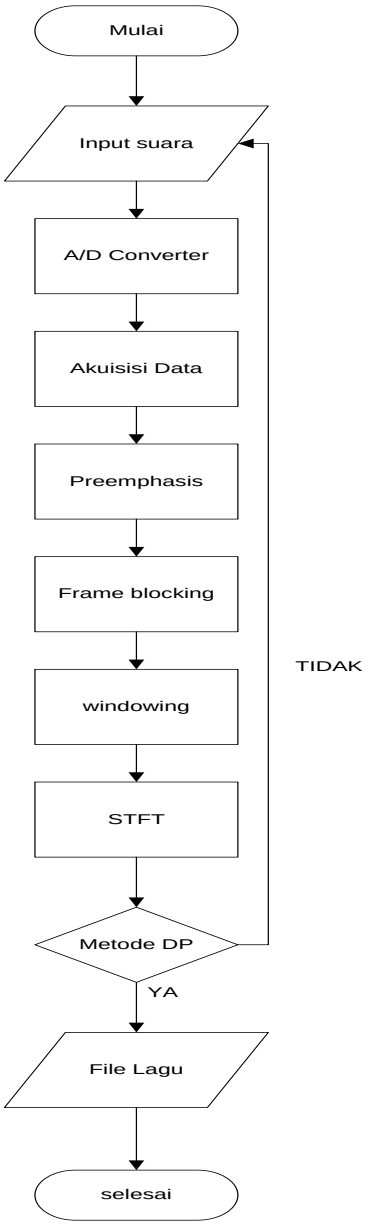
BAB 3

ANALISIS DAN PERANCANGAN

3.1 Analisis Masalah

Analisis masalah merupakan penguraian dari suatu masalah yang utuh kedalam bagian – bagian komponennya dengan tujuan untuk mengidentifikasi dan mengavaluasi permasalahan, hambatan, dan kebutuhan dengan harapan mendapatkan sistem yang sesuai dengan kebutuhan. Tahap analisis sistem ini merupakan tahapan kritis dan penting agar meminimalisir kesalahan – kesalahan yang dapat menyebabkan kesalahan pada tahap yang selanjutnya. Hasil akhir dari analisis sistem merupakan deskripsi dan pemeteaan sistem yang akan dibangun yang digambarkan dengan diagram alur.

Dalam proses pembuatan suatu sistem dilakukan penganalisaan tentang sistem yang akan dibangun, berikut ini alur pencarian lirik lagu melalui media suara.

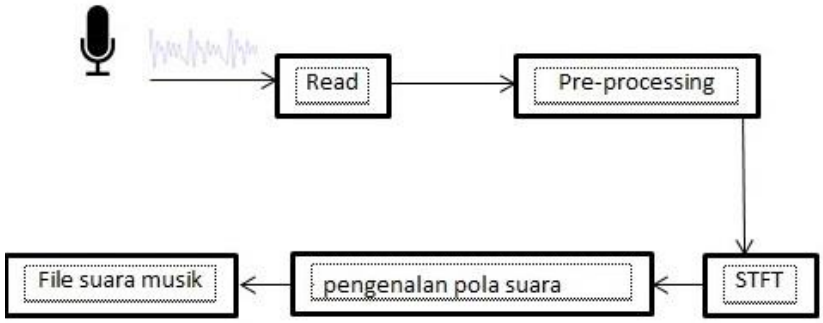


Gambar 3.1 Alur diagram pengenalan suara ucapan pada pencarian judul lagu

3.1.1 Deskripsi umum sistem

Proses pengenalan suara pengucapan sepenggal lirik lagu dan judul lagu dapat dijelaskan sebagai berikut, pertama-tama pengguna merekam suara pada *recognition* pada *smartphone* sebanyak beberapa kali sebagai sampel, setelah itu

barulah sistem akan mengenali suara dari pengguna karena sistem memiliki sampel lagu atau judul lagu yang akan dijadikan perbandingan. Sistem kemudian akan mengeluarkan keluaran file yang menjadi perbandingan kesamaan dengan suara yang di ucapkan di awal.



Gambar 3.2 Proses pengidentifikasi pengenalan suara ucapan

Gambar 3.2 menunjukan garis besar proses pengidentifikasi pengenalan suara ucapan yang dirancang.

1. Tahap read adalah tahap *analog to digital converter* dimana merubah sinyal yang diinputkan berupa sinyal analog menjadi sinyal digital.

2. Tahap p*re-procesing* adalah proses pengkodisisn sinyal suara menggunakan *filter.*

3. Tahap STFT (*Short time fourier term)* adalah dimana sinyal suara dari domain waktu menjadi spektrum energi dalam domain frekuensi. Suara di tangkap *microphone* dan ditransformasi menjadi citra dua dimensi (*spektogram)*. *Spektogram* adalah penggambaran dua dimensi dari sinyal suara dengan sumbu *horisontal* menunjukan waktu, sumbu *vertikal* menunjukan frekuensi, dan kerapatan titik menggambarkan amplitudo atau energi akustik.

4. Pengenalan pola suara adalah tahap dimana pengenalan sinyal suara dengan file lagu sebagai referensi. untuk membandingkan suara input pembicara dengan sampel suara yang ada di file musik dan menghasilkan keluaran data file musik.

5. File Suara musik adalah tahap dimana hasil dari suara masukkan menghasilkan file lagu yang dicari.

3.2 Tahap Read

Analog to digital converter dimana sinyal suara yang akan di proses bersifat analog sehingga jika akan dilakukan pengolahan secara digital, sinyal suara tersebut harus dikonversi menjadi sinyal digital, berupa urutan angka dengan tingkat presisi tertentu yang di namakan analog to digital *conversion* dengan menggunakan *analog-to-converter*. Konsep kerja *analog-to-converter* terdiri dari tiga proses :



Gambar 3.3 konsep kerja *analog-to-converter*

Keterangan konsep kerja *analog-to-converter :*

a. Sampling adalah konversi sinyal kontinu dalam domain waktu menjadi sinyal diskrit, melalui proses sampling sinyal pada selang waktu tertentu. Sehingga jika $X_0(t)$ adalah sinyal *input,* maka *output*-nya adalah $X_0(nT)$, dengan T adalah interval sampling.

b. *Kuantisasi* adalah konversi dari *descrete time,continous value,* menjadi *decrete time* dan *continous value*.

c. *Coding*, pada proses ini, tiap nilai diskrit yang telah di dapat, direpresentasikan dengan angkat binary n-bit.

3.3 Tahap *pre procesing*

3.3.1 Akuisisi Data

Data berupa sinyal suara diperoleh dengan caramerekam suara melalui *microphone* yang dihubungkan. Perekaman suara dilakukan dengan pengambilan sample suara dengan frekuensi sampling diambil sebesar 8000 Hz, 8 bit karena kecepatan pencuplikan tersebut dilakukan dengan didasarkan asumsi bahwa sinyal percakapan (*speech*) berada pada daerah frekuensi 300-3400 Hz , dengan durasi maximal 5-10 detik.

Suara diucapkan oleh pengguna dimana setiap suara bisa hanya satu kali pengucapan , dapat pula lebih dari satu kali apabila file yang dicari belum ditemukan atau kata yang diucapkan belum tertangkap oleh sistem. Suara yang akan dijadikan sampel adalah nama, contoh CANTIK.

Proses demi proses yang dijalankan guna mencari frekuensi utama dari sebuah sinyal yang masuk dengan melihat angka frekuensiyang nilai magnitudonya tertinggi.

3.3.2 Preemphasis

Setelah dilakukan akuisisi data dengan nama CANTIK dilakukan proses *preemphasis.* Preemphasis adalah suatu proses produksi suara manusia, radiasi pada bibir dan lidah ketika proses mengakibatkan komponen frekuensi tingginya. Tujuan *preemphasis* untuk memperbaiki sinyal dari gangguan *noise*. Sehingga dapat meningkatkan tingkat akurasi pengenalan suara untuk tahap selanjutnya, *default* dari nilai *alpha* yang di gunakan dalam proses *preemphasis* adalah 0,97. Rumus yang digunakan untuk proses preemphasis :

Dik : perhitungan digunakan jumlah data sebanyak 3 titik sampel, data sinyal adalah (100,200,300). Dengan mengguanakan Rumus 2.1

$$y(n) = s(n) – \alpha s\ (n-1)$$

nilai α : 0,97

Contoh perhitungan :

Signal : [100,200,300]

y(1) = 100

y(2) = 200- (0,97*100) = 103

y(3) = 300- (0,97*200) = 106

hasil sinyal setelah *preemphasis* adalah [100,103,106] karena ini sinyal frekuensi maka jumlah gelombang sinyal analog dalam satuan detik (t)

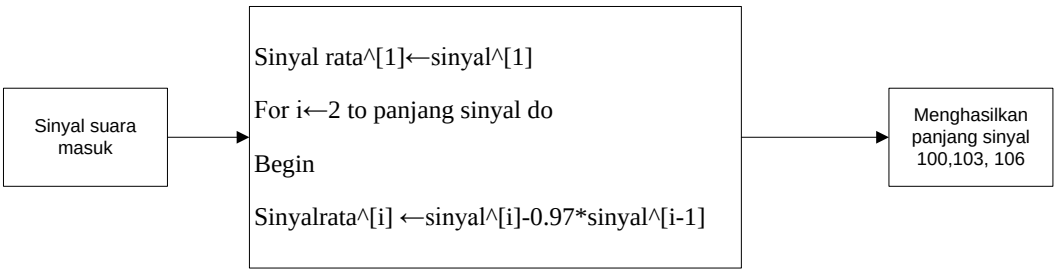
hasil dari proses pre-emphasis ini kemudian dipotong-potong menjadi beberapa potongan kecil, setiap potongan tersebut disebut *frame*. Nilai-nilai ini lah akan di pakai untuk menghitung Frame Blocking dan menjadi nilai y.

algoritma *procedure preemphasis*, keteragan :

Psinyal disini adalah panjang sinyal yang akan menjadi acuan untuk *procedure preemphasis.*

```
Procedure preemphasis (sinyal:Psinyal;var sinyalrata:Psinyalrata);
Kamus
        I: Integer
Algoritma
        Sinyal rata^[1]←sinyal^[1]
                For i←2 to panjang sinyal do
        Begin
                Sinyalrata^[i] ←sinyal^[i]-0.97*sinyal^[i-1]
                endfor
        end
```

Alur masuk dengan suara ucapan dimulai dari sinyal suara masuk , proses dan hasil dari masukan sinyal suara pada proses *preemphasis.*



Hasil masukan sinyal suara pada proses *preemphasis* dengan nama CANTIK sebai berikut:



Gambar 3.4 Sinyal hasil *preemphasis*

3.3.3 Frame Blocking

Frame blocking adalah suatu proses pada tahapan sinyal yang telah di *preempahasis,* sinyal suara yang telah di *preemphasis* dibagi menjadi beberapa frame yang nantinya dapat memudahkan dalam perhitungan dan analisa suara, setiap potongan-potongan disebut *frame.* Satu *frame* terdiri dari beberapa sampel tergantung tiap berapa detik suara akan disampel dan berapa besar frekuensi samplingnya.

Dengan menggunakan rumus 2.2 berikut perhitungan *frame blocking* :

Dik :

Dengan waktu tiap 20ms, dan sampel rate 11025 Hz

Maka didapatkan frame size (N)

N = 11025 * 0.02 = 220.5 sampel

Overlapping frame (M) = 110.25

x(n) = y(M+n)

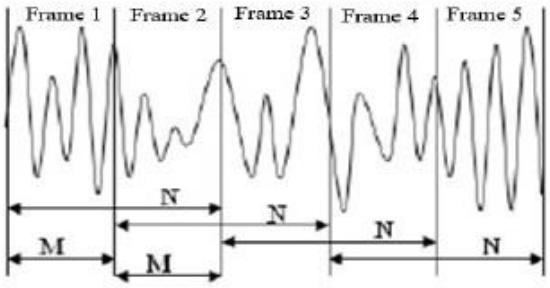
x(1) =100(110.25+1) = 11125 Hz

x(2) =103(110.25+2) = 11561 Hz

x(3) =106(110.25+3) = 12004 Hz

nilai dari beberapa buah potongan *frame* ini yang akan dipakai untuk mencari Wo.

potongan-potongan *frame* digambarkan dengan gambar dibawah ini:



Gambar 3.5 *Frame blocking*

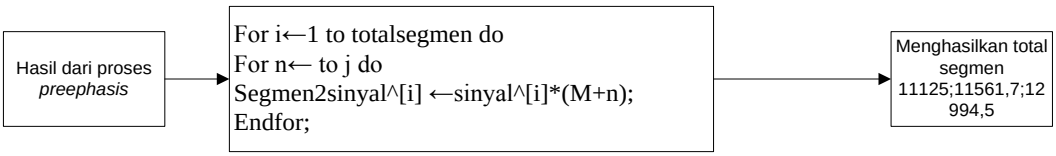
Keterangan

*Frame*1,*frame*2, *frame*3, *frame*4, *frame*5 adalah sinyal suara yang di *frame blocking*, dimana M panjang tiap *frame* dan N *overlapping* tiap *frame*-nya.

Keterangan :

Sinyal disini adalah panjang sinyal dimana panjang sinyal ini berfungsi untuk mengetahui maksimum dan nilai-nilai puncak minimum pada selang pengamatan suara (*1 frame*)

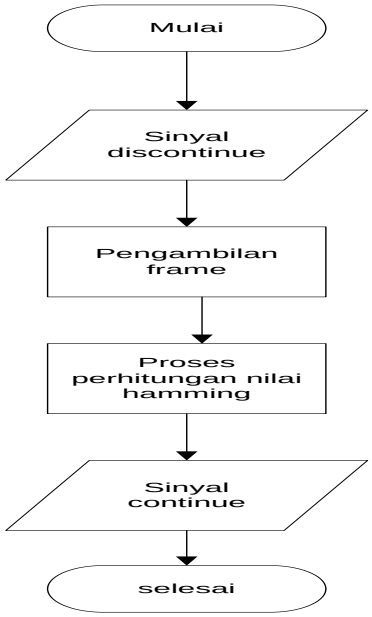
Alur masukan suara setelah dilakukan proses *preemphasis* dengan kata ucapan dimulai masukan hasil dari *preemphasis*, proses dan hasil panjang sinyal yang dihasilkan pada proses *frame blocking.*



3.3.4 Windowing

Hasil dari proses *frame blocking* menghasilkan efek sinyal *discontinue*, agar tidak terjadi kesalahan data pada proses *fourier transform* maka sampel suara yang telah di bagi menjadi beberapa frame perlu dijadikan suara *continue* dengan menggunakan proses *windowing*. Hal ini dilakukan untuk meminimalkan pada bagian awal dan akhir sinyal. Jika didefinisikan sebuah *window w(n)* dan sinyal tiap bagian adalah i maka sinyal hasil proses *windowing.*

Proses window hamming digambarkan sebagai berikut :



Gambar 3.6 Alur proses *window hamming*

Jenis window yang digunakan adalah *Hamming Window* dengan rumus 2.3 diperoleh hasil sebagai berikut :

$$w[n] = 0.54-0.46*\cos(2*\pi*n/(N-1))$$

contoh perhitungan *Hamming Window* :

Dik : π= 3.14

n= 1

N=220,5

Penyelesaian window w(n) :

w(n)= 0.54-0.46 cos (2 πn/(N-1))

w(1)= 0.54-0.46 cos(2.3.14.1/(220,5-1))

w(1)= 0.54-0.46 cos 0

w(1)= 0.54-0.46*1

w(1)= 0.08 dst

untuk data hasil frame blocking yaitu (11125 , 11458, 11792) sehingga,

Wo= 0.08 * (11125) = 890

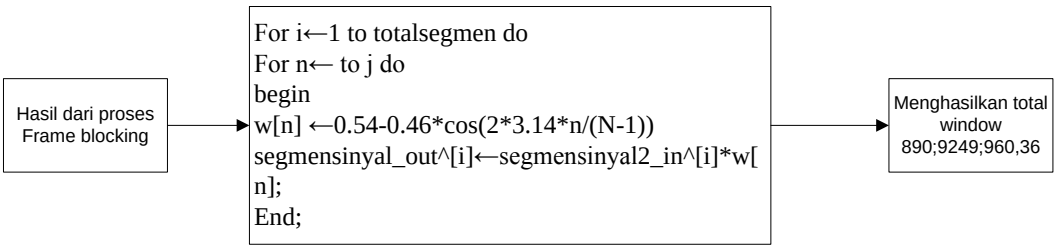
Wo= 0.08 * (11458) = 916.6

Wo= 0.08 * (11792) = 943.36

Hasil dari *Hamming Window* adalah [890;916,6;943,36]

Nilai *Hamming Window* akan dipakai untuk menghitung W pada STFT yang akan menghasilkan sinyal tetap atau hasil akhir dari perhitungan.

Alur masukan suara setelah dilakukan proses *frame blocking* kata CANTIK dimulai masukan hasil *frame blocking*, proses , dan hasil total frame yang dihasilkan pada proses *windowing.*



3.3.5 Short Time Fourier Transform (STFT)

Proses windowing menghasilkan spektrum suara domain waktu, agar tidak terjadi kesalahan maka spektrum domain waktu dirubah menjadi sinyal frekuensi dengan menggunakan proses *Short Time Fourier Transforme*. STFT merupakan salah satu metode untuk transformasi sinyal suara menjadi sinyal frekuensi. Artinya proses perekaman suara disimpan dalam bentuk *digital* berupa gelombang *spektrum* suara berbasis frekuensi. Hasil dari proses *fourier transform* menghasilkan pendeteksian gelombang frekuensi domain dalam bentuk diskrit.

Tujuan dari proses STFT adalah untuk mendapatkan informasi spektrum frekuensi pada seluruh nilai amplitudo yang telah melalui proses windowing.

Diketahui sinyal hasil windowing:

(890 ; 916,6 ; 943,36)

Untuk F_0 maka diperoleh perhitungan sebagai berikut , dengan rumus 2.4 didapatkan hasil sebagai berikut :

$$X[k] = \sum_{n=1}^{N=1} x(n)W$$

$$X[k] = \frac{1}{N}\sum_{n=1}^{N=1} x(n)\cos\left(2 * \pi * k * \frac{n}{N}\right) - jSin(2 * \pi * k * \frac{n}{N})$$

$$X[1] = \frac{1}{3}[890(\cos(\frac{2\pi*1*1}{3}))]\, jsin\, \frac{2\pi*1*1}{3} +$$

$$[916{,}6(\cos(\frac{2\pi*1*2}{3}))] - jsin\, \frac{2\pi*1*2}{3} +$$

[943,36(cos$\left(\frac{2\pi*1*3}{3}\right)$)] - $jsin\frac{2\pi*1*2}{3}$ = 922 + 0,073j = 922

masukan kedalam rumus :

$[R2 + J2]^{½}$= $\sqrt{988^2 + 0{,}299^2}$ = 1093

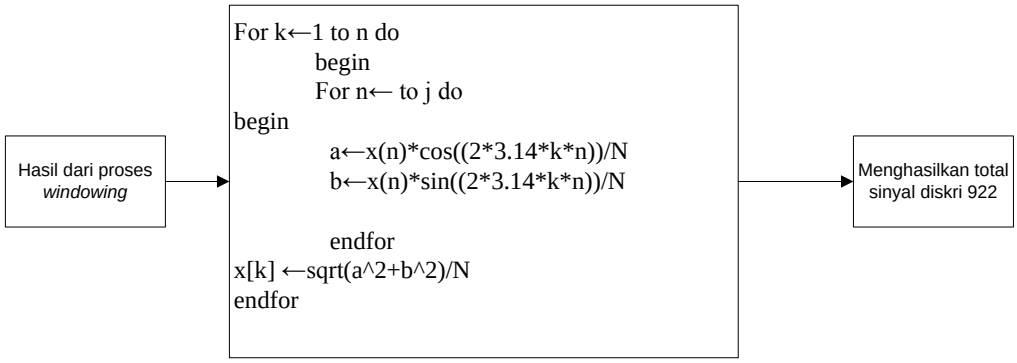
Hamming Window Didapat nilai F_0 = 1093 dan nilai ini untuk hasil frame dari frekuensi sebuah sinyal suara.

tahapan disni adalah mengubah tiap frame dari domain waktu ke domain frekuensi

[960,36(cos$\left(\frac{2\pi*1*3}{3}\right)$)] - $jsin\frac{2\pi*1*3}{3}$ = $1093 - 0{,}0299j$

Gunakan rumus $\left|[R^2 + I^2]^{1/2}\right| = \left|\sqrt{922^2} + 0{,}073^2\right| = |922| = 1093$

Alur masukan suara setelah dilakukan proses windowing dengan kata CANTIK dimulai masukan hasil *windowing*, proses, dan hasil total window yang dihasilkan pada proses STFT.



3.3.6 Verifikasi pengenalan pola ucapan

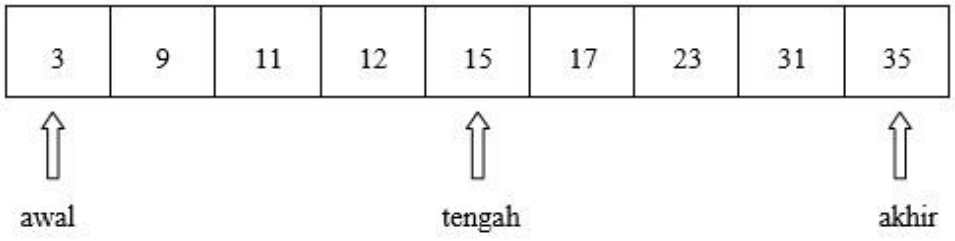
Dalam sistem pengenalan kata menggunakan metode word detection, ada beberapa tahap, yaitu membagi lebih dari satu proses ekstrasi dan pencocokan kata, dan pencocokan dilakukan algoritma pencarian binary.

Prinsib dari pencarian dapat dijelaskan sebagai berikut :

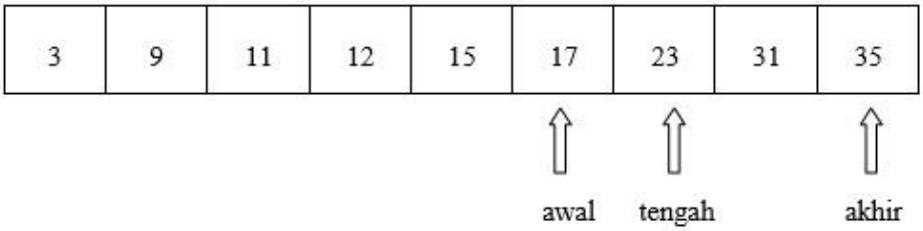
1. Mula-mula diambil posisi awal= 1 dan posisi akhir = N
2. Cari posisi data tengan dengan rumus (posisi awal + posisi akhir)/2
3. Data yang dicari dibandingkan dengan data tengan
4. Jika lebih kecil, proses dilakukan kembali tetapi posisi akhir dengan posisi tengan – 1
5. Jika lebih besar, proses dilakukan kembali tetapi posisi awal dianggap sama dengan posisi tengan + 1
6. Demikian seterusnya sampai data tengan sama dengan yang dicari.

   Contoh :

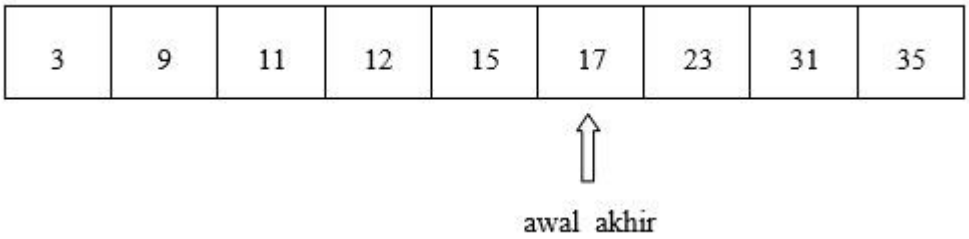
   Misal mencari data 17 pada sekumpulan data berikut :



1. Mula-mula dicari data tengan, dengan rumus (1+9)/2= 5
2. Berarti data tengan adalah data ke 5, yaitu 15.
3. Data yang dicari, yaitu 17, dibandingkan dengan data tengah ini
4. Karena 17 ? 15, berarti proses dilanjutkan tetapi kali ini posisi dianggap sama dengan + 1 atau 6



1. Data tengah yang baru didapat dengan rumus (6+9)/2 = 7. Berarti data tengah yang baru adalah data ke-7, yaitu 23.
2. Data yang di cari yaitu 17 dibandingkan dengan data tengan ini.
3. Karena 17 < 23, berarti proses dilanjutkan tetapi kali ini posisi dianggap sama dengan posisi tengan -1 atau 6.

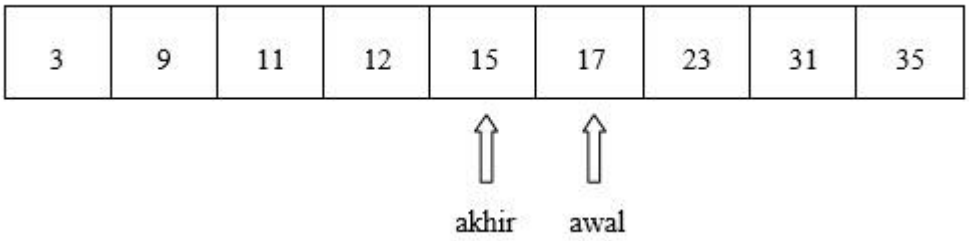


1. Data tengan yang baru didapat dengan rumus (6+6)/2 = 6 berarti data tengah yang baru adalah data ke-6, yaitu 17
2. Data yang dicari dibandingkan dengan data tengah ini dan ternyata sama jadi data ditemukan pada indeks ke-6.
3. Bagaimana jika data yang dicari tidak ada, misalnya 16 ?

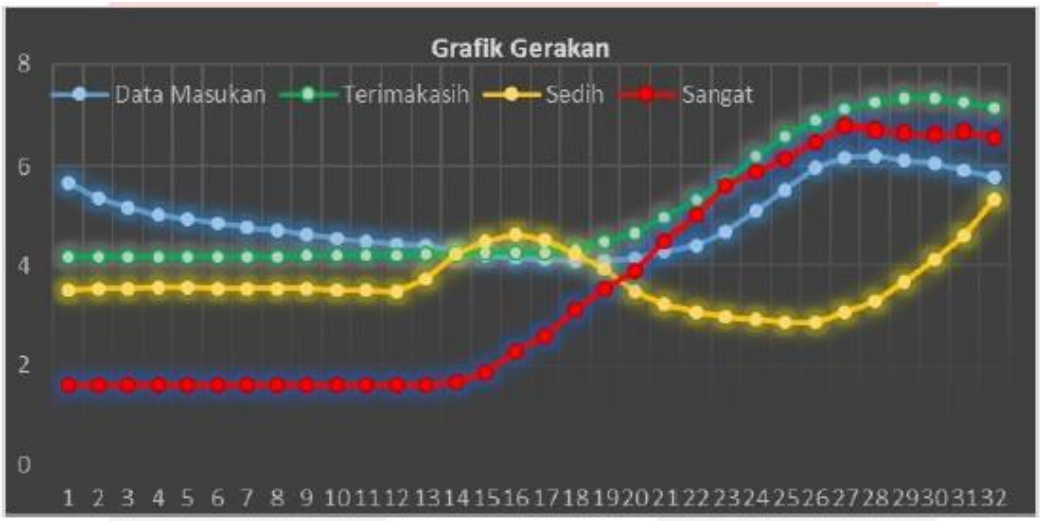
4. Pencarian biner ini akan berakhir jika data ditemukan atau posisi awal lebih besar dari posisi akhir
5. Jika posisi awal sudah lebih besar daripada posisi akhir berarti data tidak ditemukan

Untuk lebih jelas proses dibawah ini proses pencarian 16. Prosesnya sama seperti proses pencarian 17. Tetapi setelah posisi awal = posisi akhir = 6, proses masih dilanjutkan lagi dengan posisi awal = 6 dan posisi akhir = 5 .

Disini dapat dilihat bahwa posisi awal lebih besar dari pada posisi akhir yang artinya data tidak ditemukan.



Untuk lebih jelas melihat posisi pola yang di cari dalam proses sinyal dapat dilihat dalam gambar 3.7



Gambar 3.7 Deret Frame

Kurva biru menunjuka data masukan, sedangkan yang lain menunjukan data referensi yang ada. Disinilah bisa dilihat kemiripan antara pola data masukkan dengan data referensi satu persatu.

Kesimpulan untuk mengambil hasil adalah suatu kata masukan dinyatakan mirip dengan suatu data referensi apabila hasilnya jarak menunjukan jarak paling minimum dibandingkan data referensi lainnya dan juga masih dalam batas toleransi.

Apabila di konferensi ke dalam grafik huruf dapat dengan 3 cara berikut :

1. *Local Distances*

   Local distances adalah suatu ukuran yang dipakai untuk menentukan kemiripan antara dua buah data dan memberikan informasi tentang beberapa suara yang telah diketahui antara lain panjang *frame* dan nilai spektrum dari tiap frame.

   Proses *local distance* adalah mengurutkan sampel-sampel suara pada titik-titik berjarak sama pada waktunya, berikut algoritma *procedure local distance:*

   Dengan kata lain, pertama-tama *local distance* menghitung biaya minimum untuk mencapai setiap jalur, dimulai dari (0,0), dan jalur optimalnya dari kata ‘cantik’.

   Alur masukan suara setelah dilakukan proses *short time fourier transform* dengan kata cantik dimulai masukan hasil dari *short time fourier transform*, proses, dan hasil total sinyal diskrit yang dihasilkan pada proses *local distance.*

2. *Accumulated Distances*

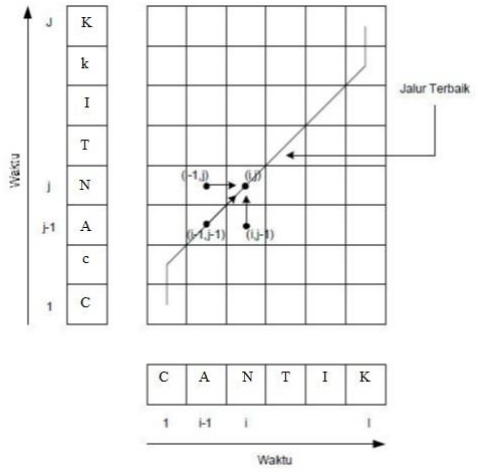
   Setelah di ketahui jalur optimal nama suara, panjang frame dan nilai spektrum dari tiap frame selanjutnya proses *Accumulated distances* adalah jarak perhitungan antara jarak lokal satu dengan jarak lokal sampingya. *Accumulated distances* dapat dicari dengan mencari jarak minimal antara jarak lokal ditambah jarak akumulai sampingnya.

3. *The Distances*

The distances adalah jarak akhir yang menunjukan *warping path* jarak antara dua suara yang diinputkan dengan jarak suara yang ada di file suara yang di simpan. Hasil dari proses *the distances* adalah deskripsi suara ucapan pembicara.

Alur masuk suara setelah dilakukan proses *accumulated distances* dengan ucapan dimulai masukan hasil dari *accumulated distances*, proses, dan hasil lokal sinyal yang dihasilkan pada proses *The Distances.*

Gambar mengetahui jarak jarak terbaik dari kata cantik :



Gambar 3.8 Hasil verifikasi

Keterangan gambar diatas :

Contohnya adalah “c” pada CcANTIkK paling dekat dengan “K” dari semua huruf yang ada pada CANTIK, begitu juga dengan “c” dari suara uji akan memiliki jarak yang terkecil dengan “C” dari *template* referensi. Pendekatan ini akan dilakukan untuk semua *template* referensi yang ada pada sistem. Pencocokan waktu diperlihatkan oleh jalur terbaik yang terbentuk.

Sistem ini bekerja berdasarkan pencocokan pola, dengan jalan membandingkan jarak, yaitu jarak pada posisi diagonal bawah (i-1,j-1), samping kiri (i-1,j) dan bawah (i,j-1) jarak manakah yang paling kecil jarak itulah yang

dipilih. Berdasarkan gambar 3.9 dapat dihitung jarak dari jarak lokal yang ada (d). Apabila d(i,j) adalah jarak lokal pada posisi (i,j) pada matrik, maka jarak D(i,j) pada posisi (i,j) dengan kondisi awal (inisialisasi) D(1,1) = d(1,1). Nilai jarak terakhir D(I,J) pada posisi (I,J) pada matrik, menyatakan jarak pencocokan keseluruhan untuk pencocokan sebuah *template* referensi dengan suara uji. Masukan ucapan vokal akan dikenali sebagai sebuah vokal sesuai *template* referensi yang memiliki jarak pencocokan keseluruhan yang terkecil, atau nilai D(I,J) terkecil bila dibandingkan nilai D(I,J) dari pencocokan dengan *template* referensi lainnya.

3.3.7 Analisis Kebutuhan Sistem

Analisis kebutuhan sistem menjelaskan spesifikasi dari perangkat lunak yang dibutuhkan untuk menjalankan sistem agar sesuai aplikasi yang dirancang serta sesuai dengan kebutuhan pengguna. Sebelum melakukan perancangan sistem, terlebih dahulu dilakukan analisis kebutuhan sistem dengan memperhatikan cara penyampaian informasi penggunaan agar dapat diatasi dalam sistem yang akan dirancang. Adapun kebutuhan sistem yang diperlukan antara lain :

1. Menampilkan informasi judul lagu dan lirik lagu sesuai dengan input pencarian yang terdapat dalam sistem.
2. Menampilkan informasi judul lagu dan lirik lagu berupa file musik yang dapat di dengar.

Sistem yang akan dibangun ini adalah suatu aplikasi mobile menggunakan sistem operasi Android dengan memanfaatkan metode pitch detection dan backpropagation. Teknologi dengan aplikasi mobile ini dapat memudahkan pengguna untuk mengakses aplikasi dimana saja dan kapan saja selama aplikasi ini terkoneksi dengan internet.

Beberapa hal yang diperhatikan lainnya dalam perekaman suara diantaranya :

1. kondisi lingkungan yang minim *noise*
2. penggunaan perangkat yang sama , seperti penentuan peralatan *microphone,* soundcard, volume , dan pelafalan atau pengucapan kata-kata
3. pelafalan dengan kondisi normal (tidak terlalu cepat dan tidak terlalu lambat)
4. amplitudo (keras lemah) dengan intonasi normal (tidak terlalu tinggi dan tidak terlalu lambat.

3.3.8 Kebutuhan Non Fungsional

Analisis kebutuhan non fungsional merupakan analisis yang dibutuhkan untuk menentukan spesifikasi kebutuhan sistem. Spesifikasi yang meliputi elemen atau komponen-komponen yang dibutuhkan untuk sistem yang akan dibangun hingga sistem tersebut diimplementasikan. Analisis kebutuhan ini menentukan spesifikasi masukan yang diperlukan sistem, keluaran yang akan dihasilkan sistem dan proses yang dibutuhkan untuk mengolah masukan sehingga menghasilkan suatu keluaran yang diinginkan.

3.3.8.1 Kebutuhan Perangkat Keras

Agar aplikasi dapat berjalan dengan baik, maka dibutuhkan perangkat keras (*hardware*) yang sesuai dengan kebutuhan aplikasi. Tabel 3.9 adalah spesifikasi minimum perangkat keras yang dibutuhkan agar dapat menjalankan aplikasi pengoptimalan *algoritma pitch detection* pada aplikasi pencarian judul lagu melalui media suara secara optimal.

Tabel 3.7 Spesifikasi perangkat keras

Smartphone	Spesifikasi Perangkat Keras
Prosesor	Prosesor 1,6 Ghz
Resolusi Layar	800 x 600 *pixel*
RAM	128 Mb
Hardisk	2Gb
Keyboard dan *mouse*	Standar
Sound card	Standar

3.3.8.2 Kebutuhan Perangkat Lunak

Perangkat lunak (*software*) merupakan hal terpenting dalam mendukung kinerja sebuah sistem. Perangkat lunak digunakan dalam sebuah sistem untuk memberikan perangkat keras (*hardware*) sebuah perintah agar perangkat keras melakukan apa yang diinginkan oleh pemain. Berikut ini, kebutuhan perangkat lunak untuk pengoptimalan *algoritma pitch detection* pada aplikasi pencarian judul lagu melalui media suara pada table 3.8

Tabel 3.8 Spesifikasi perangkat lunak

Spesifikasi Perangkat Lunak
OS *Android* versi 2.3 ke atas (Gingerbread)

3.3.9 Kebutuhan Fungsional

Analisis kebutuhan fungsional menggambarkan proses kegiatan yang akan diterapkan dalam sebuah sistem dan menjelaskan kebutuhan yang diperlukan sistem agar sistem dapat berjalan dengan baik serta sesuai dengan kebutuhan. Analisis kebutuhan fungsional ini meliputi analisis kebutuhan sistem, analisis kebutuhan data dan pemodelan sistem.

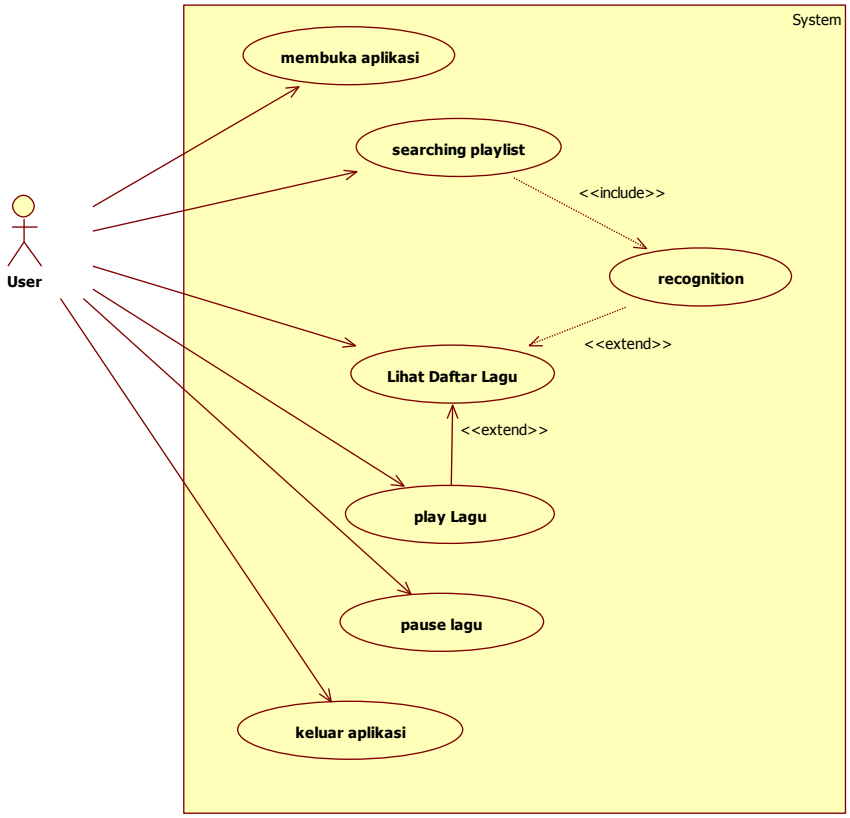
Pemodelan sistem dimodelkan dengan menggunakan *UML (Unified Modeling Language)*. Tahap-tahap pemodelan dalam analisis tersebut antara lain *Use Case diagram, Class Diagram, Activity Diagram, Sequence Diagram, Component Diagram* dan *Deployment Diagram.*

3.9.1 Analisis Kebutuhan Sistem

3.9.2 Pemodelan Sistem

A. Use Case Diagram

Diagram *Use Case* merupakan bagian tertinggi dari fungsionalitas yang dimiliki sistem yang akan menggambarkan bagaimana seseorang atau aktor akan menggunakan dan memanfaatkan sistem. Pemodelan yang digunakan pada *aplikasi* yang akan dibuat adalah analisis perancangan dan pengembangan perangkat lunak berbasis objek, dimana alat yang akan digunakan untuk pengembangan perangkat lunak yaitu pemodelan *Unified Modeling Language* (UML).



Gambar 3.9 u*se case diagram*

1. Identifikasi Aktor

   Dalam sistem ini terdapat satu aktor yaitu pengguna, berikut adalah penjelasannya :

Tabel 3.9 Identifikasi Aktor

No.	Aktor	Definisi
1	Pengguna	Merupakan aktor yang menggunakan aplikasi pencarian judul lagu

2. Definisi *Use case*

   Definisi *use case* berfungsi untuk menjelaskan fungsi dari *use case* yang terdapat pada diagram use case. Definisi *use case* dijelaskan pada table

Tabel 3.10 Definisi Use Case

No	*Use Case*	Keterangan
1.	Membuka Aplikasi	Fungsionalitas untuk menampilkan informasi list menu yang ada dalam aplikasi
2	Mencari Musik	Proses untuk memulai mencari file musik
3	Recognition	Proses Pengenalan Ucapan
4	Play Music	Fungsionalitas untuk dapat memainkan lagu yang telah keluar sebagai hasil pencarian
5	Pause Music	Fungsionalitas untuk dapat memainkan lagu yang telah keluar sebagai hasil pencarian
6	Keluar	Proses untuk keluar dari aplikasi

3. Skenario *Use case*

Skenario *use case* merupakan penjelasan mengenai alur kejadian yang di dalamnya menjelaskan urutan interaksi antara aktor dengan *use case* tersebut. Adapun skenario *use case* digambarkan sebagai berikut.

Tabel 3.11 Skenarion Use Case Mebuka Aplikasi

Skenario	
Nomor	1
Nama	Membuka Aplikasi
Tujuan	Masuk kedalam menu utama
Deskripsi	Proses untuk masuk ke menu utama
Aktor	Pengguna
Skenario Utama	
Kondisi Awal	User menekan tombol aplikasi
Aksi Aktor	Reaksi Sistem
	1. Menampilkan screen pemutar lagu dengan tombol play, dan menu
	2. Pengguna melihat informasi list menu di main menu yang dapat dipilih pengguna

Tabel 3.12 Skenarion Use Case Mencari File Musik

Skenario	
Nomor	2
Nama	Recognition
Tujuan	Mencocokan kata-kata untuk mencari
Deskripsi	Proses untuk mencari file musik
Aktor	Pengguna
Skenario Utama	
Kondisi Awal	User menekan tombol speech recognition
Aksi Aktor	Reaksi Sistem
	1. Mencari file yang sesuai dengan kata yang diucapkan
	2. *Mencocokan* dengan file yang ada
	3. *Menampilkan* hasil pencarian

Tabel 3.13 Skenarion Use Case Proses Recognition

Skenario	
Nomor	3
Nama	Recognition
Tujuan	Mencocokan kata-kata
Deskripsi	Proses untuk mencocokan kata-kata untuk mencari
Aktor	Pengguna
Skenario Utama	
Kondisi Awal	User menekan tombol recognition
Aksi Aktor	Reaksi Sistem
	1. Proses pengucapan
	2. *Mencocokan* dengan file yang ada
	3. *Menampilkan* hasil pencarian

Tabel 3.14 Skenarion Use Case Play Music

Skenarion	
Nomor	4
Nama	Play Music
Tujuan	Memainkan lagu yang keluar sebagai hasil pencarian
Deskripsi	Proses untuk mendengar lagu
Aktor	Pengguna
Skenario Utama	
Kondisi Awal	User menekan tombol play music
Aksi Aktor	Reaksi Sistem
	1. Mengetahui judul lagu dan file lagu
	2. *Memainkan lagu*

Tabel 3.15 Skenarion Use Case Pause Music

Skenarion	
Nomor	5
Nama	Pause Music
Tujuan	Memainkan lagu yang kuleuar sebagai hasil pencarian
Deskripsi	Proses untuk mendengar lagu
Aktor	Pengguna
Skenario Utama	
Kondisi Awal	User menekan tombol play music
Aksi Aktor	Reaksi Sistem
	1. Mengetahui judul lagu dan file lagu
	2. *Memainkan lagu*

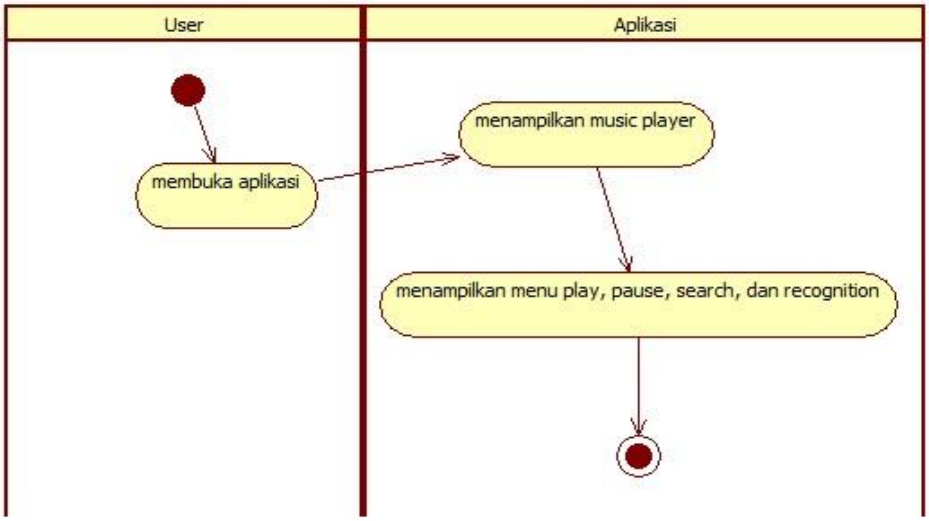
Tabel 3.16 Skenarion Use Case Keluar Aplikasi

Skenarion	
Nomor	6
Nama	Play Music
Tujuan	Keluar dari aplikasi
Deskripsi	Proses Keluar dari menu utama
Aktor	Pengguna
Skenario Utama	
Kondisi Awal	User menekan tombol keluar
Aksi Aktor	Reaksi Sistem
	1. Keluar dari menu aplikasi

B. Activity Diagram

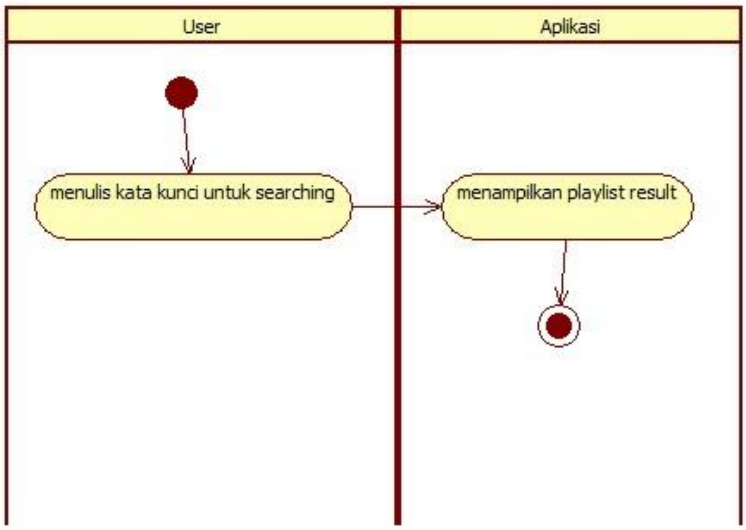
Activity Diagram merupakan bagian dari penggambaran sistem secara fungsional menjelaskan proses-proses logika atau fungsi yang terimplementasi oleh kode program. *Activity Diagram* memodelkan event-event yang terjadi didalam suatu *Use Case* dan digunakan untuk pemodelan aspek dinamis dari sistem.

1. *Activity membuka aplikasi*



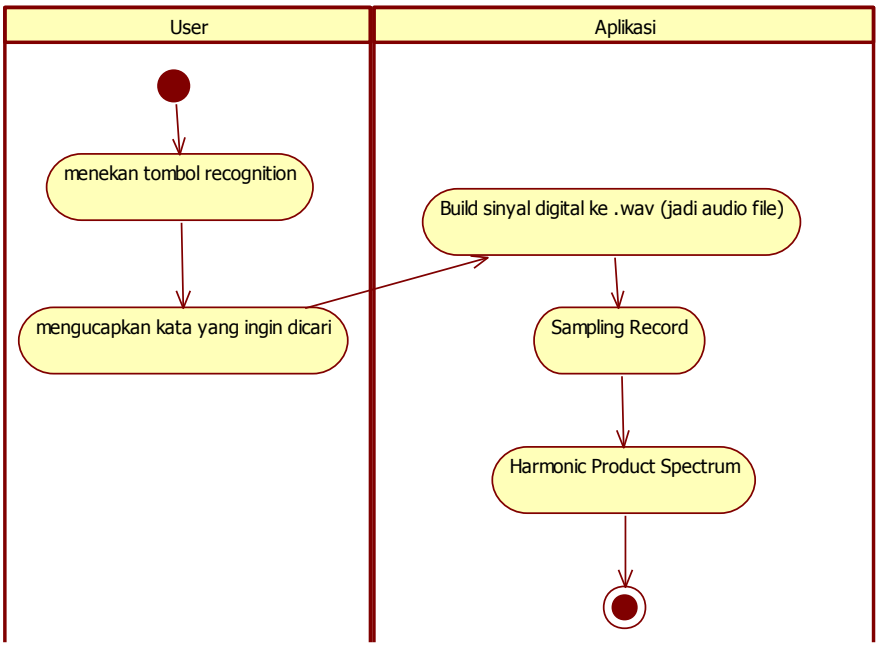
Gambar 3.10 *Activity* membuka aplikasi

2. *Activity mencari lagu*



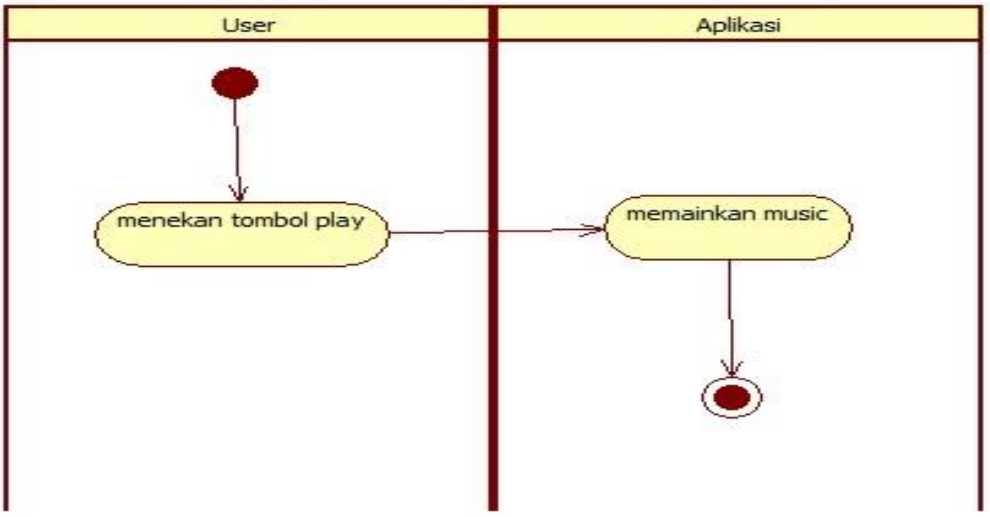
Gambar 3.11 *Activity* mencari lagu

3. *Activity recognition* lagu



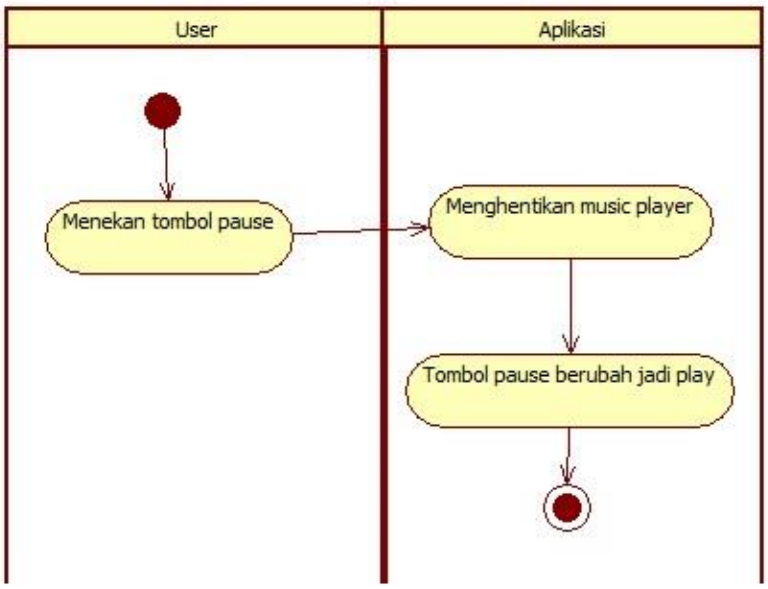
Gambar 3.12 *Activity recognition lagu*

4. *Activity memulai lagu*



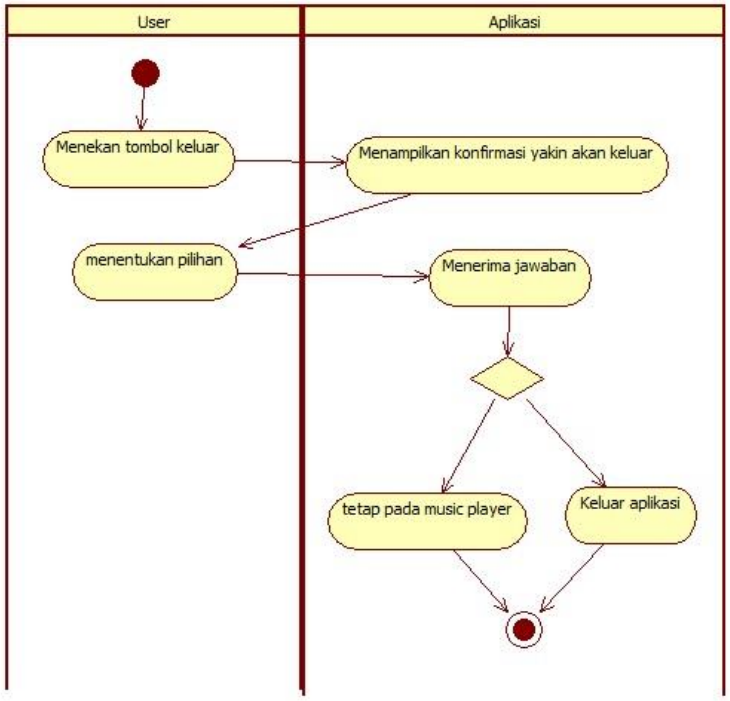
Gambar 3.13 *Activity* memulai lagu

5. *Activity mempause lagu*



Gambar 3.14 *Activity* mempause lagu

6. *Activity* keluar aplikasi

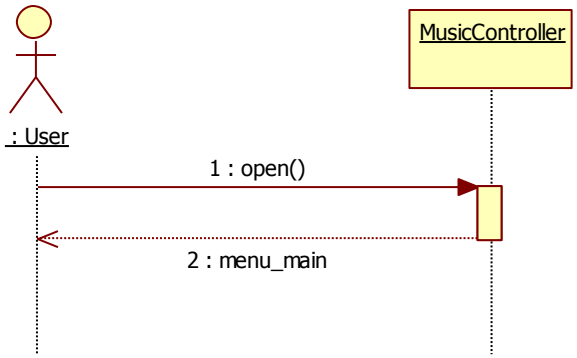


Gambar 3.15 *Activity* keluar aplikasi

C. Squence Diagram

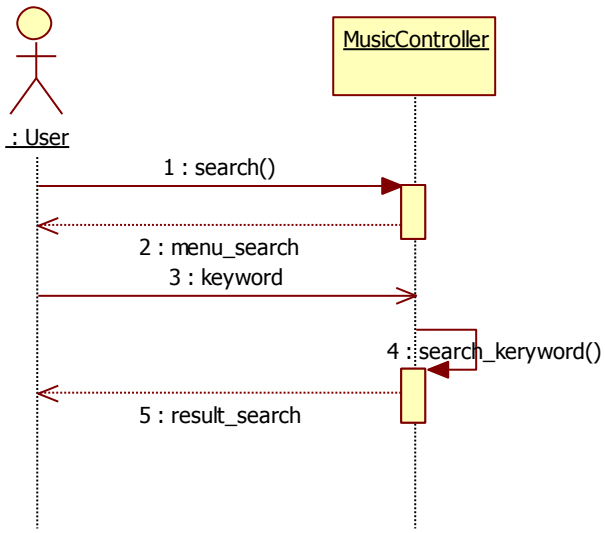
Sequence diagram menggambarkan tindakan objek pada *use case* dengan mendeskripsikan waktu hidup objek dan *message* yang dikirimkan dan diterima antar objek. *Sequence diagram* dapat dilihat pada gambar berikut.

7. *Squence diagram* membuka aplikasi



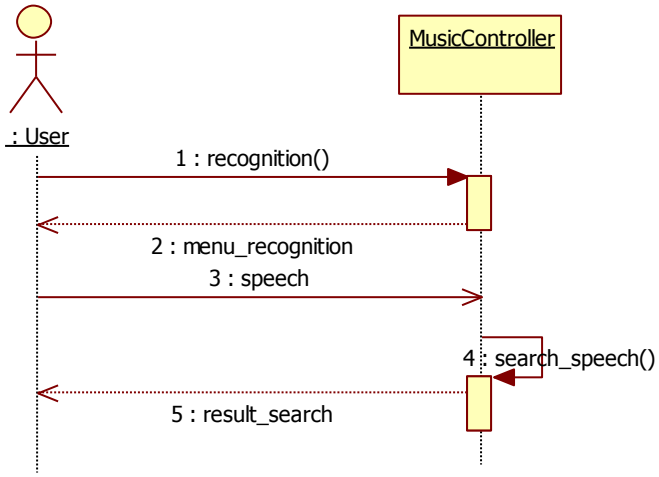
Gambar 3.16 *Squence diagram* membuka aplikasi

8. *Squence diagram* mencari lagu



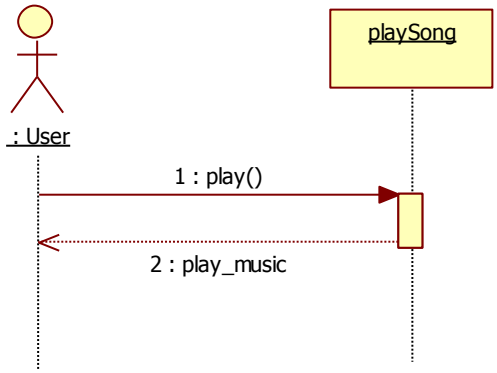
Gambar 3.17 *Squence diagram* mencari lagu

9. *Squence diagram recognition* lagu



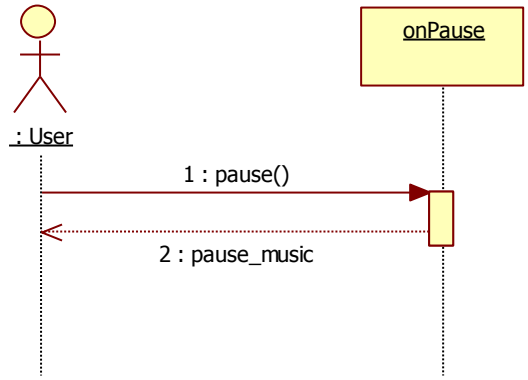
Gambar 3.18 *Squence diagram recognition* lagu

10. *Squence diagram* memulai lagu



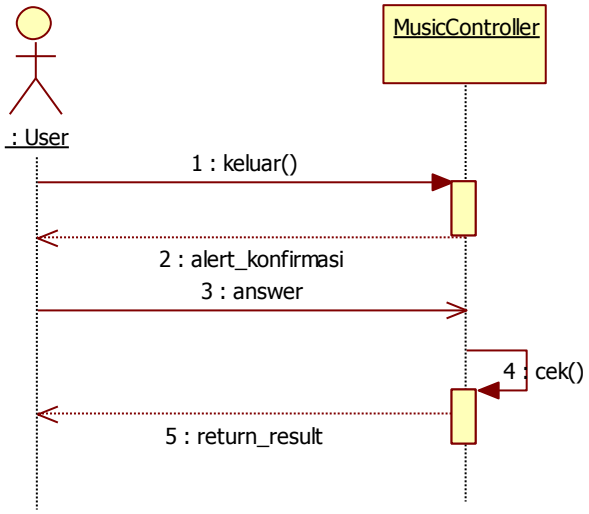
Gambar 3.19 *Squence diagram* memulai lagu

11. *Squence diagram* mempause lagu



Gambar 3.20 *Squence diagram* mempause lagu

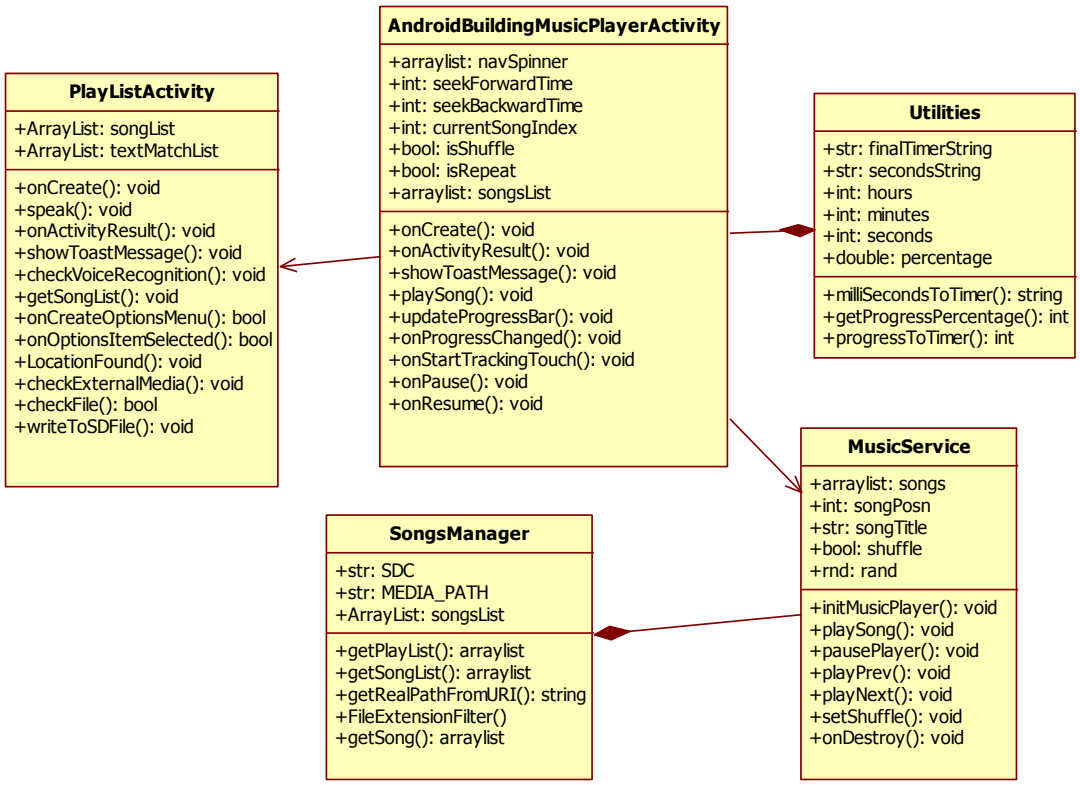
12. *Squence diagram* keluar aplikasi



Gambar 3.21 *Squence diagram* keluar aplikasi

D. Class Diagram

Class diagram merupakan gambaran sistem berupa relasi antar kelas-kelas ataupun paket-paketnya. *Class diagram* dijelaskan melalui gambar 3.22



Gambar 3.22 *Class diagram* aplikasi pemutar lagu

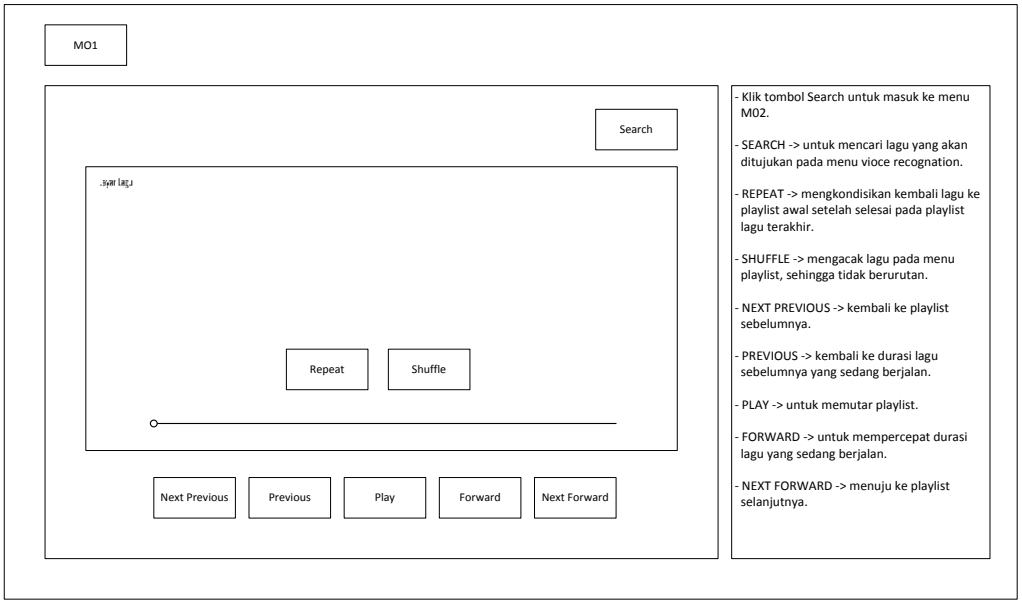
3.2 Perancangan Sistem

Perancangan sistem bertujuan untuk menspesifikasikan aspek-aspek teknik yang menjadi solusi dalam perencanaan. Pada tahap ini perancangan akan didefinisikan secara detail untuk mengatasi masalah-masalah yang lebih teknis, berkaitan dengan kegiatan implementasi seperti perancangan antarmuka.

3.2.1 Perancangan Antarmuka

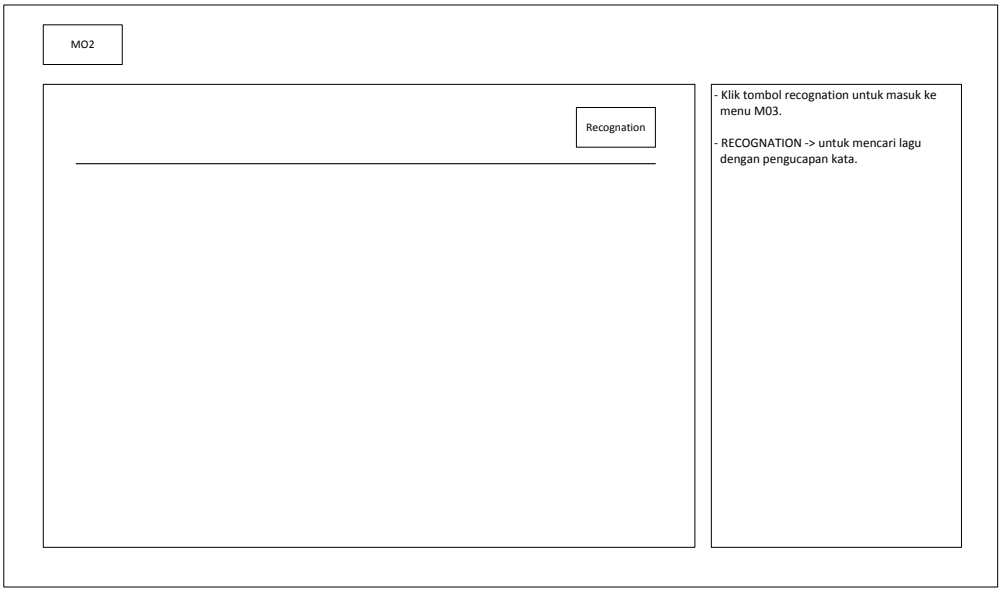
Perancangan antarmuka bertujuan untuk memberikan gambaran tentang aplikasi yang akan dibangun, sehingga akan mempermudah dalam mengimplementasikan aplikasi serta akan memudahkan pembuatan aplikasi. Antarmuka aplikasi merupakan halaman yang muncul pertama kali pada saat menjalankan aplikasi.

1. Antarmuka Aplikasi



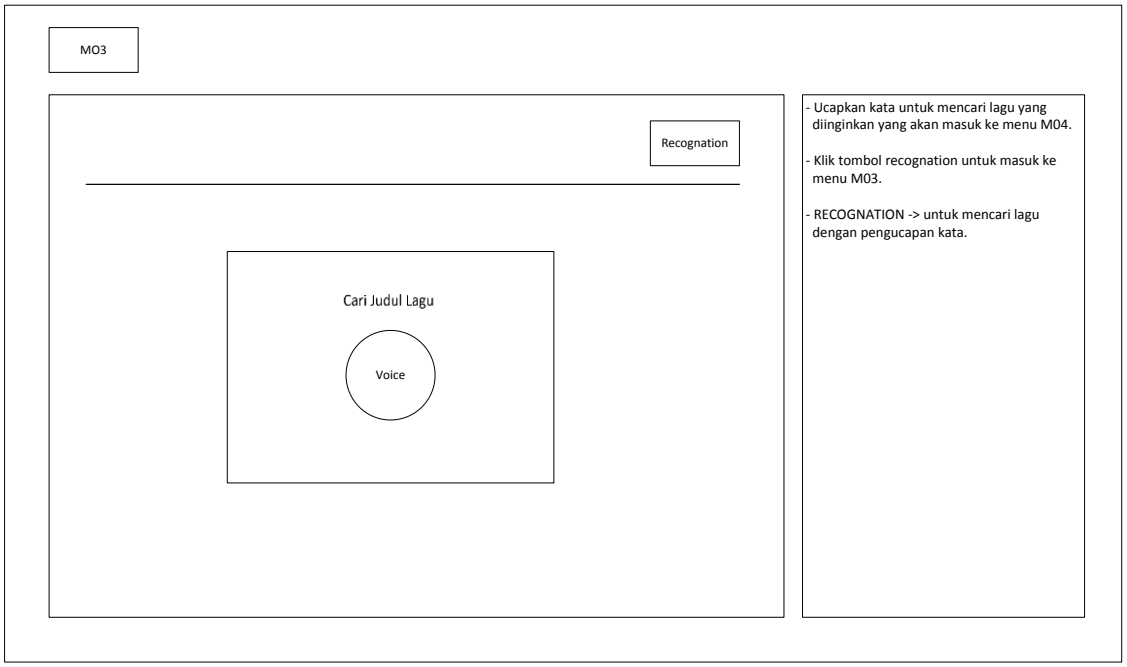
Gambar 3.23 Antarmuka aplikasi

2. Tampilan *playlist*



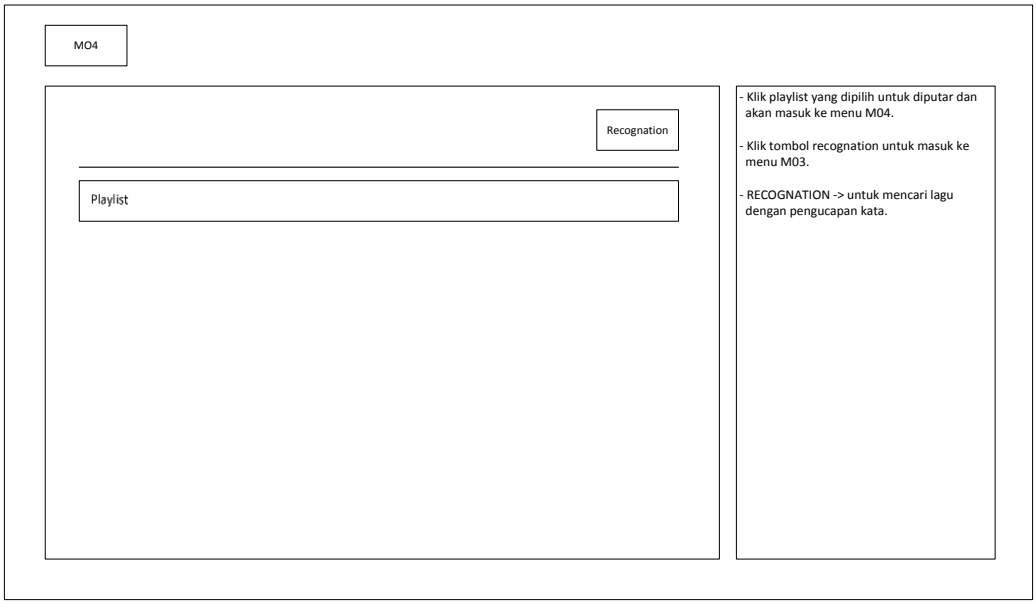
Gambar 3.24 playlist

3. Menu Pencarian lagu



Gambar 3.25 Pencarian lagu.

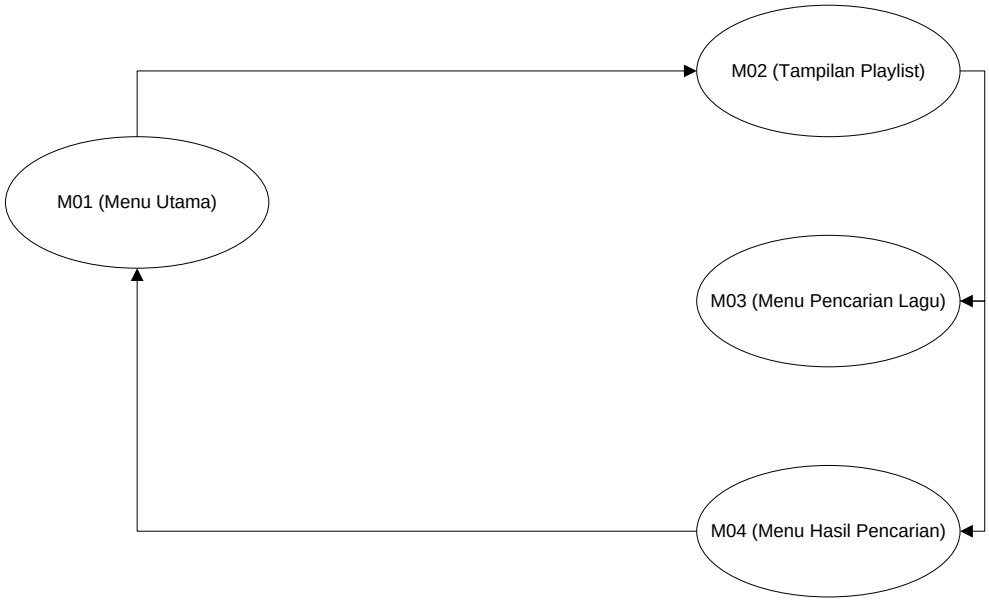
4. Menu Hasil Pencarian



Gambar 3.26 Hasil Pencarian Lagu

3.2.1.1 Jaringan Semantik

Untuk mengetahui keterhubungan dari setiap tampilan yang dibuat dapat menggunakan jaringan semantik pada pengoptimalan *algoritma pitch detection* pada aplikasi pencarian judul lagu melalui media suara.



Gambar 3.27 Jaringan Semantik Aplikasi