BAB III

ANALISIS DAN DESAIN SISTEM

III.1. Analisis Masalah

Email sudah digunakan orang sejak awal terbentuknya *internet* dan merupakan salah satu fasilitas yang ada pada saat itu. Tak jarang orang menyimpan berbagai data penting pada *email* tersebut. Sepeti informasi akun-akun, nomor rekening relasi, dan masih banyak lainnya. Hal ini dikarenakan orang-orang takut lupa mengenai informasi penting tersebut dan dipilihlah *email* sebagai tempat penyimpanannya. Namun, akibat dari banyaknya orang yang menggunakan *email* tersebut sebagai alat penyimpan informasi, tak sedikit orang yang berbuat nakal untuk mencari tahu mengenai informasi tersebut. Salah satu caranya adalah dengan melakukan *hack* ke *email* sang korban. Permasalahan tersebut dapat diatasi dengan proses enkripsi. Salah satu algoritma enkripsi yang cukup dikenal adalah dengan algoritma enkripsi RC4. Untuk itulah dibutuhkan sebuah aplikasi yang dapat mengirimkan *email* dengan kemampuan untuk melakukan enkripsi dan dekripsi pesan teks pada *email*.

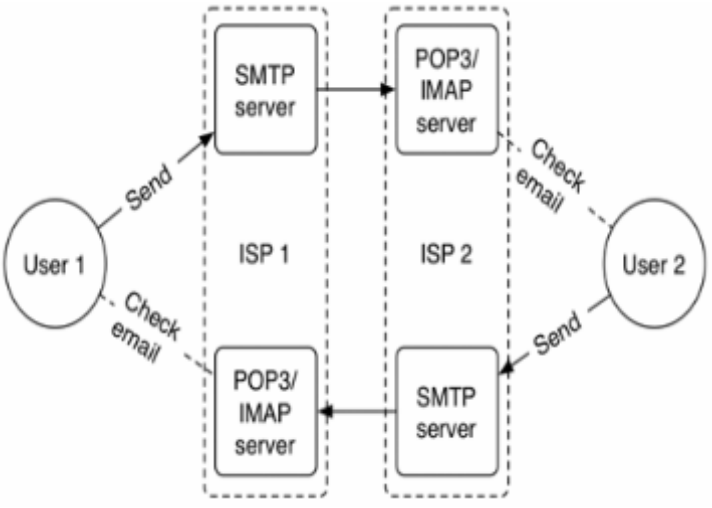
III.1.1. Penerapan Metode

Dalam pembuatan aplikasi ini diterapkan sebuah algoritma kriptografi RC4 yang mana algoritma kriptografi RC4 merupakan algoritma kriptografi kunci simetris yang berbentuk *stream cipher* dimana algoritma ini melakukan proses enkripsi/dekripsi dalam satu byte dan menggunakan kunci yang sama. Sehingga nantinya dalam penggunaannya pengirim dan penerima email sebelumnya akan

menyepakati secara bersama kunci yang akan digunakan untuk proses enkripsi sehingga kunci tersebut juga dapat digunakan untuk proses dekripsi email. algoritma kriptografi RC4 pada aplikasi ini diterapkan pada proses pembuatan pesan email, sehingga sebelum dikirim email tersebut akan di enkripsi oleh pengguna. Hingga nantinya pesan yang akan dikirimkan adalah pesan dalam bentuk ciphertext. Pada proses dekripsi pesan nantinya pengguna sebelumnya akan melihat pesan pada kotak masuk email, untuk melakukan dekripsi pengguna cukup dengan memilih email dengan pesan ciphertext lalu menginputkan kunci yang telah disepakati dalam proses enkripsi sehingga pesan ciphertext tersebut dapat di dekripsi menjadi pesan aslinya.

III.1.2. POP-IMAP-SMTP *Email*

Desain pada program dibuat untuk memudahkan pengguna HP untuk mengirim dan menerima informasi. Desain memiliki persamaan dengan *email client* pada *desktop* yaitu menerima *email* dari *server* POP/IMAP dan mengirim *email* melewati *server* SMTP.



Gambar III.1. Arsitektur email

Daur hidup sebuah pesan *email* adalah sebagai berikut : Pertama, pengirim akan menghubungi SMTP *server*. Sistem akan memberitahu SMTP *server* tentang pengirim dan isi dari pesan. Kedua, SMTP akan mencari *server* penerima *email* tersebut lewat *internet* dan mengirimnya. *Server* penerima dapat berupa POP/IMAP *server*. Ketiga, *server* penerima akan menyimpan *email* tersebut. Kapasitas penyimpanan tergantung pada *server* penerima dan jenis *account email* penerima. Keempat, penerima akan melihat *email* yang dikirim tersebut apabila *login* ke sistem dan meminta *email*.

III.2. Kriptografi Algoritma RC4

Algoritma RC4 ini dikembangkan oleh Ronald Rivest untuk RSA *data security* pada tahun 1987 dan baru dipublikasikan untuk umum pada tahun 1994. RC4 merupakan salah satu algoritma kunci simetris yang berbentuk *stream cipher* dimana algoritma ini melakukan proses enkripsi/dekripsi dalam satu byte dan menggunakan kunci yang sama. RC4 menggunakan variabel yang panjang kuncinya dari 1 sampai 256 bit yang digunakan untuk menginisialisasikan tabel sepanjang 256 bit. Tabel ini digunakan untuk generasi yang berikut dari *pseudo random bit* dan kemudian untuk menggenerasikan aliran *pseudo random* digunakan operasi XOR dengan *plaintext* untuk menghasilkan *ciphertext*. Masing-masing elemen dalam tabel ditukarkan minimal sekali. (Kurniadi ; 2015 : 1-13)

Berikut adalah implementasi algoritma RC4 dengan mode 4 *byte* (untuk lebih menyederhanakan dalam perhitungan manual) serta untuk kebutuhan sistem yang sangat terbatas. S-*Box* dengan panjang 4 *byte*, dengan S[0]=0, S[1]=1, S[2]=2 dan S[3]=3 sehingga *array* S menjadi: 0 1 2 3

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

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

Iterasi 1

i = 0

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

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

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

Hasil Array S

2 1 0 3

Iterasi 2

i = 1

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

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

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

Hasil Array S

1 2 0 3

Iterasi 3

i = 2

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

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

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

Hasil

1 2 3 0

Iterasi 4

i = 3

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

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

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

Hasil Array S

1 2 0 3

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

Array S

1 2 0 3

Inisialisasi

i = 0

j = 0

Iterasi 1

i = (0 + 1) mod 4 = 1

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

4 = 2

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

1 0 2 3

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

mod 4] = 2

K1 = 00000010

Iterasi 2

i = (1 + 1) mod 4 = 2

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

4 = 0

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

2 0 1 3

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

mod 4] = 3

K2 = 00000011

Iterasi 3

i = (2 + 1) mod 4 = 3

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

4 = 3

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

1 0 2 3

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

mod 4] = 2

K3 = 00000010

Iterasi 4

i = (3 + 1) mod 4 = 0

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

4 = 0

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

1 0 2 3

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

mod 4] = 2

K4 = 00000010

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

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

Berikut adalah proses pengXORan dari *plaintext* dengan *key* yang telah didapat:

H A L O : 01001000 01000001 01001100 01001111

Key : 00000010 00000011 00000010 00000010

Ciphertext : 01001010 01000010 01001110 01001101

Proses dekripsi *ciphertext* menggunakan algoritma RC4 ini sama untuk proses *key-schedule*-nya. Untuk mendapatkan *plaintext, ciphertext* yang diperoleh di XORkan dengan *pseudo random byte* yang didapat sebelumnya. Maka hasilnya adalah *plaintext* atau teks asli.

Pesan dikirim dalam bentuk *ciphertext* sehingga setelah sampai di penerima pesan dapat kembali diubah menjadi *plaintext* dengan meng-XOR-kan dengan kunci yang sama. Pemrosesan pesan setelah sampai pada penerima dapat dilihat pada dibawah ini.

Proses XOR *pseudo random byte* dengan *ciphertext* pada dekripsi yaitu:

Ciphertext : 01001010 01000010 01001110 01001101

Key : 00000010 00000011 00000010 00000010

Plaintext : 01001000 01000001 01001100 01001111

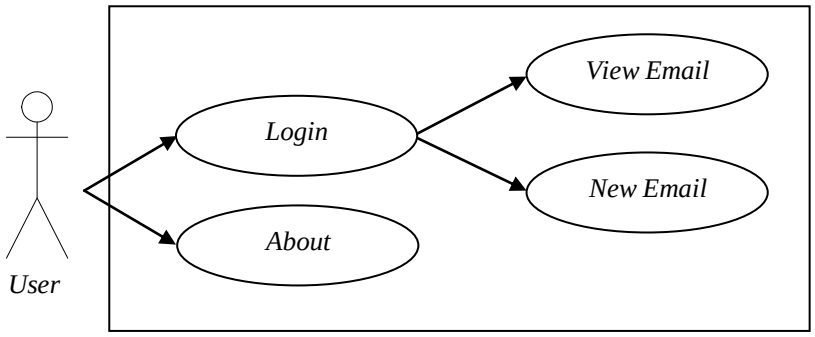
H A L O

III.3. Desain Sistem

Implementasi Algoritma RC4 Dalam Pengamanan Data *Email* Berbasis Android dirancang dengan menggunakan perangkat lunak *Eclipse* Juno. Perancangan sistem yang dirancang terdiri dari *use case*, *flowchart*, *activity diagram* serta desain dan penjelasan dari sistem yang dirancang. Berikut adalah perancangannya :

III.3.1. *Use Case Diagram*

Use case mendiskripsikan sebuah interaksi antara satu atau lebih aktor dengan sistem yang akan dibuat. *Use case* digunakan untuk mengetahui fungsi yang ada didalam sistem informasi tersebut. Berikut adalah *use case diagram* dari sistem yang dirancang :

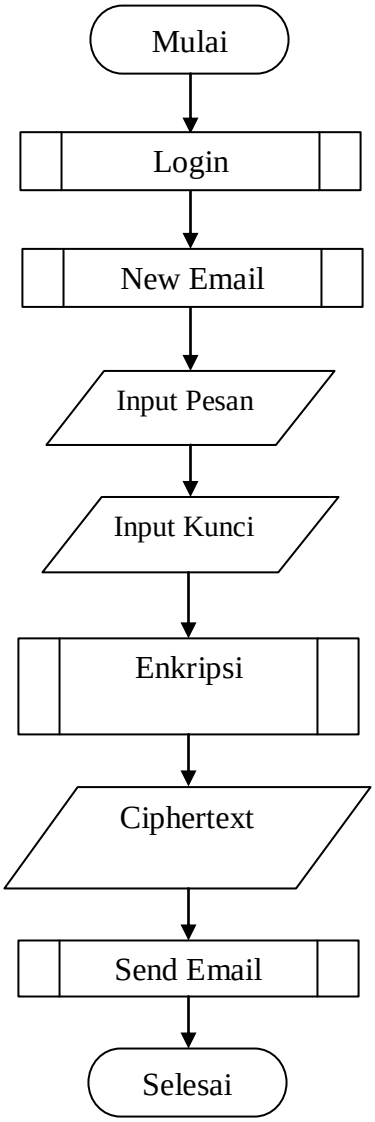


Gambar III.2. *Use Case Diagram*

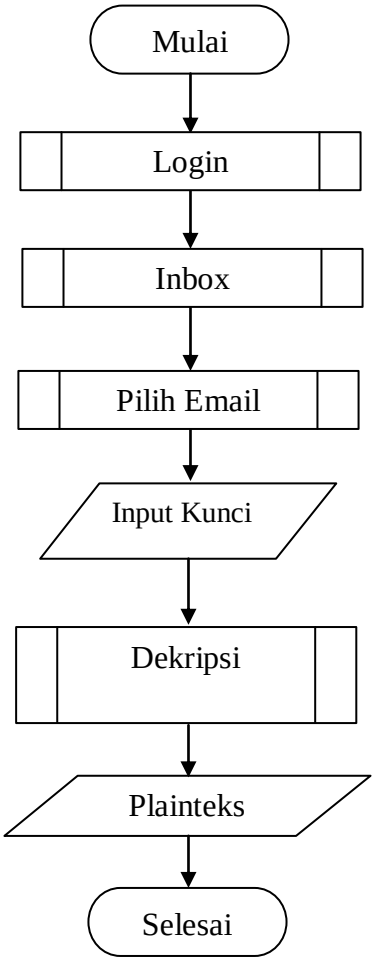
III.3.2. *Flowchart*

Flowchart adalah penggambaran secara grafik dari langkah-langkah dan urutan-urutan prosedur dari suatu program.

Tujuan utama dari penggunaan *flowchart* adalah untuk menggambarkan suatu tahapan penyelesaian masalah secara sederhana, terurai, rapi dan jelas dengan menggunakan simbol-simbol yang standar. Dalam perancangan aplikasi ini digunakan bagan alir (*flowchart*) untuk menjelaskan proses kerja dari perangkat lunak yang dirancang.



Gambar III.3. *Flowchart* Enkripsi Pesan *Email*



Gambar III.4. *Flowchart* Dekripsi Pesan *Email*

1. Proses *Flowchart* Enkripsi Pesan *Email*
    a. Mulai
    b. *Login* kedalam akun *email*
    c. Pilih *new email*
    d. Masukkan pesan dan kunci
    e. Enkripsi *email*
    f. Kirim *email* dengan pesan terenkripsi
2. Proses *Flowchart* Dekripsi Pesan *Email*
    a. Mulai

b. *Login* kedalam akun *email*

c. Pilih *inbox*

d. Pilih pesan yang terenkripsi

e. Masukkan kunci

f. Dekrip pesan *email*

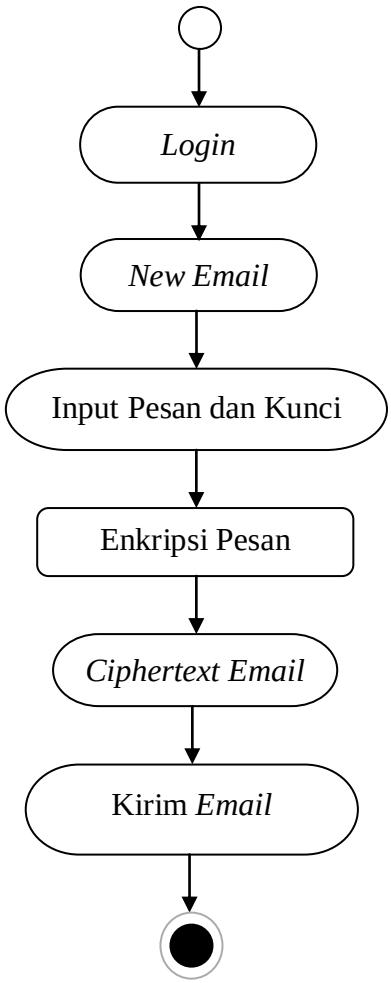
g. Selesai

III.3.3. *Activity Diagram*

Activity diagram menggambarkan berbagai alir aktivitas dalam sistem yang sedang dirancang, bagaimana masing-masing alir berawal, *decision* yang mungkin terjadi, dan bagaimana mereka berakhir. *Activity diagram* yang terdapat pada aplikasi yaitu sebagai berikut :

1. *Activity Diagram* Enkripsi Pesan *Email*

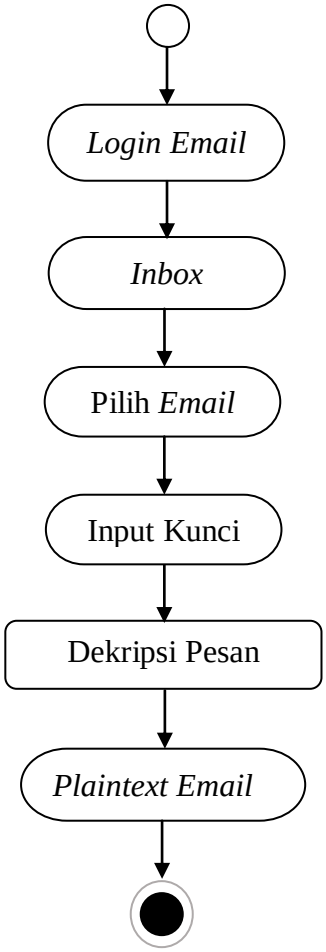
Activity diagram enkripsi pesan *email* menggambarkan alir aktifitas pengenkripsian pesan yang dilakukan oleh pengguna dan diproses didalam sistem.



Gambar III.5. *Activity Diagram* Enkripsi Pesan *Email*

2. *Activity Diagram* Dekripsi Pesan *Email*

Activity diagram dekripsi pesan *email* menggambarkan alir aktifitas pendeskripsian pesan *email* yang dilakukan oleh pengguna dan diproses didalam sistem.



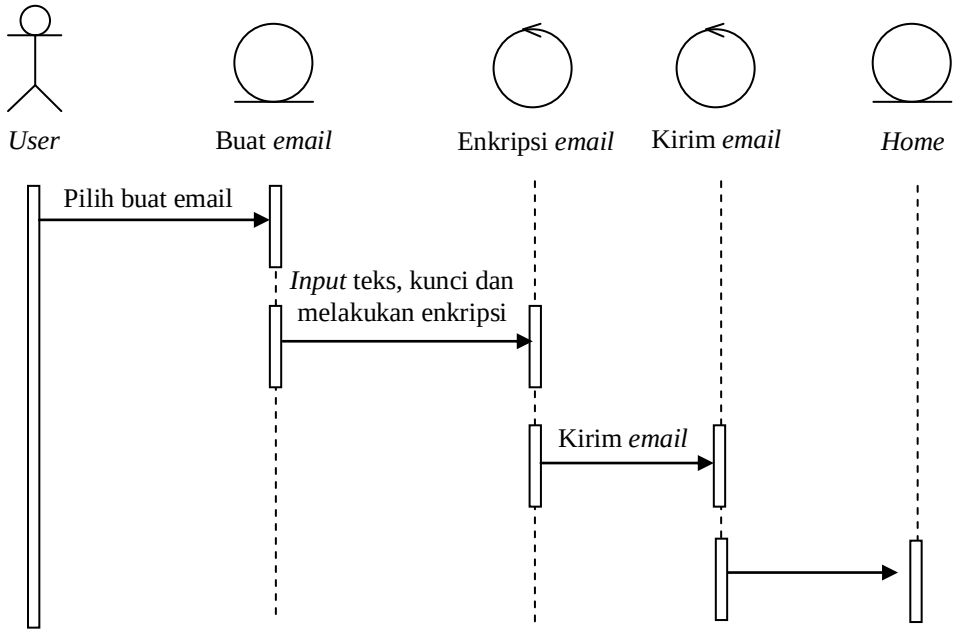
Gambar III.6. *Activity Diagram* Dekripsi Pesan *Email*

III.3.4. *Sequence Diagram*

Sequence diagram pada aplikasi yang akan dibuat yaitu : *sequence diagram* enkripsi dan *sequence diagram* dekripsi.

III.3.4.1. *Sequence Diagram* Enkripsi

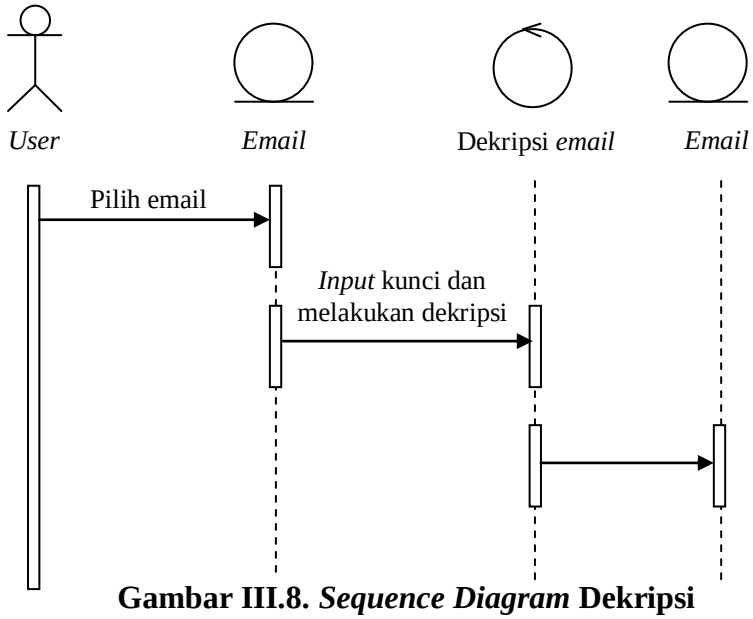
Sequence diagram enkripsi menggambarkan proses yang terjadi dalam melakukan enkripsi pesan teks pada *email*. *Sequence diagram* enkripsi ditunjukkan pada gambar III.7.



Gambar III.7. *Sequence Diagram* Enkripsi

III.3.4.2. *Sequence Diagram* Dekripsi

Sequence diagram dekripsi menggambarkan proses yang terjadi dalam melakukan dekripsi pesan teks pada *email*. *Sequence diagram* dekripsi ditunjukkan pada gambar III.8.

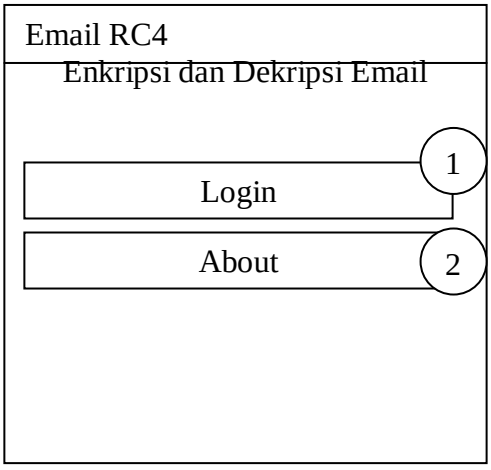


Gambar III.8. *Sequence Diagram* Dekripsi

III.4. Desain *User Interface*

Antarmuka pemakai (*user interface*) adalah tampilan program yang dapat dilihat, didengar atau dipersepsikan oleh pengguna dan perintah-perintah atau mekanisme yang digunakan pemakai untuk mengendalikan operasi dan memasukkan data. Berikut ini merupakan perancangan aplikasi kriptografi yang dirancang dengan antarmuka pada perangkat *mobile*, yaitu :

1. Desain *Form Menu* Utama

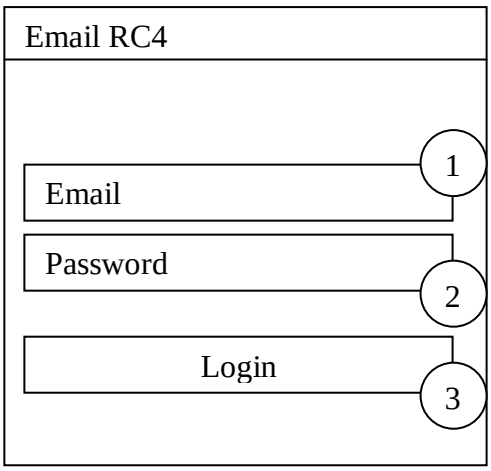


Gambar III.9. Desain *Form Menu* Utama

Keterangan tampilan *menu* utama, yaitu :

1) Tombol untuk melakukan *login email*.
2) Tombol untuk membuka halaman tentang *programer*.

2. Desain *Form Login*

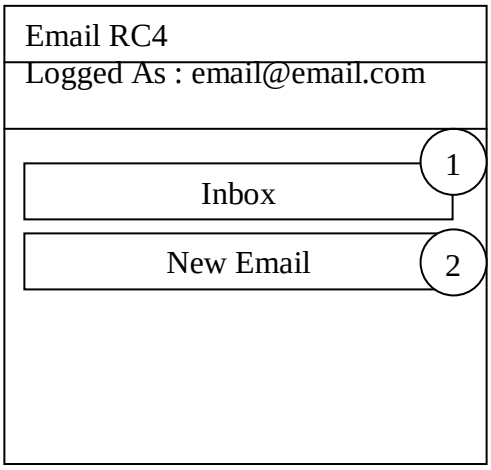


Gambar III.10. Desain *Form Login*

Merupakan tampilan rancangan form login. Adapun keterangannya sebagai berikut :

1) *Textbox* yang digunakan untuk memasukkan *username*
2) *Textbox* yang digunakan untuk memasukkan *password*.
3) Tombol yang digunakan untuk masuk ke akun *email*.

3. Desain *Form* Halaman Akun *Email*



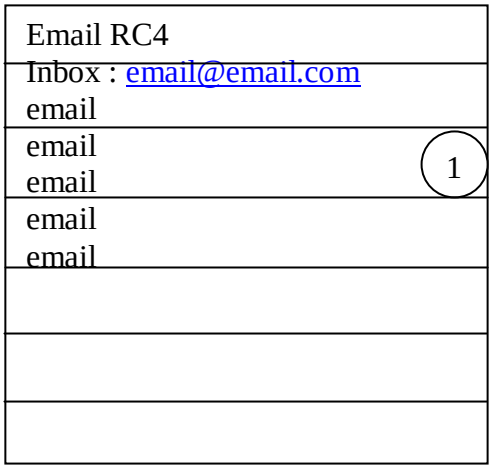
Gambar III.11. Desain *Form* Halaman Akun *Email*

Keterangan tampilan halaman akun *email,* yaitu :

1) Tombol untuk melihat *email*.

2) Tombol untuk membuat *email* baru.

4. Desain *Form* Halaman *Inbox*

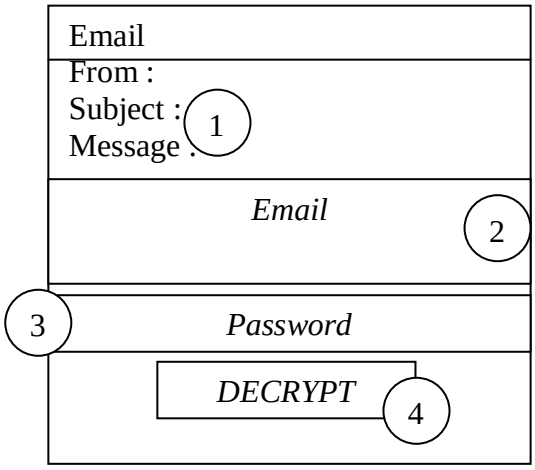


Gambar III.12. Desain *Form* Halaman *Inbox*

Keterangan tampilan halaman *inbox,* yaitu :

1) Daftar *email* masuk.

5. Desain *Form* Halaman *Detail Email*

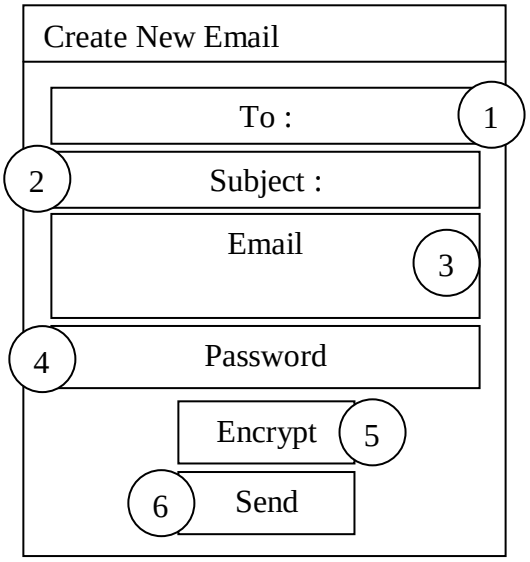


Gambar III.13. Desain *Form* Halaman Akun *Email*

Keterangan tampilan halaman akun *email,* yaitu :

1) Teks yang menampilkan *email* pengirim dan subjek *email*.
2) *Textbox* untuk menampilkan pesan *email*.
3) *Textbox* untuk menginputkan kata kunci.
4) Tombol untuk melakukan dekripsi *email*.

6. Desain *Form* Halaman *New Email*



Gambar III.14. Desain *Form* Halaman *New Email*

Keterangan tampilan halaman *new email,* yaitu :

1) *Textbox* untuk menginputkan alamat *email* penerima.
2) *Textbox* untuk menginputkan judul *email*.
3) *Textbox* untuk menginputkan pesan *email.*
4) *Textbox* untuk menginputkan kata kunci.
5) Tombol untuk melakukan enkripsi.
6) Tombol untuk mengirim *email*.

5. Desain *Form About*

Gambar III.15. Desain *Form About*

Adapun keterangannya sebagai berikut :

1) Menampilkan data dari *programer*.