**KEMENTERIAN PENDIDIKAN DAN KEBUDAYAAN**

**SEKOLAH TINGGI MANAJEMEN INFORMATIKA DAN KOMPUTER**

**PALCOMTECH**

**SKRIPSI**

# PENGGABUNGAN ALGORITMA XOR, ROT 47 DAN BASE64 UNTUK PENINGKATAN KEAMANAN DATA


**Diajukan Oleh:**

**1. AMILIUS PRATAMA / 011160060**

**2. TIRA SUNATA MJ / 011160009**


**Untuk Memenuhi Sebagian dari Syarat**

**Mencapai Gelar Sarjana Komputer**

**PALEMBANG**

**2020**

**KEMENTERIAN PENDIDIKAN DAN KEBUDAYAAN**

**SEKOLAH TINGGI MANAJEMEN INFORMATIKA DAN KOMPUTER**

**PALCOMTECH**

**SKRIPSI**

# PENGGABUNGAN ALGORITMA XOR, ROT 47 DAN BASE 64 UNTUK PENINGKATAN KEAMANAN DATA


**Diajukan Oleh:**

**1. AMILIUS PRATAMA / 011160060**

**2. TIRA SUNATA MJ / 011160009**


**Untuk Memenuhi Sebagian dari Syarat**

**Mencapai Gelar Sarjana Komputer**

**PALEMBANG**

**2020**

**KEMENTERIAN PENDIDIKAN DAN KEBUDAYAAN**
**SEKOLAH TINGGI MANAJEMEN INFORMATIKA DAN KOMPUTER**
**PALCOMTECH**

---

## HALAMAN PENGESAHAN PEMBIMBING SKRIPSI

| | |
|---|---|
| **NAMA / NPM** | **: 1. AMILIUS PRATAMA / 011160060** |
| | **: 2. TIRA SUNATA MJ / 011160009** |
| **PROGRAM STUDI** | **: S1 INFORMATIKA** |
| **JENJANG PENDIDIKAN** | **: STRATA SATU ( S1 )** |
| **JUDUL SKRISI** | **: PENGGABUNGAN ALGORITMA XOR, ROT 47 DAN BASE 64 UNTUK PENINGKATAN KEAMANAN DATA** |

**Tanggal : 29 Januari 2020** **Mengetahui,**

**Pembimbing,** **Ketua,**

**Alfred Tenggono, S.Kom., M.Kom.** **Benedictus Effendi, S.T., M.T.**
**NIDN : 0205108901** **NIP : 09.PCT.13**

**KEMENTERIAN PENDIDIKAN DAN KEBUDAYAAN**
**SEKOLAH TINGGI MANAJEMEN INFORMATIKA DAN KOMPUTER**
**PALCOMTECH**

---

**HALAMAN PENGESAHAN PENGUJI SKRIPSI**

| | |
|---|---|
| **NAMA / NPM** | **: 1. AMILIUS PRATAMA / 011160060** |
| | **: 2. TIRA SUNATA MJ / 011160009** |
| **PROGRAM STUDI** | **: S1 INFORMATIKA** |
| **JENJANG PENDIDIKAN** | **: STRATA SATU ( S1 )** |
| **JUDUL** | **: PENGGABUNGAN ALGORITMA XOR, ROT 47 DAN BASE 64 UNTUK PENINGKATAN KEAMANAN DATA** |

**Tanggal : 20 Februari 2020** **Tanggal : 21 Februari 2020**

**Penguji 1,** **Penguji 2,**

**Guntoro Barovih, S.Kom., M.Kom.** **Surahmat, S.Kom., M.Kom.**
**NIDN : 0201048601** **NIDN : 0217058703**

**Menyetujui,**
**Ketua,**

**Benedictus Effendi, S.T., M.T.**
**NIP : 09.PCT.13**

# *ABSTRACT*

AMILIUS PRATAMA, TIRA SUNATA MJ. *Incorporation of Xor, Rot47 and Base64 Algorithms for Increased Data Security*

*Data confidentiality and security are important for a company or agency or an individual, moreover information from that data can cause risks that can be detrimental such as leakage of sensitive information. Of the problems that arise to overcome data leakage, there are several ways to get around to secure information to be protected, one with a technique called cryptography. Cryptography is the study of how to protect data when it is sent from one place to another. In this study the authors developed several algorithms, the results can later be used to protect important information data by combining 3 cryptographic algorithms namely XOR, Rot47 and Base64, so it can be difficult for those who are not responsible for accessing confidential data. The author implements a merging algorithm on a web-based system and can be implemented on other devices, in this research, namely android.*

***Keyword : Merge algorithm, Cryptography, XOR, ROT47, Base64, Data.***

# ABSTRAK

AMILIUS PRATAMA, TIRA SUNATA MJ. Penggabungan Algoritma Xor, Rot 47 dan Base 64 Untuk Peningkatan Keamanan Data.

Kerahasiaan dan keamanan data merupakan hal yang penting bagi suatu perusahaan atau instansi maupun individu, apalagi informasi dari data tersebut bisa menimbulkan resiko yang dapat merugikan seperti kebocoran informasi sensitif. Dari permasalahan yang timbul untuk mengatasi kebocoran data, ada beberapa cara menyiasati untuk mengamankan informasi yang akan dilindungi, salah satu dengan teknik disebut kriptografi. Kriptografi yaitu ilmu yang mempelajari bagaimana cara menjaga data saat dikirimkan dari suatu tempat ke tempat lainnya. Dalam penelitian ini penulis mengembangkan beberapa algoritma, hasilnya nanti dapat dimanfaatkan untuk melindungi data informasi penting dengan cara penggabungan 3 algoritma kriptografi yaitu XOR, Rot47 dan Base64, sehingga dapat mempersulit pihak-pihak yang tidak bertanggung jawab mengakses data rahasia. Penulis menerapkan algoritma penggabungan pada sistem berbasis web dan dapat di implementasikan pada perangkat yang lain, dalam penelitian ini yaitu android.

**Kata Kunci : Algoritma Penggabungan, Kriptografi, XOR, ROT47, Base64, Data.**

# BAB I

# PENDAHULUAN

## 1.1. Latar Belakang

Kerahasiaan dan keamanan data merupakan suatu hal yang sangat penting bagi suatu perusahaan atau instansi maupun individu, apalagi informasi dari data tersebut bisa menimbulkan resiko yang dapat merugikan seperti kebocoran informasi sensitif bagi perusahaan atau instansi maupun individu. Hal ini terjadi karena pesatnya pengetahuan ilmu teknologi sehingga munculnya berbagai teknik-teknik yang baru yang di salah gunakan seperti mengancam keamanan data untuk kepentingan sepihak.

Dari permasalahan yang timbul untuk mengatasi kebocoran data sehingga tidak diketahui oleh pihak-pihak yang tidak berkepentingan ada beberapa cara menyiasati untuk mengamankan informasi yang akan dilindungi, salah satu dengan teknik disebut kriptografi .

Kriptografi yaitu ilmu yang mempelajari bagaimana cara menjaga data atau pesan informasi saat dikirimkan dari suatu tempat ke tempat lainnya. Kriptografi yang kita kenal merupakan salah satu dari berbagai macam teknik mengamankan data dan memiliki berbagai metode algoritma yang bisa mempersulit pihak-pihak yang tidak bertanggung jawab yang ingin mengakses data yang bukan haknya.

Dalam Kriptografi sudah banyak penerapan algoritma-algoritma untuk kebutuhan keamanan data sampai saat ini, seiring kemajuan teknologi semakin banyak pula pihak-pihak yang tidak diinginkan mencoba mencari kelemahan pada algoritma yang ada sekarang, salah satunya tools yang beredar di internet untuk mendeksripsikan atau menerjemahkan hasil enkripsi untuk kepentingan pribadi dan juga sebagai media pembelajaran.

Penggabungan algoritma ini dapat mengenkripsi data dan dapat didekripsikan kembali dengan menggunakan tiga algoritma ini, sehingga dapat mempersulit pihak-pihak yang tidak bertanggung jawab mengakses data rahasia milik orang lain. Maka dari Pernyataan tersebut penulis mengambil topik keamanan data untuk bahan penelitian yang berjudul **"Penggabungan Algoritma Xor,Rot47 dan Base64 Untuk Peningkatan Keamanan Data"**

## 1.2. Perumusan Masalah

Berdasarkan latar belakang, maka penulis merumuskan permasalahan dalam penelitian ini yaitu "Bagaimana cara merancang penggabungan beberapa algoritma diatas menjadi satu hasil baru"

## 1.3. Batasan Masalah

Agar pembahasan lebih terarah dan sesuai dengan latar belakang pernyataan diatas, maka penulis membatasi permasalahan sebagai berikut :

1. Penelitiaan ini menggunakan tiga algoritma yaitu Xor,Rot47 dan Base64 untuk bahan penggabungan sehingga dapat menghasilkan teknik keamanan data (kriptografi) yang baru.
2. Pengujian algoritma yang telah digabungkan diterapkan pada informasi berupa tulisan (teks) dan gambar.
3. Algoritma yang telah digabungkan, diterapkan pada aplikasi sederhana untuk dilakukannya analisis terkait algoritma yang dibuat.
4. Pengujian tingkat kemananan algoritma tersebut akan dilakukan dengan teknik brute force.

## 1.4. Tujuan Penelitian

Penelitian ini bertujuan untuk Memberikan hasil algorima kriptografi yang baru dari tiga penggabungan algoritma yang telah diuraikan agar mengatasi kebocoran data yang format tulisan (teks) dan gambar.

## 1.5. Manfaat Penelitian

### 1.5.1. Manfaat Bagi Penulis

Manfaat penelitian ini untuk penulis adalah :

1. Dapat menambah wawasan, pengetahuan dan pemahaman bagi penulis terutama pada teknik kriptografi.
2. Menerapkan ilmu pengetahuan yang telah didapatkan selama menjalani proses perkuliahan

### 1.5.2. Manfaat Bagi Akademik

Manfaat penelitian ini bagi Akademik adalah dapat menjadi salah satu referensi atau acuan bagi akademik dalam membantu penelitian yang

akan datang dan dapat menambah pengetahuan bagi pihak yang bersangkutan mengenai keamanan data dan teknik kriptografi.

### 1.5.3. Manfaat Bagi Umum

Manfaat penelitian ini bagi umum adalah sebagai ilmu pengetahuan dan mengetahui bagaimana proses enkripsi dan dekripsi bekerja dalam suatu sistem kemanan data.

## 1.6. Sistematika Penulisan

Sistematika Penulisan Skripsi ini bertujuan memberikan penjelasan tentang garis-garis besar isi penelitian, agar lebih terlihat berhubungan, yang disusun dalam kerangka bab dan sub-bab. Adapun sistematika penulisan dijabarkan di bawah ini sebagai berikut :

**BAB I PENDAHULUAN**

Pada bab ini membahas tentang latar belakang, perumusan masalah, batasanmasalah, tujuan dan manfaat penelitian, dan sistematika penulisan.

**BAB II GAMBARAN UMUM PENELITIAN**

Bab ini membahas mengenai fenomena tentang penelitian yang sedang dilakukan.

**BAB III TINJAUAN PUSTAKA**

Bab ini membahas tentang tinjauan pustaka yang digunakan dalam penelitian antara lain landasan teori atau teori pendukung, hasil penelitian terdahulu dan kerangka penelitian.

**BAB IV METODE PENELITIAN**

Bab ini membahas tentang jenis data yang akan diamankan dan teknik perhitungan data untuk penggabungan algoritma tersebut.

**BAB V HASIL DAN PEMBAHASAN**

Bab ini berisi hasil perhitungan manual yang di dapatkan dari perhitungan pada algoritma. Pada tahapan pembahasan, peneliti membahas proses bagaimana perhitungan pada algoritma dilakukan.

**BAB VI PENUTUP**

Bab ini berisi hasil yang telah didapatkan dari kesimpulan dalam penelitian dan saran - saran untuk pengembangan selanjutnya.

# BAB II

# GAMBARAN UMUM PENELITIAN

## 2.1. Fenomena Penelitian

Seiring dengan kemajuannya teknologi juga membuat perkembangan sistem keamanannya semakin lebih maju lagi, dengan adanya sistem keamanan yang baik membuat sistem atau aplikasi menjadi lebih baik dan bermanfaat. Keamanan data adalah cara atau teknik memastikan data yang disimpan menjadi lebih aman terhindar dari kendali orang yang tidak punya hak untuk mengaksesnya.

Namun masalah keamanan data sering kali menjadi permasalahan dan kurang perhatian bagi pemilik dan pengguna sistem atau aplikasi, salah satu contohnya pada tahun 2014 terjadi pencurian data dengan skala besar menyerang salah satu perusahaan yang terkenal yaitu *Sony Picture*, akibatnya data-data yang penting dan bersifat rahasia dibuka ke publik oleh peretas sehingga saham *Sony Pictures* menjadi menurun drastis.

Beberapa kejadian serupa juga terjadi pada pengguna *marketplace* di Indonesia sehingga membuat kerugian pengguna dan perusahaan *marketplace* menjadi merugi karena data yang bersifat pribadi di pergunakan tanpa persetujuan pemilik data yang sebenarnya oleh pihak-pihang yang tidak bertanggung jawab.

Dari dua kejadian tersebut memberikan kita gambaran bahwa semakin canggihnya suatu sistem dan terdigitalisasi kehidupan kita, maka juga di butuhkan peningkatan pengamanan data.

Dengan adanya peningkatan keamanan data akan mengatasi dan membatasi pergerakan pihak yang tidak berhak untuk bebas melakukan pencurian data, salah satu solusi dengan menerapkan metode enkripsi pada suatu data individu maupun data pada sistem lainnya.

Enkripsi perlahan akan menjadi kebutuhan dan sudah menjadi kewajiban perusahaan atau individu yang terkena imbas *General Data Protection Regulation* (GDPR) besutan *European Union* (EU), mereka yang memiliki bisnis dengan negara Eropa wajib mengunakan Enkripsi untuk keamanan data. Kekuatan solusi enkripsi dalam menjaga kerahasiaan data menjadi sorotan yang sangat penting dan di utamakan.

# BAB III

# TINJAUAN PUSTAKA

## 3.1. Landasan Teori

### 3.1.1. Keamanan Data

Keamanan data adalah cara atau teknik memastikan data yang disimpan menjadi lebih aman terhindar dari kendali orang yang tidak punya hak untuk mengaksesnya.

Keamanan data memiliki salah satu teknik yang sering digunakan untuk mengamankan data yaitu kriptografi. Kriptografi adalah ilmu yang mempelajari bagaimana cara menjaga data dan memiliki banyak metode algoritma yang digunakan untuk mengamankan data dari pihak yang tidak memiliki hak untuk mengakses data tersebut.

### 3.1.2. Algoritma Kriptografi

Menurut Harun Mukhtar (2018), Kriptografi adalah ilmu dan seni untuk menjaga keamanan pesan yang dikirim dari satu tempat ke tempat lain.

Menurut Rifki S (2012),Kriptografi adalah ilmu yang bersandarkan pada teknik matematika untuk berurusan dengan keamanan informasi seperti kerahasiaan, keutuhan data dan otentikasi entitas. Pada awalnya kriftografi dijelaskan sebagai ilmu yang mempelajari bagaimana menyembunyikan pesan.

Algoritma kriptografi modern umumnya beroperasi dalam mode bit ketimbang mode karakter (seperti yang dilakukan pada cipher sibstitusi atau cipher transposisi dari algoritma kriptografi klasik). Operasi dalam mode bit berarti semua data dan informasi (baik kunci, plaintext maupun ciphertext) dinyatakan dalam rangkaian (string) bit binner, 0 dan 1. Algoritma enkripsi dan deskripsi memproses semua data dan informasi dalam bentuk mode bit. Rangkaian bit yang menyatakan plaintext dienkripsi menjadi ciphertext dalam bentuk rangkaian bit, demikian sebaliknya.

Algoritma kriptografi modern terdiri dari 3 bagian, yaitu :

a. Algoritma Simetris

Algoritma simetris adalah algoritma yang menggunakan kunci yang sama untuk enkripsi dan deskripsinya. Algoritma kriptografi simetris sering disebut algoritma kunci rahasia, algoritma kunci tunggal atau algoritma satu kunci dan megharuskan pengirim dan penerima menyetujui suatu kunci tersebut. Kelebihan dari kriptografi simetris waktu proses untuk enkripsi dan deskripsi relatif cepat. Hal ini disebabkan efesiensi yang terjadi pada pembangkit kunci. Maka proses *relative* cepat maka algoritma ini tepat untuk digunakan pada sistem komunikasi digital secara *real time* seperti GSM.

Aplikasi dari algoritma simetris digunakan oleh beberapa algoritma dibawah ini :

1. *Data Encryption Standar (DES)*
2. *Advance Encryption Standar (AES)*
3. *International Data Encryption Algoritma*
4. *A5*
5. *RC4*

b. Algoritma Asimetris

Algoritma asimetris adalah pasangan kunci kriptografi yang salah satunya digunakan untuk proses enkripsi dan satu lagi deskripsi. Semua orang yang mendapatkan kunci publik dapat menggunakannya untuk mengenkripsi suatu pesan, sedangkan hanya satu orang saja yang memiliki rahasia itu, dalam hal ini kunci rahasia, untuk melakukan pembongkaran terhadap kode yang dikirim untuknya. Contoh algoritma terkenal yang menggunakan kunci asimetris adalah RSA (merupakan singkatan dari nama penemunya, yaitu Rivest, Shamir dan Adleman).

c. Algoritma Hibrida (*Hybrid*)

Algoritma hibrida adalah algoritma yang memanfaatkan dua tingkatan kunci, yaitu kunci rahasia (simetri) yang disebut juga *session key* (kunci sesi) untuk enkripsi data dan pasangan kunci rahasia adalah kunci publik untuk pemberian tanda tangan digital serta melindungi kunci simetris.

Algoritma Kriptografi yang beroperasi dalam mode bit dapat dikelompokan menjadi dua kategori, yaitu :

1. Aliran Cipher

   Algoritma kriptografi beroperasi pada *plainteks* atau *cipherteks* dalam bentuk bit tunggal, yang dalam hal ini rangkaian rangkaian bit dienkripsikan atau dideskripsikan bit per bit. Stream *Chiper* atau *Stream Encryption* merupakan suatu teknik enkripsi data dengan cara melakukan transformasi dari tiap bit secara terpisah berdasarkan posisi tiap bit dalam aliran data yang biasanya dikendalikan menggunakan operasi XOR. Enkripsi aliran data merupakan hasil dari operasi XOR setiap bit plainteks dengan setiap bit kuncinya. Pada stream chiper bila terjadi kesalahan selama transisi maka kesalahan pada teks enkripsi penerima akan terjadi tepat ditempat kesalahan tersebut terjadi. Dalam praktek pertimbangan kesalahan yang mungkin terjadi sangatlah penting untuk penentuan teknik enkripsi yang akan digunakan.

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.

Misalnya panjang blok adalah 64 bit, maka itu algoritma enkripsi memperlakukan 8 karakter setiap kali penyandian (1 karakter = 8 bit dalam pengkodean ASCII)

Adapun algoritma yang dipakai peneliti adalah :

**1. Algoritma XOR**

Menurut Safaat (2014), Algoritma *XOR* adalah algoritma sederhana yang menggunakan prinsip operator logika *XOR* . Proses dalam melakukan enkripsinya adalah dengan meng-*XOR*-kan *plaintext* dengan kunci sehingga didapatkan *ciphertext*-nya. Sebaliknya untuk proses dekripsi adalah dengan meng-*XOR*-kan *ciphertext* dengan kunci sehingga didapatkan *plaintext*-nya kembali. Untuk kriptografi klasik, penulis memilih algoritma ini dikarenakan mudah diimplementasikan dan operasi *XOR* tidak sulit secara komputasional. Karenanya algoritma *XOR* masih sering digunakan untuk mengamankan informasi atau pesan dan kemudian dilengkapi dengan suatu mekanisme keamanan tambahan yang dalam hal ini peneliti menambahkan algoritma AES.

Secara singkat, operasi XOR akan mengembalikan nilai 1 jika jumlah operand bernilai satu ganjil, jika tidak maka akan mengembalikan hasil 0. Berikut ini contohnya:

1 XOR 1 = 0

1 XOR 0 = 1

0 XOR 1 = 1

0 XOR 0 = 0

Dalam kriptografi, pembuatan chiper (teks hasil enkripsi) melalui operasi XOR merupakan suatu algoritma enskripsi yang relatif sederhana. Teknik ini beroperasi sesuai dengan prinsip:

A XOR 0 = A,

A XOR A = 0,

(B XOR A) XOR A = B XOR 0 = B,

Dengan logika ini, suatu string teks dapat diekripsi dengan menerapkan operasi XOR berbasis bit (binary digit) terhadap setiap karakter menggunakan key tertentu. Bagaimana mendekripsi outputnya untuk mendapatkan plaintext kembali? Dengan menerapkan operasi XOR terhadap chiper.

Sebagai contoh, string “Wiki” jika ditulis dalam format ASCII 8 bit menjadi 01010111 01101001 01101011 01101001 dapat dienkripsi dengan suatu key misalnya 11110011 sebagai berikut:

01010111 01101001 01101011 01101001

11110011 11110011 11110011 11110011

——————————————— (XOR)

10100100 10011010 10011000 10011010 (Hasil)

Dan sebaliknya, untuk dekripsi adalah:

10100100 10011010 10011000 10011010

11110011 11110011 11110011 11110011

——————————————— (XOR)

01010111 01101001 01101011 01101001 (Hasil)

**2. Algoritma ROT47**

Menurut Aulia,Rachmat, Ahmad Z dan Dian AP (2018) Algoritma ROT47 adalah turunan dari ROT13. ROT47 memperkenalkan huruf dan simbol campuran, oleh karena itu, teks yang dikodekan terlihat lebih jelas bahwa teks telah dienkripsikan. ROT47 juga dapat dengan mudah diimplementasikan oleh bahasa pemrograman modern dengan banyak cara. Cara kerja Algortma ROT47 yaitu bekerja berdasarkan nilai ASCII dengan rentang nilai 33 - 126 dan melihat nilai dimiliki oleh setiap karakter, contoh huruf “a” bernilai “97” maka dengan melakukan pergeseran sebanyak 47 langkah akan menjadi angka “2” dengan nilai “50”.

### 3. Algoritma Base64

Menurut Ariyus (2008), teknik enkripsi *base*64 sebetulnya sederhana, jika terdapat sebuah (*string*) *bytes* yang akan disandikan kealgoritma *base*64 maka tahapannya yaitu:

1. Pecah *string bytes* tersebut ke per-3 *bytes*.
2. Gabungkan 3 *bytes* menjadi 24 *bit*. dengan catatan 1 *bytes* = 8 *bit*, sehingga 3x8 = 24 *bit*.
3. Lalu 24 *bit* yang disimpan di-*buffer* (disatukan) dipecah-pecah menjadi 6 *bit*,maka akan menghasilkan 4 pecahan.
4. Masing masing pecahan diubah ke dalam nilai desimal, dimana maksimal nilai6 *bit* dalah 63.
5. Terakhir, jadikan nilai-nilai desimal tersebut menjadi *index* untuk memilih maksimal *index* ke 64 atau karakter ke 63 dari penyusun *base*64. Dan seterusnya hingga akhir *string bytes* yang akan mengalami konversi. Apabila dalam proses *encoding* terdapat sisa pembagi, maka tambahkan karakter *pad* (=) sebagai penggenap sisa tersebut. Oleh karena itu, terkadang pada *base*64 akan muncul satu atau dua karakter (=).

   Algoritma Base64 menggunakan kode ASCII dan kode index base64 dalam melakukan proses enkripsi ataupun dekripsinya. Dalam melakukan enkripsi pada URL website, kode index base64 perlu dimodifikasi. Simbol

"+" dimodifikasi menjadi "-" dan simbol simbol "/" menjadi"_" [17].

Adapun tahapan - tahapan dekripsi menggunakan Algoritma Base64 adalah sebagai berikut :

**Gambar 3.1 Tabel Base64**

| Binary | ASCII | Binary | ASCII | Binary | ASCII | Binary | ASCII |
|---|---|---|---|---|---|---|---|
| 000000 | A | 010000 | Q | 100000 | g | 110000 | w |
| 000001 | B | 010001 | R | 100001 | h | 110001 | x |
| 000010 | C | 010010 | S | 100010 | i | 110010 | y |
| 000011 | D | 010011 | T | 100011 | j | 110011 | z |
| 000100 | E | 010100 | U | 100100 | k | 110100 | 0 |
| 000101 | F | 010101 | V | 100101 | l | 110101 | 1 |
| 000110 | G | 010110 | W | 100110 | m | 110110 | 2 |
| 000111 | H | 010111 | X | 100111 | n | 110111 | 3 |
| 001000 | I | 011000 | Y | 101000 | o | 111000 | 4 |
| 001001 | J | 011001 | Z | 101001 | p | 111001 | 5 |
| 001010 | K | 011010 | a | 101010 | q | 111010 | 6 |
| 001011 | L | 011011 | b | 101011 | r | 111011 | 7 |
| 001100 | M | 011100 | c | 101100 | s | 111100 | 8 |
| 001101 | N | 011101 | d | 101101 | t | 111101 | 9 |
| 001110 | O | 011110 | e | 101110 | u | 111110 | + |
| 001111 | P | 011111 | f | 101111 | v | 111111 | / |

1. Mengkonversi karakter Base64 ke biner dengan menggunakan 6 bit.
2. Konversi 24 bit dari empat kelompok 6 bit ke tiga kelompok 8 bit.
3. Konversi masing-masing tiga kelompok 8 bit ke desimal.
4. Gunakan masing-masing tiga desimal untuk mencari karakter ASCII untuk nilai yang ada.

### 4. ASCII Kode

Kode ASCII memiliki kepanjangan dari *American Standard Code for Information Interchange*. Kode ASCII adalah suatu standar internasional dalam kode huruf dan simbol seperti Hex dan Unicode tetapi ASCII lebih bersifat universal. Kode ASCII terdiri dari deretan angka dan huruf.

Fungsi Kode ASCII adalah sebagai karakter kode dari user atau admin dalam menjalankan perintah dan akan ditranslasikan agar komputer memahami kode dari user karena komputer hanya membaca angka *binner*. Berikut adalah tabel ASCII

**Tabel 3.1 Tabel ASCII**

| Decimal | Octal | Hex | Binary | Value | Description |
|---|---|---|---|---|---|
| 000 | 000 | 00 | 0000 0000 | NUL | "null" character |
| 001 | 001 | 01 | 0000 0001 | SOH | start of header |
| 002 | 002 | 02 | 0000 0010 | STX | start of text |
| 003 | 003 | 03 | 0000 0011 | ETX | end of text |
| 004 | 004 | 04 | 0000 0100 | EOT | end of transmission |
| 005 | 005 | 05 | 0000 0101 | ENQ | Enquiry |
| 006 | 006 | 06 | 0000 0110 | ACK | Acknowledgment |
| 007 | 007 | 07 | 0000 0111 | BEL | Bell |
| 008 | 010 | 08 | 0000 1000 | BS | Backspace |
| 009 | 011 | 09 | 0000 1001 | HT | horizontal tab |
| 010 | 012 | 0A | 0000 1010 | LF | line feed |
| 011 | 013 | 0B | 0000 1011 | VT | vertical tab |
| 012 | 014 | 0C | 0000 1100 | FF | form feed |
| 013 | 015 | 0D | 0000 1101 | CR | carriage return |
| 014 | 016 | 0E | 0000 1110 | SO | shift out |
| 015 | 017 | 0F | 0000 1111 | SI | shift in |
| 016 | 020 | 10 | 0001 0000 | DLE | data link escape |
| 017 | 021 | 11 | 0001 0001 | DC1 | device control 1 (XON) |
| 018 | 022 | 12 | 0001 0010 | DC2 | device control 2 |
| 019 | 023 | 13 | 0001 0011 | DC3 | device control 3 (XOFF) |

| Decimal | Octal | Hex | Binary | Value | Description |
|---|---|---|---|---|---|
| 020 | 024 | 14 | 0001 0100 | DC4 | device control 4 |
| 021 | 025 | 15 | 0001 0101 | NAK | negative acknowledgement |
| 022 | 026 | 16 | 0001 0110 | SYN | synchronous idle |
| 023 | 027 | 17 | 0001 0111 | ETB | end of transmission block |
| 024 | 030 | 18 | 0001 1000 | CAN | Cancel |
| 025 | 031 | 19 | 0001 1001 | EM | end of medium |
| 026 | 032 | 1A | 0001 1010 | SUB | Substitute |
| 027 | 033 | 1B | 0001 1011 | ESC | Escape |
| 028 | 034 | 1C | 0001 1100 | FS | file separator |
| 029 | 035 | 1D | 0001 1101 | GS | group separator |
| 030 | 036 | 1E | 0001 1110 | RS | request to send/record separator |
| 031 | 037 | 1F | 0001 1111 | US | unit separator |
| 032 | 040 | 20 | 0010 0000 | SP | Space |
| 033 | 041 | 21 | 0010 0001 | ! | exclamation mark |
| 034 | 042 | 22 | 0010 0010 | " | double quote |
| 035 | 043 | 23 | 0010 0011 | # | number sign |
| 036 | 044 | 24 | 0010 0100 | $ | dollar sign |
| 037 | 045 | 25 | 0010 0101 | % | Percent |
| 038 | 046 | 26 | 0010 0110 | & | Ampersand |
| 039 | 047 | 27 | 0010 0111 | ' | single quote |
| 040 | 050 | 28 | 0010 1000 | ( | left/opening parenthesis |
| 041 | 051 | 29 | 0010 1001 | ) | right/closing parenthesis |
| 042 | 052 | 2A | 0010 1010 | * | Asterisk |
| 043 | 053 | 2B | 0010 1011 | + | Plus |
| 044 | 054 | 2C | 0010 1100 | , | Comma |
| 045 | 055 | 2D | 0010 1101 | - | minus or dash |
| 046 | 056 | 2E | 0010 1110 | . | Dot |
| 047 | 057 | 2F | 0010 1111 | / | forward slash |
| 048 | 060 | 30 | 0011 0000 | 0 | |
| 049 | 061 | 31 | 0011 0001 | 1 | |
| 050 | 062 | 32 | 0011 0010 | 2 | |
| 051 | 063 | 33 | 0011 0011 | 3 | |
| 052 | 064 | 34 | 0011 0100 | 4 | |
| 053 | 065 | 35 | 0011 0101 | 5 | |
| 054 | 066 | 36 | 0011 0110 | 6 | |
| 055 | 067 | 37 | 0011 0111 | 7 | |
| 056 | 070 | 38 | 0011 1000 | 8 | |
| 057 | 071 | 39 | 0011 1001 | 9 | |
| 058 | 072 | 3A | 0011 1010 | : | Colon |

| Decimal | Octal | Hex | Binary | Value | Description |
|---|---|---|---|---|---|
| 059 | 073 | 3B | 0011 1011 | ; | semi-colon |
| 060 | 074 | 3C | 0011 1100 | < | less than |
| 061 | 075 | 3D | 0011 1101 | = | equal sign |
| 062 | 076 | 3E | 0011 1110 | > | greater than |
| 063 | 077 | 3F | 0011 1111 | ? | question mark |
| 064 | 100 | 40 | 0100 0000 | @ | "at" symbol |
| 065 | 101 | 41 | 0100 0001 | A | |
| 066 | 102 | 42 | 0100 0010 | B | |
| 067 | 103 | 43 | 0100 0011 | C | |
| 068 | 104 | 44 | 0100 0100 | D | |
| 069 | 105 | 45 | 0100 0101 | E | |
| 070 | 106 | 46 | 0100 0110 | F | |
| 071 | 107 | 47 | 0100 0111 | G | |
| 072 | 110 | 48 | 0100 1000 | H | |
| 073 | 111 | 49 | 0100 1001 | I | |
| 074 | 112 | 4A | 0100 1010 | J | |
| 075 | 113 | 4B | 0100 1011 | K | |
| 076 | 114 | 4C | 0100 1100 | L | |
| 077 | 115 | 4D | 0100 1101 | M | |
| 078 | 116 | 4E | 0100 1110 | N | |
| 079 | 117 | 4F | 0100 1111 | O | |
| 080 | 120 | 50 | 0101 0000 | P | |
| 081 | 121 | 51 | 0101 0001 | Q | |
| 082 | 122 | 52 | 0101 0010 | R | |
| 083 | 123 | 53 | 0101 0011 | S | |
| 084 | 124 | 54 | 0101 0100 | T | |
| 085 | 125 | 55 | 0101 0101 | U | |
| 086 | 126 | 56 | 0101 0110 | V | |
| 087 | 127 | 57 | 0101 0111 | W | |
| 088 | 130 | 58 | 0101 1000 | X | |
| 089 | 131 | 59 | 0101 1001 | Y | |
| 090 | 132 | 5A | 0101 1010 | Z | |
| 091 | 133 | 5B | 0101 1011 | [ | left/opening bracket |
| 092 | 134 | 5C | 0101 1100 | \ | back slash |
| 093 | 135 | 5D | 0101 1101 | ] | right/closing bracket |
| 094 | 136 | 5E | 0101 1110 | ^ | caret/circumflex |
| 095 | 137 | 5F | 0101 1111 | _ | Underscore |
| 096 | 140 | 60 | 0110 0000 | ` | |
| 097 | 141 | 61 | 0110 0001 | a | |
| 098 | 142 | 62 | 0110 0010 | b | |
| 099 | 143 | 63 | 0110 0011 | c | |
| 100 | 144 | 64 | 0110 0100 | d | |
| 101 | 145 | 65 | 0110 0101 | e | |

| Decimal | Octal | Hex | Binary | Value | Description |
|---|---|---|---|---|---|
| 102 | 146 | 66 | 0110 0110 | f | |
| 103 | 147 | 67 | 0110 0111 | g | |
| 104 | 150 | 68 | 0110 1000 | h | |
| 105 | 151 | 69 | 0110 1001 | i | |
| 106 | 152 | 6A | 0110 1010 | j | |
| 107 | 153 | 6B | 0110 1011 | k | |
| 108 | 154 | 6C | 0110 1100 | l | |
| 109 | 155 | 6D | 0110 1101 | m | |
| 110 | 156 | 6E | 0110 1110 | n | |
| 111 | 157 | 6F | 0110 1111 | o | |
| 112 | 160 | 70 | 0111 0000 | p | |
| 113 | 161 | 71 | 0111 0001 | q | |
| 114 | 162 | 72 | 0111 0010 | r | |
| 115 | 163 | 73 | 0111 0011 | s | |
| 116 | 164 | 74 | 0111 0100 | t | |
| 117 | 165 | 75 | 0111 0101 | u | |
| 118 | 166 | 76 | 0111 0110 | v | |
| 119 | 167 | 77 | 0111 0111 | w | |
| 120 | 170 | 78 | 0111 1000 | X | |
| 121 | 171 | 79 | 0111 1001 | Y | |
| 122 | 172 | 7A | 0111 1010 | Z | |
| 123 | 173 | 7B | 0111 1011 | { | left/opening brace |
| 124 | 174 | 7C | 0111 1100 | \| | vertical bar |
| 125 | 175 | 7D | 0111 1101 | } | right/closing brace |
| 126 | 176 | 7E | 0111 1110 | ~ | Tilde |
| 127 | 177 | 7F | 0111 1111 | DEL | Delete |

### 3.1.3. *RestFul Application Programming Interface (API)*

Menurut Richardson dalam putra (2017:10) Salah satu kriteria desain layanan *web* yang paling sering digunakan adalah *RESTful*, Layanan web *RESTful* bekerja dengan cara *resource-oriented*. Pada layanan web *RESTful* klien mengakses *services* yang ditawarkan oleh *web server*, yaitu dengan cara mengakses alamat dari *resource* menggunakan *method* pada *HTTP*.

Beberapa *method HTTP* yang sering digunakan pada layanan web *RESTful* adalah:

- *GET*: mengambil sumber daya dari *web server* melalui pengaturan nilai parameter dari permintaan klien.
- *POST*: menyimpan sumber daya ke *web server* melalui penyisipan pesan pada badan pesan permintaan klien.
- *PUT*: bekerja layaknya *method POST*, namun digunakan untuk memperbarui sebagian sumber daya yang telah tersimpan pada *web server*.
- *DELETE*: menghapus sumber daya pada *web server*.
- *HEAD*: memperoleh informasi mengenai URL di *web server*.
- *OPTION*: melihat daftar *method HTTP* yang dapat diakses oleh klien pada *web server*.

## 3.2. Penelitian Terdahulu

Hasil penelitian terdahulu digunakan sebagai pedoman dasar, acuan, pertimbangan, maupun perbandingan bagi penelitian terbaru yang sejenis,adapun penelitian terdahulu yang penulis gunakan seperti pada tabel 3.2.

**Tabel 3.2 Penelitian Terdahulu**

| No | Judul | Penulis dan Tahun | Hasil |
|---|---|---|---|
| 1. | Penerapan Algoritma Gabungan RC4 dan Base64 Pada Sistem Keamanan E-Commerce<br><br>Jurnal Nasional Aplikasi Teknologi Informasi, ISSN 1907-5022 | Febrian Wahyu,Adriana.P Rahangiar,Febry de Fretes<br><br>Tahun 2012 | Sistemkeamanan menggunakan Algoritma Kriptografi *RC4* dan *Base64* dapat menjamin keamanan data transakis pembayaran online yang dilakukan oleh pelanggan karena password pelanggan |

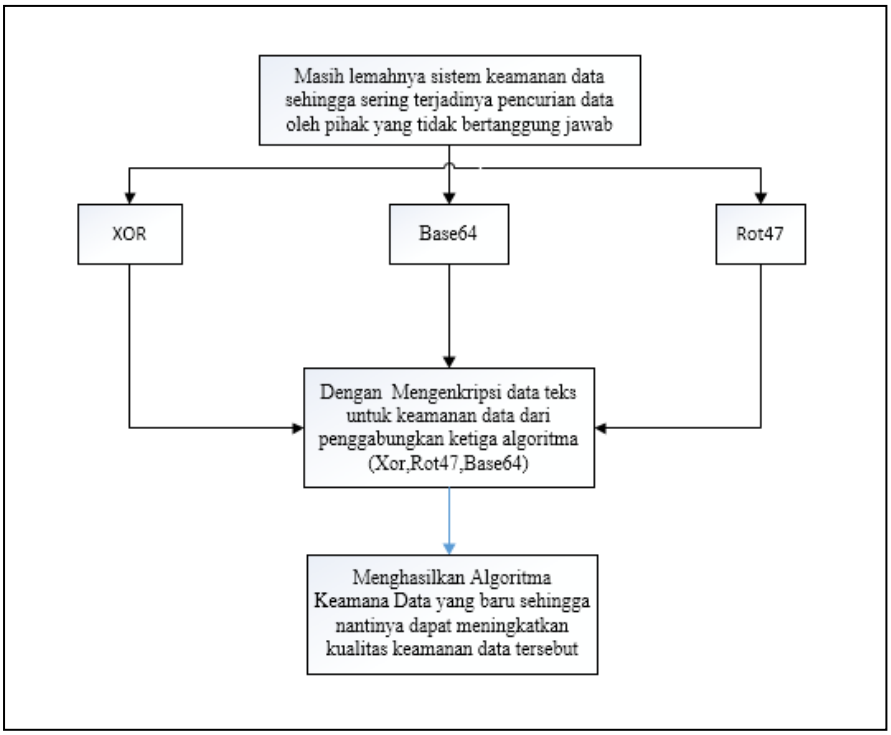
**Tabel 3.2 Penelitian Terdahulu**

| No | Judul | Penulis dan Tahun | Hasil |
|---|---|---|---|
| | | | di Bank telah disamarkan dengan proses enkripsi dan sangat sulit dipecahkan apabila kunci dan perhitungan algoritma berbeda. selain itu, disisi penyedia jasa e-commerce dapat menjamin kenyamanan bagi para pelanggan yang menggunakan jasa layanan e-commerce. |
| 2. | Penerapan Kombinasi Algoritma Base64 dan Rot47 Untuk Enkripsi Database Pasien Rumah Sakit Jiwa Prof.DR.Muhammad Ildrem<br><br>Jurnal Nasional Informatika dan Teknologi Jaringan<br><br>E-ISSN 2540-7600<br>P-ISSN 2540-7597 | Rachmat Aulia,Ahmad Zakir, Dian Agung Purwanto<br><br>Tahun 2018 | Dari hasil penerapan Kombinasi algoritma ROT47 dan Base64 dapat menjadi sebuah algoritma dengan keamanan yang memadai dan dapat digunakan untuk mengamankan database pasien pada rumah sakit jiwa Prof. Dr. M. Ildrem dan Kombinasi algoritma ROT47 dan Base64 dapat di imlementasikan ke dalam sistem enkripsi database pasien dengan baik. |
| 3. | Aplikasi Kriptografi Keamanan Data Menggunakan Algoritma Base64<br><br>Volume 7,No.2,Desember 2018<br>ISSN 2528-0090 | Azlin dan Fitriah Musadat<br><br>Tahun 2018 | Dari proses pengujian pada aplikasi kriptografi algoritma Base64 dapat di simpulkan bahwa algoritma base64 mampu melakukan pengamanan data text dengan mengenkripsi file text menjadi sebuah karakter acak dan mengembalikan data text dengan cara mendeskripsi dari hasil enkripsi menjadi file text kebentuk semula. |

### 3.3. Kerangka Pemikiran

Adapun kerangka pemikiran yang akan dibahas, dapat dilihat pada gambar di bawah ini :

**Gambar 3.2 Kerangka Penelitian**




Maka dari gambar 3.2 dapat diamati bahwa penulis membuat algoritma penggabungan karena masih ada kerentanan kemanan data sehingga sering terjadinya pencurian data oleh pihak yang tidak bertanggung jawab.

Oleh karena itu penulis berinisiatif membuat algoritma baru dengan metode penggabungan algoritma XOR, Rot47 dan Base64 untuk meningkatkan kualitas keamanan data.

# BAB IV

# METODE PENELITIAN

## 4.1. Lokasi dan Waktu Penelitian

### 4.1.1. Lokasi

Tempat penelitian untuk skripsi ini dilakukan di Laboratorium Komputer di STMIK PalComTech yang berlokasi di jalan Basuki Rahmat No. 05 Palembang.

### 4.1.2. Jadwal Penelitian

Dalam penelitian ini, penulis menyusun segala kegiatan dalam sebuah jadwal penelitian yang berlangsung kurang lebih selama lima bulan terhitung mulai bulan Oktober 2019 sampai dengan bulan Februari 2020. Jadwal penelitian dapat dilihat pada tabel 4:

**Tabel 4.1 Jadwal Penelitian**

| No | Kegiatan | Tahun 2019 | | | | | | | | | | | | Tahun 2020 | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | Oktober | | | | November | | | | Desember | | | | Januari | | | | Februari | | | |
| | | 1 | 2 | 3 | 4 | 1 | 2 | 3 | 4 | 1 | 2 | 3 | 4 | 1 | 2 | 3 | 4 | 1 | 2 | 3 | 4 |
| 1 | Pengumpulan Data | | | | | | | | | | | | | | | | | | | | |
| | a. Mengumpulkan jurnal-jurnal terkait algoritma yang dipakai. | | | | | | | | | | | | | | | | | | | | |
| 2 | Merancang Algoritma | | | | | | | | | | | | | | | | | | | | |
| | a. Mencoba | | | | | | | | | | | | | | | | | | | | |

| No | Kegiatan | Tahun 2019 | | | | | | | | | | | | Tahun 2020 | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | Oktober | | | | November | | | | Desember | | | | Januari | | | | Februari | | | |
| | | 1 | 2 | 3 | 4 | 1 | 2 | 3 | 4 | 1 | 2 | 3 | 4 | 1 | 2 | 3 | 4 | 1 | 2 | 3 | 4 |
| | memodifikasi pola penggabungan algoritma dari jurnal serupa | | | | | | | | | | | | | | | | | | | | |
| | b. Membuat beberapa pola penggabungan algoritma | | | | | | | | | | | | | | | | | | | | |
| 3 | Implementasi | | | | | | | | | | | | | | | | | | | | |
| | a. Uji coba pola penggabungan algoritma dengan objek teks | | | | | | | | | | | | | | | | | | | | |
| 4 | Evaluasi | | | | | | | | | | | | | | | | | | | | |
| 5 | *Result* (Hasil) | | | | | | | | | | | | | | | | | | | | |

## 4.2. Jenis Data

Dalam penelitian ini penulis memperjelas apakah data dalam penelitian ini merupakan data primer atau data sekunder.

### 4.2.1. Data Primer

Menurut Riadi (2016:48), Data Primer adalah data informasi yang diperoleh tangan pertama yang dikumpulkan secara langsung dari sumbernya. Data primer adalah data yang paling asli dalam karakter dan tidak mengalami perlakuan statistik apapun.

Data Primer yang di dapatkan penulis berupa hasil penelitian yang dilakukan dari penggabungan algoritma sehingga menghasilkan algoritma baru dari teknik kriptograpi.

### 4.2.2. Data Sekunder

Menurut Riadi (2016:48), Data Sekunder adalah informasi tangan kedua yang sudah dikumpulkan oleh beberapa orang (organisasi) untuk tujuan tertentu dan tersedia untuk berbagi penelitian. Data sekunder tersebut tidak murni dalam karakter dan telah menjalani *treatment* setidaknya satu kali. Contoh data sekunder adalah data yang diperoleh dari Biro Pusat Statistik (BPS), buku, laporan, jurnal dan lain-lain.

Data sekunder penulis dapatkan dari penelitian terdahulu, buku-buku refrensi dan jurnal terkait penelitian serta memperkaya dan membantu tentang pemahaman dari penelitian yang akan penulis lakukan.

## 4.3. Teknik Pengumpulan Data

### 4.3.1. Studi Pustaka

Menurut Sunyoto (2016:21), studi kepustakaan (*library research*) adalah teknik pengumpulan data dengan mempelajari buku-buku yang ada hubungannya dengan obyek penelitian atau sumber-sumber lain yang mendukung penelitian.

Dalam penelitian ini penulis melakukan studi pustakaan dengan mengadakan penggalian data-data yang telah ada, baik itu dari buku serta jurnal sebagai referensi dan informasi untuk mendapatakan konsep serta pengetahuan yang sesuai dengan masalah yang akan diteliti.

## 4.4. Jenis Penelitian

Jenis penelitian yang dilakukan pada penelitian ini adalah penelitian eksperimen. Menurut Nazir (2014:51), Metode Eksperimental merupakan metode penelitian yang sering digunakan, lebih-lebih dalam penelitian eksakta. Eksperimen adalah observasi dibawah kondisi buatan (*artifical condition*) dimana kondisi tersebut dibuat dan diatur oleh si peneliti. Dengan demikian, penelitian eksperimental adalah penelitian yang dilakukan dengan mengadakan manipulasi terhadap objek penelitian serta adanya kontrol.

## 4.5. Alat Pengembangan Sistem

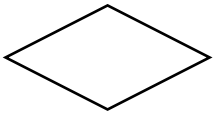
Dalam pengembangan sistem ini penulis menggunakan metode analisis tertruktur yaitu *flowchart* sebagai bentuk proses algoritma enkripsi dan dikripsi dan menggunakan UML yang merupakan singkatan *Unified Modelling Language* sebagai metode permodelan secara visual untuk sarana perancangan sistem berorientasi objek yaitu *Use Case, Class Diagram, dan Activity Diagram.*

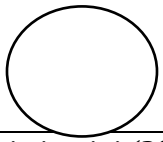
### *4.5.1. Flowchart*

Menurut Listiyarini (2019:166), *flowchart* adalah penggambaran langkah-langkah secara grafik dari urutan-urutan prosedur suatu program. *Flowchart* menolong analis maupun *programmer* untuk memecahkan masalah kedalam segmen-segmen yang lebih kecil dan menolong dalam menganalisis alternatif-alternatif lain dalam pengoperasian.

simbol-simbol yang biasanya dipakai adalah simbol-simbol *flowchart* yang dikeluarkan oleh ANSI dan ISO. Beberapa simbol standar diperlihatkan pada Tabel 4.2

**Tabel 4.2. Simbol-Simbol *Flowchart***

| No. | Simbol | Nama | Keterangan |
|---|---|---|---|
| 1. |  | Simbol proses | Menyatakan proses yang sedang dilakukan oleh prosesor/*computer*. |
| 2. |  | Proses manual | Menyatakan proses yang tidak dilakukan oleh *computer*. |
| 3. |  | Simbol *keying operation* | Menyatakan segala jenis operasi yang diproses dengan menggunakan suatu mesin yang mempunyai *keyboard.* |
| 4. |  | Simbol *decision* | *M*enunjukkan kondisi tertentu yang akan menghasilkan pilihan keluaran diputuskan. |

| No. | Simbol | Nama | Keterangan |
|---|---|---|---|
| 5. |  | Simbol *manual input* | Memasukkan data secara manual. |
| 6. |  | Simbol *Input/Output* | Menyatakan proses *input* atau *output* tanpa tergantung jenis peralatannya, |
| 7. |  | Simbol *predefined process* | Menyatakan penyediaan tempat penyimpanan suatu proses untuk memberi harga awal. |
| 8. |  | Simbol terminal | Menyatakan permulaan dan akhir sebuah program. |

Sumber: Listiyarini (2019:167)

### 4.5.2. UML (*Unified Modeling Language*)

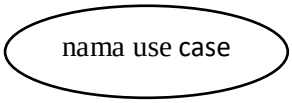
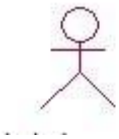
Menurut Pratama (2014:48), pada jenis pemrograman berbasis objek (*object oriented*), misalkan dengan bahasa Java, digunakan pemodelan UML. UML (*Unifed Modelling Language*) adalah standarisasi internasional untuk notasi dalam bentuk grafik yang menjelaskan tentang analisis dan desain perangkat lunak yang dikembangkan dengan pemrograman berorientasi objek.

#### 4.5.2.1. *Use Case* Diagram

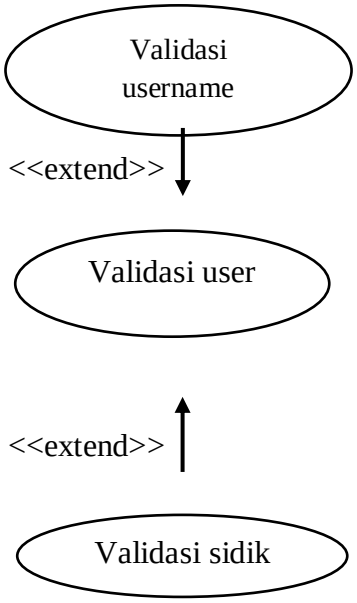
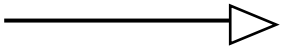
Menurut Rosa dan M. Shalahudin (2014), *use case* atau diagram *use case* merupakan pemodelan untuk

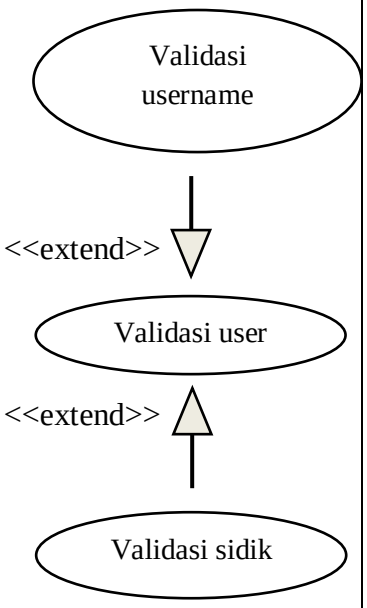
kelakuan (*behavior*) sistem informasi yang akan dibuat. *Use case* mendeskripsikan sebuah interaksi antara satu atau lebih aktor dengan sistem informasi yang akan dibuat. Secara kasar, *use case* digunakan untuk mengetahui fungsi apa saja yang ada di dalam sebuah sistem informasi dan siapa saja yang berhak menggunakan fungsi-fungsi itu. Berikut adalah simbol-simbol yang ada pada diagram *use case* dapat dilihat pada tabel 4.3

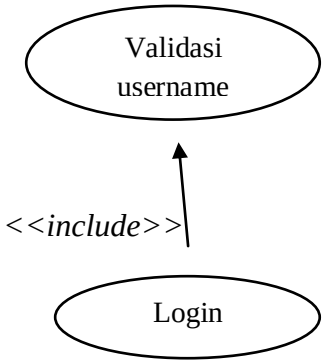
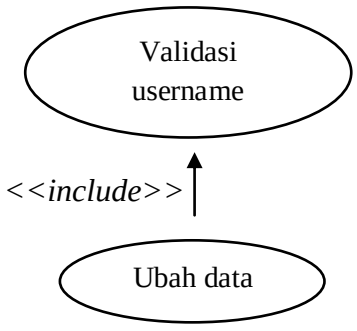
**Tabel 4.3 Simbol-simbol *Use Case* Diagram**

| No. | Simbol | Dekripsi |
|---|---|---|
| **1.** | *Use case*<br>  | Fungsionalitas yang disediakan sistem sebagai unit-unit yang saling bertukar pesan antar unit atau aktor; biasannya dinyatakan dengan menggunakan kata kerja di awal di awal frase nama *use case.* |
| **2.** | *Actor*<br>  | Orang, proses , atau sistem lain yang berinteraksi dengan sistem informasi yang akan dibuat di luar sistem informasi yang akan dibuat dibuat itu sendiri,jadi walau simbol |

| No. | Simbol | Dekripsi |
|---|---|---|
| | | dari aktor adalah gambar orang, tapi aktor belum tentu merupakan orang, biasanya dinyatakan menggunakan kata benda di awal frase nama aktor. |
| **3.** | *Association*  | Komunikasi antara aktor dan *use case* yang berpartisipasi pada *use case* atau *use case* memiliki interaksi dengan aktor |
| **4.** | *Extend* <<extend>>  | Relasi *use case* tambahankan dapat dimana *use case* yang ditambahkan dapat berdiri sendiri walau tanpa *use case* tambahan itu;mirip dengan prinsip *inheritance* pada pemprogram berorientasi objek, biasanya *use case* tambahan memiliki nama depan yang sama dengan *use case* yang ditambahkan , misal |

| No. | Simbol | Dekripsi |
|---|---|---|
| | |  <br>Arah panah mengarah pada Iuse case yang ditambahkan ; biasanya *use case* yang menjadi *extend*-nya merupakan jenis yang sama dengan *use case* yang menjadi induknya. |
| **5.** | *generalization*<br>  | Arah panah mengarahkan pada *use case* yang menjadi generelisasinya (umum). |

| No. | Simbol | Dekripsi |
|---|---|---|
| | |  |
| **5.** | <<include>><br><<uses>> | Dijalankan *use case* ini ada dua sudut pandang yang cukup besar mengenai *include* di *use case :*<br>• *Include* berarti *use case* yang ditambahkan akan selalu dipanggil saat *use case* tambahan dijalankan, misal pada kasus berikut : |

| No. | Simbol | Dekripsi |
|---|---|---|
| | |  <br>• *Include* berarti *use case* yang tambahan akan<br>• selalu melakukan pengecekan apakah *use case* yang ditambahkan telah dijalankan sebalum *use case* tambahan dijalankan ,misal pada kasus berikut:<br>  |

| No. | Simbol | Dekripsi |
|---|---|---|
| | | Kedua interpretasi di atas dapat dianut salah satu atau keduanya tergantung pada pertimbangan dan interpretasi yang dibutuhkan. |

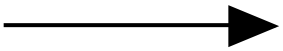
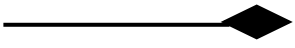
(Sumber : Shalahuddin, 2014:156-158)

#### 4.5.2.2. *Class* Diagram

Menurut Rosa dan M. Shalahuddin (2014), *Class* diagram menggambarkan struktur sistem dari segi pendefinisian kelas-kelas yang akan dibuat untuk membangun sistem. Kelas memiliki apa yang disebut atribut dan *method* atau operasi. Berikut merupakan simbol-simbol yang ada pada *class* diagram dapat dilihat pada tabel 4.4.

**Tabel 4.4 Simbol-simbol *class* diagram**

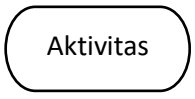
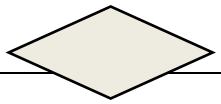
| No. | Simbol | Deskripsi |
|---|---|---|
| **1.** | Kelas<br>**nama_kelas**<br>+atribut<br>+operasi() | Kelas pada struktur sistem |
| **2.** | Antarmuka / *interface*<br>**nama_interface** | Sama dengan konsep *Interface* dalam pemrograman berorientasti objek |
| **3.** | Asosiasi / *association* | Relasi antarkelas |

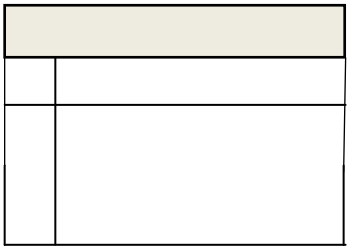
| No. | Simbol | Deskripsi |
|---|---|---|
| |  | dengan makna umum, asosiasi biasanya juga disertai dengan *multiplicity.* |
| **4.** | Asosiasi berarah / *directed association*  | Relasi antarkelas dengan makna kelas yang satu digunakan oleh kelas yang lain, asosiasi biasanya juga disertai dengan *multiplicity.* |
| **5.** | Generalisasi  | Relasi antarmuka dengan makna generalisasi spesialisasi (umum khusus). |
| **6.** | Kebergantungan / dependency  | Relasi antarmuka dengan mukna kebergantungan antarkelas |
| **7.** | Agregasi / aggregation  | Relasi antarkelas dengan makna semua bagian (*whole-part*) |

#### 4.5.2.3. *Activity* Diagram

Menurut Rosa dan M. Shalahuddin (2014), menggambarkan *workflow* atau aktifitas dari sebuah sistem atau proses bisnis atau menu yang ada pada perangkat lunak. Yang perlu di perhatikan disini adalah bahwa diagram aktifitas menggambarkan aktifitas sistem bukan apa yang dilakukan aktor, jadi aktifitas yang dapat dilakukan oleh sistem. Berikut adalah simbol-simbol yang ada pada pada *activity* diagram dapat dilihat pada tabel 4.5

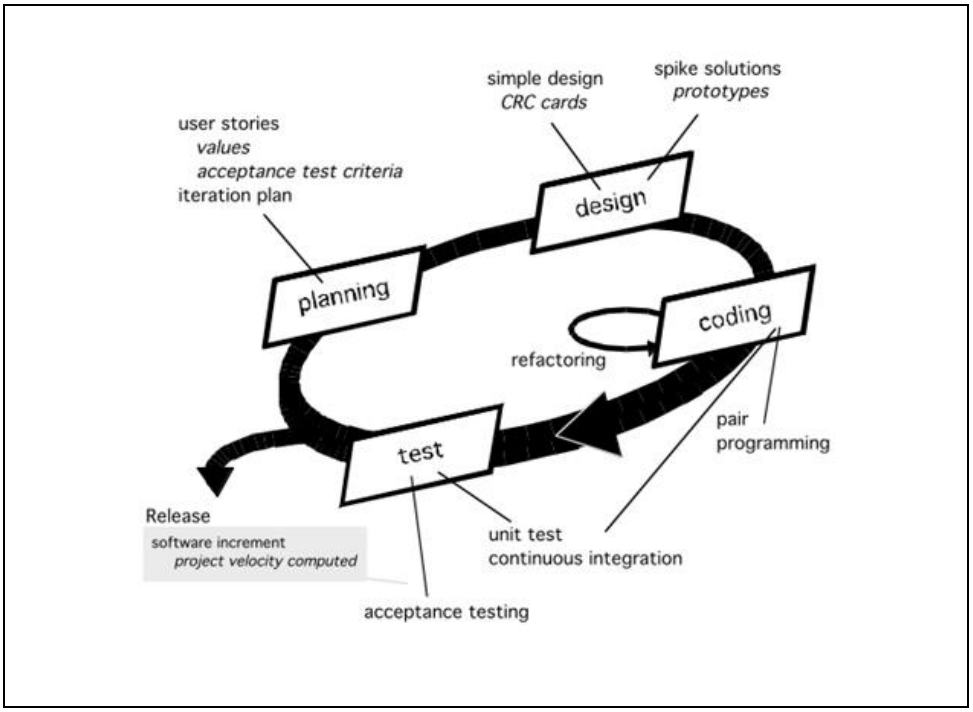
**Tabel 4.5. Simbol-simbol *activity* diagram**

| No. | Simbol | Deskripsi |
|---|---|---|
| **1.** | Status awal  | Satatus awal aktivitas sistem,sebuah diagram aktivitas memiliki sebuah status awal. |
| **2.** | Aktivitas   | Aktifitas yang dilakukan sistem,aktifitas biasanya diawali dengan kata kerja |
| **3.** | Percabangan / *decision*  | asosiasi percabangan |

| No. | Simbol | Deskripsi |
|---|---|---|
| | | dimana jika ada pilihan aktifitas lebih dari satu |
| **4.** | Penggabungan / *join*  | Asosiasi penggabungan dimana lebih dari satu aktivitas digabungkan menjadi satu |
| **5.** | Status akhir  | Status akhir yang dilakukan sistem, sebuah diagram aktivitas memiliki sebuah status akhir |
| **6.** | Swimlane  | Memisahkan organisasi bisnis yang bertanggung jawab terhadap aktifitas yang terjadi |

## 4.6. Metode Penelitian *Extreme Programming*

Menurut Suryantara (2014:24), para pengembang perangkat lunak banyak menggunakan metodologi *Extreme Programming* untuk mengembangkan perangkat lunak dengan cepat. Tahapan pengembangan perangkat lunak dengan *XP (Extreme Programming)* meliputi: *Planning, Design, Coding dan Testing.*berikut ini adalah gambar metodologi *extreme programming* dapat dilihat pada gambar 4.1.




**Gambar 4.1 Metodologi *Extreme Programming***

1. *Planning*

Tahap ini dimulai dengan pemahaman konteks dari aplikasi,mendefinisikan *output* , fitur yang ada pada aplikasi, fumgsi dari aplikasi yang dibuat, penentuan waktu dan biaya pengembangan aplikasi, serta alur pengembangan aplikasi.

Maka pada tahap ini penulis akan melakukan perancangan algoritma enkripsi dan dekripsi menggunakan *flowchart* dan akan merancang sistem berorientasi objek dengan metode UML (*Unified Modelling Language)* menggunakan *use case, activity* diagram, *class* diagram dan menjelaskan *Struktur table* pada sistem.

2. *Design*

Tahap ini menekankan pada desain aplikasi secara sederhana. Di tahap ini penulis akan merancang desain rancangan awal sistem sederhana yaitu *wireframe.*

3. *Coding*

Pada tahap ini penulis akan menyiapkan kode pada aplikasi dan akan memperlihatkan bentuk sistem yang sudah dibuat dari bentuk *interface* yang ada pada sistem.

4. *Testing Bruteforce*

*Bruteforce* adalah metode pencarian *password* dengan mencoba segala kemungkinan yang ada. Ada juga cara lain yang biasanya juga dilakukan oleh *software*, yaitu metode *dictionary*. Untuk metode *bruteforce* akan memakan waktu yang cukup lama, karena akan mencoba segala kemungkinan yang ada.

Misalnya membongkar *password* dengan panjang lima karakter (huruf), berarti ada $256^5$ kemungkinan ( 256 adalah semua karakter ASCII ). Berarti ada 1.099.511.627.776. kemungkinan.

# BAB V

# HASIL DAN PEMBAHASAN

## 5.1. Hasil

Berikut ini adalah tahapan-tahapan metode *extreme programming* yang penulis lakukan untuk pembuatan sistem enkripsi dan dekripsi dengan penerapan penggabungan algoritma *XOR, Rot47 dan Base64* pada sistem yang dibuat.

### 5.1.1. *Planning*

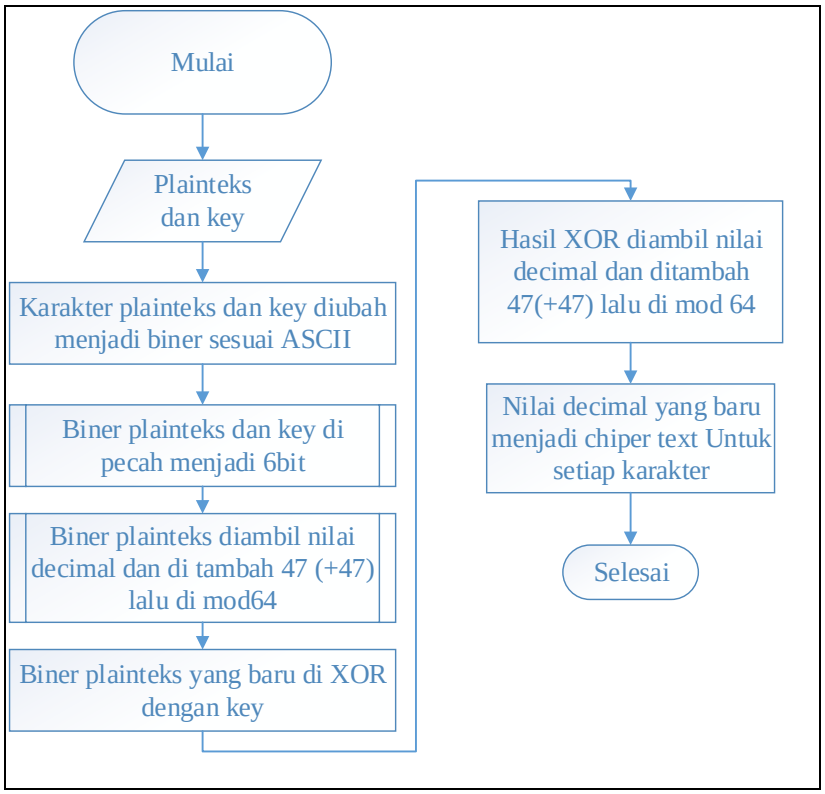
Pada tahapan ini penulis membuat perancangan proses enkripsi algoritma dan dekripsi algoritma menggunakan *flowchart* dan merancang UML (*Unified Modelling Language)* untuk sarana perancangan sistem berorientasi objek ini.

#### 5.1.1.1. *Flowchart* Proses Penggabungan Algoritma

Pada proses ini penulis membuat perancangan *flowchart* enkripsi dan dekripsi dapat dilihat *flowchart* dibawah ini :

1. *Flowchart* Enkripsi

   Adapun proses enkripsi yang berjalan dapat dilihat pada gambar 5.1.




**Gambar 5.1 *Flowchart* Enkripsi**

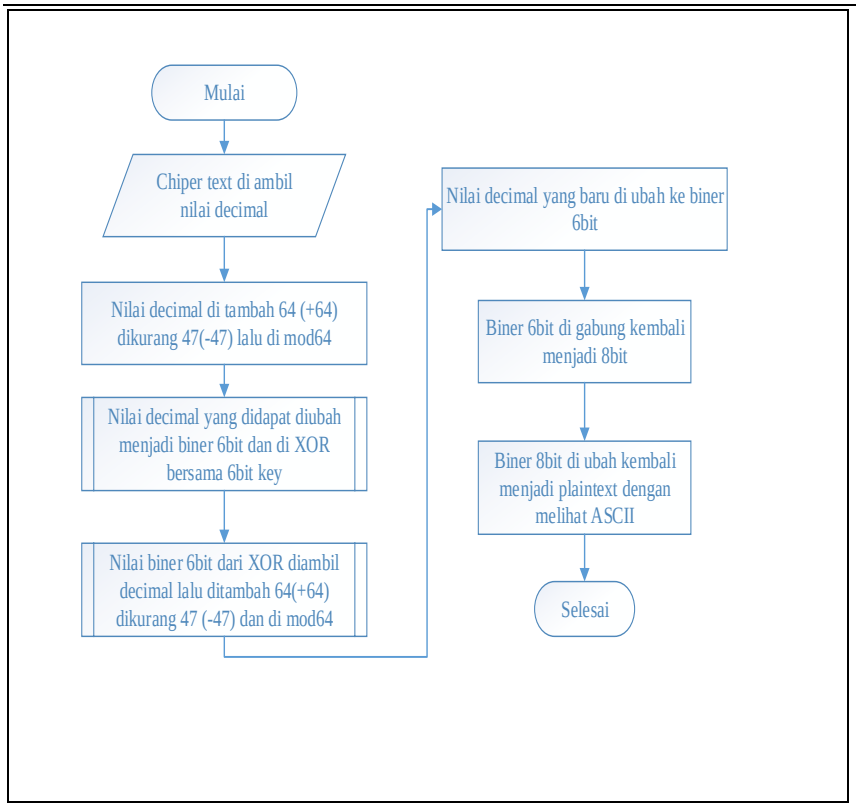
Adapun tahapan yang ada pada gambar 5.1 *flowchart* enkripsi adalah sebagai berikut :

1. Plainteks dan key di ubah menjadi biner sesuai ASCII bit per karakter
2. Biner ASCII 8bit pada plainteks dan key dipecah menjadi 6bit
3. Biner plainteks 6bit diambil nilai *decimal* , ditambah 47(+47) lalu di mod64
4. 6bit pada plainteks yang telah diubah kemudian di lakukan proses *XOR* bersama dengan key

5. Hasil *XOR* mendapatkan 6bit yang baru, diambil nilai *decimal* dan ditambah 47(+47) lalu di mod 64
6. Nilai *decimal* terakhir yang didapatkan menentukan *chiper text* untuk setiap karakter.

2. *Flowchart* Dekripsi

Adapun proses dekripsi yang berjalan dapat dilihat pada gambar 5.2.




**Gambar 5.2 *Flowchart* Dekripsi**

Adapun tahapan yang ada pada gambar 5.2 *flowchart* dekripsi adalah sebagai berikut :

1. *Chiper text* diambil nilai decimal berdasarkan tabel *base64.*

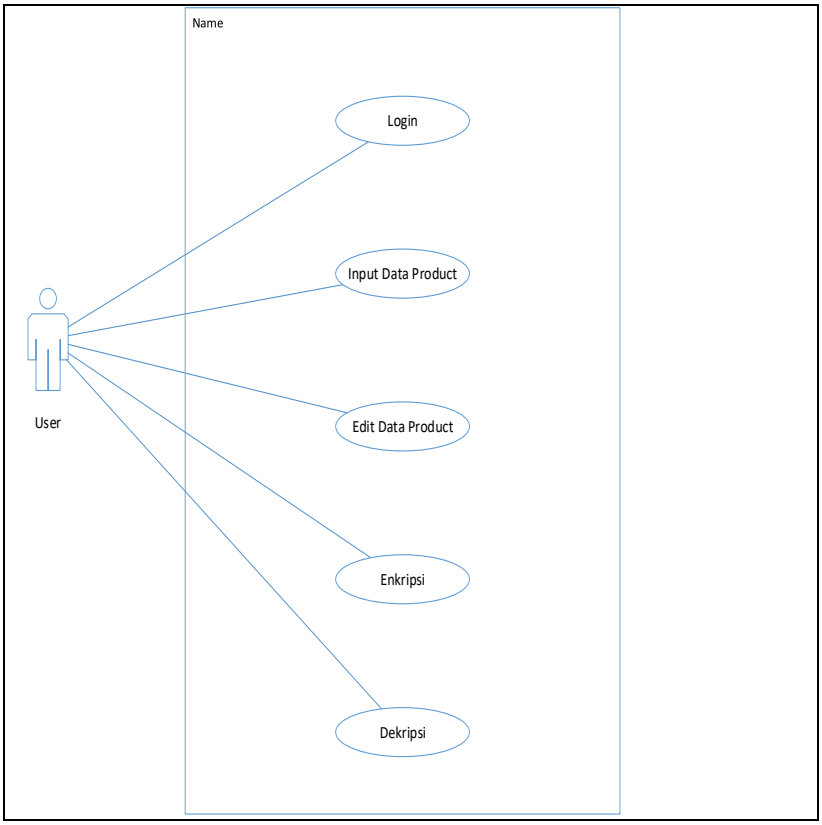
2. Nilai *decimal* ditambah 64(+64), dikurang 47(-47) lalu di mod64.
3. Nilai *decimal* yang telah didapatkan diubah menjadi biner 6bit lalu di *XOR* dengan biner 6bit key yang sudah diketahui.
4. Biner 6bit hasil *XOR* kemudian diambil nilai *decimal* lalu ditambah 64(+64) dikurang 47(-47) dan di mod64.
5. Nilai *decimal* yang didapat diubah kembali menjadi biner 6bit plainteks.
6. Biner 6bit digabung kembali menjadi 8bit dan diambil karakter plainteks sesuai ASCII.

### 5.1.2. UML *(Unified Modelling Language)*

Pada tahapan UML *(Unified Modelling Language)* yang akan di buat terdapat *use case*, *activity* dan *class* diagram untuk menjelaskan tentang desain sistem yang akan dibuat dengan pemrograman berorientasi objek. Adapun beberapa tahapan tersebut sebagai berikut:

1. *Use Case* Diagram

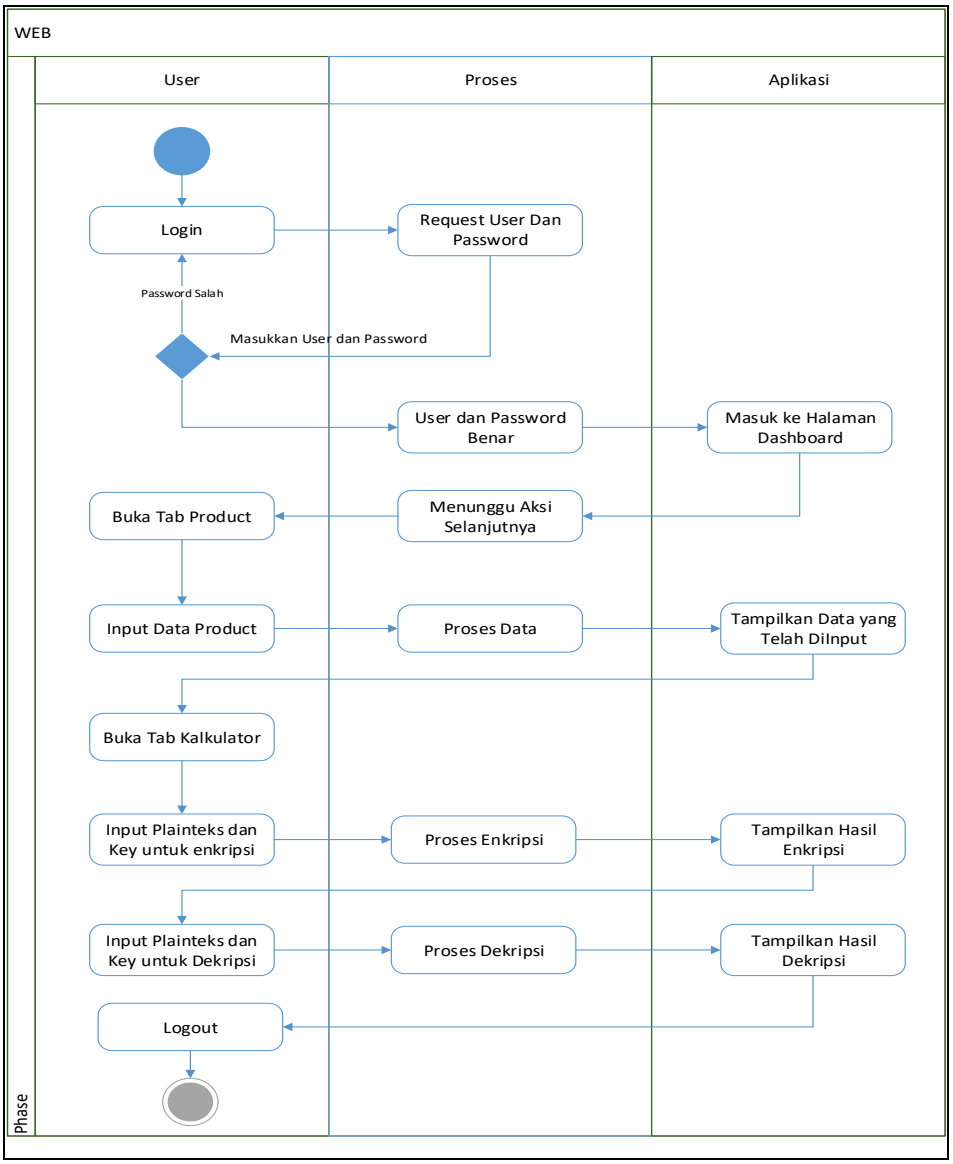
   *Use case* yang penulis rancang nantinya dapat sesuai dengan sistem yang dinginkan yang mana penggunanya dapat mengakses sistem ini yang bertujuan untuk bahan pengujian algoritma. Adapun dapat dilihat pada gambar 5.3 dibawah ini:




**Gambar 5.3 *Use Case***

2. *Activity* Diagram

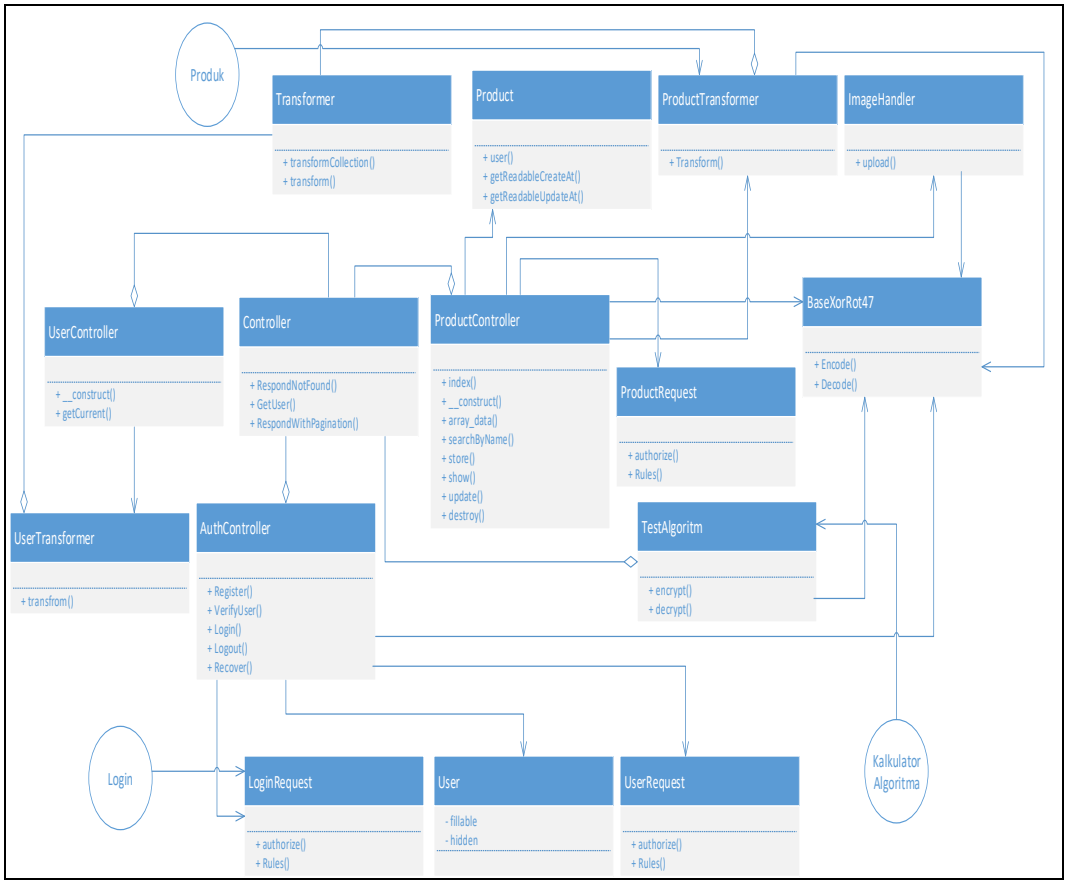
*Activity* diagram pada sistem yang dibuat ini menjelaskan proses dalam penggunaan sistem ini. Adapun penjelasan *activity* diagram dibawah ini pada gambar 5.4 sebagai berikut :




**Gambar 5.4 *Activity* Diagram**

3. *Class* Diagram

*Class* diagram pada sistem ini menampilkan nama *class* objek, atribut yang ada pada sistem, dan metode yaitu operasi kelas sistem yang ada. Adapun penjelasan *class* pada sistem ini dapat dilihat pada gambar 5.5 dibawah ini.




**Gambar 5.5 *Class* Diagram**

### 5.1.3. Struktur Tabel

Struktur tabel-tabel yang akan dibuat pada sistem ini akan dibuat berisiskan *field*, *type field*, ukuran dan keterangan tabel sehingga tabel tersebut dapat digunakan untuk menampung data yang ada pada sistem yang akan dibuat. Adapun tabelnya sebagai berikut:

1. Tabel *Migrations*

   Tabel *Migrations* ini merupakan tabel dari *Laravel*, yang berisikan *field id*, *migration*, dan *batch*. Adapun lebih jelas dapat dilihat pada tabel 5.1 dibawah ini.

**Tabel 5.1 Tabel *Migrations***

| No. | *Field Name* | *Type* | *Size* | Ket |
|---|---|---|---|---|
| 1. | *Id* | *Integer* | 10 | *Primary Key, Auto_Increment* |
| 2 | *Migration* | *Varchar* | 255 | - |
| 3 | *Batch* | *Integer* | 11 | - |

2. Tabel *Password_Resets*

Tabel *Password_Resets* ini merupakan tabel dari *Laravel,* yang berisikan *field Email, Token, dan Created_at.* Adapun tabel dapat dilihat pada tabel 5.2.

**Tabel 5.2 Tabel *Password_Resets***

| No. | *Field Name* | *Type* | *Size* | Ket |
|---|---|---|---|---|
| 1. | *Email* | *Varchar* | 255 | - |
| 2 | *Token* | *Varchar* | 255 | - |
| 3 | *Created_at* | *Timestamp* | - | - |

3. Tabel *Products*

Tabel *Products* merupakan tabel yang ada pada sistem yang nantinya akan menampilkan *field* berupa *Id, Name, Price, Description, Image_Path, Last_updated_by, Created_at, Updated_at.* Adapun lebih jelas dapat dilihat pada tabel 5.

**Tabel 5.3 Tabel *Products***

| No. | *Field Name* | *Type* | *Size* | Ket |
|---|---|---|---|---|
| 1. | *Id* | *Integer* | 10 | *Primary Key, Auto_Increment* |
| 2 | *Name* | *Varchar* | 255 | - |
| 3 | *Price* | *Integer* | 11 | - |
| 4 | *Description* | *Varchar* | 255 | - |
| 5 | *Image_Path* | *Varchar* | 255 | - |
| 6 | *Last_updated_by* | *Integer* | 11 | - |
| 7 | *Created_at* | *timestamp* | - | - |
| 8 | *Updated_at* | *timestamp* | - | - |

4. Tabel *Users*

Tabel *Users* merupakan tabel untuk pengguna mengakses sistem tersebut yang terdapat *field Id, Name, Email,Password, Remember_token, Created_at, Updated_at* dan *Is_verified.* Adapun dapat dilihat dengan jelas pada tabel 5.4.

**Tabel 5.4 Tabel *Users***

| No. | *Field Name* | *Type* | *Size* | Ket |
|---|---|---|---|---|
| 1. | *Id* | *Integer* | 10 | *Primary Key, Auto_Increment* |
| 2 | *Name* | *Varchar* | 255 | - |
| 3 | *Email* | *Varchar* | 255 | - |

| No. | *Field Name* | *Type* | *Size* | Ket |
|---|---|---|---|---|
| 4 | *Password* | *Varchar* | 255 | - |
| 5 | *Remember_token* | *Varchar* | 100 | - |
| 6 | *Created_at* | *Timestamp* | | - |
| 7 | *Updated_at* | *Timestamp* | | - |
| 8 | *Is_verified* | *Tinyint* | 1 | - |

5. Tabel *User_verifications*

*Tabel User_verifications* merupakan tabel yang ada pada *Laravel,* sehingga terdapat *field Id,User_id* dan *Token.* Adapun lebih jelas dapat dilihat pada tabel 5.5.

**Tabel 5.5 Tabel *User_verifications***

| No. | *Field Name* | *Type* | *Size* | Ket |
|---|---|---|---|---|
| 1. | *Id* | *Integer* | 10 | *Primary Key, Auto_Increment* |
| 2 | *User_id* | *Integer* | 10 | - |
| 3 | *Token* | *Varchar* | 255 | - |

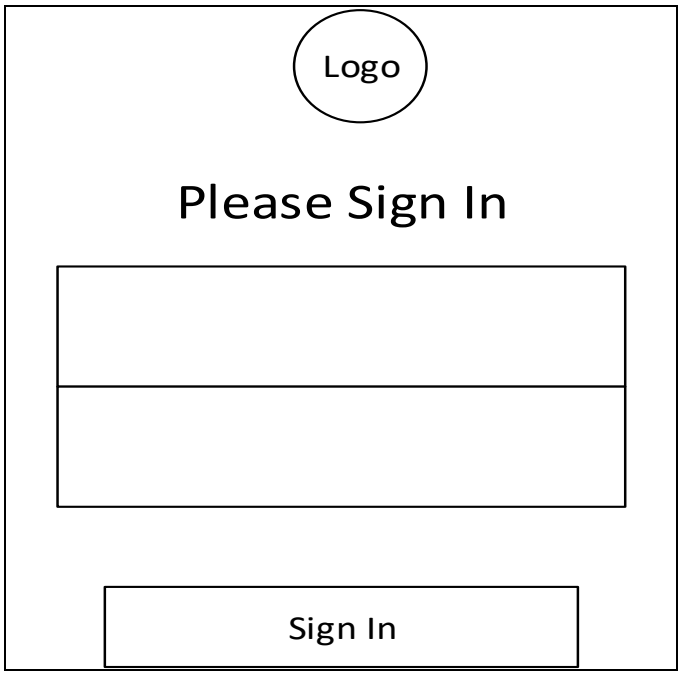
## 5.2. *Design*

Pada tahapan ini penulis merancang bentuk awal tampilan sistem yang akan dibuat dengan model *Wireframe.* Rancangan awal desain sistem ini akan dibuat dalam bentuk versi *web base* dan versi *android*-nya sebagai bentuk penerapan di berbagai *platform***.**

### 5.2.1. *Wireframe* dalam bentuk *Web Base*

1. Tampilan *Login*

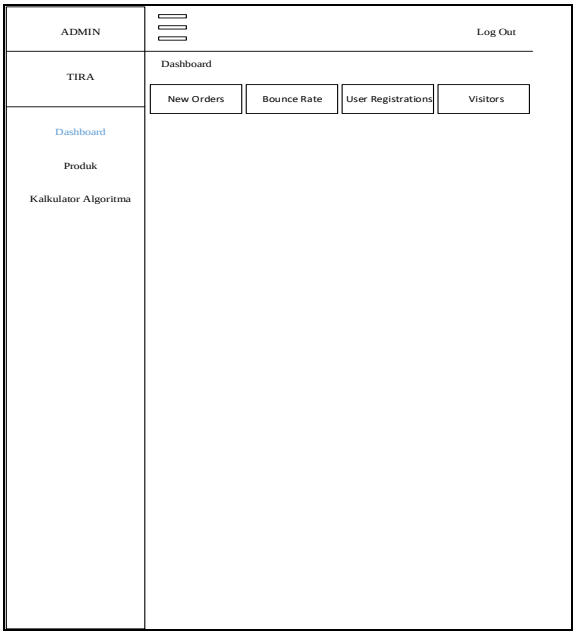
Desain tampilan *login* digunakan pengguna untuk masuk ke halaman beranda:




**Gambar 5.6 Halaman *Login Web Base***

Pada Halaman *Login* ini juga akan diterapkan proses enkripsi untuk segi keamanan *password* untuk meningkatkan keamanan data pengguna.

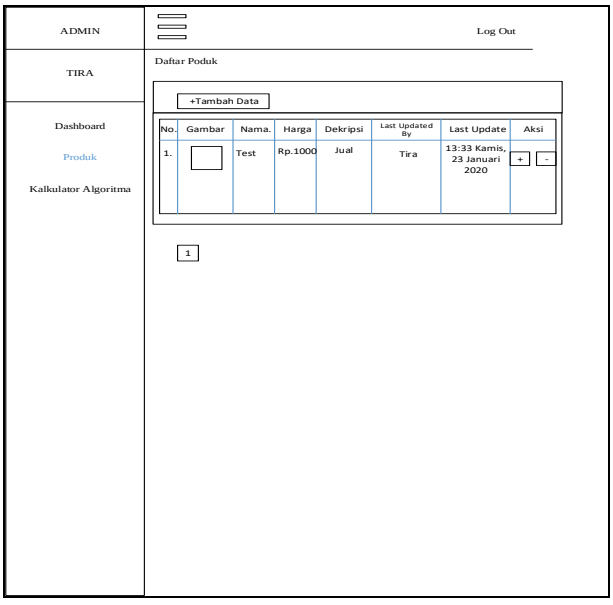
**2.** Tampilan beranda sistem *web base*




**Gambar 5.7 Tampilan *Dashboard* Sistem**

Pada tampiahan halaman depan yaitu *dashboard* hanya menampilkan keterangan sebagai notifikasi yang menjelaskan berapa orderan, pengunjung pada sistem.

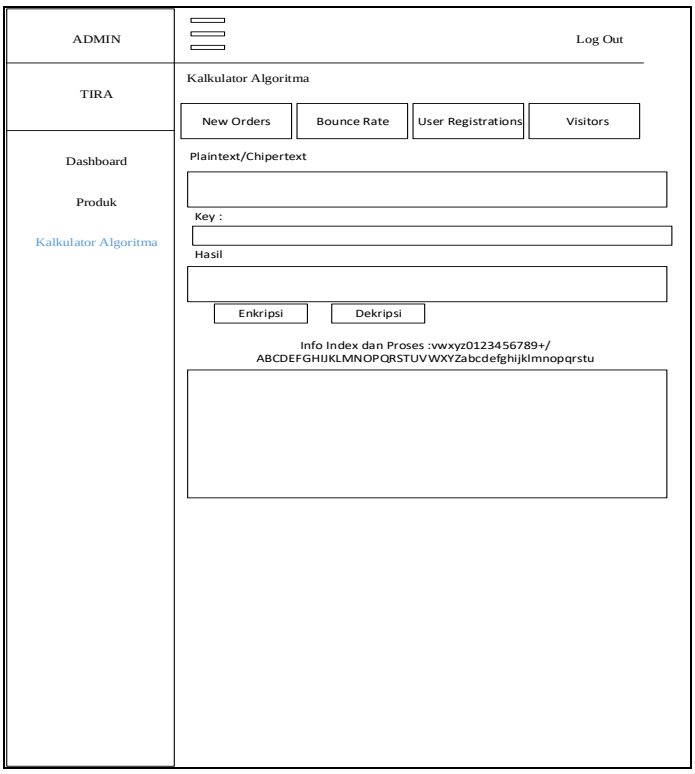
3. Tampilan Data Produk




**Gambar 5.8 Tampilan Data Produk**

Pada Halaman Ini Terdapat Data Produk yang sudah di enkripsi atau dekripsi, sehingga nantinya data yang di input akan muncul di bagian halaman ini. Di halaman ini langsung bisa menambahkan produk secara langsung. Apabila data yang di *input* salah juga dapat dirubah di halaman ini maupun di hapus.

4. Tampilan Kalkulator Algoritma



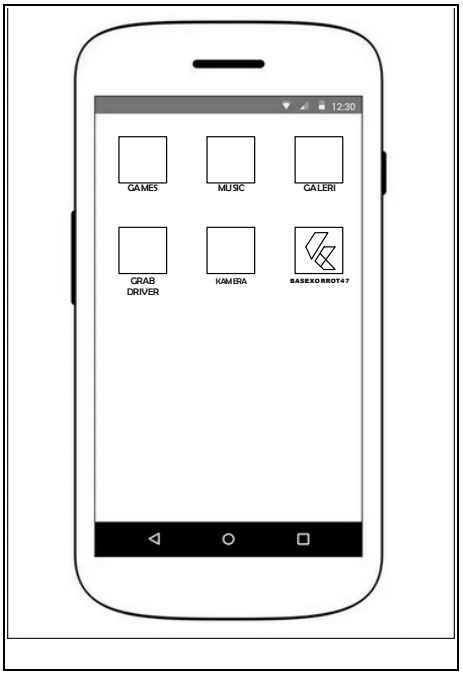

**Gambar 5.9 Tampilan Kalkulator Algoritma**

Pada rancangan tampilan halaman kalkulator algoritma kita dapat menggunakan *tools* enkripsi dan dekripsi algoritma sebagai bahan uji coba dan pemanfaatan enkripsi dan dekripsi data**.** Halaman ini nantinya juga akan menampilkan proses enkripsi atau enkripsi data yang di buat.

### 5.2.2. *Wireframe* Dalam Bentuk *Android*

Pada rancangan ini penulis menyajikan bentuk rancangan awal dalam bentuk *android* sebagai bentuk penerapan *multi PlatForm.*

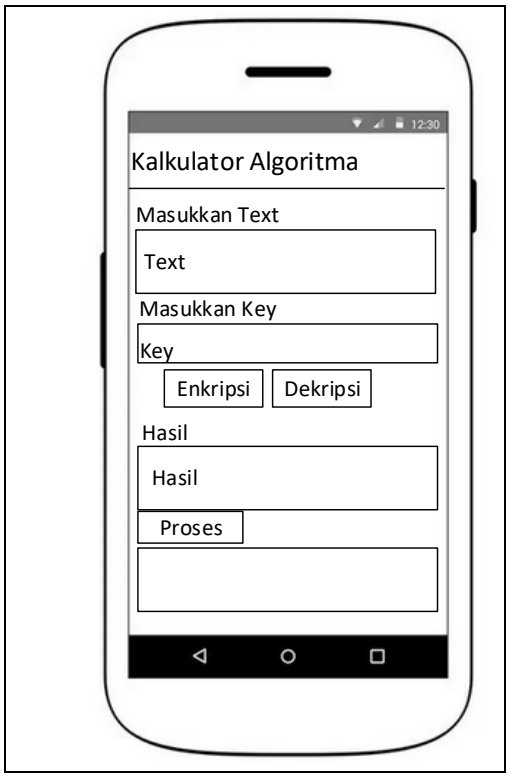
1. Tampilan *Home* Menu Aplikasi




**Gambar 5.10 Tampilan *Home* Pada *Android***

Pada halaman tampilan pada gambar 5.10 dapat dilihat menu aplikasi kalkulator algoritma yang di gunakan di platform bebrbentuk *android,* sehingga penerapannnya dapat digunakan pada berbagai *PlatForm.*

2. Tampilan Kalkulator Algoritma *Android*




**Gambar 5.11 Tampilan Aplikasi Kalkulator Algoritma pada *Android***

Pada tampilan ini kalkulator algoritma di rancang dalam bentuk *android* sehingga nantinya *tools* algoritma ini dapat digunakan di berbagai *PlatForm* yang menggunakan *Rest-API* sebagai penghubungnya.

## 5.3. *Coding*

Pada tahapan ini penulis akan membuat bentuk sistem sehingga sistem ini dapat digunakan untuk penerapan proses enkripsi dan dekrisi data menggunakan proses penggabungan Algoritma *XOR*, *Rot47* dan *Base64*. Pada tahapan ini pula penulis akan menyajikan bentuk penerapan sistem pada *multi platform* yang penerapannya menggunakan *Rest-API* sebagai penghubungnya.

### 5.3.1. Penggunaan *Rest-API*

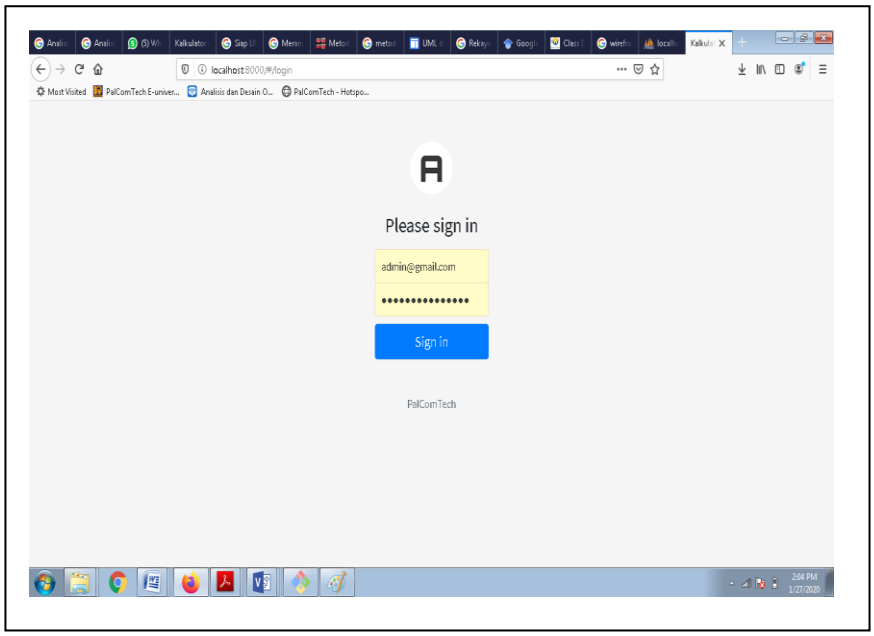
Penggunaan *Rest-API* ini digunakan penulis untuk menghubungkan sistem dapat di gunakan pada berbagai *platform* sehingga dapat mendukung penerapan penggabungan algoritma keamanan data yang dibuat lebih *efisien* di gunakan di berbagai bentuk *platform*. Adapun bentuk tampilan *view interface rest-API* dapat dilihat pada gambar 5.12 sebagai berikut

```
ButtonBar(
  alignment: MainAxisAlignment.start,
  children: <Widget>[
    FlatButton(
      child: Text('Enkripsi'),
      color: Colors.blue,
      onPressed: () async {
        var client = http.Client();
        String url = 'http://restapi.fn-computer.com/api/encrypt';
        var response = await client.post(url, body: {'text': textController.text, 'key': keyController.text});
        print('Response status: ${response.statusCode}');
        var data = json.decode('${response.body}');
        print('Response body: $data');
        resultController.text = '${utf8.decode(base64.decode(data["result"]))}';
      },
    ), // FlatButton
    FlatButton(
      child: Text('Deskripsi'),
      color: Colors.redAccent,
      onPressed: () async {
        var client = http.Client();
        String url = 'http://restapi.fn-computer.com/api/decrypt';
        var response = await client.post(url, body: {'text': textController.text, 'key': keyController.text});
        print('Response status: ${response.statusCode}');
        var data = json.decode('${response.body}');
        print('Response body: $data');
        resultController.text = '${utf8.decode(base64.decode(data["result"]))}';
      },
    ), // FlatButton
  ], // <Widget>[]
), // ButtonBar
```

**Gambar 5.12 Tampilan *Interface Rest-API***

### 5.3.2. Penerapan Algoritma dan *Interface* Pada Halaman *Login*

Pada halaman ini penulis menampilkan bentuk *interface* halaman *login*. Adapula bentuk *interface* dapat dilihat pada gambar 5.13 sebagai berikut:



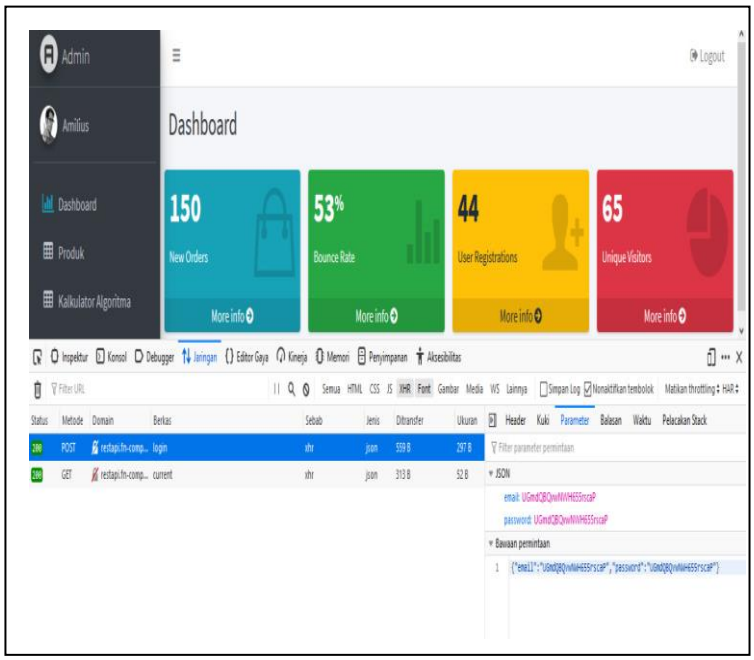

**Gambar 5.13 Tampilan Halaman *Login***

Pada halaman ini penulis menampilkan bentuk *interface login* yang di buat. Apabila pengguna sudah dapat *login* maka akun pengguna akan secara langsung terenkripsi menggunakan algoritma penggabungan yang dibuat penulis. Adapun bentuk enkripsi data pada *login* dapat dilihat pada gambar 5.14.

### 5.3.3. Tampilan *Interface* Halaman *Dashboard* dan Penerapan Algoritma Pada Halaman *Login*

1. Tampilan *Dashboard* dan Rincian Penerapan Algoritma *Login*

Pada halaman ini penulis akan memaparkan bentuk *interface* halaman *dashboard* dan penerapan penggabungan algoritma yang penulis buat pada gambar 5.14.

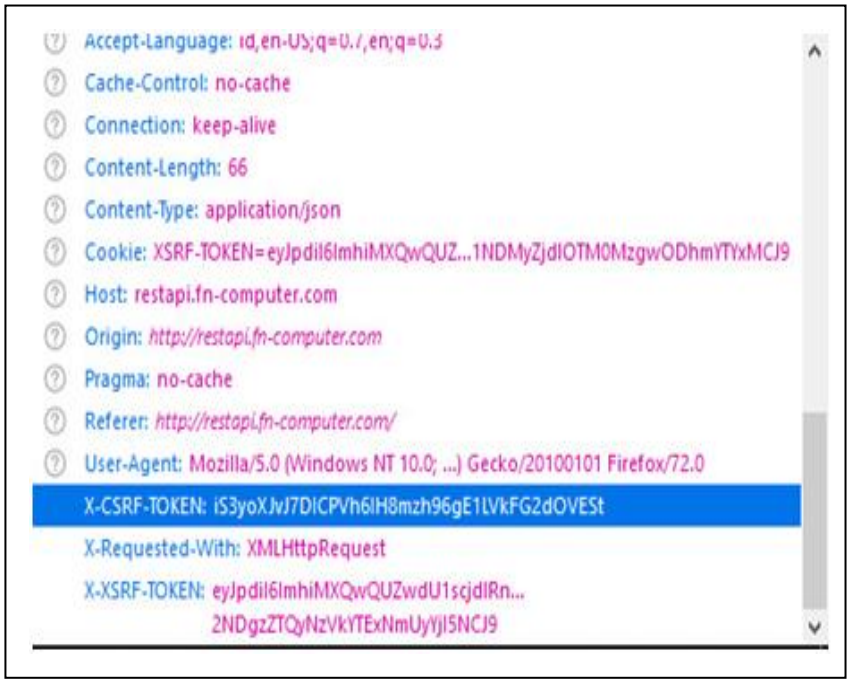



**Gambar 5.14 Tampilan *Dashboard* dan Bentuk Penerapan Penggabungan Algoritma pada Halaman *Login***

Pada halaman *dashboard* ini penulis menampilkan bentuk penggabungan enkripsi pada halaman *login*. Dapat dilihat pada gambar 5.14. pada bagian parameter email dan password sudah terenkripsi.

2. *Key* Yang Digunakan

Adapula *key* yang dipakai pada enkripsi *login* dapat dilihat pada gambar 5.15 sebagai berikut




**Gambar 5.15 *Key* Halaman *Login***

Pada penerapan penggabungan algoritma pada halaman *login key* yang di gunakan pada proses enkripsi adalah *CSRF-Token*.

3. Pengujian enkripsi halaman *login*

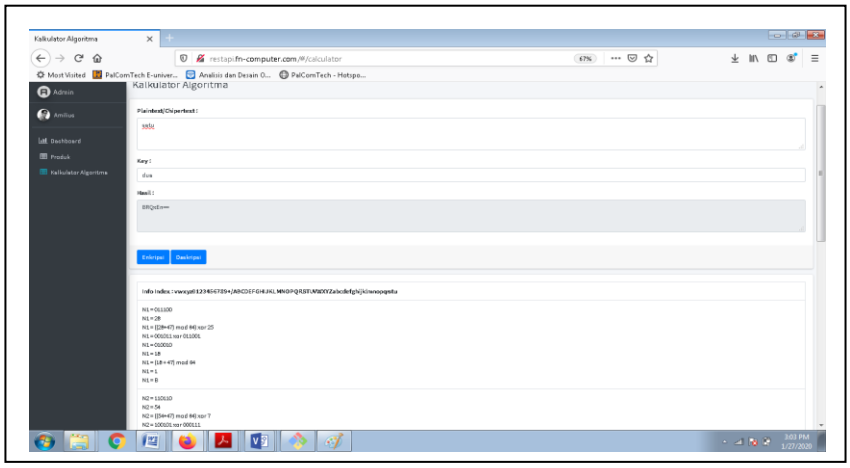
Pada tahap ini penulis menampilkan proses enkripsi yang ada pada gambar 5.14 sesuai dengan pengujian yang ditampilkan pada gambar 5.16.


**Gambar 5.16 Pengujian Enkripsi**

### 5.3.4. Tampilan *Interface Tools* Kalkulator Algoritma

Pada tahap ini penulis menampilkan bentuk *interface* dari halaman kalkulator algoritma yang berfungsi sebagai *tools* penerjemah proses enkripsi dan dekripsi pada penerapan penggabungan algoritma *XOR*, *Rot47* dan *Base64*.

1. Pengujian *Tools* Kalkulator Algoritma Enkripsi




**Gambar 5.17 Tampilan Kalkulator Algoritma Enkripsi**

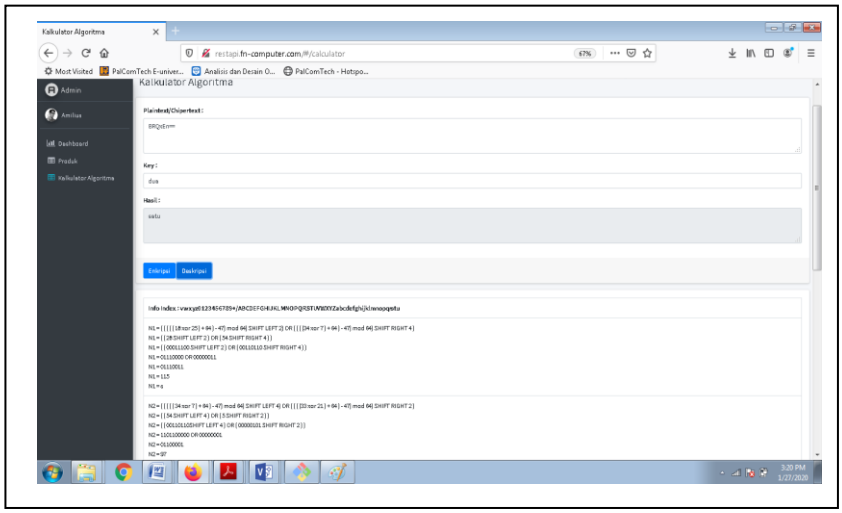
Pada tahapan ini penulis melakukan pengujian pada kalkulator algoritma pada gambar 5.17. dengan plainteks satu dan *key*nya berupa dua maka hasilnya adalah BRQxEn==. Penulis juga melakukan perbandingan hasil dengan metode manual dapat dilihat pada tabel 5.6 sebagai berikut :

**Tabel 5.6. Pengujian Manual Enkripsi**

| ***Plaintext*** | | | | | | | |
|---|---|---|---|---|---|---|---|
| ***s*** | ***a*** | ***t*** | ***u*** | | | | |
| *0111 0011* | *0110 0001* | *0111 0100* | *0111 0101* | | | | |
| *Encode 6bit* | | | | | | | |
| *011 100* | *110 110* | *000 101* | *110 100* | *011 101* | *01 (0000)* | | |
| *28* | *54* | *5* | *52* | *29* | *16* | | |
| *(+47)* | | | | | | | |
| *75* | *101* | *52* | *99* | *76* | *63* | | |
| *MOD64* | | | | | | | |

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| *11* | *37* | *52* | *39* | *12* | *63* | | |
| *001 011* | *100 101* | *110 100* | *100 011* | *001 100* | *111 111* | | |
| | | | | | | | |
| *KEY* | | | | | | | |
| ***d*** | ***u*** | ***a*** | | | | | |
| *0110 0100* | *0111 0101* | *0110 0001* | | | | | |
| | | | | | | | |
| *Encode 6bit* | | | | | | | |
| *011 001* | *000 111* | *010 101* | *100 001* | | | | |
| *25* | *7* | *21* | *33* | | | | |
| *001 011* | *100 101* | *110 100* | *100 011* | *001 100* | *111 111* | | |
| *011 001* | *000 111* | *010 101* | *100 001* | *011 001* | *000 111* | *Key* | |
| *XOR* | | | | | | | |
| *010 010* | *100 010* | *100 001* | *000 010* | *010 101* | *111 000* | | |
| *18* | *34* | *33* | *2* | *21* | *56* | | |
| *(+47)* | | | | | | | |
| *65* | *81* | *80* | *49* | *68* | *103* | | |
| *MOD64* | | | | | | | |
| *1* | *17* | *16* | *49* | *4* | *39* | | |
| *B* | *R* | *Q* | *x* | *E* | *n* | = | = |

2. Pengujian *Tools* Kalkulator Dekripsi




**Gambar 5.18 Pengujian Kalkulator Algoritma Dekripsi**

Pada tahapan ini penulis melakukan pengujian pada kalkulator algoritma pada gambar 5.17. dengan *chipertext* BRQxEn== dan *key*nya berupa dua maka hasilnya adalah satu. Penulis juga melakukan perbandingan hasil dengan metode manual dapat dilihat pada tabel 5.7 sebagai berikut :

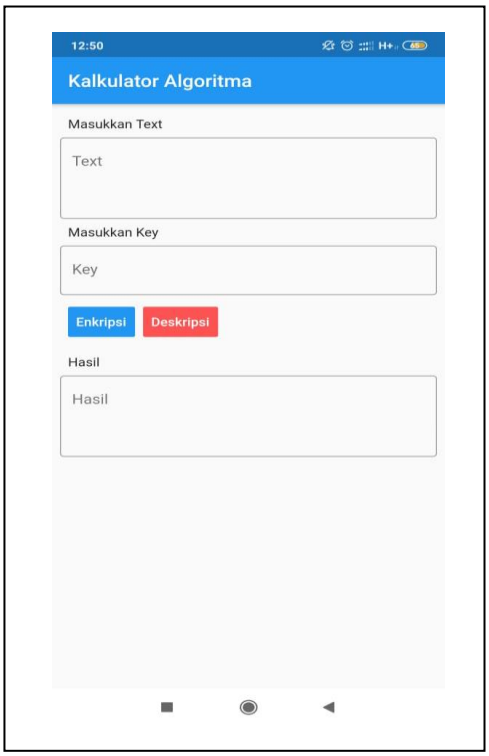
**Tabel 5.7 Pengujian Manual Enkripsi**

| ***Chipertext*** | | | | | | | |
|---|---|---|---|---|---|---|---|
| *B* | *R* | *Q* | *x* | *E* | *n* | = | = |
| *1* | *17* | *16* | *49* | *4* | *39* | | |
| *(+64)* | | | | | | | |
| *65* | *81* | *80* | *113* | *68* | *103* | | |
| *(-47)* | | | | | | | |
| *18* | *34* | *33* | *66* | *21* | *56* | | |
| *MOD64* | | | | | | | |
| *18* | *34* | *33* | *2* | *21* | *56* | | |
| *010 010* | *100 010* | *100 001* | *000 010* | *010 101* | *111 000* | | |
| *011 001* | *000 111* | *010 101* | *100 001* | *011 001* | *000 111* | *Key* | |
| *XOR* | | | | | | | |
| *001 011* | *100 101* | *110 100* | *100 011* | *001 100* | *111 111* | | |
| *11* | *37* | *52* | *35* | *12* | *63* | | |
| *(+64)* | | | | | | | |
| *75* | *101* | *116* | *99* | *76* | *127* | | |
| *(-47)* | | | | | | | |
| *28* | *54* | *69* | *52* | *29* | *80* | | |
| *MOD64* | | | | | | | |
| *28* | *54* | *5* | *52* | *29* | *16* | | |
| *011 100* | *110 110* | *000 101* | *110 100* | *011 101* | *010 000* | | |
| *Decode* | | | | | | | |
| *0111 0011* | *0110 0001* | *0111 0100* | *0111 0101* | | | | |
| ***S*** | ***a*** | ***t*** | ***u*** | | | | |

Maka dari hasil pengujian penerapan penggabungan algoritma *XOR*, *Rot47* dan *Base64* pada *tools* kalkulator algoritma yang dibuat penulis dalam bentuk penerjemah proses enkripsi dan enkripsi data sesuai dengan bentuk hasil proses manual enkripsi dan dekripsi yang dibuat.

3. Tampilan *Tools* Kalkulator Algoritma Dalam Bentuk *Android*

Pada tampilan *tools* kalkulator algoritma yang berada pada versi android penulis menggunakan *Rest-API* sebagai penghubung penerapan algoritma yang penulis buat dapat di implementasikan di berbagai bentuk *platfortm.*




**Gambar 5.19 Tampilan *Tools* Kalkulator Pada *Android***

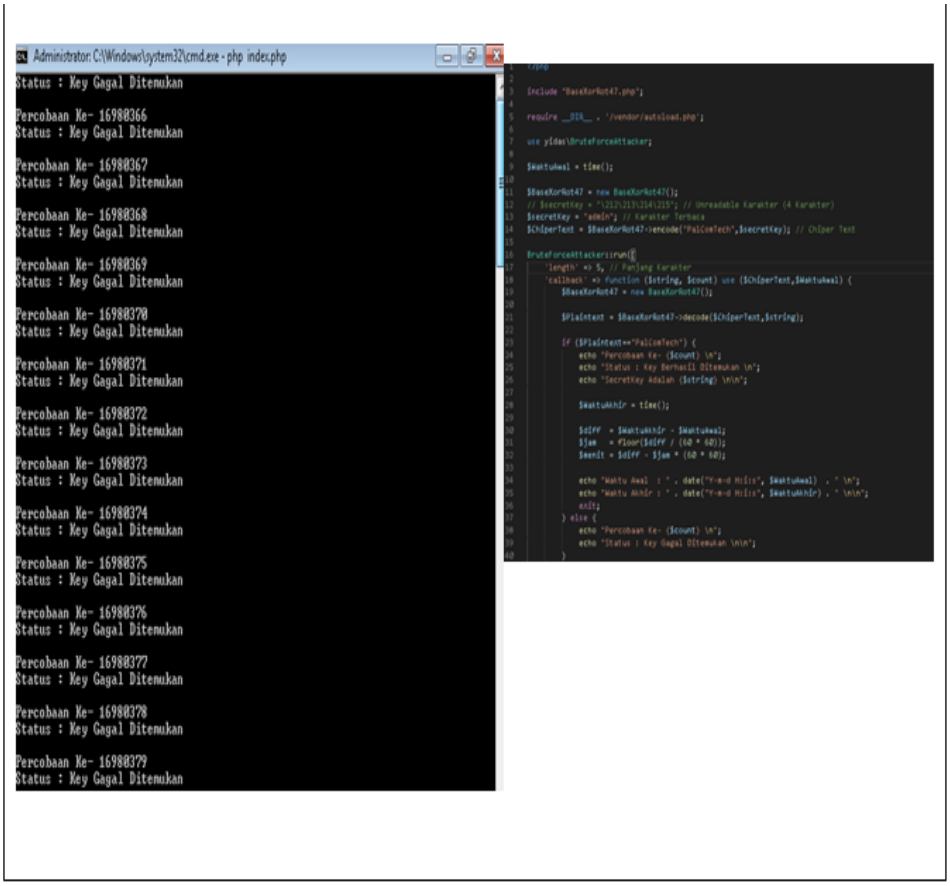
## 5.4. Pengujian

Pada metode *extreme programming* tahapan yang terakhir di lakukan adalah pengujian, maka dari itu penulis melakukan pengujian tingkat kerentanan keamanan pada algoritma yang telah dibuat dengan cara bruteforce untuk melakukannya dan ada pula hasil enkripsi dari penerapan algoritma akan langsung tersimpan pada tempat penyimpanan (*database).*

### 5.4.1. Pengujian Bruteforce

Pada pengujian ini penulis melakukan teknik *bruteforce attack* pada algoritma yang dibuat sehingga bisa mengetahui tingkat kerentanan dari keamanan algoritma yang penulis buat pada gambar dibawah diketahui pada percobaan ke-16.980.379 belum ada muncul

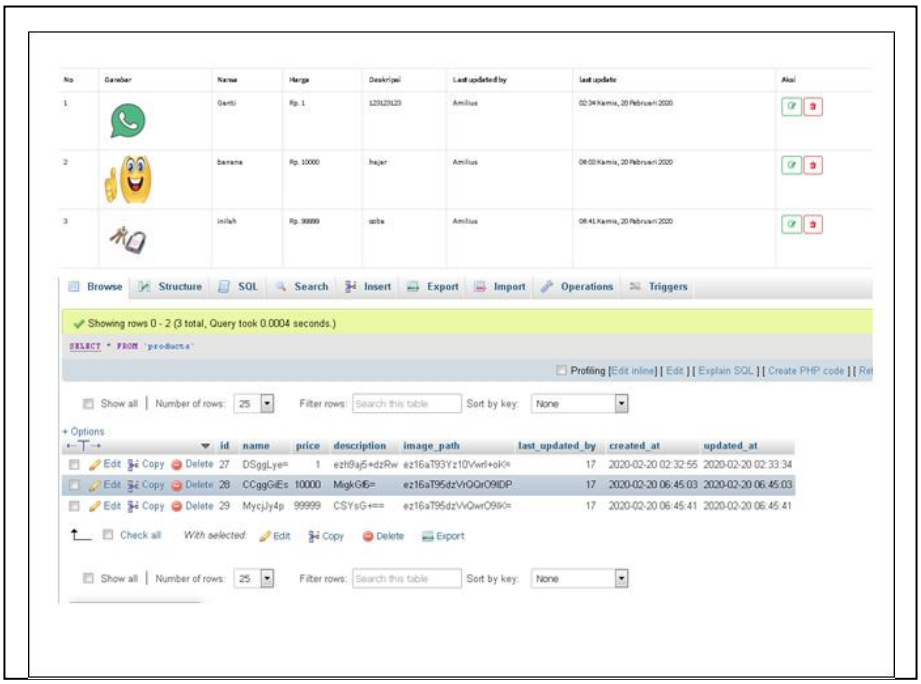
*key* yang dienkripsi menggunakan algoritma yang dibuat dan masih butuh waktu lama untuk mengetahuinya. Untuk lebih detail dapat dilihat pada gambar 5.20 :




**Gambar 5.20 Pengujian Bruteforce**

### 5.4.2. Pengujian Enkripsi *Database*

Pada pengujian ini penulis akan menampilkan hasil dari enkripsi yang dilakukan menggunakan algoritma yang dibuat yaitu *basexorot47* akan langsung tersimpan pada tempat penyimpanan (*database)* sehingga dapat meningkatkan kualitas dari keamanan data. Adapula penerapannya dapat dilihat pada gambar 5.21 :




**Gambar 5.21. Pengujian Enkripsi *Database***

# BAB VI

# KESIMPULAN DAN SARAN

## 6.1. Kesimpulan

Berdasarkan hasil pembahasan pada bab sebelumnya terhadap penggabungan algoritma *XOR*, *Rot47* dan *Base64* untuk peningkatan keamanan data maka dapat diambil kesimpulan sebagai berikut :

1. Penggabungan algoritma *XOR, Rot47* dan *Base64* menjadi sebuah algoritma baru dengan keamanan yang mencukupi dan dapat digunakan untuk pengamanan data.
2. Penulis menambahkan algoritma penggabungan pada proses *login* dengan parameter *email* sebagai plainteks dan *csrf token* sebagai *key*.
3. Penulis menggunakan *rest-api* pada aplikasi kalkulator algoritma penggabungan sehingga bisa diterapkan pada perangkat *android*.
4. Penggabungan algoritma dapat di implementasikan kedalam sistem dengan baik.

## 6.2. Saran

Adapun saran yang dapat dikembangkan untuk penelitian selanjutnya :

1. Algoritma ini dapat digabungkan lagi dengan algoritma lain dimasa yang akan datang agar dapat memperkuat tingkat keamanan data.
2. Menambah implementasi pada *multiplatform* agar algoritma penggabungan ini bisa diterapkan pada *platform* yang lain.