BAB III

METODOLOGI DAN PERANCANGAN APLIKASI

3.1 Metodologi Penelitian

Metode penelitian yang akan digunakan dalam penelitian ini antara lain adalah sebagai berikut:

a. Studi Literatur

Pada tahap ini dilakukan studi mengenai teori yang berkaitan dengan penelitian, seperti teori tentang algoritma kriptografi Twofish, mode operasi *block cipher*, pengembangan aplikasi menggunakan bahasa C#, dan teori pendukung lainnya. Referensi yang digunakan dapat berupa buku, jurnal ilmiah, artikel, dan lain-lain.

b. Analisis dan perancangan

Pada tahap ini dilakukan analisis dan perancangan awal terhadap aplikasi yang dibangun, seperti analisis hasil studi literatur dan aplikasi yang akan dibuat, serta perancangan alur aplikasi dan desain antarmuka.

c. Implementasi

Pada tahap ini, aplikasi dibangun dengan mengimplementasi hasil analisis dan rancangan yang telah dilakukan sebelumnya dengan menggunakan bahasa pemrograman yang telah ditentukan.

d. Uji coba dan evaluasi

Pada tahap ini dilakukan uji coba terhadap aplikasi dan melakukan evaluasi terhadap hasil uji coba yang didapatkan.

3.2 Analisis Aplikasi

Pada penelitian ini, mode operasi *block cipher* yang digunakan adalah mode *Cipher Block Chaining* (CBC). Dengan menggunakan mode CBC, tiap enkripsi yang menggunakan kunci yang sama akan menghasilkan *ciphertext* berbeda selama IV yang digunakan unik untuk tiap proses. Dengan demikian, penggunaan *key* yang sama berulang kali menjadi lebih aman. Untuk menghasilkan IV yang unik pada tiap prosesnya, digunakan *pseudorandom number generator* dari *framework* .NET.

Seluruh data yang dibutuhkan untuk melakukan proses dekripsi, seperti IV akan disimpan dalam tiap berkas yang dienkripsi, sehingga tidak dibutuhkan sebuah basis data terpusat. *Key* yang digunakan untuk proses enkripsi akan menggunakan *password* yang dimasukkan oleh user dan diproses menjadi *key* 256-bit dengan menggunakan algoritma PBKDF2 (*Password-Based Key Derivation Function* 2) agar dapat mengurangi kemampuan serangan yang menggunakan *precomputed hash* seperti penggunaan *rainbow table*.

Aplikasi dibuat dengan menggunakan IDE *Microsoft Visual Studio* 2008 *Service Pack* 1 dan menggunakan bahasa pemrograman C#. *Framework* yang digunakan adalah .NET *Framework* 3.5 dan dibuat untuk *platform* windows 32-bit dan 64-bit.

3.3 Perancangan Aplikasi

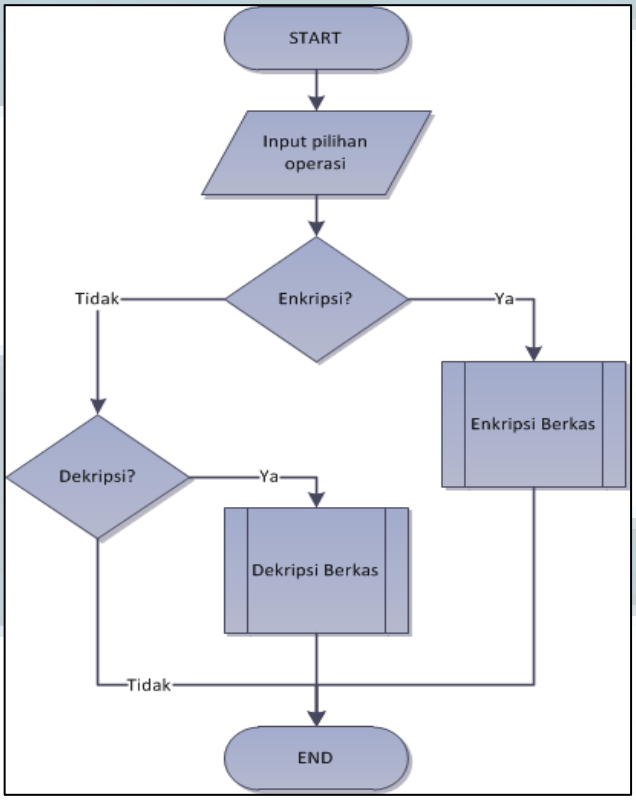
Pada subbab ini dipaparkan hasil rancangan dalam dua bagian, yaitu *system flow* dan *flow chart*, serta sketsa antarmuka aplikasi.

3.3.1 System Flow dan Flow Chart

System flow dan *flow chart* merupakan diagram yang memiliki arus dan digunakan untuk menggambarkan langkah-langkah dalam menyelesaikan suatu proses atau algoritma.

A. System Flow Aplikasi

Gambar 3.1 menggambarkan alur aplikasi secara umum. Pada saat aplikasi dijalankan, aplikasi menampilkan halaman utama bagi pengguna untuk memilih proses yang dilakukan. Apabila pengguna ingin melakukan proses enkripsi terhadap sebuah berkas digital, pengguna akan dibawa menuju halaman enkripsi. Sedangkan apabila pengguna ingin mendekripsi sebuah berkas yang sudah dienkripsi terlebih dahulu, pengguna akan dibawa menuju halaman dekripsi.



Gambar 3.1 *System flow* aplikasi

B. Flow Chart Subproses Enkripsi Berkas

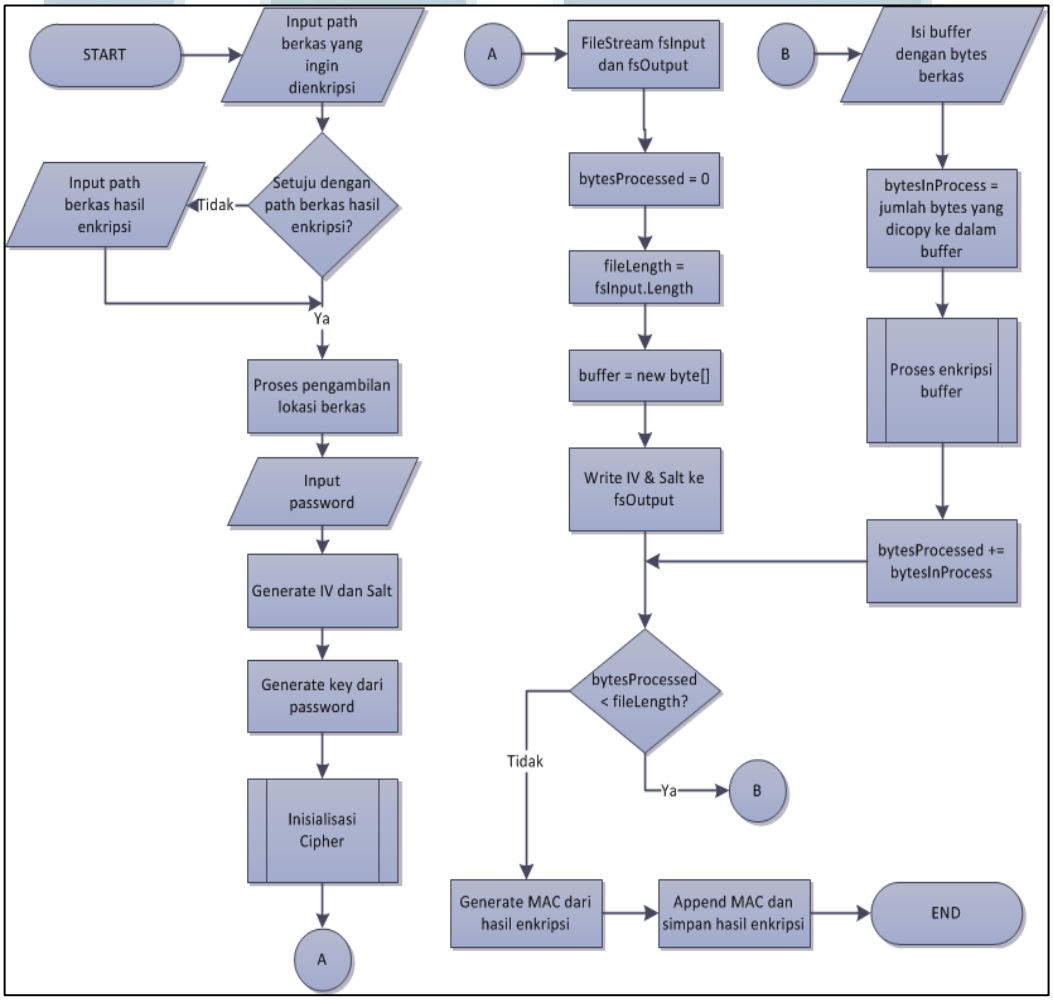
Subproses enkripsi berkas merupakan proses yang dijalankan ketika pengguna ingin mengenkripsi sebuah berkas digital. Dalam proses ini, pengguna diminta untuk memasukkan lokasi berkas yang ingin dienkripsi. Secara *default*, aplikasi akan meletakkan hasil enkripsi ke dalam lokasi yang sama dengan berkas aslinya, namun pengguna dapat mengubah lokasi yang digunakan untuk menyimpan hasil enkripsi. Kemudian pengguna akan memasukkan *password* yang digunakan untuk mengenkripsi berkas.

Sistem akan menghasilkan IV dan *salt* yang akan digunakan dalam proses enkripsi, kemudian dilanjutkan dengan menciptakan *key* berdasarkan *password* yang telah dimasukkan oleh pengguna dan memulai proses inisialisasi *cipher*. Setelah *cipher* berhasil diinisialisasi, dibuat dua *FileStream* yang digunakan untuk membaca berkas asli (fsInput) dan menulis berkas hasil enkripsi (fsOutput). Kemudian diikuti dengan inisialisasi variabel yang digunakan dalam tahap enkripsi berkas, yaitu:

- Variabel *long bytesProcessed* yang merepresentasikan jumlah *byte* yang telah diproses.
- Variabel *long fileLength* yang merepresentasikan ukuran berkas dalam satuan *byte*.
- Variabel *array of bytes buffer* yang akan digunakan untuk menyimpan *bytes* yang dibaca dari berkas asli untuk sementara sebelum diproses.

Sebelum memulai proses enkripsi, IV dan *salt* ditulis ke dalam berkas hasil enkripsi melalui *fsOutput*. Kemudian dilakukan pengecekan terhadap variabel

bytesProcessed, apakah nilai variabel tersebut lebih kecil dari variabel *fileLength*. Jika benar, *buffer* akan diisi menggunakan *bytes* dari berkas asli, variabel *bytesInProcess* diisi dengan jumlah *bytes* dalam *buffer*, kemudian *buffer* tersebut akan dienkripsi menggunakan algoritma *twofish* pada subproses enkripsi *buffer* dan variabel *bytesProcessed* akan ditambahkan dengan variabel *bytesInProcess*. Apabila salah, sistem akan menghasilkan MAC berdasarkan hasil enkripsi dan menambahkan MAC tersebut pada bagian akhir berkas hasil enkripsi.

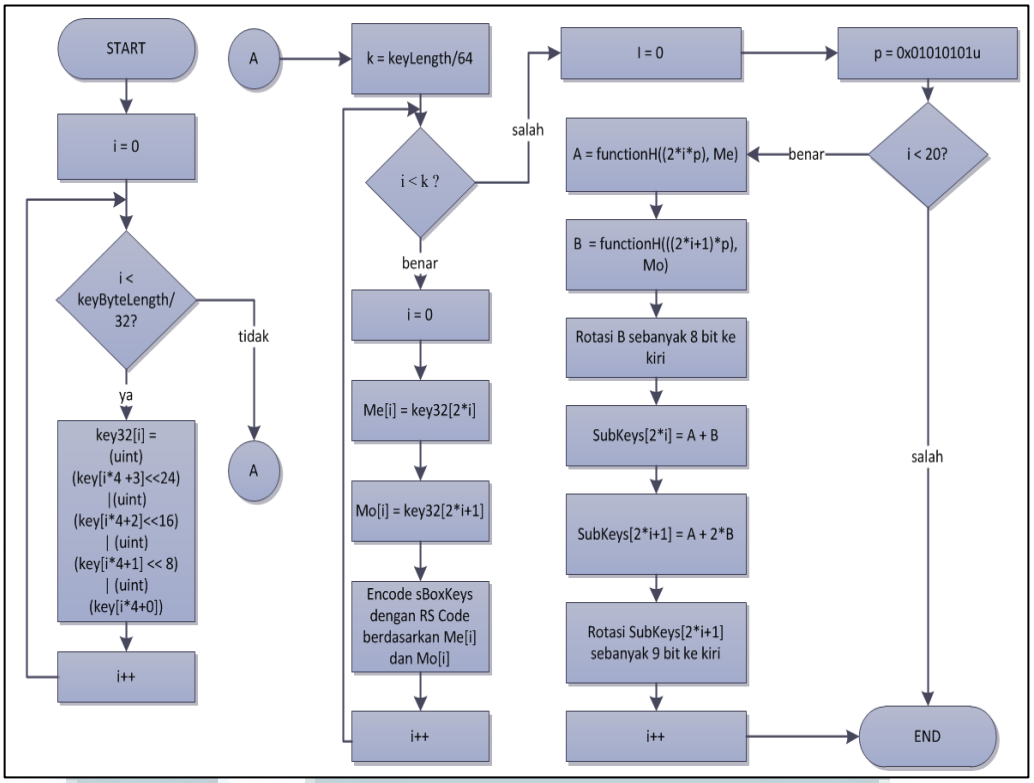


Gambar 3.2 *Flow chart* subproses enkripsi berkas

C. Flow Chart Subproses Inisialisasi Cipher

Subproses inisialisasi cipher merupakan proses dimana seluruh *expanded key* yang dibutuhkan untuk mengenkripsi diciptakan. Proses dimulai dengan mengubah *key* menjadi *unsigned integer* yang berukuran 32 bit, lalu dipecah menjadi dua *array* berdasarkan ganjil dan genap. Selain menghasilkan dua *array* ganjil dan genap, sistem juga melakukan perhitungan menggunakan matriks RS terhadap *bytes key* untuk menghasilkan sBoxKeys, yaitu *key* yang akan digunakan dalam substitusi *key dependent* S-Box.

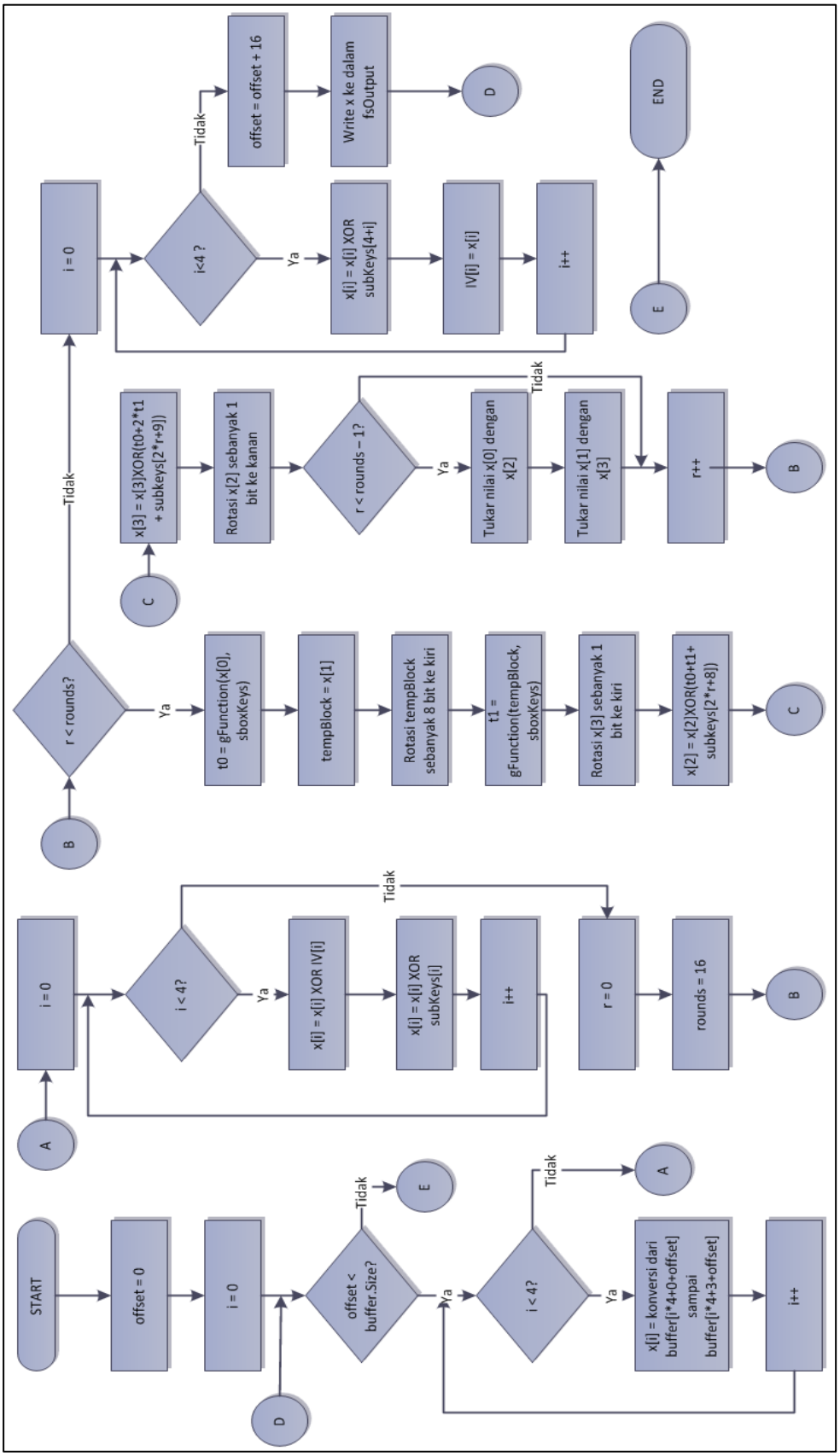
Kemudian sistem mulai memroses *round key* subKeys yang digunakan pada tahap *whitening* dan *round function* proses enkripsi dan dekripsi, dimulai dengan membuat konstanta *p* yang bernilai $2^{24} + 2^{16} + 2^{8} + 0$. Variabel A akan diisi dengan hasil dari *function h* dengan menggunakan [2 * i * p] sebagai masukan pertama dan *array* genap sebagai masukan kedua. Variabel B diisi dengan hasil dari *function h* dengan menggunakan [(2 * i + 1) * p] sebagai masukan pertama dan *array* ganjil sebagai masukan kedua. Variabel B akan dirotasi sebanyak delapan bit ke kiri. Subkeys[2 * i] diisi menggunakan hasil dari PHT A dan B, sedangkan subKeys[2 * i + 1] diisi menggunakan hasil dari PHT A dan 2 * B, kemudian dirotasi ke kiri sebanyak delapan bit. Proses ini diulang hingga seluruh *round key* berhasil dibuat.



Gambar 3.3 *Flow chart* subproses inisialisasi *cipher*

D. Flow Chart Subproses Enkripsi Buffer

Subproses enkripsi *buffer* merupakan proses yang melakukan transformasi enkripsi pada *buffer* sebelum ditulis ke dalam berkas hasil enkripsi. Dalam proses ini, sebuah variabel *array of unsigned integer x* berukuran 4 dibuat dan diisi dengan hasil konversi *buffer*, diikuti dengan melakukan XOR dengan IV dan melalui proses *whitening*. Kemudian *x* akan diproses melalui *round function cipher twofish* sebanyak 16 kali. Setelah melalui 16 putaran, pertukaran terakhir yang dilakukan oleh *cipher* dibatalkan dan diikuti dengan melakukan *whitening* terakhir. Sebelum memroses blok berikutnya, IV akan diisi dengan nilai *ciphertext* sebelumnya.



Gambar 3.4 *Flow chart* subproses enkripsi *buffer*

E. Flow Chart Subproses Dekripsi Berkas

Subproses dekripsi berkas merupakan proses yang dijalankan ketika pengguna ingin mendekripsi sebuah berkas digital. Dalam proses ini, pengguna diminta untuk memasukkan lokasi berkas yang ingin didekripsi. Secara *default*, aplikasi akan meletakkan hasil dekripsi ke dalam lokasi yang sama dengan berkas aslinya, namun pengguna dapat mengubah lokasi yang digunakan untuk menyimpan hasil dekripsi. Kemudian pengguna akan memasukkan *password* yang akan digunakan untuk mendekripsi berkas.

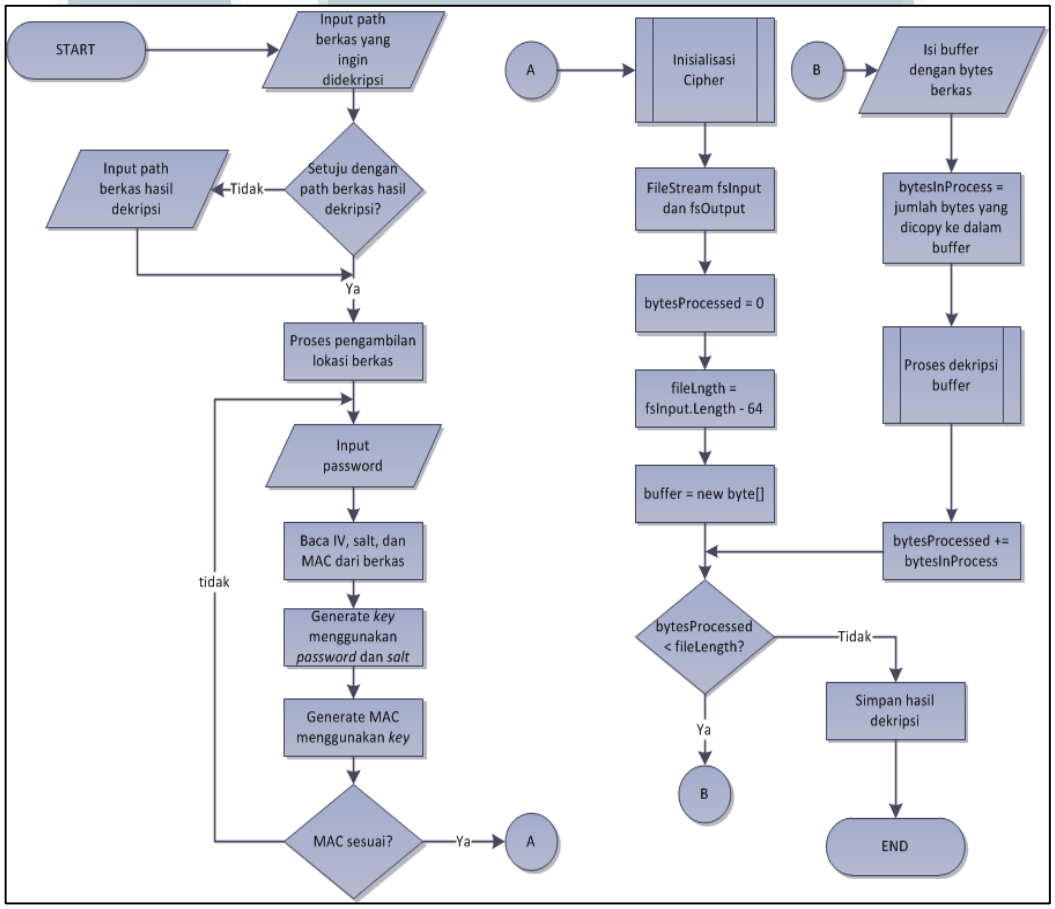
Sistem akan membaca IV, MAC, dan *salt* dari berkas yang digunakan dan menghasilkan *key* berdasarkan *salt* dan *password* yang dimasukkan pengguna. Sistem akan menghasilkan MAC dari berkas tersebut berdasarkan *key* dan membandingkan MAC tersebut dengan MAC yang tersimpan pada berkas. Apabila tidak sama, pengguna diminta untuk memasukkan kembali *password*. Sebaliknya, sistem memulai proses inisialisasi *cipher* jika MAC sudah sesuai.

Setelah *cipher* berhasil diinisialisasi, dibuat dua *FileStream* yang digunakan untuk membaca berkas yang terenkripsi (fsInput) dan menulis berkas hasil dekripsi (fsOutput). Kemudian diikuti dengan inisialisasi variabel yang digunakan dalam tahap dekripsi berkas, yaitu:

- Variabel *long bytesProcessed* yang merepresentasikan jumlah *byte* yang telah diproses.
- Variabel *long fileLength* yang merepresentasikan ukuran berkas dalam satuan *byte* tanpa IV, *salt*, dan MAC.

– Variabel *array of bytes buffer* yang akan digunakan untuk menyimpan *bytes* yang dibaca dari berkas untuk sementara sebelum diproses.

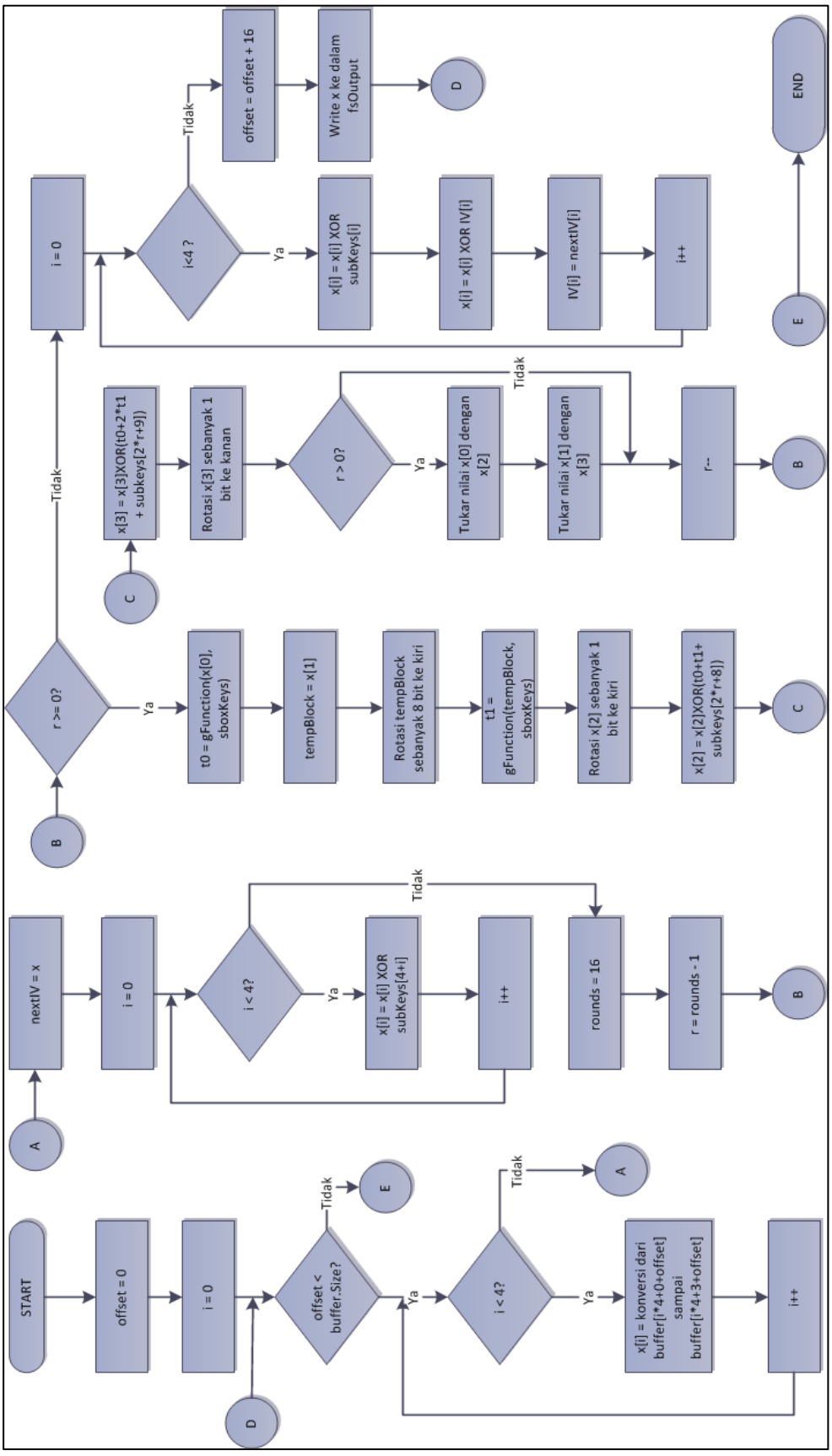
Sistem melakukan pengecekan terhadap variabel *bytesProcessed*, apakah nilai variabel tersebut lebih kecil dari variabel *fileLength*. Jika benar, *buffer* akan diisi menggunakan *bytes* dari berkas yang terenkripsi, variabel *bytesInProcess* diisi dengan jumlah *bytes* dalam *buffer*, kemudian *buffer* tersebut akan didekripsi menggunakan algoritma dekripsi *twofish* pada subproses dekripsi *buffer* dan variabel *bytesProcessed* ditambahkan dengan variabel *bytesInProcess*. Apabila salah, sistem akan menyimpan hasil dekripsi dalam *path* yang telah ditentukan sebelumnya.



Gambar 3.5 *Flow chart* subproses dekripsi berkas

F. Flow Chart Subproses Dekripsi Buffer

Subproses dekripsi buffer merupakan proses yang melakukan transformasi dekripsi pada buffer sebelum ditulis ke dalam berkas hasil enkripsi. Dalam proses ini, sebuah variabel *array of unsigned integer x* berukuran empat dibuat dan diisi dengan hasil konversi buffer, diikuti dengan melakukan proses *whitening* terhadap *x*. Kemudian *x* akan diproses melalui *round function* Twofish sebanyak 16 kali. Setelah melalui 16 putaran, pertukaran terakhir yang dilakukan oleh *cipher* dibatalkan dan diikuti dengan melakukan *whitening* terakhir. Sebelum memroses blok berikutnya, IV akan diisi dengan nilai *ciphertext* sebelum diproses.



Gambar 3.6 *Flow chart* subproses dekripsi *buffer*

3.3.2 Sketsa Antarmuka Aplikasi

Sketsa antarmuka dibuat untuk menjadi acuan implementasi antarmuka aplikasi. Berikut adalah hasil perancangan sketsa antarmuka aplikasi dalam penelitian ini.

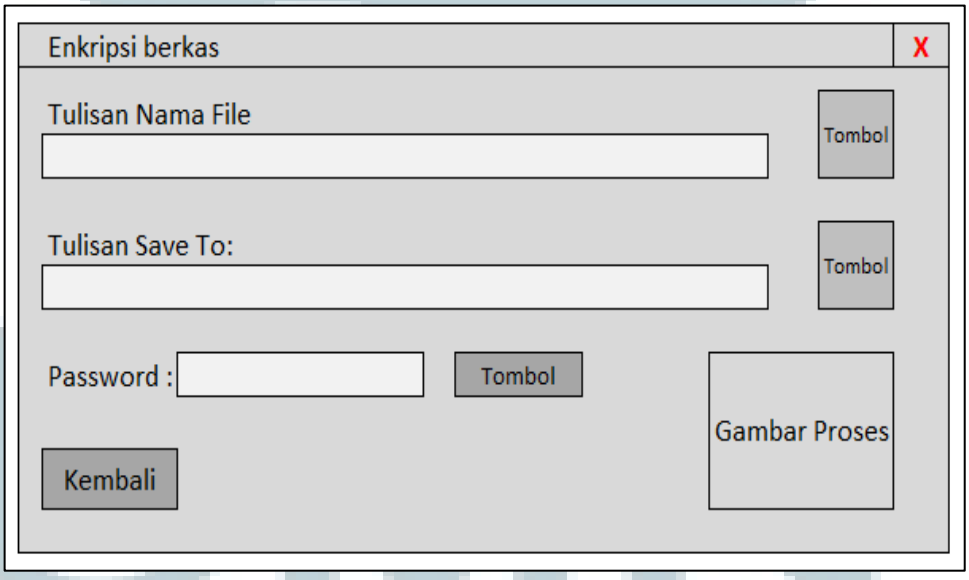
A. Halaman Utama

Halaman utama adalah halaman yang pertama kali ditampilkan kepada pengguna saat memulai aplikasi. Pada halaman utama, terdapat dua pilihan menu utama yang masing - masing dapat dipilih oleh pengguna, yaitu menu enkripsi dan menu dekripsi berkas. Selain itu, terdapat dua tombol yang memiliki fungsi untuk menampilkan halaman berisi bantuan penggunaan aplikasi dan halaman berisi informasi aplikasi. Di bagian bawah terdapat tulisan hak cipta aplikasi dan Universitas Multimedia Nusantara.

Gambar 3.7 Sketsa antarmuka halaman utama

B. Halaman Enkripsi

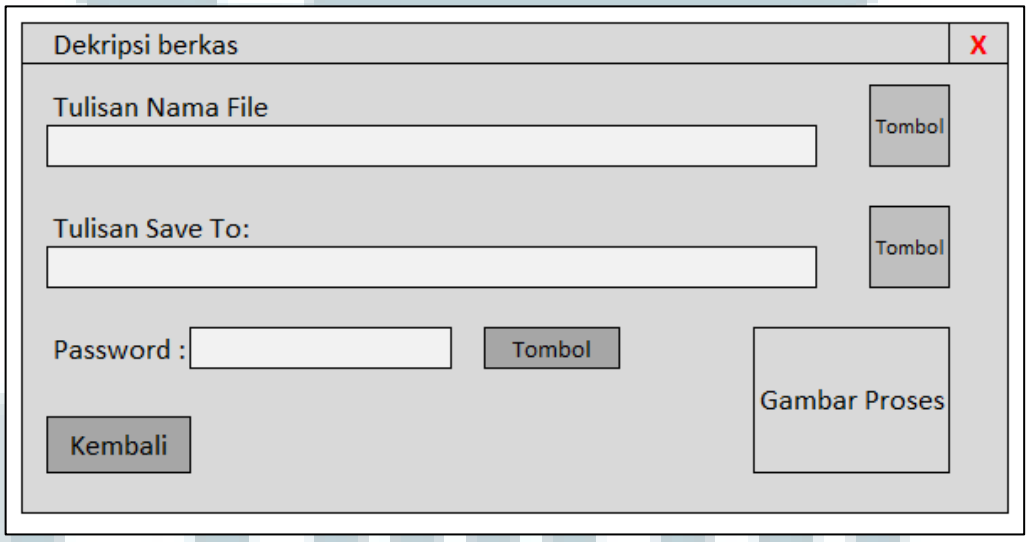
Halaman enkripsi adalah halaman yang ditampilkan kepada pengguna setelah pengguna menekan tombol enkripsi. Halaman ini terdiri dari dua *textbox* yang berisi informasi *path* berkas yang ingin dienkripsi dan *path* untuk menyimpan berkas hasil enkripsi. Untuk mengisi kedua *textbox* tersebut, pengguna dapat menekan dua tombol yang terletak di samping *textbox* masing-masing. Setelah memilih berkas yang ingin dienkripsi, pengguna harus mengisi *textbox password* dan menekan tombol di sebelahnya untuk memulai proses enkripsi. Selama proses enkripsi berjalan, akan ditampilkan gambar agar pengguna mengetahui bahwa aplikasi sedang bekerja.



Gambar 3.8 Sketsa antarmuka halaman enkripsi

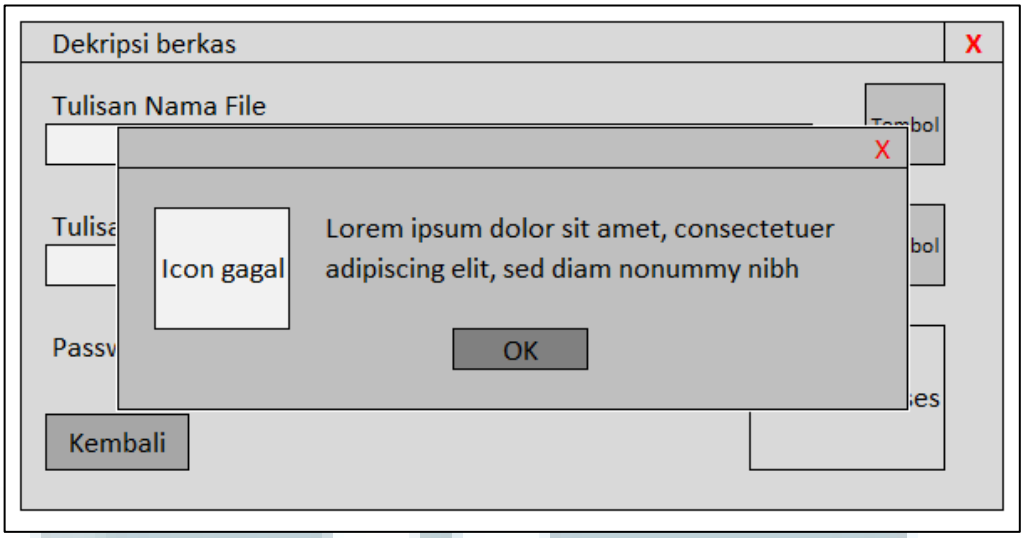
C. Halaman Dekripsi

Halaman dekripsi adalah halaman yang ditampilkan kepada pengguna setelah pengguna menekan tombol dekripsi. Halaman ini terdiri dari dua *textbox* yang berisi informasi *path* berkas yang ingin didekripsi dan *path* untuk menyimpan berkas hasil dekripsi. Untuk mengisi kedua *textbox* tersebut, pengguna dapat menekan dua tombol yang terletak di samping *textbox* masing-masing. Setelah memilih berkas yang ingin dienkripsi, pengguna harus mengisi *textbox password* dan menekan tombol di sebelahnya untuk memulai proses dekripsi. Selama proses dekripsi berjalan, akan ditampilkan gambar agar pengguna mengetahui bahwa aplikasi sedang bekerja.



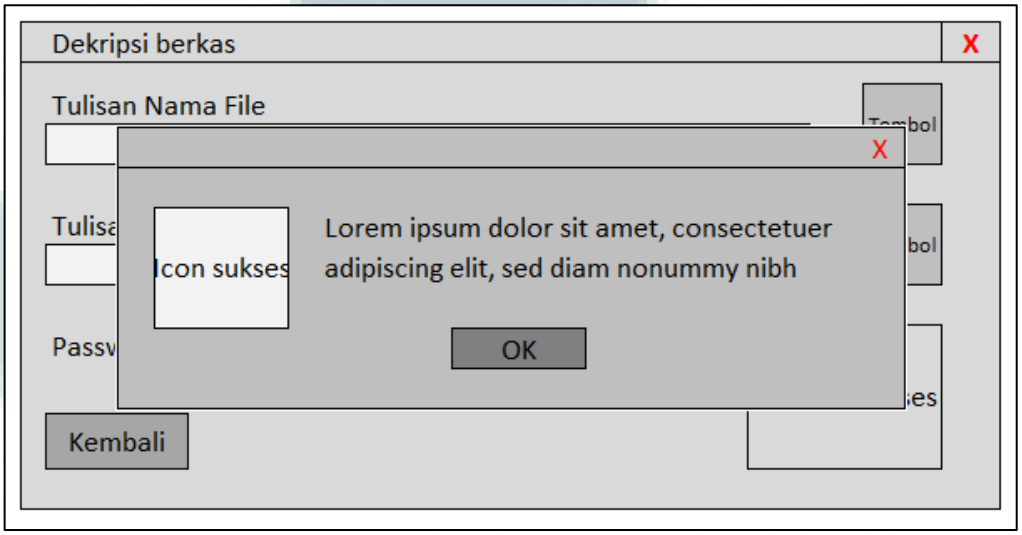
Gambar 3.9 Sketsa antarmuka halaman dekripsi

Apabila *password* yang dimasukkan oleh pengguna salah, akan ditampilkan *popup dialog* yang akan memberikan notifikasi kegagalan dekripsi kepada pengguna.



Gambar 3.10 Sketsa antarmuka *popup dialog* gagal dekripsi

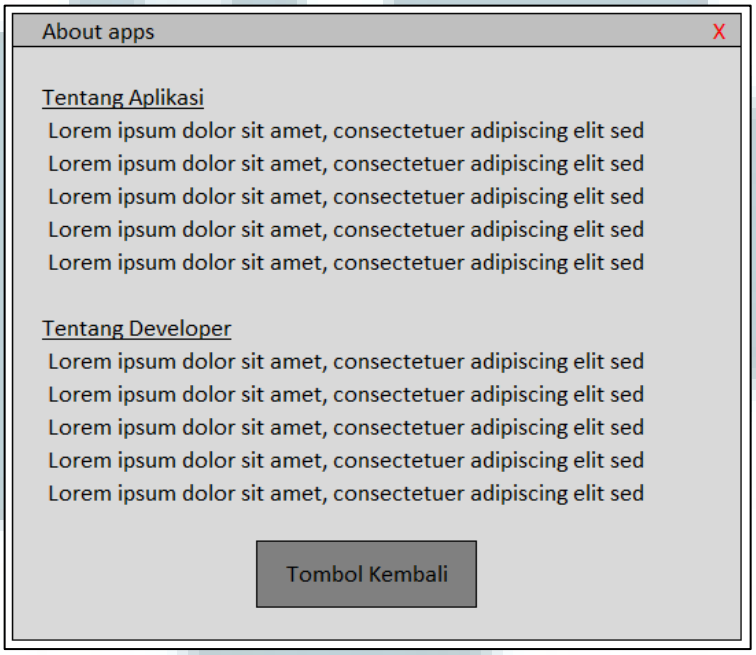
Sedangkan apabila *password* yang dimasukkan oleh pengguna benar, aplikasi akan mendekripsi berkas dan menampilkan *popup dialog* yang akan memberikan notifikasi keberhasilan dekripsi kepada pengguna.



Gambar 3.11 Sketsa antarmuka *popup dialog* berhasil dekripsi

D. Halaman Tentang Aplikasi

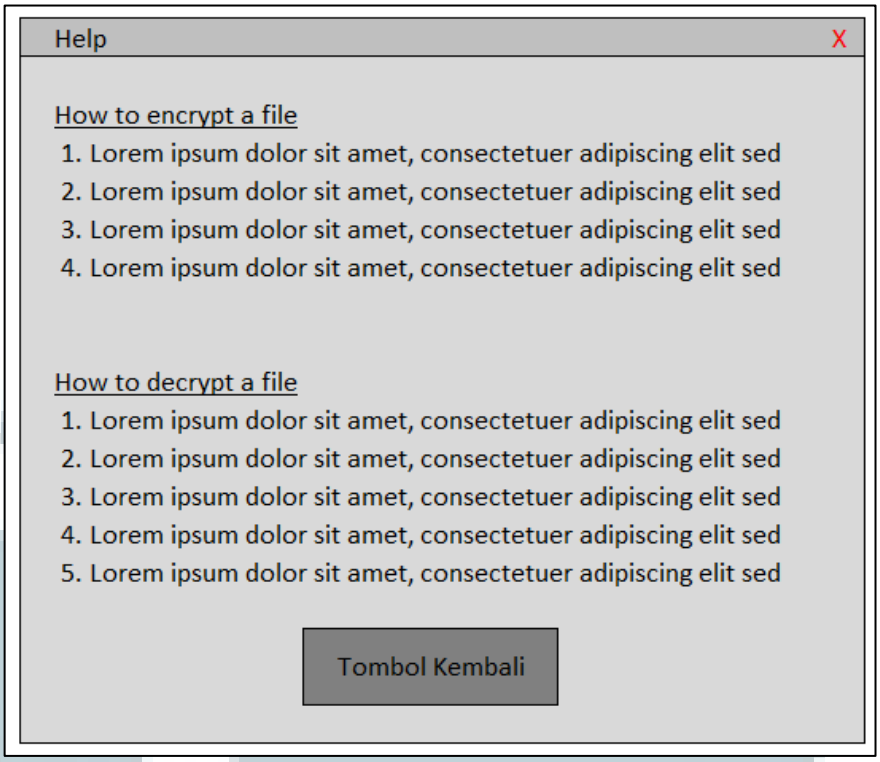
Halaman ini akan ditampilkan ketika pengguna menekan tombol “About” yang terdapat pada halaman utama. Halaman ini berisikan informasi tentang aplikasi dan informasi pembangun aplikasi.



Gambar 3.12 Sketsa antarmuka halaman tentang aplikasi

E. Halaman Bantuan

Halaman bantuan akan ditampilkan saat pengguna menekan tombol bantuan pada halaman utama. Pengguna dapat mempelajari cara mengenkripsi dan mendekripsi berkas melalui informasi yang ditampilkan pada halaman ini.



Gambar 3.13 Sketsa antarmuka halaman tentang aplikasi