BAB 3

ANALISIS DAN PERANCANGAN SISTEM

3.1 Analisis Sistem

Dalam proses pembuatan suatu sistem harus dilakukan penelitian dan penganalisaan tentang sistem yang akan dibangun, berikut adalah beberapa analisis yang dilakukan untuk membangun Sistem Perintah Suara pada Aplikasi Doa Sehari-hari Berbasis Android:

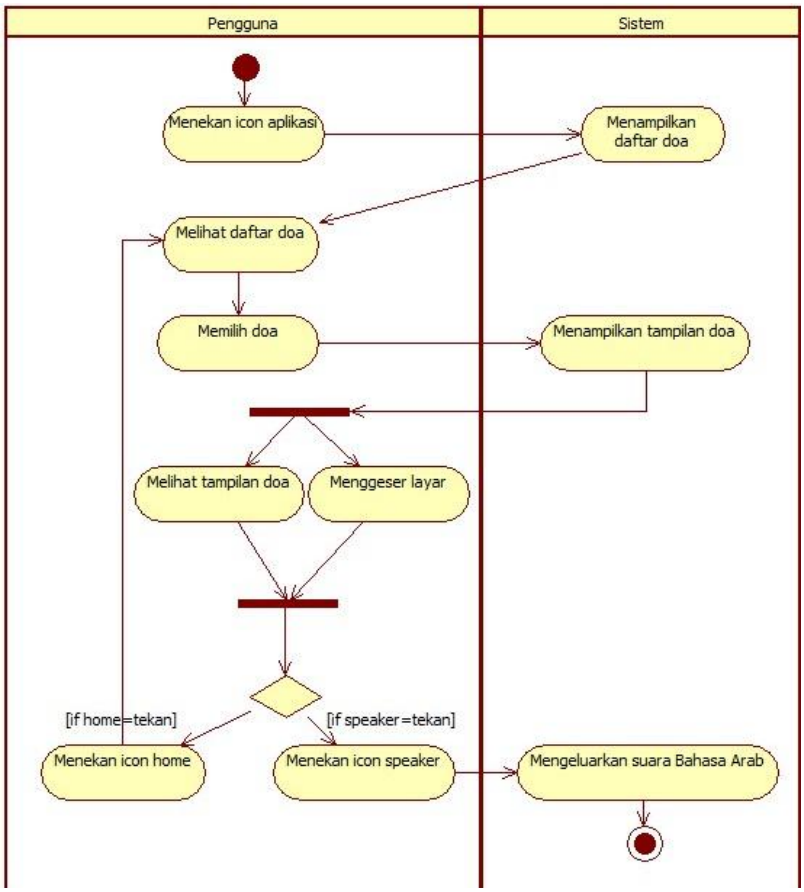
1. Analisis sistem yang berjalan
2. Deskripsi umum sistem yang akan dibangun
3. Analisis pengolahan sinyal suara
4. Proses *Fast Fourier Transform* (FFT)
5. Analisis Pencocokan Kata

3.1.1 Analisis Sistem Yang Berjalan

Dengan kemajuan teknologi dan komunikasi, sekarang sudah tersedia aplikasi pada *smartphone* yang menyediakan doa sehari-hari. Aplikasi doa-doa yang ada masih membuat pengguna kerepotan untuk menggunakannya karena masih menggunakan papan ketik yang kecil pada *handphone* untuk mecari doa yang diinginkan lalu menekan tombol cari yang bisa membuat pengguna lama untuk melakukan pencarian doa yang diinginkan. Jika tidak seperti itu, aplikasi yang ada masih menyuguhkan tampilan daftar doa yang terurut sehingga pengguna harus membaca dan mencari judul doa yang akan dibaca satu persatu. Semua itu

membuat pengguna kurang puas dengan aplikasi yang sudah tersedia, maka dari itu dibuat sistem perintah suara pada aplikasi doa sehari-hari berbasis android.

Gambar 3.1 di bawah ini merupakan *activity diagram* aplikasi yang berjalan.



Gambar 3.1 *Activity Diagram* Aplikasi yang berjalan

3.1.2 Deskripsi Umum Sistem Yang Akan Dibangun

Proses pengidentifikasian kata pada sistem dapat dijelaskan sebagai berikut pertama-tama ketika pengguna membuka aplikasi ini maka tampilan pertamanya ialah daftar doa sehari-hari. Daftar doa tersebut dimulai dari doa bangun tidur sampai tidur kembali yang bisa dipilih dengan perintah suara berdasarkan namanya setelah menekan tombol bicara atau dengan cara mengklik nama doa

tersebut. Jika nama doa yang dipilih tidak sesuai dengan data yang ada, maka akan keluar sebuah peringatan bahwa doa tersebut tidak ada. Jika nama doa yang dipilih sesuai, maka keluarannya *(output)* berupa tampilan doa yang terdiri dari tulisan Bahasa Arab dilengkapi dengan terjemahan berbahasa Indonesia serta sumber yang jelas.

Deskripsi secara umum dalam sistem perintah suara ini adalah dimulai dari tahap pengolahan sinyal, ekstraksi dan pencocokan kata.

Gambar di bawah ini merupakan diagram blok proses sistem perintah suara.



Gambar 3.2 Proses Sistem Perintah Suara

Gambar 3.2 menunjukkan garis besar proses perintah suara pada sistem yang akan dibangun. Berikut merupakan penjelasan dari diagram blok di atas:

1. Analisis Pengolahan Sinyal Suara

Dalam analisis pengolahan sinyal suara dibagi menjadi beberapa tahap. Tahap pertama adalah pemilteran sinyal suara yang berupa sinyal analog dan mengubah sinyal analog ke digital, yaitu dengan proses konversi analog ke digital (ADC). Tahap kedua adalah tahap mengekstrak file suara yang menghasilkan sebuah informasi dari file suara tersebut dengan *Fast Fourier Transform* (FFT). Tahap ketiga adalah mencocokan file suara tersebut dengan *sample* yang sesuai menggunakan *Hidden Markov Models* (HMM).

1.2 Konversi Analog ke Digital (ADC)

Sinyal suara yang akan diproses bersifat analog sehingga jika dilakukan pengolahan secara digital, sinyal suara tersebut harus dikonversi menjadi sinyal digital, berupa urutan angka dengan tingkat presisi tertentu yang dinamakan *analog to digital conversion* dengan menggunakan *analog-to-digital converter* (ADC). Konsep kerja ADC terdiri dari tiga proses, yaitu:



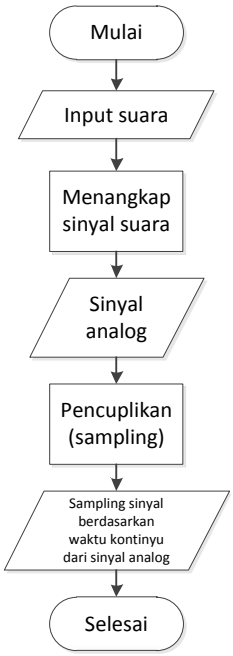
Gambar 3.3 Konsep Kerja ADC *(Analog to Digital Converter)*

Keterangan konsep kerja ADC :

1. *Sampling*

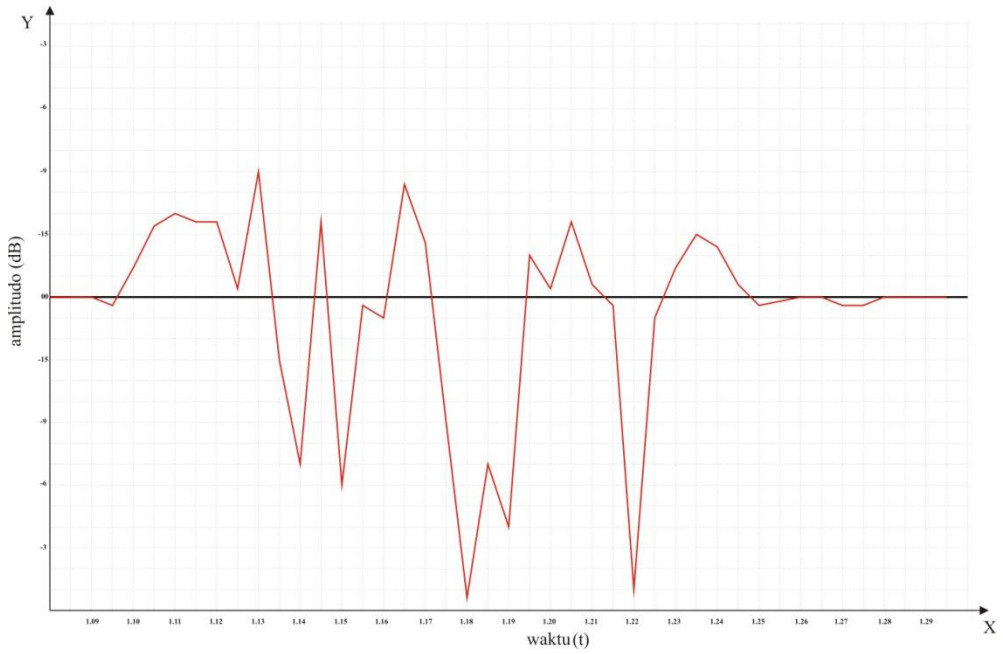
Pada tahap ini yang pertama dilakukan ialah mengambil *sample* yaitu dengan amplitudo yang mengarah ke atas dari sumbu x, lalu mengambil nilai titik-titik diskrit dari sinyal waktu kontinyu supaya mudah untuk dikuantisasi. Titik diskrit merupakan data yang satuannya selalu bulat dalam bilangan asli, tidak berbentuk pecahan.

Proses *sampling* ini mempunyai alur yang digambarkan pada gambar 3.5 berikut.



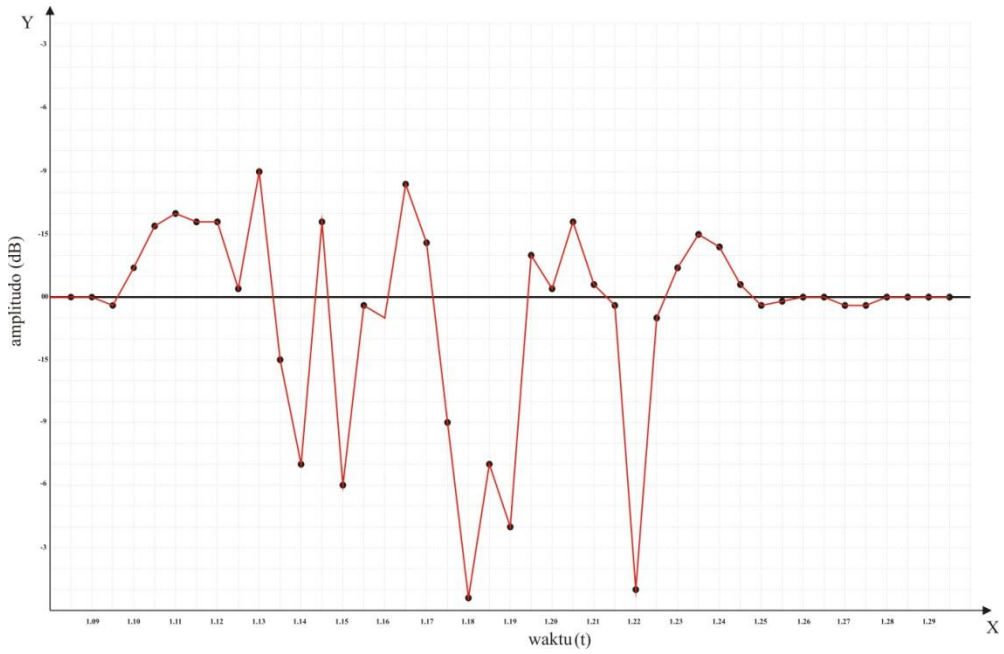
Gambar 3.5 *Flowchart sampling*

Dalam proses *sampling* ini dilakukan pada perintah suara “Baca” dengan panjang durasi rekaman selama 3 detik. Data berupa sinyal suara diperoleh dengan cara merekam suara melalui *microfon smartphone* Android. Suara tersebut mempunyai format .*wav* dengan frekuensi *sampling* adalah 800Hz.



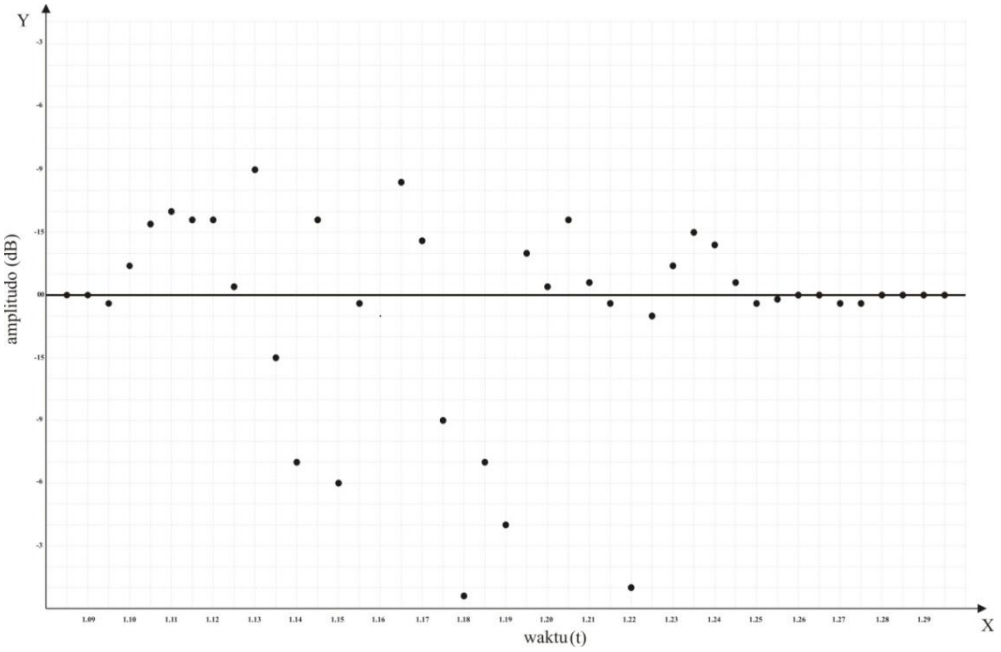
Gambar 3.6 *Sampling* Sinyal Analog

Gambar 3.6 merupakan tahap pencuplikan sinyal, hasil yang ditangkap oleh *microfon* dan digambarkan dengan ketentuan besar dB dan panjang waktu. Gambar 3.6 diperoleh dari sinyal suara yang kemudian diperjelas dengan membuat garis lurus berwarna merah mengikuti ujung-ujung gelombang dari sinyal analog supaya memperjelas bentuk *sampling*-nya.



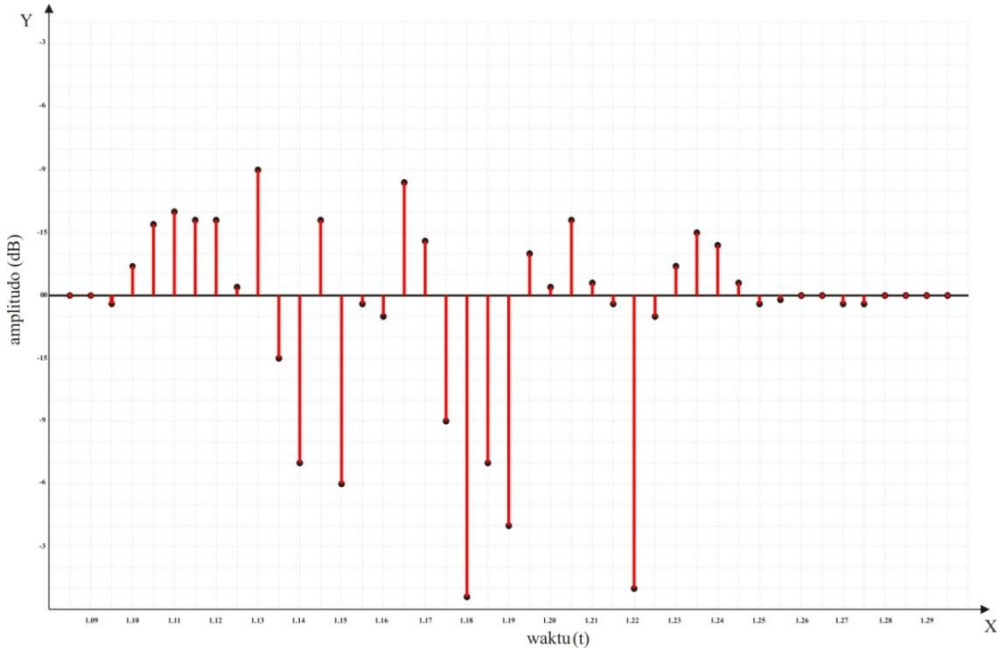
Gambar 3.7 *Sampling* sinyal waktu kontinyu

Gambar 3.7 merupakan *sampling* sinyal waktu kontinyu yang digambarkan di bawah rentang waktu *(milli second)*. Gambar 3.7 digambarkan berdasarkan Gambar 3.6 yang hanya ditambah titik-titik diskrit pada setiap ujung tertinggi suatu gelombang untuk mempermudah pada saat perhitungan di tahap *Coding.*



Gambar 3.8 Titik-titik diskrit

Gambar 3.8 dibuat berdasarkan sinyal analog yang digambarkan menjadi titik-titik diskrit menurut ujung sinyal terpanjang dari suatu gelombang.

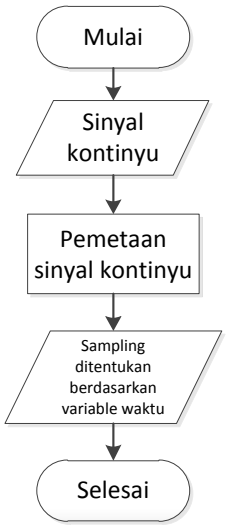


Gambar 3.9 *Sampling* setelah ditentukan berdasarkan variabel waktu

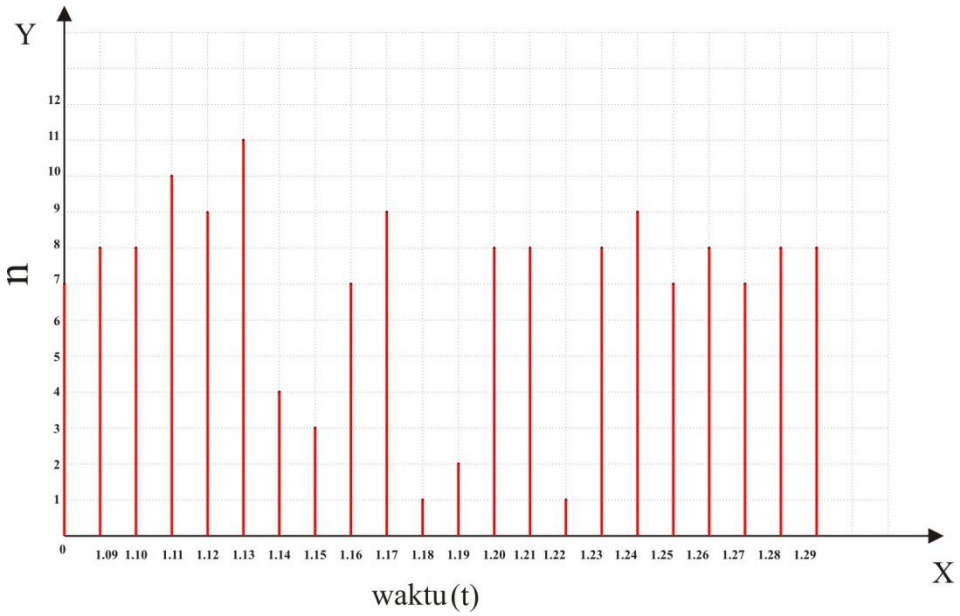
Gambar 3.9 merupakan nilai diskrit yang ditentukan dengan cara menentukan titik berdasarkan waktu yang bersesuaian dengan sinyal kontinyu tersebut. Cara pemberian titik diskrit yaitu dengan menarik garis lurus dimulai dari indeks 0 dB yang merupakan garis tengah pada sinyal sampai garis lurus tersebut sampai ke sinyal kontinyu. Jumlah titik yang diambil setiap detik dinamakan *Sampling rate.*

2. Kuantisasi

Kuantisasi merupakan proses pemetaan dari nilai sinyal kontinyu menjadi nilai-nilai yang diskrit sehingga didapatkan sinyal nilai diskrit, dan sinyal diskrit diambil dari hasil *sampling*.



Gambar 3.10 *Flowchart* kuantisasi



Gambar 3.11 Sinyal yang Dikuantisasikan

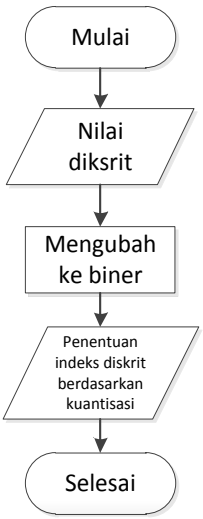
Gambar 3.11 merupakan tahap kuantisasi sinyal yang telah melewati tahap *sampling* berdasarkan indeks bilangan diskrit yang diambil dari proses *sampling*. Tahap penentuan titik nilai diskrit yaitu dengan cara menarik garis dari indeks waktu menuju titik *sampling* yang mendekati indeks bilangan diskrit (n).

Dari hasil kuantisasi pada gambar Gambar 3.11 maka dapat diambil data suara hasil sinyal diskrit dengan diwakili sebagai nilai $f(x)$ sebagai berikut:

Frekuensi	Titik diskrit
f(0)	7
f(1)	8
f(2)	8
f(3)	10
f(4)	9
f(5)	11
f(6)	4
f(7)	3

3. *Coding*

Pada proses ini, tiap nilai diskrit yang telah didapat, dipresentasikan dengan angka biner. Gambar 3.12 di bawah ini merupakan alur proses *coding.*



Gambar 3.12 *Flowchart coding*

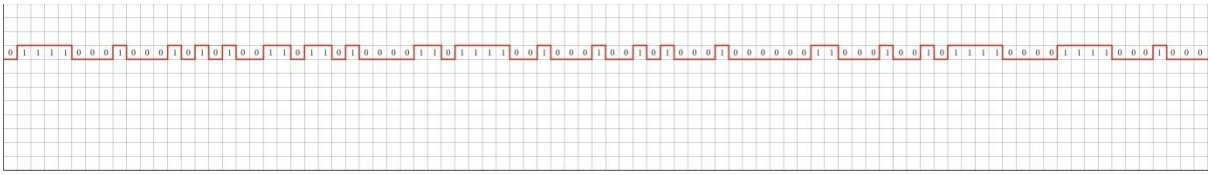
Sinyal analog pada Gambar 3.7 dipresentasikan menjadi digital yang dapat dilihat pada Tabel 3.1

Tabel 3.1 Perhitungan Konversi Sinyal Diskrit ke *Biner*

Nilai diskrit dari hasil kuantisasi	**Konversi ke biner**	**Kode biner**
Titik ke-1 : 7	7 mod 2 =3 sisa 1 3 mod 2 =1 sisa 1 1 mod 2 = 0 sisa 1	0111
Titik ke-2 : 8	8 mod v 2 = 4 sisa 0 4 mod 2 = 2 sisa 0 2 mod 2 =1 sisa 0	1000
Titik ke-3 : 8	8 mod 2 = 4 sisa 0 4 mod 2 = 2 sisa 0 2 mod 2 =1 sisa 0	1000
Titik ke-4 : 10	10 mod 2 = 5 sisa 0 5 mod 2 = 2 sisa 1 2 mod 2 =1 sisa 0	1010
Titik ke-5 : 9	9 mod 2 = 4 sisa 1 4 mod 2 = 2 sisa 0	1001

	2 mod 2 =1 sisa 0	
Titik ke-6 : 11	11 mod 2 = 5 sisa 1 5 mod 2 = 2 sisa 1 2 mod 2=1 sisa 0	1011
Titik ke-7 : 4	4 mod 2 = 2 sisa 0 2 mod 2 = 1 sisa 0 1 mod 2 = 0 sisa 1	0100
Titik ke-8 : 3	3 mod 2 = 1 sisa 1 1 mod 2 = 0 sisa 1 0 mod 2 = 0 sisa 0	0011

Kode *biner* hanya akan dibaca oleh prosesor komputer untuk pembacaan suatu sinyal digital.



Gambar 3.13 Bentuk Sinyal Digital

Gambar 3.13 digambarkan berdasarkan angka biner 0 dan 1 yang sudah dihitung sebelumnya pada Tabel 3.1. Jika binernya 0 maka kotak garisnya ke bawah dan jika binernya 1 maka kotak garisnya ke atas.

2. Analisis Ekstraksi Informasi

Analisis sinyal adalah kegiatan melakukan ekstraksi terhadap semua informasi yang terdapat di suatu sinyal. Adapun langkah-langkah analisis LPC untuk pengenalan suara adalah sebagai berikut:



Gambar 3.14 Diagram blok langkah-langkah LPC

1. *Preemphasis* (penekanan sinyal) adalah suatu proses produksi suara manusia, radiasi pada bibir dan lidah ketika proses *phonation* mengakibatkan komponen

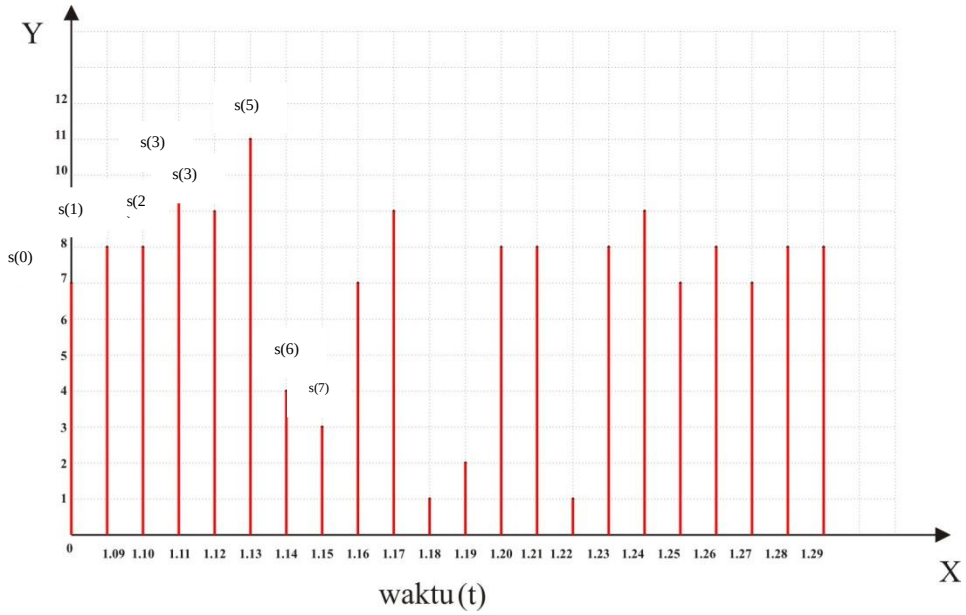
frekuensi tingginya. Tujuan preemphasis adalah untuk memperbaiki sinyal dari gangguan *noise* sehingga meningkatkan akurasi pengenalan suara untuk tahap selanjutnya. Rumus yang digunakan untuk proses *preemphasis*, yaitu:

$$s(n) = s(n) - as(n-1) \quad (4)$$

dengan keterangan yang ada dalam skripsinya Badri Munawar yang berjudul Pengidentifikasian Kata Dengan Menggunakan Metode *Hidden Markov Model* (HMM) Melalui Ekstraksi Ciri *Linear Predictive Coding* (LPC) bahwa

"$s(n)$ adalah sampel ke-n dan harga a yang paling sering digunakan adalah 0,95."

Nilai $s(n)$ diperoleh dari hasil kuantisasi seperti pada gambar berikut:



Contoh perhitungan *Preemphasis*:

$$s(n) = s(n) - as(n-1)$$
$$s(0) = 7 - 0{,}95 * s(0-1) \rightarrow \text{s(0-1) dibaca s(1)}$$
$$= 7 - 0{,}95 * (-8) \rightarrow \text{(-8) diperoleh dari nilai s(1)}$$
$$= 7 + 7{,}6 = 14{,}6$$

Demikian untuk perhitungan $s(1)$ sampai $s(7)$ dan hasilnya dapat dilihat pada tabel 3.4.

Tabel 3.2 *Preemphasis*

$s(n)$	Hasil *Preemphasis*
$s(0)$	14,6
$s(1)$	1,35
$s(2)$	0,4
$s(3)$	2,4
$s(4)$	-0,5
$s(5)$	2,45
$s(6)$	-6,45
$s(7)$	-0,8

2. *Framming*

Pada tahap ini sinyal hasil *preemphasis* dikelompokkan ke dalam bingkai-bingkai dengan ukuran masing-masing bingkai sebesar N data. Bingkai ini berurutan dengan pemisahan antara kedua bingkai sebesar M data. Biasanya M = 1/3 N.

$$s(n) = M < N \tag{5}$$

$$x_n = s(n)(M + n)$$

Contoh perhitungan *Framming*:

Dengan waktu tiap 10 ms dan *sample rate* = 800 Hz maka didapat

$N = 800 * 0{,}01 = 8$ *samples*

Overlapping frame (M) = 3

$x(0) = 14{,}6(3 + 0) = 43{,}8$

Demikian untuk perhitungan $x(1)$ sampai $x(7)$ dan hasilnya dapat dilihat pada tabel 3.5.

Tabel 3.3 *Framming*

$x(n)$	Hasil *Framming*
$x(0)$	43,8
$x(1)$	6,75
$x(2)$	2,4
$x(3)$	16,8
$x(4)$	-4
$x(5)$	22,05
$x(6)$	-64,5
$x(7)$	-8,8

3. *Windowin*g

Melakukan *windowing* terhadap setiap *frame* yang telah dibentuk untuk meminimalkan diskontinuitas pada ujung awal dan ujung akhir setiap *frame* dengan persamaan *Hamming window* untuk sampel ke-n adalah:

$$w(n) = 0{,}54 - 0{,}46 * \cos\left(\frac{2\pi n}{N-1}\right),\ 0 \leq n \leq N-1 \qquad (6)$$

hasil yang didapatkan lalu dikalikan dengan sampel.

Contoh perhitungan *windowing*:

Dik: π=3,14

N=8

n=0,1,2,3,...

Dit: w(n)

Jawab: $w(n) = 0{,}54 - 0{,}46 * \cos\left(\frac{2\pi n}{N-1}\right)$

$w(0) = 0{,}54 - 0{,}46 * \cos\left(\frac{2*3{,}14*0}{8-1}\right)$

$w(0) = 0{,}54 - 0{,}46 * \cos(0)$

$w(0) = 0{,}54 - 0{,}46 * 1$

$w(0) = 0{,}08$

Demikian untuk perhitungan $w(1)$ sampai $w(7)$ dan hasilnya dapat dilihat pada tabel 3.6.

Tabel 3.4 *Windowing*

$W(n)$	Hasil *Windowing*
$W(0)$	0,08
$W(1)$	0,08
$W(2)$	0,0792
$W(3)$	0,0792
$W(4)$	0,0792
$W(5)$	0,0792
$W(6)$	0,0792
$W(7)$	0,0792

2.1 Proses *Fast Fourier Transform (FFT)*

Fast Fourier Transform (FFT) itu bertugas mengekstrak file suara yang menghasilkan sebuah informasi dari file suara tersebut.

Fast Fourier Transform (FFT) merupakan algoritma yang mengimplementasikan *Discreet Fourier Transform* (DFT) dengan teknik perhitungan yang cepat serta memanfaatkan sifat periodikal dari transformasi *fourier*.

DFT dilakukan dengan mengimplementasikan sebuah transformasi, dengan panjang vektor *N* berdasarkan rumus :

$$X(m) = \sum_{n=0}^{N-1} x(n) e^{-\frac{j2\pi nm}{N}} \quad (7)$$

Dimana x(n) adalah sekuen diskrit nilai *sample* dalam fungsi waktu dari variabel *continue* x(t).

Persamaan (7) tersebut jika dinyatakan dalam bentuk *rectangular* didapat:

$$X(m) = \sum_{n=0}^{N-1} x(n) \left[\cos\left(\frac{2\pi nm}{N}\right) - jsin\left(\frac{2\pi nm}{N}\right)\right] \quad (8)$$

Dimana X(m) = komponen *output* DFT ke-m (contoh: x(0), x(1), dst)

m = 0,1,2,3,..., N-1

x(n) = sekuens *sample input*, x(0), x(1), x(2), dst.

n = *indeks sample input* dalam fungsi waktu, n=0,1,2,3,..., N-1

j = V-1

N = jumlah *sample* sekuens input dan jumlah titik frekuensi di *output* DFT

Untuk lebih jelasnya, sebagai contoh saat N=8, n dan m bernilai 0 sampai 7, maka persamaan (8) akan mempunyai keluaran sebagai berikut:

$$X(m) = \sum_{n=0}^{8} x(n) \left[\cos\left(\frac{2\pi nm}{8}\right) - jsin\left(\frac{2\pi nm}{8}\right)\right]$$

Sebagai contoh untuk m = 0

$$x(0) = x(0)\cos\left(2\pi * 0 * \frac{0}{8}\right) - jx(0)\sin\left(2\pi * 0 * \frac{0}{8}\right) +$$
$$x(1)\cos\left(2\pi * 1 * \frac{0}{8}\right) - jx(1)\sin\left(2\pi * 1 * \frac{0}{8}\right) +$$
$$x(2)\cos\left(2\pi * 2 * \frac{0}{8}\right) - jx(2)\sin\left(2\pi * 2 * \frac{0}{8}\right) +$$
$$x(3)\cos\left(2\pi * 3 * \frac{0}{8}\right) - jx(3)\sin\left(2\pi * 3 * \frac{0}{8}\right) +$$
$$x(4)\cos\left(2\pi * 4 * \frac{0}{8}\right) - jx(4)\sin\left(2\pi * 4 * \frac{0}{8}\right) +$$
$$x(5)\cos\left(2\pi * 5 * \frac{0}{8}\right) - jx(5)\sin\left(2\pi * 5 * \frac{0}{8}\right) +$$
$$x(6)\cos\left(2\pi * 6 * \frac{0}{8}\right) - jx(6)\sin\left(2\pi * 6 * \frac{0}{8}\right) +$$
$$x(7)\cos\left(2\pi * 7 * \frac{0}{8}\right) - jx(7)\sin\left(2\pi * 7 * \frac{0}{8}\right)$$

m = 1,2,3,...,8 mempunyai penyelesaian seperti di atas. Setiap nilai *output* X(m) adalah penjumlahan dari perkalian titik diantara nilai input sinyal sekuens dan bentuk sinusoidal kompleks. Nilai frekuensi dari gelombang sinusoidal berbeda bergantung pada *sampling rate* (fs), *sampling* sinyal awal, dan jumlah *sample* N. Contoh jika kita mensampling sinyal kontinyu dengan 800 sample/detik, dilakukan dengan *sample rate* DFTE 8-titik, frekuensi dasar gelombang sinusoidalnya adalah fs/N = 800/8 atau 100 Hz. Frekuensi analisa yang lain adalah *multiple* integral dari frekuensi dasar,

x(0) = frekuensi pertama, dengan frekuensi analisa = 0*100 = 0 Hz,

x(1) = frekuensi pertama, dengan frekuensi analisa = 1*100 = 100 Hz,

x(2) = frekuensi pertama, dengan frekuensi analisa = 2*100 = 200 Hz,

...

...

x(7) = frekuensi pertama, dengan frekuensi analisa = 7*100 = 700 Hz

Data suara yang diambil dari hasil kuantisasi sinyal diskrit dengan nilai $f(x)$ sebagai berikut:

$f(x)$	n
f(0)	7
f(1)	8
f(2)	8
f(3)	10
f(4)	9
f(5)	11
f(6)	4
f(7)	3

$$X(m) = \frac{1}{N}\sum_{x=0}^{x=N-1} f(x)\exp\left[-\frac{2j\pi ux}{N}\right] \quad (9)$$

$$X(m) = \frac{1}{N}\sum_{x=0}^{x=N-1} f(x)\left(\cos\left(\frac{2\pi ux}{N}\right) - fsin\left(\frac{2\pi ux}{N}\right)\right) \quad (10)$$

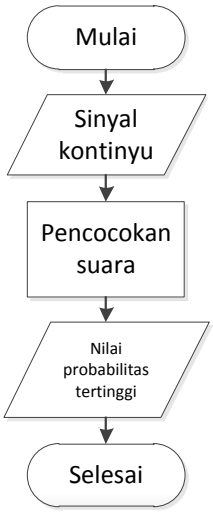
Tabel 3.5 Perhitungan DFT Sinyal Suara

N	$x(m)$	Perhitungan DFT	Hasil dari DFT/$X(m)$
1	$x(0)$	*1/8[7(cos(2*3,14*0*0/8) – j sin(2*3,14*0*0/8) + 8(cos(2*3,14*0*1/8) – j sin(2*3,14*0*1/8) + 8(cos(2*3,14*0*2/8) – j sin(2*3,14*0*2/8) + 10(cos(2*3,14*0*3/8) – j sin(2*3,14i*0*3/8)+ 9(cos(2*3,14*0*4/8) – j sin(2*3,14*0*4/8) + 11(cos(2*3,14*0*5/8) – j sin(2*3,14*0*5/8) + 4(cos(2*3,14*0*6/8) – j sin(2*3,14*0*6/8) + 3(cos(2*3,14*0*7/8) – j sin(2*3,14*0*7/8)]*	7,5
2	$x(1)$	*1/8[7(cos(2*3,14*1*0/8) – j sin(2*3,14*1*0/8) + 8(cos(2*3,14*1*1/8) – j sin(2*3,14*1*1/8) + 8(cos(2*3,14*1*2/8) – j sin(2*3,14*1*2/8) + 10(cos(2*3,14*1*3/8) – j sin(2*3,14i*1*3/8)+ 9(cos(2*3,14*1*4/8) – j sin(2*3,14*1*4/8) + 11(cos(2*3,14*1*5/8) – j sin(2*3,14*1*5/8) + 4(cos(2*3,14*1*6/8) – j sin(2*3,14*1*6/8) + 3(cos(2*3,14*1*7/8) – j sin(2*3,14*1*7/8)]*	7,4-0,3j
3	$x(2)$	*1/8[7(cos(2*3,14*2*0/8) – j sin(2*3,14*2*0/8) + 8(cos(2*3,14*2*1/8) – j sin(2*3,14*2*1/8) + 8(cos(2*3,14*2*2/8) – j sin(2*3,14*2*2/8) + 10(cos(2*3,14*2*3/8) – j sin(2*3,14i*2*3/8)+ 9(cos(2*3,14*2*4/8) – j sin(2*3,14*2*4/8) + 11(cos(2*3,14*2*5/8) – j sin(2*3,14*2*5/8) + 4(cos(2*3,14*2*6/8) – j sin(2*3,14*2*6/8) + 3(cos(2*3,14*2*7/8) – j sin(2*3,14*2*7/8)]*	7,43-0,65j
4	$x(3)$	*1/8[7(cos(2*3,14*3*0/8) – j sin(2*3,14*3*0/8) + 8(cos(2*3,14*3*1/8) – j sin(2*3,14*3*1/8) + 8(cos(2*3,14*3*2/8) – j sin(2*3,14*3*2/8) + 10(cos(2*3,14*3*3/8) – j sin(2*3,14i*3*3/8)+ 9(cos(2*3,14*3*4/8) – j sin(2*3,14*3*4/8) + 11(cos(2*3,14*3*5/8) – j sin(2*3,14*3*5/8) + 4(cos(2*3,14*3*6/8) – j sin(2*3,14*3*6/8) + 3(cos(2*3,14*3*7/8) – j sin(2*3,14*3*7/8)]*	7,38-0,95j
5	$x(4)$	*1/8[7(cos(2*3,14*4*0/8) – j sin(2*3,14*4*0/8) + 8(cos(2*3,14*4*1/8) – j sin(2*3,14*4*1/8) + 8(cos(2*3,14*4*2/8) – j sin(2*3,14*4*2/8) + 10(cos(2*3,14*4*3/8) – j sin(2*3,14i*4*3/8)+*	7,31-1,27j

		*9(cos(2*3,14*4*4/8) – j sin(2*3,14*4*4/8) +* *11(cos(2*3,14*4*5/8) – j sin(2*3,14*4*5/8) +* *4(cos(2*3,14*4*6/8) – j sin(2*3,14*4*6/8) +* *3(cos(2*3,14*4*7/8) – j sin(2*3,14*4*7/8)]*	
6	$x(5)$	*1/8[7(cos(2*3,14*5*0/8) – j sin(2*3,14*5*0/8) +* *8(cos(2*3,14*5*1/8) – j sin(2*3,14*5*1/8) +* *8(cos(2*3,14*5*2/8) – j sin(2*3,14*5*2/8) +* *10(cos(2*3,14*5*3/8) – j sin(2*3,14i*5*3/8)+* *9(cos(2*3,14*5*4/8) – j sin(2*3,14*5*4/8) +* *11(cos(2*3,14*5*5/8) – j sin(2*3,14*5*5/8) +* *4(cos(2*3,14*5*6/8) – j sin(2*3,14*5*6/8) +* *3(cos(2*3,14*5*7/8) – j sin(2*3,14*5*7/8)]*	7,24-1,59j
7	$x(6)$	*1/8[7(cos(2*3,14*6*0/8) – j sin(2*3,14*6*0/8) +* *8(cos(2*3,14*6*1/8) – j sin(2*3,14*6*1/8) +* *8(cos(2*3,14*6*2/8) – j sin(2*3,14*6*2/8) +* *10(cos(2*3,14*6*3/8) – j sin(2*3,14i*6*3/8)+* *9(cos(2*3,14*6*4/8) – j sin(2*3,14*6*4/8) +* *11(cos(2*3,14*6*5/8) – j sin(2*3,14*6*5/8) +* *4(cos(2*3,14*6*6/8) – j sin(2*3,14*6*6/8) +* *3(cos(2*3,14*6*7/8) – j sin(2*3,14*6*7/8)]*	7,13-1,87j
8	$x(7)$	*1/8[7(cos(2*3,14*7*0/8) – j sin(2*3,14*7*0/8) +* *8(cos(2*3,14*7*1/8) – j sin(2*3,14*7*1/8) +* *8(cos(2*3,14*7*2/8) – j sin(2*3,14*7*2/8) +* *10(cos(2*3,14*7*3/8) – j sin(2*3,14i*7*3/8)+* *9(cos(2*3,14*7*4/8) – j sin(2*3,14*7*4/8) +* *11(cos(2*3,14*7*5/8) – j sin(2*3,14*7*5/8) +* *4(cos(2*3,14*7*6/8) – j sin(2*3,14*7*6/8) +* *3(cos(2*3,14*7*7/8) – j sin(2*3,14*7*7/8)]*	7,07-2,18j

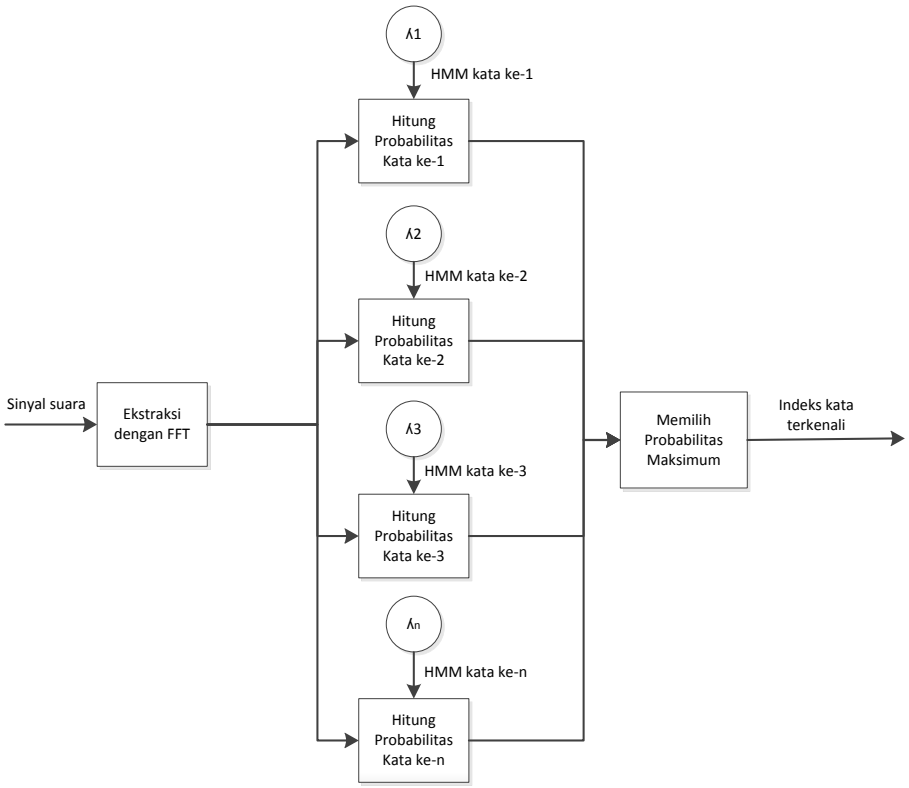
3. Analisis Pencocokan Kata

Dalam pencocokan kata menggunakan metode *Hidden Markov Models* (HMM), dimana HMM ini bertugas untuk memastikan pencocokan file suara dengan template suara yang tersedia. Ada beberapa tahap yaitu ekstraksi sinyal suara dengan menggunakan FFT dan menentukan nilai probabilitas yang maksimum dengan menggunakan *Hidden Markov Models (HMM)*. Pencocokan data suara baru dan hasil ekstraksi akan menggunakan teknik *binary search.*



Gambar 3.15 Alur Proses Pencocokan Kata

Sedangkan Gambar 3.16 di bawah ini merupakan proses umum Markov Models.



Gambar 3.16 Proses *Markov* untuk Pencocokan Kata

Gambar 3.16 menunjukkan proses *Hidden Markov Models* secara umum. Dimana sinyal analog diekstraksi terlebih dahulu menggunakan FFT lalu dihitung nilai probabilitas tertingginya sampai nilai ke-n, dimana n merupakan banyaknya *template* sinyal suara.

3.1.3 Analisis Kebutuhan Non Fungsional

Analisis kebutuhan sistem dilakukan untuk mencari kebutuhan apa saja yang diperlukan untuk membuat sistem perintah suara.

Analisa kebutuhan non fungsional yang dilakukan dibagi dalam tiga tahap, yaitu:

1. Analisis perangkat keras
2. Analisis perangkat lunak
3. Analisis pengguna

3.1.4 Analisis Perangkat Keras

Perangkat keras yang dibutuhkan untuk implementasi sistem perintah suara antara lain:

1. *Processor* yang digunakan *Pentium(R) Dual-Core*
2. *Memory* yang digunakan 2GB
3. *Harddisk*
4. *Speaker* pada *smartphone* Android
5. *Smartphone* Android dengan lebar layar 240 dp dan tinggi layar 320 dp

3.1.5 Analisis Perangkat Lunak

Pada bagian ini dijelaskan mengenai perangkat lunak dari sistem perintah suara. Adapun perangkat lunak yang dibutuhkan antara lain:

1. Sistem Operasi yang digunakan adalah *Windows 7 Professional Service Pack 1 32-bit*.
2. *Eclipse IDE 3.6*

   Merupakan *tools* yang digunakan dalam proses *coding* sistem perintah suara.
3. SDK Android
4. JDK
5. *Operating System* Android versi 2.3.3
6. *Adobe Audition 3.0*

3.1.6 Analisis Pengguna

Spesifikasi minimal bagi pengguna yang dapat menggunakan aplikasi ini adalah sebagai berikut :

1	Usia	:	Lebih dari 10-15 tahun
2	Tingkat Keterampilan	:	Dapat menggunakan *smartphone* Android
3	Pengalaman	:	Pernah memainkan aplikasi di Android

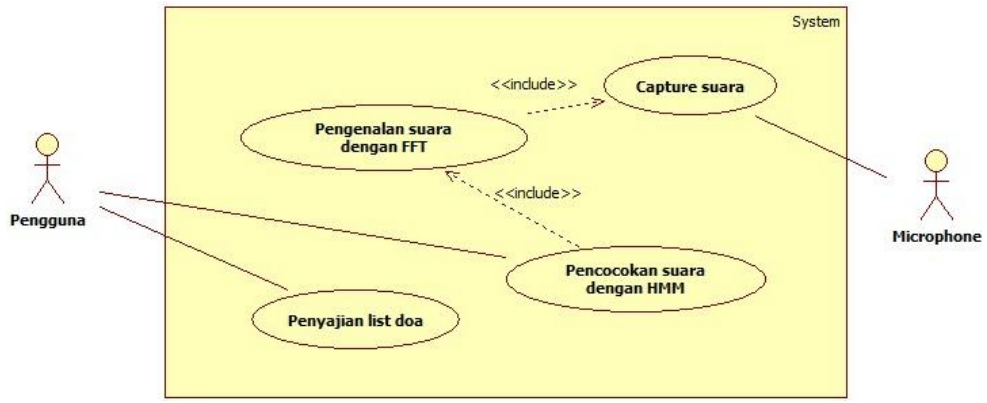
3.1.7 Analisis Kebutuhan Fungsional

Analisis kebutuhan fungsional menggambarkan proses kegiatan yang akan diterapkan dalam sebuah sistem dan menjelaskan kebutuhan yang diperlukan

sistem agar sistem dapat berjalan dengan baik sesuai kebutuhan. Analisis kebutuhan fungsional dimodelkan dengan menggunakan UML (*Unified Modeling Language*). Dimana tahap-tahap perancangan yang dilakukan dalam membangun perangkat lunak pencocokan string antara lain *Use Case Diagram, Class Diagram, Sequence Diagram, Activity Diagram, Deployment Diagram* dan *Package Diagram.*

1. Use Case Diagram

Secara umum interaksi antara pengguna dengan sistem perintah suara ini digambarkan pada diagram *use case* berikut ini:



Gambar 3.17 *Use Case Diagram* Sistem Perintah Suara

Use case terdiri dari tiga bagian, yaitu: definisi aktor, definisi *use case* dan skenario *use sace.*

1.1 Definisi *Actor*

Definisi actor berfungsi untuk menjelaskan actor yang terdapat pada *Use Case Diagram*. Definisi Actor dijelaskan pada tabel 3.6 berikut:

Tabel 3.6 Definisi *Actor*

No	*Use case*	Deskripsi
1	Pengguna	Orang yang meng-*input*-kan suara
2	*Microphone*	Alat yang digunakan untuk meng-*capture* suara

1.2 Definisi *Use Case*

Definisi *use case* berfungsi intuk menjelaskan fungsi *use case* yang terdapat pada *use case* diagram. Definisi *use case* diterangkan pada tabel 3.7 berikut:

Tabel 3.7 Definisi *Use Case*

No	*Use case*	Deskripsi
1	*Capture* suara	Proses menangkap suara
2	Pengenalan suara dengan FFT	Proses yang dilakukan *Fast Fourier Transform* untuk mengenali suara yang masuk
3	Pencocokan suara dengan HMM	Proses yang dilakukan *Hidden Markov Model* untuk mencocokan suara dengan *sample*
4	Penyajian *list* doa	Hasil pencocokan suar berupa *list*

1.3 Skenario *Use Case*

Skenario pada diagram *use case* berisi mengenai alur kejadian yang di dalamnya menjelaskan urutan interaksi antara aktor dengan *use case* tersebut sedangkan diagram aktivitas atau *activity diagram* menggambarkan *workflow* (aliran kerja) atau aktivitas dari sebuah sistem atau proses bisnis.

Berikut adalah skenario jalannya *use case* yang telah didefinisikan sebelumnya:

1.3.1 Skenario *Capture* Doa

Pada Tabel 3.8 Skenario *Use Case Capture* Doa diperlihatkan skenario sistem meng-capture suara.

Tabel 3.8 Skenario *Use Case Capture* Doa

Identifikasi			
Nomor Use Case	1		
Nama Use Case	*Capture* suara		
Deskripsi	Proses menangkap suara		
Trigger	Pengguna meng-*input*-kan suara		
Aktor	*Microphone*		
Skenario Normal			
No	Aksi Aktor	No	Reaksi Sistem
1	*Microphone* mendengarkan suara		
		2	Mengecek diterima atau tidaknya suara
		3	Me-*record* suara
Skenario Ekstention			
1	*Microphone* mendengarkan suara		
		2	Mengecek diterima atau tidaknya suara
		3	Menampilkan pesan
4	*Microphone* mendengarkan suara		
		5	Mengecek diterima atau tidaknya suara
		6	Me-*record* suara

1.3.2 Skenario Pengenalan Suara dengan FFT

Pada Tabel 3.9 Skenario Pengenalan Suara dengan FFT diperlihatkan skenario pengenalan suara.

Tabel 3.9 Skenario Pengenalan Suara dengan FFT

Identifikasi			
Nomor Use Case	2		
Nama Use Case	Pengenalan Suara dengan FFT		
Deskripsi	Proses dimana sistem mengenali suara yang ditangkap		
Trigger	Suara telah di-*capture*		
Aktor	*Microphone*		
Skenario Normal			
No	Aksi Aktor	No	Reaksi Sistem
1	*Microphone* mendengarkan suara		
		2	Mengecek diterima atau tidaknya suara
		3	Me-*record* suara
		4	Mulai membaca *record* suara
		5	Mengenali suara dengan FFT
		6	Menampilkan notifikasi *record* suara berupa gelombang sinyal
Skenario Ekstention			
1	*Microphone* mendengarkan suara		
		2	Mengecek diterima atau tidaknya suara
		3	Me-*record* suara
		4	Mulai membaca *record* suara
		5	Mengenali suara dengan FFT
		6	Menampilkan pesan tidak dikenali
7	*Microphone* mendengarkan suara		
		8	Mengecek diterima atau tidaknya suara
		9	Me-*record* suara
		10	Mulai membaca *record* suara
		11	Mengenali suara dengan FFT
		12	Menampilkan notifikasi *record* suara berupa gelombang sinyal

1.3.3 Skenario Pencocokan Suara dengan HMM

Pada Tabel 3.10 Skenario Pencocokan Suara dengan HMM diperlihatkan skenario pencocokan suara.

Tabel 3.10 Skenario Pencocokan Suara dengan HMM

Identifikasi			
Nomor Use Case	3		
Nama Use Case	Pencocokan Suara dengan HMM		
Deskripsi	Proses dimana sistem mencocokan suara yang telah dikenali		
Trigger	Suara telah dikenali		
Aktor	*Microphone*		
Skenario Normal			
No	Aksi Aktor	No	Reaksi Sistem
1	Pengguna mulai berbicara		
2	*Microphone* mendengarkan suara		
		3	Mengecek diterima atau tidaknya suara
		4	Me-*record* suara
		5	Mulai membaca *record* suara
		6	Mengenali suara dengan FFT
		7	Menampilkan notifikasi *record* suara berupa gelombang sinyal
		8	Melakukan pencocokan suara dengan *sample*
		9	Menampilkan kemungkinan hasil terdekat dengan *sample*
Skenario Ekstention			
1	Pengguna mulai berbicara		
2	*Microphone* mendengarkan suara		
		3	Mengecek diterima atau tidaknya suara
		4	Me-*record* suara
		5	Mulai membaca *record* suara
		6	Mengenali suara dengan FFT
		7	Menampilkan pesan suara tidak dikenali
		8	Melakukan pencocokan suara dengan *sample*
		9	Menampilkan pesan suara tidak cocok
10	Pengguna mulai berbicara		
11	*Microphone* mendengarkan suara		
		12	Mengecek diterima atau tidaknya suara

		13	Me-*record* suara
		14	Mulai membaca *record* suara
		15	Mengenali suara dengan FFT
		16	Menampilkan notifikasi *record* suara berupa gelombang sinyal

1.3.4 Skenario Penyajian *List* Doa

Pada Tabel 3.11 Skenario Penyajian *List* Doa diperlihatkan skenario penyajian *list* doa.

Tabel 3.11 Skenario Penyajian *List* Doa

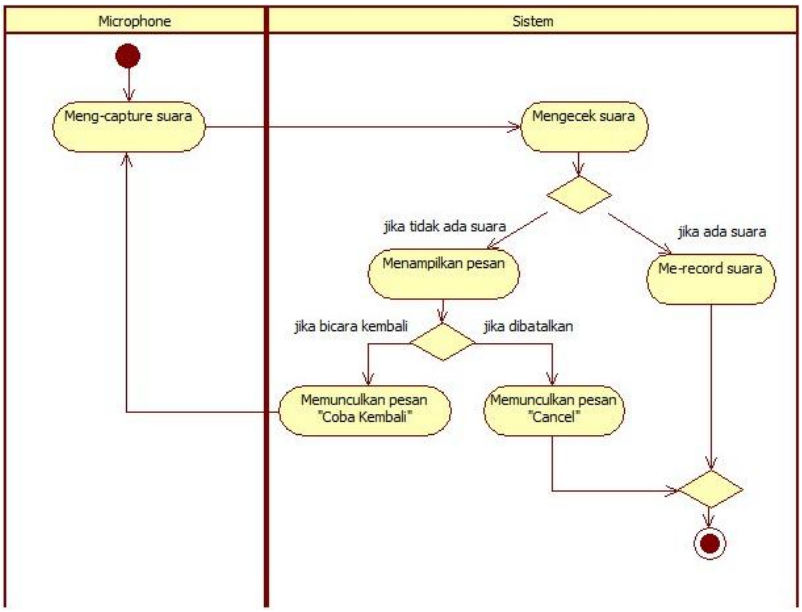
Identifikasi			
Nomor Use Case	4		
Nama Use Case	Penyajian *List* Doa		
Deskripsi	Proses sistem menyajikan *list* doa		
Trigger	Suara telah dicocokkan		
Aktor	Pengguna		
Skenario Normal			
No	Aksi Aktor	No	Reaksi Sistem
1	Melihat hasil pencocokan suara dengan doa berupa list doa		
2	Menekan hasil pencocokan doa		
		3	Menampilkan doa

2. Activity Diagram

Berikut adalah gambar *activity diagram* untuk *use case* yang telah didefinisikan sebelumnya:

1. *Activity Diagram capture* suara

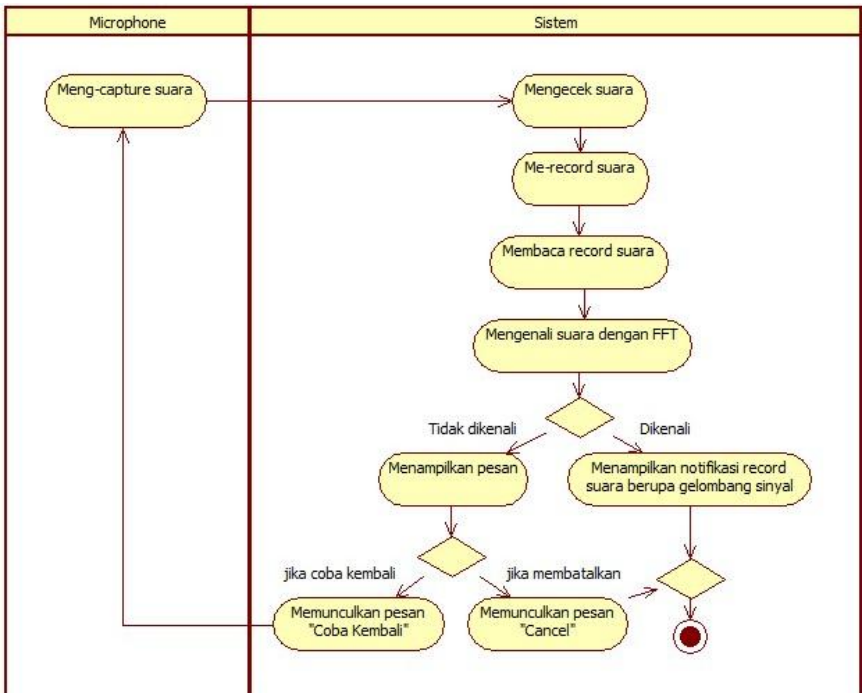
   Activity diagram capture suara pada aplikasi ini dapat dilihat pada gambar 3.18 berikut:



Gambar 3.18 *Activity Diagram capture* suara

2. *Activity Diagram* pengenalan suara dengan FFT

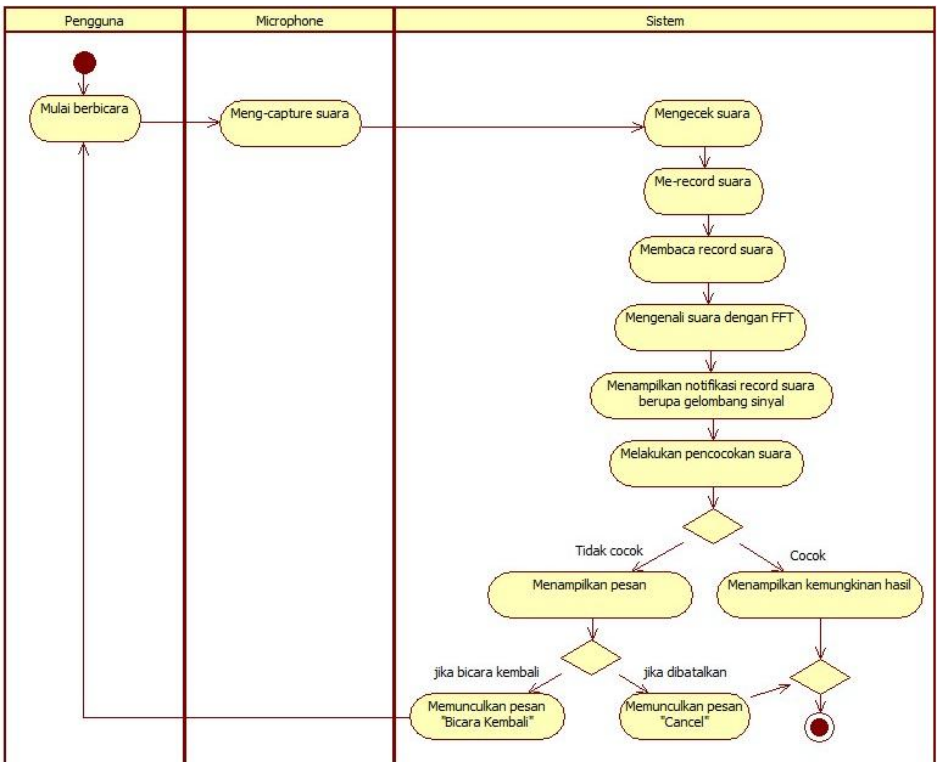
   Activity diagram pengenalan suara dengan FFT pada aplikasi ini dapat dilihat pada gambar 3.19 berikut:



Gambar 3.19 *Activity Diagram* pengenalan suara dengan FFT

3. *Activity Diagram* pencocokan suara dengan HMM

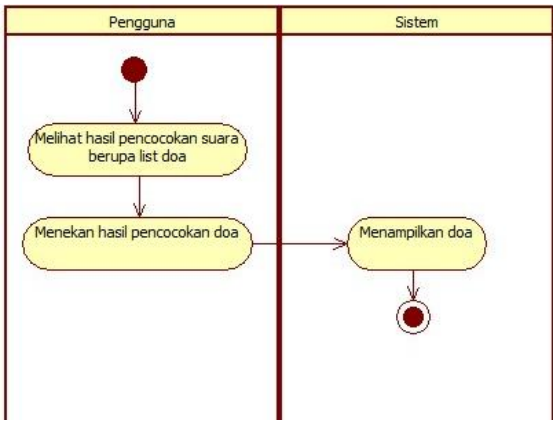
   Activity diagram pencocokan suara dengan HMM pada aplikasi ini dapat dilihat pada gambar 3.20 berikut:



Gambar 3.20 *Activity Diagram* pencocokan suara dengan HMM

4. *Activity Diagram* penyajian *list* doa

   Activity diagram penyajian *list* doa pada aplikasi ini dapat dilihat pada gambar 3.21 berikut:



Gambar 3.21 *Activity Diagram* penyajian *list* doa

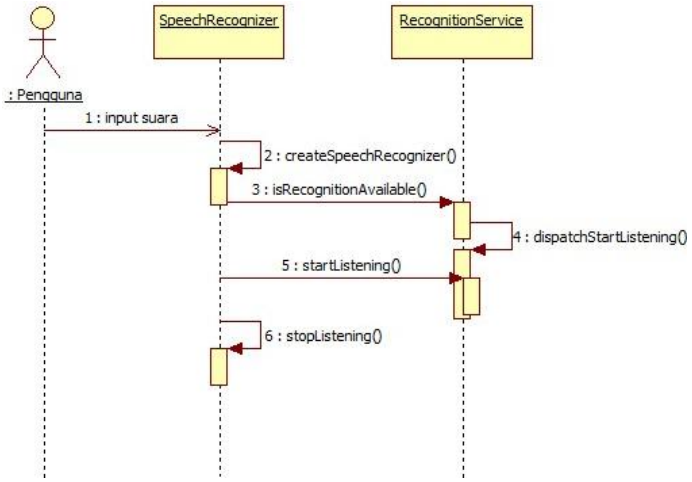
3. *Sequence Diagram*

Diagram *sequence* atau *sequence diagram* menggambarkan kelakuan objek pada *use case* dengan mendeskripsikan waktu hidup objek dan *message* yang dikirimkan dan diterima antar objek.

Berikut adalah gambar *sequence diagram* pada sistem perintah suara:

1. *Sequence Diagram capture* suara

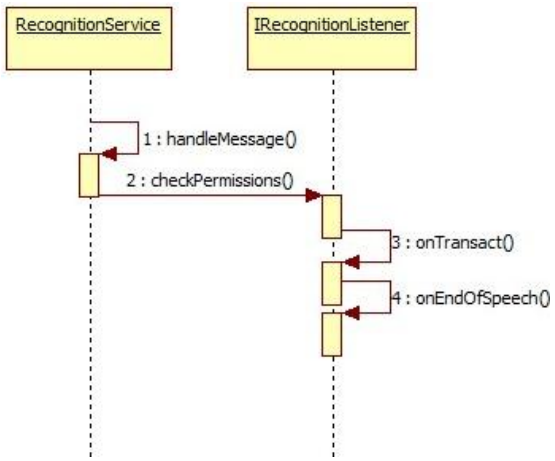
   Sequence diagram capture suara pada aplikasi ini dapat dilihat pada Gambar 3.22 berikut:



Gambar 3.22 *Sequence Diagram capture* suara

2. *Sequence Diagram* pengenalan suara dengan FFT

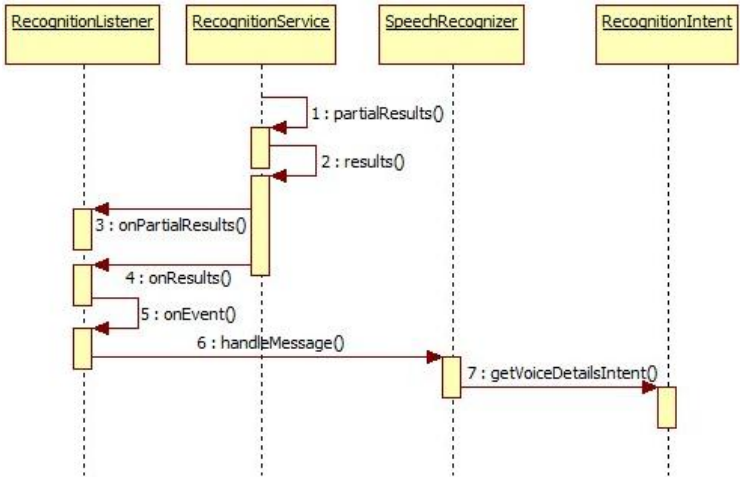
   Sequence diagram pengenalan suara dengan FFT pada aplikasi ini dapat dilihat pada Gambar 3.23 berikut:



Gambar 3.23 *Sequence Diagram* pengenalan suara dengan FFT

3. *Sequence Diagram* pencocokan suara dengan HMM

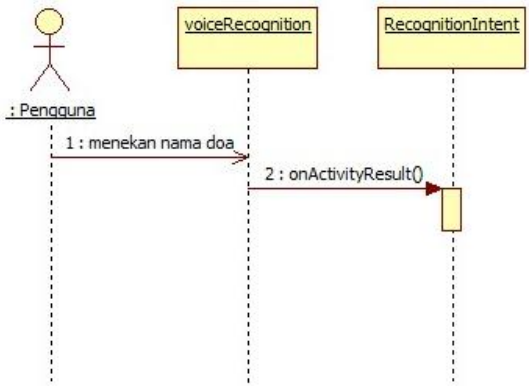
   Sequence diagram pencocokan suara dengan HMM pada aplikasi ini dapat dilihat pada Gambar 3.24 berikut:



Gambar 3.24 *Sequence Diagram* pencocokan suara dengan HMM

4. *Sequence Diagram* penyajian *list* doa

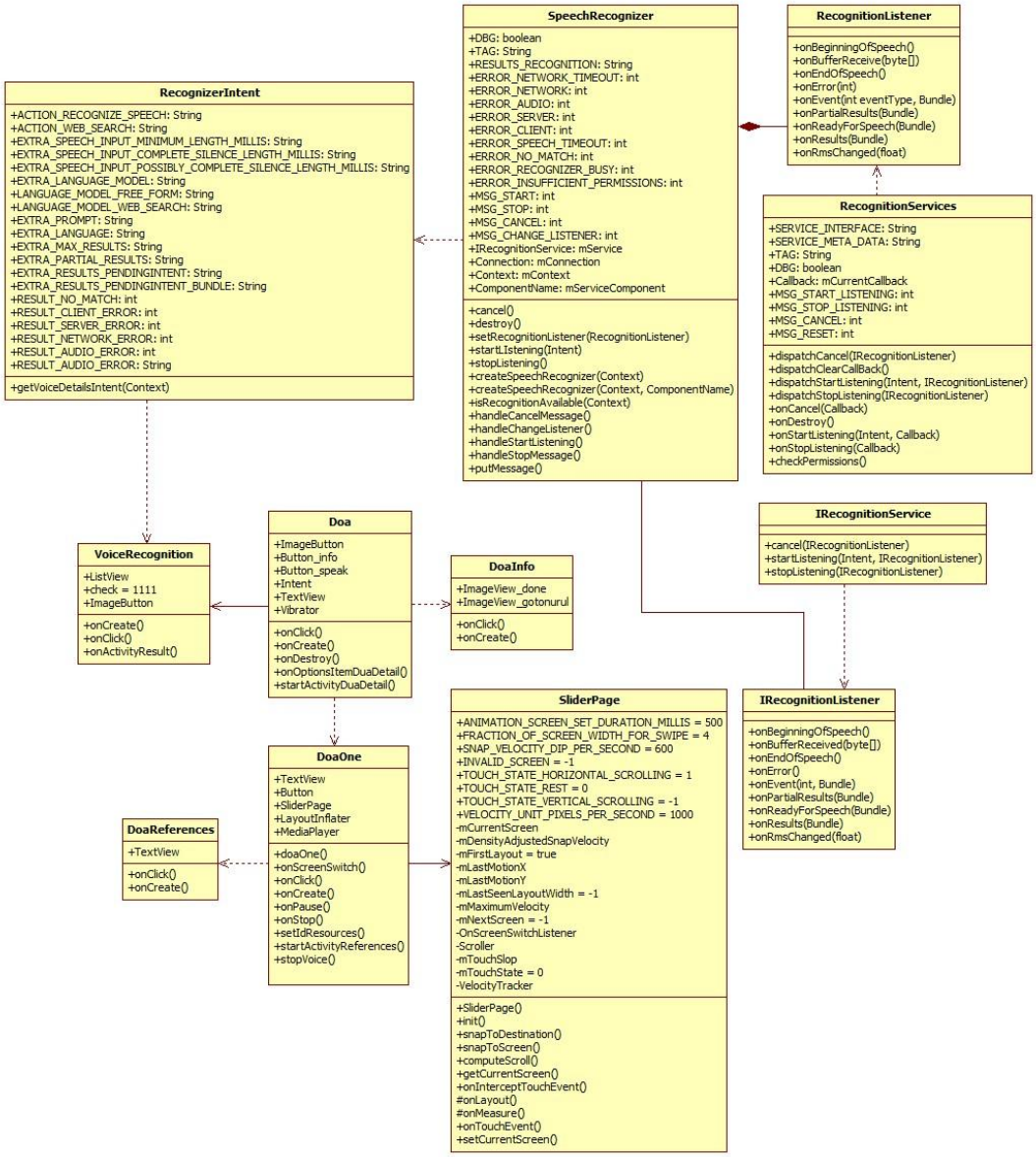
   Sequence diagram penyajian *list* doa pada aplikasi ini dapat dilihat pada Gambar 3.25 berikut:



Gambar 3.25 *Sequence Diagram* penyajian *list* doa

4. Class Diagram

Diagram kelas atau *class diagram* menggambarkan struktur sistem dari segi pendefinisian kelas-kelas yang dibuat untuk membangun sistem. Berikut ini adalah perancangan struktur sistem yang terdapat pada sistem perintah suara dengan implementasi pada aplikasi doa sehari-hari yang dapat dilihat pada Gambar 3.26 berikut:



Gambar 3.26 *Class Diagram* Sistem Perintah Suara

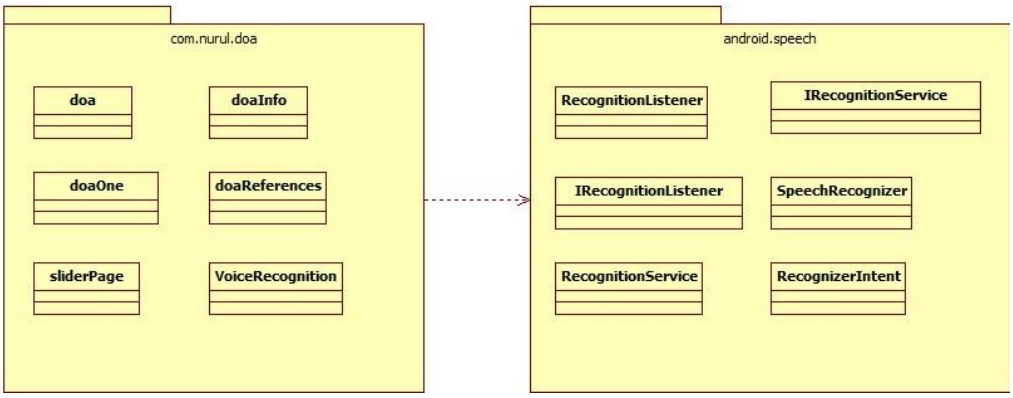
Tabel 3.12 Keterangan *Class Diagram*

Nama Kelas	Keterangan
Doa	Merupakan kelas main yang juga merangkap sebagai kelas yang menangani tampilan.
DoaInfo	Merupakan kelas proses untuk mendefinisikan info aplikasi.
DoaOne	Merupakan kelas kontrol untuk memanggil layout doa yang akan ditampilkan.

DoaReferences	Merupakan kelas proses untuk mendefinisikan referensi doa.
SliderPage	Merupakan kelas kontrol untuk menangani animasi.
VoiceRecognition	Merupakan kelas kontrol untuk menangani perintah suara.
RecognizerIntent	Merupakan konstanta untuk mendukung pengenalan suara.
RecognizerListener	Merupakan kelas *interface* yang digunakan untuk menerima notifikasi dari SpeechRecognizer ketika proses recognition terjadi.
SpeechRecognizer	Kelas ini menyediakan akses ke layanan pengenalan suara.
RecognitionService	Kelas ini menyediakan kelas dasar untuk implementasi layanan pengenalan.
IRecognitionService	Merupakan kelas *interface*
IRecognitionListener	Merupakan kelas *interface*

5. Package Diagram

Package diagram menyediakan cara mengumpulkan elemen-elemen yang saling terkait dengan diagram UML. Hampir semua diagram dalam UML bisa dikelompokkan menggunakan *package diagram*. *Package diagram* aplikasi ini dapat dilihat pada Gambar 2.27 berikut:

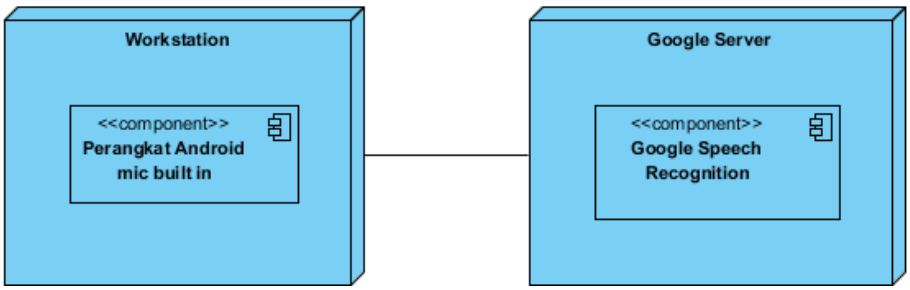


Gambar 3.27 *Package Diagram* Sistem Perintah Suara

Gambar 3.27 menggambarkan *diagram package* untuk sistem perintah suara pada aplikasi doa sehari-hari. *Diagram package* di atas terdiri dari *package* com.nurul.doa, yaitu *package* untuk semua *class* dari aplikasi doa dan *package* android.speech, yaitu *package* untuk semua *class speech* yang digunakan untuk jalannya sistem perintah suara. Adanya *Association* diantara kedua *package* karena *package* com.nurul.doa dan *package* android.speech harus berinteraksi untuk bisa menjalankan perintah suara.

6. Deployment Diagram

Deployment Diagram menunjukkan konfigurasi komponen dalam proses eksekusi aplikasi. *Deployment Diagram* untuk sistem perintah suara dapat dilihat pada Gambar 3.28 berikut:



Gambar 3.28 *Deployment Diagram* Sistem Perintah Suara

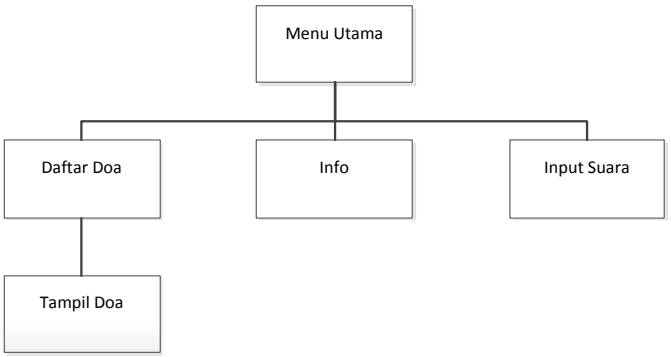
Sistem perintah suara pada aplikasi doa sehari-hari ini hanya menggunakan perangkat keras berupa smartphone berbasis sistem operasi Android yang diletakan pada sebuah node kemudian di dalam node terdapat sebuah *component* yang berupa sistem operasi itu sendiri yaitu Android. Terdapat sebuah *Association* yang menghubungkan antara node satu dan node dua yang berisi berupa jaringan internet. Jadi dari perangkat android ini, akan terkoneksi dengan *server google*

sebagai *node*-nya dan di dalamnya terdapat *component* berupa *speech recognition* yang digunakan. Apabila tidak tersedia internet maka perintah suara tidak dapat digunakan.

3.2 Perancangan Sistem

3.2.1 Perancangan Struktur Menu

Struktur menu adalah bentuk umum dari suatu rancangan program untuk memudahkan pengguna dalam menjalankan suatu aplikasi. Pada perancangan ini dibuat menu yang dapat mengintegrasikan seluruh data dalam suatu sistem dan disertai dengan instruksi yang ada pada pilihan menu tersebut. Adapun struktur menu tersebut dapat dilihat pada gambar 3.29 berikut:



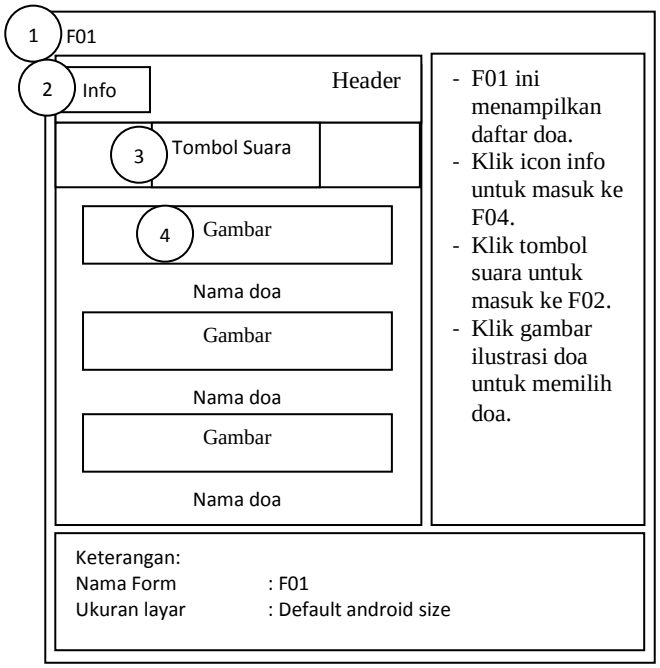
Gambar 3.29 Struktur Menu Aplikasi

3.2.2 Perancangan Antarmuka

Perancangan antarmuka bertujuan untuk memberikan gambaran tentang aplikasi yang akan dibangun, sehingga akan mempermudah dalam mengimplementasikan aplikasi serta akan memudahkan pembuatan aplikasi. Berikut ini merupakan perancangan antarmuka sistem perintah suara pada aplikasi doa sehari-hari:

1. Perancangan Antarmuka Utama

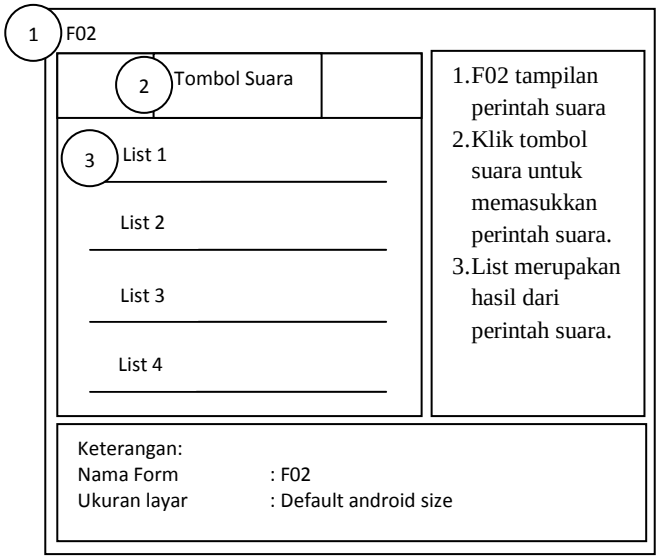
Perancangan antar muka utama pada aplikasi ini dapat dilihat pada Gambar 3.30 berikut:



Gambar 3.30 Perancangan Tampilan Daftar Doa

2. Perancangan Antarmuka Perintah Suara

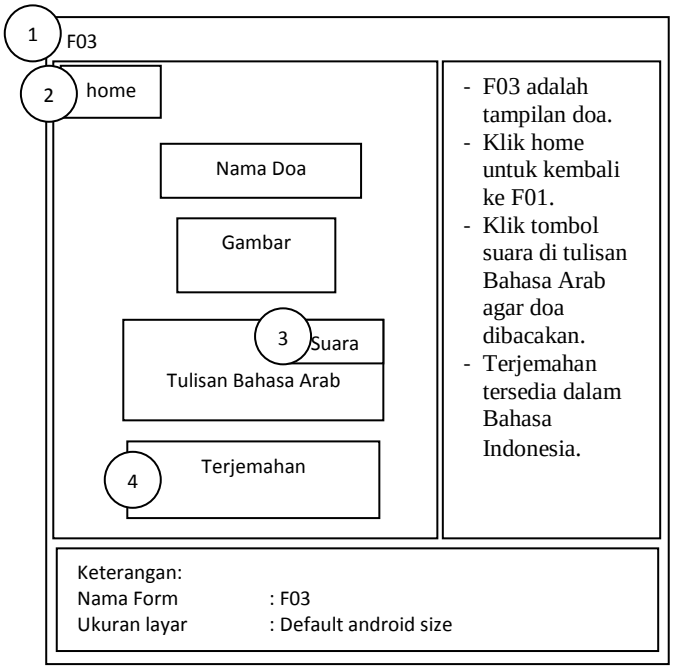
Perancangan antar muka perintah suara pada aplikasi ini dapat dilihat pada Gambar 3.31 berikut:



Gambar 3.31 Perancangan Tampilan Perintah Suara

3. Perancangan Antarmuka Doa

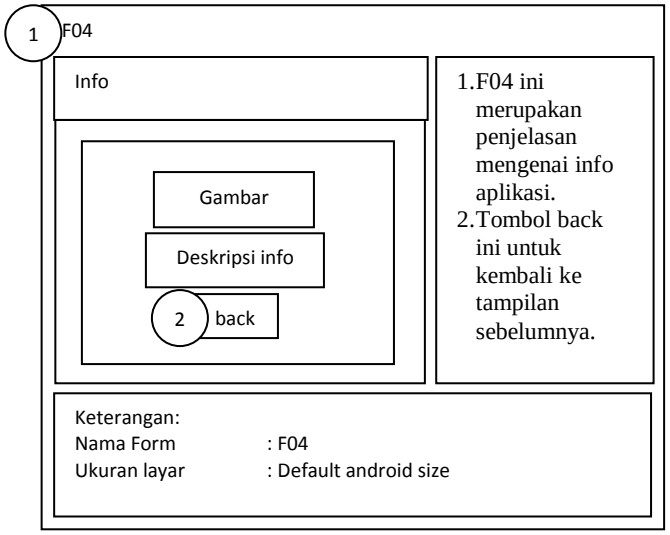
Perancangan antar muka doa pada aplikasi ini dapat dilihat pada Gambar 3.32 berikut:



Gambar 3.32 Perancangan Tampilan Doa

4. Perancangan Antarmuka Info

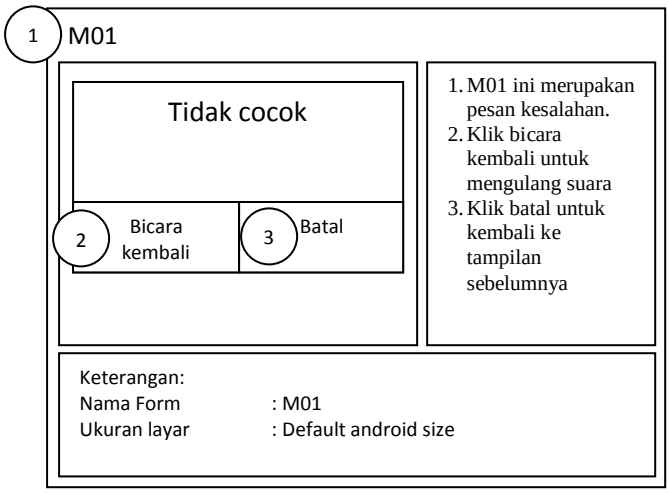
Perancangan antar muka info pada aplikasi ini dapat dilihat pada Gambar 3.33 berikut:



Gambar 3.33 Perancangan Tampilan Info

5. Perancangan Antarmuka Pesan

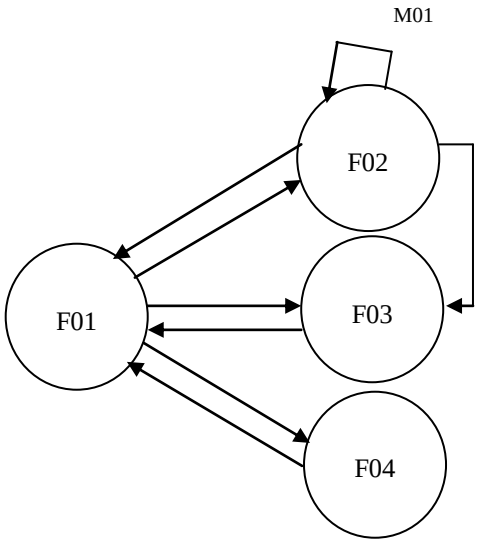
Perancangan antar muka pesan pada aplikasi ini dapat dilihat pada Gambar 3.34 berikut:



Gambar 3.34 Perancangan Pesan Jika Perintah Suara Tidak Cocok

3.2.3 Jaringan Semantik

Jaringan semantik adalah gambaran tentang relasi dari masing- masing halaman yang bisa diakses oleh seorang pengguna. Jaringan semantik ini, digunakan sebagai panduan penggunaan menu-menu dalam sebuah aplikasi. Terdapat tiga tampilan pada aplikasi ini, yaitu F01 untuk tampilan utama aplikasi berupa daftar doa, F02 merupakan tampilan doa, sedangkan F03 merupakan tampilan untuk info aplikasi. Perancangan semantik ini dapat dilihat sebagai berikut:



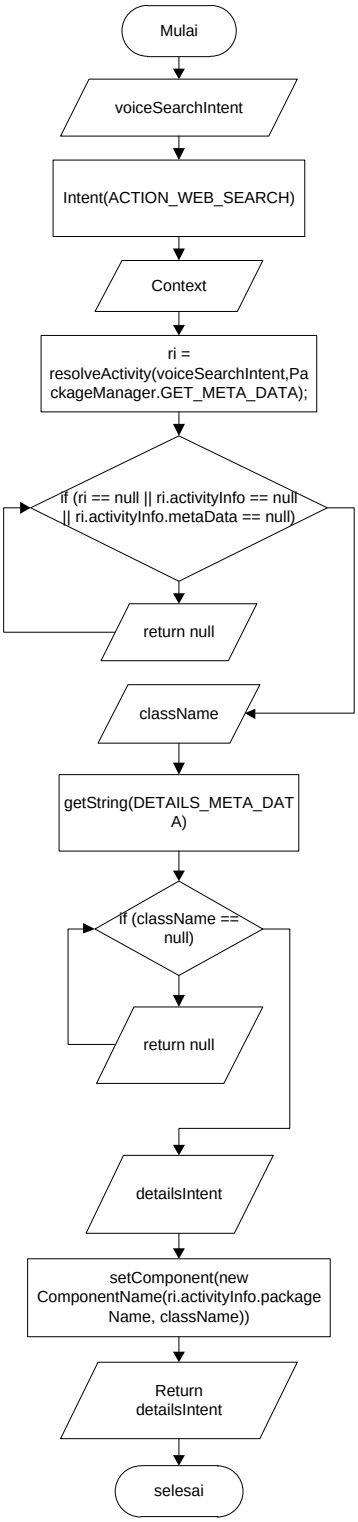
Gambar 3.35 Jaringan Semantik

3.2.4 Perancangan *Method*

Perancangan *method* berfungsi untuk mendeskripsikan *method-method* yang berada di dalam aplikasi dalam bentuk *flowchart*. Adapun penjelasan *method-method* yang terdapat dalam sistem perintah suara yang akan dibangun adalah sebagai berikut:

3.2.4.1 *Method* Pengenalan Suara

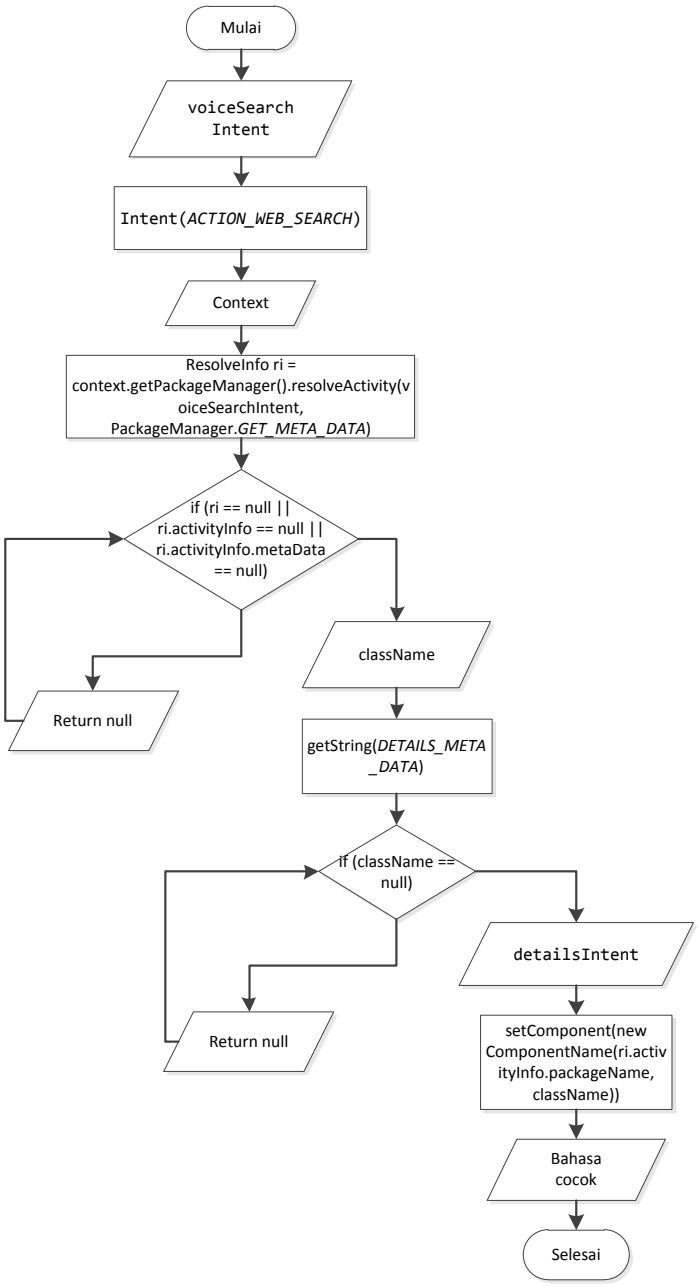
Method ini bekerja ketika sistem mengenali suara yang telah di-*capture* oleh *microphone*. Untuk lebih jelasnya dapat dilihat dari gambar 3.36 berikut:



Gambar 3.36 *Flowchart* Pengenalan Suara

3.2.4.2 *Method* Pencocokan Suara

Method ini bekerja ketika sistem telah meng-*capture* suara oleh *microphone* dan suara tersebut telah dikenali. Untuk lebih jelasnya dapat dilihat dari gambar 3.37 berikut:



Gambar 3.37 *Flowchart* Pencocokan Suara