BAB III

METODOLOGI PENELITIAN

3.1 Metode Penelitian

Untuk proses pembuatan dan perancangan sistem pada aplikasi alat bantu pembelajaran metode SHA256, RSA, dan *twofish*. Penelitian dilakukan dengan mengikuti tahapan berikut

1. Studi Literatur

Tahap awal pada penelitian ini adalah studi mengenai referensi-referensi yang berhubungan dengan inti bahasan penelitian tugas akhir yaitu pembuatan dan perancangan aplikasi berkas pengaman digital, aplikasi alat bantu pembelajaran, metode *digital signature*, algoritma SHA256, algoritma RSA, algoritma *twofish* dan berbagai konsep pendukung lainnya. Referensi-referensi ini berupa jurnal, artikel, dan buku.

2. Perancangan Sistem

Perancangan aplikasi alat bantu pembelajaran dibagi menjadi dua tahap, tahap pertama adalah perancangan alur aplikasi dalam bentuk *flowchart*. Tahap kedua adalah melakukan perancangan aplikasi untuk menampilkan dan mencatat hasil dari proses *hashing* dan enkripsi dekripsi file, rancangan *user interface* dan proses – proses dari ketiga algoritma langkah demi langkah, agar pengguna dapat mempelajari proses – proses dari ketiga algoritma tersebut.

3. Penulisan Kode Program

Penulisan kode program merupakan tahap dari implementasi perancangan dan pembuatan sistem aplikasi ke dalam komputer dengan menggunakan bahasa pemrograman untuk membentuk sebuah aplikasi alat bantu pembelajaran yang dapat dijalankan oleh pengguna di komputer.

4. **Melakukan Uji Coba Aplikasi**

Melakukan uji coba terhadap aplikasi untuk memastikan aplikasi yang dibuat telah sesuai dengan rancangan yaitu sebagai aplikasi alat bantu pembelajaran dan ketiga algoritma berjalan sesuai dengan fungsinya. Selain itu, uji coba juga dilakukan untuk mendapatkan data tentang penggunaan memori dan waktu proses ketiga algoritma. Hasil uji coba dievaluasi untuk mendapatkan kesimpulan dan dokumentasi tentang kinerja dan hasil dari aplikasi yang telah dibuat.

5. Melakukan Dokumentasi

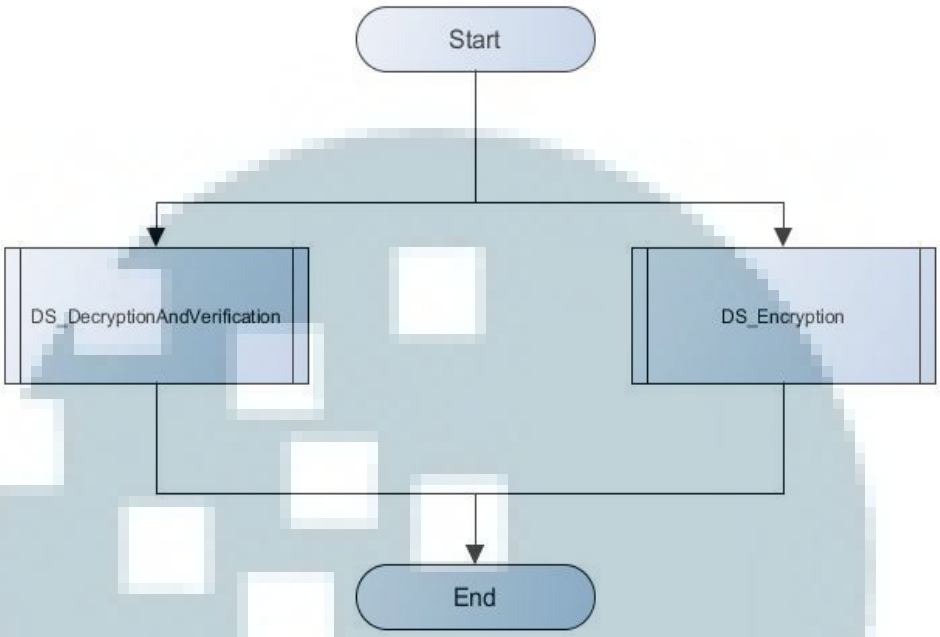
Dokumentasi dapat dilakukan dengan menulis keterangan pada program aplikasi yang dibuat dan mencatat hasil pengujian aplikasi. Dokumentasi diperlukan agar peneliti dapat menganalisis hasil dari aplikasi untuk dituliskan ke dalam laporan tugas akhir dan dengan adanya dokumentasi ini dapat membantu peneliti lainnya yang ingin melanjutkan penelitian ini.

3.2 Perancangan Sistem

3.2.1 Diagram Alir

Diagram Alir adalah bagan yang memiliki arus yang bertujuan untuk menggambarkan langkah – langkah dan proses dari suatu sistem aplikasi. Diagram alir sistem pada aplikasi yang dibuat oleh peneliti dibagi menjadi 13 bagian yaitu diagram alir menu utama, diagram alir menu enkripsi, diagram alir *sub* proses pembangkit kunci RSA dan diagram alir proses enkripsi algoritma RSA, diagram alir proses *hashing* algoritma SHA256, diagram alir *sub* proses enkripsi, diagram alir proses inisialisasi *cipher* Twofish, diagram alir *sub* proses enkripsi berkas algoritma *twofish*, diagram alir proses enkripsi algoritma *twofish*, diagram alir menu dekripsi, diagram alir *sub* proses dekripsi dan menu verifikasi, diagram alir *sub* proses dekripsi berkas algoritma *twofish*, diagram alir proses dekripsi algoritma *twofish*, diagram alir proses pengambilan kunci publik RSA dan proses dekripsi algoritma RSA, dan diagram alir *sub* proses Verifikasi.

A. Diagram Alir Menu Utama

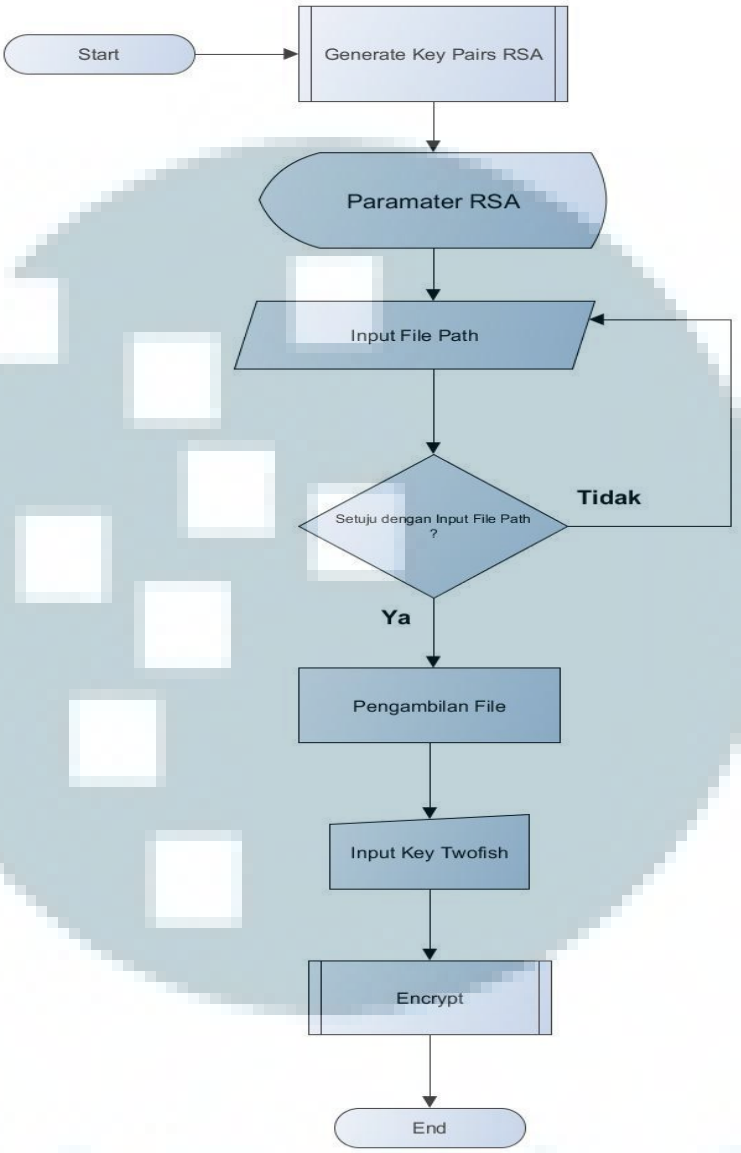


Gambar 3.1 Diagram Alir Menu Utama

Ketika pertama kali pengguna mengakses aplikasi, sistem menampilkan halaman utama. Pada halaman utama, pengguna diberikan pilihan berupa dua tombol yaitu tombol menuju menu utama aplikasi dan menu yang berisi tentang penjelasan aplikasi.

Bila pengguna mengakses tombol menu utama maka pengguna dapat melihat menu yang terdiri dari dua menu tab yaitu menu *DS_Encryption* untuk pengguna melakukan proses enkripsi suatu file dan menu *DS_DecryptionAndVerification* untuk pengguna melakukan proses dekripsi dan verfikasi file. Pengguna dapat melakukan proses enkripsi, dekripsi, dan verifikasi secara terpisah tanpa perlu melakukan proses lainnya terlebih dahulu.

B. Diagram Alir Menu Enkripsi

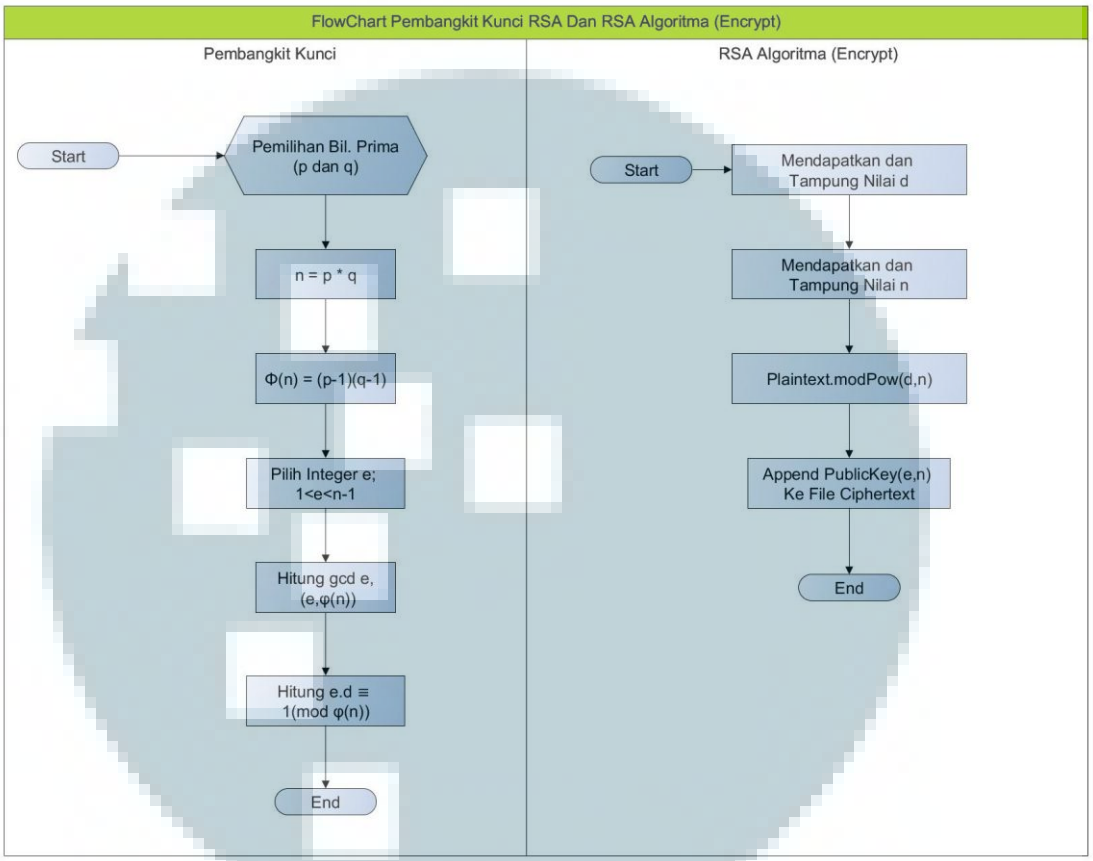


Gambar 3.2 Diagram Alir Menu Enkripsi

Pada menu enkripsi yang terlihat pada gambar 3.2, pengguna diberikan langkah – langkah dalam melakukan proses enkripsi suatu berkas. Langkah – langkah tersebut berupa proses –proses yang harus dilakukan secara satu persatu sebelum pengguna dapat melakukan proses enkripsi.

Terlebih dahulu pengguna mengakses proses *Generate Key Pairs* RSA untuk membuat parameter – parameter. Parameter – parameter tersebut berupa p, q, n, φ, e, dan d. Nilai dari parameter – parameter digunakan untuk membuat kunci publik dan privat. Kedua kunci digunakan untuk proses enkripsi dan dekripsi RSA. Setelah pengguna mengakses proses tersebut maka pengguna dapat memasukan alamat dari berkas yang ingin dienkripsi oleh pengguna. Langkah berikutnya adalah menuliskan *Input Key Twofish*. Lalu tombol untuk proses enkripsi aktif di saat pengguna telah mengisi input untuk kunci dari algoritma *twofish*.

C. Diagram Alir *Sub* Proses Pembangkit Kunci RSA (*Generate Key Pairs RSA*) Dan Diagram Alir Proses Enkripsi Algoritma RSA

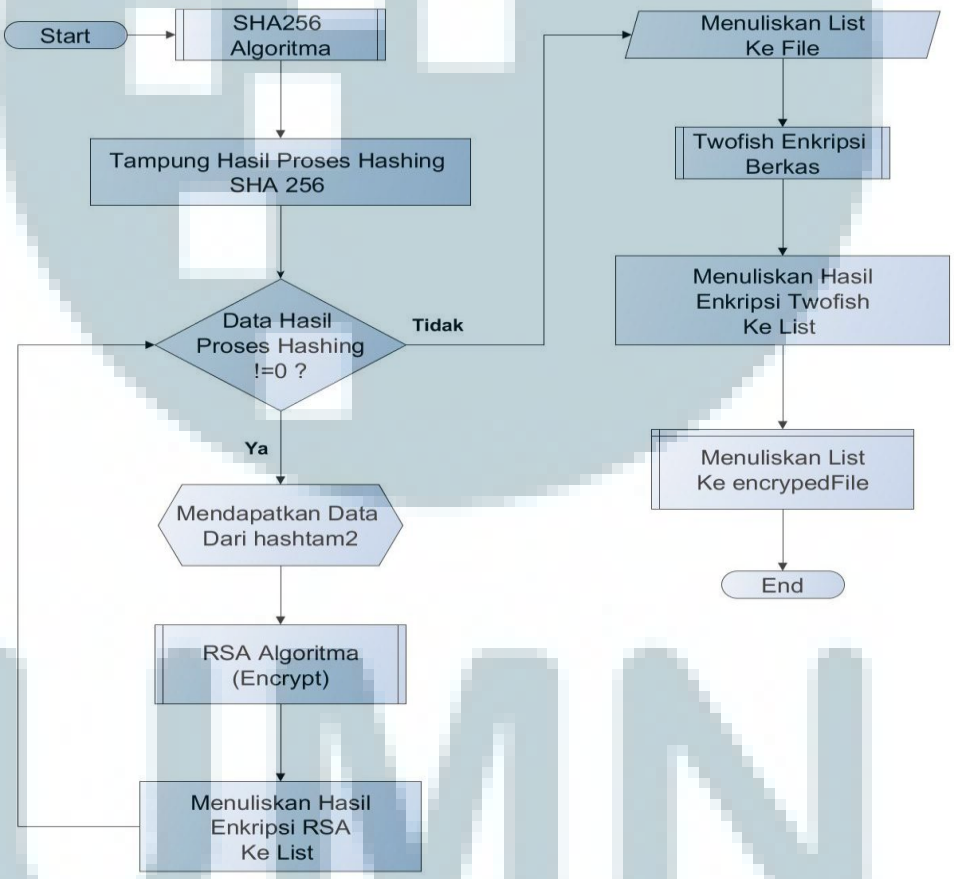


Gambar 3.3 Diagram Alir Proses Pembangkit Kunci RSA (*Generate Key Pairs RSA*) Dan Diagram Alir Proses Enkripsi Algoritma RSA

Diagram ini menjelaskan dua proses yaitu proses pembangkit kunci RSA dan proses enkripsi algoritma RSA. Proses pembangkit kunci RSA dijalankan terlebih dahulu untuk menghasilkan parameter – parameter pembentuk kunci publik (e, n) dan kunci privat (d, n) yaitu n, e, dan d. Nilai n merupakan hasil perkalian dari nilai p dan q, setelah pemilihan nilai bilangan primer p dan q didapatkan.

Kemudian nilai e didapatkan dari pemilihan nilai yang koprima dengan $\varphi(n)$ antara 1 dan phi ($\varphi(n)$). Untuk nilai d didapatkan hasil perhitungan *inverse* dari e modulo $\varphi(n)$. Kunci privat (d, n) digunakan pada saat pengguna ingin melakukan enkripsi pada suatu file (*plaintext*). Kunci publik (e, n) digabungkan dengan hasil enkripsi RSA dan digunakan proses dekripsi algoritma RSA nantinya.

D. Diagram Alir *Sub* Proses Enkripsi



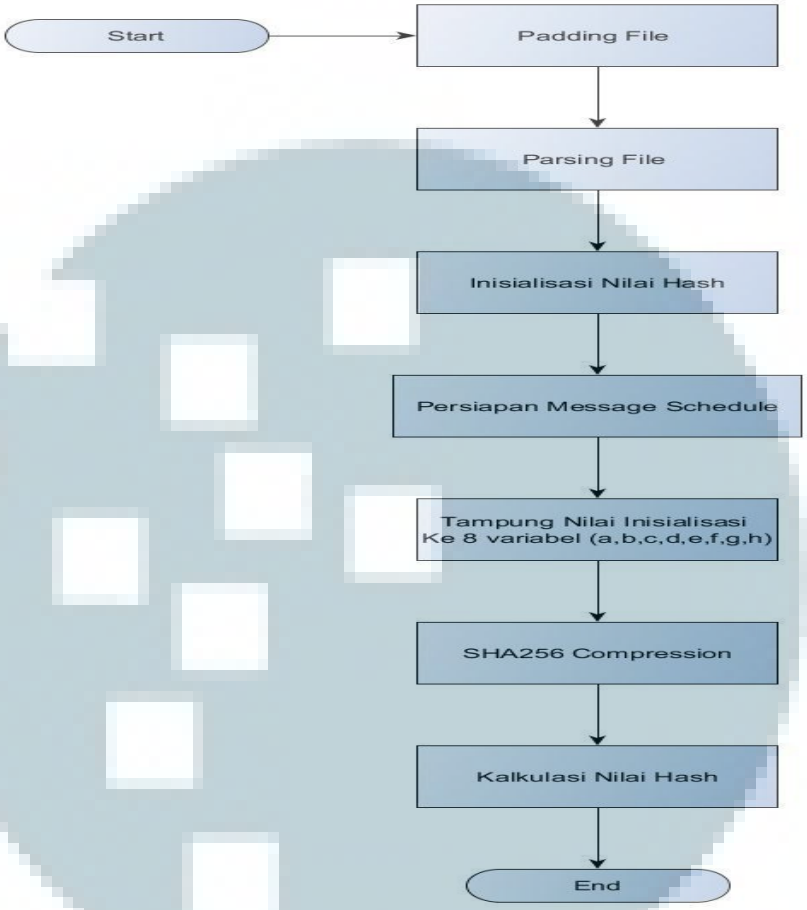
Gambar 3.4 Diagram Alir Proses Enkripsi

Diagram di atas menunjukkan alir proses – proses enkripsi file ketika pengguna mengakses tombol *encrypt* pada menu *DS_Encryption*, setelah pengguna melengkapi langkah – langkah untuk mengaktifkan tombol *encrypt*. Di dalam diagram ini terdapat beberapa diagram *Sub* proses yaitu diagram proses *hashing* algoritma SHA256, diagram proses enkripsi algoritma RSA seperti yang telah diperlihatkan pada gambar 3.3, dan diagram proses enkripsi *Twofish*. Hasil dari proses enkripsi algoritma *twofish* disimpan ke dalam file *internal* di aplikasi.

Sub proses yang dijalankan pertama kali adalah proses hashing algoritma SHA256 yang bertujuan untuk mengolah input data menjadi *digest file*. *Digest file* ditampung dalam suatu variabel , agar dapat diproses oleh proses enkripsi algoritma RSA. Proses enkripsi algoritma RSA dilakukan dengan menggunakan kunci privat yang telah dibentuk pada proses pembangkit kunci di atas.

Hasil dari proses enkripsi algoritma RSA dikumpulkan dalam sebuah file dan kunci publik digabungkan dengan ke dalam file tersebut, di mana file ini menjadi input untuk proses enkripsi *twofish*. Data hasil enkripsi *twofish* dicetak ke dalam file.

E. Diagram Alir Proses *Hashing* Algoritma SHA256

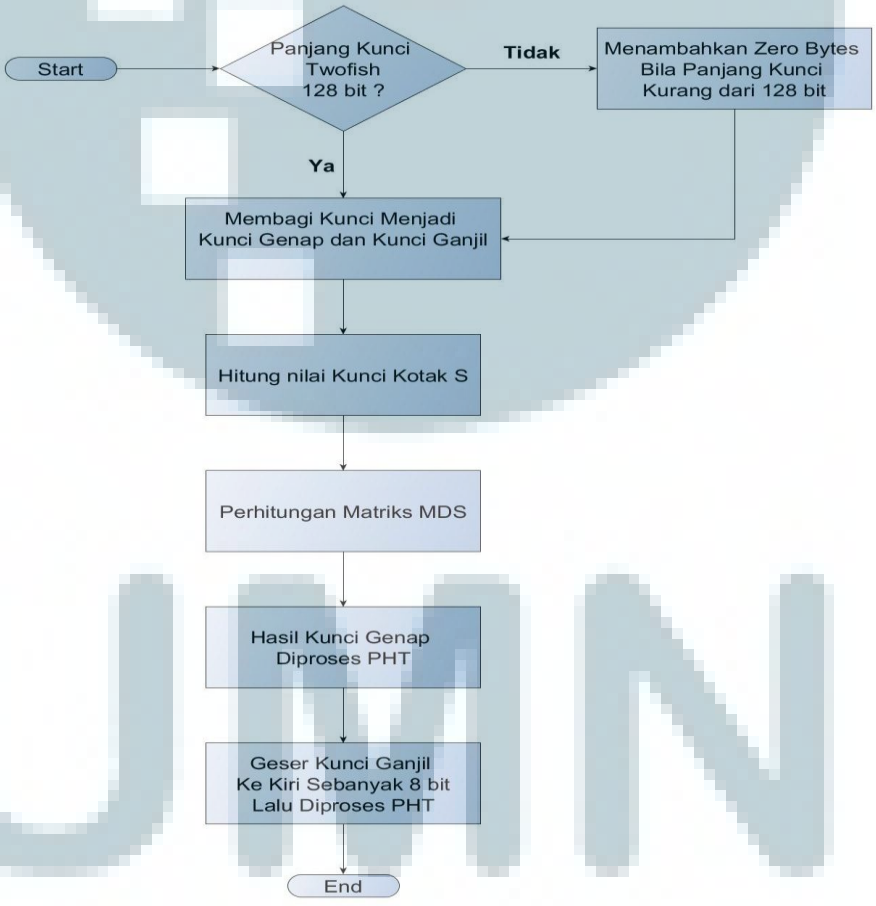


Gambar 3.5 Diagram Alir Proses *Hashing* SHA256

Diagram alir ini menunjukkan proses *hashing* pada file yang telah dimasukkan oleh pengguna terlebih dahulu. Proses *hashing* dimulai dengan *padding* file yang telah dimasukkan. Setelah input file di*padding*, file tersebut di*parsing* menjadi input blok sebanyak 16 dengan masing – masing ukurannya 32 bit. Sebelum proses komputasi dari *hashing* dimulai, proses inisialisasi 8 nilai hash dimulai. Setelah itu proses komputasi *hash* dimulai.

Proses ini terbagai menjadi empat langkah yaitu pembagian file yang telah di*parsing* menjadi 64 bagian (W_0, W_1,…, W_{63}) (*Message Schedule*), nilai delapan inisialisasi ditampung ke dalam delapan varibel, tukar nilai dari delapan variabel sebanyak 64 kali (*SHA256 Compression*), dan hitung nilai akhir dari nilai hasil *hashing* (Kalkulasi Nilai Hash), nilai akhir merupakan *digest file*. Selanjutnya *digest file* menjadi *input* untuk dienkripsi oleh algoritma RSA.

F. Diagram Alir *Sub* Proses Insialisasi Kunci Algoritma *Twofish*

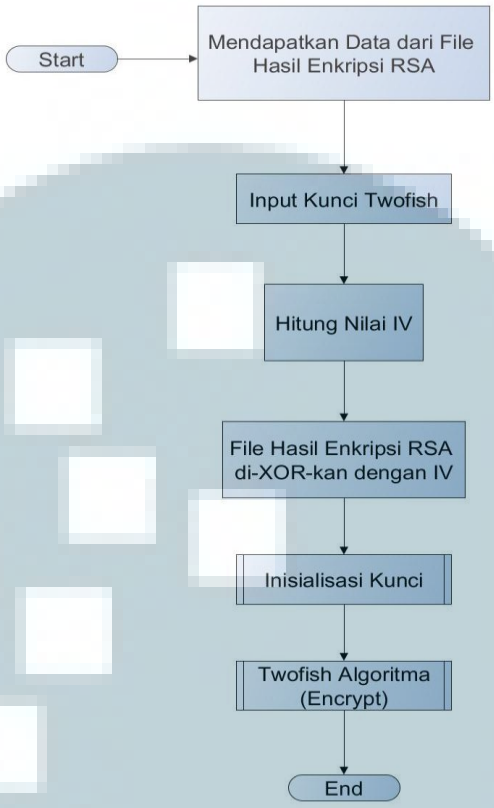


Gambar 3.6 Diagram Alir Insialisasi Kunci Algoritma *Twofish*

Diagram pada gambar 3.6 menjelaskan *sub* proses inisialisasi kunci untuk algoritma *twofish*. *Sub* proses ini dilakukan sebelum enkripsi atau dekripsi algoritma *twofish* dijalankan. Inisialisasi kunci merupakan proses pengacakan kunci yang diinput manual oleh user untuk melakukan proses enkripsi dekripsi. Dengan demikian tingkat kerumitan proses enkripsi dekripsi meningkat.

Proses inisialisasi kunci dimulai dengan membagi kunci menjadi dua bagian yaitu kunci genap dan kunci ganjil. Lalu dilanjutkan dengan perhitungan kunci kotak S. Kunci kotak S akan digunakan untuk menghitung matrik MDS. Setelah kunci genap dan kunci ganjil diproses oleh PHT, maka akan terbentuk kumpulan kunci yang terjadwal. Kumpulan kunci ini akan digunakan untuk proses enkripsi atau dekripsi.

G. Diagram Alir *Sub* Proses Enkripsi Berkas Algoritma *Twofish*



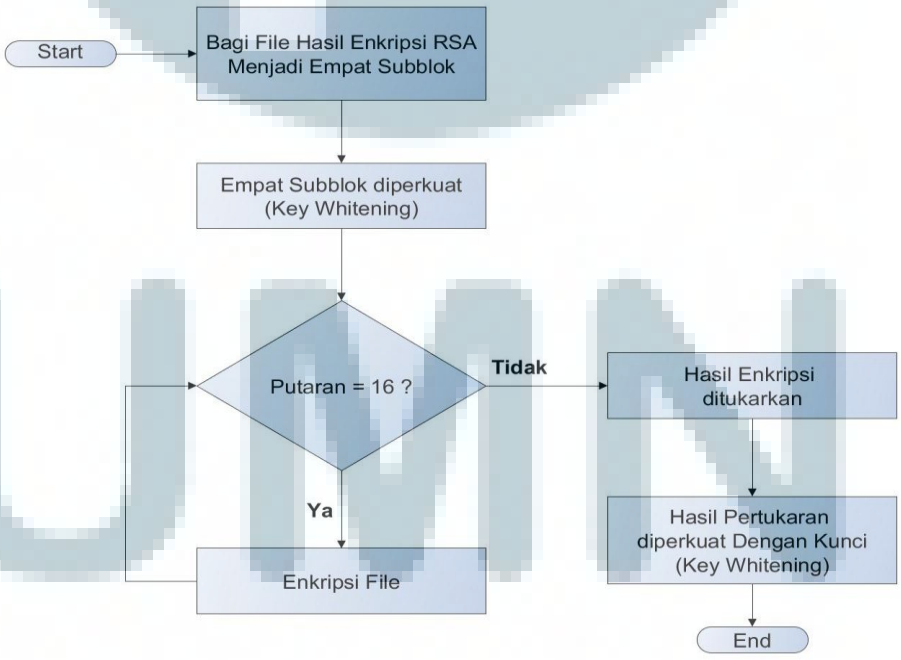
Gambar 3.7 Diagram Alir Proses Enkripsi Berkas Algoritma *Twofish*

Diagram ini menjelaskan *sub* proses pengolahan file sebelum file dienkripsi oleh algoritma *twofish*. Sistem menghasilkan *Initialization Vector* (IV) untuk proses enkripsi, kemudian menciptakan kunci berdasarkan angka atau huruf yang dimasukkan oleh pengguna dan memulai proses inisialisasi kunci. Data file yang belum dienkripsi akan di-XOR-kan dengan IV. Input kunci yang dimasukkan oleh pengguna akan diproses oleh fungsi inisialisasi kunci, agar dapat digunakan oleh proses enkripsi algoritma *twofish*.

H. Diagram Alir Proses Enkripsi Algoritma *Twofish*

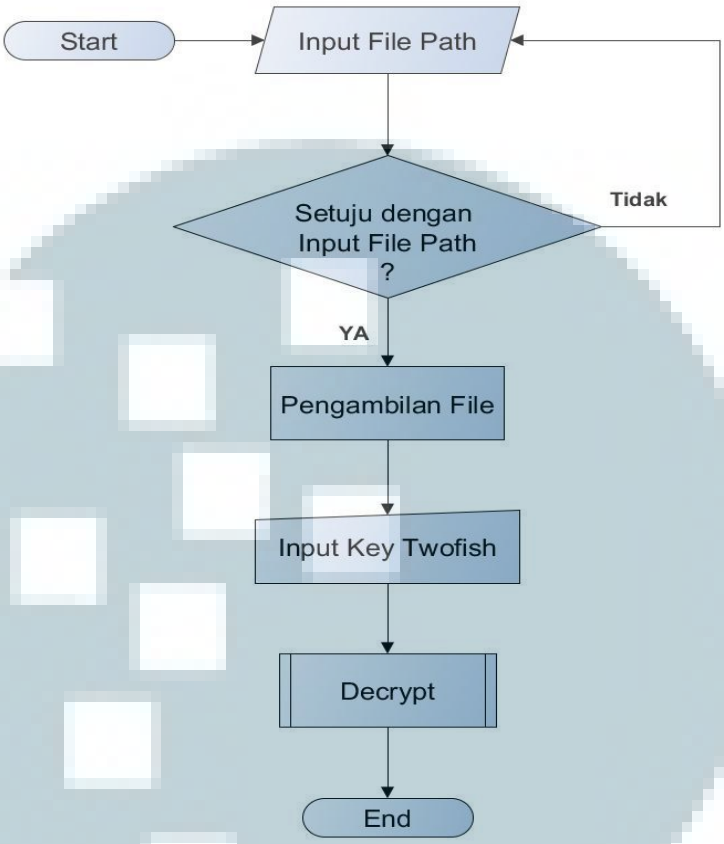
Diagram pada gambar 3.9 menjelaskan tentang proses enkripsi berkas dengan algoritma *twofish*. Proses dijalankan, setelah berkas hasil proses enkripsi algoritma RSA dibuat. Proses enkripsi algoritma *twofish* dimulai dengan membagi data file menjadi empat *sublock*. Empat *sublock* tersebut akan diperkuat dengan teknik *key whitening* (XOR). Teknik tersebut meng-XOR empat *subblock* dengan empat kunci yang dihasilkan dari 128 bits *subkey* pada putaran pertama.

Lalu proses enkripsi file dilakukan sebanyak 16 putaran. Setelah putaran selesai hasil enkripsi akan dipertukarkan dan hasil tersebut akan diperkuat dengan teknik *key whitening* (XOR). Teknik ini meng-XOR empat *subblock* hasil enkripsi dengan empat kunci yang dihasilkan dari 128 bits *subkey* pada putaran terakhir.



Gambar 3.8 Diagram Alir Proses Enkripsi *Twofish*

I. Diagram Alir Menu Dekripsi

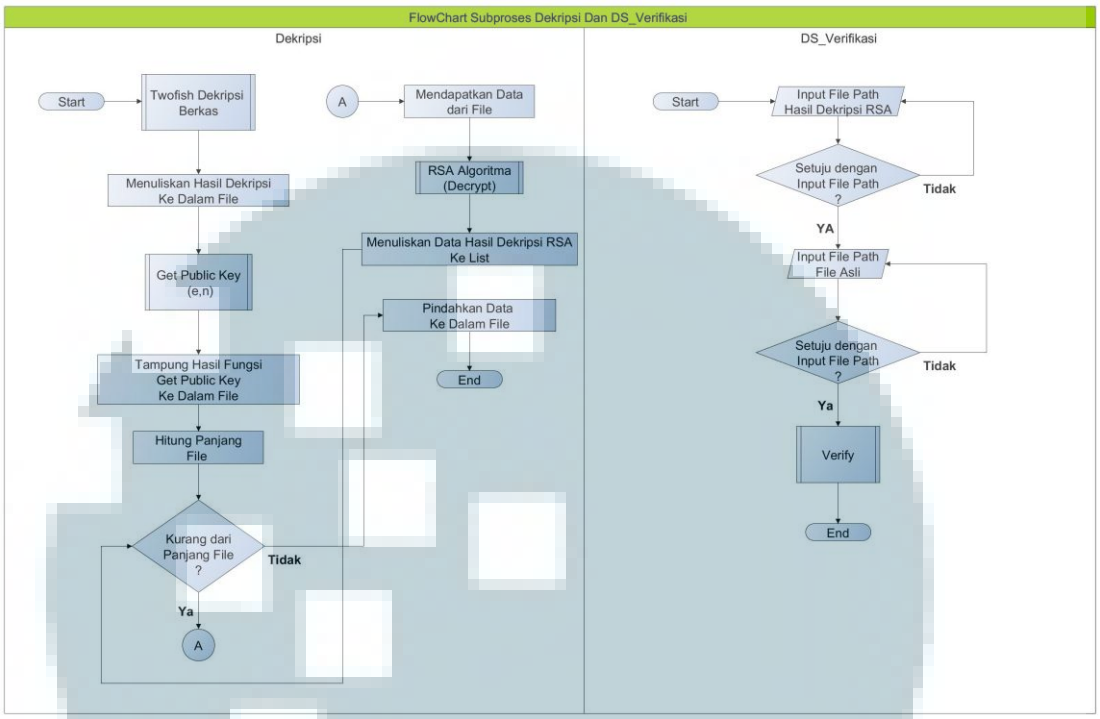


Gambar 3.9 Diagram Alir Menu Dekripsi

Diagram ini menjelaskan langkah – langkah pada menu DS_Decryption. Seperti pada menu DS_Encryption Pengguna diarahkan untuk melakukan proses dekripsi sesuai dengan langkah – langkah yang sudah ditetapkan. Pengguna wajib untuk mengisi *input* alamat berkas yang ingin didekripsi file. Setelah itu pengguna dapat mengisi *Input Key Twofish*. Lalu tombol decrypt dapat diakses untuk melanjutkan proses dekripsi

.

J. Diagram Alir *Sub* Proses Dekripsi Dan Menu Verifikasi



Gambar 3.10 Diagram Alir *Sub* Proses Dekripsi Dan Menu Verifikasi

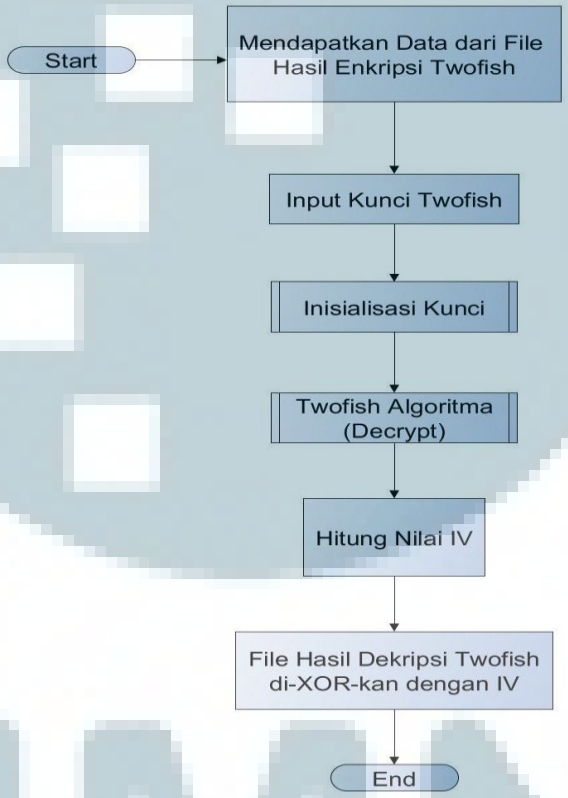
Diagram ini menjelaskan hubungan antara dua diagram alir yaitu *Sub* proses dekripsi dan DS_Verifikasi. Pada diagram *sub* proses dekripsi menjelaskan tentang proses dekripsi secara keseluruhan, di mana proses dekripsi melibatkan proses dekripsi dari dua algoritma yaitu *twofish* dan RSA. Hasil dari proses dekripsi algoritma *twofish* berupa hasil proses enkripsi algoritma RSA ditulis ke dalam file *internal* aplikasi.

Kemudian *sub*proses *Get Public Key* dijalankan untuk mendapatkan kunci publik dan data yang berisi hasil enkripsi file oleh algoritma RSA. Kunci publik tersebut digunakan untuk proses dekripsi algoritma RSA, sedangkan data hasil enkripsi algoritma RSA ditampung dalam sebuah file

internal aplikasi (*cipherFile*). File *cipherFile* dijadikan input untuk memulai proses dekripsi RSA.

Pada diagram alir *DS_Verifikasi*, pengguna memasukkan berkas hasil dekripsi RSA yang ada dalam *filename.dectrsa* dan berkas asli menjadi input untuk diproses oleh *sub* proses *verify*.

K. Diagram Alir *Sub* Proses Dekripsi Berkas Algoritma *Twofish*

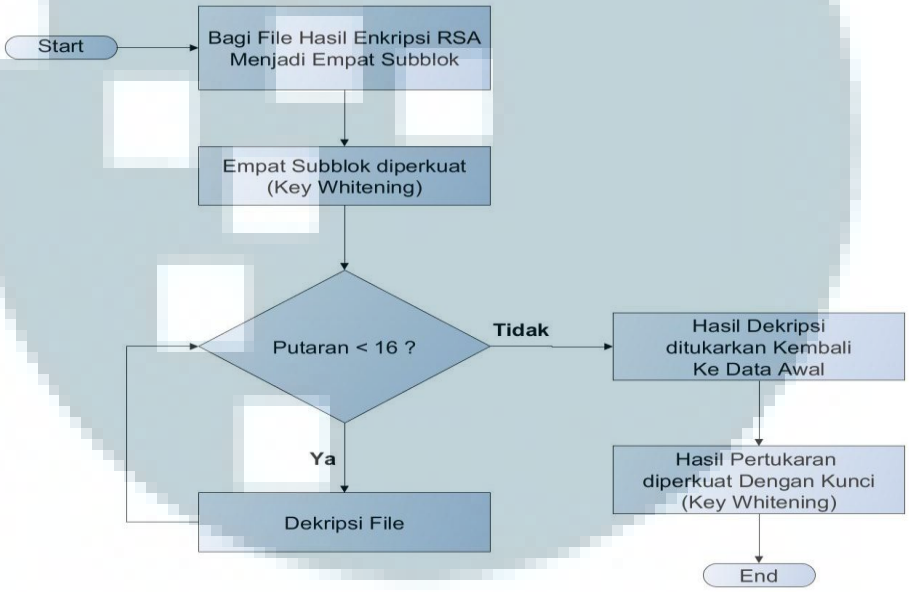


Gambar 3.11 Diagram Alir *Sub* Proses Dekripsi Berkas Algoritma *Twofish*

Diagram ini menjelaskan *sub*proses pengolahan file sebelum file didekripsi oleh algoritma *twofish*. Sistem menghasilkan *Initialization Vector* (IV) untuk proses dekripsi.

Kemudian menciptakan kunci berdasarkan angka atau huruf yang dimasukkan oleh pengguna dan memulai proses inisialisasi kunci. Data file yang sudah didekripsi akan di-XOR-kan dengan IV. Input kunci yang dimasukkan oleh pengguna akan diproses oleh fungsi inisialisasi kunci, agar dapat digunakan oleh proses dekripsi algoritma *twofish*.

L. Diagram Alir Proses Dekripsi Algoritma *Twofish*

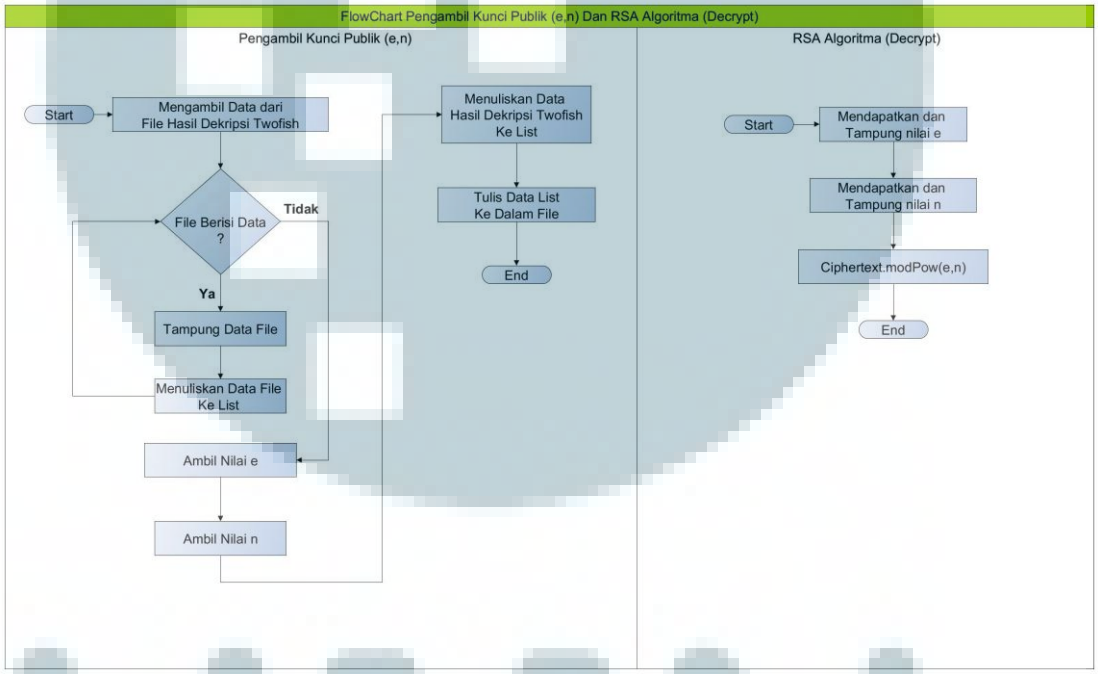


Gambar 3.12 Diagram Alir Proses Dekripsi Algoritma *Twofish*

Diagram pada gambar 3.12 menjelaskan tentang proses dekripsi berkas dengan algoritma *twofish*. Proses dijalankan, setelah file hasil proses enkripsi algoritma *twofish* dibuat. Proses dekripsi algoritma *twofish* dimulai dengan membagi data file menjadi empat *subblock*. Empat *subblock* tersebut akan diperkuat dengan teknik *key whitening* (XOR). Teknik tersebut meng-XOR empat *subblock* dengan empat kunci yang dihasilkan dari 128 bits *subkey* pada putaran pertama.

Lalu proses dekripsi file dilakukan sebanyak 16 putaran. Setelah putaran selesai hasil dekripsi akan ditukarkan dan hasil tersebut akan diperkuat dengan teknik *key whitening* (XOR). Teknik ini meng-XOR empat *subblock* hasil dekripsi dengan empat kunci yang dihasilkan dari 128 bits *subkey* pada putaran terakhir.

M. Diagram Alir Proses Pengambilan Kunci Publik RSA Dan Proses Dekripsi Algoritma RSA

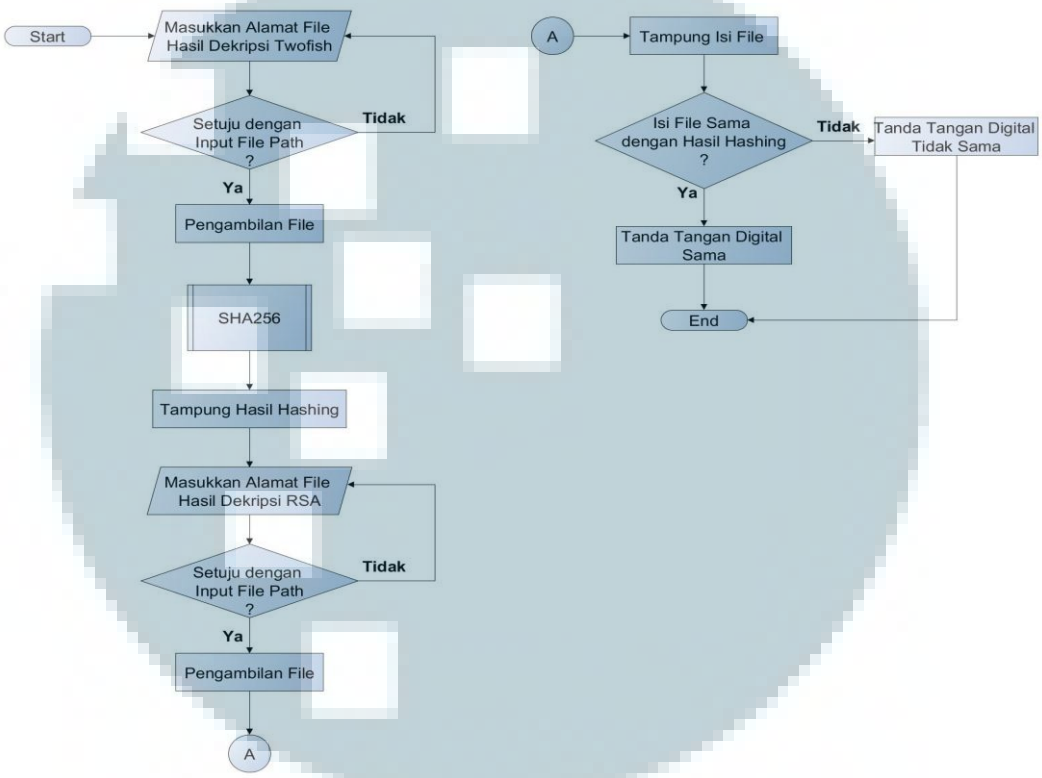


Gambar 3.13 Diagram Alir Proses Pengambilan Kunci Publik RSA Dan Proses Dekripsi Algoritma RSA

Diagram di atas menjelaskan proses pengambilan parameter pembentuk kunci publik RSA (e,n) pada berkas hasil dekripsi algoritma *twofish* dan data lainnya. Data lainnya merupakan hasil dari enkripsi RSA

terhadap *digest file* dimasukkan ke dalam file *internal* aplikasi. File tersebut akan menjadi input untuk proses dekripsi algoritma RSA.

N. Diagram Alir *Sub* Proses Verifikasi



Gambar 3.14 Diagram Alir *Sub* Proses Verifikasi

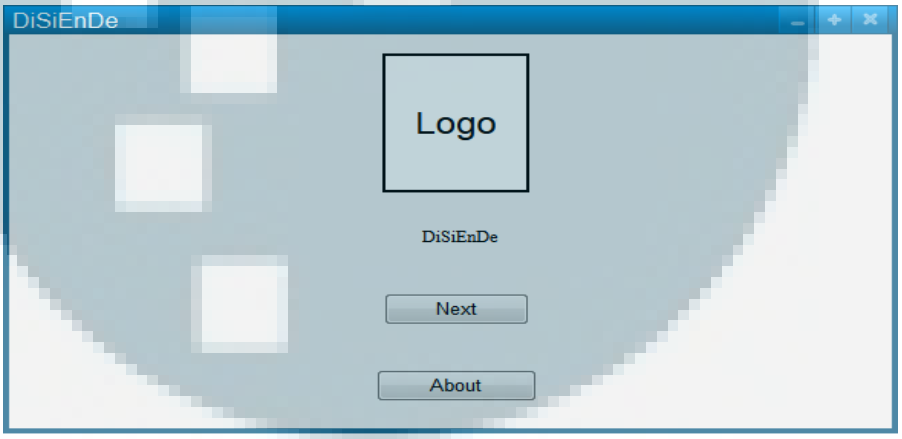
Diagram ini menjelaskan proses verifikasi dengan membandingkan *digest file* hasil proses dekripsi algoritma RSA dengan *digest file* dari hasil proses *hashing* algoritma SHA256 terhadap file asli yang dimasukkan oleh pengguna. Hasil dari proses ini berupa pembuktian apakah *digest file* dari file asli sama dengan *digest file* hasil proses dekripsi.

3.3 Desain Antarmuka

Pada subbab ini ditampilkan sketsa – sketsa antarmuka dari aplikasi yang dibuat.

3.3.1 Halaman Pembuka (*Welcome Screen*)

Halaman ini adalah halaman pertama kali ditampilkan ketika pengguna mengakses aplikasi. Pada bagian atas terdapat logo dari aplikasi ini. Dibawahnya terdapat nama dari aplikasi. Lalu terdapat dua tombol yaitu tombol *next* untuk masuk ke menu utama dan tombol *about* untuk menjelaskan tentang aplikasi.

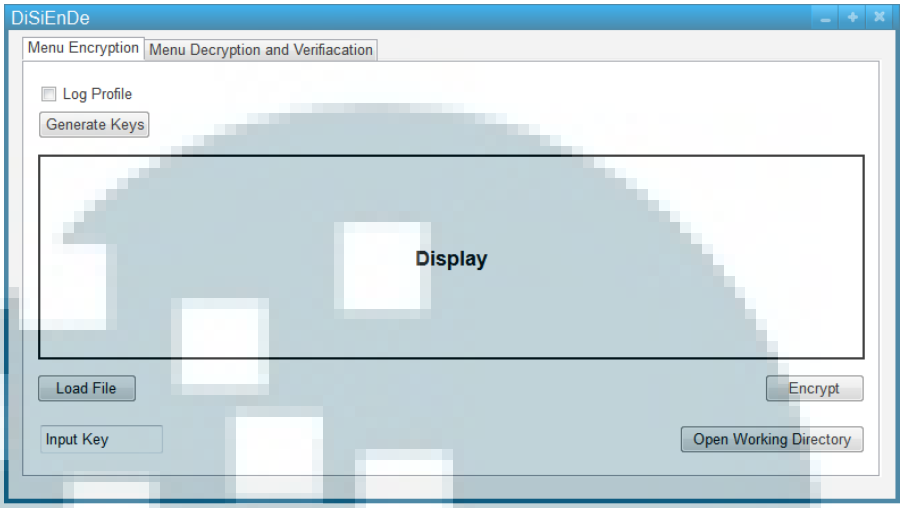


Gambar 3.15 Desain Halaman Pembuka

3.3.2 Halaman Menu Utama

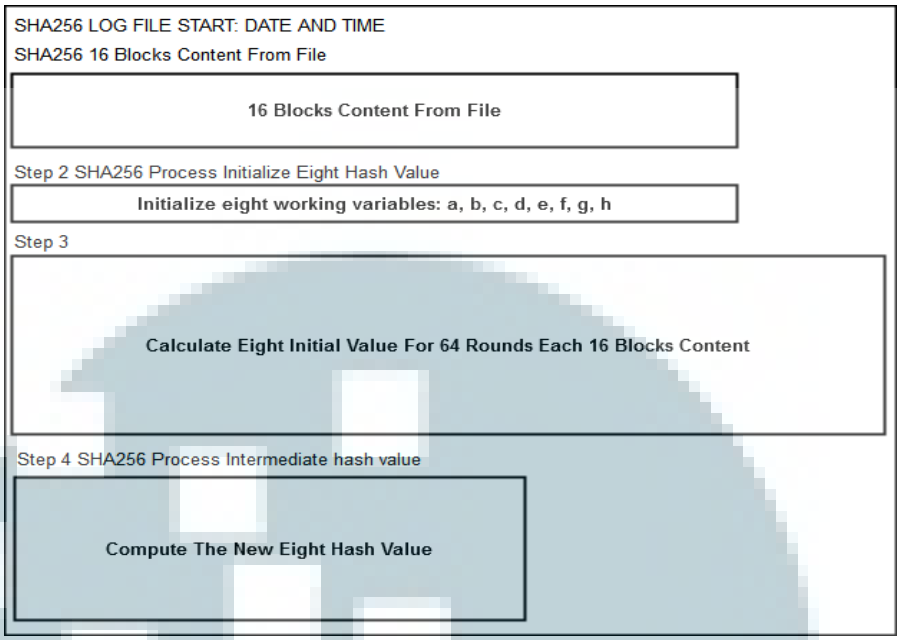
Halaman ini dapat diakses setelah pengguna mengakses tombol menu pada halaman pembuka. Halaman menu utama dibagi menjadi dua bagian yaitu menu *encryption* dan menu *decryption*. Menu *encryption* dapat ditampilkan setelah pengguna mengakses *tab* untuk menu tersebut seperti

yang terlihat pada gambar 3.16. Pada bagian pojok kiri atas terdapat *checkbox log profile*.

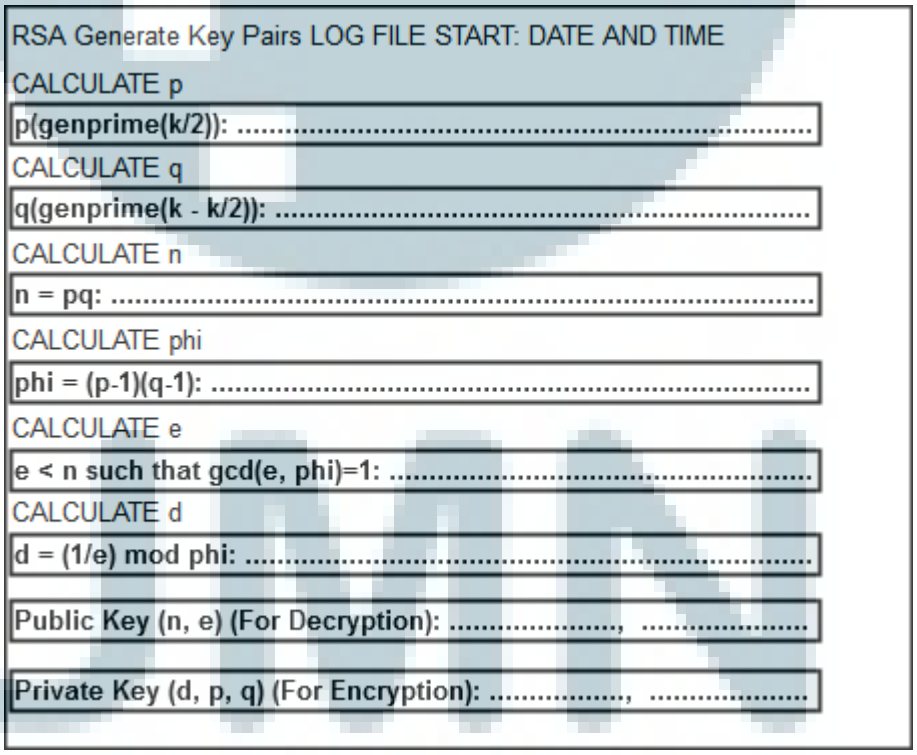


Gambar 3.16 Halaman Menu Utama *Encryption*

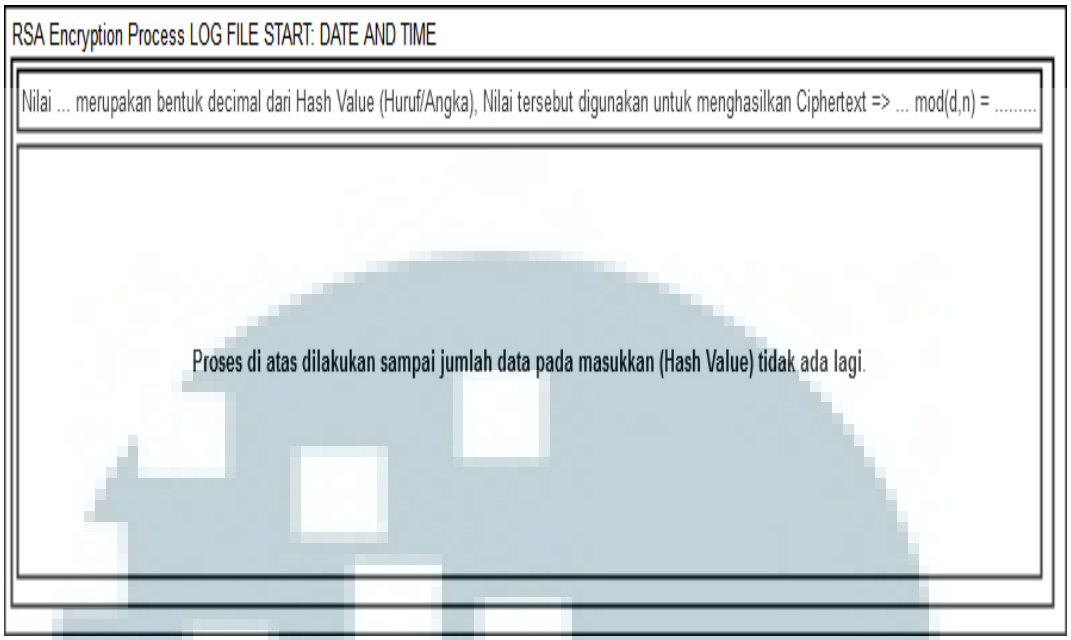
Ketika *checkbox* diklik, proses pencatatan dari proses *hashing* dan enkripsi dimulai. Hasil dari pencatatan ini dapat digunakan oleh pengguna untuk mempelajari proses *hashing* algoritma SHA256 dan enkripsi algoritma *twofish* dan RSA. Di bawah *checkbox*, terdapat tombol *generate keys* untuk menghasilkan parameter – parameter yang digunakan untuk proses enkripsi algoritma RSA. Proses pencatatan tersebut menghasilkan file – file yang mencatat langkah demi langkah proses *hashing* terhadap masukkan menjadi *hash value* oleh SHA256, dan proses enkripsi terhadap masukkan menjadi *ciphertext* oleh RSA dan *twofish*. File – file tersebut dapat dibuka dan memberikan informasi yang dapat dipelajari oleh pengguna. Desain untuk log profile proses masing – masing algoritma terlihat pada gambar 3.17, gambar 3.18, gambar 3.19, dan gambar 3.20..



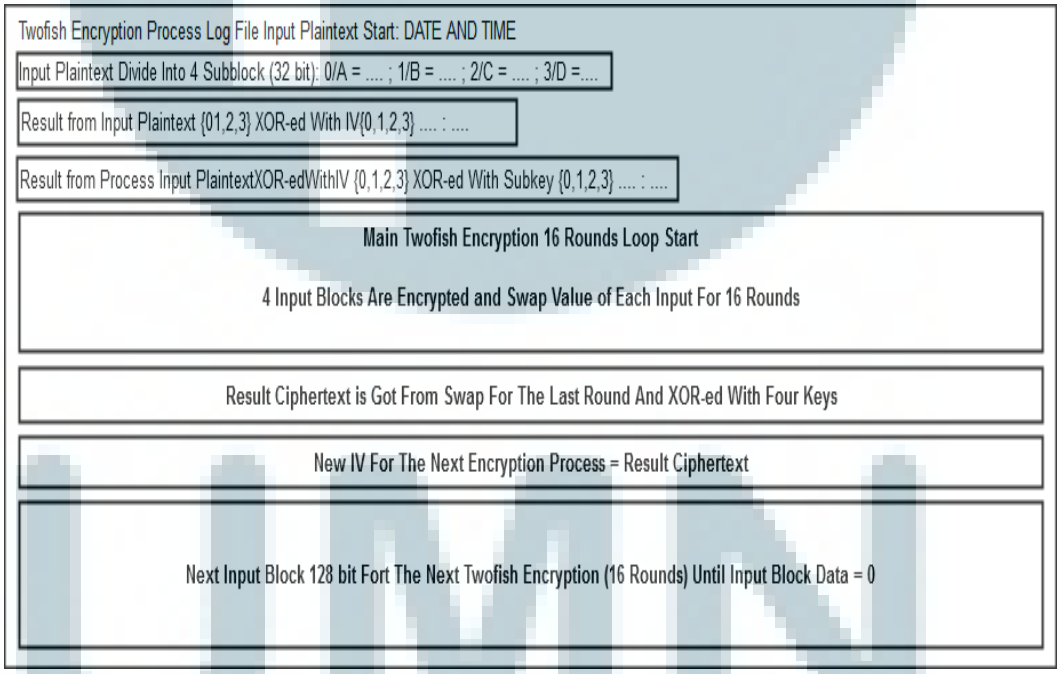
Gambar 3.17 Desain *Log Profile* Proses Hashing SHA256



Gambar 3.18 Desain *Log Profile* Proses *Generate Key* RSA



Gambar 3.19 Desain *Log Profile* Proses Enkripsi RSA

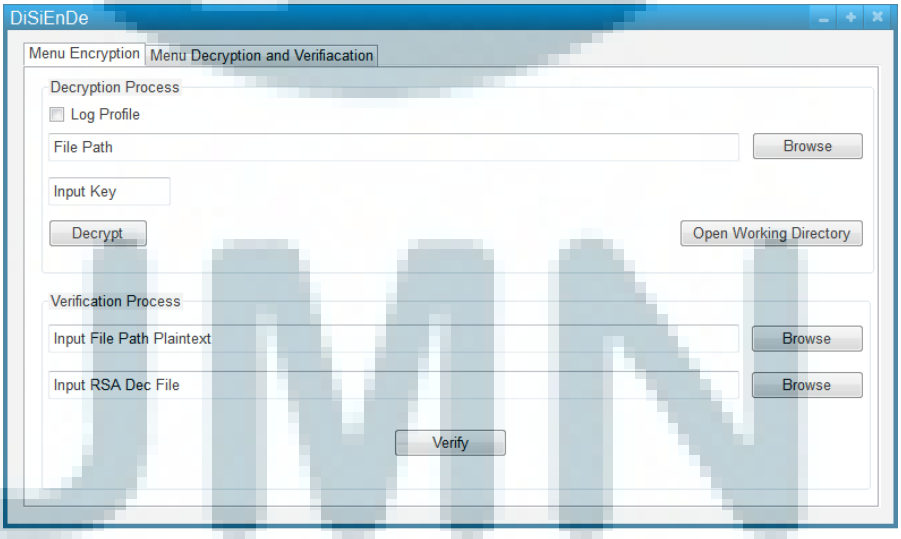


Gambar 3.20 Desain *Log Profile* Proses Enkripsi *Twofish*

Di bawah tombol *generate key* terdapat bagian untuk menampilkan parameter – parameter tersebut, memberitahukan pengguna apakah berkas sudah dimasukkan atau belum, dan waktu mulai dan selesai proses enkripsi.

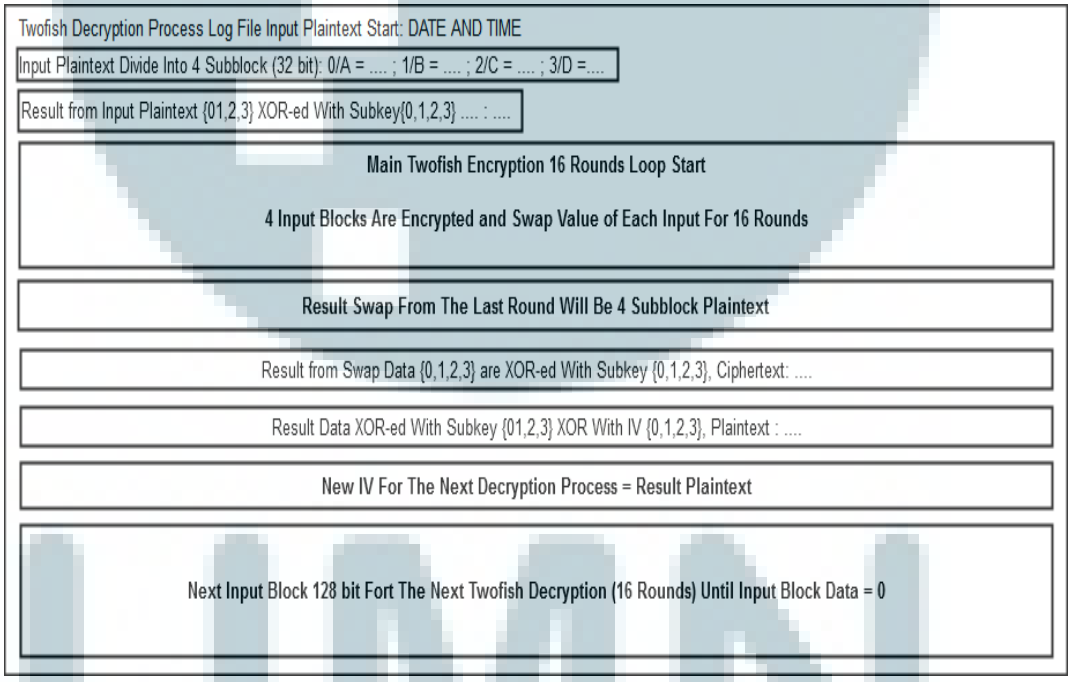
Pada pojok kiri bawah terdapat tombol untuk memasukkan berkas (*Load File*) dan bagian untuk menuliskan masukkan untuk kunci *twofish* (*Input Key*) berupa karakter – karakter huruf atau angka atau gabungan keduanya. Pada pojok kanan bawah terdapat dua tombol yaitu tombol untuk memulai proses enkripsi (*Encrypt*) dan tombol untuk membuka direktori (*Open Working Directory*) untuk menyimpan berkas – berkas hasil dari proses – proses yang ada pada aplikasi.

Pada menu *decryption* seperti yang dapat dilihat pada gambar 3.21. Pada menu ini dibagi menjadi dua bagian yaitu bagian untuk proses dekripsi dan proses verifikasi yang berada di bawah tombol dekripsi. Bagian untuk proses dekripsi terdapat *checkbox log profile*.

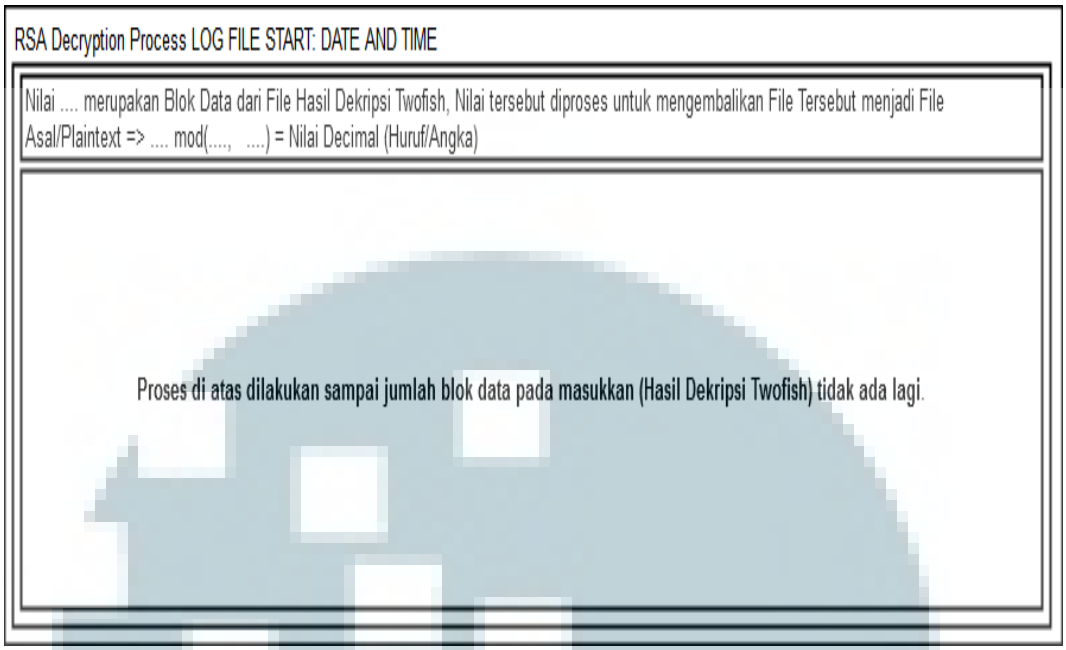


Gambar 3.21 Halaman Menu Utama *Decryption*

Ketika *checkbox* diklik, proses pencatatan dari proses dekripsi dimulai. Hasil dari pencatatan ini dapat digunakan oleh pengguna untuk mempelajari proses dekripsi algoritma *twofish* dan RSA. Tombol *load file* dan alamat dari berkas yang dipilih ditampilkan di bagian alamat berkas (*file path*). Proses pencatatan tersebut menghasilkan file – file yang mencatat langkah demi langkah proses dekripsi terhadap *ciphertext* menjadi *plaintext* oleh *twofish* dan RSA. File – file tersebut dapat dibuka dan memberikan informasi yang dapat dipelajari oleh pengguna. Desain untuk log profile proses masing – masing algoritma terlihat pada gambar 3.22 dan gambar 3.23..



Gambar 3.22 Desain *Log Profile* Proses Dekripsi *Twofish*



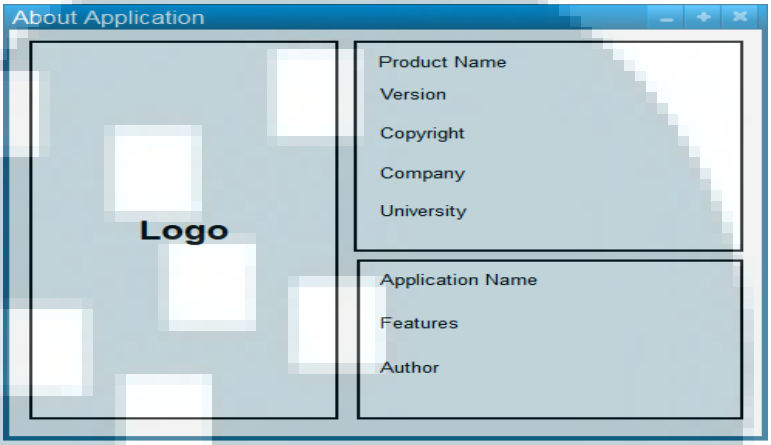
Gambar 3.23 Desain *Log Profile* Proses Dekripsi RSA

Pada bagian bawah terdapat bagian untuk menuliskan masukkan untuk kunci *twofish* berupa karakter – karakter angka atau huruf atau gabungan keduanya. Pengguna diberitahukan untuk memasukkan kunci yang sama dengan masukkan kunci pada proses enkripsi.

Pada bagian proses verifikasi terdapat dua buah tombol untuk mencari berkas hasil dekripsi proses algoritma RSA dan berkas asli. Setelah kedua alamat berkas didapatkan, kedua alamat tersebut ditampilkan di bagian alamat berkas. Lalu pengguna dapat mengakses proses verifikasi dan bukti dari verfikasi berupa asli atau tidaknya dimunculkan pada layar menu setelah pengguna mengakses proses verifikasi melalu tombol *verify*.

3.3.3 Halaman Penjelasan Tentang Aplikasi

Halaman ini dapat diakses melalui tombol *about* pada halaman pembuka. Halaman ini menampilkan hal – hal seputar aplikasi dan data tentang penulis seperti yang dapat dilihat pada gambar 3.24.



Gambar 3.24 Halaman Penjelasan Tentang Aplikasi

3.4 Skenario Pengujian

Pengujian yang dilakukan untuk mencapai tujuan dari penelitian diurutkan sebagai berikut

1. Melakukan uji fungsionalitas aplikasi

   Pengujian dilakukan dengan menjalankan aplikasi untuk mengenkripsi atau mendekripsi suatu file. Fungsi pencatatan juga akan diuji. Fungsi tersebut akan dijalankan bersamaan dengan proses enkripsi atau dekripsi.

2. Membuat variasi ukuran file yang sesuai untuk penelitian

Pembuatan variasi ukuran file yang diinginkan didapatkan dengan menggunakan piranti lunak yaitu *parkdale* 2.93 untuk membuat file dengan ukuran maksimum 50 MB.

3. Pengujian waktu dan memori untuk proses enkripsi dan dekripsi

   Pengujian ini dilakukan dengan menggunakan piranti lunak *ANTS Performance Profilers* 5 dan *YourKit Profilers* 2014 untuk mendapatkan data pengujian berupa waktu dan memori yang dibutuhkan oleh aplikasi untuk mengenkripsi atau mendekripsi file dengan pembagian pengujian ukuran file mulai dari 5, 10 ,15, 20, 25, 30, 35, 40, 45, 50 (MB).