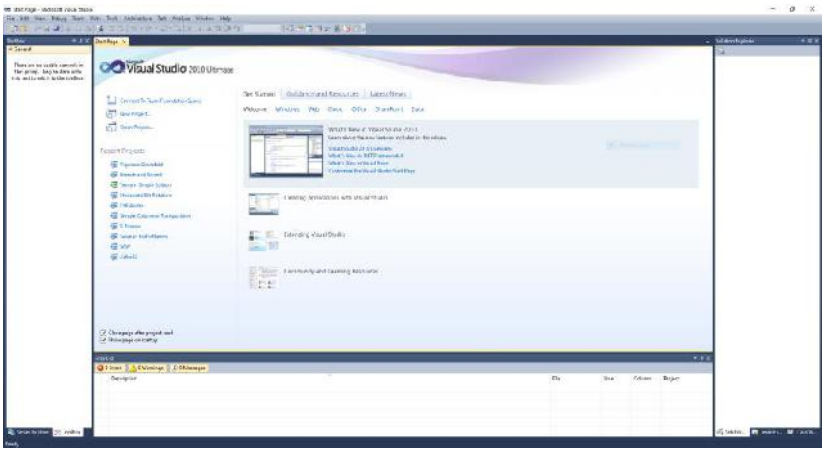
Microsoft Visual Studio merupakan sebuah perangkat lunak lengkap (suite) yang dapat digunakan untuk melakukan pengembangan aplikasi, baik itu aplikasi bisnis, aplikasi personal, ataupun komponen aplikasinya, dalam bentuk aplikasi *console,* aplikasi *Windows,* ataupun aplikasi *Web. Visual Studio* mencakup kompiler, *SDK, Integrated Development Environment* (IDE), dan dokumentasi (umumnya berupa *MSDN Library*). Kompiler yang dimasukkan ke dalam paket *Visual Studio* antara lain *Visual* C++, *Visual* C#, *Visual Basic, Visual Basic.NET, Visual InterDev, Visual* J++, *Visual* J#, *Visual FoxPro,* dan *Visual SourceSafe.*

Microsoft Visual Studio dapat digunakan untuk mengembangkan aplikasi dalam *native code* (dalam bentuk bahasa mesin yang berjalan di atas *Windows*)

ataupun *managed code* (dalam bentuk *Microsoft Intermediate Language* di atas *.NET Framework*). Selain itu, Visual Studio juga dapat digunakan untuk mengembangkan aplikasi *Silverlight*, aplikasi *Windows Mobile* (yang berjalan di atas *.NET Compact Framework*).

2.6.2 Antarmuka *Visual Basic.NET*

Visual Basic.Net memiliki beberapa versi. Berikut ini adalah tampilan dari *Visual Basic.Net versi 2010.*



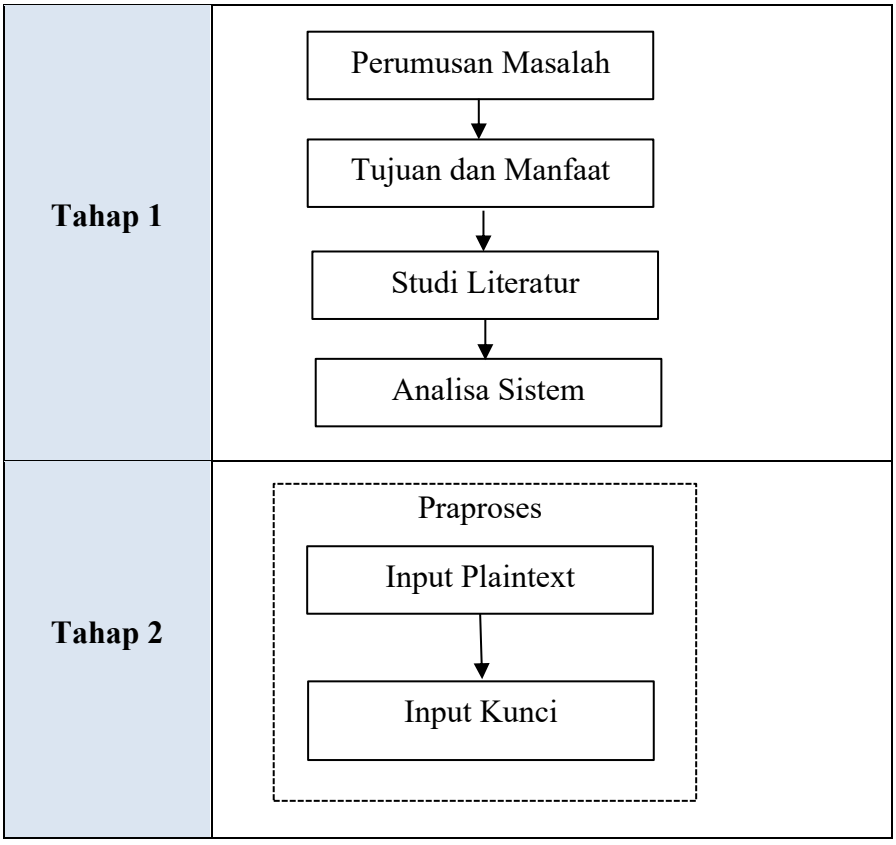
Gambar 2.9 Antarmuka *Visual Basic.NET 2010*
Sumber: (Rahmel, 2008)

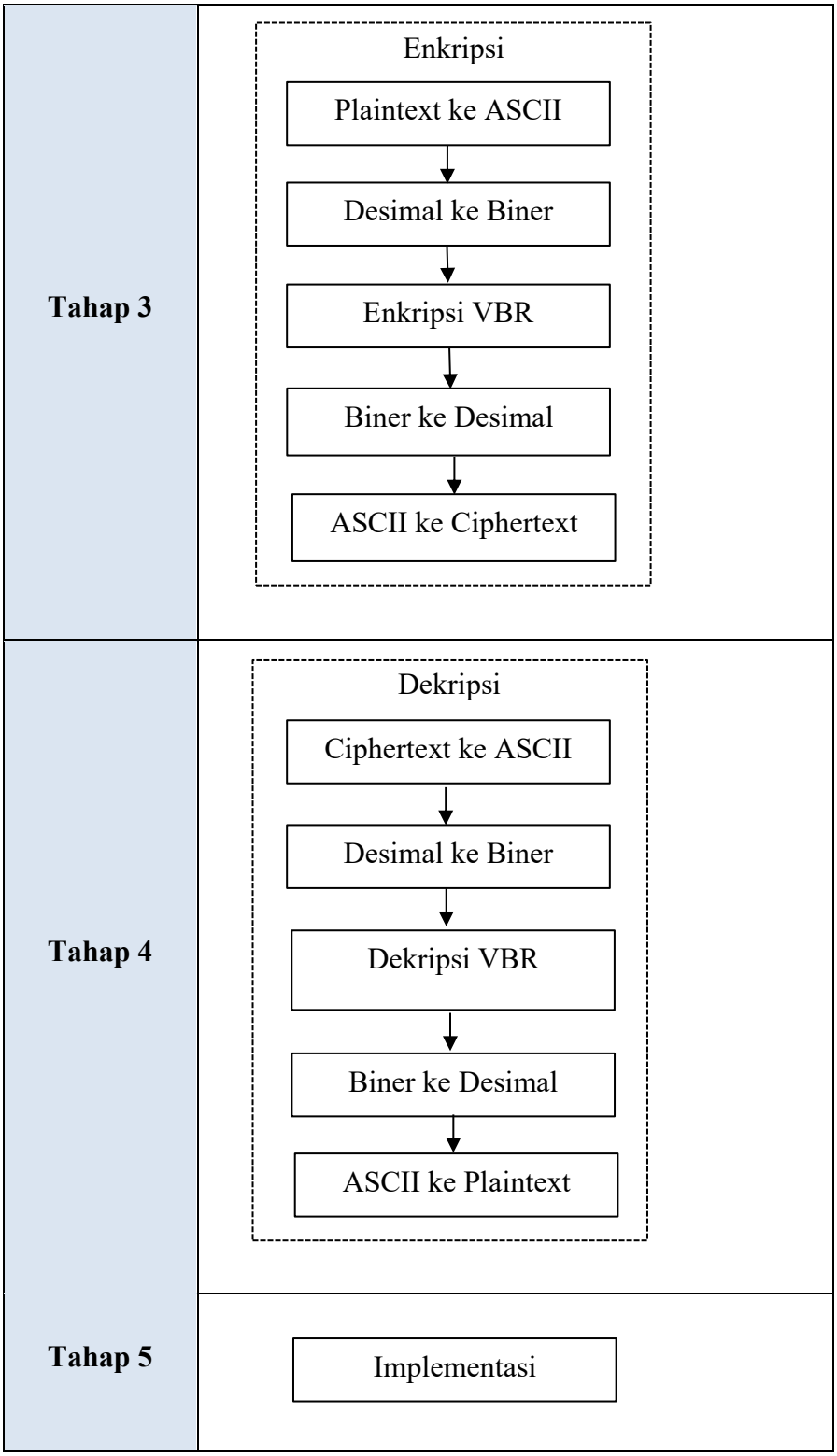
BAB III

METODE PENELITIAN

3.1 Tahapan Penelitian

Tahapan penelitian adalah penggambaran metode secara terstruktur dalam melakukan analis dan pengumpulan data. Tahapan ini dilakukan untuk memberi gambaran penelitian dan program aplikasi yang akan dibuat. Tahapan penelitian berfungsi untuk menjelaskan struktur program aplikasi *Vertical Bit Rotation*. Ada beberapa tahapan yang dilakukan dalam menyelesaikan peneltian ini. Gambar 3.1adalah tahapan penelitian yang dilakukan.





Gambar 3.1 Tahapan Penelitian

Berikut adalah tahapan penelitian yang dilakukan:

1. Perumusan Masalah

   Tahap ini melakukan penentuan rumusan masalah dan bagaimana permasalahan tersebut dikerjakan dalam penelitian ini.

2. Tujuan dan Manfaat

   Tahap ini bertujuan untuk menentukan tujuan dan manfaat dari penelitian yang diperoleh sebelum dan sesudah penelitian dikerjakan.

3. Studi Literatur

   Tahap studi literatur dilakukan untuk mendapatkan teori-teori tentang ilmu kriptografi tentang pergeseran bit secara horizontal. Studi ini dapat dilakukan melalui buku dan informasi yang ada di internet.

4. Analisa Sistem

   Tahap analisa sistem menerangkan proses penguraian dan bagaimana rumusan maslaah dapat diselesaikan dengan cara terstruktur. Tahap ini menentukan bagaimana implementasi dari rancangan yang akan telah ditentukan.

5. Praproses

   Tahap ini berfungsi untuk memproses data awal yang akan dikerjakan sistem. Data yang disediakan adalah plaintext dan kunci.

6. Enkripsi

   Tahap ini melakukan proses penyandian karakter terhadap karakter yang sudah mengalamipraproses. Enkripsi dilakukan berdasarkan kunci pada vertical bit rotataion yang telah ditentukan sebelumnya.

7. Dekripsi

   Tahap ini melakukan proses pengembalian ciphertext menjadi plaintext yang sebelumnya sudah mengalami proses enkripsi.

8. Implementasi

   Tahap ini dilakukan pengujian kebenaran hasil program aplikasi. Hasil akan dicocokkan kepada perhitungan manual berdasarkan perhitungan vertical bit rotation.

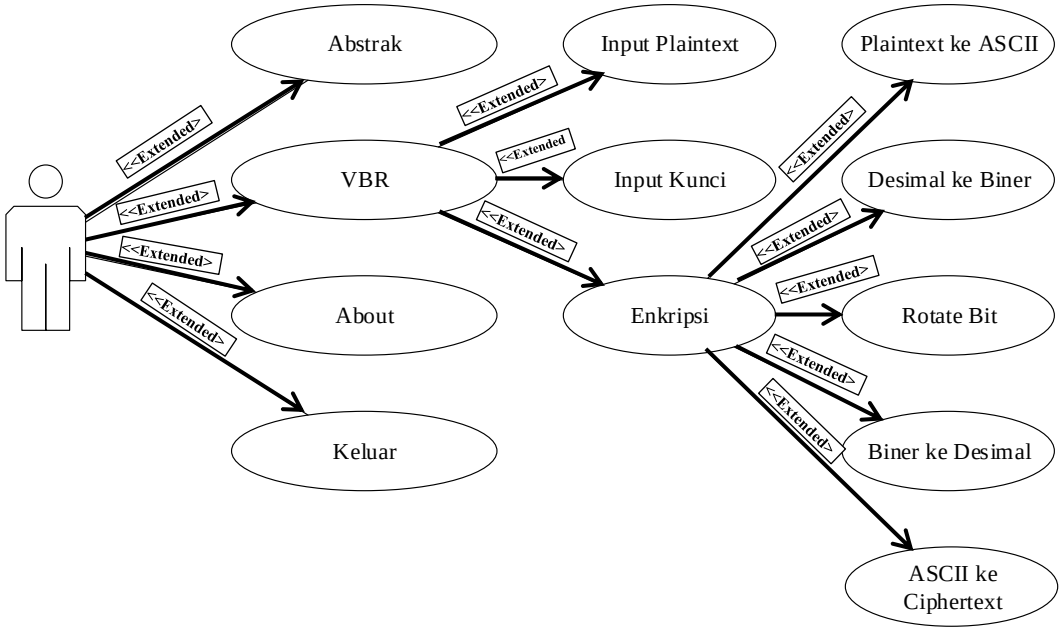
3.2 Perancangan Penelitian

Perancangan penelitian diciptakan untuk memberi gambaran hasil dari program aplikasi yang akan dibuat. Aplikasi tersebut dibuat dengan menggunakan *Microsoft Visual Basic.Net* 2010. Perancangan penelitian dijelaskan dalam bentuk *Unified Modelling Language (UML)*. Ada dua buah diagram yang terlibat dalam menentukan perancangan, yaitu *Use Case Diagram* dan *Activity Diagram*.

3.2.1 Use Case Diagram

Use case adalah daftar tindakan atau langkah-langkah peristiwa yang biasanya mendefinisikan interaksi antara peran aktor dan sistem untuk mencapai tujuan. *Use case* adalah teknik yang berguna untuk mengidentifikasi, mengklarifikasi, dan mengatur persyaratan sistem. Sebuah use case terdiri dari serangkaian kemungkinan interaksi antara sistem dan pengguna yang mendefinisikan fitur yang akan diimplementasikan dan resolusi dari setiap kesalahan yang mungkin terjadi.

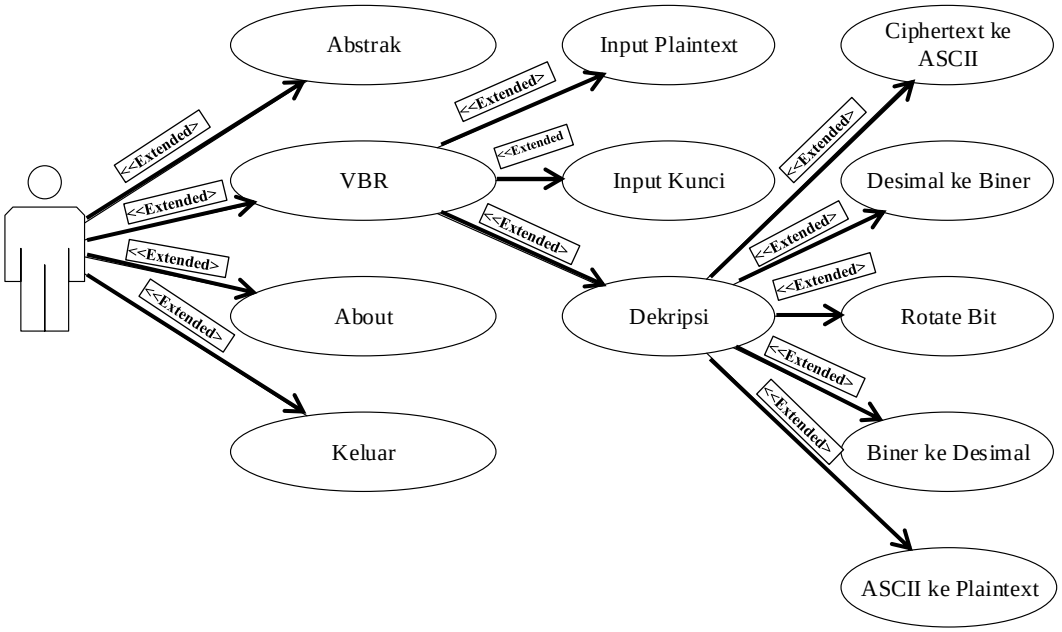
Use Case berfungsi untuk membentuk dari sebuah sistem dari segi pengguna. Diagram ini bekerja dengan cara menjelaskan hubungan dan interaksi antar pengguna dengan sistem melalui sebuah alur yang menjelaskan bagaimana sistem tersebut digunakan. Gambar 3.2 adalah perancangan *Use Case* untuk enkripsi program aplikasi pergeseran bit.



Gambar 3.2 *Use Case* Diagram Enkripsi VBR

Proses enkripsi terjadi setelah plaintext dan kunci telah diinputkan pada *textbox* yang sudah disediakan. Proses ini akan mengkonversi setiap deretan *plaintext* menjadi berbentuk *byte* sesuai dengan tabel ASCII. Bilangan ini kemudian akan dikonversi kembali ke bilangan biner agar dapat digeser secara vertikal. Bilangan biner akan dilakukan proses rotasi secara vertikal sesuai dengan kunci yang telah ditetapkan. Hasil pergeseran akan dikonversi kembali menjadi

desimal yang kemudian akan ditransformasikan kembali ke dalam bentuk ASCII. Deretan ASCII ini akan membentuk sebuah untaian kata yang merupakan *ciphertext* yang dihasilkan.

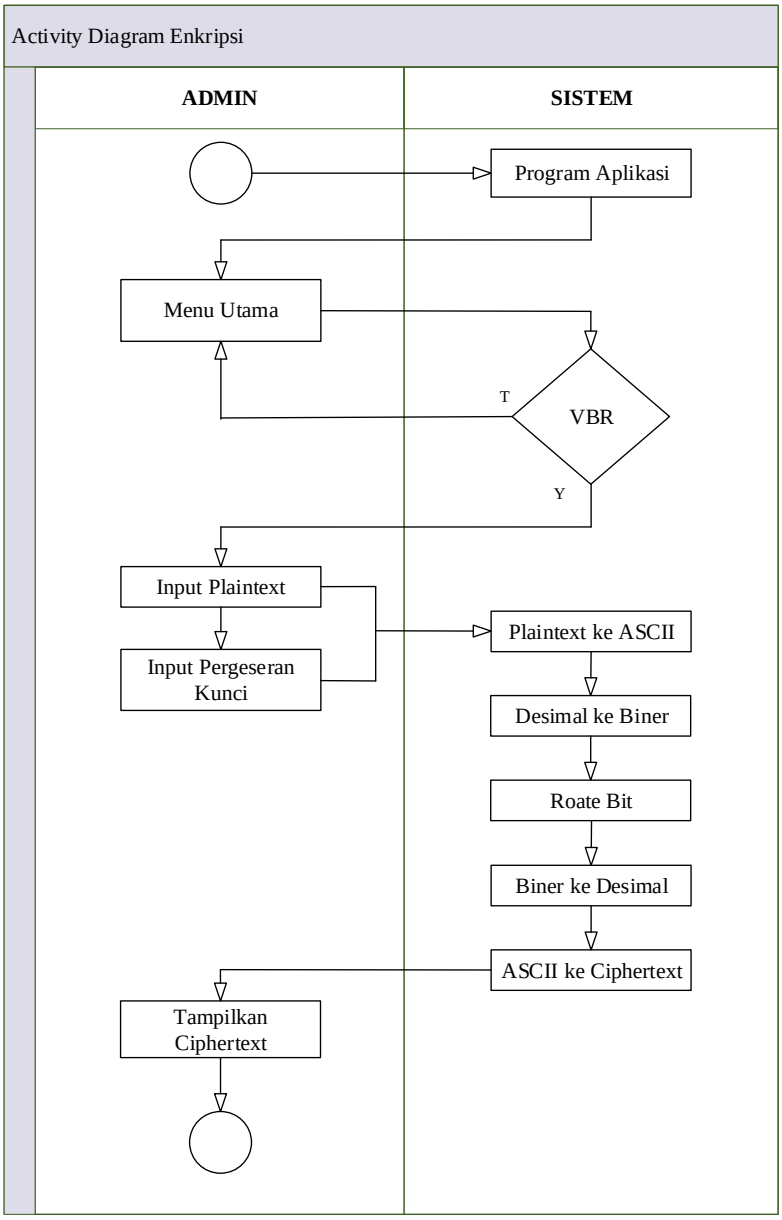


Gambar 3.3 *Use Case* Diagram Dekripsi VBR

Proses dekripsi melakukan cara yang sama dengan apa yang dilakukan pada proses enkripsi. *Ciphertext* akan diubah menjadi byte sesuai tabel ASCII sehingga berbentuk bilangan desimal. Bilangan ini kemudian akan dikonversi kembali ke bilangan biner. Bilangan biner akan dilakukan proses rotasi sebesar kunci yang telah ditentukan. Hasil pergeseran akan dikonversi kembali menjadi desimal yang kemudian akan ditransformasikan kembali ke dalam bentuk ASCII. ASCII ini akan membentuk string yang merupakan *plaintext* dari *ciphertex*t sebelumnya

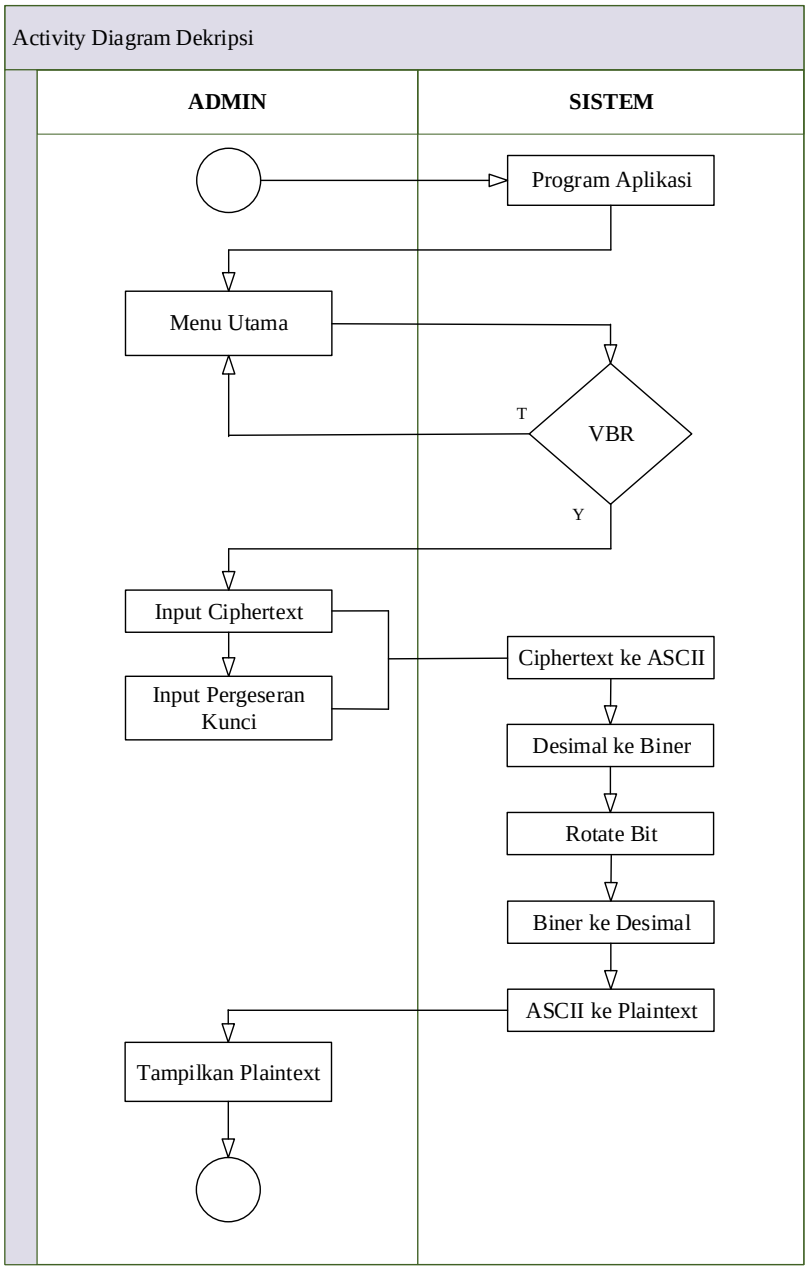
3.2.2 Activity Diagram

Activity diagram akan menggambarkan alur aktifitas dari sistem. Gambar 3.4 akan menjelaskan *Activity diagram* dari algoritma VBR pada proses enkripsi.



Gambar 3.4 *Activity Diagram* Enkripsi VBR

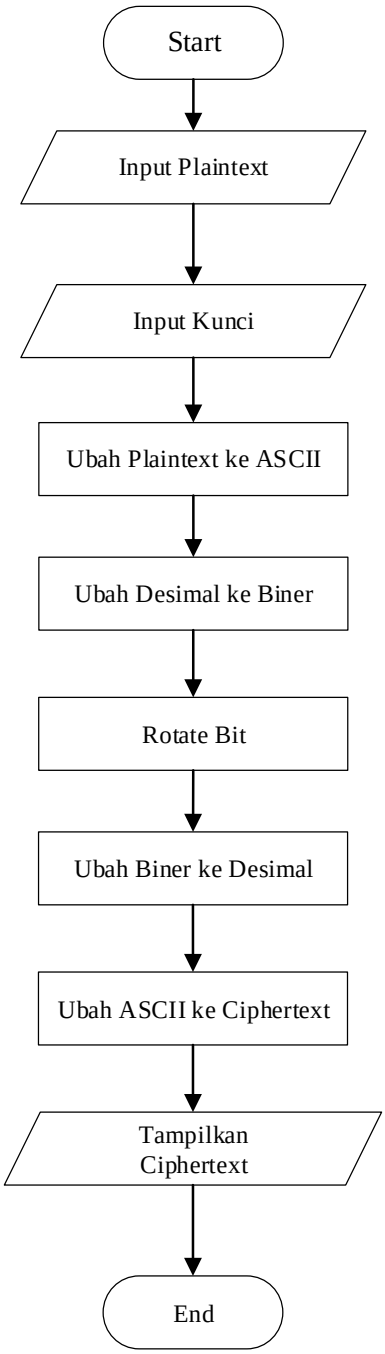
Gambar 3.5 akan menjelaskan *Activity diagram* dari algoritma VBR pada proses dekripsi.



Gambar 3.5 *Activity Diagram* Dekripsi VBR

3.2.3 *Flowchart* Enkripsi

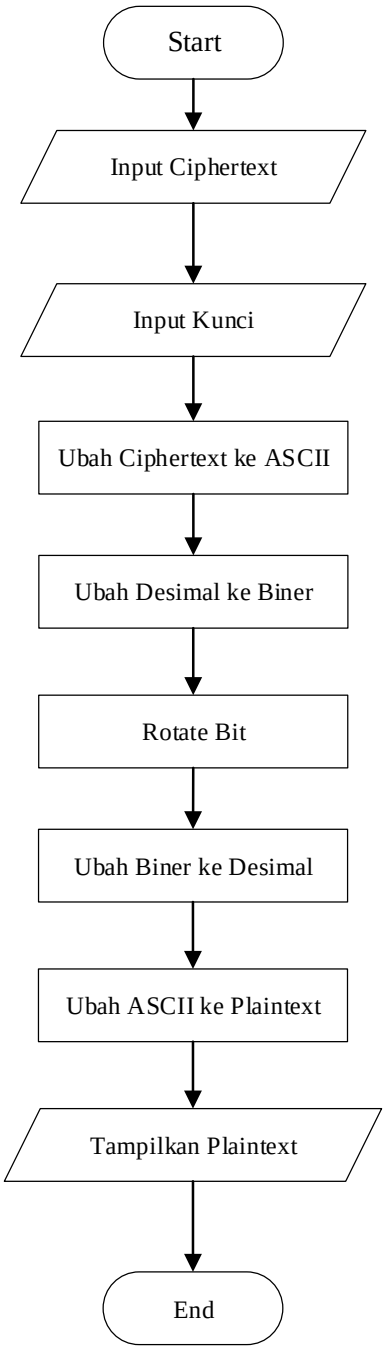
Flowchart enkripsi akan menerangkan alur proses enkripsi dengan algoritma VBR. Ini enkripsi dapat dilihat pada Gambar 3.6.



Gambar 3.6 *Flowchart* enkripsi algoritma VBR

3.2.4 Flowchart Dekripsi

Flowchart dekripsi akan menjelaskan alur dari proses dekripsi dengan algoritma VBR. Flowchart ini dapat dilihat pada Gambar 3.*7*.



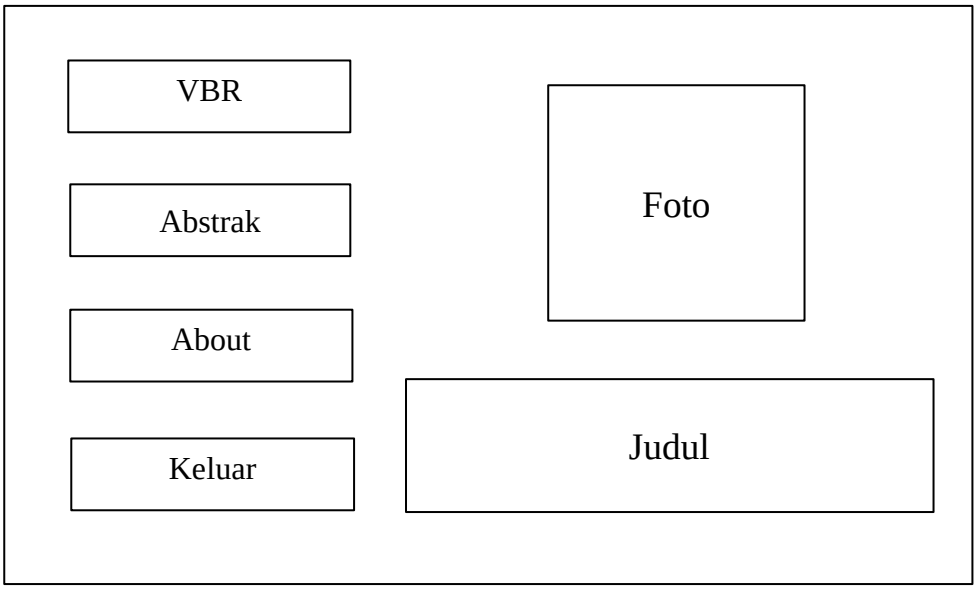
Gambar 3.*7* *Flowchart* dekripsi algoritma VBR

3.3 Desain Interface

Desain *interface* adalah perancangan program aplikasi yang akan dibuat. Kode program akan dibuat menggunakan *Microsoft Visual Basic.Net* 2010. Desain *interface* ini terbagi menjadi beberapa sub-menu dan memiliki sebuah menu utama yang berfungsi sebagai menjalankan program utama. Beberapa desain berikut ini merupakan desain *interface* dari form pada pembuatan program *Vertical Bit Rotation*.

3.3.1 Desain Menu Utama

Menu ini adalah bagian utama dari program aplikasi. Menu ini memiliki beberapa fungsi yang berbeda-beda dalam menjalankan menu-menu yang ada. Gambar 3.8 adalah hasil desain Menu Utama.



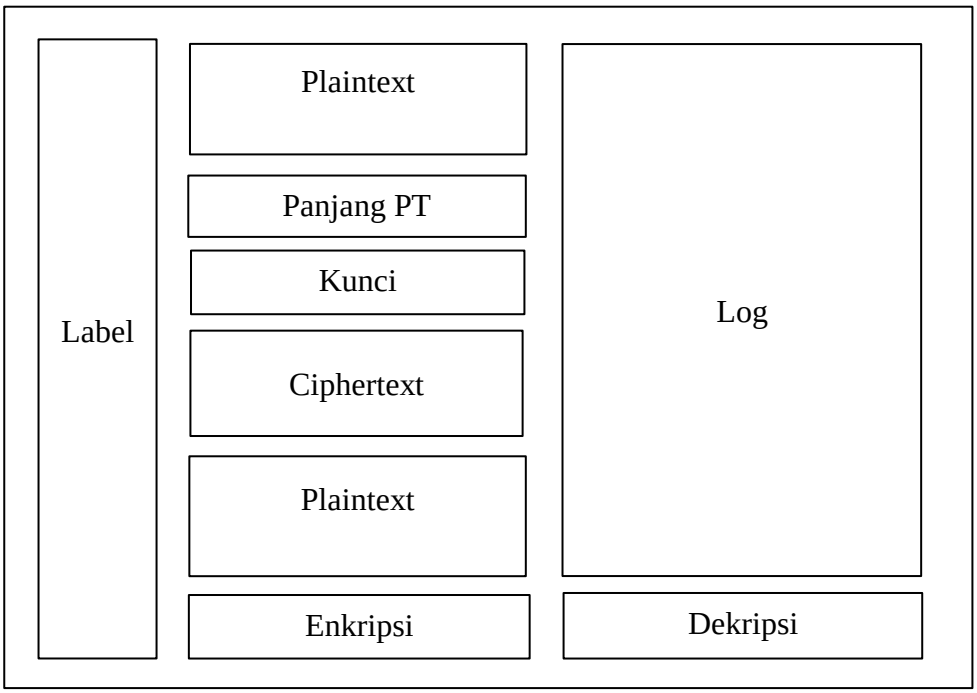
Gambar 3.8 Desain Menu Utama

Menu utama memiliki beberapa komponen antara lain:

1. VBR
2. Info
3. About
4. Keluar
5. Foto
6. Judul

3.3.2 Desain Menu *Vertical Bit Rotation*

Desain ini menjelaskan bentuk dari program aplikasi *Vertical Bit Rotation* akan dirancang. Bagian ini adalah bagian yang terpenting dalam pelaksanaan penelitian ini. Gambar 3.9 adalah tampilan desain algoritma VBR.



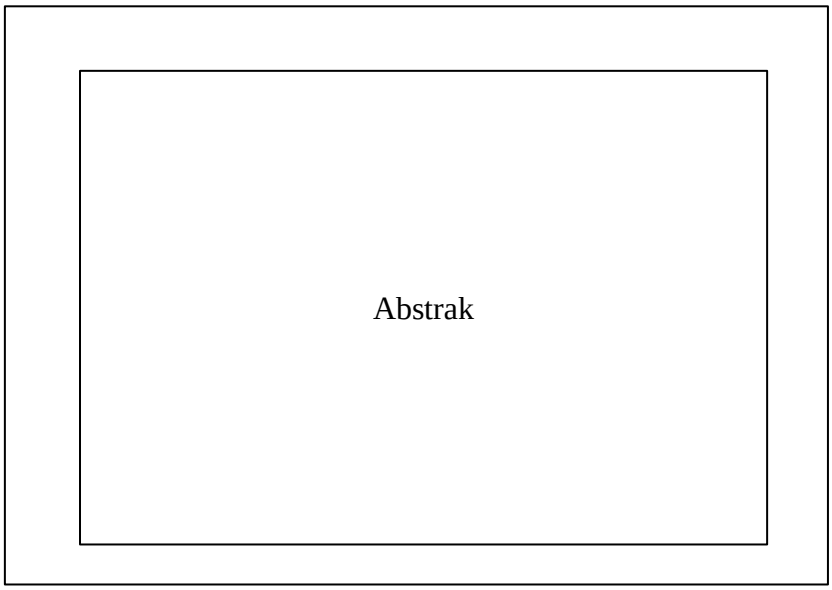
Gambar 3.9 Desain Menu *Vertical Bit Rotation*

Menu *Vertical Bit Rotation* memiliki beberapa bagian antara lain:

1. *Plaintext*
2. Panjang PT
3. Kunci
4. *Ciphertext*
5. Tombol Enkripsi
6. Tombol Dekripsi
7. *Log*

3.3.3 Desain Menu *Abstrak*

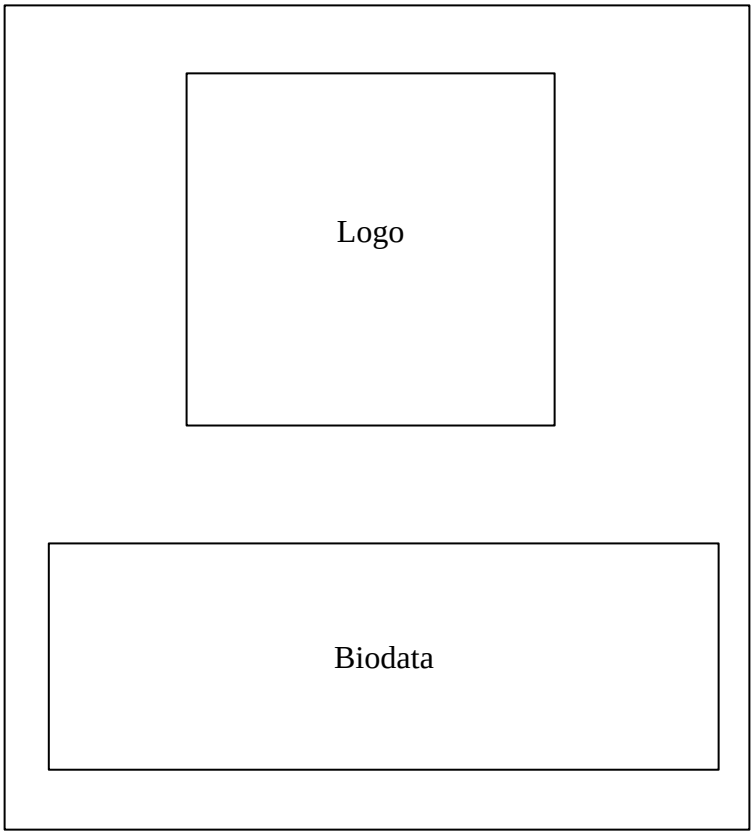
Desain abstrak menampilkan abstrak dari penelitian yang penulis lakukan. *Abstrak* menjelaskan rumusan, hasil dan kesimpulan dari penelitian yang dilakukan. Gambar 3.10 adalah hasil perancangan desain Info.



Gambar 3.10 Desain Menu *Abstrak*

3.3.4 Desain Menu *About*

Desain menu about menampilkan informasi seputar tentang penulis. Menu ini memiliki objek *picturebox* dan *label* yang berfungsi untuk menampilkan gambar logo dari Universitas Pembangunan Panca Budi dan biodata penulis. Gambar 3.11 adalah hasil desain Tentang Saya.



Gambar 3.11 Desain Menu *About*

BAB IV

HASIL DAN PEMBAHASAN

Hasil dan pembahasan merupakan target yang telah dicapai dalam pembuatan program aplikasi menggunakan perangkat lunak. Dalam hasil ini, dipaparkan bagaimana proses perhitungan dari algoritma *Vertical Bit Rotation* bekerja. Tetapi, untuk melaksanakan tahap tersebut, dibutuhkan sistem yang dapat mendukung penelitian tersebut.

Program aplikasi diciptakan berdasarkan ketentuan-ketentuan yang telah dipaparkan pada bab-bab sebelumnya. Untuk mendukung pembuatan program, dibutuhkan suatu sistem yang dapat menjalankan program aplikasi tersebut dengan baik dan lancar. Beberapa kriteria harus dipenuhi dalam menyediakan suatu sistem.

4.1 Spesifikasi Sistem

Spesifikasi sistem menjelaskan persyaratan operasional dan kinerja suatu sistem, seperti komputer. Ini dianggap sebagai dokumen tingkat tinggi yang menentukan fungsi global. Spesifikasi sistem membantu untuk menentukan pedoman operasional dan kinerja untuk suatu sistem. Ini dapat menguraikan bagaimana sistem diharapkan untuk melakukan, dan apa yang mungkin termasuk. Spesifikasi utama dapat mencakup definisi antarmuka, aturan desain dokumen, dan area fungsional. Spesifikasi sistem dapat diuraikan selama proses evaluasi dan disepakati selama proses pengujian.

4.1.1 Spesifikasi Perangkat Keras

Teknik *Vertical Bit Rotation* membutuhkan perangkat keras untuk mendukung pengerjaan program aplikasi. Perangkat keras ini sebagai sarana utama. Tabel 4.1 adalah spesifikasi perangkat keras yang dibutuhkan dalam melakukan penelitian ini.

Tabel 4.1 Spesifikasi perangkat keras

No.	Nama Komponen	Spesifikasi
1	Processor	Intel Core i5 2.4 GHz
2	RAM	4096 MB
3	Storage	500 GB
4	Display	14 inches

4.1.2 Spesifikasi Perangkat Lunak

Perangkat lunak tidak ketinggalan dalam membantu perangkat keras agar dapat beroperasi dengan baik. Setiap perangkat keras membutuhkan sistem operasi dan program aplikasi pendukung agar dapat digunakan sesuai dengan tujuan. Kebutuhan perangkat lunak ini bertujuan sebagai sarana non-fisik untuk menghasilkan program aplikasi VBR tersebut. Tabel 4.2 adalah spesifikasi perangkat lunak yang digunakan pada penelitian ini.

Tabel 4.2 Spesifikasi perangkat lunak

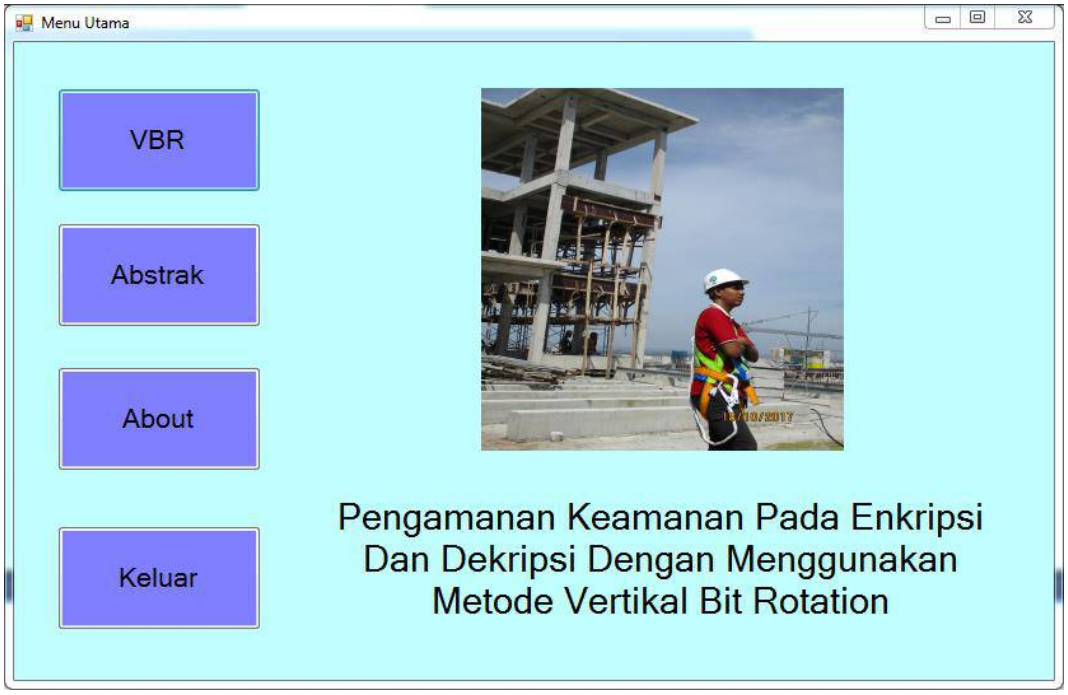
No.	Nama Komponen	Spesifikasi
1	Sistem Operasi	Windows 10 64 Bit
2	IDE Pemrograman	Microsoft Visual Basic.NET 2010
3	Tangkap Gambar	Snipping Tool
4	Data Editor	Microsoft Excel

4.2 Implementasi Antarmuka

Penerapan antarmuka berfungsi untuk mengubah perancanganmenjadi program aplikasi. Proses implementasi memiliki langkah-langkah umum dalam menciptakan hasil yang baik. Beberapa syarat harus dilakukan dalam membuat program aplikasi seperti kode program harus dilaksanakan dengan baik. Pengujian terhadap antarmuka sangat harus diterapkan agar tidak terjadi kekeliruan yang membuat hasil program aplikasi tersebut menyimpang. Pemeriksaan lebih lanjut dibutuhkan untuk menentukan hasil program aplikasi yang dibuat. Uji coba harus dilakukan terhadap antarmuka dan terhadap hasil *output* dari program aplikasi dalam menghitung proses enkripsi dan dekripsi pada kriptografi VBR.

4.2.1 Implementasi Menu Utama

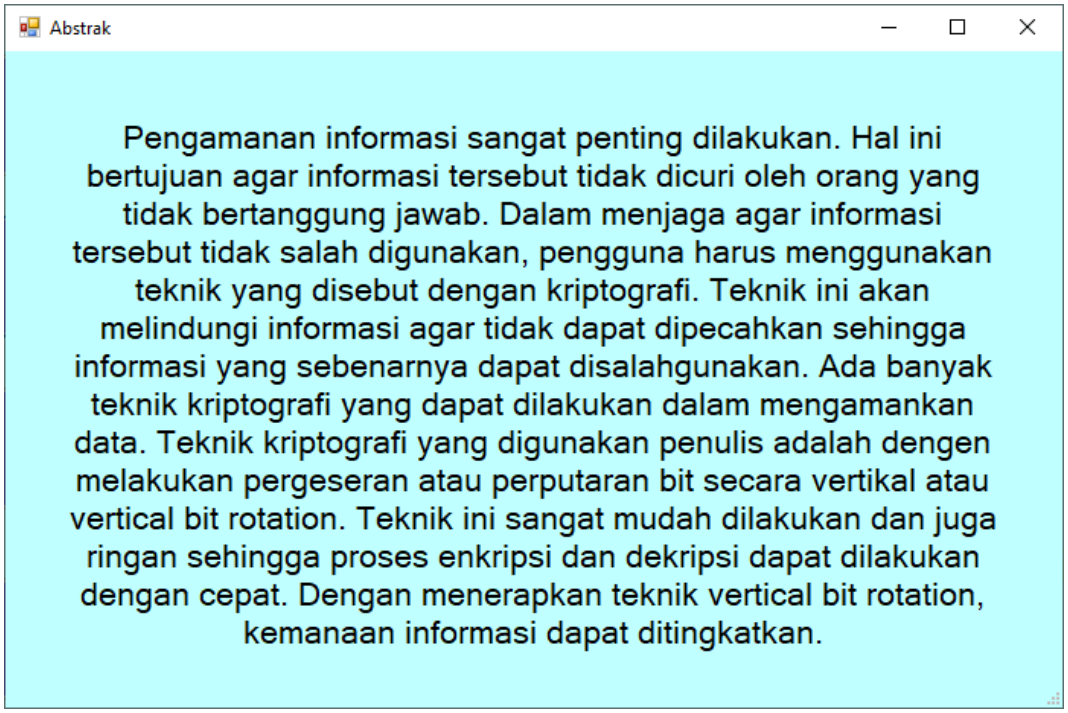
Implementasi Menu Utama berfungsi untuk menampilkan menu utama yang akan membawa pengguna untuk bernavigasi ke menu-menu lainnya. Menu ini akan muncul pada saat program aplikasi dijalankan. Ada beberapa bagian yang akan tampil sebagai alat pemilih dari menu-menu lainnya. Pengguna akan memilih menu mana yang akan mereka tuju. Menu utama bertujuan untuk memberikan pilihan fungsi yang ada pada *Vertical Bit Rotation* tersebut. Menu utama pada penelitian terdiri dari tiga buah sub-menu dan satu buah tombol untuk keluar dari aplikasi tersebut. Gambar 4.1 adalah hasil implementasi Menu Utama.



Gambar 4.1 Implementasi Menu Utama

4.2.2 Implementasi Menu *Abstrak*

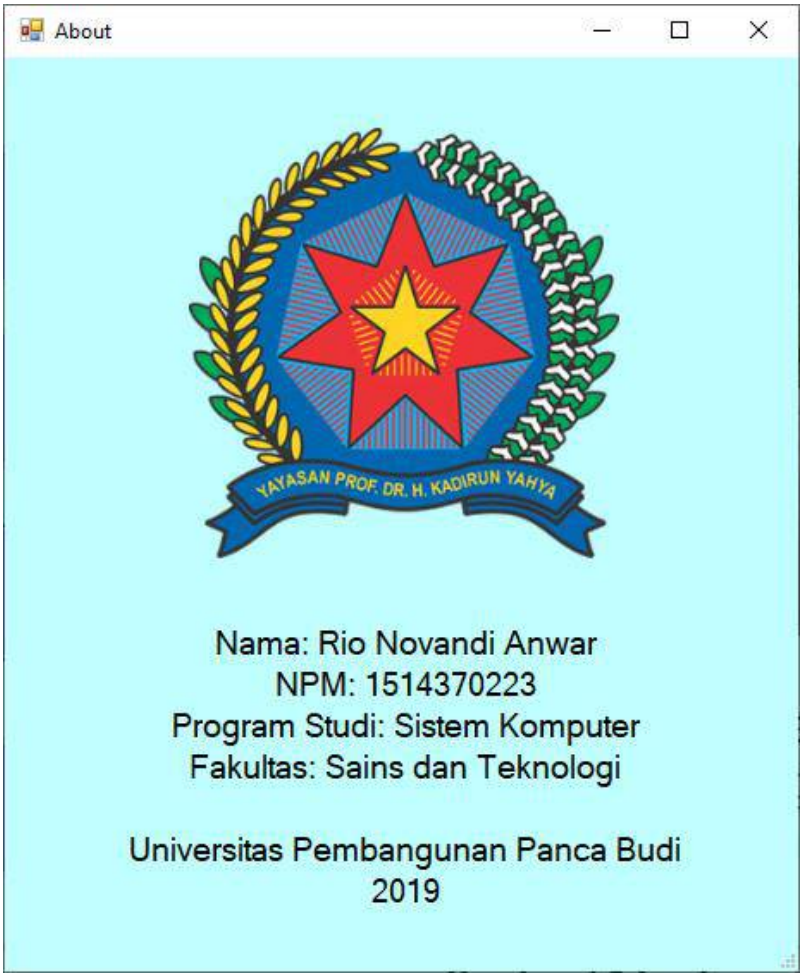
Halaman *abstrak* adalah menu yang menampilkan abstrak dari penelitian ini. Pada *abstrak* akan dijelaskan secara singkat hasil dari penelitian yang sudah dilakukan. Rumusan, tujuan dan mafaat penelitian akan dipaparkan pada bagian awal dari *abstrak*. Pada bagian tengah akan dijelaskan hasil yang akan dicapai. Tidak ketinggalan penjelasan tentang kesimpulan yang diperoleh setelah hasil program aplikasi dijakankan. Gambar 4.2 adalah hasil implementasi dari Info.



Gambar 4.2 Implementasi *Abstrak*

4.2.3 Implementasi Menu *About*

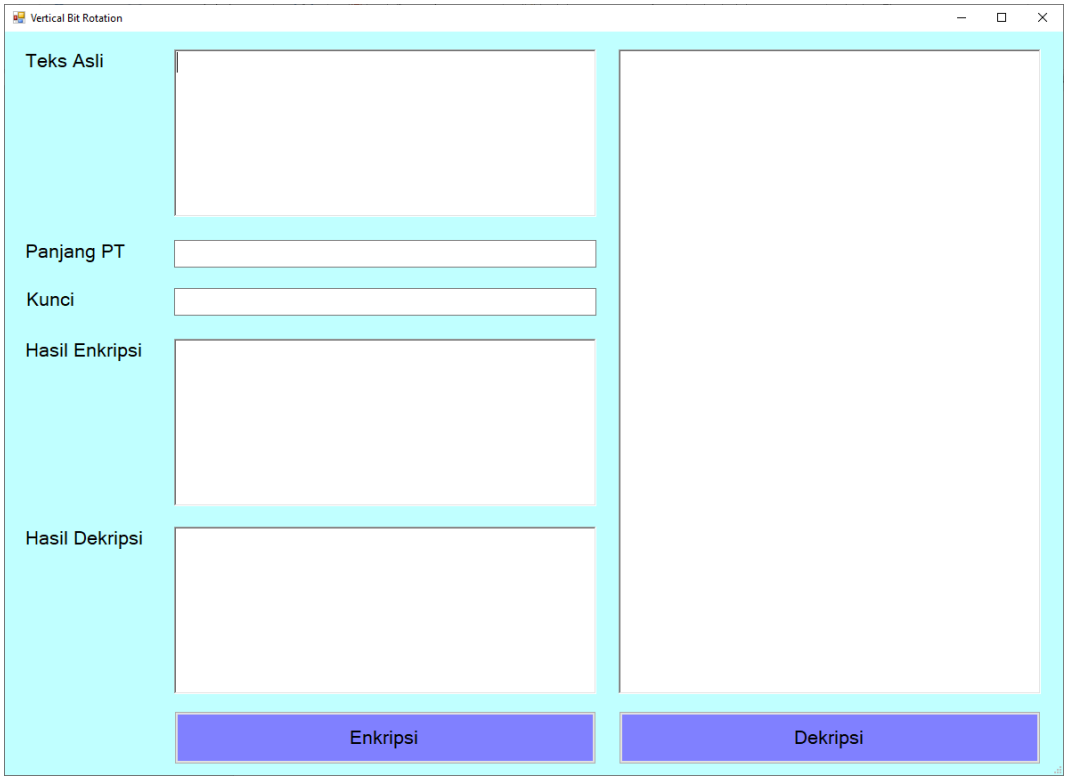
Halaman *About* adalah tampilan seputar keterangan mengenai penulis. Halaman ini menampilkan nama, NPM, fakutas, program studi dan universitas. Gambar 4.3 adalah tampilan dari implementasi Tentang Saya.



Gambar 4.3 Implementasi Tentang Saya

4.2.4 Implementasi Menu *Vertical Bit Rotation*

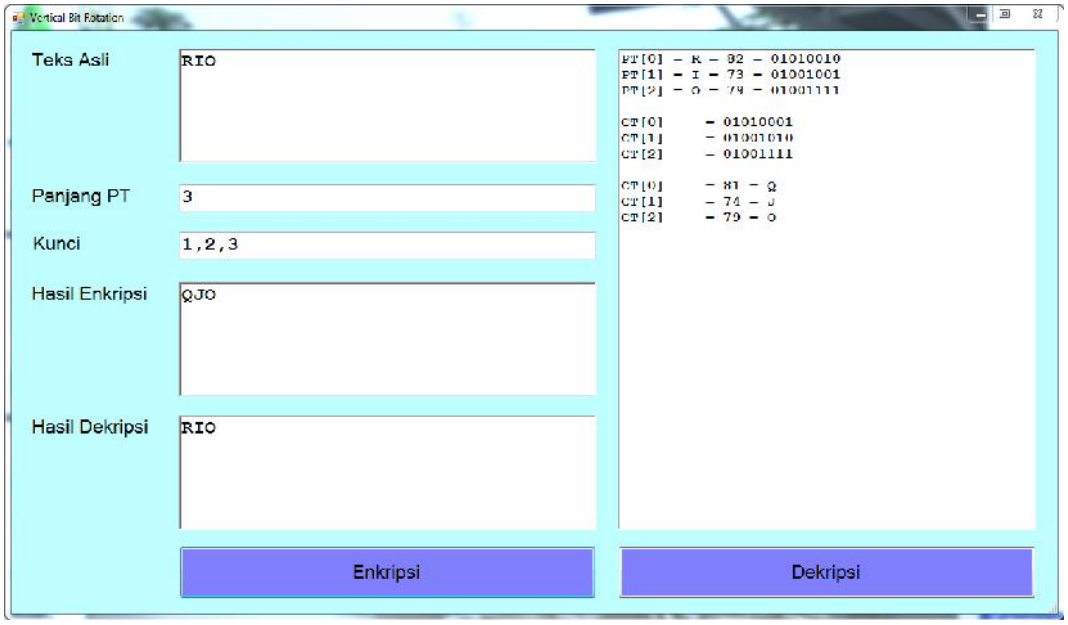
Halaman ini adalah bagian utama yang melakukan enkripsi dan dekripsi. Halaman ini juga menampilkan perhitungan lengkap teknik *Vertical Bit Rotation*. Halaman terdiri dari beberapa buah *textbox* yang terdiri dari *plaintext*, kunci dan *ciphertext*. Pada proses enkripsi dan dekripsi, halaman ini memiliki beberapa tombol eksekusi. Gambar 4.4 adalah hasil tampilan dari implementasi *Vertical Bit Rotation*.



Gambar 4.4 Implementasi Kriptografi VBR

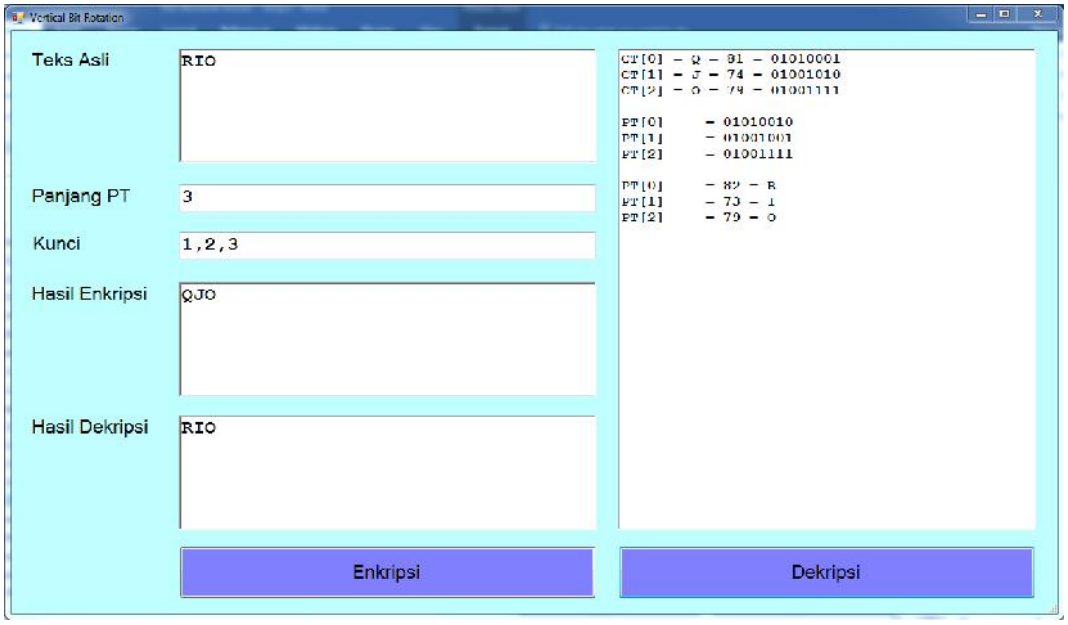
4.2.5 Hasil Perhitungan Kriptografi VBR

Halaman ini menjelaskan perhitungan teknik *Vertical Bit Rotation* yang dikerjakan oleh program aplikasi. Proses yang terlibat adalah enkripsi dan dekripsi. *Plaintext* dan Kunci adalah dua bagian yang harus diisi untuk menentukan *ciphertext*. Kedua nilai dari parameter ini akan diproses sehingga menghasilkan *ciphertext*. Setiap karakter pada *plaintext* akan dilakukan pergeseran kunci secara vertikal untuk mendapatkan *ciphertext*. Apabila pergeseran *plaintext* dan kunci melebihi dari jumlah *plaintext*, hasil *ciphertext* akan bermula kembali dari awal karena proses modulo. Gambar 4.5 adalah tampilan dari hasil perhitungan proses enkripsi *Vertical Bit Rotation*.



Gambar 4.5 Hasil perhitungan enkripsi VBR

Proses dekripsi dilakukan dengan membalikkan proses rotasi *vertikal* ke arah semula. Apabila pada saat proses enkripsi, kunci dilakukan dengan menggeser ke bawah, pada saat dekripsi dapat dilakukan menggeser kembali ke atas. Karena sistem pegeseran kunci adalah rotasi, maka kunci harus dilakukan proses modulo untuk membatasi agar hasil tidak kurang dari nol. Proses dekripsi akan melakukan pergeseran bit secara *vertikal* pada setiap bit pada sepanjang *plaintext*. Gambar 4.6 adalah tampilan dari hasil perhitungan proses dekripsi dengan teknik *Vertical Bit Rotation.*



Gambar 4.6 Hasil perhitungan dekripsi VBR

4.3 Pengujian Perhitungan

Bagian ini akan dilakukan perhitungan secara manual untuk mendapatkan hasil apakah sesuai dengan program aplikasi. Pengujian melibatkan sebuah plaintext dan kunci untuk diproses dengan enkripsi dan dekripsi. Berikut ini adalah proses lengkap perhitungan enkripsi dan dekripsi.

Plaintext = RIO NOVANDI ANWAR

Kunci = 1,2,3

Hasil Enkripsi

```
PT[0] = R = 82 = 01010010
PT[1] = I = 73 = 01001001
PT[2] = O = 79 = 01001111

CT[0]     = 01010001
CT[1]     = 01001010
CT[2]     = 01001111

CT[0]     = 81 = Q
CT[1]     = 74 = J
CT[2]     = 79 = O
```

Ciphertext = QJO

Kunci = 1,2,3

Hasil Dekripsi

```
CT[0] = Q = 81 = 01010001
CT[1] = J = 74 = 01001010
CT[2] = O = 79 = 01001111

PT[0]     = 01010010
PT[1]     = 01001001
PT[2]     = 01001111

PT[0]     = 82 = R
PT[1]     = 73 = I
PT[2]     = 79 = O
```

BAB V

PENUTUP

5.1 Kesimpulan

Penulis dapat menarik beberapa kesimpulan berdasarkan hasil pengujian yang dilakukan setelah melakukan penelitian. Adapun kesimpulan yang diperoleh adalah antara lain:

1. *Vertical Bit Rotation* melakukan pertukaran bit secara vertikal sesuai dengan kunci yang diberikan pada setiap bit.
2. Kunci dapat diterapkan pada bit-bit tertentu saja tanpa harus menerapkan kunci untuk ke delapan bit pada setiap karakter.
3. Kunci akan dimodulo sebanyak panjang *plaintext* untuk melakukan proses rotasi ke awal *plaintext*.
4. Jumlah maksimal kunci adalah 7.

5.2 Saran

Penelitian juga meiliki kekurangan. Terdapat beberapa saran yang dapat penulis kemukakan untuk meningkatkan kualitas penelitian ini. Adapun saran tersebut adalah antara lain:

1. Teknik *Vertical Bit Rotation* sebaiknya dikombinasikan dengan teknik stream *cipher* agar mengurangi peluang perusakan *ciphertext*.
2. Keamanan teknik *Vertical Bit Rotation* dapat ditingkatkan dengan menambahkan skema three-pass protocol pada penerapan kunci sehingga

menghindari adanya pendistribusian kunci atau pertukaran kunci dalam proses enkripsi dan dekripsi.

DAFTAR PUSTAKA

Ariyus, D. (2008). *Pengantar Ilmu Kriptografi: Teori Analisis & Implementasi*. Yogyakarta: Andi Offset.

Edraw. (2019). What is Algorithm - Definition, Types and Application. Retrieved October 27, 2019, from https://www.edrawsoft.com/algorithm-definition.php

EDUCBA. (2017). What is Cryptography? | Types and Advantages of

Cryptography. Retrieved October 23, 2019, from https://www.educba.com/what-is-cryptography/

Fachri, barany, agus perdana windarto, and ikhsan parinduri. "penerapan backpropagation dan analisis sensitivitas pada prediksi indikator terpenting perusahaan listrik." jepin (jurnal edukasi dan penelitian informatika) 5.2 (2019): 202-208.

Fachri, b., windarto, a. P., & parinduri, i. (2019). Penerapan backpropagation dan analisis sensitivitas pada prediksi indikator terpenting perusahaan listrik. Jepin (jurnal edukasi dan penelitian informatika), 5(2), 202-208.

Fachri, barany; windarto, agus perdana; parinduri, ikhsan. Penerapan backpropagation dan analisis sensitivitas pada prediksi indikator terpenting perusahaan listrik. Jepin (jurnal edukasi dan penelitian informatika), 2019, 5.2: 202-208.

Firmansyah, E. R. (2012). Algoritma Kriptografi & Contohnya. *Universitas Islam Negeri Syarif Hidayatullah Jakarta*.

Hamdi, nurul. "model penyiraman otomatis pada tanaman cabe rawit berbasis programmable logic control." jurnal ilmiah core it: community research information technology 7.2 (2019).

Hidayat, A. (2012). Algoritma Kriptografi Vigenere Cipher. Retrieved November 4, 2019, from https://arfianhidayat.com/algoritma-kriptografi-vigenere-cipher

Kurniawan, T. A. (2018). Pemodelan Use Case (UML): Evaluasi Terhadap beberapa Kesalahan dalam Praktik. *Jurnal Teknologi Informasi Dan Ilmu Komputer*, *5*(1), 77. https://doi.org/10.25126/jtiik.201851610

Ladjamudin, A.-B. bin. (2005). *Analisis dan Desain Sistem Informasi*. Yogyakarta: Graha Ilmu.

Lee, C. (2014). *Buku Pintar Pemrograman Visual Basic 2010*. Jakarta: Elex Media Komputindo.

Mollin, R. A. (2001). *An Introduction to Cryptography, Second Edition* (2nd ed.). Chapman & Hall / CRC Press.

Munir, R. (2006). *Kriptografi*. Bandung: Informatika.

Nakatsu, R. T. (2009). *Reasoning with Diagrams : Decision-Making and Problem-Solving with Diagrams*. John Wiley & Sons.

Permana, aminuddin indra. "kombinasi algoritma kriptografi one time pad dengan generate random keys dan vigenere cipher dengan kunci em2b." (2019).

Putra, randi rian. "sistem informasi web pariwisata hutan mangrove di kelurahan belawan sicanang kecamatan medan belawan sebagai media promosi." jurnal ilmiah core it: community research information technology 7.2 (2019).

Putra, randi rian, et al. "decision support system in selecting additional employees using multi-factor evaluation process method." (2019).

Putra, randi rian. "implementasi metode backpropagation jaringan saraf tiruan dalam memprediksi pola pengunjung terhadap transaksi." jurti (jurnal teknologi informasi) 3.1 (2019): 16-20.

Rahmel, D. (2008). *Visual Basic.NET*. New York: McGraw-Hill.

Saputra, muhammad juanda, and nurul hamdi. "rancang bangun aplikasi sejarah kebudayaan aceh berbasis android studi kasus dinas kebudayaan dan pariwisata aceh." journal of informatics and computer science 5.2 (2019): 147-157

Sidik, a. P., efendi, s., & suherman, s. (2019, june). Improving one-time pad algorithm on shamir's three-pass protocol scheme by using rsa and elgamal algorithms. In journal of physics: conference series (vol. 1235, no. 1, p. 012007). Iop publishing.

Sitepu, n. B., zarlis, m., efendi, s., & dhany, h. W. (2019, august). Analysis of decision tree and smooth support vector machine methods on data mining. In journal of physics: conference series (vol. 1255, no. 1, p. 012067). Iop publishing.

Stallings, W. (2013). *Cryptography and Network Security: Principles and Practice*. New Jersey: Prentice Hall Press.

Sukmawati, R., & Priyadi, Y. (2019). Perancangan Proses Bisnis Menggunakan UML Berdasarkan Fit/Gap Analysis Pada Modul Inventory Odoo. *INTENSIF: Jurnal Ilmiah Penelitian Dan Penerapan Teknologi Sistem Informasi*, *3*(2), 104. https://doi.org/10.29407/intensif.v3i2.12697

Sumandri. (2017). Studi Model Algoritma Kriptografi Klasik dan Modern. *SEMINAR MATEMATIKA DAN PENDIDIKAN MATEMATIKA*.

Tasril, v., wijaya, r. F., & widya, r. (2019). Aplikasi pintar belajar bimbingan dan konseling untuk siswa sma berbasis macromedia flash. Jurnal informasi komputer logika, 1(3).

Wagner, N. R. (2003). *The Laws of Cryptography with Java Code*.

Wasserkrug, S., Dalvi, N., Munson, E. V., Gogolla, M., Sirangelo, C., Fischer-Hübner, S., … Snodgrass, R. T. (2009). Unified Modeling Language. In

Encyclopedia of Database Systems (pp. 3232–3239). https://doi.org/10.1007/978-0-387-39940-9_440

Weerasinghe, T. D. B. (2013). An effective RC4 stream cipher. *2013 IEEE 8th International Conference on Industrial and Information Systems*, 69–74. https://doi.org/10.1109/ICIInfS.2013.6731957

Yakub. (2012). *Pengantar Sistem Informasi*. Yogyakarta: Graha Ilmu.