APLIKASI ENKRIPSI SMS PADA TELEPON SELULAR

BERBASIS J2ME

DENGAN METODE VIGENERE CIPHER

SKRIPSI

Oleh:
BAYU KRISTIAN NUGROHO
J2F 004 262

PROGRAM STUDI TEKNIK INFORMATIKA
JURUSAN MATEMATIKA
FAKULTAS MATEMATIKA DAN ILMU PENGETAHUAN ALAM
UNIVERSITAS DIPONEGORO
SEMARANG

2010

APLIKASI ENKRIPSI SMS PADA TELEPON SELULAR

BERBASIS J2ME

DENGAN METODE VIGENERE CIPHER

Oleh:
BAYU KRISTIAN NUGROHO
J2F 004 262

SKRIPSI

Telah diperiksa dan disetujui sebagai salah satu syarat untuk memperoleh gelar Sarjana
Komputer pada
Program Studi Teknik Informatika

PROGRAM STUDI TEKNIK INFORMATIKA
JURUSAN MATEMATIKA
FAKULTAS MATEMATIKA DAN ILMU PENGETAHUAN ALAM
UNIVERSITAS DIPONEGORO
SEMARANG

2010

PENGESAHAN

Judul : Aplikasi Enkripsi SMS pada Telepon Selular berbasis J2ME dengan Metode Vigenere Cipher

Nama : Bayu Kristian Nugroho

NIM : J2F 004 262

Telah diujikan pada sidang Tugas Akhir tanggal 3 Agustus 2010 dan dinyatakan lulus pada tanggal 20 Agustus 2010.

Semarang, 21 Agustus 2010
Panitia Penguji Tugas Akhir
Ketua,

Drs. Eko Adi Sarwoko, M.Ikom
NIP. 196511071992031003

Mengetahui,
Ketua Jurusan Matematika

Dr. Widowati, S.Si, M.Si
NIP.196902141994032002

Mengetahui,
Ketua Program Studi Teknik Informatika

Drs. Eko Adi Sarwoko, M.Ikom
NIP.196511071992031003

PENGESAHAN

Judul : Aplikasi Enkripsi SMS pada Telepon Selular berbasis J2ME dengan Metode Vigenere Cipher

Nama : Bayu Kristian Nugroho

NIM : J2F 004 262

Telah diujikan pada sidang Tugas Akhir tanggal 3 Agustus 2010.

Semarang, 20 Agustus 2010

Pembimbing Utama

Drs. Kushartantya, M.IKom
NIP. 195003011979031003

Pembimbing Anggota

Drs. Putut Sri Warsito
NIP.195306281980031001

ABSTRAK

Perkembangan teknologi telekomunikasi yang begitu pesat telah memberikan manfaat yang begitu besar. Dengan adanya teknologi telekomunikasi, jarak dan waktu bukan lagi menjadi sebuah kendala yang berarti. Salah satu hasil teknologi telekomunikasi yang sangat terkenal adalah *Short Message Service* (SMS). Dengan menggunakan SMS, penggunanya dapat saling bertukar pesan teks dengan pengguna lain.

Pada tugas akhir ini dikembangkan sebuah aplikasi pada telepon selular untuk memodifikasi pesan SMS menjadi cipherteks agar isi informasi dari SMS tersebut tidak diketahui oleh orang lain. Untuk pengiriman SMS sistem mengenkripsi pesan menjadi cipherteks menggunakan key yang diinputkan oleh pengirim kemudian mengirimkan ke nomor tujuan. Untuk penerimaan SMS sistem mendekripsi cipherteks menjadi plainteks menggunakan key yang diinputkan oleh penerima kemudian menampilkan pesan asli kepada penerima. Aplikasi ini dapat dimanfaatkan oleh seseorang yang ingin mengirimkan suatu informasi rahasia kepada orang lain melalui SMS tanpa takut informasi dari pesan tersebut akan diketahui oleh orang lain.

Metode yang digunakan sistem dalam mengenkripsi dan mendekripsi pesan adalah metode enkripsi substitusi vigenere cipher dan implementasinya menggunakan bahasa pemrograman Java 2 *Micro Edition* (J2ME).

Kata kunci: SMS, enkripsi, dekripsi.

ABSTRACT

Rapid expansion of telecommunication technology has brought a very big benefit for us. With the existence of telecommunication technology, many constraints such as distance, location, or time can be overcome. One of the technological results in telecommunication technology is Short Message Service or usually known as SMS. By using an SMS, the subscribers can do some exchange of text messages over each other.
In this final assignment, developed an aplication on a cell phone to modify a text into ciphertext so that information from the message is not known by others. For sending an SMS, system encrypts plaintext into ciphertext using a key that is inputed by a sender and then send it to destination number. For receiving an SMS, system decrypt ciphertext into plaintext using a key that is inputed by receiver and then displaying plaintext to receiver. This application can be used by someone who wants to send a secret information to other through an SMS without fear of information from those messages will be known by others.
Metode that is used by system to encrypt and decrypt message is vigenere cipher substitution encryption metode and it will implemented using Java 2 Micro Edition programming languages.

Keyword: SMS, encrypt, decrypt

KATA PENGANTAR

Alhamdulillah, segala puji penulis panjatkan kehadirat Allah SWT yang telah melimpahkan rahmat dan hidayah-Nya sehingga penulis dapat menyelesaikan penyusunan tugas akhir ini. Serta tidak lupa sholawat dan salam penulis sampaikan kepada Rasulullah SAW beserta keluarganya, sahabatnya, dan orang-orang yang mengikuti sunnahnya.

Tugas akhir berjudul "**Aplikasi Enkripsi SMS pada Telepon Selular berbasis J2ME dengan Metode Enkripsi Substitusi**" ini disusun sebagai salah satu syarat untuk memperoleh gelar sarjana pada Program Studi Ilmu Komputer Fakultas Matematika dan Ilmu Pengetahuan Alam Universitas Diponegoro Semarang. Ucapan terima kasih ingin penulis sampaikan kepada seluruh pihak yang telah membantu dalam proses penyelesaian tugas akhir ini, diantaranya adalah :

1. Ibu Dra. Widowati, M.Si, selaku ketua Jurusan Matematika fakultas MIPA Universitas Diponegoro Semarang.
2. Bapak Drs. Eko Adi Sarwoko, M.Kom, selaku ketua Program Studi Teknik Informatika Jurusan Matematika fakultas MIPA Universitas Diponegoro Semarang.
3. Bapak Drs. Kushartantya, M.Ikomp, selaku dosen pembimbing I yang telah meluangkan waktunya untuk membimbing penulis.
4. Bapak Drs. Putut Sri Warsito, selaku dosen pembimbing II yang telah meluangkan waktunya untuk membimbing penulis.
5. Bapak Priyo Sidik S, S.Si, M.Kom, selaku dosen wali penulis.
6. Seluruh dosen Program Studi Teknik Informatika jurusan Matematika FMIPA UNDIP yang telah membantu penulis selama masa studi.
7. Bapak dan Ibu serta kakak yang telah memberikan dorongan dan doa kepada penulis dalam usahanya menyelesaikan tugas akhir ini.
8. Dhita Yutantri, setiap hari dengan cerewetnya yang khas selalu memberikan semangat kepada penulis.
9. Keluarga B49 dan SamSat yang telah banyak memberikan kenyamanan di Semarang dan banyak memberikan wawasan baru walaupun tanpa praktek.

10. Semua pihak yang telah membantu penulis yang tidak dapat disebutkan namanya satu persatu oleh penulis.

Usaha yang maksimal telah penulis lakukan demi selesainya penulisan tugas akhir ini. Baik dengan mengimplementasikan pengetahuan yang didapat dari perkuliahan maupun teori-teori penunjang lain yang didapat dari luar perkuliahan. Diharapkan dengan menerapkan ilmu yang didapat dari perkuliahan dan menambahnya dengan ilmu yang didapat dari luar perkuliahan dapat membuka mata kita agar tidak melihat dari satu sudut pandang saja.

Penulis sadar bahwa masih banyak kekurangan dalam penulisan laporan tugas akhir ini, untuk itu penulis mengharapkan saran serta kritik yang membangun dari pembaca. Penulis berharap bahwa tugas akhir ini dapat menginspirasi orang lain sehingga di masa mendatang tugas akhir ini dapat dikembangkan lebih lanjut dan semoga tugas akhir ini dapat bermanfaat. Amin.

Semarang, Agustus 2010

Penulis

DAFTAR ISI

DAFTAR TABEL

DAFTAR GAMBAR

BAB I
PENDAHULUAN

1.1 Latar Belakang

Beberapa tahun terakhir ini terjadi perkembangan yang pesat pada teknologi, salah satunya adalah telepon selular (ponsel). Mulai dari ponsel yang hanya bisa digunakan untuk bicara dan sms hingga "ponsel cerdas" (*smart phone*) yang memiliki berbagai fungsi seperti *multimedia*, *multiplayer games*, transfer data, video *streaming* dan lain-lain. Berbagai perangkat lunak untuk mengembangkan aplikasi ponselpun bermunculan, diantaranya yang cukup dikenal luas adalah Java 2 *Micro Edition* (J2ME).

Salah satu fasilitas yang disediakan ponsel adalah untuk melakukan pengiriman data berupa pesan singkat melalui *Short Message Service* (SMS). Namun dengan fasilitas SMS yang ada, timbul pertanyaan mengenai keamanan informasi jika seseorang ingin mengirimkan suatu informasi rahasia melalui fasilitas SMS.

Di luar negeri pemanfaatan SMS untuk mengirim pesan rahasia telah lebih dulu dikembangkan. Misalnya di Inggris sebuah perusahaan operator telepon selular, staellium UK, mengeluarkan layanan bernama "*stealth text*" yang dapat digunakan untuk mengirim pesan dengan aman, yaitu dengan cara menghapus pesan secara otomatis segera setelah 40 detik pesan dibaca atau yang dikenal dengan nama *self-destruct text message*. Kini dengan memanfaatkan *Wireless Messaging* API *(Application Programming Interface)* dari J2ME para pembuat program Java dapat mengembangkan sendiri sebuah aplikasi pengiriman pesan singkat atau SMS yang dimodifikasi untuk mengamankan pesan.

1.2 Rumusan Masalah

Berdasarkan latar belakang masalah tersebut dapat dibuat suatu rumusan masalah, yaitu:

"Bagaimana cara memanfaatkan layanan SMS yang dikenal mudah dalam hal penggunaan agar dapat juga dipakai untuk mengirim dan menerima pesan yang

bersifat rahasia, dimana informasi atau isi dari pesan tersebut akan tidak mudah diketahui oleh pihak yang tidak berhak".

1.3 Batasan Masalah

Agar fokus penelitian ini terjaga, diberikan beberapa batasan sebagai berikut :

1. *Input* berupa pesan SMS.
2. Spesifikasi SMS (panjang 1 pesan SMS) disesuaikan dengan standar teknologi *Global System for Mobile Communication* (GSM).
3. Pengujian aplikasi hanya dilakukan pada emulator *Wireless Toolkit.*
4. Aplikasi ini tidak memiliki *inbox* sehingga pesan yang masuk tidak dapat disimpan untuk dibaca kembali.
5. Pengiriman pesan dengan menggunakan fasilitas *Wireless Messaging* API (WMA) dari Java 2 *Micro Edition* (J2ME).
6. Bahasa pemrograman yang digunakan adalah Java 2 *Micro Edition*.

1.4 Tujuan dan Manfaat

Tujuan dari penulisan tugas akhir ini adalah menghasilkan suatu aplikasi pada telepon selular yang dapat digunakan untuk mengirim dan menerima pesan teks sekaligus memiliki fasilitas untuk mengamankan atau menyembunyikan informasi dari pesan yang dikirimkan.

Manfaat dari penulisan tugas akhir ini :

1. Bagi pembaca

   Dengan menggunakan aplikasi pada tugas akhir ini seseorang dapat mengirimkan suatu informasi rahasia tanpa takut diketahui isi informasi tersebut oleh orang lain
2. Bagi penulis
   - Bisa lebih memahami lebih dalam bahasa pemrograman J2ME
   - Lebih memahami algoritma enkripsi substitusi vigenere cipher

1.5 Sistematika Penulisan

Sistematika penulisan yang digunakan dalam tugas akhir ini terbagi dalam beberapa pokok bahasan, yaitu :

BAB I PENDAHULUAN

Bab ini menguraikan tentang latar belakang, batasan masalah, tujuan dan manfaat penelitian dan sistematika penulisan

BAB II DASAR TEORI

Bab ini membahas dasar-dasar teori dari Java 2 *Micro Edition*, *Wireless Messaging API*, *Over The Air (OTA)* dan ilmu kriptografi

BAB III ANALISIS KEBUTUHAN DAN PERANCANGAN

Bab ini berisi tentang analisis kebutuhan pada aplikasi KriptoSMS dan perancangan perangkat lunak dengan menggunakan bahasa pemrograman Java 2 *Micro Edition*

BAB IV IMPLEMENTASI, PENGUJIAN DAN ANALISIS HASIL

Bab ini berisi tentang implementasi dan pengujian dari perangkat lunak yang telah dibuat beserta analisis hasilnya

BAB V PENUTUP

Bab ini berisi tentang kesimpulan dan saran yang didapatkan selama proses perancangan dari sistem serta rencana pengembangan dari perangkat lunak di masa yang akan datang

DAFTAR PUSTAKA

Bab ini berisi tentang referensi-referensi yang telah digunakan selama pembuatan tugas akhir ini sebagai acuan yang mendukung

BAB II
DASAR TEORI

2.1 Teknologi Java 2

Java adalah bahasa pemrograman *Object Oriented Programming (OOP)* yang dibuat oleh Sun Microsystem. Java dirancang untuk menjadi bahasa yang memiliki kemampuan tinggi dalam hal portabilitas dan pemanfaatan jaringan tanpa mengabaikan kestabilan, kemanan, serta kemudahan dari sisi desain dan pemrograman aplikasi.

Sebutan Java 2 diberikan untuk Java versi 1.2 dan versi berikutnya. Java 2 terbagi dalam 3 kategori[6], yaitu:

- Java 2 *Standard Edition* (J2SE). Kategori ini digunakan untuk menjalankan mengembangkan aplikasi Java pada level komputer *personal*.
- Java 2 *Enterprise Edition* (J2EE). Kategori ini dikhususkan untuk pengembangan aplikasi Java pada lingkungan *enterprise*/server.
- Java 2 *Micro Edition* (J2ME). Kategori ini digunakan untuk pengembangan aplikasi Java yang diimplementasikan pada perangkat semacam ponsel, Palm, PDA dan PocketPC.

2.2 Java 2 *Micro Edition*

Java 2 *Micro Edition* (J2ME) dirancang untuk dapat menjalankan program Java pada perangkat yang memiliki kemampuan terbatas misalnya kecilnya jumlah memori yang dimiliki perangkat tersebut.

2.2.1 Konfigurasi J2ME

Konfigurasi J2ME adalah spesifikasi yang mendefinisikan sebuah *virtual machine* dari kumpulan API-API dasar yang dapat digunakan dalam kelas tertentu dari sebuah peralatan. *Virtual machine* pada J2ME berbeda dengan yang ada pada J2SE karena hanya fitur-fitur penting yang berkaitan dengan perangkat tanpa kabel *(Wireless)* saja yang diimplementasikan.

Ada 2 konfigurasi pada J2ME[6], yaitu:

1. CLDC (*Connected Limited Device Configuration*)

   CLDC merupakan perangkat atau konfigurasi dasar dari J2ME. CLDC sebenarnya berupa library dan API (*Application Programming Interface)* yang diimplementasikan pada J2ME. Konfigurasi ini biasanya untuk alat kecil seperti telepon seluler (*handphone*), pager dan PDA. Peralatan tersebut biasanya mempunyai keterbatasan memori (RAM), sumber daya, dan kemampuan memproses.

2. CDC (*Connected Device Configuration*)

   CDC merupakan perangkat atau konfigurasi superset dari CLDC. Konfigurasi ini biasanya dipakai untuk alat seperti Internet TV, Nokia *Communicator* dan *Car* TV.

2.2.2 Profil J2ME

Sebuah profil dibangun dalam sebuah konfigurasi, namun ditambahkan beberapa API khusus agar dihasilkan sebuah lingkungan yang lengkap untuk membangun aplikasi. Profil berisi daur hidup *(life cycle),* antarmuka pemakai *(user interface),* serta penyimpanan. Salah satu profil J2ME adalah *Mobile Information Device Profile* (MIDP). Profil MIDP menyediakan sebuah *platform* standar untuk peralatan komunikasi bergerak yang memiliki kapasitas memori terbatas sehingga cocok untuk pengembangan aplikasi pada ponsel.

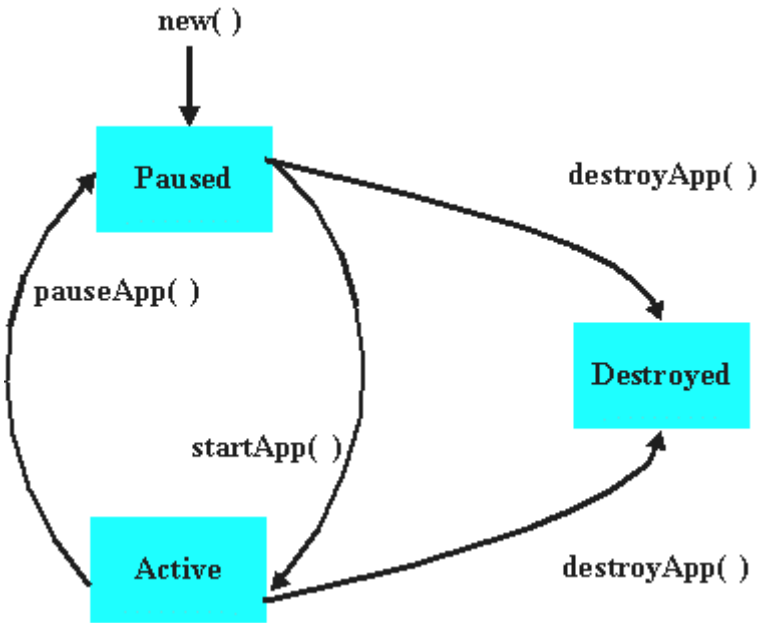
2.3 MIDP dan MIDlet

MIDP merupakan profil yang banyak digunakan dan popular dari J2ME dan MIDlet merupakan aplikasi-aplikasi yang dibuat di dalam handphone menggunakan profil MIDP.

2.3.1 Daur Hidup MIDlet

Daur hidup dari sebuh MIDlet ditangani oleh *Aplication Management Software* (AMS). AMS adalah sebuah lingkungan tempat siklus dari sebuah MIDlet diciptakan, dijalankan, dihentikan, maupun dihilangkan. AMS sering disebut dengan

Java Aplication Manager (JAM). Dalam daur hidupnya MIDlet memiliki tiga status, yaitu *Pause*, *Active* dan, *Destroy*. Ketika masing-masing status dipanggil, beberapa fungsi standar yang bersesuaian akan dipanggil.



Gambar 2.1 Daur hidup MIDlet.

Dari gambar 2.1 dapat dijelaskan sebagai berikut :

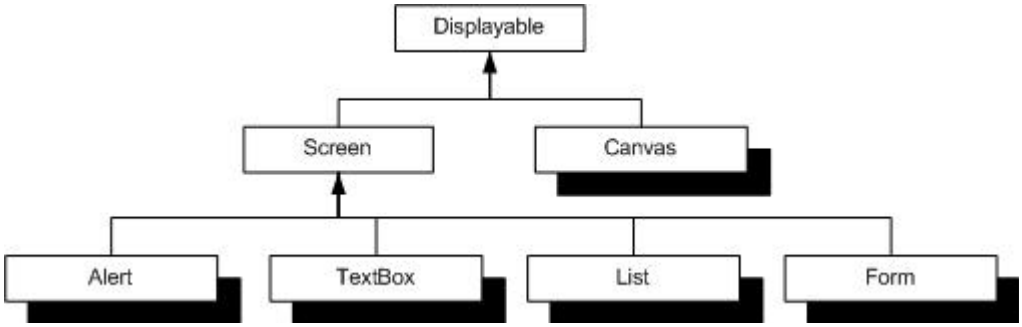
- Ketika MIDlet pertama kali diciptakan dan diinisialisasi, MIDlet berada dalam status *pause.*
- Jika terjadi kesalahan selama konstruksi MIDlet, MIDlet akan berpindah ke status *destroyed* dan MIDlet batal diciptakan dengan memanggil fungsi *destroyApp().*
- Saat MIDlet dijalankan, MIDlet akan berada pada status *active* dan memanggil fungsi *startApp()*.
- Jika saat dijalankan MIDlet dihentikan sementara, maka MIDlet berada pada status *pause* dan memanggil fungsi *pauseApp().*

2.3.2 Antarmuka Pemakai

Untuk membuat suatu antarmuka bagi pemakai, program MIDlet harus mengimpor paket *javac.microedition.lcdui.* Kelas yang dipergunakan untuk membuat dan memanipulasi antarmuka tersebut adalah kelas yang diturunkan dari kelas

Displayable. Melalui kelas-kelas inilah sebuah aplikasi dapat berinteraksi dengan pemakai.

Pada MIDP, antarmuka terdiri dari API tingkat tinggi (*High-level*) dan API tingkat rendah (*Low-level*). API tingkat tinggi berbasis pada kelas *Screen*, sedangkan API tingkat rendah berbasis pada kelas *Canvas*. Diagram hirarki dari kelas *Displayable* [2] dapat dilihat pada gambar 2.2.



Gambar 2.2 Hirarki kelas *Displayable*

Dari gambar 2.2 dapat dilihat bahwa kelas *Screen* terdiri dari beberapa kelas, yaitu:

- *Alert*, merupakan kelas yang menyediakan informasi kecil kepada pemakai yang ditampilkan ke layar sebelum kemudian berpindah ke objek lain. Biasanya digunakan untuk menampilkan informasi kesalahan *(error).*
- *TextBox*, merupakan kelas yang menyediakan media untuk menerima masukan berupa teks.
- *List*, merupakan kelas yang menyediakan masukan pilihan *(multiple choice)* pada layar.
- *Form*, merupakan kelas yang menyediakan fasilitas untuk menampung beberapa item dalam satu layar, seperti gambar *(images),* kolom tanggal *(datefield),* kolom teks *(textfield), gauge* dan daftar pilihan *(choice group).*

2.3.3 Event

Ketika terjadi interaksi antara pengguna dan sebuah perangkat, misalnya ponsel, maka akan dihasilkan suatu *event*, seperti memilih suatu menu atau melakukan masukan pada layar. Kelas yang bertugas untuk menerima event dari pengguna harus mengimplementasikan antarmuka *CommandListener*. Fungsi-fungsi yang harus digunakan antara lain :

- *command(),*digunakan untuk menciptakan tombol perintah *(command)* baru.
- *addCommand(),* digunakan untuk mendefinisikan sebuah tombol perintah ke dalam kelas *Displayable.*
- *setCommandListener(),* digunakan untuk mendengarkan aksi yang dilakukan terhadap sebuah tombol perintah.
- *commandAction(),* digunakan untuk menentukan aksi yang akan dikerjakan oleh sebuah tombol perintah.

2.3.4 Pemaketan aplikasi MIDlet

Hasil dari kompilasi program sumber Java adalah satu atau lebih berkas *bytecode* yang dikenali dengan akhiran "*.class". Pada aplikasi MIDlet, semua berkas *byte-code* dipaketkan menjadi suatu berkas terkompresi yang disebut *Java Archive* (JAR) yang dikenali dengan ekstensi "*.jar".

Selain berkas JAR, terdapat juga berkas *Java Application Descripter* (JAD) yang berekstensi "*.jad", yaitu berkas yang berisi informasi mengenai suatu berkas JAR. Kedua berkas inilah yang harus di-*upload* ke perangkat ponsel agar aplikasi dapat dijalankan. Untuk meng-*upload* berkas tersebut ke perangkat ponsel dapat dilakukan dengan cara transfer data antara komputer dengan perangkat komunikasi bergerak melalui media sinar infra merah, koneksi kabel data atau *bluetooth*.

2.4 *Wireless Messaging* API

Wireless Messaging API (WMA) adalah teknologi yang dimiliki oleh MIDP 2 yang dapat digunakan untuk mengirim dan menerima *Short Message Service* (SMS).

Dua paket utama yang harus diimplementasikan untuk mengembangkan aplikasi WMA adalah :

- *javax.wireless.messaging*, paket ini menyediakan antar muka yang member fasilitas untuk mengirim dan menerima pesan
- *javax.microedition.io*, paket ini mengatur masalah jaringan atau konektifitas pada aplikasi *Wireless messaging*.

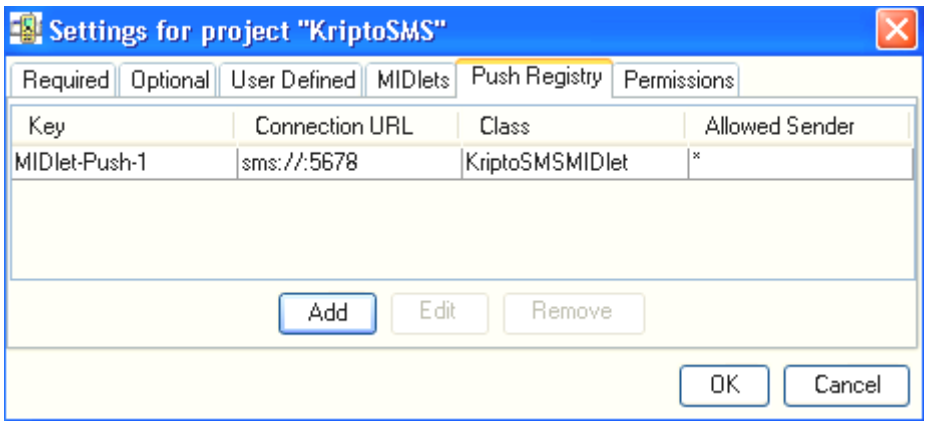
Dalam pengembangan aplikasi WMA lebih lanjut terdapat dua teknologi J2ME yang cukup penting, yaitu fungsi *push (Push Technology)* dan *Over The Air* (OTA).

2.4.1 Push Technology

Fungsi *Push* mulai diterapkan pada MIDP versi 2.0. Teknologi *push* memiliki kemampuan untuk menjalankan respon pada emulator jika ada pesan masuk. Jenis koneksi yang diijinkan, yaitu SMS, datagram dan *socket*.

Untuk menggunakan teknologi *push* sebelumnya perlu dilakukan registrasi *push (push registry)*. Terdapat 2 cara untuk melakukan registrasi, yaitu :

1. Secara statis, yaitu dengan cara mendefinisikan pada berkas JAD (pada menu *Project* → *Setting*).



Gambar 2.3 Registrasi *push* secara statis.

Dimana :

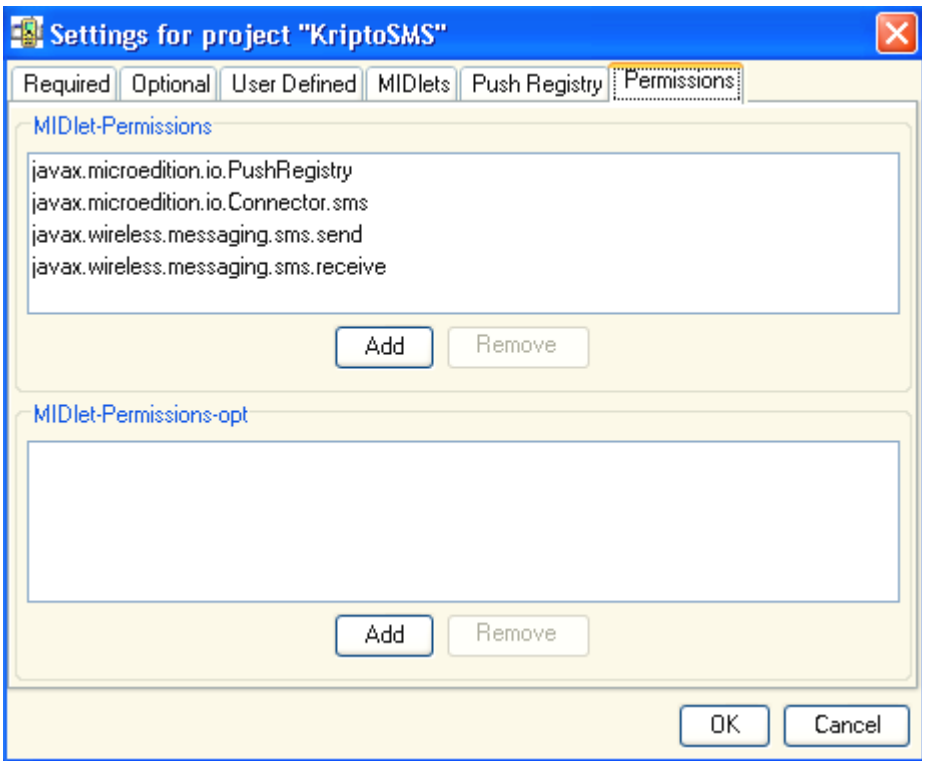
- *Connection URL* merupakan tipe koneksi yang masuk melalui sebuah port.
- *Class* merupakan nama kelas yang meimplementasikan MIDlet.

- *Allow Sender* merupakan pengiriman pesan yang akan direspon, tanda "*" berarti dapat menerima pesan dari semua pengirim

2. Secara dinamis, yaitu dengan memanggil fungsi *registerConnection()* yang terdapat pada kelas *PushRegistry*. Hal ini dilakukan melalu kode program, bentuk umumnya :

   registerConnection(String connection, String midlet, String filter)

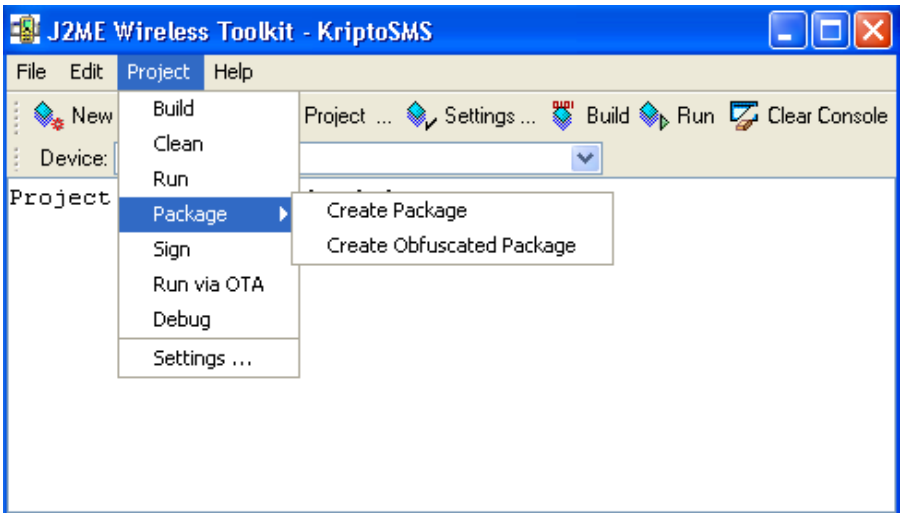
Setelah melakukan registrasi *push* perlu didefinisikan *permission*. Untuk aplikasi SMS paling tidak harus diatur seperti pada gambar



Gambar 2.4 Jendela untuk Setting Permissions

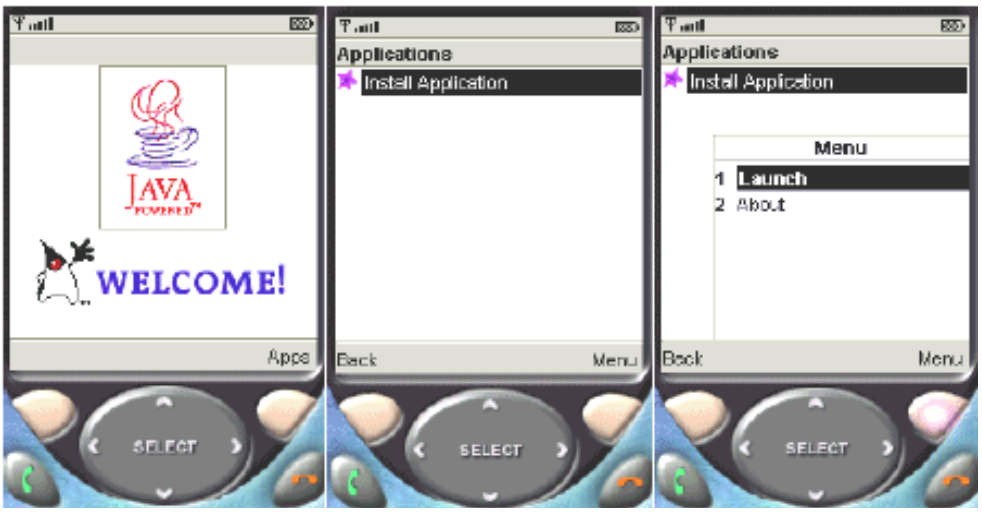
2.4.2 Over The Air (OTA)

Over The Air (OTA) digunakan untuk menginstal dan menjalankan aplikasi MIDlet. Sehingga melalui OTA dapat dilakukan simulasi pengiriman dan penerimaan pesan SMS. Sebelum menginstal MIDlet, hal pertama yang harus dilakukan adalah membuat paket (berkas JAR) untuk aplikasi.



Gambar 2.5 Pemaketan Aplikasi MIDlet

Setelah itu dilakukan penginstalan dengan memilih menu *Run via OTA*, maka akan muncul layar AMS kemudian pilih *Apps* → *Install Aplication* → *Launch.*, Selanjutnya tinggal mengikuti langkah-langkah penginstalan yang diberikan.



Gambar 2.6 Tampilan Awal Layar AMS

2.5 Kriptografi

Kriptografi adalah ilmu yang mempelajari tentang cara menjaga keamanan suatu pesan atau informasi. Pesan atau informasi dapat dikategorikan ke dalam dua jenis, yaitu pesan yang dapat dibaca dengan mudah *(plaintext)* dan pesan yang tidak mudah dibaca *(ciphertext).*

Untuk melakukan kriptografi digunakan algoritma kriptografi. Algoritma kriptografi terdiri dari dua bagian, yaitu fungsi enkripsi dan dekripsi. Enkripsi adalah proses untuk mengubah *plaintext* menjadi *ciphertext*, sedangkan dekripsi adalah kebalikannya yaitu mengubah *ciphertext* menjadi *plaintext*. Terdapat dua jenis algoritma kriptografi berdasar jenis kuncinya[1], yaitu :

1. Algoritma Simetri, adalah algoritma yang menggunakan kunci enkripsi yang sama dengan kunci dekripsinya. Algoritma standar yang menggunakan prinsip kunci simetri antara lain OTP, DES, RC2, RC4, RC5, RC6, IDEA, Twofish, Blowfish, dan lain lain.
2. Algoritma Asimetri, adalah algoritma yang kunci untuk enkripsi dan dekripsinya jauh berbeda. Algoritma standar yang termasuk algoritma asimetri adalah ECC, LUC, RSA, EI, Gamal dan DH.

Salah satu teknik enkripsi menggunakan kunci simetri adalah teknik subtitusi, yaitu mengganti setiap karakter *plaintext* dengan karakter lain. Terdapat empat cara dalam menggunakan teknik subtitusi[1], yaitu :

1. Monoalphabet, dimana setiap karakter *ciphertext* mengganti satu macam karakter *plaintext* tertentu.
2. Polialphabet, dimana setiap karakter *ciphertext* mengganti lebih dari satu macam karakter *plaintext*.
3. Monograf/unilateral, dimana satu enkripsi dilakukan terhadap satu karakter *plaintext*.
4. Poligraf/multilateral, dimana satu enkripsi dilakukan terhadap lebih dari satu karakter *plaintext*.

2.5.1 Viginere cipher

Vigenere cipher merupakan salah satu algoritma klasik dengan teknik substitusi. Nama vigenere diambil dari seorang yang bernama Blaise de Vigenere.

Vigenere cipher menggunakan suatu kunci yang memiliki panjang tertentu. Panjang kunci tersebut bisa lebih pendek ataupun sama dengan panjang plainteks. Jika panjang kunci kurang dari panjang plainteks, maka kunci yang tersebut akan

diulang secara periodik hingga panjang kunci tersebut sama dengan panjang plainteksnya.

Algoritma enkripsi vigenere cipher :

Ci = (Pi + Ki) mod 26

Algoritma dekripsi vigenere cipher :

Pi = (Ci – Ki) mod 26

Dimana :

Ci = nilai desimal karakter *ciphertext* ke-i

Pi = nilai desimal karakter *plaintext* ke-i

Ki = nilai desimal karakter kunci ke-i

Sebagai contoh, jika *plaintext* adalah THEBEAUTYANDTHEBEAST dan kunci adalah ABC maka proses enkripsi yang terjadi adalah sebagai berikut :

Plaintext : THEBEAUTYANDTHEBEAST

Kunci : ABCABCABCABCABCABCAB

Chipertext : TIGBFCUUAAOFTIGBFCSU

Pada contoh di atas kata kunci ABC diulang sedemikian rupa hingga panjang kunci sama dengan panjang plainteksnya. Kemudian setelah panjang kunci sama dengan panjang plainteks, proses enkripsi dilakukan dengan melakukan menggeser setiap huruf pada plainteks sesuai dengan huruf kunci yang bersesuaian dengan huruf plainteks tersebut. Pada contoh di atas plainteks huruf pertama adalah T akan dilakukan pergeseran huruf dengan kunci Ki=0 (kunci huruf pertama adalah A yang memiliki Ki=0) menjadi T. Huruf kedua pada plainteks adalah H akan dilakukan pergeseran huruf dengan kunci Ki=1 (kunci huruf kedua adalah B yang memiliki Ki=1) menjadi I. Begitu seterusnya dilakukan pergeseran sesuai dengan kunci pada tiap huruf hingga semua plainteks telah terenkripsi menjadi ciphertext.

2.6 Software Development Life Cycle

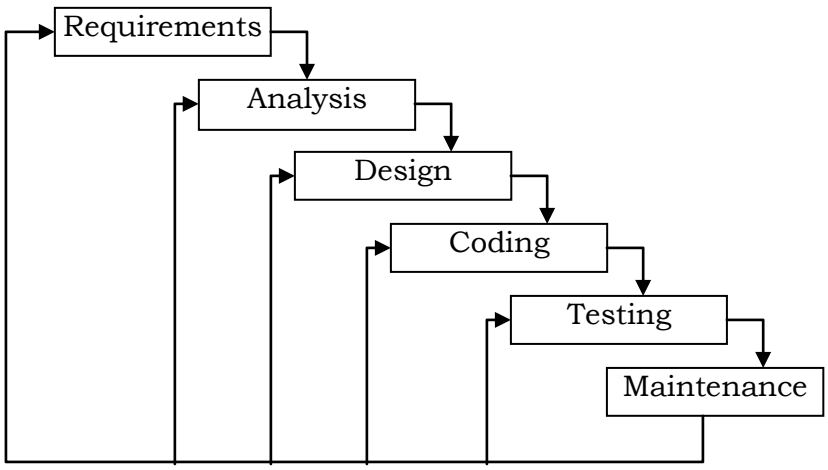
Pada setiap pengembangan perangkat lunak tidak akan terlepas dari sebuah *Software Development Life Cycle* (SDLC). SDLC merupakan sebuah siklus pengembangan perangkat lunak yang terdiri dari beberapa tahapan. Tahapan-tahapan tersebut adalah *requirements* (analisis kebutuhan), *analysis* (analisis sistem), *design* (perancangan), *coding*/*implementation* (implementasi), *testing* (pengujian), dan

maintenance (perawatan). Dalam SDLC terdapat banyak metode yang dapat dipakai untuk mengembangkan sebuah perangkat lunak misal *Waterfall*, Spiral, *Rapid Application Development* (RAD), dsb. Metode-metode tersebut akan dijelaskan pada sub bab berikut.

2.6.1 Metode Waterfall

Dalam perancangan aplikasi pada tugas akhir ini penulis menggunakan metode *Waterfall*. Metode *Waterfall* adalah metode yang menyarankan sebuah pendekatan yang sistematis dan sekuensial melalui tahapan-tahapan yang ada pada SDLC untuk membangun sebuah perangkat lunak[3].

Gambar menjelaskan bahwa metode *Waterfall* menekankan pada sebuah keterurutan dalam proses pengembangan perangkat lunak. Metode ini adalah sebuah metode yang tepat untuk membangun sebuah perangkat lunak yang tidak terlalu besar dan sumber daya manusia yang terlibat dalam jumlah yang terbatas.



Gambar 2.7 Diagram Waterfall Model

Dari gambar 2.7 dapat dilihat bahwa tahapan pada metode *Waterfall* diawali oleh tahap analisis kebutuhan yang merupakan tahap awal pembangunan sebuah perangkat lunak. Tahap ini didefinisikan sebagai sebuah tahap yang menghasilkan sebuah kondisi yang diperlukan oleh pengguna untuk menyelesaikan permasalahan ataupun mencapai sebuah tujuan. Tahap ini bertujuan untuk megumpulkan kebutuhan-kebutuhan pengguna dan kemudian mentransformasikan ke dalam sebuah deskripsi yang jelas dan lengkap.

Tahapan kedua adalah tahap analisis sistem yang bertujuan untuk menjabarkan segala sesuatu yang nantinya akan ditangani oleh perangkat lunak. Tahapan ini adalah tahapan dimana pemodelan merupakan sebuah representasi dari *object* di dunia nyata. Untuk memahami sifat perangkat lunak yang akan dibangun, analis harus memahami domain informasi, dan tingkah laku yang diperlukan.

Tahap ketiga adalah tahap perancangan perangkat lunak yang merupakan proses multi langkah dan berfokus pada beberapa atribut perangkat lunak yang berbeda yaitu struktur data, arsitektur perangkat lunak, dan detil algoritma. Proses ini menerjemahkan kebutuhan ke dalam sebuah model perangkat lunak yang dapat diperkirakan kualitasnya sebelum dimulainya tahap implementasi.

Tahap implementasi adalah tahap yang mengkonversi apa yang telah dirancang sebelumnya ke dalam sebuah bahasa yang dimengerti komputer. Kemudian komputer akan menjalankan fungsi-fungsi yang telah didefinisikan sehingga mampu memberikan layanan-layanan kepada penggunanya.

Tahap selanjutnya adalah tahap pengujian. Terdapat dua metode pengujian perangkat lunak yang umum digunakan, yaitu metode *black-box* dan *white-box*. Pengujian dengan metode *black-box* merupakan pengujian yang menekankan pada fungsionalitas dari sebuah perangkat lunak tanpa harus mengetahui bagaimana struktur di dalam perangkat lunak tersebut. Sebuah perangkat lunak yang diuji menggunakan metode *black-box* dikatakan berhasil jika fungsi-fungsi yang ada telah memenuhi spesifikasi kebutuhan yang telah dibuat sebelumnya. Sedangkan metode *white-box* menguji struktur internal perangkat lunak dengan melakukan pengujian pada algoritma yang digunakan oleh perangkat lunak.

Tahap akhir dari metode *Waterfall* adalah tahap perawatan. Tahap ini dapat diartikan sebagai tahap penggunaan perangkat lunak yang disertai dengan perawatan dan perbaikan. Perawatan dan perbaikan suatu perangkat lunak diperlukan, termasuk di dalamnya adalah pengembangan, karena dalam prakteknya ketika perangkat lunak tersebut digunakan terkadang masih terdapat kekurangan ataupun penambahan fitur-fitur baru yang dirasa perlu.

2.7 J2ME *Wireless Toolkit*

J2ME *Wireless Toolkit* adalah alat bantu berupa perangkat lunak untuk melakukan kompilasi dan pengujian aplikasi J2ME.

2.7.1 *KToolbar*

KToolbar merupakan bagian utama dari aplikasi untuk pengembangan Java MIDP atau lebih dikenal dengan istilah MIDlet. *KToolbar* merupakan lingkungan pengembangan minimal yang disediakan oleh J2ME *Wireless Toolkit* untuk pengembangan MIDlet.

Untuk melakukan pengembangan aplikasi MIDlet melaui *KToolbar*, hal pertama yang harus dilakukan adalah membuat proyek. Proyek ini secara fisik akan diletakkan pada **C:\WTK20\apps\{nama-proyek}** dengan asumsi J2ME *Wireless Toolkit* diinstal pada C:\WTK20\. Proyek yang dibuat terdiri atas direktori-direktori berikut:

- *Src*, adalah direktori yang berisi berkas-berkas kode java
- *Bin*, adalah direktori yang berisi berkas-berkas JAD, JAR dan manifest
- *Res*, adalah direktori yang berisi berkas-berkas sumber, misalnya gambar
- *Lib*, adalah direktori yang berisi berkas-berkas kelas java yang diperlukan dalam format gambar atau zip

BAB III

ANALISIS KEBUTUHAN DAN PERANCANGAN SISTEM

3.1 Analisis Kebutuhan

Aplikasi KriptoSMS ini digunakan untuk mengirim dan menerima pesan. KriptoSMS akan mengenkripsi pesan yang akan dikirim menjadi *ciphertext* dan KriptoSMS akan mendekripsi pesan masuk berupa *ciphertext* menjadi *plaintext*.

Dalam membangun aplikasi KriptoSMS, diperlukan batasan yang jelas sebagai tujuan utamanya agar tidak keluar dari rencana yang telah ditetapkan. Beberapa kebutuhan sistem yang akan didefinisikan antara lain :

1. Memiliki kemampuan untuk mengirimkan dan menerima pesan.
2. Memiliki kemampuan untuk mengenkripsi pesan dan memberikan header KriptoSMS pada pesan yang telah di enkripsi.
3. Memiliki kemampuan untuk mendekripsi pesan yang mengandung header KriptoSMS.
4. Menampilkan output berupa pesan asli yang telah di dekripsi.

Tidak semua telepon selular dapat menjalankan aplikasi KriptoSMS. Berikut spesifikasi dari telepon selular agar dapat menjalankan aplikasi KriptoSMS :

- Mempunyai *Java Runtime Environment*.
- Mempunyai *MicroEdition-profile* MIDP 2.0.
- Mempunyai *MicroEdition-configuration* CLDC 1.0.

3.1.1 *Software Requirement Spesification* (SRS)

Spesifikasi kebutuhan perangkat lunak yang akan dikembangkan di KriptoSMS dapat dilihat di table 3.1 pada halaman 20.

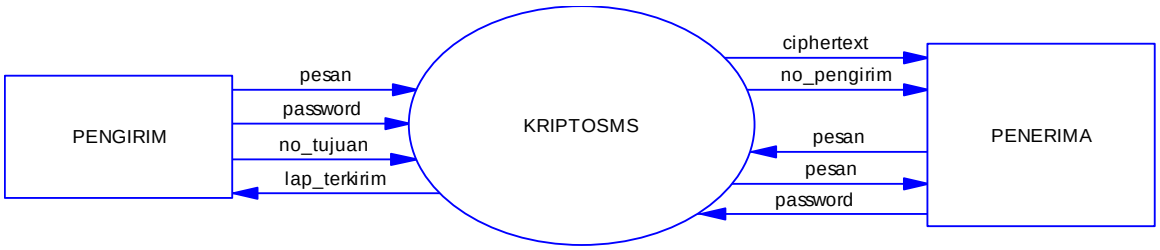
Tabel 3.1 Spesifikasi kebutuhan perangkat lunak

No	SRS ID	Deskripsi
1	SRS – KSMS – 01	Menerima input berupa no. tujuan, pesan dan password
2	SRS – KSMS – 02	Mengubah pesan dan password ke kode ASCII
3	SRS – KSMS – 03	Mengenkripsi pesan menggunakan *key* (password) yang telah diinputkan
4	SRS – KSMS – 04	Memberikan header KRIPTOSMS pada pesan yang telah dienkripsi (ciphertext)
5	SRS – KSMS – 05	Mengirimkan pesan dan memberikan laporan kepada pengirim bahwa pesan telah dienkripsi dan dikirim ke no.tujuan
6	SRS – KSMS – 06	Memeriksa pesan apakah memiliki header KRIPTOSMS
7	SRS – KSMS – 07	Memberikan no.pengirim kepada penerima
8	SRS – KSMS – 08	Mendekripsi ciphertext menggunakan *key* (password) yang telah diinputkan
9	SRS – KSMS – 9	Mengubah pesan dari kode ASCII ke karakter
10	SRS – KSMS – 10	Menampilkan pesan yang telah di dekripsi kepada penerima

3.2 Pemodelan Fungsional

3.2.1 *Data Context Diagram*

Pendefinisian dengan *Data Context Diagram* (DCD) atau DFD level 0 memberikan data yang mengalir antara sistem dengan lingkungan yang digambarkan secara global. DCD dapat dilihat pada gambar 3.1 pada halaman 21.

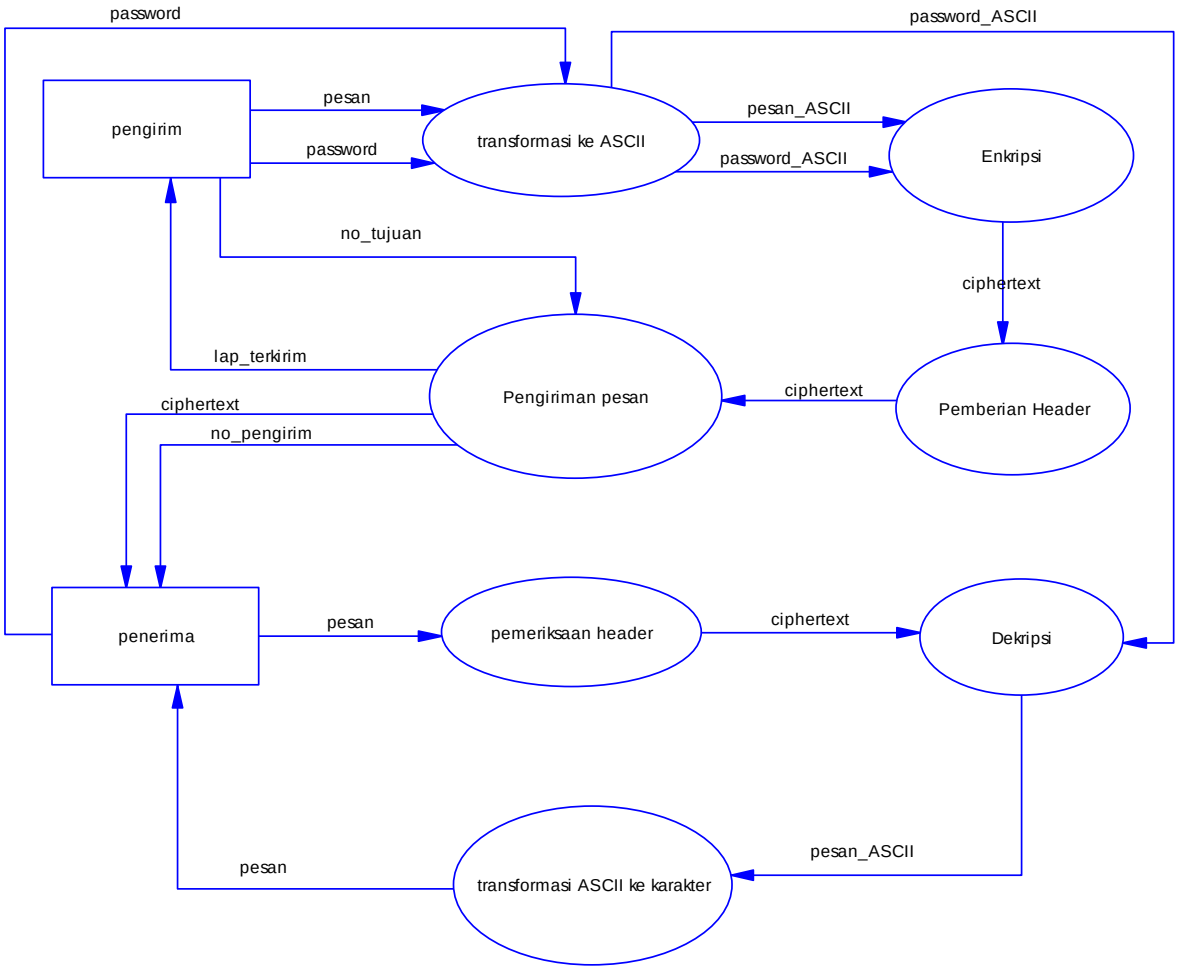


Gambar 3.1 DCD Aplikasi Perangkat Lunak KriptoSMS

Pada gambar 3.1 menjelaskan bahwa pengirim meng*input*kan pesan, *password* dan nomor tujuan kepada sistem. Sistem memberikan *output* berupa laporan pesan terkirim kepada pengirim. Penerima meng*input*kan *password* kepada sistem dan sistem memberikan *output* berupa pesan dan nomor pengirim kepada penerima.

3.2.2 *Data Flow Diagram*

Dari *Data Context Diagram* yang diperlihatkan pada gambar 3.1, dapat dibuat diagram alir data (*Data Flow Diagram*) yang merupakan penjelasan lebih rinci dari DCD pada setiap tahapan. DFD dapat dilihat pada gambar 3.2 pada halaman 22.



Gambar 3.2 DFD level 1 Aplikasi Perangkat Lunak KriptoSMS

Gambar 3.2 merupakan DFD *level* 1 dari aplikasi perangkat lunak KriptoSMS. Pada DFD *level* 1, sistem akan dipecah menjadi proses-proses kecil sehingga dapat menjelaskan proses-proses dan arus data yang mengalir dalam sistem.

Proses-proses yang terdapat dalam DFD *level* 1 adalah :

1. Transformasi kode ASCII

   Proses ini melakukan transformasi pesan dan password ke dalam kode ASCII.

2. Enkripsi

   Proses ini melakukan pengenkripsian pesan dengan metode enkripsi vigenere cipher dengan *key* sesuai dengan *password* yang di*input*kan.

3. Pemberian Header

   Pesan yang telah dienkripsi menjadi *ciphertext* akan diberikan *header* sebagai penanda bahwa pesan tersebut adalah pesan KriptoSMS.

4. Pengiriman pesan

   Proses ini melakukan pengiriman pesan yang telah dienkripsi dan memberikan laporan pengiriman kepada pengirim bahwa pesan telah dienkripsi dan dikirimkan ke nomor yang telah di*input*kan.

5. Pemeriksaan Header

   Pada proses ini pesan yang masuk akan diperiksa apakah memiliki *header* KriptoSMS. Apabila pesan memiliki *header* maka pesan akan diteruskan ke proses selanjutnya yaitu dekripsi. Jika pesan tidak memiliki *header* maka sistem akan mengabaikan pesan tersebut.

6. Dekripsi

   Proses ini melakukan pendekripsian pesan sesuai dengan *password* yang di*input*kan. Apabila *password* benar maka *ciphertext* akan menjadi pesan asli. Jika *password* salah pesan akan tetap didekripsi akan tetapi pesan yang didapat bukan pesan asli. Ini dikarenakan *key* yang dipakai untuk mendekripsi *ciphertext* salah.

3.3 Perancangan Sistem

Proses perancangan merupakan deskripsi dari kebutuhan yang direpresentasikan ke dalam perangkat lunak sehingga dapat diperkirakan kualitasnya sebelum dimulai pembuatan *code* atau *coding*. Aplikasi KriptoSMS ini terdiri dari 7 kelas yaitu : KriptoSMSMIDlet, SendScreen, ReceiveScreen, MessageCodec, MyKripto, ReportScreen dan ErrorScreen. Proses *coding* dibuat menggunakan aplikasi *Jcreator*. Secara garis besar, fungsi dari masing-masing kelas adalah sebagai berikut :

1. KriptoSMSMIDlet, kelas ini adalah kelas utama yang mengatur pengiriman dan penerimaan pesan. Kelas ini yang mengatur tampilan utama dari aplikasi KriptoSMS.
2. SendScreen, kelas ini digunakan untuk mengatur tampilan pengiriman pesan dan mengirimkan pesan ke nomor tujuan yang telah di*input*kan.

3. ReceiveScreen, kelas ini digunakan untuk mengatur tampilan penerimaan pesan.
4. MessageCodec, kelas ini digunakan untuk melakukan enkripsi dan dekripsi pesan.
5. MyKripto, kelas ini berisi algoritma untuk enkripsi dan dekripsi pesan.
6. ReportScreen, kelas ini digunakan untuk mengatur tampilan laporan.
7. ErrorScreen, kelas ini digunakan untuk mengatur tampilan pesan kesalahan

3.3.1 Perancangan Fungsi

3.3.1.1 Fungsi *Wireless Messaging* API

a. Membuat koneksi

Agar proses pengiriman dan penerimaan dapat berlangsung perlu dibuat fungsi untuk membuat koneksi. Fungsi untuk membuat koneksi seperti di bawah ini.

```
. . . . .
String connection ="sms://:" + port;
        try
        {
              conn = (MessageConnection)
                    Connector.open(connection);
              conn.setMessageListener(this);
        }
. . . . .
```

b. Mengirimkan pesan

Untuk mengirimkan pesan dibuat fungsi *sendMessage()*.

```
. . . . .
void sendMessage (String number, String plainText,
String password)
  {
        if (conn != null)
        {
              BinaryMessage        binarySMS        =
(BinaryMessage)conn.newMessage(MessageConnection.BIN
ARY_MESSAGE);
              String       address       =       new
StringBuffer("sms://").append(number).append(":").ap
pend(port).toString();
              binarySMS.setAddress(address);
```

```
. . . . .
```

c. Penerimaan pesan

Jika ada pesan yang datang maka fungsi yang digunakan adalah

```
. . . . .
public void notifyIncomingMessage(MessageConnection
conn)
   {
         if (conn==this.conn)
         {
               try
               {
                     BinaryMessage
incomingMessage=(BinaryMessage)conn.receive();
                     String                  message=new
String(incomingMessage.getPayloadData());
                     if
(message.startsWith("KRIPTOSMS: "))
                     {
                           ReceiveScreen handler = new
ReceiveScreen(this,incomingMessage);

   Display.getDisplay(this).setCurrent(handler);
                     }
                     else
                           ErrorScreen.showError("pesan
yang diterima bukan"+"\n kriptosms", displayable);
                     }
               catch (IOException e)
               {
                     ErrorScreen.showError("gagal
menerima pesan karena "+e.getMessage(),displayable);
               }

         }
   }
. . . . .
```

d. Menutup koneksi

Untuk menutup koneksi digunakan fungsi *close()*.

```
. . . . .
public void destroyApp(boolean unconditional)
   {
         try
         {
```

```
                    if (conn!=null)
                    {
                          conn.close();
                    }
              }
              catch (IOException e)
              {
              }
        }
    . . . . .
```

3.3.1.2 Fungsi enkripsi dan deskripsi pesan

a. Enkripsi

Untuk mengenkripsi pesan digunakan fungsi sebagai berikut

```
. . . . .
public byte[] enkrip(byte[] input, byte[] kunci)
     {
          byte[] output = new byte[input.length];
          byte buffer, bufferkunci;
          for(int i = 0; i<input.length; i++)
          {
               buffer = input[i];
               bufferkunci = kunci[i%kunci.length];

               buffer+=bufferkunci;
               if(buffer>=256) buffer-=256;
               output[i]=buffer;
          }
          return output;
     }
. . . . .
```

b. Dekripsi

Fungsi untuk mendekripsi pesan sebagai berikut

```
. . . . .
public byte[] dekrip(byte[] input, byte[] kunci)
     {
          byte[] output = new byte[input.length];
          byte buffer, bufferkunci;
          for(int i=0;i<input.length;i++)
          {
               buffer = input[i];
               bufferkunci=kunci[i%kunci.length];
```

```
                    if(buffer<bufferkunci) buffer+=256;

                    buffer-=bufferkunci;
                    output[i]=buffer;
               }
               return output;
          }
. . . . .
```

c. Memberikan *header* pada pesan

Plaintext dirubah menjadi *ciphertext* dan diberikan *header* tambahan untuk menandakan bahwa pesan itu adalah pesan yang telah dienkripsi. Fungsi yang digunakan

```
. . . . .
byte[] cipherText = kriptoku.enkrip(content, key);
          StringBuffer       buffer       =       new
StringBuffer("KRIPTOSMS: ");
          buffer.append(new String(cipherText));
          cipherText = buffer.toString().getBytes();
          return cipherText;
. . . . .
```

d. Menyimpan pesan ke dalam variabel *array* dari *byte*

Pesan akan disimpan ke dalam variabel *array* dari *byte* (kode ASCII) untuk di enkripsi. Fungsi yang digunakan

```
. . . . .
static byte[] encodeMessage(String plainText, String
password) throws Exception
     {
          byte content[] = plainText.getBytes();
          byte key[] = password.getBytes();
. . . . .
```

e. Memeriksa *header* dalam pesan

Pesan yang masuk akan diperiksa apakah memiliki *header* KriptoSMS atau tidak di dalamnya. Fungsi yang digunakan

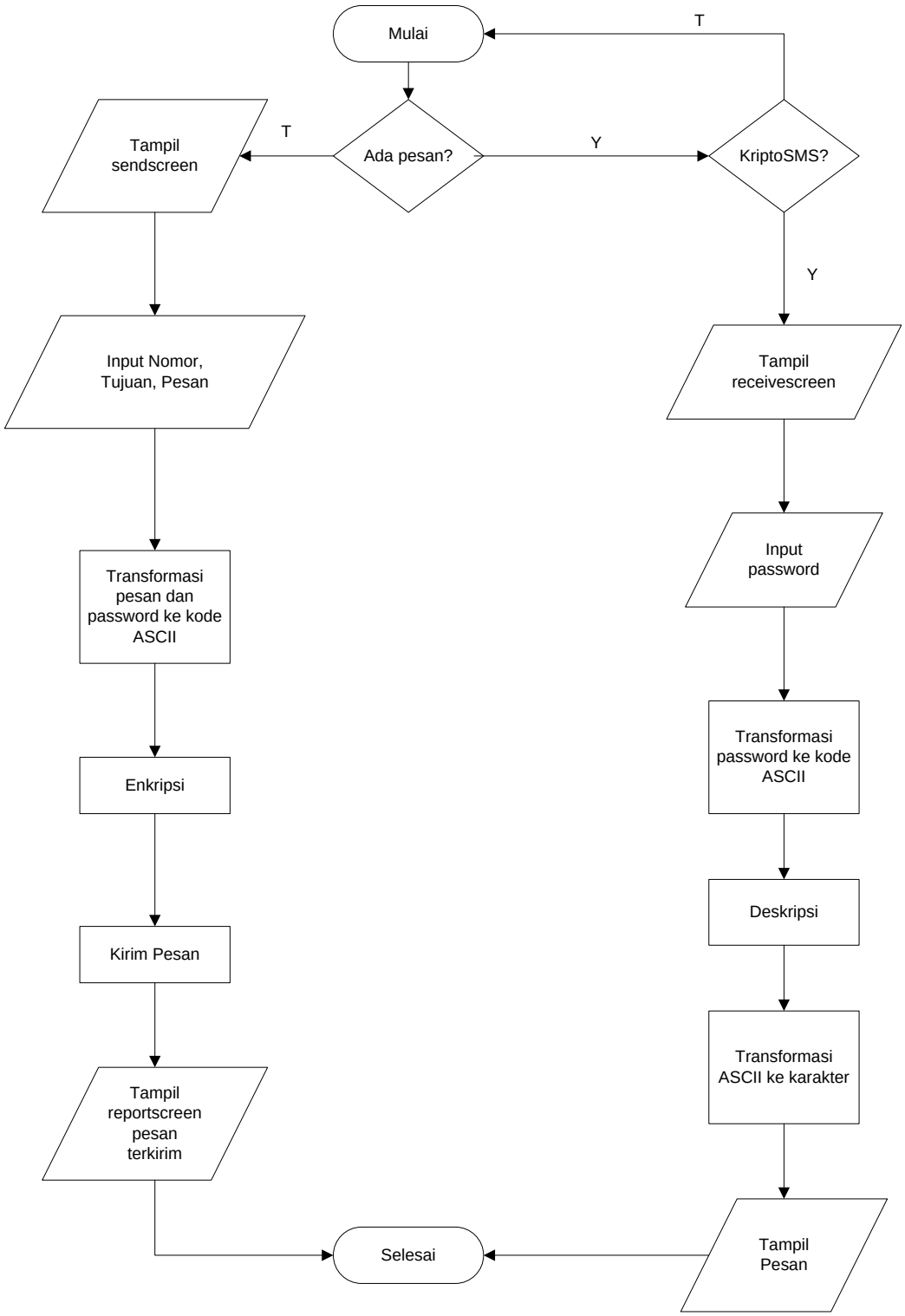
```
. . . . .
if (message.startsWith("KRIPTOSMS: "))
                         {
```

```
                              ReceiveScreen handler = new
ReceiveScreen(this,incomingMessage);

     Display.getDisplay(this).setCurrent(handler);
                         }
                         else
                              ErrorScreen.showError("pesan
yang diterima bukan"+"\n kriptosms", displayable);
. . . . .
```

3.3.2 Perancangan Antarmuka

Perancangan antarmuka adalah proses membuat perancangan *form-form* tampilan layar, selain itu dalam proses ini juga ditentukan bentuk dan isi dokumen sumber untuk memasukkan data yang kemudian diolah menjadi keluaran yang dapat digunakan oleh pengguna. Diagram alur aplikasi KriptoSMS dapat dilihat pada gambar 3.3 pada halaman 28.

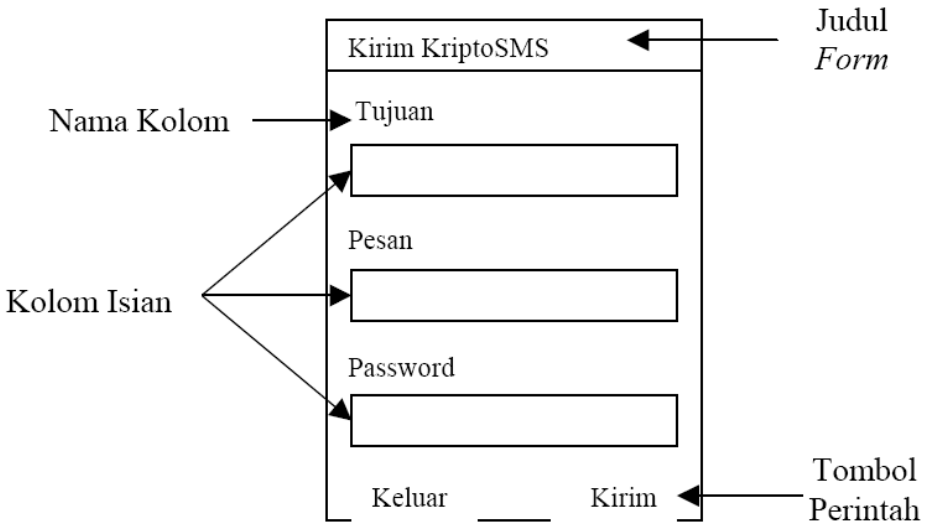


Gambar 3.3 *Flowchart* aplikasi KriptoSMS

Tampilan utama aplikasi dari KriptoSMS ini ada dua yaitu layar untuk mengirim pesan dan layar untuk menerima pesan.

a. *Form* pengiriman pesan

Perancangan *form* ini digunakan *user* untuk mengirimkan SMS.

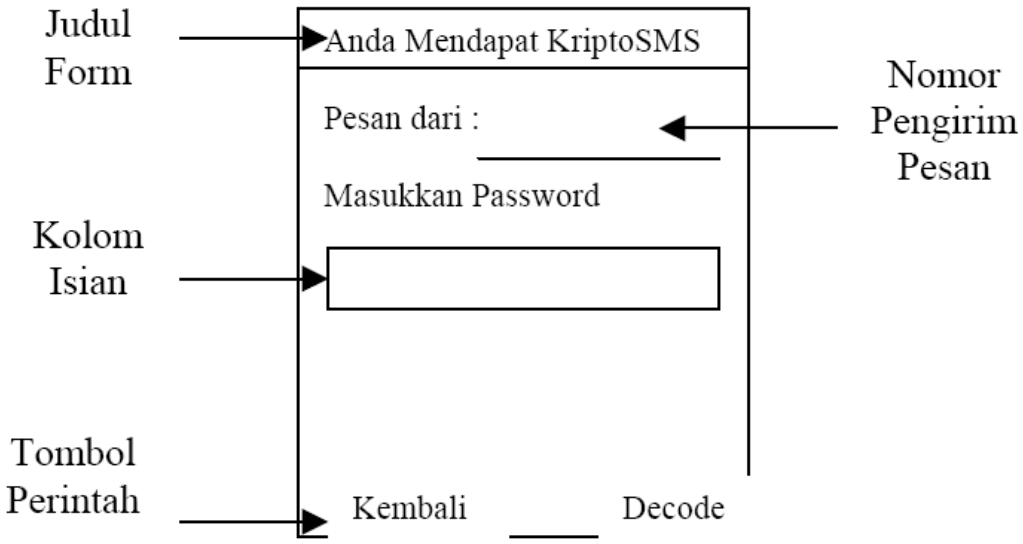


Gambar 3.4 *Form* kirim pesan

Pada *form* kirim pesan seperti pada gambar 3.4, *user* menginputkan nomor tujuan, pesan dan *password*. *Password* akan digunakan sebagai *key* untuk mengenkripsi pesan. Setelah pesan dienkripsi, pesan dikirimkan ke nomor tujuan yang telah di*input*kan. Setelah proses pengiriman selesai maka muncul *form* laporan yang berisi laporan pesan telah dienkripsi, ukuran pesan dan pesan telah terkirim ke nomor tujuan.

b. *Form* penerimaan pesan

Perancangan *form* ini digunakan *user* untuk menerima SMS yang berupa *ciphertext* dan untuk mendekripsikan *ciphertext* tersebut.



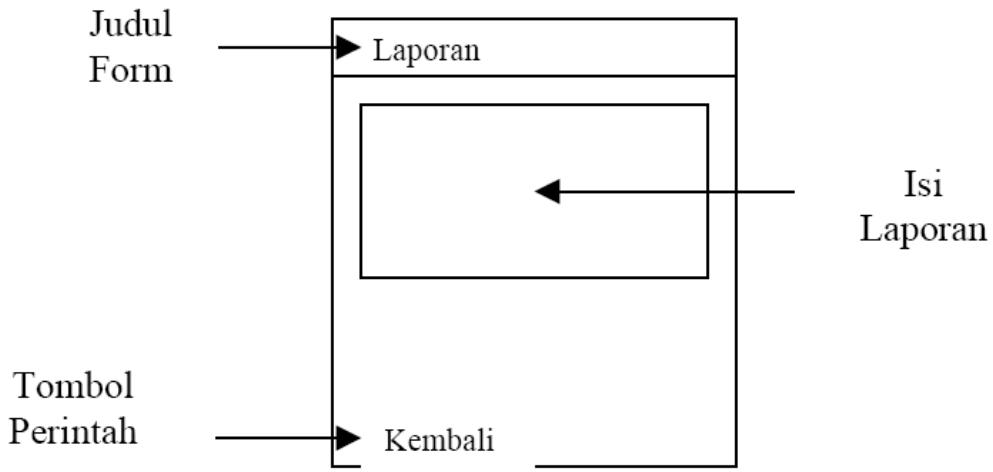
Gambar 3.5 *Form* penerimaan pesan

Setelah pesan masuk, maka muncul tampilan *form* penerimaan pesan seperti pada gambar 3.5, *user* diminta menginputkan *password* sebagai *key* untuk mendekripsi *ciphertext*. Jika password benar maka *ciphertext* akan didekripsi menjadi *plaintext* yang merupakan pesan asli.

Jika *password* yang di*input*kan salah, maka *ciphertext* akan tetap di dekripsi menjadi *plaintext* tetapi *plaintext* yang diterima bukan merupakan pesan asli. Ini dikarenakan *key* yang digunakan untuk mendekripsikan *ciphertext* salah.

c. *Form* Reportscreen

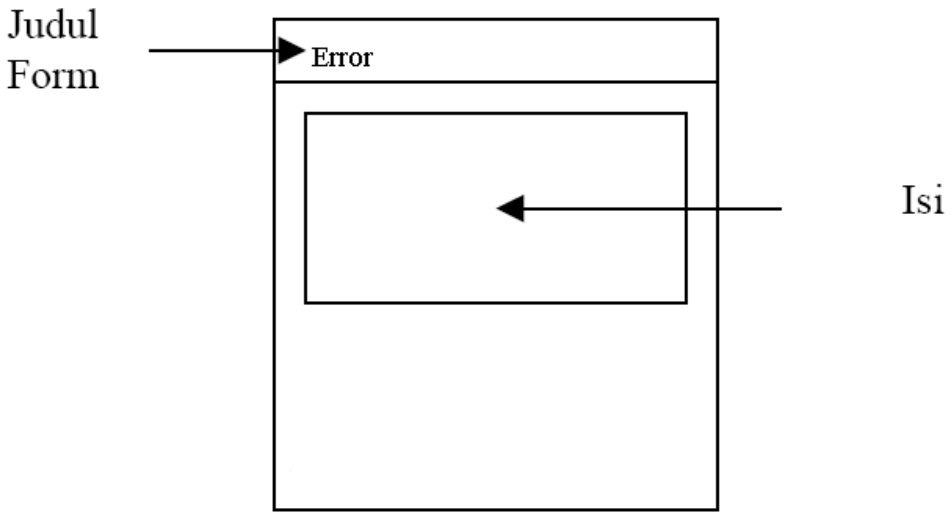
Perancangan *form* ini berfungsi untuk menampilkan laporan kepada *user* bahwa pesan telah dienkripsi atau didekripsi. Perancangan *form* Reportscreen dapat dilihat pada gambar 3.6 pada halaman 30.



Gambar 3.6 Reportscreen

d. *Form* Errorscreen

Perancangan *form* ini berfungsi untuk menampilkan error apabila *user* salah meng*input*kan data. Perancangan *form* Errorscreen dapat dilihat pada gambar 3.7.



Gambar 3.7 Errorscreen

BAB IV

IMPLEMENTASI, PENGUJIAN DAN ANALISIS HASIL

4.1 Implementasi

4.1.1 Implementasi Rancangan Antarmuka

Implementasi perancangan antarmuka, terbagi menjadi 4 bagian utama. Diantaranya adalah :

1. *Form* pengiriman SMS.

   Implementasinya dapat dilihat pada gambar 4.1.

Gambar 4.1 *Form* pengiriman SMS

2. *Form* penerimaan SMS.

   Implementasinya dapat dilihat pada gambar 4.2.

Gambar 4.2 *Form* penerimaan SMS

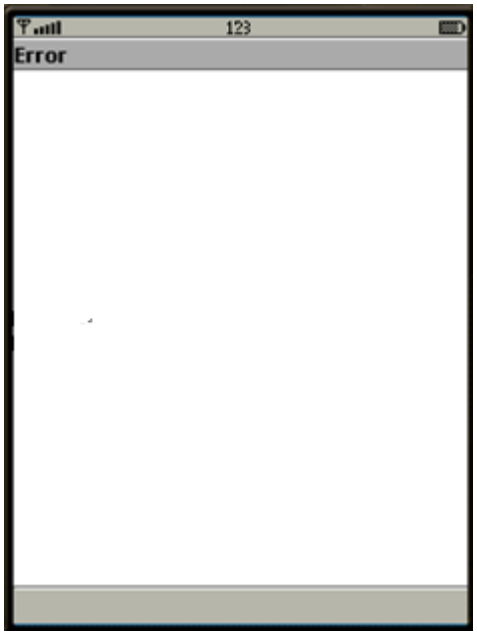
3. *Form* laporan

   Implementasinya dapat dilihat pada gambar 4.3.

Gambar 4.3 *Form* laporan pesan

4. *Form* Error

   Implementasinya dapat dilihat pada gambar 4.4.



Gambar 4.4 *Form* error

Form error akan keluar apabila user salah menginputkan data.

4.2 Pengujian

Pengujian merupakan tahap yang utama dalam pembuatan suatu aplikasi perangkat lunak. Hasil pengujian yang didapat, akan dijadikan sebagai tolak ukur dalam proses pengembangan selanjutnya. Pengujian ini dilakukan untuk mengetahui hasil yang didapat dari perangkat lunak yang telah dibuat.

4.2.1 Lingkungan Pengujian

Lingkungan pengujian merupakan penjelasan alat bantu apa saja yang digunakan dalam proses pembuatan aplikasi perangkat lunak KriptoSMS.

4.2.1.1 Perangkat Keras

Perangkat keras yang digunakan berupa *Home* PC dengan spesifikasi sebagai berikut :

1. Processor : AMD Athlon 64 X2 5600+, 2910 Mhz
2. RAM : 2048 Mb (PC3200 DDR2)
3. VGA : ATI Radeon 4670 (256 Mb)

4.2.1.2 Perangkat Lunak

Aplikasi yang telah dibuat, diujikan dalam lingkungan perangkat lunak dengan spesifikasi sebagai berikut :

1. *Operating System* yang digunakan adalah *Microsoft Windows XP SP2*.
2. *Software* yang digunakan untuk membuat aplikasi ini adalah *J2ME Wireless Toolkit 2.2*.
3. *JCreator* digunakan untuk pengetikan *source code* program.

4.2.2 Material Pengujian

Materi yang akan diujikan pada aplikasi KriptoSMS ini adalah sebagai berikut:

1. Pengiriman pesan

   Akan dilakukan pengujian pada proses pengiriman pesan apakah aplikasi KriptoSMS dapat mengenkripsi pesan dan mengirimkan pesan yang telah dienkripsi ke nomor yang dituju atau tidak.
2. Penerimaan pesan

   Akan dilakukan pengujian pada proses penerimaan pesan apakah aplikasi KriptoSMS dapat menerima sebuah pesan dan mendekripsikannya menjadi pesan asli.
3. Laporan

   Akan dilakukan pengujian apakah akan keluar sebuah laporan apabila pesan telah dienkripsi atau didekripsi.
4. *Error*

   Akan dilakukan pengujian apakah *error* akan keluar apabila *user* salah meng*input*kan sebuah data.

4.2.3 Pelaksanaan Pengujian

Pengujian aplikasi KriptoSMS ini menggunakan metode *blackbox*. Sesuai dengan material pengujian maka akan dilaksanakan pengujian sebagai berikut:

1. Pengiriman pesan

Proses ini berfungsi untuk mengenkripsi pesan dan mengirimkan pesan ke nomor tujuan. Langkah-langkah yang dilakukan *user* dalam proses ini adalah sebagai berikut :

1. *Input* pesan

   User diminta untuk meng*input*kan pesan yang akan dikirim. Pesan tersebut akan dienkripsi terlebih dahulu sebelum dikirimkan ke nomor tujuan.

2. *Input* nomor tujuan

   User diminta untuk meng*input*kan nomor tujuan kemana pesan akan dikirimkan.

3. *Input password*

   Password yang di*input*kan merupakan *key* yang akan dipakai untuk mengenkripsi pesan. Proses selengkapnya dapat dilihat pada gambar 4.5.

Gambar 4.5 Proses pengiriman pesan

2. Penerimaan pesan

Apabila ada pesan masuk maka tampilan layar akan berubah ke layar receivescreen. Pada proses ini *user* diminta untuk memasukkan *password* yang akan menjadi *key* untuk mendekripsikan pesan. Proses selengkapnya dapat dilihat pada gambar 4.6.

Gambar 4.6 Proses penerimaan pesan

3. Laporan

Laporan disini terbagi menjadi 2 yaitu laporan setelah proses pengiriman pesan dan laporan yang menampilkan pesan setelah proses pendekripsian pesan.

a. Laporan pengiriman pesan

Setelah pesan dienkripsi dan dikirimkan ke nomor tujuan maka akan ditampilkan laporan bahwa pesan telah dienkripsi dan dikirimkan ke nomor tujuan. Proses selengkapnya dapat dilihat pada gambar 4.7.

Gambar 4.7 Laporan pengiriman pesan

b. Laporan tampilan pesan telah didekripsi

Pesan yang masuk akan didekripsi memakai *password* yang di*input*kan oleh *user*. Jika *password* benar maka laporan yang ditampilkan adalah pesan asli dari pengirim. Apabila *password* salah maka yang ditampilkan bukan pesan asli, ini dikarenakan *key* yang digunakan untuk mendekripsikan pesan salah. Proses selengkapnya dapat dilihat pada gambar 4.8 dan gambar 4.9.

Gambar 4.8 Laporan dengan *password* benar

Gambar 4.9 Laporan dengan *password* salah

4. *Error*

Error akan keluar apabila *user* salah menginputkan data. Selengkapnya dapat dilihat pada gambar 4.10.

Gambar 4.10 Tampilan error

4.3 Analisis Hasil

Pada tahap ini akan dijelaskan analisis hasil kinerja dari aplikasi perangkat lunak KriptoSMS.

a. Analisis enkripsi pesan

Pesan dan password (yang akan dijadikan key) akan di transformasi terlebih dahulu ke kode ASCII. Kemudian pesan akan dienkripsi menggunakan key yang telah diinputkan dengan metode vigenere cipher yaitu dengan menggeser setiap karakter sesuai dengan keynya.

Pesan : Sstt…ini adalah pesan rahasia

Key : qwert

Ciphertext : C4 EA D9 E6 A2

9F A5 CE E0 DD 91 D8 C9 D3 E0 D2 DF 85 E2 D9 E4

D8 D3 92 E6 D2 DF C6 E5 DD D2

Ciphertext masih berupa kode ASCII.

b. Analisis dekripsi pesan

Ciphertext akan tetap didekripsi oleh *sistem* walaupun *user* salah menginputkan password. Hasil pendekripsian dengan password benar sebagai berikut:

Cipherteks : C4 EA D9 E6 A2

9F A5 CE E0 DD 91 D8 C9 D3 E0 D2 DF 85 E2 D9 E4

D8 D3 92 E6 D2 DF C6 E5 DD D2

Key : qwert

Plainteks : Sstt…ini adalah pesan rahasia

Berikut hasil jika penerima salah menginputkan password:

Cipherteks : C4 EA D9 E6 A2

9F A5 CE E0 DD 91 D8 C9 D3 E0 D2 DF 85 E2 D9 E4

D8 D3 92 E6 D2 DF C6 E5 DD D2

key : asdfg

Plainteks : cwu€>2jzv0eemyql!|r*f*eo,qlbvq

Hasil uji selengkapnya dapat dilihat di table 4.1

Tabel 4.1 Hasil pengujian

Identifikasi Pengujian	Deskripsi	Prosedur Pengujian	Input	Metode Pengujian	Hasil Yang Didapat	Kesimpulan
SRS – KSMS – 01	Pengujian *input*	-Masukkan no.tujuan -Masukkan *password* -Masukkan pesan	No.tujuan, pesan dan *password*	*Black Box*		Diterima
SRS – KSMS – 02	Mengubah pesan dan *password* ke kode ASCII			*Black Box*		Diterima
SRS – KSMS – 03	Mengenkripsi pesan	Mengirimkan *ciphertext*		*Black Box*	Pesan berupa *ciphertext*	Diterima
SRS – KSMS – 04	Memberikan *header* pada *ciphertext*	*Ciphertext* akan diberi *header*		*Black Box*		Diterima
SRS – KSMS – 05	Mengirimkan pesan dan memberikan laporan ke pengirim	Mengirimkan pesan menggunakan aplikasi KriptoSMS		*Black Box*	Pesan terkirim dan pengirim menerima laporan bahwa pesan telah terkirim	Diterima
SRS – KSMS – 06	Memeriksa pesan apakah memiliki *header* KriptoSMS	Menerima sebuah pesan	Pesan yang mengandung *header* KriptoSMS	*Black Box*	Pesan diproses lebih lanjut	Diterima
			Pesan tidak mengandung *header* KriptoSMS		Pesan diabaikan	Diterima

Lanjutan **tabel 4.1 Hasil pengujian**

SRS – KSMS – 07	Menampilkan no.pengirim kepada penerima	Menampilkan no.pengirim kepada penerima		*Black Box*	No.pengirim ditampilkan	Diterima
SRS – KSMS – 08	Mendekripsi *ciphertext*	Mendekripsi *ciphertext*	*Password*	*Black Box*	*Ciphertext* didekripsi	Diterima
SRS – KSMS – 9	Mengubah pesan ASCII ke karakter	Pesan_ASCII diubah kembali menjadi karakter		*Black Box*	Pesan diubah dari kode ASCII menjadi karakter	Diterima
SRS – KSMS – 10	Menampilkan pesan kepada penerima	*Ciphertext* yang telah didekripsi ditampilkan kepada penerima		*Black Box*	Pesan ditampilkan	Diterima

BAB V

KESIMPULAN DAN SARAN

5.1 KESIMPULAN

Kesimpulan yang dapat diambil dari penulisan tugas akhir ini adalah sebagai berikut :

1. Dengan memanfaatkan fitur *Wireless Messaging* API dari J2ME seseorang dapat membuat suatu aplikasi pada telepon selular yang dapat dimanfaatkan untuk mengirimkan sebuah pesan
2. Aplikasi KriptoSMS ini dapat mengirimkan dan menerima sebuah SMS sekaligus memiliki fasilitas untuk mengamankan atau menyembunyikan informasi dari pesan yang dikirimkan.
3. Metode enkripsi vigenere cipher adalah metode enkripsi substitusi klasik yang telah ketinggalan zaman dan mudah dipecahkan oleh *cryptanalysis*. Kelemahan ini ditutup dengan cara merubah pesan ke kode ASCII lalu dienkripsi dengan metode enkripsi vigenere cipher. Dengan cara ini *ciphertext* akan lebih sulit dipecahkan oleh *cryptanalysis*.

5.2 SARAN

Saran yang ingin disampaikan adalah aplikasi KriptoSMS ini dapat dikembangkan lebih lanjut dengan cara menambahkan basis data sehingga dapat memiliki *inbox* pesan.

DAFTAR PUSTAKA

[1] Menezes Alfred, Oorschot Paul Van and Vanston Sean, 1996. "*Handbook of Applied Cryptography*", CRC Press.

[2] Piroumian Vartan. 2002. "*Wireless J2ME Platform Programming*". Prentice Hall PTR.

[3] Sommerville, Ian, 2003. "*Rekayasa Perangkat Lunak*". Erlangga. Jakarta.

[4] Supardi Yuniar. 2008 "*Pemrograman Handphone dengan J2ME"*, PT elex Media Komputindo. Jakarta.

[5] Suyanto Asep Herman. Review Metodologi Pengembangan Perangkat Lunak. http://www.asep-hs.web.ugm.ac.id/Artikel/RPL/RPL.pdf. Tanggal akses 20 Januari 2010.

[6] Wicaksono Kukuh Nasrul. Modifikasi Vigenere Cipher Dengan Menggunakan Teknik Substitusi Berulang Pada Kuncinya. http://www.informatika.org/~rinaldi/Kriptografi/2008-2009/Makalah1/MakalahIF30581-2009-a002.pdf. Tanggal akses: 20 Januari 2010.

[7] http://ocw.gunadarma.ac.id/course/industrial-technology/informatics-engineering-s1/rekayasa-perangkat-lunak-1/pendahuluan. Tanggal akses 20 Janurai 2010