



### **Hak cipta dan penggunaan kembali:**

Lisensi ini mengizinkan setiap orang untuk menggubah, memperbaiki, dan membuat ciptaan turunan bukan untuk kepentingan komersial, selama anda mencantumkan nama penulis dan melisensikan ciptaan turunan dengan syarat yang serupa dengan ciptaan asli.

### **Copyright and reuse:**

This license lets you remix, tweak, and build upon work non-commercially, as long as you credit the origin creator and license it on your new creations under the identical terms.

## BAB III

### ANALISIS DAN PERANCANGAN APLIKASI

#### 3.1 Analisis Aplikasi

##### 3.1.1 Spesifikasi Umum Kebutuhan Sistem

Pada penelitian ini, akan dibangun sebuah aplikasi *listening post* yang berbeda dengan *listening post* yang sudah ada pada umumnya. Aplikasi *listening post* ini akan menggunakan sebuah *database* yang berfungsi sebagai media penyimpanan dan pada aplikasi ini juga terdapat *music player* yang dapat memutar *file* audio yang terdapat pada *database* tersebut. Selain penggunaan *database*, aplikasi juga menggunakan algoritma kriptografi Blowfish yang berguna untuk melakukan proses enkripsi dan dekripsi pada *file* audio yang terdapat dalam *database* sehingga *file* audio memiliki tingkat keamanan yang baik dan dapat mengurangi pembajakan.

Aplikasi ini membutuhkan suatu *database* yang berguna untuk menyimpan data-data mengenai *file* audio yang akan dimasukkan kedalam *database* tersebut. *File* audio yang masuk kedalam *database* akan mengalami proses enkripsi sehingga *file* audio tersebut tidak dapat diputar pada *music player* pada umumnya. *File* audio dienkripsi menggunakan algoritma Blowfish, karena algoritma Blowfish memiliki tingkat kecepatan yang cukup cepat dalam melakukan proses enkripsi dan dekripsi dibandingkan dengan algoritma lainnya. *File* audio yang dimasukkan melalui aplikasi akan ditampilkan dalam bentuk *list*, sehingga *user*

atau pengguna dapat memilih untuk memutar lagu-lagu yang telah disimpan dalam *database* tersebut.

Untuk memutar lagu yang telah dipilih oleh user atau pengguna diperlukan sebuah *music player* pada aplikasi ini. Akan tetapi, *music player* pada aplikasi ini melakukan proses dekripsi pada saat user memilih salah satu lagu. Proses dekripsi pada aplikasi ini tidak menghasilkan *file* audio yang dapat diputar di seluruh *music player*, akan tetapi proses dekripsi pada aplikasi ini menghasilkan *byte array* data yang akan diputar menggunakan teknik *buffering*. Karena proses dekripsi dengan menggunakan algoritma Blowfish yang memiliki kelebihan pada kecepatannya dalam melakukan kriptografi, maka teknik *buffering* akan menjadi teknik yang sangat tepat dalam memutar lagu menggunakan data berupa *byte array*. Dengan menggunakan teknik *buffering* ini diharapkan dapat mengurangi adanya pembajakan dengan melakukan pencurian *file* audio yang telah mengalami proses dekripsi.

Guna memudahkan *user* atau pengguna dalam mencari dan memilih lagu untuk diputar, maka aplikasi ini membutuhkan fitur pencarian dan pengurutan *file* audio. Dengan adanya kedua fitur tersebut dapat mempercepat *user* atau pengguna dalam menentukan *file* audio mana yang akan diputar sesuai dengan keinginannya.

Dari seluruh kebutuhan-kebutuhan sistem diatas, maka dapat disimpulkan aplikasi akan terdiri dari beberapa fitur yang harus dipenuhi untuk memenuhi kebutuhan-kebutuhan tersebut. Fitur-fitur tersebut secara terperinci berupa:

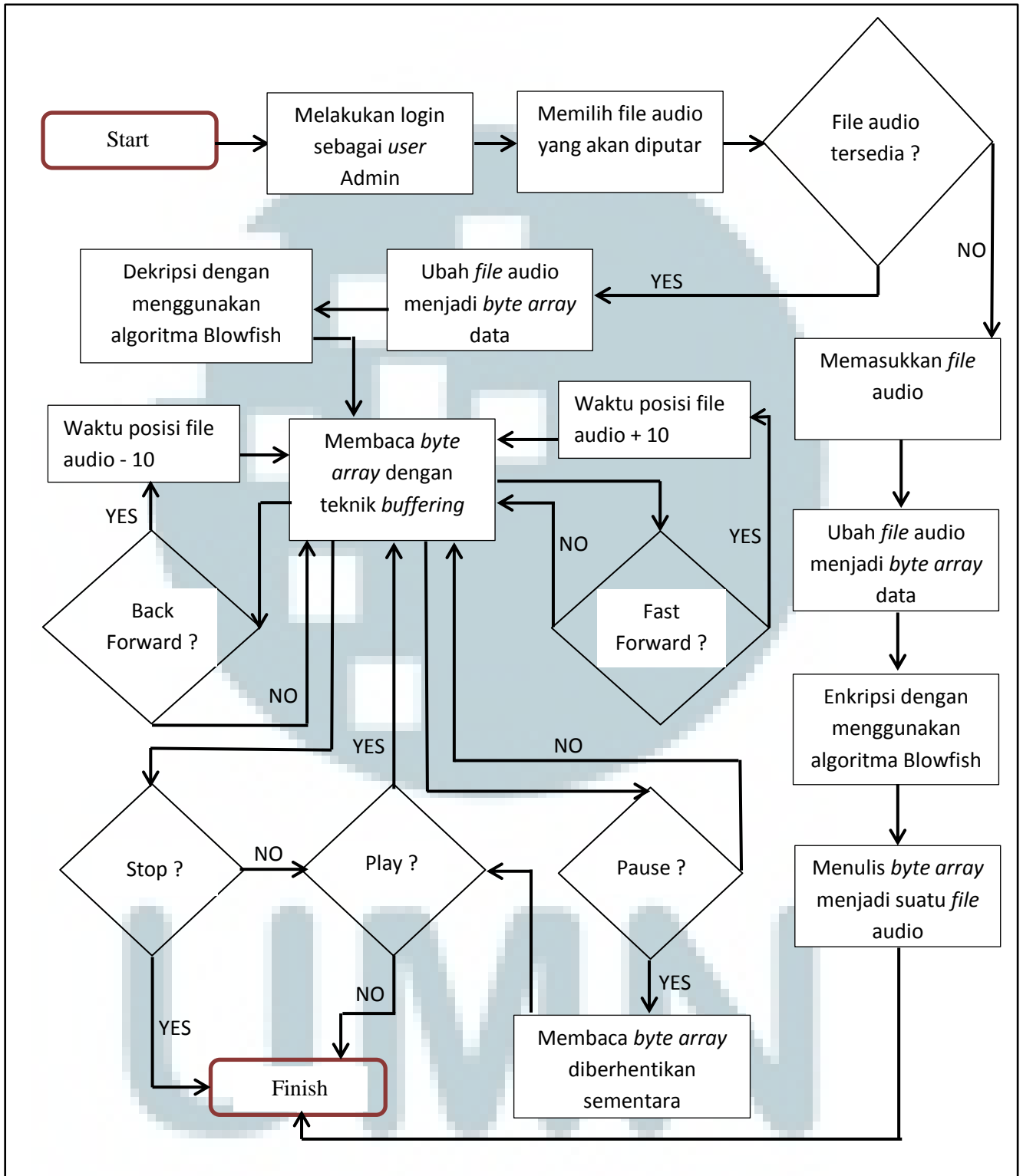
1. Fitur memasukkan *file* audio ke dalam *database* atau yang biasa disebut dengan *upload* *file* audio ke dalam *database* yang hanya dapat dilakukan oleh *user* admin.
2. Fitur pencarian atau *search file* audio untuk mempermudah pencarian *file* audio yang akan diputar
3. Fitur pengurutan atau *sort file* audio untuk mempermudah pencarian *file* audio yang akan diputar
4. Fitur *music player* yang berguna untuk memutar *file* audio yang dipilih oleh *user* atau pengguna, mengatur *volume* suara dari suatu *file* audio, mengatur *pan* dari suatu *file* audio, serta mengetahui posisi durasi *file* audio selama *file* audio diputar.
5. Fitur *List Added Songs* yang berguna untuk menampilkan *file-file* audio yang telah di *upload* ke dalam *database*, sehingga *user* atau pengguna dapat memilih salah satu *file* audio tersebut untuk diputar.
6. Fitur *Help* yang berfungsi untuk membantu *user* atau pengguna dalam mengetahui cara menggunakan aplikasi ini dengan optimal, sehingga *user* tidak merasa bingung dalam menjalankan aplikasi ini.

### 3.1.2 Perancangan Sistem

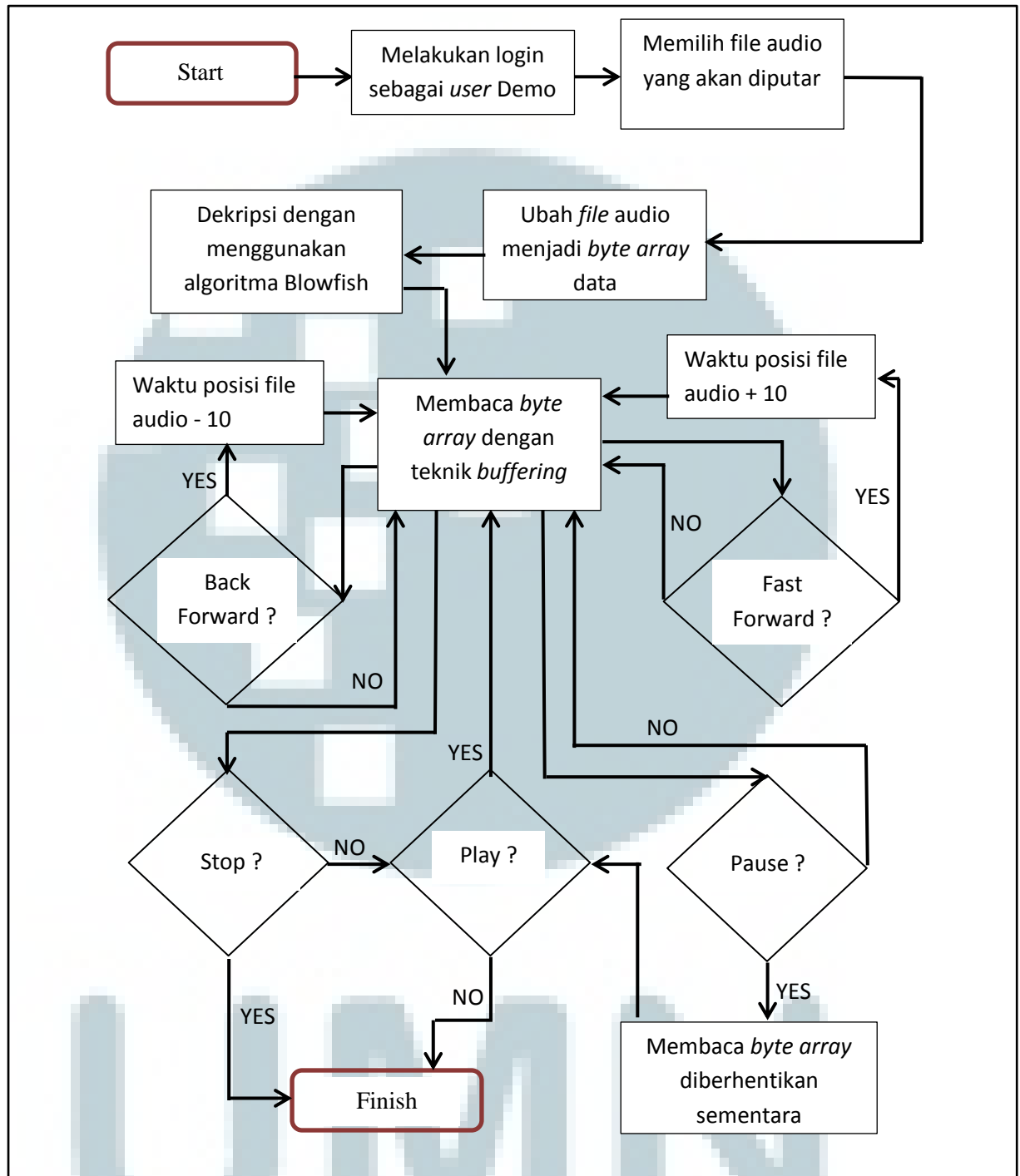
Pada penelitian ini, *user* dibagi kedalam dua jenis, yaitu admin dan demo dimana masing-masing jenis *user* memiliki *level* akses yang berbeda. Pada *user* admin, selain dapat melakukan pencarian dan pemutaran *file* audio yang terdapat dalam *database*, *user* admin juga dapat melakukan penambahan *file* audio ke

dalam *database* apabila *file* audio tersebut tidak ada dalam *database*. Sedangkan pada *user* demo hanya dapat melakukan pencarian dan pemutaran *file* audio dan tidak dapat melakukan penambahan *file* audio ke dalam *database*.

Proses kriptografi pada aplikasi ini menggunakan algoritma Blowfish dimana pada proses enkripsi *file* audio akan disimpan dalam *database* dan *user* atau pengguna akan memilih *file* audio yang sudah terenkripsi untuk diputar. Pada proses dekripsi *file* audio tidak ditulis dalam bentuk *file*, akan tetapi *file* audio diubah menjadi *byte array* data dan diputar menggunakan teknik *buffering/streaming*. Jika *user* atau pengguna melakukan proses *pause* dengan menekan *button pause*, maka proses membaca *byte array* akan diberhentikan secara sementara. Proses membaca *byte array* akan dilanjutkan apabila *user* atau pengguna melakukan proses *play* dengan menekan *button play*. Apabila *user* melakukan proses *stop* dengan menekan *button stop*, maka proses membaca *byte array* diberhentikan dan dihapus, sehingga ketika *user* atau pengguna melakukan proses *play* kembali, maka *byte array* pada *file* audio tersebut akan dibaca dari awal. Selain ketiga *button* tersebut, terdapat *button fast forward* dan *back forward* yang memiliki fungsi masing-masing. Jika *user* menekan *button fast forward*, maka posisi waktu *file* audio pada waktu diputar akan ditambah sebanyak sepuluh detik dari durasi *total file* audio tersebut. Sedangkan pada *button back forward*, posisi waktu *file* audio yang diputar akan dikurangi sebanyak sepuluh detik dari durasi *total file* audio tersebut. Berikut adalah *system flow* kedua *user* yang menggambarkan cara kerja aplikasi ini.



Gambar 3.1 System Flow User Admin Aplikasi dTunez Studio



Gambar 3.2 System Flow User Demo Aplikasi dTunez Studio

Dari *system flow* pada gambar 3.1, dapat dilihat bahwa setelah melakukan login sebagai admin, selanjutnya *user* dapat melakukan pemilihan *file* audio sesuai dengan keinginan *user*. Apabila *file* audio yang akan diputar tersedia, maka *file* audio tersebut dapat langsung diputar, sedangkan jika *file* audio yang akan diputar tidak tersedia, maka *user* atau pengguna akan memasukkan *file* audio terlebih dahulu ke dalam *database*. Dari gambar 3.1, dapat dilihat bahwa proses enkripsi *file* audio dimulai dengan memasukkan *file* audio terlebih dahulu. Ketika memasukkan *file* audio, data-data dalam *file* audio tersebut akan dimasukkan ke dalam *database*. *File* audio yang telah dimasukkan akan diubah menjadi *byte array* data dimana data tersebut akan mengalami proses enkripsi dengan menggunakan algoritma Blowfish. Setelah proses enkripsi selesai, *byte array* data hasil enkripsi ditulis kembali menghasilkan *file* audio yang terenkripsi. Setelah *file* audio baru yang terenkripsi selesai dibuat, maka proses memasukkan *file* audio ke dalam *database* telah selesai, dan *file* audio sudah siap untuk diputar. Pada waktu *user* atau pengguna memilih *file* audio yang akan diputar, maka *file* audio yang telah dienkripsi akan diubah kembali menjadi *byte array* data dan mengalami proses dekripsi dengan menggunakan algoritma yang sama yaitu algoritma Blowfish. Kali ini *byte array* hasil proses dekripsi tidak ditulis untuk menghasilkan *file* audio yang terdekripsi, melainkan *byte array* data tersebut secara langsung dibaca dengan menggunakan teknik *buffering*. Dengan menggunakan teknik *buffering* diharapkan dapat mengurangi tindakan pembajakan dengan melakukan pencurian *file* audio hasil proses dekripsi. Proses membaca *byte array* akan berhenti apabila proses *stop* dijalankan. Proses membaca



*byte array* akan berhenti sementara apabila proses *pause* dijalankan, jika ingin melanjutkan proses membaca *byte array* maka proses *play* dapat dijalankan sehingga proses membaca *byte array* yang tadinya berhenti akan dilanjutkan kembali. Selama *file* audio diputar, *user* dapat melakukan *fast forward* atau *back forward*, dimana *fast forward* berfungsi untuk menambah waktu posisi *file* audio sebanyak sepuluh detik, sedangkan *back forward* berfungsi untuk mengurangi waktu posisi *file* audio sebanyak sepuluh detik.

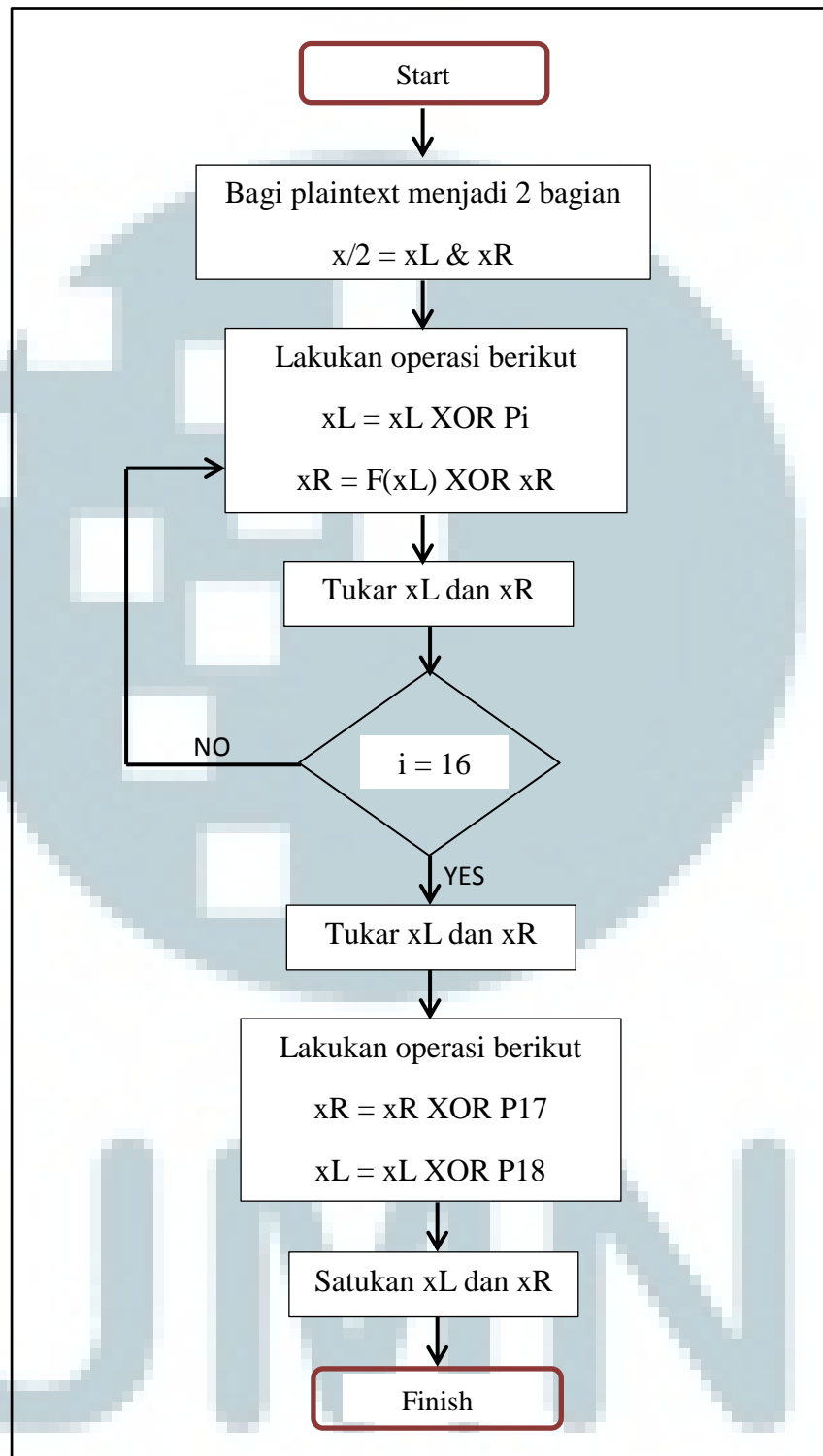
Pada *system flow* pada *user* demo yang ditunjukkan pada gambar 3.2 memperlihatkan letak perbedaan antara *user* admin dengan *user* demo yang terdapat pada fitur penambahan *file* audio. Pada *user* demo, *user* hanya dapat melakukan pemilihan dan pemutaran *file* audio yang telah dipilih sesuai dengan keinginan *user*. Pemutaran *file* audio pada *user* admin dan demo memiliki proses yang sama, dimana *file* audio yang telah dienkripsi akan diubah menjadi *byte array* data yang akan mengalami proses dekripsi. Selanjutnya *byte array* data hasil dekripsi tersebut akan dibaca menggunakan teknik *buffering* sehingga proses dekripsi ini tidak menghasilkan *file* audio baru.

UMMN

Algoritma Blowfish yang digunakan dalam proses kriptografi *file* audio pada aplikasi ini merupakan algoritma yang menerapkan jaringan Feistel, dimana terdiri dari 16 putaran. Dengan input adalah 64 bit elemen data. Alur algoritma Blowfish dapat dijelaskan sebagai berikut.

1. Bentuk inisial *P-array* sebanyak 18 buah ( $P_1, P_2, \dots, P_{18}$ ) masing-masing bernilai 32 bit.
2. Bentuk *S-box* sebanyak empat buah masing-masing bernilai 32 bit yang memiliki masukan 256.
3. *Plaintext* diambil sebanyak 64 bit, apabila kurang dari 64 bit maka tambahkan bitnya
4. Bagi menjadi dua bagian masing-masing terdiri dari 32 bit ( $x_L$  dan  $x_R$ )
5. Lakukan operasi  $x_L = x_L \text{ XOR } P_i$  dan  $x_R = F(x_L) \text{ XOR } x_R$
6. Hasil operasi ditukar, dimana  $x_L$  menjadi  $x_R$  dan  $x_R$  menjadi  $x_L$
7. Ulangi sebanyak 16 kali, dan pada perulangan 16 lakukan kembali penukaran  $x_L$  dan  $x_R$
8. Lakukan operasi  $x_R = x_R \text{ XOR } P_{17}$  dan  $x_L = x_L \text{ XOR } P_{18}$
9. Satukan kembali  $x_L$  dan  $x_R$  sehingga menjadi 64 bit kembali

Berikut adalah *flowchart* dari algoritma Blowfish.



Gambar 3.3 *Flowchart* Algoritma Blowfish

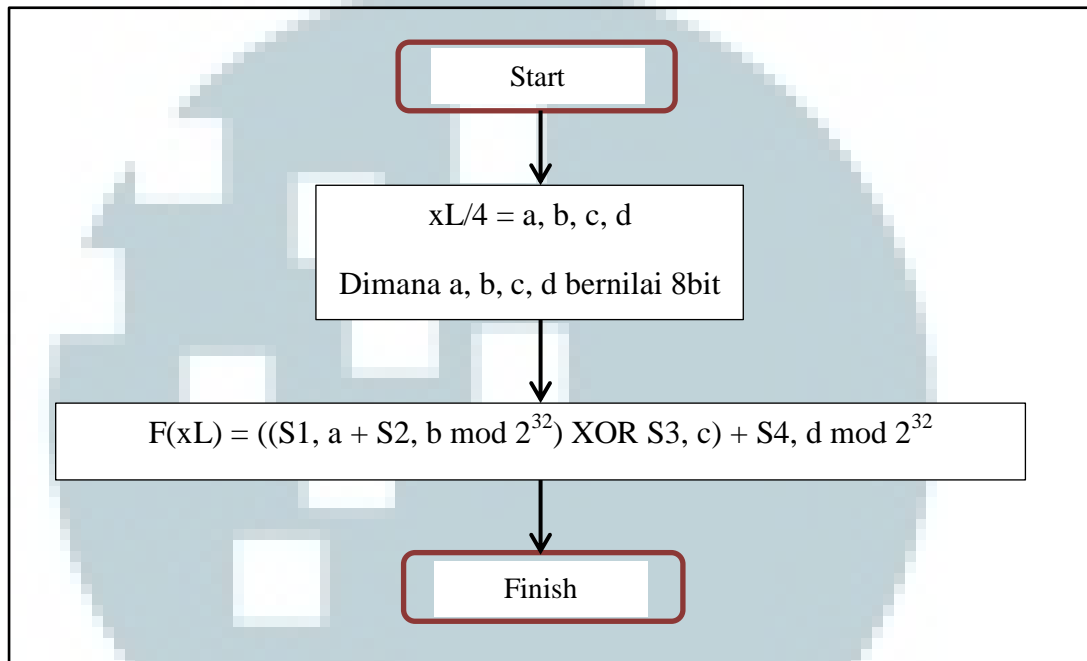
Jaringan Feistel yang digunakan pada algoritma Blowfish biasa disebut dengan fungsi F. Fungsi F adalah bagi XL menjadi empat bagian 8-bit: a,b,c dan d. Setelah itu, dilakukan perhitungan seperti berikut.

$$F(XL) = ((S1,a + S2,b \bmod 2^{32}) \text{ XOR } S3,c) + S4,d \bmod 2^{32}$$

Subkunci dihitung menggunakan algoritma Blowfish, dengan langkah-langkah sebagai berikut.

1. Inisialisasi P-array dan empat S-box secara berurutan dengan *string* yang tetap. *String* ini terdiri dari digit *hexadecimal* dari Pi
2. XOR P1 dengan 32 bit pertama kunci, XOR P2 dengan 32 bit kedua dari kunci dan seterusnya untuk setiap bit dari kunci (sampai 18). Ulangi terhadap bit kunci hingga seluruh P-array di XOR dengan bit kunci.
3. Enkripsi seluruh string nol dengan algoritma Blowfish dengan menggunakan subkunci seperti yang dijelaskan pada langkah satu dan dua
4. Ganti P1 dan P2 dengan keluaran dari langkah tiga
5. Enkrip keluaran dari langkah tiga dengan algoritma Blowfish dengan subkunci yang sudah dimodifikasi
6. Ganti P3 dan P4 dengan keluaran dari P5
7. Lanjutkan proses tersebut, ganti seluruh elemen dari P-array, kemudian seluruh keempat S-box berurutan, dengan keluaran yang berubah secara kontinu dari algoritma Blowfish

Untuk mempermudah pemahaman, berikut adalah flowchart fungsi F pada algoritma Blowfish.



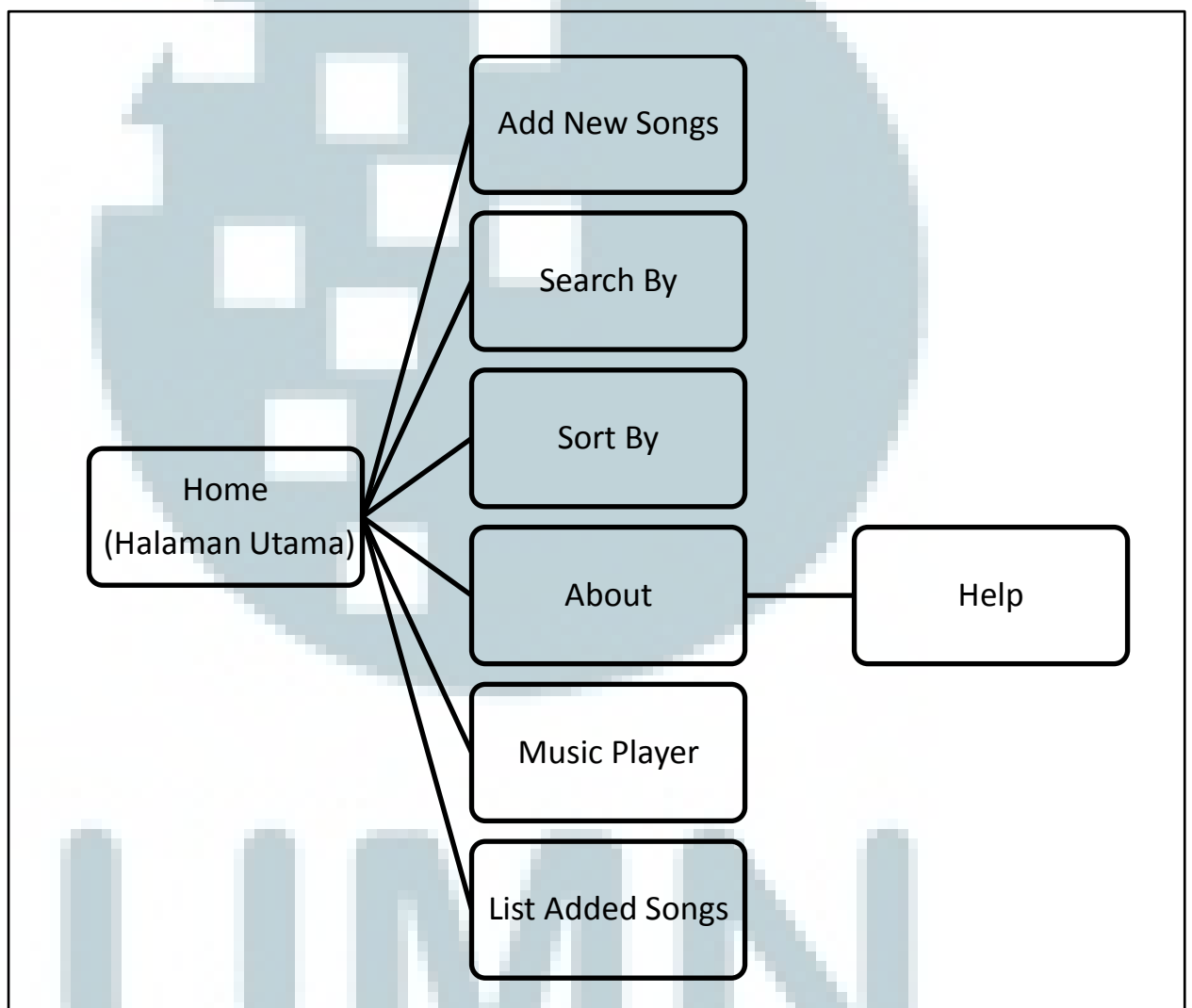
Gambar 3.4 *Flowchart* Fungsi F

U  
M  
M  
N

## 3.2 Perancangan Aplikasi

### 3.2.1 Fitur-Fitur Aplikasi

Berikut adalah fitur-fitur aplikasi yang dapat dipilih atau digunakan oleh pengguna atau *user* pada saat menjalankan aplikasi.



Gambar 3.5 Fitur-Fitur yang Terdapat pada Aplikasi

Pada halaman utama aplikasi (halaman pertama yang tampil saat pengguna menjalankan aplikasi) terdapat beberapa fitur dengan rincian masing – masing fitur adalah sebagai berikut.

1. *Add New Songs*

Pada fitur ini, *user* admin dapat menambahkan *file* audio dengan format .mp3 sehingga *file* audio tersebut akan disimpan kedalam *database*. Seluruh *file* audio yang telah disimpan dalam *database* akan ditampilkan pada *List Added Songs* yang menjadi pilihan bagi *user* ketika ingin mendengarkan salah satu dari *file* audio tersebut.

2. *Search By*

Pada fitur ini, *user* admin dan demo dapat melakukan pencarian *file* audio yang terdapat pada *List Added Songs* secara spesifik sehingga memudahkan *user* dalam pencarian suatu *file* audio. Pada fitur ini terdapat sebuah *combobox* dan *textbox* yang memiliki fungsi masing-masing. *Combobox* berfungsi untuk melakukan pencarian berdasarkan opsi-opsi yang tersedia, sebagai contoh *user* dapat melakukan pencarian berdasarkan judul lagu, genre, dan lainnya. *Textbox* pada fitur ini berfungsi untuk melakukan pencarian sesuai dengan isi dari *textbox* yang dimasukkan atau di-*input* oleh *user*.

3. *Sort By*

Pada fitur ini, *user* admin dan demo dapat melakukan pengurutan *file* audio yang terdapat pada *List Added Songs* sehingga mempermudah *user* dalam pencarian suatu *file* audio. Pada fitur ini terdapat sebuah *combobox* yang

dapat dipilih oleh *user* sehingga *file* audio akan mengalami proses pengurutan berdasarkan hasil pilihan *user*.

#### 4. *About*

fitur ini menampilkan informasi-informasi mengenai aplikasi ini. Informasi-informasi tersebut antara lain adalah kegunaan aplikasi ini dan informasi mengenai pembuat aplikasi. Pada fitur ini terdapat sebuah *button* yang akan membawa *user* kepada suatu halaman baru yang berisikan penjelasan mengenai setiap fitur yang ada dan bagaimana cara menggunakan aplikasi ini dengan benar.

#### 5. *Music Player*

Pada halaman utama terdapat berbagai *button* dan *trackbar* dimana dikelompokkan kedalam suatu *groupbox* yang berguna untuk memutar *file* audio yang dipilih oleh *user*. *Button-button* yang terdapat pada *music player* tersebut antara lain *button play/pause*, *button stop*, *button volume*, *button fast forward*, dan *button back forward*. Sedangkan *trackbar* yang terdapat pada *music player* tersebut antara lain *trackbar volume* dan *trackbar duration*.

#### 6. *List Added Songs*

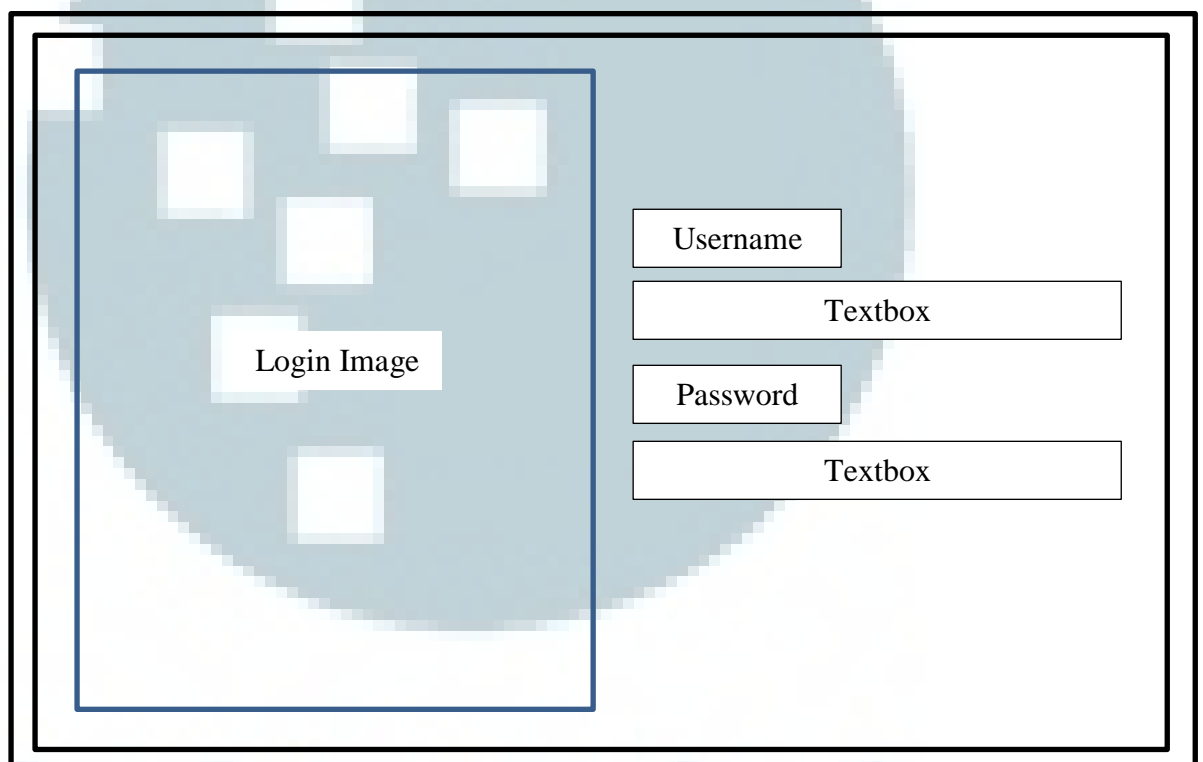
Pada halaman utama terdapat sebuah *listview* yang berisikan seluruh *file* audio yang telah dimasukkan ke dalam *database* yang siap diputar apabila *user* ingin mendengarkannya. *List Added Songs* dibagi menjadi lima buah kolom, yaitu *name* (judul lagu), *time* (lama durasi lagu), *artist* (nama penyanyi), *album* (nama album), dan *genre* (genre lagu).



### 3.2.2 Design Antarmuka Aplikasi

Design antarmuka atau *user interface* aplikasi akan digunakan sebagai pedoman dalam membangun *interface* yang akan dilihat dan digunakan oleh pengguna atau *user* untuk menggunakan aplikasi. Berikut adalah hasil perancangan design antarmuka aplikasi dalam penelitian ini.

#### 1. Halaman Login



The diagram illustrates the layout of a login page. It is enclosed in a black rectangular border. On the left side, there is a blue-bordered box containing a placeholder for a 'Login Image'. To the right of this box, there is a vertical stack of four input fields. The first field is labeled 'Username'. The second field is a simple 'Textbox'. The third field is labeled 'Password'. The fourth field is another 'Textbox'.

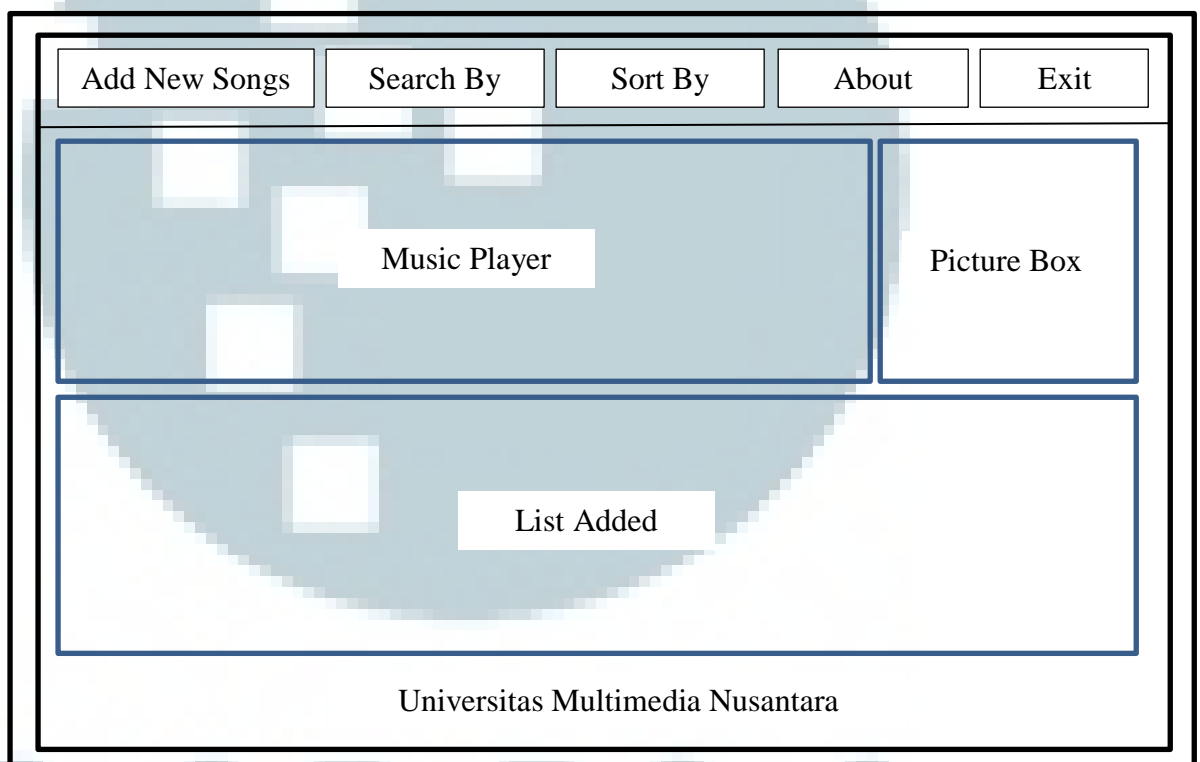
Gambar 3.6 Halaman Login Aplikasi

Halaman login aplikasi merupakan halaman yang pertama kali akan ditampilkan kepada *user* atau pengguna pada saat menjalankan aplikasi ini.

*User* pada aplikasi ini dibagi menjadi dua, yaitu admin dan demo. Perbedaan antara kedua *user* tersebut adalah akses level yang diberikan. Pada *user*

admin, *user* dapat melakukan penambahan *file* audio baru dengan menggunakan fitur *Add New Songs*, sedangkan pada *user* demo tidak dapat melakukan penambahan *file* audio, melainkan hanya dapat melakukan pencarian *file* audio dan pemutaran *file* audio dimana kedua hal tersebut juga dapat dilakukan oleh *user* admin.

## 2. Halaman Utama *User* Admin

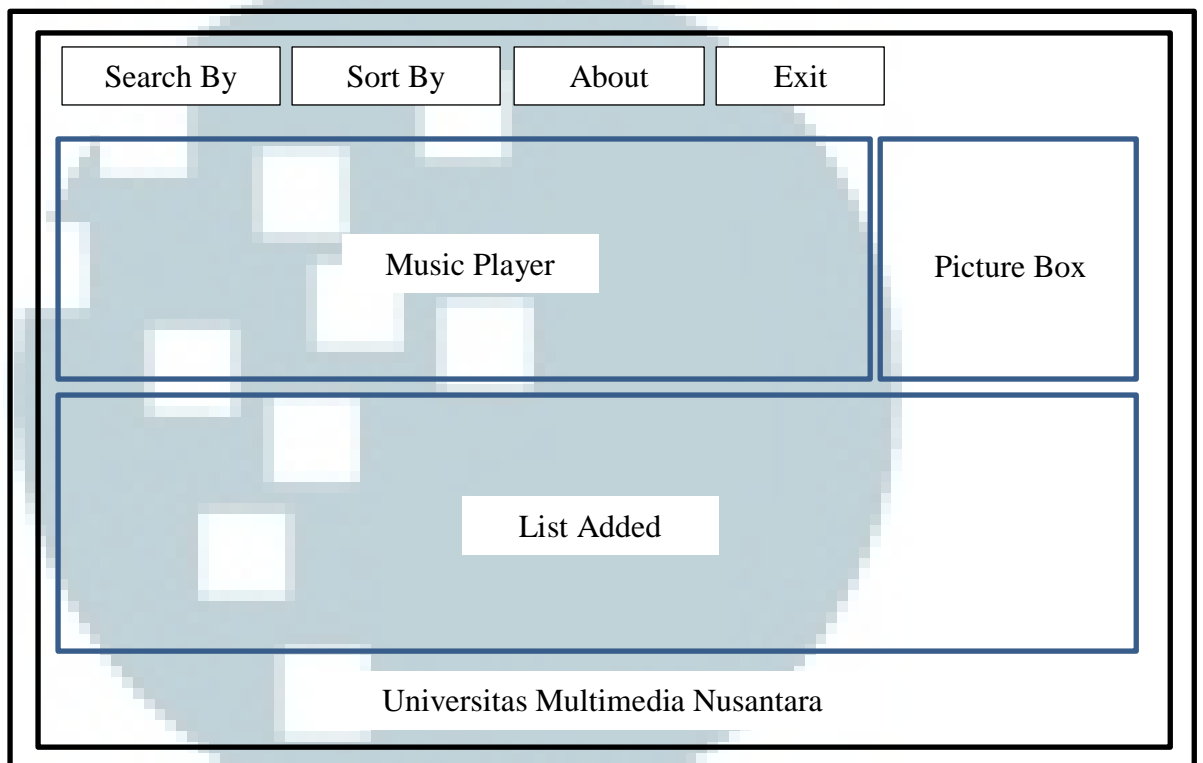


Gambar 3.7 Halaman Utama *User* Admin Aplikasi

Halaman ini akan tampil apabila *user* melakukan login sebagai admin. *User* admin merupakan *user* yang dapat menggunakan seluruh fitur yang terdapat pada aplikasi ini. Pada terdapat enam buah fitur yang dapat digunakan oleh *user* admin, yaitu *Add New Songs*, *Search By*, *Sort By*, *About*, *Music Player*,

*List Added Songs*. Pada bagian bawah halaman utama selalu terdapat tulisan Universitas Multimedia Nusantara.

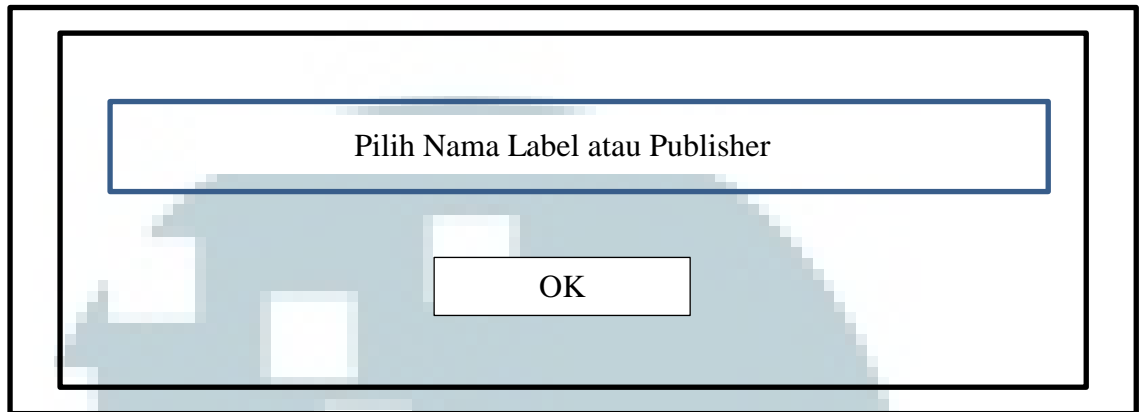
### 3. Halaman Utama *User Demo*



Gambar 3.8 Halaman Utama *User Demo* Aplikasi

Halaman ini akan tampil apabila *user* melakukan login sebagai demo. Berbeda dengan *user* admin, *user* demo hanya tidak dapat menggunakan satu buah fitur, yaitu *Add New Songs* yang berguna untuk menambah *file* audio ke dalam *database*.

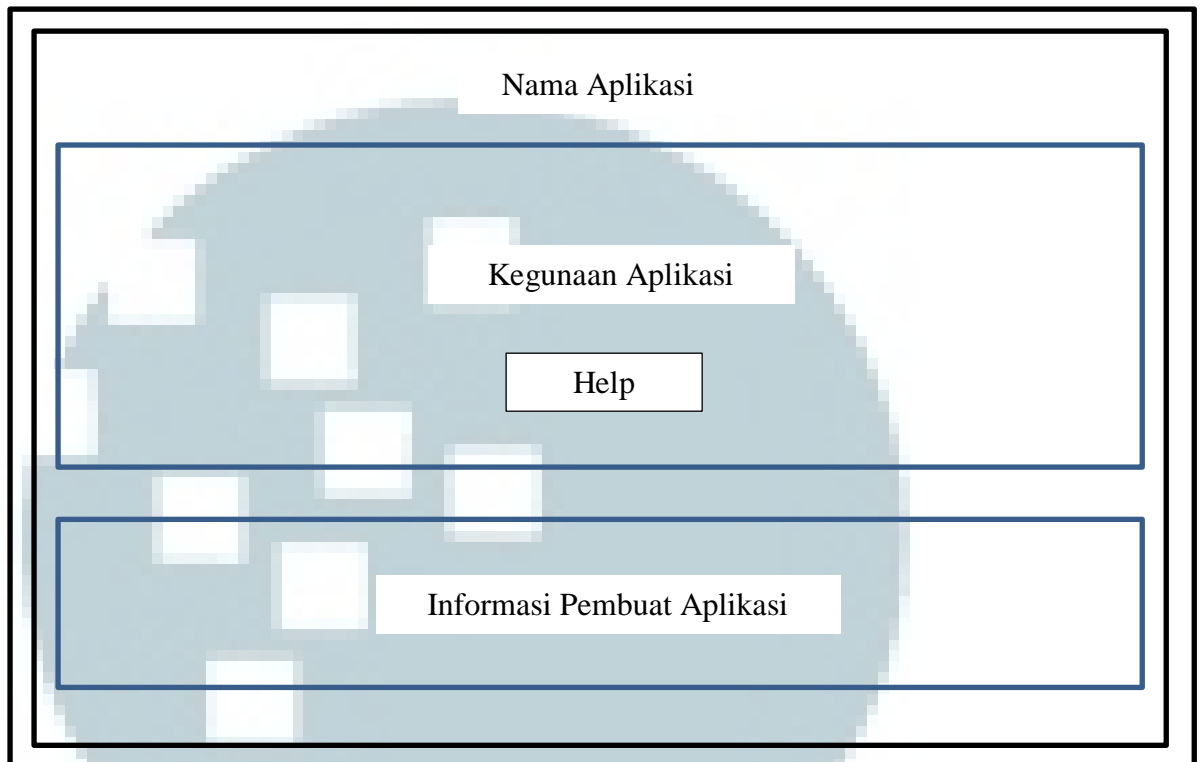
#### 4. Halaman *Add Publisher*



Gambar 3.9 Halaman *Add Publisher* Aplikasi

Halaman ini akan tampil apabila *user* atau pengguna melakukan proses penambahan *file* audio pada *database* dengan menggunakan fitur *Add New Songs*. Dengan menggunakan fitur *Add New Songs*, *user* atau pengguna akan memilih *file* audio yang akan di *upload* ke dalam *database*. Setelah memilih *file* audio tersebut, halaman ini akan muncul untuk menentukan nama label atau *publisher* dari *file* audio yang akan dimasukkan ke dalam *database*.

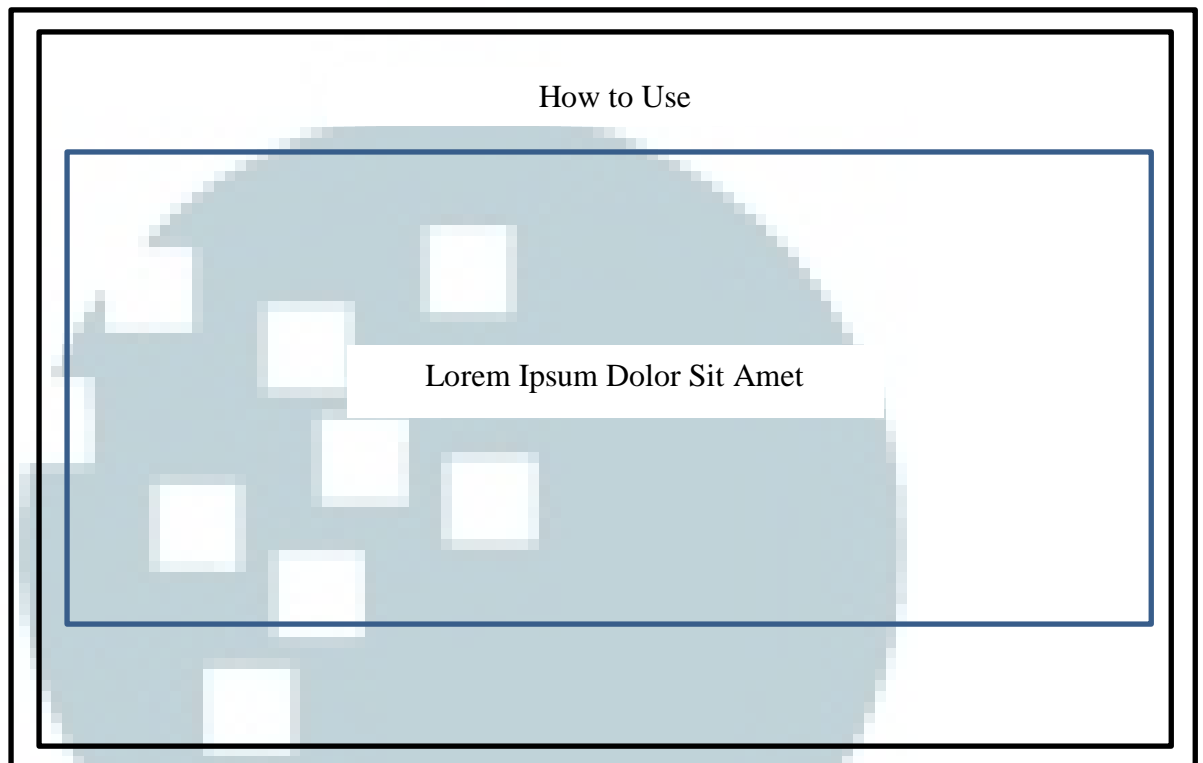
## 5. Halaman *About*



Gambar 3.10 Halaman *About* Aplikasi

Halaman ini akan tampil apabila *user* atau pengguna memilih fitur *About* pada halaman utama. Halaman *About* merupakan halaman yang berisikan kegunaan dari aplikasi, fitur-fitur yang tersedia pada halaman utama, dan informasi-informasi singkat mengenai pembuat aplikasi. Pada halaman ini pula terdapat sebuah *button* yang berguna untuk menampilkan halaman *Help* kepada *user* atau pengguna.

## 6. Halaman *Help*



Gambar 3.11 Halaman *Help* Aplikasi

Halaman ini akan tampil apabila *user* atau pengguna memilih fitur *Help* pada halaman *About*. Halaman *Help* merupakan halaman yang berisikan penjelasan-penjelasan mengenai fitur-fitur yang terdapat pada halaman utama. Selain berisikan penjelasan-penjelasan, pada halaman *Help* ini *user* diajarkan bagaimana cara menggunakan aplikasi ini.

### 3.2.3 Struktur Tabel

Berbeda dengan *listening post* pada umumnya, pada aplikasi ini tidak menggunakan perantara CD untuk melakukan penyimpanan *file* audio. Aplikasi ini menggunakan sebuah *database*, sehingga *file* audio akan dimasukkan kedalam *database* dimana seluruh *file* audio yang masuk ke dalam *database* akan ditampilkan sehingga *user* atau pengguna dapat memilih *file* audio yang akan diputar. Aplikasi *dTunez Studio* ini menggunakan *database* yang terdiri dari lima buah tabel. Dimana setiap tabel memiliki hubungan satu dengan yang lainnya melalui *primary key* dan *foreign key* yang ada pada tabel-tabel tersebut. Berikut adalah struktur tabel yang digunakan pada aplikasi *dTunez Studio*.

#### 1. Tabel *Songs*

Tabel ini berfungsi untuk menyimpan informasi-informasi umum mengenai suatu *file* audio. Informasi-informasi umum tersebut berupa judul dari *file* audio, genre musik pada *file* audio, tahun rilis, dan lain sebagainya.

Tabel 3.1 Tabel *Songs*

| Field Name | Type    | Length | Constraint           | Information       |
|------------|---------|--------|----------------------|-------------------|
| KodeLagu   | Varchar | 5      | Primary Key          | Kode lagu         |
| JudulLagu  | Varchar | 50     |                      | Nama file audio   |
| KodeLabel  | Varchar | 5      | Foreign Key (Label)  | Kode label        |
| KodeArtist | Varchar | 5      | Foreign Key (Artist) | Kode artist       |
| Genre      | Varchar | 50     |                      | Jenis genre       |
| Duration   | Varchar | 5      |                      | Durasi file audio |
| TahunRilis | Int     |        |                      | Tahun rilis       |
| PathFile   | Text    |        |                      | Letak file audio  |

## 2. Tabel Album

Tabel ini berfungsi untuk menyimpan informasi-informasi umum mengenai album dari suatu *file* audio. Informasi-informasi mengenai album dari suatu *file* audio dapat berupa nama album, jenis genre album, dan lain sebagainya.

Tabel 3.2 Tabel Album

| Field Name | Type    | Length | Constraint  | Information       |
|------------|---------|--------|-------------|-------------------|
| KodeAlbum  | Varchar | 5      | Primary Key | Kode album        |
| NamaAlbum  | Varchar | 50     |             | Nama album        |
| TahunRilis | Int     |        |             | Tahun rilis       |
| GenreAlbum | Varchar | 50     |             | Jenis genre album |

## 3. Tabel *Detail* Album

Tabel ini berfungsi sebagai penghubung antara suatu *file* audio dengan suatu album. Satu album terdiri dari beberapa *file* audio. Dengan adanya table ini, mempermudah untuk mengetahui *file* audio mana saja yang tersusun kedalam sebuah album.

Tabel 3.3 Tabel *Detail* Album

| Field Name | Type    | Length | Constraint          | Information     |
|------------|---------|--------|---------------------|-----------------|
| KodeLagu   | Varchar | 5      | Foreign Key (Songs) | Kode lagu       |
| KodeAlbum  | Varchar | 5      | Foreign Key (Album) | Nama file audio |



#### 4. Tabel *Artist*

Tabel ini berfungsi untuk menyimpan informasi mengenai nama *artist* atau penyanyi pada suatu *file* audio. Pada suatu *file* audio tentunya terdapat penyanyi yang menyanyikan sebuah lagu, nama penyanyi tersebut merupakan informasi yang disimpan pada tabel ini.

Tabel 3.4 Tabel *Artist*

| Field Name | Type    | Length | Constraint  | Information |
|------------|---------|--------|-------------|-------------|
| KodeArtist | Varchar | 5      | Primary Key | Kode artist |
| NamaArtist | Varchar | 50     |             | Nama artist |

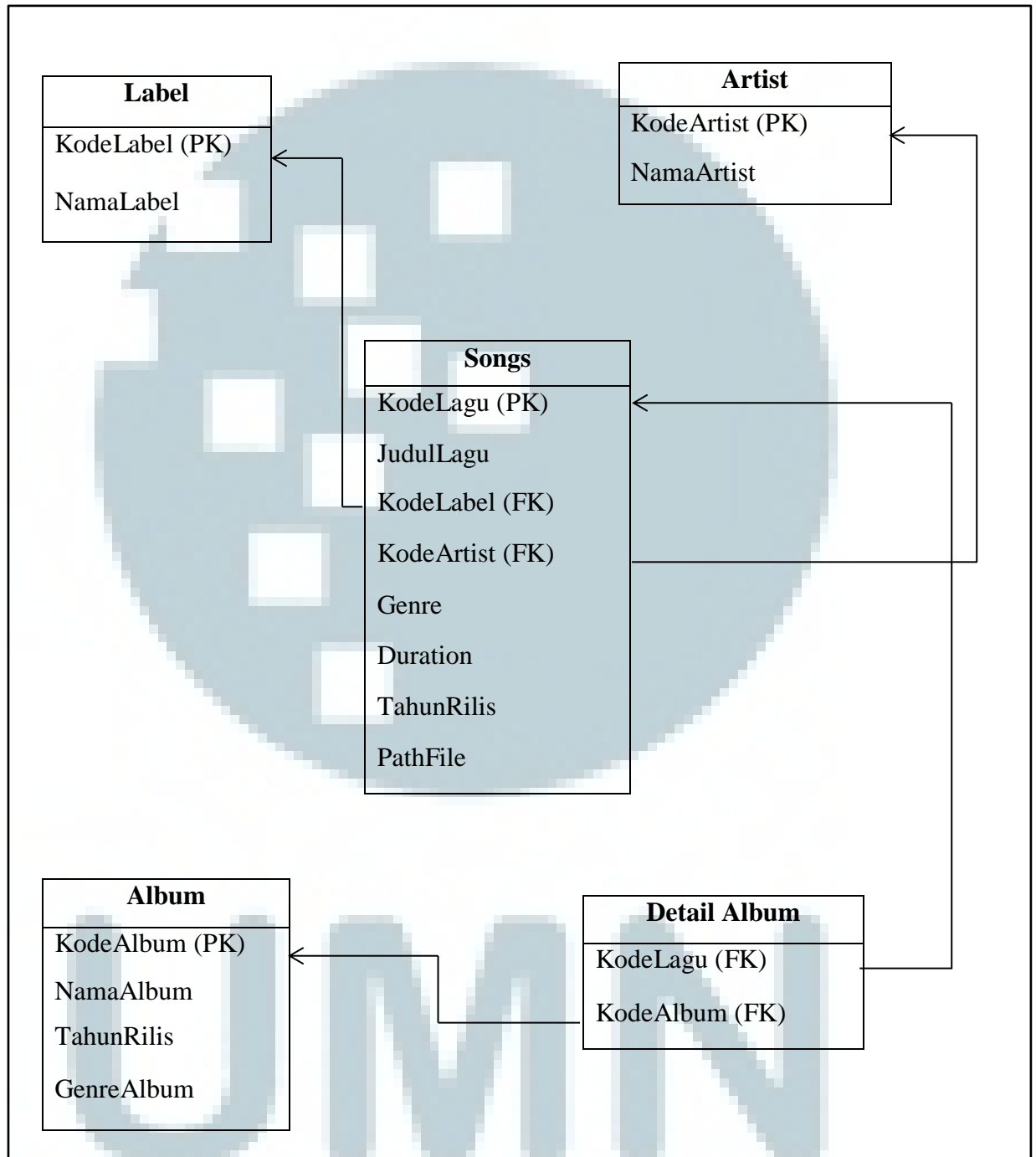
#### 5. Tabel Label

Tabel ini berfungsi untuk menyimpan informasi mengenai nama *publisher* atau yang biasa disebut label pada suatu *file* audio. Label merupakan sponsor bagi penyanyi dalam proses pembuatan album. Nama label dari *file* audio ini merupakan informasi yang disimpan pada tabel ini.

Tabel 3.5 Tabel Label

| Field Name | Type    | Length | Constraint  | Information |
|------------|---------|--------|-------------|-------------|
| KodeArtist | Varchar | 5      | Primary Key | Kode artist |
| NamaArtist | Varchar | 50     |             | Nama artist |

### 3.2.4 Hubungan Antar Tabel



Gambar 3.12 Hubungan Antar Tabel

Pada gambar diatas, dapat dilihat terdapat lima buah tabel yang saling berhubungan antara satu dengan yang lainnya. Hubungan antara tabel satu dengan tabel lain dihubungkan oleh *primary key* dan *foreign key*. Pada tabel Label, kolom KodeLabel menjadi *primary key* dari tabel tersebut, dimana pada tabel Songs terdapat kolom KodeLabel yang merupakan *foreign key* yang menunjuk pada kolom KodeLabel yang terdapat pada tabel Label. Selain itu, pada tabel Artist terdapat kolom KodeArtist yang merupakan *primary key* pada tabel tersebut. Pada tabel Songs juga terdapat kolom KodeArtist yang merupakan *foreign key* yang menunjuk pada kolom KodeArtist pada tabel Artist. Data-data mengenai album dari suatu *file* audio disimpan pada tabel Album, dimana terdapat kolom KodeAlbum yang menjadi *primary key* pada tabel tersebut. Pada tabel Detail Album terdiri dari dua buah kolom dimana kedua buah kolom tersebut merupakan *foreign key* yang menunjukkan ke suatu tabel yang berbeda. Dua buah kolom tersebut adalah kolom KodeLagu yang menunjuk pada kolom KodeLagu yang terdapat pada tabel Songs, dan kolom KodeAlbum yang menunjuk pada kolom KodeAlbum yang terdapat pada tabel Album.

U  
M  
N