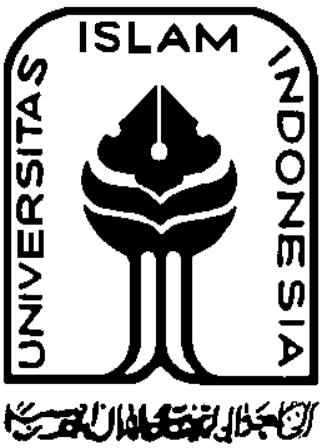
APLIKASI ENKRIPSI SMS (SHORT MESSAGE SERVICE) MENGGUNAKAN ALGORITMA DES

TUGAS AKHIR

Diajukan Sebagai Salah Satu Syarat untuk Memperoleh Gelar Sarjana

Teknik Informatika



oleh :

Nama : NANANG YUNENDAR

No. Mahasiswa : 05 523 335

JURUSAN TEKNIK INFORMATIKA

FAKULTAS TEKNOLOGI INDUSTRI

UNIVERSITAS ISLAM INDONESIA

YOGYAKARTA

2012

LEMBAR PERNYATAAN KEASLIAN

HASIL TUGAS AKHIR

Yang bertanda tangan dibawah ini,saya:

Nama : Nanang Yunendar

No. Mahasiswa : 05 523 335

Menyatakan bahwa Tugas Akhir dengan judul :

APLIKASI ENKRIPSI SMS (SHORT MESSAGE SERVICE) MENGGUNAKAN ALGORITMA DES

Yang diajukan untuk diuji pada tanggal 4 Juni 2012 adalah hasil karya saya.

Dengan ini saya menyatakan bahwa seluruh komponen dan isi didalam Laporan Tugas Akhir ini adalah hasil karya saya sendiri. Apabila dikemudian hari terbukti bahwa ada beberapa bagian dari karya ini adalah bukan karya saya sendiri, maka saya siap menanggung resiko dan konsekuensi apapun.

Demikian pernyataan saya ini saya buat, semoga dapat dipergunakan sebagaimana mestinya.

Yogjakarta, 24 Mei 2012

(Nanang Yunendar**)**

LEMBAR PENGESAHAN PEMBIMBING

APLIKASI ENKRIPSI SMS (SHORT MESSAGE SERVICE)

MENGGUNAKAN ALGORITMA DES

TUGAS AKHIR

oleh :

Nama : Nanang Yunendar

No. Mahasiswa : 05 523 335

Yogyakarta, 24 Mei 2012

Pembimbing

Zainudin Zukhri S.T.,M.I.T.

LEMBAR PENGESAHAN PENGUJI

APLIKASI ENKRIPSI SMS (SHORT MESSAGE SERVICE)

MENGGUNAKAN ALGORITMA DES

TUGAS AKHIR

oleh:

Nama : Nanang Yunendar

No. Mahasiswa : 05 523 335

Telah Dipertahankan di Depan Sidang Penguji Sebagai Salah Satu Syarat

Untuk Memperoleh Gelar Sarjana Jurusan Teknik Informatika

Fakultas Teknologi Industri Universitas Islam Indonesia

Yogyakarta, 4 Juni 2012

Tim Penguji

Zainudin Zukhri S.T.,M.I.T _____
Ketua

Syarif Hidayat, S.Kom., M.I.T _____
Anggota I

Ahmad Munasir Raf'ie Pratama ,S.T.,M.I.T _____
Anggota II

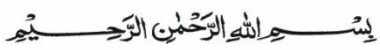
Mengetahui,

Ketua Jurusan Teknik Informatika Fakultas Teknologi Industri

Universitas Islam Indonesia

(Yudi Prayudi, S.Si., M.Kom.)

HALAMAN PERSEMBAHAN



Insya Allah dengan ikhlas kupersembahkan kepada-Nya

Semoga memberikan kebaikan di dunia dan di kehidupan sesudahnya

Maha Suci Allah yang telah menyempurnakan ikhtiar insan-Nya

KEPADA KEDUA ORANG TUAKU DAN ADIKKU YANG SANGAT AKU SAYANGI

AKU UCAPKAN MAAF DAN TERIMA KASIH YANG SEBESAR-BESARNYA ATAS KESABARANYA SLAMA INI MENDIDIK AKU DAN MEMBERIKAN AKU SEGALANYANYA SEHINGGA AKU BISA MENJADI SEPERTI INI

Dan seluruh pihak yang telah berjasa besar

Hanya ucapan terima kasih yang dapat terucap

MOTTO

“ Sesungguhnya sesudah kesulitan itu ada kemudahan. Maka apabila kamu telah selesai (dari suatu urusan), kerjakanlah dengan sungguh-sungguh (urusan) yang lain. Dan hanya kepada Tuhan-mulah hendaknya kamu berharap “

(Q.S. Al Insyirah : 6-8)

Orang-orang yang berhenti belajar akan menjadi pemilik masa lalu. Orang-orang yang masih terus belajar, akan menjadi pemilik masa depan

Sekotor dan seburuk apapun masa lalumu, masa depanmu itu suci.

Time is free but it’s priceless, u can’t own it but u can use it. U can’t keep it but u can spend it

KATA PENGANTAR

Assalamu'alaikum wa rahmatullaahi wa barakaatuh

Segala puji dan syukur hanyalah milik Allah Subhanahu Wa Ta'ala yang Maha Pengasih dan Maha Penyayang, yang telah menjadikan manusia beriman dan berilmu setelah sebelumnya berada dalam kondisi yang lemah dan diliputi kebodohan. Atas izin dan kehendak-Nya, tugas akhir berjudul "APLIKASI ENKRIPSI SMS (SHORT MESSAGE SERVICE) MENGGUNAKAN ALGORITMA DES " ini akhirnya dapat terselesaikan.

Tugas Akhir ini merupakan salah satu syarat untuk memperoleh gelar sarjana Teknik Informatika di Universitas Islam Indonesia.

Penulis dalam kesempatan ini mengucapkan terima kasih kepada :

1. Kedua Orang Tua
2. Adikku maaf dan terima kasih banyak atas dukungannya
3. Bapak Prof.Dr Edy Suandi Hamid, M.Ec, selaku Rektor Universitas Islam Indonesia
4. Bapak Ir. Gumbolo HS., M.Sc., selaku Dekan Fakultas Teknologi Industri
5. BapakYudi Prayudi, SSi., M. Kom, selaku Ketua Jurusan Teknik Informatika
6. Bapak, Zainudin Zukhri S.T.,M.I.T.selaku Dosen Pembimbing Tunggal
7. Seluruh staf pengajar dan dosen Teknik Informatika UII, atas perjuangan mereka dalam mengemban amanah untuk menyebarkan ilmu pengetahuan
8. Temen-temen kos yang telah banyak membantu dalam proses penyusunan Tugas Akhir ini

9. Pihak-pihak yang tidak dapat saya sebutkan satu persatu

Dalam penyelesaian Tugas Akhir ini penulis menyadari bahwa masih banyak terdapat kelemahan dan kekurangan. Oleh karena itu penulis mengharapkan kritik dan saran yang bersifat membangun agar pada masa mendatang menjadi lebih baik.

Akhir kata, Semoga Tugas Akhir ini dapat berguna bagi para penuntut ilmu, para praktisi, dan seluruh masyarakat IT untuk tujuan kemaslahatan dan kepentingan bersama.

Amin

Wassalamu'alaikum wa rahmatullaahi wa barakaatuh

Yogyakarta, 20 Mei 2012

Penyusun

SARI

Perkembangan teknologi komunikasi sekarang ini sangat pesat, dan salah satu dampaknya adalah semakin terjangkau alat komunikasi seperti *handphone* dan layanannya. Meskipun layanan telepon semakin murah tetapi sms tetap menjadi pilihan utama dalam melakukan komunikasi diantara banyak orang.

Di sisi lainnya orang mulai memperhatikan keamanan dan privasi untuk melakukan komunikasi lewat sms. Sehingga dibutuhkan sebuah aplikasi yang dapat menjaga keamanan dan privasi isi sms.

Java, sebuah platform sekaligus bahasa pemrograman yang mampu berjalan diberbagai sistem operasi menyediakan platform J2ME yang dapat dijalankan disebuah perangkat dengan sumberdaya terbatas seperti ponsel. J2ME inilah yang kemudian dikenal di kalangan pengguna telepon seluler lebih dikenal dengan sebutan "ponsel java", maksudnya adalah telepon seluler yang telah memilki dukungan untuk menjalankan aplikasi-aplikasi java didalamnya.

Aplikasi enkripsi sms dengan metode DES adalah aplikasi J2ME yang menerapkan metode enkripsi DES yang memanfaatkan library bouncycastle untuk menajaga isi sms. Dengan menggunakan java sebagai basis pengembangan aplikasi, aplikasi enkripsi sms diharapkan akan mampu berjalan dengan baik pada semua perangkat telepon seluler berteknologi java. Selain itu pada aplikasi ini juga dilengkapi dengan kompresi untuk mengurangi pembengkakan besarnya data hasil enkripsi. Metode Kompresi yang digunakan adalah metode Huffman

Kata kunci : J2ME SMS, Bouncycastle, DES Encryption,Huffman

TAKARIR

plain text	teks awal atau dasar
library	kumpulan kelas fungsi
paper	makalah atau jurnal
prefix	awalan
bottom-up	bawah ke atas
syntax	aturan penulisan kode
robust and secure	kuat dan aman
use case	perilaku
sequence	kronologis
handling	penanganan
requirement	permintaan
confidentiality	kerahasiaan
nonrepudiation	tidak bisa ditolak
integrity	keutuhan
key	kunci
userfriendly	mudah dipahami
layer	lapisan
web defacing	serangan merubah tampilan web
exception	keadaan tidak normal
buffer overflow	keadaan program menerima input berlebihan
state	keadaan atau status

DAFTAR ISI

DAFTAR TABEL

DAFTAR GAMBAR

BAB I
PENDAHULUAN

1.1 Latar Belakang

Jaringan selular adalah sebuah komponen yang sangat penting dalam perekonomian dan kehidupan sosial saat ini. Selain layanan suara, pesan teks merupakan layanan yang sangat sering digunakan oleh pelanggan. Hal ini terungkap dari penelitian yang dilakukan Nielsen Mobile. Lembaga survei ini mengungkapkan bahwa rata-rata pengguna ponsel di Amerika berkirim 357 SMS (Short Message Service) setiap bulan, dan hanya menerima atau melakukan panggilan telepon sebanyak 204 kali per bulan.

Secara umum SMS (Short Message Service) tidak menjamin kerahasiaan dan keutuhan pesan yang dikirimkan oleh pengguna. Oleh karena pesan-pesan teks yang dikirim pengguna terkadang merupakan pesan yang rahasia dan pribadi, sehingga kerahasiaan pesan menjadi sangat penting untuk dijaga dari orang-orang yang tidak berhak mendapatkannya. SMS (Short Message Service) didisain untuk komunikasi dimana content yang dikirimkan adalah *plain text*. Bagaimanapun data *plain text* seperti ini dapat dicegat di jalan oleh siapa saja yang memiliki akses ke sistem SMS (Short Message Service). Server SMS (Short Message Service) milik operator merupakan salah satu pihak yang dapat mengambil data ini, walaupun dalam setiap perjanjian terdapat klausul tentang kerahasiaan data, akan tetapi data *plain text* yang terkirim dan berkasnya tersimpan di server milik operator dan SMS (Short Message Service) tanpa penyandian adalah satu potensi bahaya yang besar.

1.2 Rumusan Masalah

Permasalahan yang menjadi obyek penelitian tugas akhir ini adalah bagaimana membangun sebuah aplikasi untuk mengenkripsi SMS (Short Message Service) untuk tujuan *privacy* (pribadi) dan *security* (keamanan) kemudian

mengkompresi hasil enkripsi SMS (Short Message Service) agar ukuran menjadi lebih kecil

1.3 Batasan Masalah

Pemberian batasan masalah ini bertujuan agar terjadi penyederhanaan masalah, dan sebagai kontrol agar tidak terjadi penyimpangan dari apa yang diharapkan dalam penelitian ini. Batasan–batasan tersebut adalah:

1. Aplikasi digunakan untuk mengenkripsi SMS (Short Message Service) dan mengkompres hasil enkripsi
2. Aplikasi hanya bisa digunakan pada handphone yang mempunyai teknologi java (J2ME), baik pengirim maupun penerima
3. Algoritma DES berasal dari *library* Bouncy castle
4. Proses pengembangan dan analisis kinerja aplikasi menggunakan emulator Default Color Phone dari J2ME Wireless Toolkit (WTK)
5. Masalah gangguan koneksi atau jaringan pada operator tidak diperhitungkan dalam aplikasi ini

1.4 Tujuan Penelitian

Tujuan dari pembuatan tugas akhir ini adalah membangun sebuah aplikasi untuk mengenkripsi SMS (Short Message Service) dengan algoritma DES dan mengkompres hasil enkripsi dengan metode Huffman

1.5 Manfaat Penelitian

Semua penelitian yang dilakukan pada hakekatnya selalu diharapkan mempunyai manfaat, adapun manfaat penelitian ini adalah:

1. Membuat komunikasi lewat SMS (Short Message Service) lebih aman
2. Menjamin privasi atau kerahasiaan isi SMS (Short Message Service)

1.6 Metodologi Penelitian

Metodologi yang digunakan dalam melakukan tugas akhir ini terdiri dari metode pengumpulan data dan metode perancangan sistem

1. Metode Pengumpulan Data

   Metode yang digunakan dalam melakukan pengumpulan data adalah melalui studi pustaka, yaitu dengan mengumpulkan data-data melalui buku-buku dan situs internet yang berhubungan dengan permasalahan.

2. Metode Perancangan Sistem

   a) Analisis kebutuhan perangkat lunak

      Pada analisis kebutukan sistem ini peneliti melakukan observasi terhadap data-data yang diperlukan berdasarkan sumber- sumber yang berkaitan dengan penelitian, baik literatur, dokumentasi, atau catatan-catatan yang berkaitan penelitian.

   b) Perancangan

      Dalam hal ini peneliti menentukan perancangan proses, perancangan input dan output serta antarmuka (*interface*). Perancangan sistem ini dilakukan sesuai dengan sumber-sumber yang ada kaitannya dengan data-data yang diperlukan.

   c) Implementasi sistem

      Metode yang digunakan pada implementasi sistem ini adalah praktek langsung pada sistem yang telah dibuat dengan melakukan pengujian. Selain itu, metode yang digunakan adalah pengamatan terhadap sistem yang dibuat, apakah perlu perbaikan atau tidak, sesuai dengan kebutuhan yang diperlukan atau tidak.

   d) Analisis hasil

      Analisis hasil yang diperoleh dari implementasi yang telah disempurnakan dan kekurangannya serta apakah sudah layak untuk digunakan.

1.7 Sistematika Penulisan

Laporan Tugas Akhir ini disusun secara sistematis dalam bentuk bab, sebagai berikut:

BAB 1 PENDAHULUAN

Menjelaskan tentang latar belakang penelitian, mengapa topik tentang enkripsi SMS (Short Message Service) dengan algoritma DES ini menjadi pilihan, tujuan penelitian, batasan penelitian dan metode penelitian yang digunakan.

BAB II LANDASAN TEORI

Menjelaskan tentang landasan teori yang digunakan seperti penjelasan mengenai Java, J2ME, Algoritma DES, Metode Huffman dan hal-hal lain yang diperlukan.

BAB III ANALISIS KEBUTUHAN

Bab ini berisi mengenai analisa sistem enkripsi SMS (Short Message Service), dari hasil analisa tersebut dapat diketahui lebih jelas mengenai masalah yang dihadapi pada sistem yang sedang berjalan.

BAB IV PERANCANGAN

Bab ini menguraikan tentang perancangan sistem yang terdiri metode perancangan dan hasil perancangan, yang meliputi perancangan diagram sistem dan perancangan antarmuka (*interface*) program enkripsi SMS (Short Message Service).

BAB V IMPLEMENTASI

Bab ini membahas tentang implementasi aplikasi enkripsi SMS (Short Message Service) sesuai dengan analisis kebutuhan dan desain berdasarkan hasil perancangan sistem.

BAB VI ANALISIS KINERJA

Bab ini membuat dokumentasi hasil pengkajian terhadap perangkat lunak enkripsi SMS (Short Message Service) yang dibandingkan dengan kebenaran dan kesesuiannya serta bagaimana kelebihan dan kekurangan tersebut dapat digunakan penelititan lebih lanjut.

BAB VII KESIMPULAN DAN SARAN

Bab ini membuat kesimpulan-kesimpulan dalam proses pengembangan perangkat lunak enkripsi SMS (Short Message Service), baik pada analisis kebutuhan, perancang perangkat lunak, implementasi, dan analisis kinerja. Juga berisi saran yang perlu diperhatikan berdasar analisis kerja program, keterbatasan-keterbatasan yang ditemukan dan asumsi-asumsi yang dibuat selama pelaksanaan TA.

BAB II
LANDASAN TEORI

Aplikasi ini menggunakan enkripsi DES dengan memanfaatkan library dari bouncy castle dan kompresi dengan metode huffman,serta dibangun dengan J2ME yang merupakan teknologi java khusus untuk perangkat mobile,sehingga sifat-sifat dan tata cara algoritma java juga berlaku di J2ME dengan tambahan beberapa kemampuan khusus.

2.1 Kriptografi Algoritma DES

2.1.1 Kriptografi

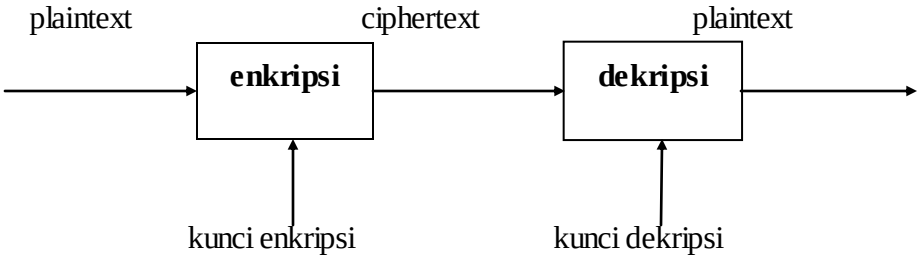
Kriptografi berasal dari bahasa yunani, menurut bahasa dibagi menjadi dua *kripto* dan *graphia*, kripto berarti *secret* (rahasia) dan graphia berarti *writing* (tulisan). Menurut terminologinya kriptografi adalah ilmu dan seni untu menjaga keamanan pesan ketika pesan dikirim dari suatu tempat ketempat yang lain[ARI06].

Tujuan sistem kriptografi adalah sebagai berikut [SIM06] :

1. *Confidentiality*, yaitu memberikan kerahasiaan pesan dan menyimpan data dengan menyembunyikan informasi lewat teknik-teknik enkripsi
2. *Message Integrity*, yaitu memberikan jaminan bahwa setiap bagian tidak akan mengalami perubahan dari saat data itu dibuat atau dikirim sampai dengan saat data tersebut dibuka
3. *Nonrepudiation*, yaitu memberikan cara untuk membuktikan bahwa suatu dokumen datang dari seseorang apabila ia mencoba menyangkal memiliki dokumen tersebut.
4. *Aunthentication*, yang memberikan dua layanan. Yang pertama mengidentifikasi keaslian suatu pesan dan memberkan jaminan keotentikannya. Kedua, untuk menguji identitas seseorang apabila ia akan memasuki sebuah system

Para pelaku atau praktisi kriptografi disebut *cryptographers*. Sebuah algoritma kriptografik (*cryptographic algorithm*), disebut *cipher*, merupakan persamaan matematik yang digunakan untuk proses enkripsi dan dekripsi.

Proses yang dilakukan untuk mengamankan sebuah pesan (yang disebut *plaintext*) menjadi pesan yang tersembunyi (disebut *ciphertext*) adalah enkripsi (*encryption*). *Ciphertext* adalah pesan yang sudah tidak dapat dibaca dengan mudah. Menurut ISO 7498-2, terminologi yang lebih tepat digunakan adalah "*encipher*". Proses sebaliknya, untuk mengubah *ciphertext* menjadi *plaintext*, disebut dekripsi (*decryption*). Menurut ISO 7498-2, terminologi yang lebih tepat untuk proses ini adalah "*decipher*". Cryptanalysis adalah seni dan ilmu untuk memecahkan ciphertext tanpa bantuan kunci. Cryptanalyst adalah pelaku atau praktisi yang menjalankan cryptanalysis. Cryptology merupakan gabungan dari cryptography dan cryptanalysis. [RAH02]



Gambar 2.1 *Diagram proses enkripsi dan dekripsi*

Dalam sistem komputer, pesan terbuka (*plaintext*) diberi lambang M, yang merupakan singkatan dari *Message*. *Plaintext* ini dapat berupa tulisan, foto, atau video yang berbentuk data biner. *Plaintext* inilah yang nantinya akan dienkripsi menjadi pesan rahasia atau *ciphertext* yang dilambangkan dengan C (*Ciphertext*). Secara matematis, fungsi enkripsi ini dinotasikan dengan :

$$E(M) = C$$

Sedangkan fungsi dekripsi adalah proses pembalikan dari *ciphertext* menjadi *plaintext* kembali. Secara matematis dinotasikan sebagai berikut :

$$D(C) = M$$

$$D(E(M)) = M$$

2.1.2 Algoritma DES (Data Encryption Standart)

Algoritma DES dikembangkan di IBM dibawah kepemimpinan W.L. Tuchman pada tahun 1972. Algoritma ini didasarkan pada algoritma *LUCIFER* yang dibuat oleh Horst Feistel. Algoritma ini telah disetujui oleh National Bureau of Standard (NBS) setelah penilaian kekuatannya oleh National Security Agency (NSA) Amerika Serikat.Sampai pertengahan tahun 1974 tidak ada satupun algoritma sandi yang diusulkan. Hingga akhirnya pada tanggal 6 Agustus 1974, algoritma sandi yang didesain oleh IBM yang bernama sistem sandi *Lucifer* ditawarkan kepada NBS. Kemudian setelah dilakukan evaluasi dan modifikasi dengan bantuan *National Security Agency* (NSA), pada tanggal 15 Juli 1977 NBS menetapkan algaritma *Lucifer* yang telah dimodifikasi tersebut dengan nama baru *Data Encryption Standard* atau lebih populer dengan sebutan sistem sandi DES.

DES termasuk ke dalam sistem kriptografi simetri dan tergolong jenis *cipher* blok.DES beroperasi pada ukuran blok 64 bit. DES mengenkripsikan 64 bit plainteks menjadi 64 bit cipherteks dengan menggunakan 56 bit kunci internal (*internal key*). Kunci internal dibangkitkan dari kunci eksternal (*external key*) yang panjangnya 64 bit.

Skema global dari algoritma DES adalah sebagai berikut:

1. Blok plainteks dipermutasi dengan matriks permutasi awal (*initial permutation* atau IP).
2. Hasil permutasi awal kemudian di-*enciphering*- sebanyak 16 kali (16 putaran). Setiap putaran menggunakan kunci internal yang berbeda.
3. Hasil *enciphering* kemudian dipermutasi dengan matriks permutasi balikan (*invers initial permutation* atau IP^{-1}) menjadi blok cipherteks.

2.2 Teknik Kompresi Huffman

2.2.1 Kompresi Data

Istilah kompresi tersebut diterjemahkan dari kata bahasa inggris "*compression*" yang berarti pemampatan. Secara teknik, kompresi berarti proses memampatkan sesuatu yang berukuran besar sehingga menjadi kecil. Dengan

demikian kompresi data berarti proses untuk memampatkan data agar ukuran menjadi lebih kecil[KRI03].

2.2.2 Algoritma Huffman

Algoritma *Huffman* dikembangkan oleh David A. Huffman pada tahun 1951. Pada awalnya David A. Huffman menulis *paper* sebagai salah satu tugas kuliahnya di MIT. Algoritma ini merupakan pengembangan lebih lanjut dari algoritma kompresi yang dilakukan oleh Claude Shannon dan R.M. Ran

Algoritma *Huffman* disebut juga algoritma *prefix*, dimana setiap kode yang dihasilkan tidak akan menghasilkan kode yang sama dari karakter yang muncul. Algoritma ini berdasarkan frekuensi kemunculan tiap karakter, yang akan menghasilkan kode-kode baru yang dihasilkan dari pembentukan pohon *Huffman*. Pembentukan pohon *Huffman* dilakukan secara *bottom-up*, yakni berdasarkan frekuensi kemunculan karakter yang terkecil ke yang terbesar.

Jika ditijau dari segi teknik pengkodean karakter yang digunakan, algoritma *Huffman* ini termasuk dalam algoritma yang menggunakan metode *symbolwise*, yaitu suatu metode yang menghitung probabilitas kemunculan suatu karakter dalam satu waktu, karakter yang sering muncul akan dikodekan dalam suatu untaian bit yang lebih pendek dan karakter yang jarang muncul akan dikodekan dalam untaian bit yang lebih panjang. Hal ini dimaksudkan untuk mempercepat proses inisialisasi karakter yang lebih sering muncul dalam sebuah string, yang tentu saja akan berpengaruh saat *encoding* dan *decoding* kode Huffman.

Pada setiap model kompresi, dilakukan proses konversi simbol dari representasi *string* menjadi representasi kode lain dan sebaliknya. Ada dua alat konversi dalam hal ini, yaitu *encoder* dan *decoder*. *Encoder* merupakan alat untuk menjalankan suatu mekanisme untuk melakukan konversi simbol dari representasi *string* menjadi suatu representasi kode lainnya yang bersifat unik. Mekanisme yang dilakukan oleh *encoder* disebut *encoding*. Sedangkan *decoder* merupakan alat yang menjalankan suatu mekanisme untuk mengembalikan representasi unik

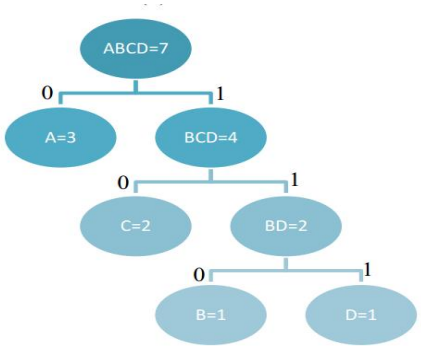
dari suatu *string* menjadi sebuah representasi *string* awalnya. Mekanisme yang dilakukan *encoder* adalah *decoding*.

a. Encoding

Pembentukan pohon *Huffman* adalah untuk membuat *prefix code*, yaitu kode berupa *string* biner dimana kode yang satu bukan merupakan awalan dari kode yang lain, sehingga pembentukan kode biner setelah dikompresi tidak menghasilkan kode yang sama. Secara garis besar proses kompresi dimulai dengan membuat pohon *Huffman*, setelah itu mengubah setiap karakter menjadi kode biner dan menyimpan kembali menjadi karakter yang lain karena ada pengerutan jumlah bit. Yang perlu diperhatikan adalah penyimpanan pohon *Huffman*, jika terjadi kesalahan maka *file* kompresi tidak akan terbaca. Berikut adalah Algoritma *Huffman*:

1. Baca string yang akan di*encoding*
2. Hitung frekuensi dari setiap karakter yang ada.
3. Urutkan dari frekuensi kemunculan yang terkecil hingga yang terbesar.
4. Buat pohon *Huffman* dengan cara :
    a. Setiap karakter dinyatakan sebagai pohon bersimpul tunggal dan setiap karakter disertai dengan jumlah frekuensi kemunculannya.
    b. Gabungkan dua pohon dengan frekuensi terkecil pada sebuah akar.
    c. Setelah digabungkan akar tersebut akan mempunyai frekuensi yang merupakan jumlah frekuensi dari dua pohon penyusun tersebut.
    d. Ulangi hingga hanya tersisa satu buah pohon *Huffman*. Dan urutkan berdasarkan frekuensi dari yang terkecil ke yang terbesar.
    e. Jika frekuensi kedua pohon sama, maka beri simbol 0 untuk sebelah kiri dan simbol 1 untuk sebelah kanan. Dan jika frekuensi kedua pohon berbeda, maka beri simbol 0 untuk frekuensi yang lebih kecil dan 1 untuk frekuensi yang lebih besar.
5. Setelah pohon *Huffman* selesai. Urutkanlah biner yang terbentuk dari akar hingga ke karakter tunggal.

6. Urutan biner tersebut adalah kode *Huffman* yang menggantikan kode *ASCII* untuk mempresentasikan karakter tersebut.



Gambar 2.2 Gambar Pohon Huffman

Contoh teks "ABACCDA", B dan D karena muncul paling jarang maka ditaruh paling bawah , kemudian C dan setelah itu huruf A, seperti pada gambar 2.2. Setelah itu dibuat tabel seperti tabel 2.1, jadi bitnya berubah menjadi 0 110 0 10 10 111 0 yang berjumlah 13 bit, yang awalnya 49 bit.

Tabel 2.1 Tabel Huffman

Karakter	String Biner Huffman
A	0
B	110
C	10
D	111

b. Decoding

Decoding merupakan proses mengembalikan suatu data dari suatu kode tertentu. Proses *Decoding* ini merupakan proses kebalikan dari proses *encoding*. Terdapat dua cara yang cukup cepat untuk melakukan *decoding* simbol, yaitu:

1. Membaca dari pohon *Huffman*

   Hal ini dapat dilakukan dengan cara membaca sebuah bit dari kode binernya dan menelusuri hingga sampai pada simpul daun yang mengandung simbol tersebut untuk setiap bitnya. Ketika suatu bit sampai pada daun suatu pohon, suatu simbol yang terkandung dalam daun tersebut ditulis untuk *decoded* data tersebut dan mengulanginya kembali dari akar pohon tersebut.

2. Menggunakan Tabel Kode Huffman

   Decoding cara ini dilakukan dengan menyimpan setiap kode pada suatu tabel yang terurut berdasarkan panjang kode dan mencari kesamaan dari setiap bit yang dibaca.

2.3 Java

Bahasa pemrograman java dibuat pada tahun 1991 oleh James Gosling, seorang ahli pemrograman yang bekerja di Sun Microsystem. Awalnya bahasa pemrograman ini diberi nama Oak. Yang mendapat inspirasi dari sebuah pohon yang berada di seberang kantornya. Akan tetapi nama Oak sendiri merupakan nama bahasa pemrograman yang telah ada sebelumnya, maka kemudian Sun mengganti namanya menjadi java. Nama java sendiri diinspirasi saat Gosling dan rekannya sedang menikmati secangkir kopi di sebuah kedai kopi yang kemudian dengan tidak sengaja salah satu dari mereka menyebutkan kata java yang mengandung arti kopi. Akhirnya mereka sepakat untuk memberikan nama bahasa pemrograman tersebut dengan java.

Java merupakan teknologi yang cukup fenomenal, dan mampu memberikan model perkembangan perangkat lunak yang inovatif. Bahasan mengenai teknologi Java akan selalu melibatkan dua bagian, yaitu Java sebagai bahasa pemrograman (*programming language*) dan Java sebagai lingkungan dijalankannya aplikasi (*platform*) [WIC02]

2.3.1 Java Programming Language

Sebagai bahasa pemrograman, Java memiliki beberapa ciri khas yang menjadi keunggulan dibandingkan dengan bahasa pemrograman lain: [WIC02]

1. *Simple*. Java dirancang untuk mudah dipelajari,dengan *syntax* yang sangat mirip dengan bahasa pemrograman C/C++ yang sudah sangat popular
2. *Object –oriented*. Java merupakan bahasa pemrograman yang sangat konsisten dalam mengimplementasikan konsep OOP (*Object Oriented Programming*). Hal ini banyak memberikan keuntungan kepada para pengembang yang sudah mulai mengubah paradigmanya kearah konsep OOP.

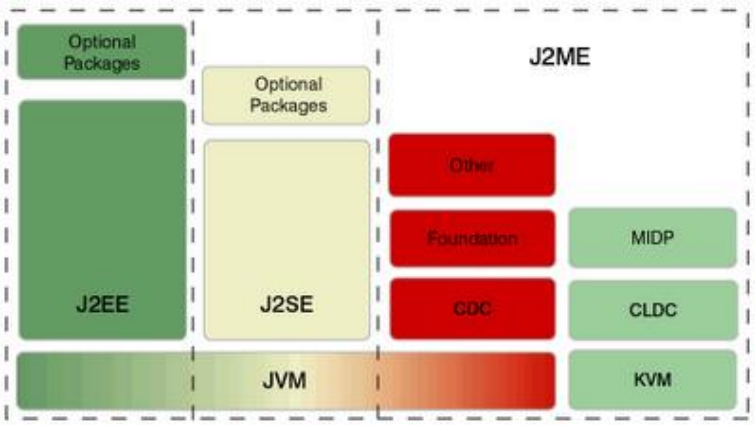
3. *Robust and secure*. Java menawarkan penggunaan sumber daya system operasi dengan lebih aman,karena dijalankan di atas Java platform,bukan diatas sistem operasi secara langsung. Selain itu Java juga menghilangkan operasi pointer yang sudah dikenal luas dalam bahasa C/C++.Sudah tak terhitung banyaknya insiden keamanan seperti *web defacing* bahkan *host compromise* yang diakibatkan oleh kelemahan pemrograman yang dikenal dengan nama *buffer overflow*, yang disebabkan oleh kecerobohan dalam melakukan operasi pointer. Hal semacam ini tidak akan terjadi di aplikasi Java karena operasi pointer bahkan telah dihilangkan.
4. *Netral architecture and portable*. Dengan semboyan *write once run anywhere*, Java dirancang untuk dapat dijalankan disemua platform, tanpa peduli apakah arsitekturnya berbasis Intel,AMD,Sun SPARC atau PowerPC, aplikasi Java tetap mampu berjalan di berbagai platform tanpa perlu dikompilasi ulang
5. *High performance*. Java dirancang untuk menghasilkan aplikasi-aplikasi dengan performa yang terbaik. Contoh nyata aplikasi besar saat ini yang mungkin bias dilihat aplikasi database Oracle 8i/9i yang *core*-nya dibangun menggunakan bahasa pemrograman Java

2.3.2 Java Platform

Pada Java 2, Sun Microsystems mengeluarkan tiga buah edisi yang memiliki platform yang berbeda-beda, yaitu:

1. Java 2 Standard Edition (J2SE)
   digunakan untuk mengembangkan aplikasi-aplikasi *desktop* dan *aplet* (aplikasi Java yang dapat dijalankan di dalam browser web).
2. Java 2 Enterprise Edition (J2EE)
   merupakan bagian dari J2SE, digunakan untuk mengembangkan aplikasi-aplikasi berskala besar (*enterprise*), seperti aplikasi server, aplikasi web, dan teknologi lain seperti CORBA dan XML.
3. Java 2 Micro Edition (J2ME)
   merupakan bagian dari J2SE, digunakan untuk mengembangkan aplikasi-aplikasi di dalam perangkat-perangkat kecil, dengan memori terbatas, display

terbatas dan pemrosesan yang terbatas, yang tidak memungkinkan untuk mendukung implementasi J2SE secara penuh.



Gambar 2.3 Platform Java 2: J2SE, J2EE dan J2ME
(Sumber: http://www.8mobile.org/j2me01.aspx)

Sebagai sebuah platform,Java terdiri atas dua bagian utama, yaitu *Java Virtual Machine* (JVM) dan *Java Application Programming Interface* (Java API)[WIC02]

a. Java Virtual Machine

JVM adalah sebuah mesin *virtual* yang bekerja selayaknya sebuah mesin. JVM memiliki spesifikasi hardwarenya sendiri beserta platform yang dibutuhkan untuk melakukan kompilasi terhadap *source code* java. Setelah dikompilasi akan menghasilkan *bytecode* yang disebut berkestensi *.class* yang isinya menyerupai kode mesin. Kode mesin sendiri di terjemahkan oleh mesin dimana dia dijalankan, sedangkan *bytecode* Java diterjemahkan oleh JVM juga. Untuk menjalankan program Java komputer memerlukan JVM dan JVM dapat bekerja di platform apa saja. Selama memiliki JVM program Java dapat dijalankan tanpa memperhatikan platform. Sebab itulah Java memiliki sifat platform *independent*

b. Java Application Programming Interface (Java API)

Java API adalah sekumpulan pustaka dasar yang disediakan oleh vendor Java yang menjadi pondasi utama untuk melakukan pengembangan perangkat lunak berbasis teknologi Java. Pada setiap versi dan rilis, biasanya terdapat

perbedaan pustaka serta adanya penyempurnaan implementasi Java API.Perkembangan Java API termasuk dipengaruhi oleh permintaan dari pengembang aplikasi yang menggunakan Java, kemudian juga adanya tuntutan perkembangan teknologi computer yang ada

2.4 J2ME

J2ME dipublikasikan pada bulan Juni 1999 pada Konferensi Pengembang JavaOne (*JavaOne Developer Conference*). J2ME membawa fungsi–fungsi *cross platform* Java ke dalam sebuah perangkat yang lebih kecil yang memungkinkan perangkat tersebut dijalankan dimana saja secara mudah.

J2ME menggunakan konfigurasi dan profil khusus sehingga aplikasi yang dibangun dapat berjalan pada perangkat dengan *memory* terbatas. Konfigurasi berisi seperangkat kelas–kelas inti yang dapat berjalan pada *device* tertentu. Sementara profil, berisi fungsi–fungsi tambahan konfigurasi yang dapat digunakan aplikasi.

2.4.1 J2ME Configuration

Untuk konfigurasi J2ME terdapat 2 (dua) kategori, yaitu:

a. CLDC (*Connected Limited Device Configuration*)

   kategori ini umumnya digunakan untuk aplikasi Java pada handphone semacam Nokia, Samsung Java Phone, Motorola i85s, PDA. Umumnya perangkat–perangkat ini memiliki ukuran memory sekitar 160–512 Kilobytes, memiliki koneksi jaringan yang lambat, dan power yang terbatas.

b. CDC (*Connected Device Configuration*)

   Kategori ini umumnya digunakan untuk aplikasi Java pada perangkat dengan ukuran memory minimal 2 Megabytes. Contohnya adalah pada Internet TV, Nokia Communicator, TV pada mobil. Pada dasarnya, aplikasi yang dibangun menggunakan CLDC, akan berjalan juga pada konfigurasi ini.

2.4.2 J2ME Profile

J2ME Configuration menyediakan library Java untuk implementasi fitur–fitur standar dari sebuah perangkat mobile. Profil J2ME (*J2ME Profile*) menyediakan implementasi tambahan yang lebih spesifik terhadap sebuah perangkat mobile. Beberapa kategori profil J2ME antara lain:

a. Mobile Information Device Profile (MIDP)
b. Foundation Profile (FP)
c. Personal Profile
d. RMI Profile
e. Personal Digital Assistance Profile

Dari sekian banyak profil yang tersedia, MIDP merupakan profil yang paling sering digunakan dalam pengembangan perangkat lunak untuk perangkat mini. Hal ini dikarenakan MIDP menyediakan library Java untuk implementasi dasar antarmuka (GUI), implementasi jaringan, database, dan timer. Selain itu, saat ini semakin banyak vendor mengadopsi profil ini

2.4.3 Midlet

Midlet adalah aplikasi Java yang dijalankan pada lingkungan MIDP yang merupakan sebuah kelas dalam paket *java.microedition.midlet*. Setiap aplikasi MIDP yang dibuat harus merupakan penurunan (*extends* dari kelas Midlet)

Midlet berubah *state* dari satu *state* ke *state* lain dalam satu siklus hidup, yaitu[HAR03]:

1. **Start –** mengambil sumber daya dan memulai eksekusi
2. **Pause –** melepaskan sumber daya dan menjadi menunggu
3. **Destroy –** melepaskan semua sumber daya, menghancurkan thread, dan mengakhiri semua aktivitas

Paket Midlet terdiri dari [HAR03]:

1. Midlet dipaketkan di satu file JAR yang berisi:
   a) File-file kelas yang termasuk Midlet
   b) File-file sumber daya
   c) *Manifest* dari properti kelas

2. Deskriptor aplikasi dan menyediakan informasi termasuk
   a) Properti-properti konfigurasi
   b) Properti-properti sebelum download seperti ukuran, versi, kebutuhan penyimpanan

2.4.4 WMA (Wireless Messaging API)

Secara garis besar *Wireless Messaging System* dapat mengembangkan kemampuan *networking* dan kemampuan I/O dari aplikasi J2ME untuk mengirim dan menerima pesan menggunakan *messaging services* pada jaringan GSM, seperti Short Message Service (SMS) dan Cell Broadcast Service (CBS).

Untuk mengirim SMS kita memanfaatkan fungsi MessageConnection untuk membuka koneksi ke alamat yang dituju melalui port 1234. Setelah menentukan alamat dengan fungsi setAddress dan data yang dikirimkan melalui fungsi setPayloadData maka dikirimkanlah SMS tersebut.

```
MessageConnection conn = null;
String url = "sms://+6281675670:1234";
try {
     conn=(MessageConnection)Connector.open(url );
     BinaryMessage msg =
     conn.newMessage( conn.BINARY_MESSAGE );
     msg.setAddress(url);
     msg.setPayloadData( data );
     conn.send( msg );
```

Untuk menerima SMS kita kembali memanfaatkan fungsi MessageConnection untuk membuka koneksi terhadap port 1234. Setelah itu menerima SMS yang berbentuk biner.

```
MessageConnection conn = null;
 String a = "sms://:1234";
 try {
      conn = (MessageConnection)Connector.open(a);
      conn.setMessageListener(this);
```

```
BinaryMessage pesanBinary =
(BinaryMessage)conn.receive();
…
```

2.4.5 RMS (Record Management System)

Penyimpanan data pada MIDlet menggunakan memori *non-volatile* (memori tetap) yang disebut dengan *Record Management System* (RMS). Didalam RMS terdapat kumpulan *record*, dan *record* disimpan sebagai *array* dari byte dalam sebuah *record store*. RMS memiliki orientasi *record* basis data yang sederhana sehingga MIDlet dapat menyimpan informasi dan mengaksesnya. MIDlet yang berbeda dapat mengakses RMS yang sama. *Record* yang disimpan pada *record store* dapat diakses berdasarkan recordId yang berupa integer. Oleh karena itu *recordId* berperan sebagai *key*.

Metode-metode yang digunakan untuk mengakses RMS:

1. Untuk Membuka *record store*

```
RecordStore rs =
RecordStore.openRecordStore("dbRms",true);
//Untuk parameter true di, jika dbRms tidak ada maka akan dibuat.
```

2. Menutup *record store*

```
rs.closeRecordStore();
RecordStore.deleteRecordStore(dbRms);
```

3. Menghapus *record store*

```
RecordStore.deleteRecordStore(dbRms);
```

4. Menambah *record*

```
String data = "insert record";
byte bytes[] = data.getBytes();
rs.addRecord(bytes,0,bytes.length);
```

5. *Update record* pada recordId 1

```
String newaData = "update record";
Byte data = newaData.getBytes();
rs.setRecord(1, data, 0, data.length());
```

6. Menghapus *record* pada record Id 1

```
rs.deleteRecord(1);
```

7. *Enumerating record*

```
RecordEnumeration re =
rs.enumerateRecords(null, null, false);
If (re.hasNextElement()) Byte nextRec[] =
re.nextRecord();
```

2.5 Bouncy Castle

Bouncy castle adalah sekumpulan API (*Application Programming Interface*) yang berhubungan dengan kriptografi untuk bahasa pemrograman Java dan C. Bouncy castle pada J2ME bekerja pada API CLDC dan CDC. API mendukung banyak enkripsi seperti DES, Blowfish, IDEA, Rijindael, RC4 dan lain-lain.

Tabel 2.1 Paket-paket dalam Bouncy Castle

Jar/zip	Keterangan
jce-jdkNN-MMM.jar	Untuk Java Cryptography Extension (JCE) library Paket ini mendukung generasi sertifikasi dan This package support the generation of certificates dan penanganan permintaan PKCS10
bctsp-jdkNN-MMM.jar	Bouncy Castle TSP (Time Stamp Protocol) library
bctest-jdkNN-MMM.jar	Bouncy Castle untuk test classes library
bcprov-jdkNN-MMM.jar	Bouncy Castle untuk provider library
bcpg-jdkNN-MMM.jar	Bouncy Castle untuk OpenPGP/BCPG library paket untuk obyek OpenPGP.
bcmail-jdkNN-MMM.jar	Bouncy Castle untuk SMIME/CMS library paket untuk meproses obyek RFC 3852 Cryptographic Message Syntax (CMS) objects juga direferensikan untuk PKCS#7 (formerly RFC 2630, 3369), and juga untukS/MIME objects (RFC 3851).
cldc_classes.zip	library untuk J2ME
cldc_crypto.zip	library untuk J2ME

Catatan: NN = versi J2SE , MMM=versi rilis Bouncy Castle

BAB III
ANALISIS KEBUTUHAN PERANGKAT LUNAK

3.1 Metode Analisis

Analisis kebutuhan dilakukan untuk mengetahui semua permasalahan serta kebutuhan yang diperlukan untuk mengembangkan aplikasi. Analisis dilakukan dengan mencari dan menentukan permasalahan yang dihadapi serta semua kebutuhan seperti masukan dan keluaran sistem, antarmuka sistem, dan fungsi-fungsi yang dibutuhkan.

3.2 Hasil Analisis

Berdasarkan analisis yang telah dilakukan maka dapat diketahui apa saja yang menjadi masukan sistem, keluaran sistem, antarmuka yang diinginkan dan fungsi atau metode yang digunakan oleh sistem, sehingga sistem yang dibuat nantinya sesuai dengan yang diharapkan.

3.2.1 Masukan Sistem (input requirement analysis)

Kebutuhan masukan adalah suatu bentuk masukan dan berupa data yang telah ada yang dibutuhkan oleh perangkat lunak sehingga dapat mencapai tujuan yang dinginkan. Masukan (*input*) dari Enkripsi SMS adalah teks SMS yang akan dikompres kemudian dikirim ke HP (*Handphone)* penerima.

3.2.2 Keluaran Sistem (output requirement analysis)

Keluaran (*autput*) sistem disini adalah adanya SMS yang sudah dikompres dan dienkripsi yang akan diterima oleh HP sang penerima SMS. Setelah itu hasil dari dekompres dan dekripsi dari SMS menjadi teks asli yang dapat dibaca oleh HP penerima SMS.

3.2.3 Antarmuka Yang Diinginkan

Antarmuka yang dikembangkan pada perangkat lunak (software) ini diusahakan *userfriendly* dan sesederhana mungkin karena berjalan di layar HP yang mempunyai keterbatasan dengan harapan akan memudahkan pengguna dalam menggunakannya. Antarmuka berbentuk *form* menu yang terdiri dari kotak masuk, kotak keluar, dan terdapat menu tulis pesan dengan isian nomor yang akan dituju kemudian, isi pesan yang akan dikirim, dan *password* pesan. Setelah itu ada pilihan mengirim pesan.

Untuk *form* penerimaan pesan ada *form* pemberitahuan bahwa ada pesan masuk kemudian ada kolom untuk pengisian *password* setelah itu menuju *form* pesan yang berhasil didekripsi dan dekompres sehingga terbuka pesan yang dapat dibaca.

3.2.4 Fungsi-fungsi Yang Dibutuhkan

Perangkat yang akan dibangun nantinya diusahakan untuk menangani fitur-fitur standar pengiriman dan penerimaan SMS yang sudah dienkripsi dan dikompres. Setelah dilakukan analisis, ada beberapa hal utama yang harus dapat dilakukan oleh aplikasi SMS Enkripsi ini:

a. Mengirim pesan (SMS) yang sudah dikompres dengan metode Huffman dan dienkripsi dengan metode DES ke aplikasi yang dituju Mengenkripsi pesan dengan metode DES
b. Menerima pesan (SMS) dari aplikasi pengirim
c. Mendekripsi dan mendekompres pesan yang diterima dari aplikasi pengirim
d. Aplikasi dapat menampilkan pesan asli sehingga dapat dibaca oleh pengguna.

3.2.5 Kinerja Yang Diharapkan

Kinerja yang diharapkan dari hasil analisis diatas adalah perangkat lunak yang dibangun mampu mengkompres dan mengenkripsi SMS masukan kemudian mengirmkan SMS tersebut,dan disisi penerima dapat mengembalikan SMS yang terenkripsi dan terkompres menjadi SMS awal sehingga dapat dibaca oleh penerima.

BAB IV
PERANCANGAN

4.1 Metode Perancangan

Aplikasi yang dibuat merupakan aplikasi yang dibangun atas teknologi Java yang menerapkan paradigma berorientasi obyek. Oleh karena itu penggunaan UML (*Unified Modelling Language*) untuk melakukan perancangan merupakan pilihan yang tepat dan lebih sesuai karena permodelan yang dihasilkan dapat diorganisasi kedalam kelas-kelas (*class diagram*) yang berhubungan dengan masalah sesungguhnya sehingga lebih mudah untuk dipahami dan dikembangkan.

4.2 Hasil Perancangan

4.2.1 UML (Unified Modelling Language)

UML merupakan bahasa pemodelan yang dapat digunakan untuk berbagai tujuan, yang menggunakan standar notasi tertentu. UML umumnya menjadi standar dalam perancangan pembuatan aplikasi yang menerapkan pemrograman berorientasi objek. Untuk lebih menjelaskan perancangan aplikasi yang dibangun, digunakan 3 (tiga) model diagram, yaitu : *use case diagram*, *class diagram*, dan *sequence diagram*

a. Use Case Diagram

Use case diagram menggambarkan fungsionalitas yang diharapkan pada sebuah system dengan penekanan pada apa yang dilakukan oleh sebuah system. Dalam *use case diagram*, ada dua hal yang saling berhubungan, yaitu aktor dan *use case*.

Pada rancangan yang dibuat, hanya terdapat 1 (satu) aktor saja,yaitu aktor *user*. Kemudian untuk actor *user* tersebut memiliki beberapa use case yang didefinisikan sebagai berikut:

a. Menulis dan Mengirim Pesan

b. Membuka Kontak

c. Membaca SMS pada kotak masuk

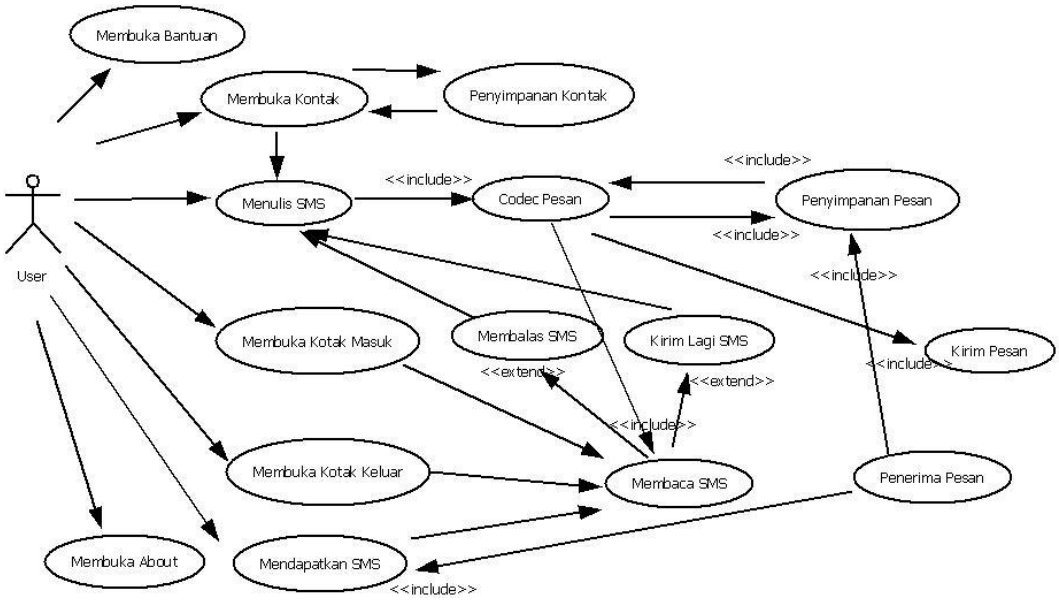
d. Membaca SMS pada kotak keluar

e. Membalas SMS

f. Mengirim Kembali SMS

g. Menerima SMS yang masuk

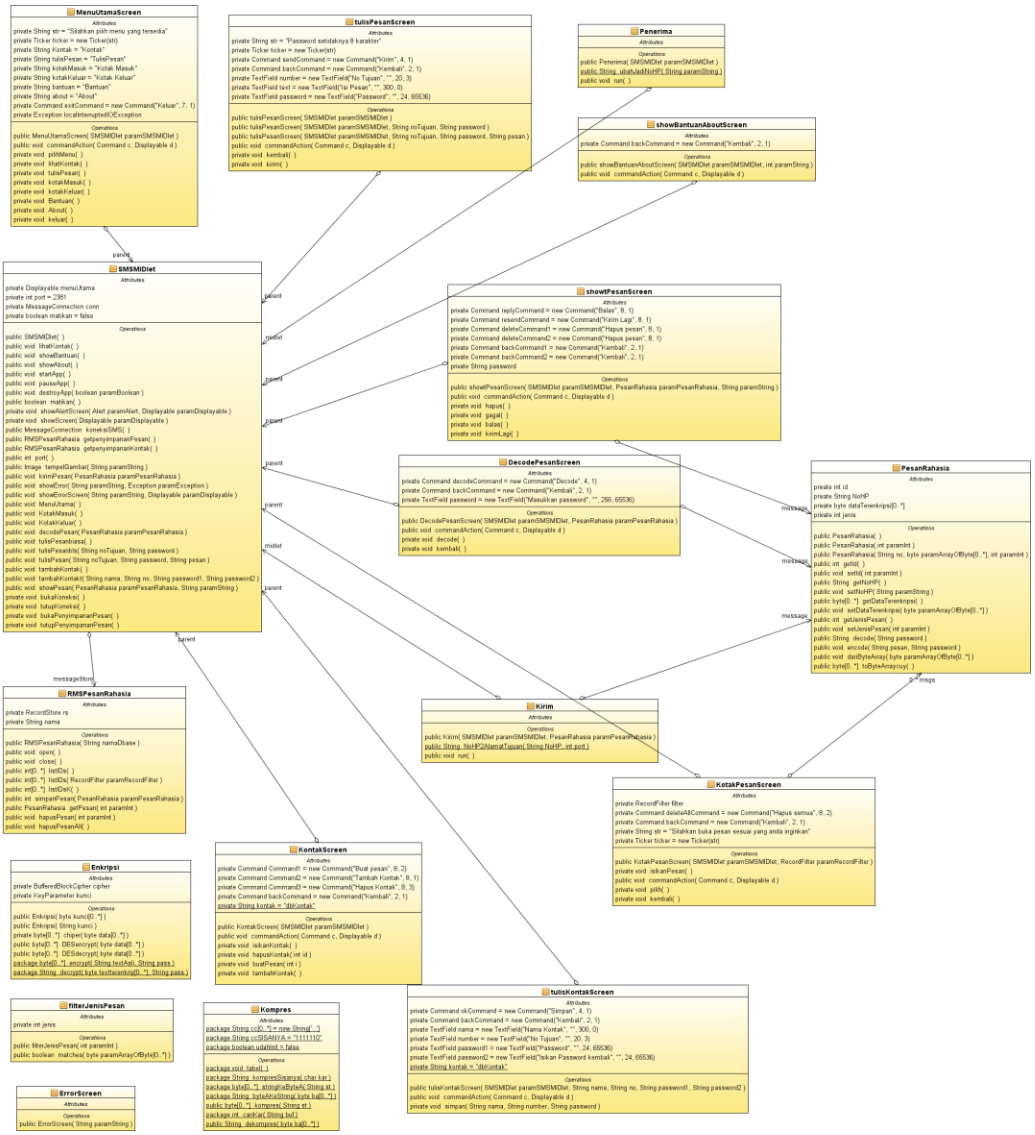
Definisi fungsi di atas akan lebih jelas terlihat pada gambar berikut



Gambar 4.1 Use Case Diagram

b. Class Diagram

Aplikasi yang menggunakan pernacangan berorientasi objek dapat diilustrasikan dalam struktur kelas-kelas dan hubungan antar kelas yang ada. Dalam UML digunakan notasi class diagram untuk menggambarkan hal tersebut.Lebih jelasnya dapat dilihat pada gambar 4.2.



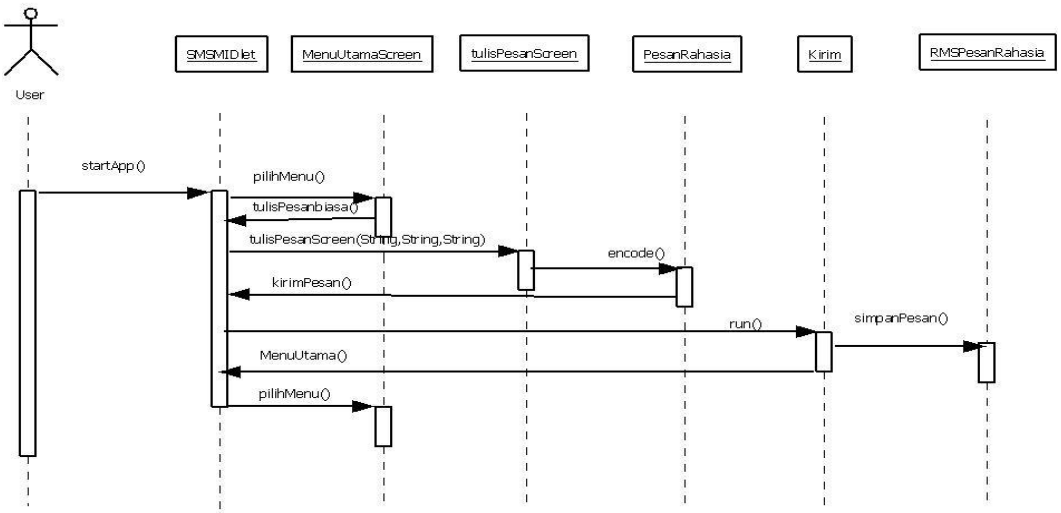
Gambar 4.2 Class Diagram

c. Sequence Diagram

Sequence diagram menunjukkan urutan waktu dan proses yang terjadi pada sebuah fungsi sistem sehingga dapat terlihat pertukaran data yang terjadi diantaranya.Pada Aplikasi ini terdapat beberapa diagram *sequence* antara lain:

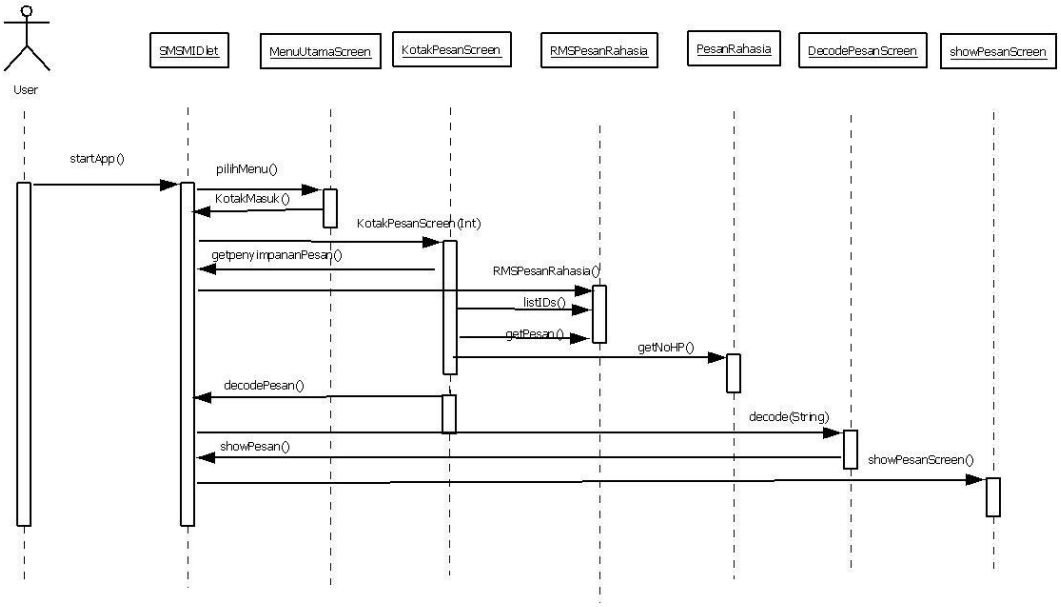
a. *Sequence* menulis SMS Rahasia
b. *Sequence* membuka SMS Rahasia
c. *Sequence* membalas SMS Rahasia
d. *Sequence* mengirim kembali SMS Rahasia
e. *Sequence* menghapus SMS Rahasia
f. *Sequence* membuka Bantuan atau About
g. *Sequence* menerima SMS Rahasia
h. *Sequence* menambah kontak
i. Sequence melihat daftar kontak

Sequence menulis SMS Rahasia merupakan skenario menulis SMS dan mengirimkan SMS Rahasia.Lebih jelasnya dapat dilihat pada gambar 4.3



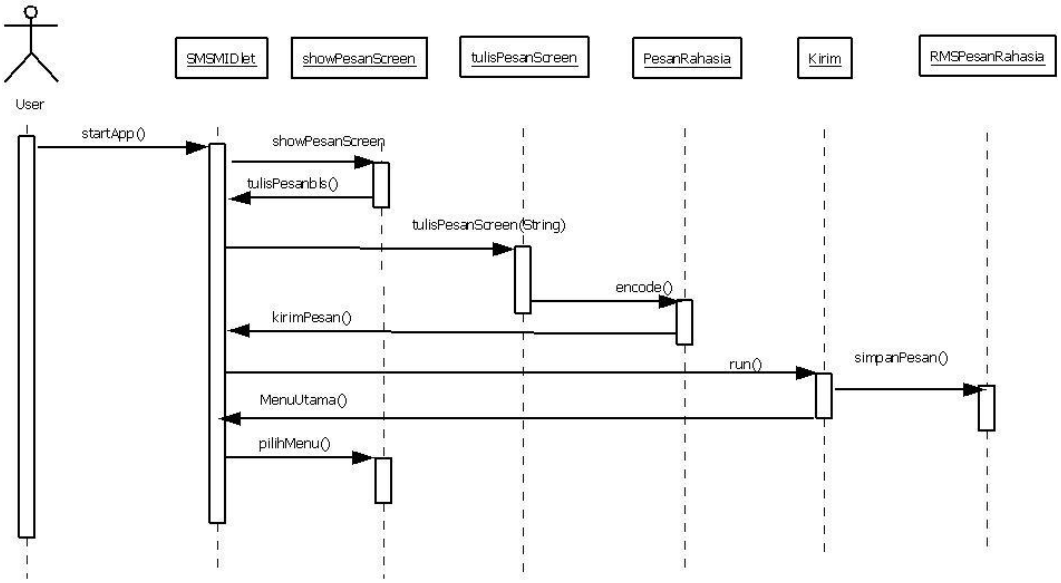
Gambar 4.3 Sequence Diagram Menulis SMS Rahasia

Sequence membuka SMS Rahasia merupakan skenario membuka SMS baik di Kotak Masuk maupun Kotak Keluar. Perbedaan diagram sequence pada kotak masuk dan kotak keluar hanya pada parameter *recordfilter* pada KotakPesanScreen yang beripe data integer. Lebih jelasnya dapat dilihat pada gambar 4.4



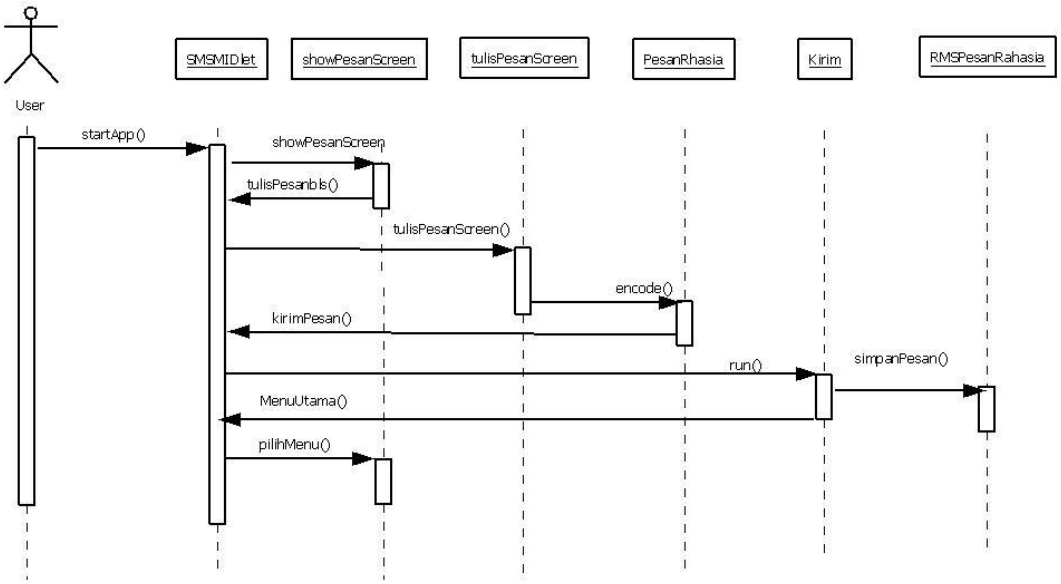
Gambar 4.4 Sequence Diagram Membuka SMS Rahasia

Sequence membalas SMS Rahasia merupakan skenario membalas SMS Rahasia setelah membaca sebuah SMS Rahasia.Sequence membalas SMS Rahasia ini membutuhkan masukan berupa String yang akan menjadi isi SMS balasan. Lebih jelasnya dapat dilihat pada gambar 4.5



Gambar 4.5 Sequence Diagram Membalas SMS Rahasia

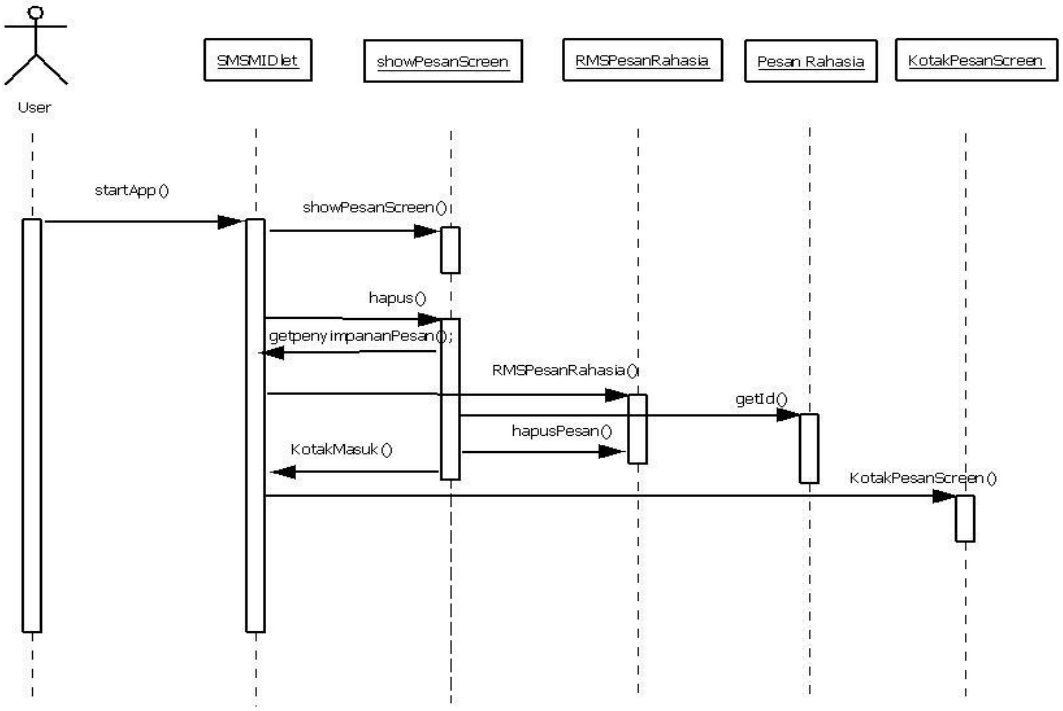
Sequence mengirim kembali SMS Rahasia ini merupakan skenario jika pengguna ingin mengirim lagi sebuah SMS Rahasia lagi karena sesuatu hal misalnya *error* pada jaringan atau yang lainnya. Lebih jelasnya dapat dilihat pada gambar 4.6



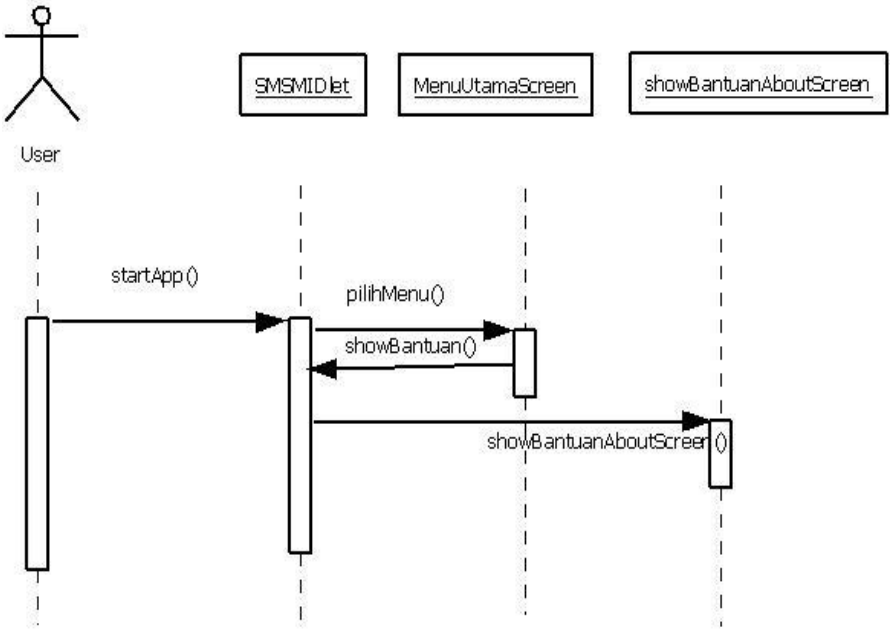
Gambar 4.6 Sequence Diagram Mengirim Kembali SMS Rahasia

Sequence menghapus SMS Rahasia ini merupakan skenario jika pengguna ingin menghapus sebuah SMS Rahasia. Lebih jelasnya dapat dilihat pada gambar 4.7

Sequence membuka bantuan ini merupakan sekenario ketika pengguna ingin mendapatkan sebuah informasi tentang cara pakai dan keterangan-keterangan mengenai aplikasi SMS Rahasia ini . Sequence About hampir sama dengan sequence about yang berbeda hanya informasi yang ditampilkan.Lebih jelasnya dapat dilihat pada gambar 4.8

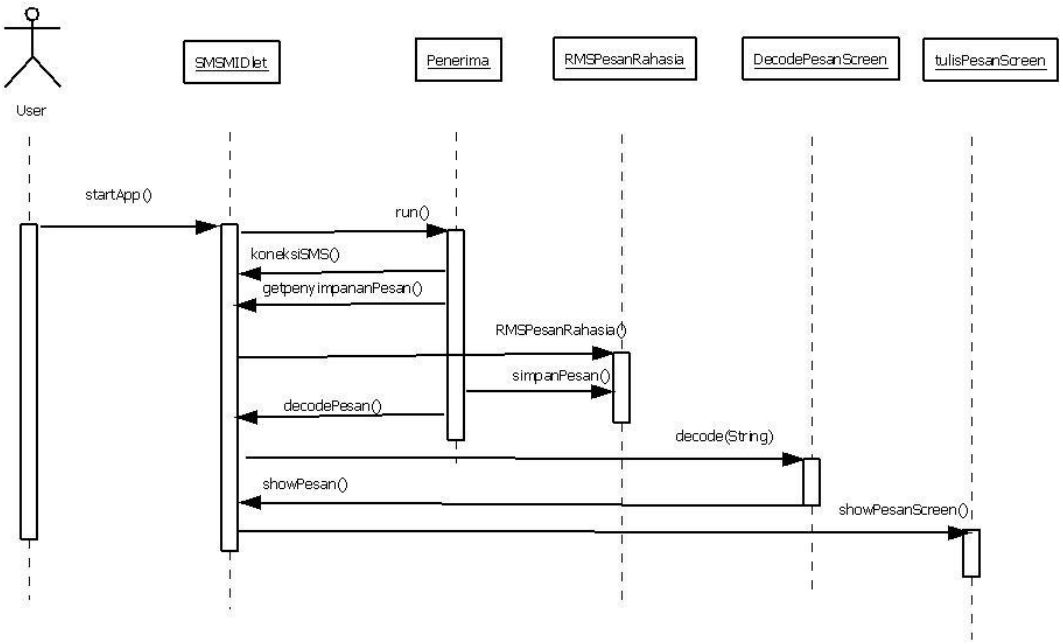


Gambar 4.7 Sequence Diagram menghapus SMS Rahasia



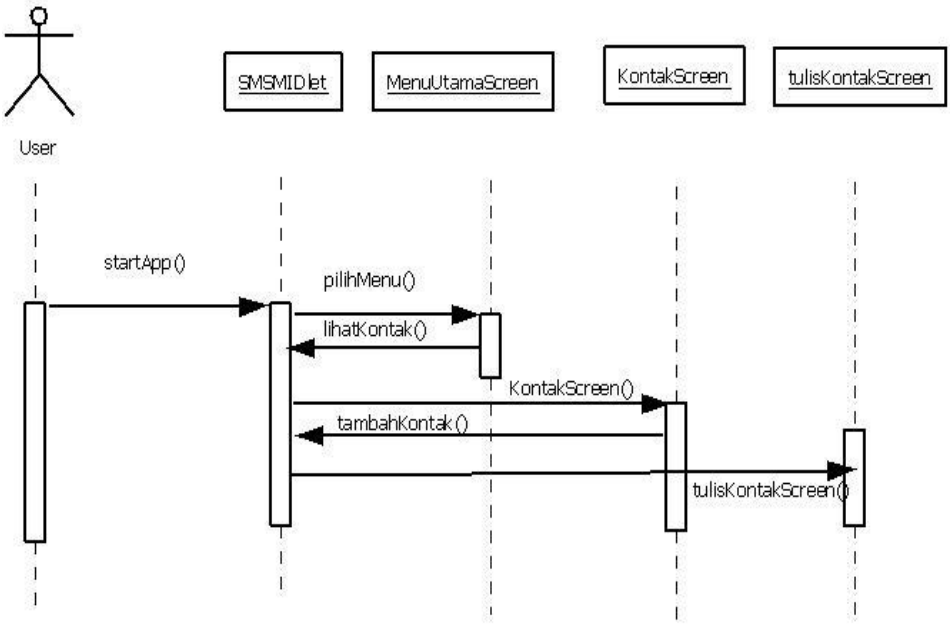
Gambar 4.8 Sequence Diagram Membuka Bantuan

Sequence menerima SMS Rahasia merupakan sekenario ketika pengguna mendapatkan sebuah SMS Rahasia. Lebih jelasnya dapat dilihat pada gambar 4.9



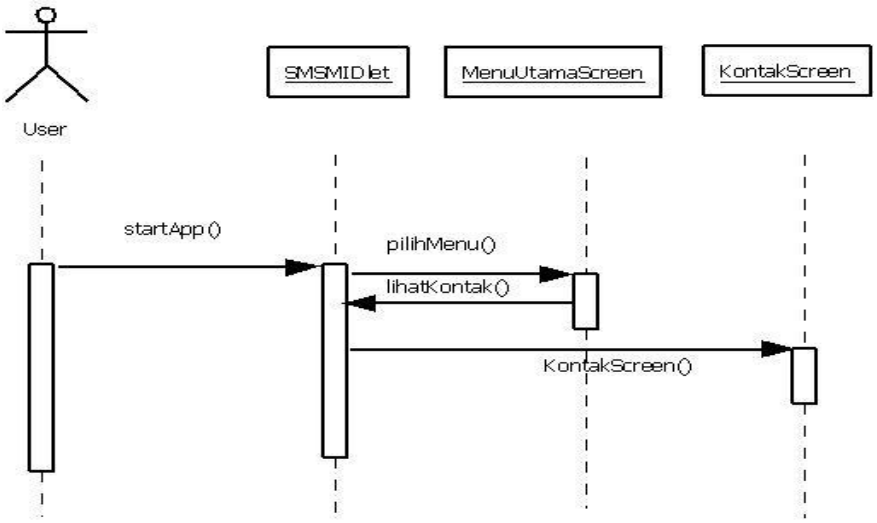
Gambar 4.9 Sequence Diagram Menerima SMS Rahasia

Sequence menambah kontak merupakan sekenario ketika pengguna ingin menambah kontak. Lebih jelasnya dapat dilihat pada gambar 4.10



Gambar 4.10 Sequence Diagram Menambah Kontak

Sequence menambah kontak merupakan sekenario ketika pengguna ingin melihat daftar kontak. Lebih jelasnya dapat dilihat pada gambar 4.11



Gambar 4.11 Sequence Diagram Melihat Daftar Kontak

4.2.2 Rancangan Database

Aplikasi ini dibuat membutuhkan dua *database* untuk menyimpan SMS Rahasia baik yang masuk (di kotak masuk) maupun keluar (di kotak keluar) dan sebuah *database* untuk menyimpan data kontak. *Database* yang digunakan untuk aplikasi ini adalah *database Record Management System* (RMS). RMS merupakan fitur khusus yang telah diadopsi secara langsung oleh profil MIDP. *Database* ini dirancang untuk aplikasi yang akan digunakan pada perangkat dengan *memory* terbatas.

Pada RMS, hanya terdapat 2 (dua) kolom data. Kolom pertama merupakan indeks penyimpanan data, dan kolom kedua merupakan data yang tersimpan. Data yang dapat disimpan dalam RMS hanyalah sebuah array byte. Aplikasi ini menggunakan satu *database* untuk menampung SMS Rahasia baik yang sudah diterima ataupun diirim. Tabel 4.1 berikut ini akan lebih menjelaskan struktur penyimpanan pesan pada *database* RMS yang akan dibangun.

Tabel 4.1 Rancangan Database Pesan

RecordID	Data
1	arrayOfByte
2	arrayOfByte
3	arrayOfByte
...	...

Data arrayOfByte ini apabila diurai akan terdiri dari:

1. NoHP, (No tujuan penerima SMS), String
2. JenisPesan (untuk membedakan SMS Rahasia dari kotak masuk atau kotak keluar), Integer
3. dataTerenkripsi (Isi SMS yang sudah dikompres dan dienkripsi), byte[]

Selain *database* untuk SMS juga dibutuhkan *database* untuk kontak. Kontak dibutuhkan untuk mempermudah pengguna untuk mengirim pesan, sehingga tidak perlu memsaukkan password. *Database* ini menyimpan data nama,no tujuan, dan *password*. Untuk lebih jelasnya dapat dilihat pada tabel 4.2.

Tabel 4.2 Rancangan Database Kontak

RecordID	Data
1	arrayOfByteKontak
2	arrayOfByteKontak
3	arrayOfByteKontak
...	...

Data arrayOfByteKontak ini apabila diurai akan terdiri dari:

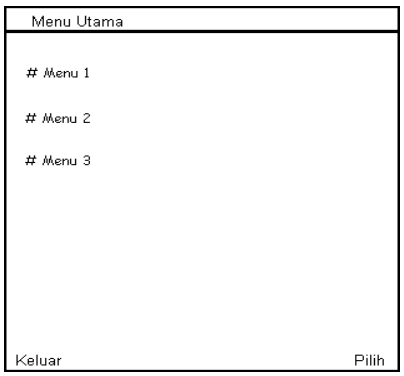
1. Nama , (Nama orang yang dituju), String
2. No tujuan, String (No tujuan penerima), String
3. *Password*, (Password SMS), String
4.

4.2.3 Rancangan Antarmuka

Antarmuka yang akan dibangun mempertimbangkan faktor kemampuan perangkat yang memiliki keterbatasan memori, maka antarmuka dirancang cukup sederhana tetapi harus memenuhi fungsi dasar aplikasi ini yaitu mengirim dan menerima SMS Rahasia, menyimpan SMS baik yang masuk dan keluar serta fitur kontak yang digunakan untuk memudahkan pengguna dalam mengirim SMS.

a. Menu Utama

Pada antarmuka ini disediakan menu-menu pilihan yang dapat dipilih oleh pengguna.Untuk lebih jelasnya dapat dilihat pada gambar 4.12



Gambar 4.12 Rancangan Antarmuka Menu Utama

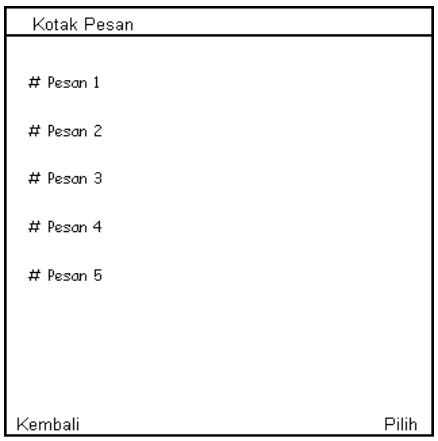
b. Menu Tulis SMS Rahasia

Untuk membuat SMS Rahasia dibutuhkan beberapa masukan yaitu No Tujuan,Isi SMS yang akan dikirim,dan *password*.

Gambar 4.13 Rancangan Antarmuka Tulis SMS Rahasia

c. Antarmuka Kotak Pesan

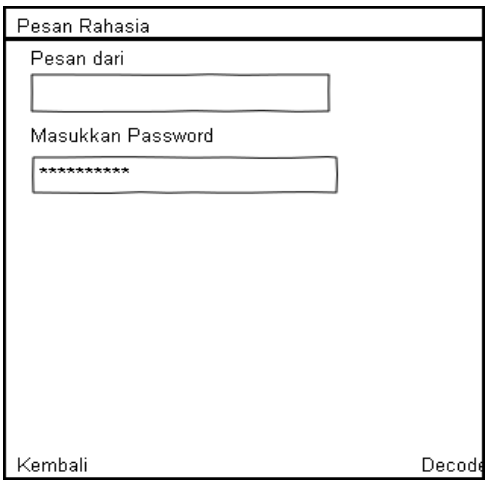
Antarmuka ini untuk menampilkan daftar SMS baik di kotak masuk atau kotak keluar. Untuk lebih jelasnya dapat dilihat pada gambar 4.14



Gambar 4.14 Rancangan Antarmuka Kotak Pesan

d. Antarmuka Membuka Pesan (Decode Pesan)

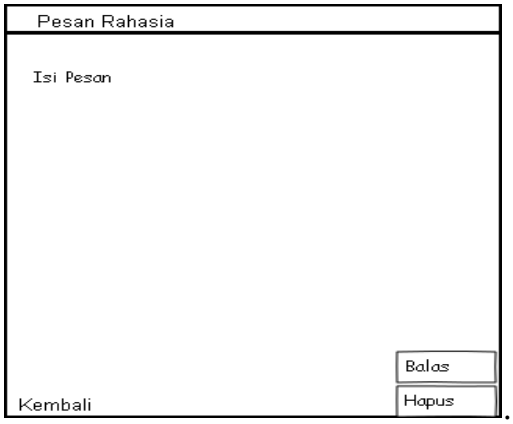
Antarmuka ini digunakan untuk memvalidasi apakah *user* berhak membuka SMS Rahasia atau tidak. Antarmuka ini juga digunakan ketika pengguna akan menghapus sebuah SMS, sehingga keamanan SMS tersebut terjaga.



Gambar 4.15 Rancangan Antarmuka Decode Pesan

e. Antarmuka Isi SMS Rahasia

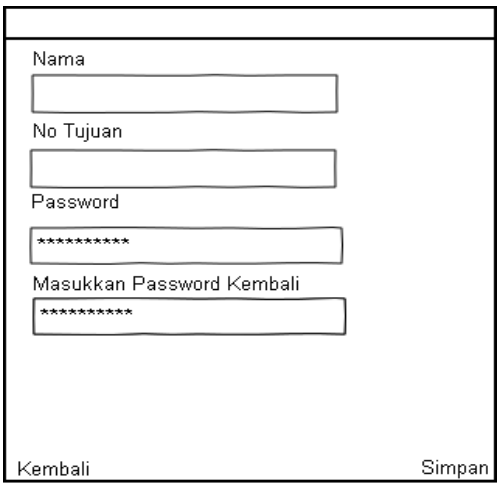
Antarmuka ini digunakan untuk menampilkan Isi SMS hasil *decode*.Untuk Isi SMS yang masuk maka menu dibawah kanan adalah balas dan hapus sedangkan SMS yang keluar menunya adalah kirim kembali dan hapus.



Gambar 4.16 Rancangan Antarmuka Isi SMS

f. Antarmuka Tambah Kontak

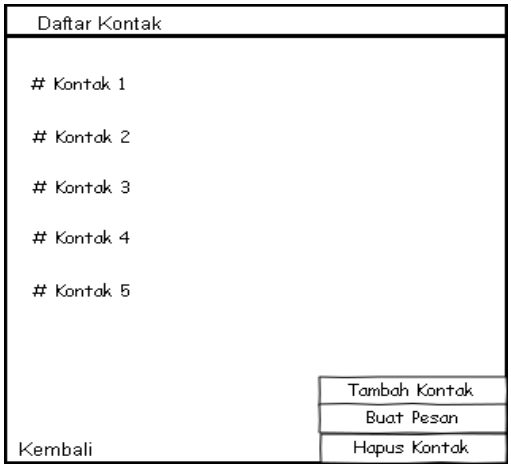
Antarmuka ini membutuhkan masukan nama,no tujuan dan *password* yang akan disimpan dalam sebuah *database*.



Gambar 4.17 Rancangan Antarmuka Tambah Kontak

g. Antarmuka Daftar Kontak

Antarmuka ini menampilkan daftar kontak-kontak yang tersadia dan menu tambah kontak, buat pesan (dari kontak yang dipilih), dan hapus kontak.



Gambar 4.18 Rancangan Antarmuka Daftar Kontak

4.2.4 Perancangan Pengujian

Dengan adanya fitur kompresi pada aplikasi, maka dibutuhkan sebuah pengujian apakah efektif atau tidak fitur kompresi tersebut. Maka akan dihitung dan dibandingkan, bit-bit pesan mulai dari pesan awal, pesan terenkripsi dan pesan yang terenkripsi dan terkompres. Akan diuji beberapa kalimat yang hurufnya berpola tertentu misalnya homogen dan heterogen.

BAB V
IMPLEMENTASI

5.1 Implementasi Secara Umum

Aplikasi SMS Rahasia diimplementasikan dengan bahasa pemrograman Java 2 Micro Edition. Implementasi dilakukan pada komputer yang memiliki spesifikasi hardware dan software sebagai berikut :

a. Prosessor Intel Core 2Duo T6670 2.20 GHz.
b. Memori 1GB
c. Sistem operasi Windows XP sp3
d. Net Beans IDE 6.5.1 dengan emulator platform Wireless Toolkit 2.5.2.

Device yang akan menjadi tempat aplikasi ini harus memenuhi spesifikasi minimum yaitu:

a. Memiliki J2ME Runtime Environment
b. J2ME Configuration yang digunakan adalah CLDC (Connected Limited Configuration) versi 1.0
c. J2ME Profile yang digunakan adalah MIDP (Mobile Information Device Profile) versi 2.0

5.2 Implementasi Pembuatan Aplikasi

5.2.1 Pemrograman WMA

WMA (*Wireless Messaging API*) merupakan sebuah API dalam J2ME yang menangani pengiriman dan penerimaan SMS. Karena didalam aplikasi ini SMS yang akan dikirimkan, dienkripsi terlebih dahulu (berubah dari text menjadi byte array) maka SMS yang akan dikirimkan harus dalam bentuk binary. Untuk lebih jelasnya dapat dilihat pada gambar 5.1

```
public void run()
{
  MessageConnection konek = this.midlet.koneksiSMS();
  int i = this.midlet.port();
  RMSPesanRahasia localSecureMessageStore = this.midlet.getpenyimpananPesan();
  BinaryMessage pesanBinary = (BinaryMessage)konek.newMessage("binary");
  pesanBinary.setAddress(NoHP2AlamatTujuan(this.message.getNoHP(), i));
  pesanBinary.setPayloadData(this.message.getDataTerenkripsi());
  try
  {
    konek.send(pesanBinary);
```

Gambar5.1. Kode Pengiriman SMS

```
public void run()
{
  MessageConnection koneksi = this.midlet.koneksiSMS();
  RMSPesanRahasia tempatPenyimpanan = this.midlet.getpenyimpananPesan();
  while (!this.midlet.matikan())
  {
    Object ygAkanDsimpan;
    try
    {
      BinaryMessage pesanBinary = (BinaryMessage)koneksi.receive();
```

Gambar 5.2 Kode Penerimaan SMS

5.2.2 Pembuatan *Encoding* Pesan

Sebuah pesan SMS mempunyai panjang maksimal 140 byte atau 160 karakter dalam penyandian ASCII 7 bit,yang mana format ini merupakan format standar yang digunakan SMS. Dengan Asumsi bahwa setelah teks terenkripsi teks menjadi lebih besar bytenya, maka dibutuhkan sebuah kompresi untuk mengusahakan agar pesan menjadi lebih kecil terlebih dahulu. Kompresi yang digunakan adalah metode Huffman, tetapi metode Huffman yang akan dipakai tidak akan memakai pohon Huffman karena keterbatasan banyaknya byte yang dapat ditampung oleh sebuah SMS. Sebagai penggantinya maka dibuatlah sebuah tabel, tabel huffman ini bersifat statis dan akan digunakan oleh aplikasi,baik sang pengirim (enkoder) maupun sang penerima(dekoder).

Jadi proses *encoding* pesan ini diawali dengan kompresi SMS terlebih dahulu, baru kemudian di enkripsi SMS itu dengan metode DES. Sedangkan

proses *decoding* pesan adalah kebalikannya, SMS diterima setelah itu didekompres baru kemudian didekripsi dengan metode DES.

Cara kerja kompresi ini mengganti bit-bit sebuah karakter dengan bit-bit huffman yang telah didefinisikan oleh sebuah tabel Huffman statis [YUK07] sebagaimana tabel yang terdapat pada lampiran 1. Beberapa karakter ASCII yang umumnya dipakai membuat sms diganti bit-nya sesuai tabel. Sisanya (simbol langka, karakter unikode lainnya), diberi awalan 1111110 diikuti kode karakter aslinya 16 bit.

Algorima DES dalam aplikasi ini berasal dari sebuah *library security* yang bernama bouncy castle. Pertama-tama inisialisasi *engine* kriptografi dan *key* (kunci yang paling sedikit harus 8 bytes,ini yang akan digunakan sebagai *password* dalam aplikasi SMS Rahasia). Seperti dalam gambar 5.3.

Setelah itu pembuatan sebuah fungsi untuk pemanggilan *chiper* yang akan digunakan untuk mengenkripsi sebuah byte array.Untuk lebih jelasnya dapat dilihat pada gambar 5.4

```
public class Enkripsi {

    private BufferedBlockCipher cipher;
    private KeyParameter kunci;

    public Enkripsi( byte[] kunci ){

        cipher = new PaddedBlockCipher(
                 new CBCBlockCipher(
                 new DESEngine() ) );

        this.kunci = new KeyParameter( kunci );
    }

    public Enkripsi( String kunci ){
        this( kunci.getBytes() );
    }
```

Gambar 5.3 Inisialisasi engine kriptografi dan key

```
private byte[] chiper( byte[] data )
throws CryptoException {
    int    size =
            cipher.getOutputSize( data.length );
    byte[] result = new byte[ size ];
    int    olen = cipher.processBytes( data, 0,
            data.length, result, 0 );
    olen += cipher.doFinal( result, olen );

    if( olen < size ){
        byte[] tmp = new byte[ olen ];
        System.arraycopy(
                result, 0, tmp, 0, olen );
        result = tmp;
    }

    return result;
}
```

Gambar 5.4 Pembuatan Chiper

Kemudian baru dibuat fungsi enkripsi dan dekripsi dengan engine kriptografi dan kunci yang sudah diinisialisasi tadi seperti pada gambar 5.5

```
public  synchronized byte[] DESencrypt( byte[] data )
throws CryptoException {
    if( data == null || data.length == 0 ){
        return new byte[0];
    }
    cipher.init( true, kunci );
    return chiper( data );
}
public synchronized byte[] DESdecrypt( byte[] data )
throws CryptoException {
    if( data == null || data.length == 0 ){
        return new byte[0];
    }

    cipher.init( false, kunci );
    return chiper( data );
}
```

Gambar 5.5 DES enkripsi dan dekripsi

5.2.3 Pembuatan Database Pesan

Setelah pesan dikirim atau diterima , pesan kemudian disimpan dalam sebuah *database*. *Database* dalam J2ME disebut RMS (*Record Managemen System*). Sebelum menyimpan sebuah pesan pertama harus membuat *database* terlebih dahulu. Selain digunakan untuk membuat *method* tersebut juga digunakan untu membuka *record* pada *database*.

Parameter this.nama merujuk pada nama *database* yang bernama dbPesan, sedangkan parameter *true* berarti dbPesan akan secara otomatis terbuat jikalau memang belum ada. Setelah itu baru membuat fungsi simpan recordpesan.

```
public void open()
  throws RecordStoreFullException, RecordStoreException
{
  try
  {
    this.rs = RecordStore.openRecordStore(this.nama, true);
  }
  catch (RecordStoreNotFoundException databasetdkketemu)
  {
```

Gambar 5.6 Membuat dan membuka database

```
public int simpanPesan(PesanRahasia paramPesanRahasia)
  throws RecordStoreNotOpenException, RecordStoreException, IOException
{
  byte[] recordPesan = paramPesanRahasia.toByteArray();
  return this.rs.addRecord(recordPesan, 0, recordPesan.length);
}
```

Gambar 5.7 Membuat fungsi simpan pesan

Isi dari recordPesan tersebut adalah NoHP,jenis pesan(yang masuk atau yang keluar) dan data pesan yang sudah dienkripsi. Untuk lebih jelasnya dapat dilihat pada gambar 5.8.

```
public byte[] toByteArray()
  throws IOException
{
  ByteArrayOutputStream localByteArrayOutputStream = new ByteArrayOutputStream();
  DataOutputStream localDataOutputStream = new DataOutputStream(localByteArrayOutputStream);
  localDataOutputStream.writeUTF(this.NoHP);
  localDataOutputStream.writeInt(this.jenis);
  localDataOutputStream.write(this.dataTerenkripsi);
  return localByteArrayOutputStream.toByteArray();
}
```

Gambar 5.8 Isi dari record Pesan

5.2.4 Penghitungan Bit

Setelah sms mengalami kompresi perlu dianalisis juga apakah bit dari sms tersebut mengalami perubahan menjadi lebih kecil atau tidak.Kode penghitungan bit sms dapat dilihat pada gambar 5.9

```
//plainteks
int pjg = pesan.length(); //hitung jmlh karakter
int awal = pjg*7;     //dikali 7 karena sms menggunakan ASCII 7 bit

//enkripsi
Enkripsi des = new Enkripsi (pass);
byte[]b =des.encryptString(pesan);
//hitung jmlh byte dikali 8 ditambah untuk port 4 byte
int desbit = (((b.length)*8)+(4*8));

//enkripsi+huffman
byte[]c =Enkripsi.encrypt(pesan, pass);
//hitung jmlh byte dikali 8 ditambah untuk port 4 byte
int deshuffbit = (((c.length)*8)+(4*8));
```

Gambar 5.9 Kode Penghitungan Bit

5.2.5 Deskripsi Error Handling

Error Handling menangani proses ketika terjadi kesalahan pada proses seperti berikut :

1. Proses penulisan pesan, jika terjadi kesalahan maka akan muncul pesan error menulis pesan
2. Proses pengiriman sms, jika terjadi kesalahan maka akan muncul pesan error gagal mengirm sms
3. Proses menyimpan pesan, jika terjadi kesalahan maka akan muncul pesan error simpan pesan
4. Proses menampilkan daftar pesan, jika terjadi kesalahan maka akan muncul pesan error gagal akses database
5. Proses decode pesan, jika terjadi kesalahan maka akan muncul pesan error salah password atau pesan telah rusak

5.2.6 Pembuatan Antarmuka

a. Menu Utama

Menu utama adalah tampilan pertama yang akan dilihat oleh pengguna ketika menjalankan aplikasi ini.Pada antarmuka ini disediakan menu-menu pilihan yang dapat dipilih oleh pengguna:

Gambar 5.9 Antarmuka Menu Utama

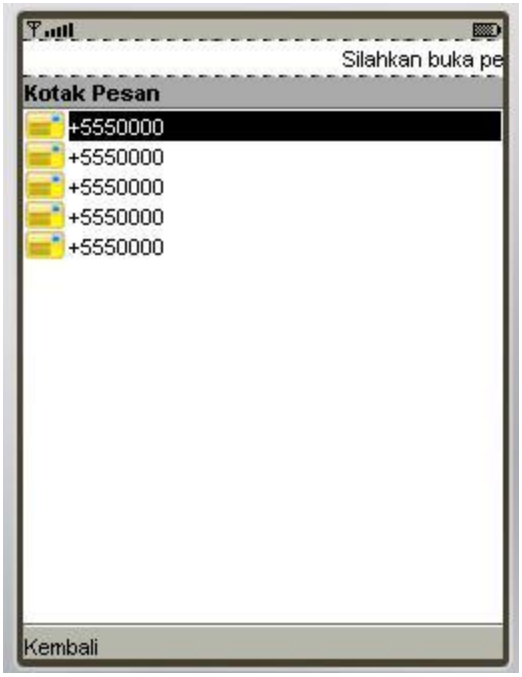
b. Menu Tulis SMS Rahasia

Menu ini digunakan untuk membuat .pesan yang akan pengguna kirimkan. Untuk lebih jelasnya seperti pada gambar 5.10.

c. Antarmuka Kotak Pesan

Antarmuka ini untuk menampilkan daftar pesan yang akan dibuka oleh pengguna. Untuk lebih jelasnya seperti pada gambar 5.11

Gambar 5.10 Antarmuka Tulis SMS Rahasia



Gambar 5.11 Antarmuka Kotak Pesan

d. Antarmuka Membuka Pesan (Decode Pesan)

Antarmuka ini digunakan untuk membuka sebuah pesan yang membutuhkan masukan *password*. Untuk lebih jelasnya seperti pada gambar 5.12

Gambar 5.12 Antarmuka Decode Pesan

e. Antar muka Isi SMS Rahasia

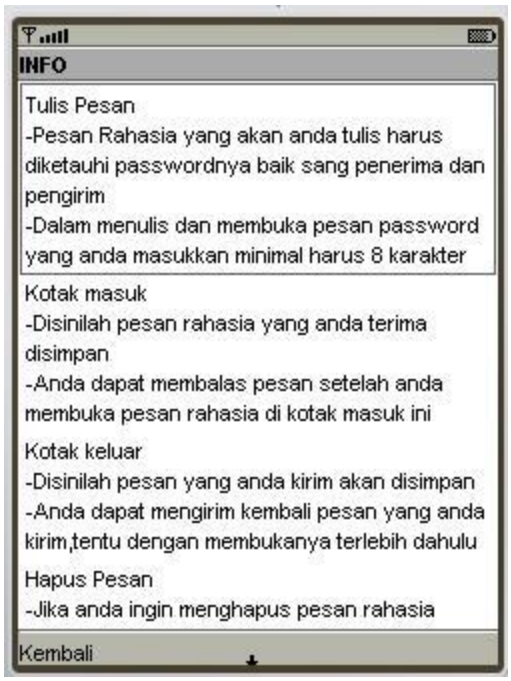
Antarmuka ini digunakan untuk menampilkan Isi SMS hasil *decode*. Untuk Isi SMS yang keluar menunya adalah kirim kembali dan hapus. Untuk lebih jelasnya seperti pada gambar 5.13

f. Antar muka Bantuan dan About

Antarmuka ini digunakan untuk menampilkan keterangan tentang tata cara pemakain aplikasi dan hal-hal yang perlu diketauhi oleh pengguna. Untuk

antarmuka about tampilannya sama hanya yang ditampilkan adalah keterangan *about* aplikasi. Untuk lebih jelasnya seperti pada gambar 5.14

Gambar 5.13 Antarmuka Isi Pesan



Gambar 5.14 Antarmuka Bantuan

g. Antar muka Daftar Kontak

Antarmuka ini digunakan untuk menampilkan daftar kontak yang ada, jika ingin menambah kontak ada menu tambah kontak, selain itu ada menu buat pesan dan hapus kontak.

Gambar 5.15 Antarmuka Daftar Kontak

h. Antar muka Tambah Kontak

Antarmuka ini digunakan untuk menambah kontak baru.

Gambar 5.16 Antarmuka Tambah Kontak

BAB VI
ANALISIS KINERJA

Setelah melalui tahap implementasi, maka aplikasi yang dibangun perlu diuji untuk mengetahui kemampuan aplikasi. Apakah aplikasi mampu menangani kesalahan (*error handling and exception*) dengan baik serta mampu mengkomunikasikannya kepada pemakai aplikasi untuk kemudian memberikan solusi atau saran kepada pemakai atau tidak.

6.1 Kinerja Aplikasi pada fitur-fitur SMS Rahasia

6.1.1 Proses Menulis Pesan

Dalam proses ini pengguna memasukkan *input* berupa No tujuan,Isi SMS dan *password* setidaknya 8 karakter. Kemudian melanjutkan ke proses pengiriman SMS. Kegagalan dalam proses ini kemungkinan terjadi karena pengguna tidak mengisi dari salah satu 3 masukan yang diminta (no,pesan,dan *password*). Tetapi kegagalan juga bisa terjadi walaupun sudah mengisi ke tiga masukan ,karena *password* yang ditulliskan kurang dari 8 karakter. Gambar 6.1 menunjukkan aksi yang diberikan oleh aplikasi ketika terjadi kesalahan dalam memasukkan data *password*.

6.1.2 Proses Mengirim Pesan

Setelah pesan mengalami *encoding*, pesan akan dikirm menuju no tujuan sesuai yang dimasukkan pada proses menulis pesan melalui port yang sudah ditentukan dalam aplikasi. Jika terjadi kesalahan dalam prose pengiriman pesan maka aplikasi akan menunjukkan tampilan *error* seperti pada gambar 6.2. Itu terjadi ketika mengirm sms dan tidak ada pulsa atau tidak adanya jaringan.

Gambar 6.1 Tampilan error salah input password

Gambar 6.2 Tampilan error saat gagal mengirim pesan

6.1.3 Proses Menyimpan Pesan

Proses penyimpanan pesan otomatis terjadi ketika pengguna selesai mengirim pesan atau menerima pesan. Jika pesan tidak berhasil disimpan dalam RMS yang bernama dbPesan maka aplikasi akan menunjukkan pesan *error* seperti pada gambar 6.3

Gambar 6.3 Tampilan error gagal menyimpan pesan

6.1.4 Proses Menampilkan Daftar Pesan di Kotak Masuk atau Keluar

Aplikasi akan menampilkan daftar pesan yang diambil dari *database* dbPesan sesuai pilihan pengguna,daftar pesan yang sudah masuk atau daftar pesan keluar. Jika aplikasi gagal mengakses pesan yang ada dalam *database* maka akan terjadi tampilan *error* seperti pada gambar 6.4

Gambar 6.4 Tampilan error saat membuka kotak pesan

6.1.5 Proses Decode Pesan

Selain terjadi ketika pengguna membuka pesan yang ada di kotak masuk dan keluar , proses decode pesan juga terjadi otomatis ketika pengguna menerima sebuah pesan. Kegagalan dalam decode pesan bisa terjadi karena password yang digunakan pengirim tidak sama dengan yang *password* penerima inputkan, Kegagalan dapat terjadi juga jika pesan mengalami kerusakan dalam proses pengiriman. Jika itu terjadi aplikasi akan menampikan pesan *error* seperti pada gambar 6.5

Gambar 6.5 Tampilan jika salah memasukkan password

6.2 Analisis Keefektifan Kompresi dalam Aplikasi

Hitungan bit teks awal (teks tanpa *encoding*) adalah berasal dari jumlah karakter dikali 7 bit (karena dalam format sms biasa 1 karakter 7 bit),. Sedangkan pada des dan des+huff didapat dari banyaknya byte ditambah 4 byte (4 byte digunakan untuk port) dikali 8 bit.Hasil analisis efektifitas kompresi dapat dilihat pada tabel 6.1.

Teks no 1 adalah jenis kalimat homogen (banyak huruf yang berjenis sama). Teks 2 berpola kalimat yang huruf-hurufnya sering kita gunakan sehari-hari misalnya a dan e. Teks no 3 adalah kalimat yang hurufnya jarang kita gunakan sehari-hari misalnya z,v,w dan x. Sedangkan teks no 4 adalah kalimat berjenis hetrerogen (tidak ada sama sekali huruf yang sama).

Jadi selain kalimat homogen, kalimat yang menggunakan huruf yang sering digunakan juga merupakan salah satu faktor pendukung keberhasilan pengkompresan.

Tabel 6.1 Efisiensi Kompresi

No	Teks	Banyak bit			Besar Kompresi(%)
		awal	des	des+huff	
1	Telatttttttttt lgiiiiiii	175	288	160	44,444
2	Maaaf,saya datang terlambat motor saya mogok di jalan,saya akan segera menyusul	553	672	480	28,571
3	Oryza sativa quw mw adoxography?	224	352	288	18,181
4	Vit, mw kzoqja brg?,yux	161	224	224	0,000

6.3 Kelebihan dan Kekurangan Aplikasi

Aplikasi ini memiliki kelebihan dan kekurangan, diantaranya yang menjadi kelebihannya yaitu:

1. Adanya fitur penyimpanan pesan dalam aplikasi ini sehingga pengguna yang ingin membuka pesan itu kembali dapat membacanya lagi, membalas pesan atau mengirim kembali pesan tersebut.
2. Penyertaan kompresi dalam proses *encoding* pesan sehingga mengurangi kelemahan proses enkripsi yang cenderung membuat pesan menjadi lebih besar.
3. Aplikasi ini mempunyai keamanan yang baik karena validasi (pemasukan *password*) tidak hanya dilakukan ketika membuka sebuah pesan tetapi juga ketika menghapus pesan. Sehingga keamanan isi pesan terjaga

Sedangkan kekurangan aplikasi ini yaitu:

1. Tidak adanya pembuatan *digest* didalam aplikasi ini yang biasanya digunakan untuk mengecek integritas sebuah data hasil enkripsi.

2. Tidak ada fitur hapus semua pesan di kotak masuk maupun kotak keluar, sehingga untuk menghapus pesan, pengguna harus menghapusnya satu demi satu pesan. Dari segi kemudahan ini merepotkan.

BAB VII
KESIMPULAN DAN SARAN

7.1 Kesimpulan

Dari pembuatan aplikasi enkripsi SMS dengan meode DES ini dapat disimpulkan bahwa:

1. Aplikasi ini dapat menjaga privasi dan kemanan pengguna dalam mengirim dan menerima sebuah SMS. Karena keamanan isi pesan terjaga dengan baik.

2. Penggunaan kompresi dalam aplikasi ini dapat mengurangi efek bertambah besarnya SMS hasil dari enkripsi.

3. Tekhnologi RMS (*database*) pada J2ME sangat berguna bagi aplikasi ini karena pesan tidak harus dibaca saat itu juga. Sekaligus juga dapat mengecek pesan-pesan apa saja yang sudah pengguna kirimkan kepada seseorang.

7.2 Saran

Beberapa saran untuk pengembangan enkripsi SMS dengan meode DES selanjutnya sebgai berikut:

1. Pembuatan aplikasi di sistem operasi lain misal android yang sedang *booming* atau di *device smartphone* lain.

2. Untuk kedepannya aplikasi dapat di lengkapi fitur penambahan *digest* pada hasil enkripsi. Walaupun harus diuji keefektifan terlebih dahulu, karena penambahan digest juga akan mengakibatkan berkurangnya kapasitas SMS.

3. Penggunaan Algoritma Enkripsi yang lain misal tripel DES, RSA atau algortima enkripsi yang lainnya.

4. Penambahan fitur master *password* (validasi *password* untuk membuka aplikasi), sehingga bisa dibuat menu hapus semua pesan ketika pengguna

ingin menghapus semua pesan di kotak masuk atau kotak keluar tanpa harus memasukkan *password* untuk setiap pesan.

DAFTAR PUSTAKA

[HAR03] Hariyanto, Bambang. *Esensi-esensi Bahasa Pemrograman Java*. Bandung: Informatika Bandung, 2003.

[ARI06] Ariyus, Dony. *Kriptografi Keamanan Data dan Komunikasi*. Yogyakarta:Graha Ilmu,2006

[WIC02] Wicaksono, A., *Pemrograman Aplikasi Wireless dengan Java*. Jakarta: PT. Elex Media Komputindo, 2002.

[SIM06] Simarmara, Janner. *Pengamanan Sistem Komputer*. Yogyakarta:Andi,2006

[KRI03] Kristanto, Andi. *Keamanan Data Pada Jaringan Komputer*. Yogyakarta:Gava Media,2003

[RAH02] Raharjo, Budi. *Keamanan Sistem Informasi Berbasis Internet*. Bandung:PT Insan Infonesia,2002

[YUK07] Yuku. *SMS Text Compression / Kompresi Teks SMS*. http://www.kejut.com/kompresisms, diakses pada tanggal 2 Mei 2012

LAMPIRAN 1

Tabel Statis Huffman

	Awalan	Badan	Ubahan
a	0	0000	00000
b		00101	000101
c		1101	01101
d		01011	001011
f		11001	011001
g		10101	010101
h		1000	01000
i		0001	00001
j		110001001	0110001001
k		1100011	01100011
l		1001	01001
m		10100	010100
n		0110	00110
o		0100	00100
p		10110	010110
q		1100010000	01100010000
r		0111	00111
s		0011	00011
t		111	0111
u		10111	010111
v		0101001	00101001
w		0101000	00101000
x		11000101	011000101
y		00100	000100
z		1100010001	01100010001
A	100	0000	1000000
B		00101	10000101
C		1101	1001101
D		01011	10001011
F		11001	10011001
G		10101	10010101
H		1000	1001000
I		0001	1000001
J		110001001	100110001001
K		1100011	1001100011
L		1001	1001001
M		10100	10010100
N		0110	1000110
O		0100	1000100
P		10110	10010110
Q		1100010000	1001100010000
R		0111	1000111
S		0011	1000011
T		111	100111
U		10111	10010111
V		0101001	1000101001
W		0101000	1000101000
X		11000101	100110001101
Y		00100	10000100
Z		1100010001	1001100010001
<space>	101	1	1011
e		0	1010
!	110	0011	1100011
"		0110	1100110
'		0100	1100100
,		0001	1100001
-		0101	1100101
.		0000	1100000
?		0010	1100010
E		0111	1100111
0	111	0000	1110000
1		0001	1110001
3		0011	1110011
4		0100	1110100
5		0101	1110101
6		0110	1110110
7		0111	1110111
8		1000	1111000
9		1001	1111001
(		11010	11111010
)		11011	11111011
/		1011	1111011
2		0010	1110010
:		1100	1111100
@		1010	1111010
SISA HURUF	1111110		1111110