BAB 3
ANALISIS DAN PERANCANGAN SISTEM

3.1 Analisis Masalah

Analisis masalah yang diangkat dalam penelitian ini adalah suatu gambaran masalah dalam menghasilkan sistem yang mampu menampilkan informasi akor dari input akor yang berurutan dalam suatu *file*. Sebelumnya telah dilakukan penelitian pengenalan akor yang melakukan pengenalan terhadap klasifikasi pengenalan akor dengan ekstraksi ciri MFCC dengan hasil percobaan menunjukkan akurasi maksimum yang diperoleh adalah 83,65% [9].

Sebagai klasifikator digunakan *Fuzzy c-means clustering* . *Fuzzy c-means clustering* melakukan pengelompokkan terhadap hasil dari *feature extraction* menggunakan *Fast Fourier Transform.* Hasil dari pelatihan *fuzzy c-means clustering* akan disimpan sebagai model *pattern* pada database yang akan digunakan untuk proses pencocokan akor. Proses pencocokan akor dilakukan dengan pengukuran jarak terdekat menggunakan *euclidean distance* untuk mengetahui seberapa besar kemiripan akor data uji dengan model *pattern.*

Berdasarkan uraian diatas, pada penelitian ini akan diterapkan *Fuzzy c-means clustering* untuk mendeteksi akor yang berurutan dalam satu *file*.

3.2 Analisis Proses

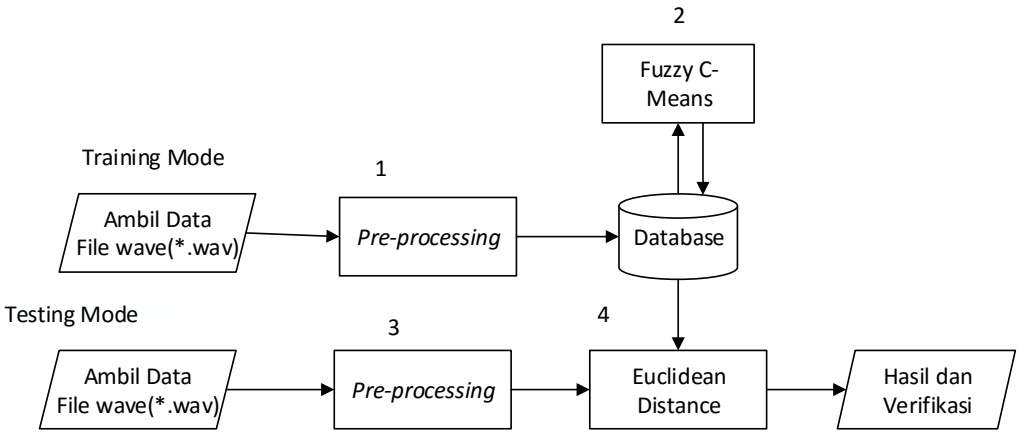
Proses dari sistem pengenalan akor menggunakan metode *Fuzzy c-means clustering* adalah proses pengambilan atau penentuan *file (file input)*, proses latih *(training mode)* dan proses pengujian *(testing mode)*.

File Input merupakan bagian yang bertugas menentukan *file* audio yang akan diekstraksi. Alat yang dibutuhkan dalam proses ini adalah *file* yang berformat (*.wav).

Training Mode adalah bagian yang bertugas mempelajari dan memodelkan akor. Pembelajaran terhadap akor dilakukan dengan membuat model dari akor kemudian menyimpannya ke dalam *database*.

Testing Mode merupakan bagian yang bertugas mengenali akor. Pengenalan akor dilakukan dengan melakukan pencocokan data uji dengan data model *pattern* pada *database*.

Proses pengenalan akor menggunakan *Fuzzy c-means clustering* dapat dilihat pada Gambar 3.1 Berikut:



Gambar 3. 1 Alur Proses Pengenalan Akor

Gambar 3.1 merupakan alur proses pengenalan akor, berikut penjelasannya :

1. Pada *file* audio (.wav) *Training mode* akan dilakukan *pre-processing.*
   Dalam *preprocessing* terdapat beberapat tahapan yaitu normalisasi, *frame blocking, windowing, powering, thresholding*, dan FFT. Kemudian data yang berupa frekuensi disimpan di *database.*
2. Keluaran *fuzzy c- means* berupa titik centroid yang kemudian dimasukkan ke *database*
3. Pada *file* audio (.wav) data uji akan dilakukan *pre-processing.*
   Dalam *preprocessing* terdapat beberapat tahapan yang sama dengan data latih yaitu normalisasi, *frame blocking*, *windowing*, *powering*, *thresholding*, dan FFT. Kemudian data yang berupa frekuensi akan di proses di *Euclidean distance.*

4. Keluaran *Euclidean distance* adalah jarak perbanding data latih dan data uji jarak terdekat akan diverifikasi sebagai akor yang di prediksi.

3.3 Analisis Data Masukan

Analisis data yaitu analisis data pada *preprocessing* untuk klasifikasi akor menggunakan *fuzzy c-means* . Pada proses *preprocessing* data masukan berbentuk berkas audio (*.wav). Data masukan berkas audio (*.wav) yang bersifat sinyal analog akan dikonversikan terlebih dahulu kedalam bentuk sinyal *digital* melalui proses konversi analog menjadi *digital* dan akuisisi data. Pada proses pencocokan akor yang digunakan merupakan data hasil fitur dari hasil akhir *preprocessing* yang berbentuk frekuensi.

3.4 Analisis Metode

Metode yang diterapkan pada pengenalan akor meliputi pre-processing dan *fuzzy c-means clustering*. *Pre-processing* terdiri dari konversi Analog menjadi *Digital* (ADC), akuisisi data, normalisasi, *frame blocking*, *windowing*, *powering*, *thresholding,* serta *Fast fourier transform* . Berikut analisis dari masing- masing metode yang digunakan:

3.4.1 *Pre-processing*

Sinyal suara yang kan diproses bersifat analog sehiingga jika akan dilakukan pengolahan secara *digital,* sinyal suara tersebut harus dikonversi menjadi sinyal *digital*, setelah menjadi sinyal digital yang berupa amplitudo maka proses dilanjutkan dengan pengolahan data sinyal dengan normalisasi, *frame blocking*, *windowing, powering*, *thresholding*, *Fast fourier transform* serta *peak detection* berikut penjelasan tiap prosesnya:

3.4.1.1 Konversi Analog Menjadi *Digital* (ADC)

Sinyal suara suara yang akan diproses 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 dinamakan *analog to digital conversion*. Konsep kerja ADC terdiri dari tiga proses yaitu :

1. Sampling

Proses sampling adalah suatu proses untuk mengambil data *signal continue* untuk setiap periode tertentu. Data berupa sinyal diambil dengan cara merekam keyboard yang terhubung dengan komputer, data akan disimpan menggunakan format berkas file audio wav. Frekuensi sampling yang digunakan pada proses perekaman adalah 44100Hz. Durasi perekaman dilakukan dengan waktu yang pendek yaitu 1 detik.

2. Kuantisasi

Proses kuantisasi adalah proses untuk membulatkan nilai data ke dalam bilangan-bilangan tertentu yang telah ditentukan terlebih dahulu. Proses ini melakukan pengkonversian nilai analog ke dalam suatu nilai diskrit. Selama proses kuantisasi, ADC menkonversi setiap nilai analog ke dalam bentuk diskrit.

3. Pengkodean

Proses pengkodean adalah proses pemberian kode untuk tiap-tiap data signal yang telah terkuantisasi berdasarkan level yang ditempati. Pada proses ini, tiap nilai diskrit yang telah didapat, direpresentasikan dengan angka binary n-bit. Cara konversinya adalah dengan membagi bilangan desimal dengan bilangan biner dengan memperhatikan hasil sisa pembagian.

Sebagai contoh pada perekaman akor A mayor yang dilakukan selama 1 detik pada frekuensi 44100Hz, dengan format berkas audio .wav menggunakan 16 bit dan *channel mono*. Diambil data sinyal amplitudo sebanyak 30 sinyal dari sekitar 87000 sinyal. Sampel 30 data sudah cukup untuk menggambarkan proses yang akan dilalui pada proses selanjutnya. Data tersebut adalah 26, 147, 365, 601, 789, 941, 1126, 1382, 1690, 2019, 2366, 2744, 3128, 3442, 3661, 3848, 4088, 4417, 4771, 5069, 5299, 5486, 5666, 5834, 5929, 5973, 6064, 6189, 6253, 6170.

3.4.1.2 Akuisisi Data

Data diperoleh dengan cara merekam suara akor dari keyboard yang terhubung dengan komputer. Data latih berjumlah 120 dengan 24 akor dan tiap-tiap akor direkam sebanyak 5 kali. Perekaman suara menggunakan frekuensi 44100 Hz dan menggunakan format berkas audio (.wav). Suara dengan format (.wav) ini bisa menggunakan 16 bits/sample dan 1 untuk channel mono.

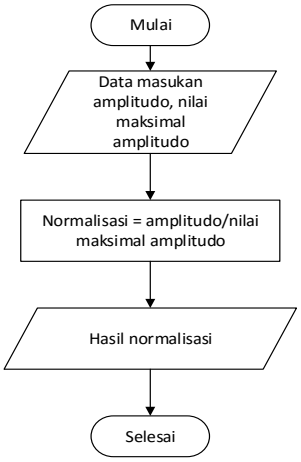
Dalam analisis ini digunakan contoh durasi rekaman akor A mayor. Durasi suara dengan jangka waktu yang pendek akan lebih mudah untuk diambil perbedaan fiturnya. Nilai maksimal amplitudo adalah 16000 yang didapat dari hasil pembacaan file yang dan total sinyal amplitudo berjumlah 87000. Sebagai contoh diambil data masukan berupa 30 angka dari amplitudo seperti yang dapat dilihat pada tabel 3.1 berikut.

Tabel 3. 1 Data Masukan

No	Data Masukan	No	Data Masukan	No	Data Masukan
0	26	10	2366	20	5299
1	147	11	2744	21	5486
2	365	12	3128	22	5666
3	601	13	3442	23	5834
4	789	14	3661	24	5929
5	941	15	3848	25	5973
6	1126	16	4088	26	6064
7	1382	17	4417	27	6189
8	1690	18	4771	28	6253
9	2019	19	5069	29	6170

3.4.1.3 Normalisasi

Pada proses normalisasi ini digunakan untuk menyeragamkan nilai amplitudo dari sinyal sampling. Tujuan dari normalisasi yaitu agar pengaruh dari kuat atau lemahnya suara yang terdapat pada *file wav* dapat diminimalkan dalam proses normalisasi. Berikut diagram alir normalisasi



Gambar 3. 2 Flowchart Normalisasi

Hasil dari proses normalisasi ini dirancang agar batasan nilai puncak maksimum dari data akor tersebut diantara nilai 1 atau -1. Data masukan normalisasi berasal dari akuisisi data. Contoh proses normalisasi berikut ini menggunakan file A mayor yang menunjukan akor A. Amplitudo didapat dari hasil pembacaan file dan nilai maksimal didapat dari sinyal tertinggi. Persamaan pada normalisasi terdapat pada persamaan 2.3 sebagai berikut.

$$X_{normalisasi} = \frac{X_{masukan}}{\max(abs(X_{masukan}))}$$

Keterangan :

$X_{normalisasi}$ = hasil data sinyal normalisasi

$X_{masukan}$ = nilai data masukan (Amplitudo)

$\max(abs(X_{masukan}))$ = nilai maksimal masukan (16000)

Pada amplitudo ke-0, maka :

$$X_{normalisasi} = \frac{X_{masukan}}{\max(abs(X_{masukan}))} = \frac{26}{16000} = 0.001625$$

Hitung dengan cara yang sama pada N lainnya. Sehingga didapat hasil pada tabel 3.3 berikut.

Tabel 3. 2 Data Masukan Normalisasi

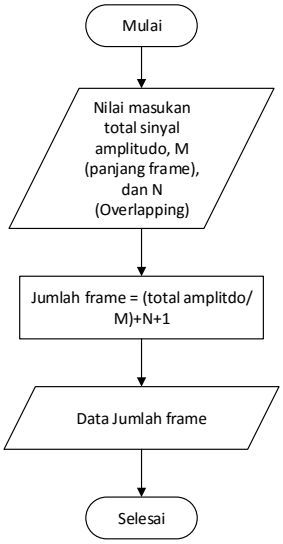
No	Data Masukan	No	Data Masukan	No	Data Masukan
0	26	10	2366	20	5299
1	147	11	2744	21	5486
2	365	12	3128	22	5666
3	601	13	3442	23	5834
4	789	14	3661	24	5929
5	941	15	3848	25	5973
6	1126	16	4088	26	6064
7	1382	17	4417	27	6189
8	1690	18	4771	28	6253
9	2019	19	5069	29	6170

Tabel 3. 3 Data Keluaran Normalisasi

No	Normalisasi	No	Normalisasi	No	Normalisasi
0	0.001625	10	0.147875	20	0.3311875
1	0.0091875	11	0.1715	21	0.342875
2	0.0228125	12	0.1955	22	0.354125
3	0.0375625	13	0.215125	23	0.364625
4	0.0493125	14	0.2288125	24	0.3705625
5	0.0588125	15	0.2405	25	0.3733125
6	0.070375	16	0.2555	26	0.379
7	0.086375	17	0.2760625	27	0.3868125
8	0.105625	18	0.2981875	28	0.3908125
9	0.1261875	19	0.3168125	29	0.385625

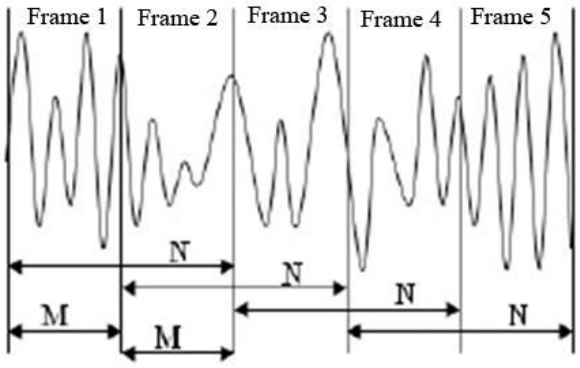
3.4.1.4 Frame Blocking

Data hasil perekaman akor yang merupakan sinyal analog yang berada dalam domain waktu yang bersifat *variant time,* yaitu suatu fungsi yang bergantung waktu. Berikut diagram alir *frame blocking.*



Gambar 3. 3 *Flowchart Frame Blocking*

Proses *frame blocking* dilakukan dengan cara memotong-motong sinyal dalam slot-slot waktu tertentu agar dapat dianggap *invariant* setiap potongan tersebut disebut *frame*. Total sinyal amplitudo didapat dari jumlah sinyal amplitudo pada pembacaan *file*. Nilai M didapat dari panjang frame 1 detik. Lebih jelas dapat dilihat pada gambar 3.4 berikut.



Gambar 3. 4 *Frame Blocking*

Persamaan untuk menghitung *frame blocking* dapat dilihat dalam persamaan 2.4 berikut.

Jumlah *frame* = ((Total sinyal amplitudo/M)+N+1)

Keterangan :

Jumlah *frame* = Total *frame* berdasarkan pembagian panjang *frame* dengan *Overlapping*

M = Panjang *frame*

N = *Overlapping* (M/2)

Total sinyal amplitudo dari pembacaan A.wav berjumlah 87000, untuk memproses sinyal dibutuhkan pemotongan-pemotongan sinyal dengan panjang satu *frame* nya 1000 dan nilai *overlapping* 500.

Ditetapkan :

M = 1000

N = 500

Hitung Jumlah *Frame* :

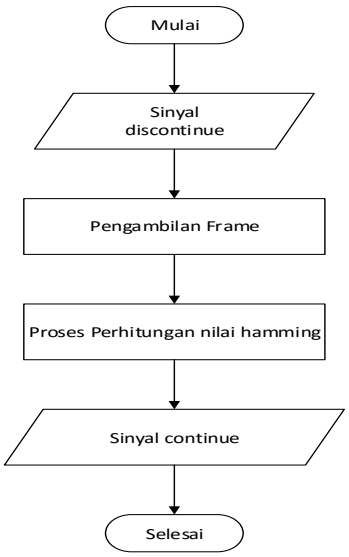
$$\text{Jumlah } frame = ((\text{ Total sinyal amplitudo/M}) + \text{N} + 1$$

$$= (\frac{87000}{1000}) + 500 + 1 = 588$$

Jumlah *frame* adalah 588 *frame* dimana tiap *frame* memiliki 1000 nilai.

3.4.1.5 Windowing

Setelah melakukan proses *frame blocking*, proses selanjutnya adalah *windowing*. Berikut diagram alir *windowing*:



Gambar 3. 5 *Flowchart Windowing*

Fungsi pada proses *windowing* pada penelitian ini menggunakan *Hamming Window* persamaan 2.6 sebagai berikut:

$$w(n) = 0.54 - 0.46\ cos\frac{2\pi n}{M-1}$$

Keterangan :

n = 0,1,2,…,999

M (Panjang *Frame*) = 1000

$$w(n) = 0.54 - 0.46\ cos\frac{2\pi n}{M-1}$$

$$w(0) = 0{,}54 - 0{,}46\cos\left(\frac{2x3{,}14x0}{1000-1}\right) = 0{,}08$$

Hitung dengan cara yang sama untuk n ke-1 hingga 999. Maka didapat hasil dari hamming window pada tabel 3.4 Berikut.

Tabel 3. 4 *Window Hamming*

No	Window Hamming
0	0.08
1	0.080009098
2	0.080036392
3	0.080081882

No	Window Hamming
4	0.080145564
5	0.080227437
6	0.080327497
7	0.080445741
8	0.080582163
9	0.080736759
10	0.080909522
….	…..
995	0.08014556
996	0.08008188
997	0.08003639
998	0.0800091
999	0. 08

Selanjutnya merepresentasikan fungsi *window* terhadap sinyal amplitudo yang telah di normalisasi yang terdapat pada tabel 3.3.

$$x(n) = x_i(n)w(n) \tag{3.3}$$

Keterangan :

n (data ke-) = 0,1,2,...,999

i (frame ke-) = 0,1,2,..,587

x(n) = nilai sampel sinyal hasil *windowing*

$x_i(n)$ = nilai sampel dari *frame signal* ke i

$w(n)$ = hasil *window hamming*

$$x(n) = x_i(n)w(n)$$

$$x(0) = x_0(0)w(0)$$

$$x(0) = 0.001625 \text{ x } 0.08 = 0.00013$$

Hitung dengan cara yang sama untuk data pada frame ke-0 dari n = 1 sampai dengan n = 999. Maka didapat hasil sampel sinyal windowing frame ke-0 pada tabel 3.5 Berikut.

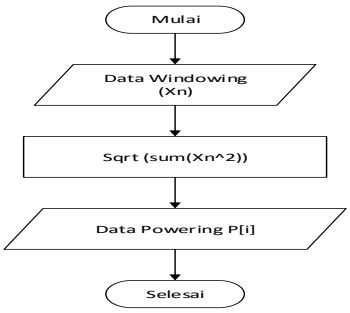
Tabel 3. 5 Data *Windowing*

No	Windowing
0	0.00013
1	0.000735084
2	0.00182583
3	0.003008076
4	0.003952178
5	0.004718376
6	0.005653048
7	0.006948501
8	0.008511491
9	0.01018797
10	0.011964496
…	….
995	-0.012147062
996	-0.013068362
997	-0.015036837
998	-0.017066941
999	-0.01808

Perhitungan yang sama dilakukan untuk frame ke-1 atau i ke-1 hingga frame ke-587 atau i ke-587.

3.4.1.6 Powering

Setelah melakukan proses *frame blocking* yang menjadi acuan dalam menentukan *powering* tiap-tiap *frame* nya, dan *windowing* yang menjadi input dari proses *powering*. Proses selanjutnya adalah operasi *powering* yaitu melakukan pengkuadratan pada sinyal untuk mendapatkan peak-peak dominan sehingga diketahui letak awal dan akhir yang berada disuatu *frame.* Berikut *flowchart powering* :



Gambar 3. 6 Flowchart Powering

Dari 0 sampai 587 *frame* yang didapat dari hasil *windowing*, tiap-tiap *frame* yang mempunyai 1000 nilai akan dilakukan *powering*. *Powering* mempunyai persamaan 2.7 sebagai berikut.

$$P(i) = \sum_{m=0}^{m=999} \left| {X_i}^2 \right|$$

Keterangan :

P = *power* sinyal (dB)

Xi = *Frame* ke-i (dB)

i = 0,1,2,…,587

m = 0,1,2,…,999

Pada file A mayor, maka didapat nilai *powering* pada *frame* ke-0 yang mempunyai 1000 data sebagai berikut.

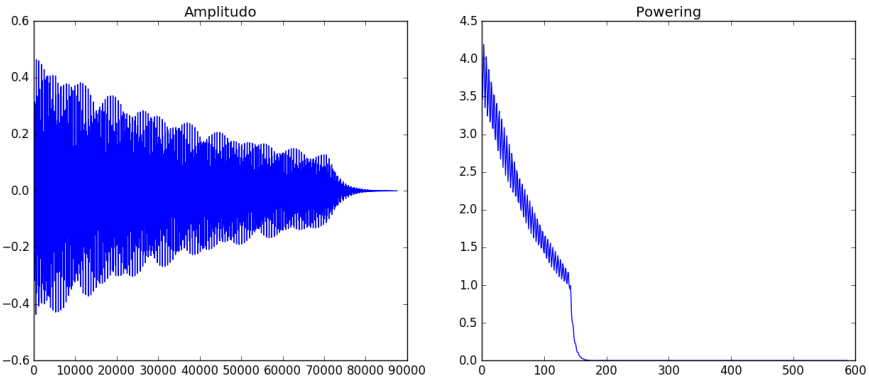
$$P(0) = \sum_{m=0}^{m=999} \left| {X_i}^2 \right| = \sum_{m=0}^{m=999} \left| {X_0}^2 \right| = 3.94346$$

Hitung dengan cara yang sama hingga powering 587 frame terpenuhi. Hasil dari powering A mayor dapat dilihat pada tabel 3.6 berikut.

Tabel 3. 6 *Powering*

No	P(n)
0	3.94346
1	3.46394
2	3.80224
3	4.19073
4	3.85932
5	3.35379
6	3.59925
…..	….
140	1.007
141	0.952
….	…..
173	0.002
..	….
587	0.00

Agar lebih jelasnya dapat dilihat pada gambar 3.7 Berikut



Gambar 3. 7 Powering A mayor

Kemudian ditentukan nilai standar deviasi dari persamaan 2.8 berikut.

$$Standar\ Deviasi = \sqrt{\frac{n\sum P_i{}^2 - (\sum P_i)^2}{n(n-1)}}$$

$$= \sqrt{\frac{(588 \times 781426) - 100100.51)}{588(588-1)}} = \sqrt{\frac{359377.79}{345156}} = 1.02$$

Keterangan :

P_i = Nilai *Frame* ke-i,

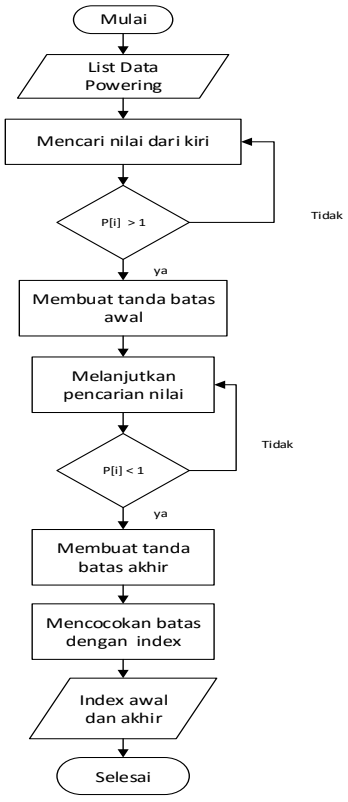
n = jumlah *frame*

i = *frame* ke-0 hingga 587

Untuk menghindari bias maka nilai standar deviasi dibulatkan menjadi yaitu 1. Bias yang dimaksud adalah nilai dikategorikan sebagai noise karena berada pada nilai mendekati nol.

3.4.1.7 Thresholding

Setelah melakukan proses *powering* maka dilanjutkan dengan *thresholding*. *Thresholding* dilakukan sebagai acuan untuk memisahkan *voice. Thresholding* adalah membuat batas dan index masing-masing awal dan akhir yang didapat dari nilai hasil powering. Berikut *flowchart* dari *thresholding*.



Standar deviasi yang adalah 1. Data hasil *powering* yang berada dibawah standar deviasi dianggap sebagai noise.

Tabel 3. 7 Hasil Powering

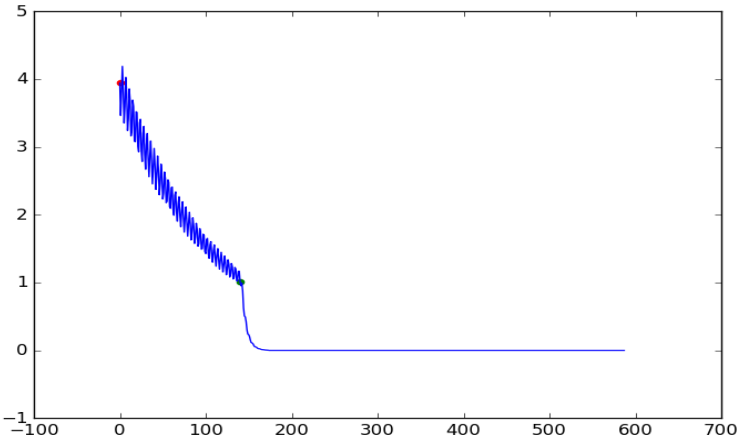
No	P(n)
0	3.94346
1	3.46394
2	3.80224
3	4.19073
…..	…..
140	1.007
141	0.952
….	…..
173	0.002
..	….
588	0.00

Pada data tabel 3.7 data yang dibawah 1 berada pada *frame* ke-141 hingga 588, data tersebut dianggap *noise*. Berikut tabel 3.8 hasil thresholding.

Tabel 3. 8 Hasil Thresholding

No	P(n)
0	3.94346
1	3.46394
2	3.80224
3	4.19073
…..	…..
140	1.007

Dari data yang berada di atas standar deviasi hasil dari thresholding didapatkan nilai batas awal, akhir, index awal dan akhir. Agar lebih jelasnya dapat dilihat pada gambar 3.8 Berikut:



Gambar 3. 8 Hasil *Thresholding* A mayor

Dari hasil pencarian data berdasarkan standar deviasi hasil *powering* dapat dilihat ada dua titik, pada posisi awal dan posisi akhir sinyal. Maka didapatkan data batas awal yang merupakan data ke-0 dan batas akhir yang merupakan data ke-140. Dari data tersebut kemudian dikembalikan ke bentuk index awal dan index akhir yang merupakan amplitudo dari hasil normalisasi. Berikut tabel batas dan index hasil *thresholding*.

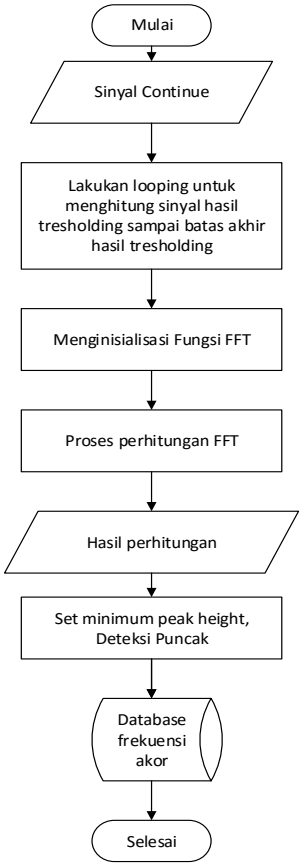
Tabel 3. 9 Index dan Batas

Batas Awal	Batas Akhir	Index Awal	Index Akhir
0	140	0	69860

3.4.1.8 Fast Fourier Transform

FFT merupakan salah satu metode untuk transformasi sinyal suara menjadi sinyal frekuensi. Artinya proses perekaman akor disimpan dalam bentuk *digital* berupa gelombang spektrum suara berbasis frekuensi. Hasil dari proses *fast fourier transform* menghasilkan pendeteksian gelombang frekuensi domain dalam bentuk diskrit.

Berikut diagram alir FFT.



Gambar 3. 9 Flowchart FFT

Setelah data Index awal dan index akhir diketahui maka selanjutnya dilakukan pengkonversian dari domain waktu ke domain frekuensi. Sampel yang akan di proses berjumlah 69860 – 1 = 69859 yang berasal dari thresholding.
Contoh perhitungan FFT.

$$S[k] = \sum_{n=0}^{N-1} s[n] e^{-\frac{j2\pi nk}{N}}, 0 \leq k \leq N-1 \qquad (3.5)$$

Keterangan :

N = jumlah sampel yang akan diproses ($N \in$ N)

S(n) = nilai sampel *signal*

K = variable frekuensi discrete

$$F(k) = \sum_{n=0}^{69859} \left(f_{(n)} \cos \frac{2\pi nkT}{N}\right) - j \sum_{n=0}^{69859} \left(f_{(n)} \sin \frac{2\pi nkT}{N}\right)$$

(3.6)

$$f(0) = \frac{1}{69859}\left[3.94346269 \left(\cos\left(\frac{2\pi*0*0}{69859}\right)\right)\right] - j\ sin\frac{2\pi*0*0}{69859} +$$
$$\cdots+\left[2.05268823 \left(\cos\left(\frac{2\pi*0*69859}{69859}\right)\right)\right] - j\ sin\frac{2\pi*0*69859}{69859} = 0.00008079$$

Hitung dengan cara yang sama untuk data yang lainnya. Namun hasil tersebut berasal dari dua sisi dikarenakan efek simetris FFT. Sehingga untuk menjadi satu frekuensi 69859/2 = 34929, maka akan didapatkan data sebagai berikut :

Tabel 3. 10 ***Fast Fourier Transform***

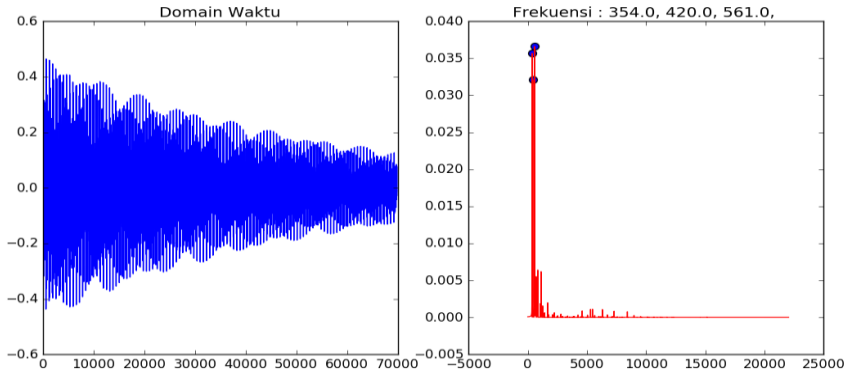
No	FFT
0	0.00008079
1	0.00008084
2	0.00008114
3	0.00008138
4	0.00008203
…	…..
560	**0.035673**
…	…
666	**0.03210423**
…	…
888	**0.03659571**
...	....
34928	0.000000319
34929	0.000000230

Ciri yang diambil dari FFT adalah deteksi puncak. 3 frekuensi yang melewati minimum height = 0.02 yaitu pada amplitudo ke-560, 666, dan 888. Kemudian dicari frekuensi pada amplitudo tersebut. Berikut tabel 3.11 frekuensi FFT.

Tabel 3. 11 Frekuensi FFT

No	Frekuensi
0	0.000
1	0.631
2	1.263
3	1.894
…	…
560	**353.51**
…	…
666	**420.42**
…	…
888	**560.56**
…	…
34928	22048.737
34929	22049.369

Frekuensi puncak dari FFT akan menjadi ciri yang akan dimasukkan kedalam database. Sehingga file A mayor menghasilkan keluaran 3 data yaitu [353.51, 420.42, 560.56].



Gambar 3. 10 Frekuensi A mayor

Data hasil deteksi puncak FFT lengkap dapat dilihat pada tabel 3.11. Data 120 didapat dari data latih yang mempunyai 24 akor tiap akor masing-masing direkam sebanyak 5 kali. Sebagai contoh A#1 yang berarti akor A# mayor pada perekaman pertama. A#2 yang berarti akor A# mayor pada perekaman kedua.

Begitu juga dengan data selanjutnya. Tabel frekuensi akor dapat dilihat pada tabel 3.12 berikut.

Tabel 3. 12 Frekuensi Puncak Akor

Nama File	Frekuensi Puncak		
A#1.wav	374.18	445.45	594.40
A#2.wav	374.02	445.04	594.18
A#3.wav	374.34	444.76	594.21
A#4.wav	373.96	444.81	593.80
A#5.wav	374.58	445.28	594.16
A#m1.wav	353.51	444.76	594.21
A#m2.wav	353.51	444.91	594.47
A#m3.wav	353.51	445.07	593.96
A#m4.wav	353.51	444.98	593.83
A#m5.wav	353.51	444.73	593.93
A1.wav	353.51	420.42	560.56
A2.wav	353.51	420.11	561.00
A3.wav	353.51	420.42	560.56
A4.wav	353.51	420.27	560.78
A5.wav	353.51	420.42	560.56
Am1.wav	333.39	420.33	561.16
Am2.wav	333.31	420.00	560.56
Am3.wav	333.39	420.33	561.16
Am4.wav	333.71	419.97	560.66
Am5.wav	333.27	420.30	560.62
B1.wav	396.22	471.34	629.68
B2.wav	396.89	471.61	629.89
B3.wav	396.86	471.90	629.48
B4.wav	396.83	471.34	629.90
B5.wav	396.92	471.34	629.49
Bm1.wav	374.40	471.61	629.89
Bm2.wav	374.66	471.34	629.21
Bm3.wav	374.13	471.34	629.68
Bm4.wav	374.30	471.34	629.90
Bm5.wav	374.51	471.63	629.14
C#1.wav	353.51	445.19	529.43
C#2.wav	353.51	445.42	529.55
C#3.wav	353.51	445.38	529.39
C#4.wav	353.51	445.53	529.35

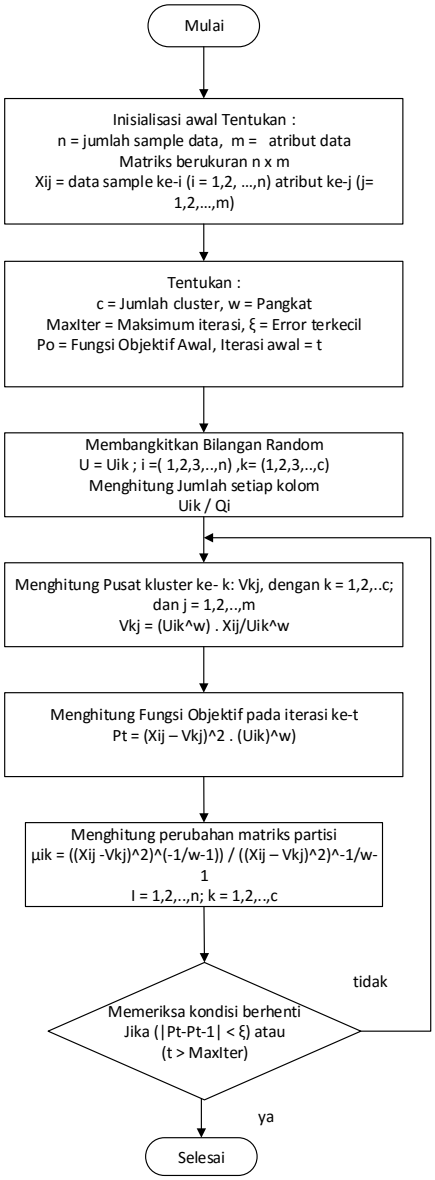
Nama File	Frekuensi Puncak		
D1.wav	374.30	471.59	560.71
D2.wav	374.18	471.82	560.91
D3.wav	374.10	471.91	561.15
D4.wav	374.02	471.87	561.03
D5.wav	374.05	471.76	560.76
Dm1.wav	374.34	444.76	561.16
Dm2.wav	374.11	445.21	560.83
Dm3.wav	374.13	444.83	560.46
Dm4.wav	374.21	445.07	560.52
Dm5.wav	374.01	445.42	560.66
E1.wav	420.16	529.51	629.12
E2.wav	419.97	529.53	629.59
E3.wav	420.37	529.49	629.40
E4.wav	420.30	529.23	629.94
E5.wav	420.37	529.49	629.40
Em1.wav	419.96	499.69	629.27
Em2.wav	419.99	499.45	629.18
Em3.wav	420.19	499.73	629.89
Em4.wav	420.19	499.73	629.89
Em5.wav	420.37	499.52	629.40
F#1.wav	353.51	471.90	594.46
F#2.wav	353.51	471.34	594.23
F#3.wav	353.51	471.84	593.92
F#4.wav	353.51	471.74	593.56
F#5.wav	353.51	471.91	593.75
F#m1.wav	353.51	471.61	560.79
F#m2.wav	353.51	471.34	560.64
F#m3.wav	353.51	471.62	560.82
F#m4.wav	353.51	471.63	560.89
F#m5.wav	353.51	471.61	560.79
F1.wav	333.53	444.96	561.00
F2.wav	333.42	445.10	560.79
F3.wav	333.60	445.07	560.52
F4.wav	333.11	445.28	560.85

Nama File	Frekuensi Puncak		
C#5.wav	353.51	445.16	529.44
C#m1.wav	353.51	420.35	529.52
C#m2.wav	353.51	420.35	529.52
C#m3.wav	353.51	420.39	529.46
C#m4.wav	353.51	420.33	529.54
C#m5.wav	353.51	420.31	529.56
C1.wav	333.62	419.79	499.33
C2.wav	333.29	420.16	499.55
C3.wav	333.87	420.46	499.91
C4.wav	333.67	420.24	499.60
C5.wav	333.79	419.97	499.58
Cm1.wav	333.68	396.46	499.70
Cm2.wav	333.77	396.41	499.37
Cm3.wav	333.68	396.46	499.70
Cm4.wav	333.25	396.77	499.88
Cm5.wav	333.42	396.89	499.73
D#1.wav	396.25	499.84	594.01
D#2.wav	396.60	499.58	593.80
D#3.wav	396.70	499.69	594.05
D#4.wav	396.83	499.94	594.38
D#5.wav	396.41	499.37	593.75
D#m1.wav	396.48	471.88	594.31
D#m2.wav	396.86	471.90	594.46
D#m3.wav	396.71	471.34	594.42
D#m4.wav	396.68	471.57	594.35
D#m5.wav	396.52	471.60	594.39

Nama File	Frekuensi Puncak		
F5.wav	333.31	445.25	560.56
Fm1.wav	333.65	445.09	529.62
Fm2.wav	333.68	445.19	529.43
Fm3.wav	333.70	444.93	529.50
Fm4.wav	333.18	445.42	529.38
Fm5.wav	333.77	445.32	529.40
G#1.wav	333.58	396.83	529.39
G#2.wav	332.95	396.67	529.23
G#3.wav	333.33	396.73	529.30
G#4.wav	333.53	396.54	529.49
G#5.wav	333.65	396.21	529.27
G#m1.wav	353.51	471.34	629.21
G#m2.wav	353.51	471.34	629.28
G#m3.wav	353.51	471.63	629.68
G#m4.wav	353.51	471.34	629.90
G#m5.wav	353.51	471.34	629.58
G1.wav	374.26	499.46	629.35
G2.wav	374.22	499.88	629.68
G3.wav	373.90	499.67	629.68
G4.wav	374.30	499.81	629.78
G5.wav	374.55	499.40	629.16
Gm1.wav	374.55	499.40	594.09
Gm2.wav	374.51	499.63	594.14
Gm3.wav	374.68	499.88	593.78
Gm4.wav	374.21	500.01	593.96
Gm5.wav	374.18	499.61	594.40

3.4.2 *Fuzzy c-means Clustering*

Dalam pembahasan analisis metode *Fuzzy c-means clustering* terdapat beberapa tahapan. Dimulai dengan menghitung pusat cluster pada inisialisasi, *Fuzzy c-means* akan menyempitkan atau konvergensi ruang solusi yang menunjukkan lokal minimum. Konvergensi dapat dideteksi dengan membandingkan perubahan yang terjadi pada membership function atau pusat cluster pada dua buah iterasi. Berikut diagram alir *fuzzy c-means clustering*:



Gambar 3. 11 ***Flowchart fuzzy c-means***

Tahap-tahap yang dilakukan pada langkah ini dapat diuraikan detailnya sebagai berikut:

a) Menetapkan matriks partisi awal (U), berupa sampel data frekuensi. Parameter yang digunakan adalah jumlah data pada tabel 3.11 diatas, n = 120 dan m = 3. Nilai n berdasarkan jumlah perekaman pada 24 akor dengan 5 kali perulangan. Nilai m berdasarkan banyaknya data dalam 1 akor.

b) Menentukan inisialisasi nilai awal pada komponen-komponen fuzzy c-means, misalnya seperti berikut ini:

1) Banyaknya kluster yang diinginkan (c) = 24

Banyaknya kluster berdasarkan jumlah identitas akor yang akan diklasifikasi.

2) Pangkat (w) = 2

Berdasarkan penelitian sebelumnya [20].

3) Maksimum Iterasi (MaxIter) = 100

Maksimum iterasi didapatkan berdasarkan penelitin sebelumnya [20].

4) Toleransi minimum (ξ) = 0.001

Toleransi Minimum Berdasarkan penelitian sebelumnya [21]

5) Fungsi Objektif awal (Po) = 100

Fungsi objektif awal diberi nilai 100 karena data belum masuk kluster maka diberi nilai awal yang besar [7].

6) Iterasi awal = (iter) = 1

c) Membangkitkan bilangan random sebagai data derajat keanggotaan awal untuk elemen matriks partisi awal (U) dengan rentang 0 sampai 1.

Tabel 3. 13 Inisialisasi *fuzzy c-means*

Data ke-i	U_{i0}	U_{i1}	U_{i2}	U_{i3}	….	U_{i22}	U_{i23}
0	0.05560	0.00963	0.04093	0.05345	….	0.04764	0.04765
1	0.03652	0.06471	0.05932	0.05549	….	0.06988	0.06901
3	0.04620	0.04213	0.02946	0.06842	….	0.07566	0.05336
….	….	….	….	….	….	….	….
117	0.05931	0.01865	0.03637	0.04640	….	0.048324	0.072418
118	0.05397	0.06350	0.02502	0.04343	….	0.066519	0.066839
119	0.05147	0.05060	0.03998	0.04715	….	0.010275	0.08323

d) Menghitung Pusat *cluster* pada iterasi awal dengan persamaan 2.10 berikut.

$$V_{kj} = \frac{\sum_{i=0}^{n}((U_{ik})^{w} * X_{ij})}{\sum_{i=0}^{n}(U_{ik})^{w}}$$

Keterangan :

V_{kj} = pusat *cluster* ke-*k* untuk atribut ke-*j*

U_{ik} = derajat keanggotaan untuk data sampel ke-*i* pada cluster ke-*k*

X_{ij} = data ke-*i*, atribut ke-*j*

$$V_{kj} = \frac{\sum_{i=0}^{n}((U_{ik})^{w} * X_{ij})}{\sum_{i=0}^{n}(U_{ik})^{w}} = \frac{\sum_{i=0}^{119}((U_{i0})^{2} * X_{ij})}{\sum_{i=0}^{119}(U_{i0})^{2}}$$

= 120.6715/0.330607 = 365.00

Hitung dengan cara yang sama untuk proses selanjutnya, hasil perhitungan dapat dilihat pada tabel 3.14 berikut:

Tabel 3. 14 Pusat kluster Awal

Data ke- i	***Cluster* 0**	**Data**			$(U_{i0})^2$	$(U_{i0})^2$ x X_{i0}	$(U_{i0})^2$ x X_{i1}	$(U_{i0})^2$ x X_{i2}
	U_{i0}	X_{i0}	X_{i1}	X_{i2}				
0	0.05560	374.18	445.45	594.40	0.003091	1.156712	1.377038	1.83752
1	0.03652	374.02	445.04	594.18	0.00133	0.49884	0.59355	0.79246
2	0.04620	374.34	444.76	594.21	0.00213	0.79901	0.94931	1.26830
3	0.009375	373.96	444.81	593.80	0.00009	0.03287	0.03909	0.05219
…	…	…	…	…	…	…	…	…
118	0.05397	374.21	500.01	593.96	0.00291	1.02968	1.29547	1.73079
119	0.05147	374.18	499.61	594.40	0.00265	0.93650	1.17863	1.57484
	$\sum_{0}^{119}$				0.330607	120.6715	151.4710	190.5089
	$V_{0j} = \frac{\sum_{i=0}^{119}((U_{i0})^{2} * X_{ij})}{\sum_{i=0}^{119}(U_{i0})^{2}}$					365.00	458.17	576.24

Lakukan hal yang sama hingga mendapatkan V_{1j} sampai V_{23j}, yang dapat dilihat pada tabel 3.15 berikut:

Tabel 3. 15 Hasil Pusat Cluster Awal V_{0j} sampai V_{23j}

Cluster	Pusat Cluster Iterasi 1		
	j = 0	j = 1	j = 2
V_{0j}	365.00	458.17	576.24
V_{1j}	366.53	462.07	582.27
V_{2j}	364.98	458.10	579.01
V_{3j}	362.37	454.38	572.58
V_{4j}	364.81	457.62	577.02
V_{5j}	365.19	457.13	572.12
V_{6j}	361.90	452.73	573.05
V_{7j}	365.42	457.43	575.61
V_{8j}	369.59	463.61	583.32
V_{9j}	363.92	457.48	576.00
V_{10j}	364.19	459.87	576.47
V_{11j}	363.94	455.29	572.52
V_{12j}	361.96	453.16	572.49
V_{13j}	365.03	457.68	574.21
V_{14j}	363.94	458.67	577.21
V_{15j}	366.09	458.64	579.49
V_{16j}	364.80	457.63	576.14
V_{17j}	362.86	456.06	575.31
V_{18j}	364.39	455.33	576.86
V_{19j}	361.29	453.71	571.55
V_{20j}	366.43	454.89	576.11
V_{21j}	368.65	462.81	582.80
V_{22j}	361.67	456.73	574.49
V_{23j}	363.21	454.89	572.81

e) Menghitung Fungsi Objektif (P)

Fungsi objektif pada iterasi 1 (P1) dapat dihitung dengan menggunakan persamaan 2.11 berikut.

$$PT_{ik} = \sum_{i=0}^{n} \sum_{k=0}^{c} \left(\left[\sum_{j=0}^{m} (X_{ij} - V_{kj})^2 \right] (U_{ik})^w \right)$$

Keterangan :

V_{kj} = pusat *cluster* ke-*k* untuk atribut ke-*j*

U_{ik} = derajat keanggotaan untuk data sampel ke-*i* pada cluster ke-*k*

x_{ij} = data ke-*i*, atribut ke-*j*

Pt_{ik} = fungsi objektif pada iterasi ke-t, data ke-i, dan kelas ke-k

$$Pt_{ik}= \sum_{i=1}^{n}\sum_{k=1}^{c}\left(\left[\sum_{j=1}^{m}(X_{ij}-V_{kj})^2\right](U_{ik})^w\right) = Pt_{ik}= \sum_{i=0}^{119}\sum_{k=0}^{23}\left(\left[\sum_{j=0}^{2}(X_{ij}-V_{kj})^2\right](U_{ik})^2\right)$$

Dimana:

$$P1_{00} = \left[\sum_{j=0}^{2}(X_{0j} - V_{0j})^2\right](U_{0-0})^2$$

$$P1_{1k} = \left[\sum_{j=0}^{2}(X_{1j} - V_{1j})^2\right](U_{1-1})^2$$

…….

$$P1_{ik} = \left[\sum_{j=0}^{2}(X_{ij} - V_{23j})^2\right](U_{i-23})^2$$

Data ke-0:

$P1_{00}$ = $((X_{0\text{-}0} - V_{0\text{-}0})^2 + (X_{0\text{-}1}-V_{0\text{-}1})^2 +(X_{0\text{-}2}-V_{0\text{-}2})^2)*(U_{0\text{-}0})^2$

$P1_{11}$ = $((X_{0\text{-}0} – V_{1\text{-}0})^2 + (X_{0\text{-}1}-V_{1\text{-}1})^2 +(X_{0\text{-}2}-V_{1\text{-}2})^2)*(U_{0\text{-}1})^2$

…...

$P1_{0-23}$ =$((X_{0\text{-}0}-V_{23\text{-}0})^2 +(X_{0\text{-}1}-V_{23\text{-}1})^2+(X_{0\text{-}2}-V_{23\text{-}2})^2)*(U_{0\text{-}23})^2$

P1 tot = $P1_{1\text{-}0}$+$P1_{1\text{-}2}$+$P1_{1\text{-}3}$ +…..+$P1_{0\text{-}23}$

Contoh :

$P1_{0\text{-}0}$ = $((X_{0\text{-}0}-V_{0\text{-}0})^2 + (X_{0\text{-}1}-V_{0\text{-}1})^2 + (X_{0\text{-}2} – V_{0\text{-}3})^2)*(U_{0\text{-}0})^2$

$P1_{0\text{-}1}$ = $((374.18 - 365.00)^2) + ((445.45 - 458.17)^2) + ((594.40 - 576.24)^2)* 0.003091$

= 0.985573

Hitung dengan cara yang sama hingga $P1_{0\text{-}23}$ Kemudian dijumlahkan

P1total = $P1_{0\text{-}0}$+$P1_{0\text{-}1}$+$P1_{0\text{-}2}$ +…..+$P1_{0\text{-}23}$

P1 tot = 0.985573 + 2.131733+ 0.632329 + …..+0.087397 = 1342.801

Hitung dengan cara yang sama hingga data ke 119 atau $P1_{119-23}$ sehingga menghasilkan hasil akhir yang ditunjukan pada tabel 3.16 berikut.

Tabel 3. 16 Fungsi Objektif Awal

Data ke-i	$P1_{i-0}$	$P1_{i-1}$	$P1_{i-2}$		$P1_{i-23}$	$P1_{total}$
0	0.985573	2.131733	0.632329	….	0.087397	1342.801
1	0.912598	0.916156	0.461468	….	0.000996	715.5787
2	0.314999	2.383967	0.109966	….	1.888613	361.4938
3	0.039556	2.21633	0.626776	….	0.005694	596.3249
..	……	…..	…	….	…	….
119	5.432152	8.690939	3.541044	….	0.462406	127.048
						79711.91

Fungsi Objektif pada Iterasi 1 ($P1$) = 79711.91

f) Menghitung Perubahan matriks partisi

Perubahan matriks partisi dihitung dengan menggunakan persamaan 2.12 berikut.

$$U_{ik} = \frac{[\sum_{j=0}^{m} (X_{ij}-V_{kj})^2]^{\frac{-1}{w-1}}}{\sum_{k=0}^{c}[\sum_{j=0}^{m}(X_{ij}-V_{kj})^2]^{\frac{-1}{w-1}}}$$

Keterangan :

Dengan i = 0,1,2,3,..,119

k = 0,1,2,..,23

m = 0,1,2.

V_{kj} = pusat *cluster* ke-*k* untuk atribut ke-*j*

U_{ik} = derajat keanggotaan untuk data sampel ke-*i* pada cluster ke-*k*

x_{ij} = data ke-*i*, atribut ke-*j*

$$U_{ik} = \frac{[\sum_{j=0}^{m} (X_{ij}-V_{kj})^2]^{\frac{-1}{w-1}}}{\sum_{k=0}^{c}[\sum_{j=0}^{m}(X_{ij}-V_{kj})^2]^{\frac{-1}{w-1}}} = \frac{\left[\sum_{j=0}^{2}\left(X_{ij}-V_{kj}\right)^2\right]^{\frac{-1}{2-1}}}{\sum_{k=0}^{23}\left[\sum_{j=0}^{2}\left(X_{ij}-V_{kj}\right)^2\right]^{\frac{-1}{2-1}}}$$

Atau

$U_{ik} = P1_1/P_{tot}$

$U_{i0} = P1_{0-0}/P1_{tot-0}$

$U_{i1} = P1_{0-1}/P1_{tot-0}$

$U_{i2} = P1_{0-2}/P1_{tot-0}$

…..

$U_{i23} = P1_{119-23}/P1_{tot-119}$

Sebagai contoh

$U_{0-0} = P1_{0-0}/P1_{tot-0}$

= 0.985573/1342.801 = 0.00073

Hasil perhitungan dapat dilihat pada tabel 3.17 berikut ini.

Tabel 3. 17 Posisi Cluster Iterasi ke-1

Data ke-i	Derajat Keanggotaan Untuk Iterasi Ke-1							Posisi Cluster
	U_{i0}	U_{i1}	U_{i2}	U_{i3}		U_{i22}	U_{i23}	
0	0.00073	0.00159	0.00047	0.00075	….	0.00003	0.00007	19
1	0.00128	0.00128	0.00064	0.00258	….	0.00288	0.00000	19
2	0.00087	0.00659	0.00030	0.00930	….	0.00606	0.00522	19
3	0.00007	0.00372	0.00105	0.00527	….	0.00426	0.00001	19
4	0.00172	0.00230	0.00089	0.00250	….	0.00136	0.00108	19
5	0.00491	0.00096	0.00407	0.00201	….	0.00341	0.00006	19
….	….	….	….	….	….	….	….	….
118	0.00543	0.03810	0.04072	0.00107	….	0.03253	0.01907	19
119	0.04276	0.06841	0.02787	0.04400	….	0.00496	0.00364	8

g) Memeriksa Kondisi Berhenti

Iterasi akan dihentikan jika memenuhi syarat berikut ini:

- Jika : ($|Pt-Pt_{-1}| < \xi$) atau (t > MaxIter) maka berhenti
- Jika tidak , t = t +1 , mengulang ke langkah ke d yaitu menghitung pusat *cluster*

Karena P1 – Po = 79711.91 – 100 = 79511.91 > 0.001 dan iterasi = 1 < MaxIter (100), maka proses dilanjutkan ke iterasi kedua (t = 2) dan seterusnya sampai kondisi berhenti.

Pada iterasi terakhir (iterasi 100), pusat *cluster* dan fungsi objektif yang dihasilkan adalah :

Tabel 3. 18 Pusat Cluster Iterasi Terakhir

Akor	*C*	Pusat *Cluster*		
A#	V_{0j}	374.2	445.1	594.2
A#m	V_{1j}	353.5	444.9	594.1
A	V_{2j}	353.5	420.3	560.7
Am	V_{3j}	333.4	420.2	560.8
B	V_{4j}	396.8	471.5	629.7
Bm	V_{5j}	374.4	471.5	629.6
C#	V_{6j}	353.5	445.3	529.4
C#m	V_{7j}	353.5	420.3	529.5
C	V_{8j}	333.7	420.1	499.6
Cm	V_{9j}	333.6	396.6	499.7
D#	V_{10j}	396.6	499.7	594
D#m	V_{11j}	396.7	471.7	594.4

Akor	*C*	Pusat *Cluster*		
D	V_{12j}	374.1	471.79	560.9
Dm	V_{13j}	374.2	445.06	560.7
E	V_{14j}	420.2	529.45	629.5
Em	V_{15j}	420.1	499.63	629.5
F#	V_{16j}	353.5	471.75	594
F#m	V_{17j}	353.5	471.56	560.8
F	V_{18j}	333.4	445.13	560.7
Fm	V_{19j}	333.6	445.19	529.5
G#	V_{20j}	333.4	396.6	529.3
G#m	V_{21j}	353.5	471.4	629.5
G	V_{22j}	374.3	499.64	629.5
Gm	V_{23j}	374.4	499.71	594.1

Derajat keanggotaan baru untuk iterasi terakhir serta tabel kecenderungan data untuk menjadi anggota *cluster* tertentu pada iterasi terakhir diperlihatkan pada tabel 3.19 berikut.

Tabel 3. 19 Data Keanggotaan Baru Iterasi Terakhir

Data ke-i	Derajat Keanggotaan Baru Iterasi Terakhir							Cluster
	U_{i0}	U_{i1}	U_{i2}	U_{i3}	…	U_{i22}	U_{i23}	
0	0.000053	0.000113	0.000057	0.000006	…	0.000022	0.000018	9
1	0.000007	0.000015	0.000008	0.000001	…	0.000003	0.000002	9
2	0.000031	0.000066	0.000032	0.000004	…	0.000012	0.000010	9
3	0.000071	0.000154	0.000077	0.000009	…	0.000029	0.000024	9
4	0.000029	0.000063	0.000031	0.000004	…	0.000012	0.000010	9
5	0.000057	0.000046	0.000088	0.000025	…	0.000043	0.000052	14
…	…	…	…	…	…	…	…	
118	0.000031	0.000037	0.000021	0.000007	…	0.000013	0.000012	11
119	0.000018	0.000022	0.000012	0.000004	…	0.000007	0.000007	11

Dari tabel 3.19 dapat dilihat bahwa ada 5 angka yang sama dan berurutan pada tiap kluster nya. Karena menggunakan bilangan random maka akan terjadi perubahan

data namun intinya tetap sama yaitu 5 angka yang sama dan berurutan pada tiap kluster nya.

3.4.3 Data Uji

Data uji diambil dari beberapa akor berurutan yang direkam dalam satu file tahap yang dilalui yaitu *preprocessing*. Tahap preprocessing data uji sama seperti data latih yang terdiri dari normalisasi, *frame blocking, windowing, Powering, Thresholding, Fast fourier transform.* Salah satu contoh data uji yang telah melalui proses *preprocessing* adalah file 'CFGCFG.wav' yang berisi akor C-F-G-C-F-G secara berurutan. Hasil deteksi puncak data uji ini dapat dilihat pada tabel 3.20

Tabel 3. 20 Frekuensi Puncak Data Uji

Akor	Frekuensi Puncak		
C	333.71	419.97	499.86
F	333.53	444.96	561.00
G	374.42	499.90	629.43
C	333.65	420.11	499.52
F	333.39	444.76	561.16
G	374.61	499.92	629.19

3.4.4 Euclidean Distance

Untuk dapat melakukan verifikasi akor, maka data *signal* baru yang masuk akan dicocokan dengan data yang telah ada dalam *database* sebelumnya yang telah dilakukan pembelajaran *fuzzy c-means clustering*. Setiap vektor dari model yang diujicobakan, dibandingkan dan dihitung Euclidean distance dengan semua vektor yang ada pada salah satu model *database*.

Salah satu data uji pengenalan akor yang telah melalui tahap *preprocessing* ditampilkan pada tabel 3.21 berikut.

Tabel 3. 21 Data Uji

Akor	Variabel	Frekuensi Puncak			Target Cluster
C	Z1	333.71	419.97	499.86	C8
F	Z2	333.53	444.96	561.00	C18
G	Z3	374.42	499.90	629.43	C22
C	Z4	333.65	420.11	499.52	C8
F	Z5	333.39	444.76	561.16	C18
G	Z6	374.61	499.92	629.19	C22

Data yang menjadi data pattern adalah centroid terakhir yang dihasilkan dari iterasi terakhir. Data dapat dilihat pada tabel 3.22 berikut.

Tabel 3. 22 Data Pattern

Akor	Var	Pusat *Cluster*			Target
A#	Y1	374.22	445.07	594.15	C0
A#m	Y2	353.51	444.89	594.08	C1
A	Y3	353.51	420.33	560.69	C2
Am	Y4	333.41	420.18	560.83	C3
B	Y5	396.75	471.51	629.69	C4
Bm	Y6	374.40	471.45	629.56	C5
C#	Y7	353.51	445.34	529.43	C6
C#m	Y8	353.51	420.34	529.52	C7
C	Y9	333.65	420.13	499.59	C8
Cm	Y10	333.56	396.60	499.68	C9
D#	Y11	396.56	499.68	594.00	C10
D#m	Y12	396.65	471.66	594.39	C11
D	Y13	374.13	471.79	560.91	C12
Dm	Y14	374.16	445.06	560.72	C13
E	Y15	420.23	529.45	629.49	C14
Em	Y16	420.14	499.63	629.52	C15
F#	Y17	353.51	471.75	593.98	C16
F#m	Y18	353.51	471.56	560.79	C17
F	Y19	333.39	445.13	560.74	C18
Fm	Y20	333.60	445.19	529.47	C19
G#	Y21	333.41	396.60	529.34	C20
G#m	Y22	353.51	471.40	629.53	C21
G	Y23	374.25	499.64	629.53	C22
Gm	Y24	374.42	499.71	594.07	C23

Pada pencocokan akor akan dibandingkan data uji (Z) dengan data pattern (Y) menggunakan perhitungan jarak terdekat Euclidean distant dengan persaman 2.10 berikut.

$$d(Z,Y) = \sqrt{\sum_{i=1}^{r}(Z_i - Y_i)^2}$$

Keterangan :

Z = data uji

Y = *data pattern*

$Z_{i,}Y_{i,}$ = nilai fitur ke i dari Z dan Y

Z1 = 333.71 419.97 499.86

Perhitungan jarak pada data uji Z1 dengan semua data pattern (Y1, Y2, Y3,..,Y24) sebagai berikut :

$$d(Z_1 - Y_1) = \sqrt{(Z1_{x1} - Y1_{y1})^2 + (Z1_{x2} - Y1_{y2})^2 + (Z1_{x3} - Y1_{y3})^2}$$

$$= \sqrt{(333.71 - 374.22))^2 + (419.97 - 445.07))^2 + (499.86 - 594.15)^2}$$

= 105.65

$$d(Z_1 - Y_2) = \sqrt{(Z1_{x1} - Y2_{y1})^2 + (Z2_{x2} - Y2_{y2})^2 + (Z1_{x3} - Y2_{y3})^2}$$

$$= \sqrt{(333.71 - 374.22))^2 + (419.97 - 445.07))^2 + (499.86 - 594.15)^2} =$$

= 99.45

Lakukan terus perhitungan jarak sampai $d(Z_1 - Y_{24})$. Hasil dari perhitungan jarak $d(Z_1 - Y_{24})$ dapat dilihat pada tabel 3.23 berikut.

Tabel 3. 23 Pencarian jarak

Variabel	Jarak		Variabel	Jarak
Y1	105.6488		Y13	89.7006
Y2	99.4508		Y14	77.2635
Y3	63.9723		Y15	190.4613

Variabel	Jarak		Variabel	Jarak
Y4	60.9711		Y16	175.0074
Y5	153.2522		Y17	109.2327
Y6	145.3546		Y18	82.2559
Y7	43.7043		Y19	65.8749
Y8	35.6636		Y20	38.8949
Y9	**0.3195**		Y21	37.6207
Y10	23.3712		Y22	140.895
Y11	138.4419		Y23	157.4964
Y12	124.7767		Y24	129.9665

Dari tabel 3.22 didapat jarak terdekat adalah Y9 yaitu 0.3195, data uji Z1 target kluster = C8, data pattern Y9 target kelasnya = C8, maka Z1 merupakan akor C. Hitung dengan cara yang sama untuk akor selanjutnya.

Hasil akhir data uji dapat dilihat pada tabel 3.24 berikut.

Tabel 3. 24 Hasil akhir Pengujian

Variabel	jarak terdekat		Prediksi	Target Cluster	Akor
	Variabel	Nilai			
Z1	Y9	0.3195	C8	C8	C
Z2	Y19	0.3407	C18	C18	F
Z3	Y23	0.3263	C22	C22	G
Z4	Y9	0.728	C8	C8	C
Z5	Y19	0.5597	C18	C18	F
Z6	Y23	0.5689	C22	C22	G

3.5 Analisis Kebutuhan Non Fungsional

Analisis non fungsional yang dilakukan dibagi ke dalam 2 tahap, yaitu analisis kebutuhan perangkat keras dan analisis kebutuhan perangkat lunak.

3.5.1 Analisis Kebutuhan Perangkat Keras

Analisis kebutuhan perangkat keras pada penelitian ini merupakan kebutuhan perangkat keras yang digunakan dalam pembangunan perangkat lunak ini. Adapun perangkat keras yang digunakan sebagai berikut.

a. Processor 2.10 GHz
b. RAM 2 GB
c. *Keyboard*
d. *Mouse*
e. *Speaker*

3.5.2 Analisis Kebutuhan Perangkat Lunak

Analisis kebutuhan perangkat lunak pada penelitian ini merupakan *tools* yang digunakan untuk pembangunan aplikasi ini. *Tools* yang digunakan adalah sebagai berikut:

a. OS Windows 10
b. Spyder (Python IDE)
c. Qt Designer (Python GUI)

3.5.3 Analisis Pengguna

Spesifikasi pengguna yang dapat menjalankan aplikasi yang akan dibangun hanya terdapat satu jenis user yaitu seorang penguji. Spesifikasi pengguna yang dibutuhkan adalah sebagai berikut :

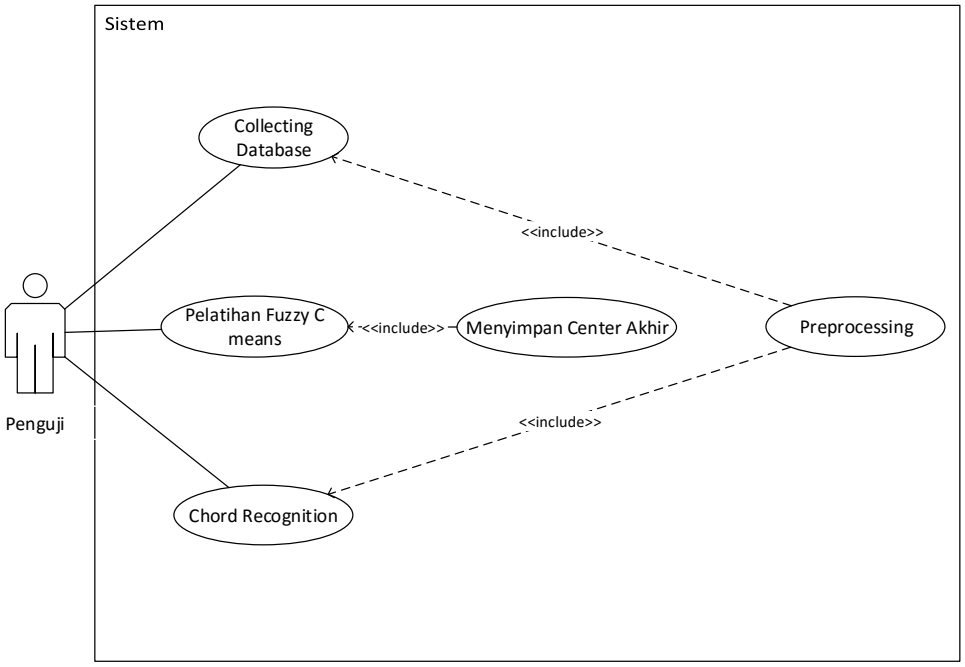
a. Menguasai dalam penggunaan komputer.
b. Mengerti secara teknis mengenai *chord recognition*.
c. Mengerti maksud penggunaan *fuzzy c-means clustering*.

3.6 Analisis Kebutuhan Fungsional

Analisis kebutuhan fungsional bertujuan untuk menganalisis proses yang diterapkan dalam perangkat lunak dan menjelaskan kebutuhan yang diperlukan. Perangkat lunak ini dimodelkan dengan menggunakan model *unified modeling language* (UML). Bagian-bagian yang dilakukan dalam analisis tersebut antara lain.

3.6.1 Use Case Diagram

Use case diagram menggambarkan proses yang ada dalam system. Gambar 3.12 adalah use case implementasi metode *Fuzzy c-means clustering* pada pengenalan akor.



Gambar 3. 12 *Use Case Diagram*

3.6.2 *Use case scenario*

Use case scenario menjelaskan skenario untuk tiap *use case* yang menggambarkan urutan interaksi aktor dengan *use case* tersebut.

Tabel 3. 25 *Use case scenario Collecting Database*

Nama *Use Case*	Collecting Database
Tujuan	Menyimpan hasil dari *preprocessing* data latih
Deskripsi	Fungsionalitas ini digunakan penguji untuk menyimpan hasil dari *preprocessing* data latih
Aktor	Penguji
Related Case	-
Kondisi Sukses	Berhasil menyimpan hasil dari *preprocessing* data latih
Kondisi Gagal	Gagal menyimpan hasil dari preprocessing data latih
Include Case	*Preprocessing*
Trigger	Collecting database
Skenario Utama	
Aksi aktor	Reaksi Sistem
1. Menjalankan *Collecting database*	
	2. Menampilkan nama folder yang akan di input dan nama folder tempat menyimpan frekuensi hasil *preprocessing*.
3. Menjalankan *preprocessing*	
	4. Melakukan *preprocessing*
	5. Menyimpan hasil *preprocessing*

Tabel 3. 26 *Use Case Scenario Preprocessing*

Nama *Use Case*	Preprocessing
Tujuan	Melakukan proses *preprocessing*
Deskripsi	Fungsionalitas ini digunakan penguji untuk melakukan proses *preprocessing*
Aktor	Penguji
Related Case	*Collecting Database, Chord Recognition*
Kondisi Sukses	Berhasil melakukan preprocessing
Kondisi Gagal	Gagal melakukan preprocessing
Include Case	-
Trigger	Proses
Skenario Utama	
Aksi aktor	Reaksi Sistem
1 .Menjalankan *preprocessing*	
	2. Melakukan normalisasi, *frame blocking, windowing, powering, thresholding, fast fourier transform,*dan *detect peaks*

Tabel 3. 27 *Use case scenario* Pelatihan *Fuzzy c-means*

Nama *Use Case*	Pelatihan *Fuzzy C-Means*
Tujuan	Melakukan pelatihan *Fuzzy C-Means* terhadap data latih
Deskripsi	Fungsionalitas ini digunakan penguji untuk melakukan pelatihan *Fuzzy C-Means* terhadap data latih.
Aktor	Penguji
Related Case	-
Kondisi Sukses	Berhasil melakukan pelatihan *Fuzzy C-Means* data latih
Kondisi Gagal	Gagal melakukan pelatihan *Fuzzy C-Means* data latih
Include Case	Menyimpan center akhir

Trigger	Pelatihan *Fuzzy c-means*
Skenario Utama	
Aksi aktor	Reaksi Sistem
1. Menjalankan pelatihan *Fuzzy c-means*	
	2. Menampilkan folder hasil preprocessing, menampilkan setting FCM
3. Mengubah atau sesuai *default* parameter *Fuzzy C-Means*	
4. Menjalankan proses pelatihan *Fuzzy C-Means*	
	5. Melakukan proses pelatihan *Fuzzy C-Means*

Tabel 3. 28 *Use Case Scenario* Menyimpan *Center* Akhir

Nama *Use Case*	Menyimpan Center Akhir
Tujuan	Menyimpan center akhir hasil pelatihan *Fuzzy C-Means*
Deskripsi	Fungsionalitas ini digunakan penguji untuk menyimpan center akhir hasil pelatihan fuzzy c-means
Aktor	Penguji
Related Case	Setting
Kondisi Sukses	Berhasil menyimpan center akhir hasil pelatihan fuzzy c-means
Kondisi Gagal	Gagal menyimpan center akhir hasil pelatihan fuzzy c-means
Include Case	Pelatihan Fuzzy C-Means
Trigger	proses
Skenario Utama	
Aksi aktor	Reaksi Sistem
1. Menjalankan proses pelatihan *Fuzzy C-Means*	
	2. Melakukan pelatihan *Fuzzy C-Means*
	3. Menyimpan center akhir hasil Pelatihan *Fuzzy C-Means*

Tabel 3. 29 *Use Case Scenario Chord Recognition*

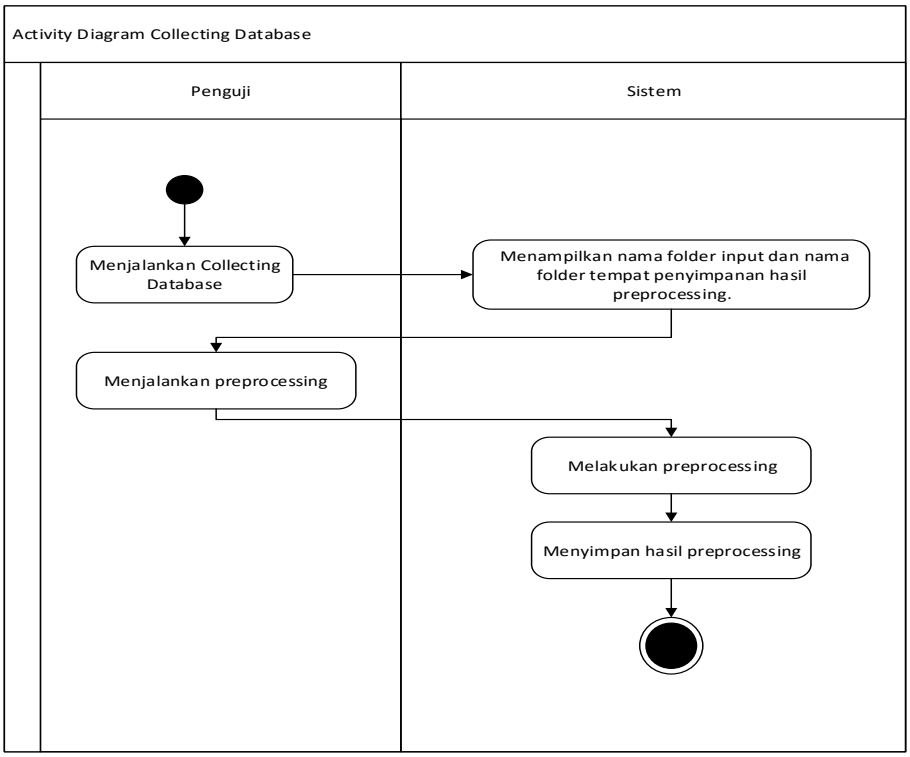
Nama *Use Case*	*Chord Recognition*
Tujuan	Mendapatkan hasil deteksi akor
Deskripsi	Fungsionalitas ini digunakan penguji untuk mendapatkan hasil deteksi akor
Aktor	Penguji
Related Case	-
Kondisi Sukses	Berhasil mendapatkan hasil deteksi akor
Kondisi Gagal	Gagal mendapatkan hasil deteksi akor
Include Case	Preprocessing
Trigger	Modul Chord recognition
Skenario Utama	
Aksi aktor	Reaksi Sistem
1. Menjalankan *chord recognition*	
	2. Menampilkan menu input data uji atau buka file
3. Menginputkan data uji	

	3. Melakukan preprocessing
	4. Menampilkan hasil deteksi akor

3.6.3 *Activity diagram*

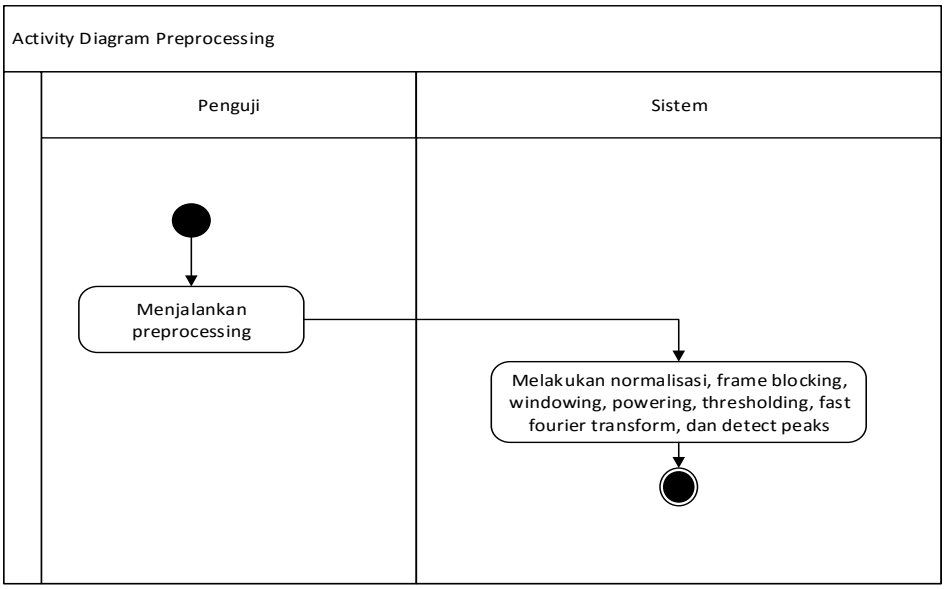
Activity diagram menggambarkan berbagai alir aktivitas dalam sistem yang dirancang, bagaimana masing-masing alir berawal, decision (pilihan), yang mungkin terjadi dan bagaimana berakhir. Berikut *activity diagram* :

1. *Activity diagram Collecting Database*



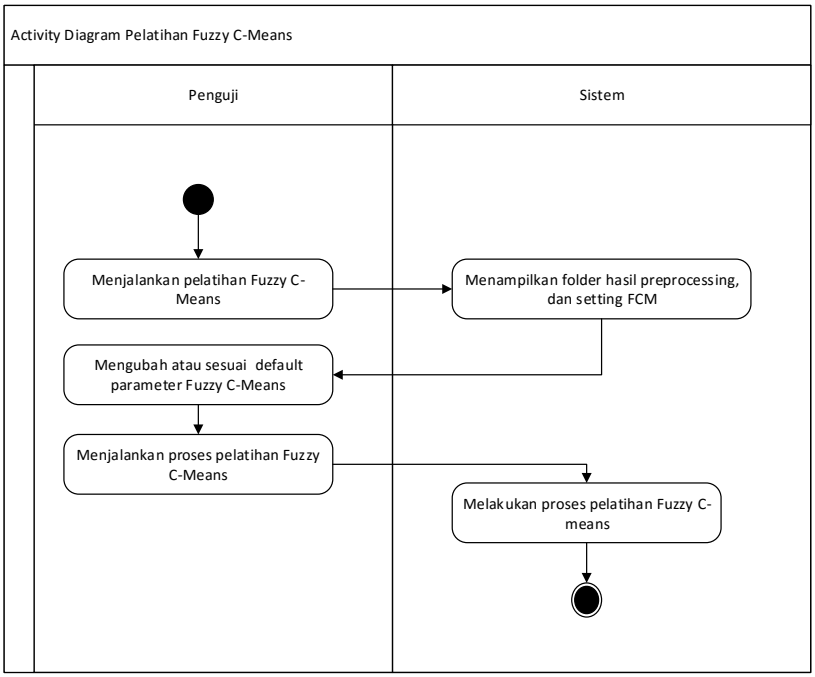
Gambar 3. 13 *Activity diagram Collecting Database*

2. *Activity diagram Preprocessing*



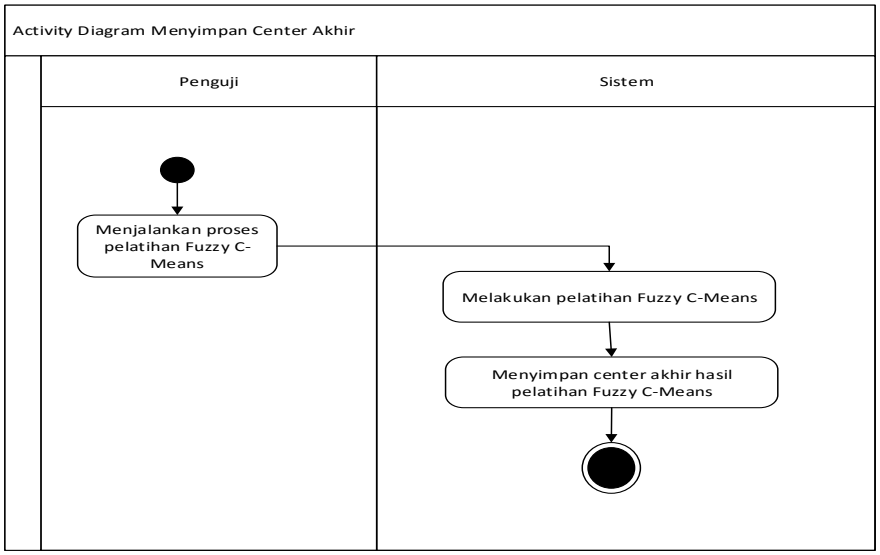
Gambar 3. 14 *Activity Diagram Preprocessing*

3. *Activity Diagram* Pelatihan *Fuzzy C-Means*



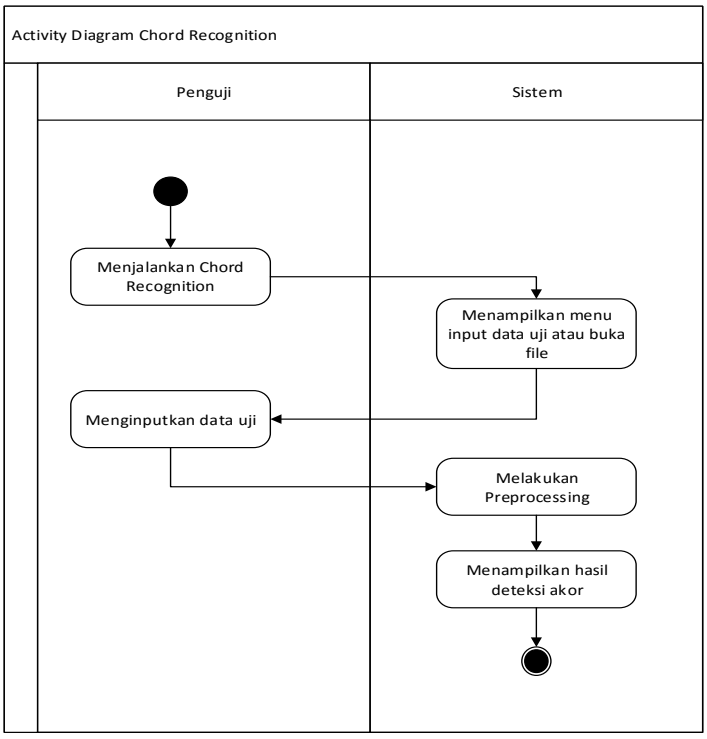
Gambar 3. 15 *Activity diagram* Pelatihan Fuzzy C-Means

4. *Activity Diagram* Menyimpan Center Akhir



Gambar 3. 16 *Activity diagram* Menyimpan Center Akhir

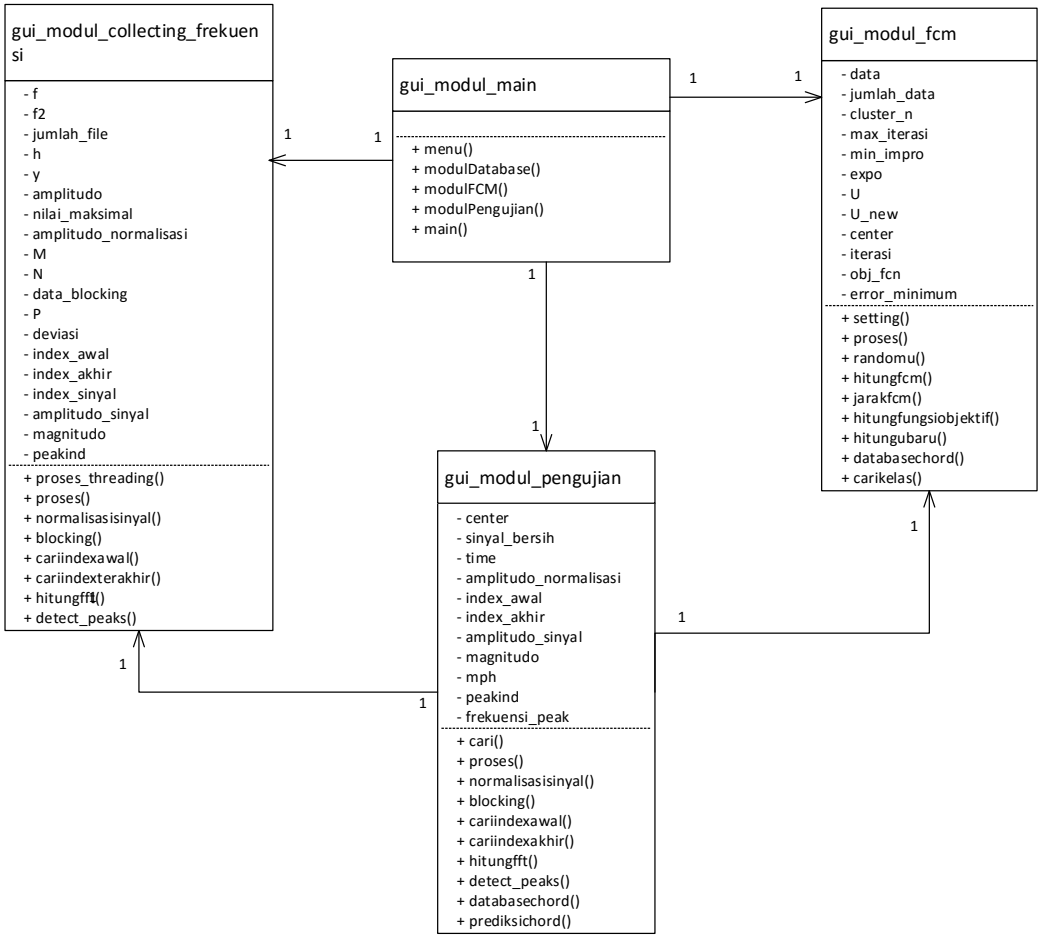
5. *Activity Diagram Chord Recognition*



Gambar 3. 17 *Activity Diagram Chord Recognition*

3.6.4 *Class diagram*

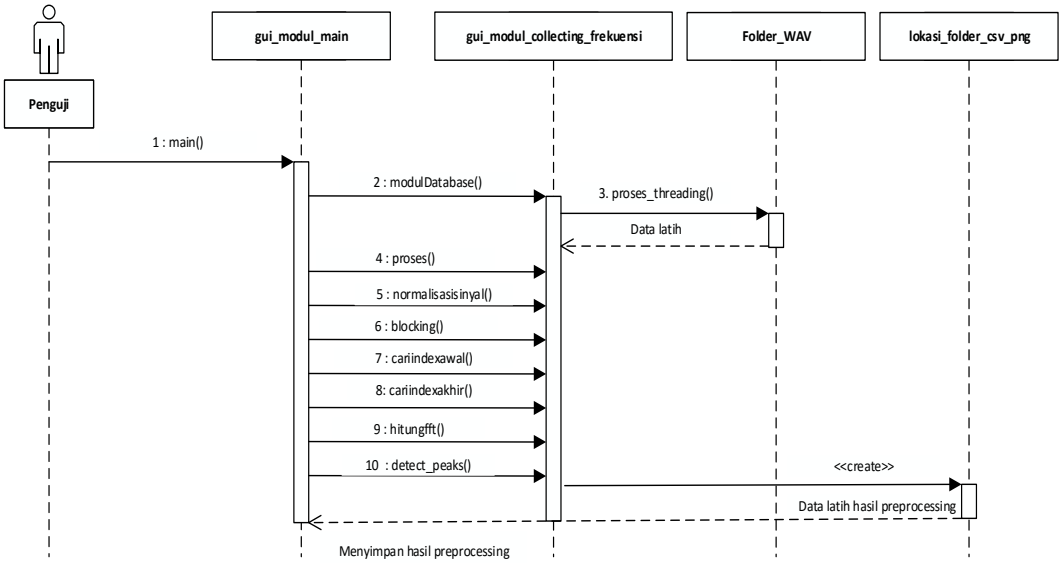
Class diagram adalah sebuah spesifikasi yang dapat menghasilkan sebuah objek dan merupakan inti dari desain berorientasi objek. *Class diagram* dapat dilihat pada gambar berikut.



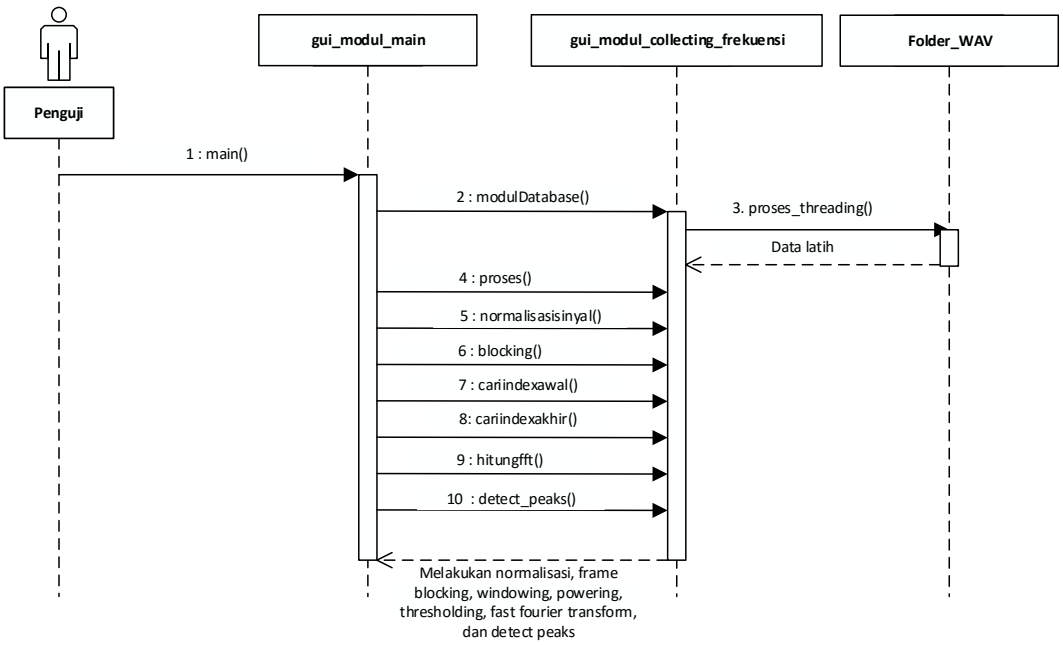
Gambar 3. 18 *Class diagram*

3.6.5 *Sequence diagram*

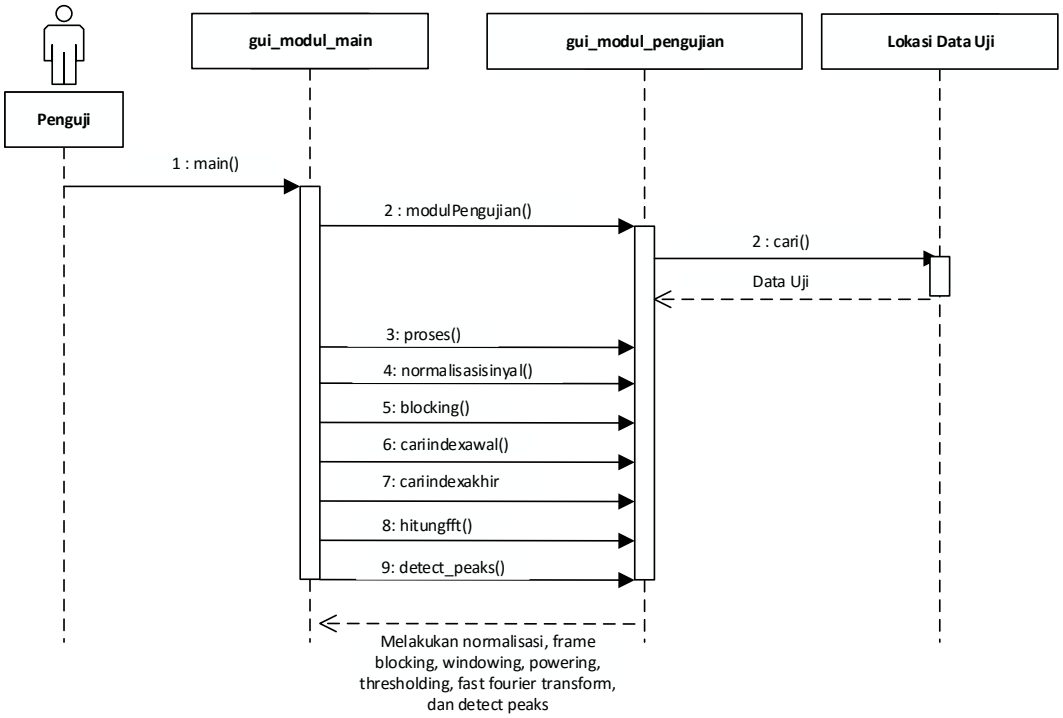
Berikut ini adalah *sequence diagram* dari pembangunan aplikasi pengenalan akor.



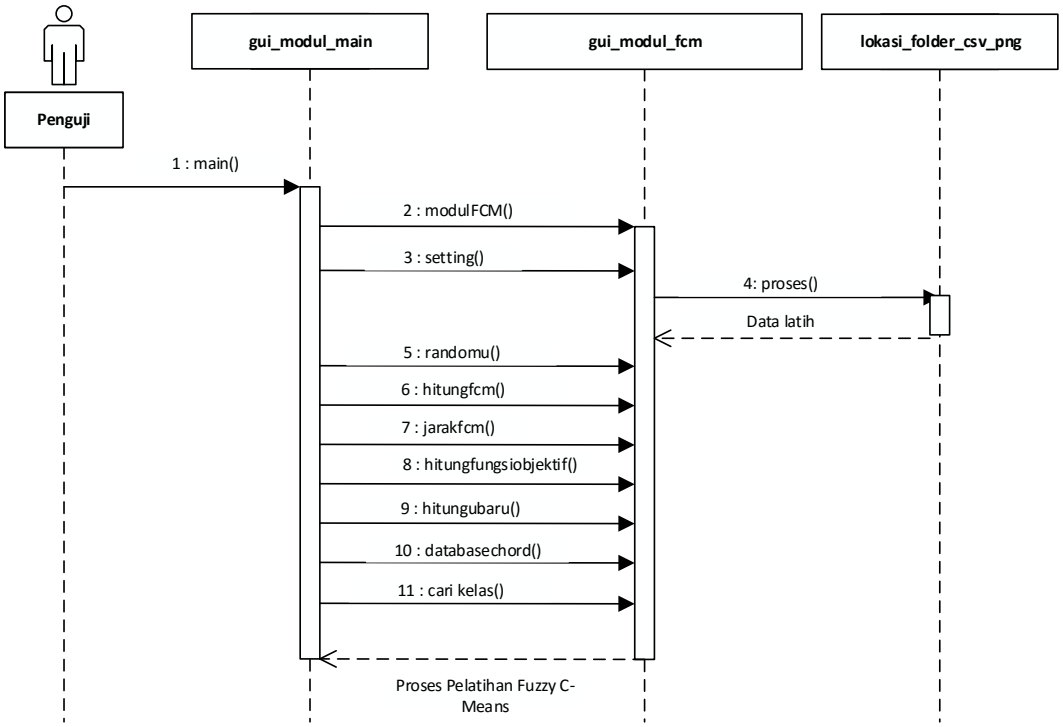
Gambar 3. 19 ***Sequence diagram Collecting Database***



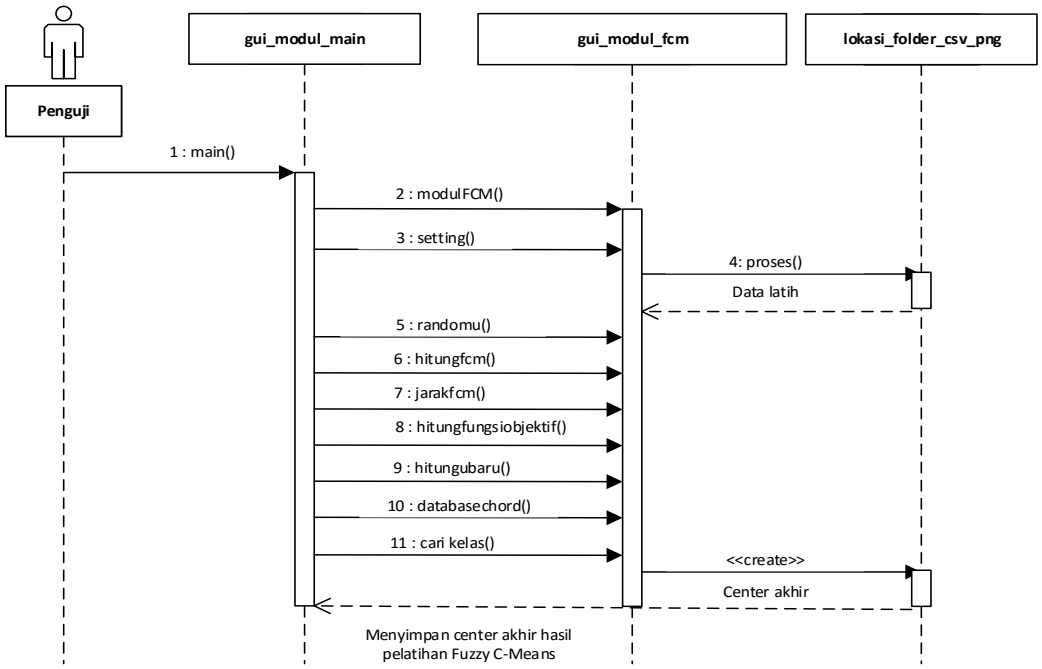
Gambar 3. 20 ***Sequence diagram Preprocessing Collecting Database***



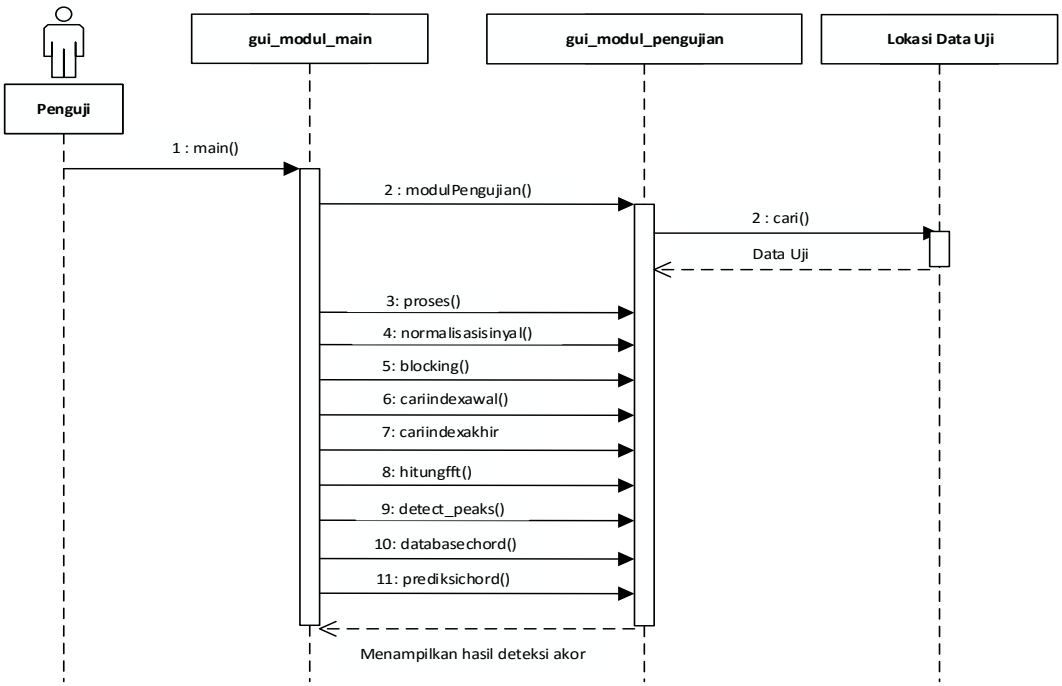
Gambar 3. 21 ***Sequence Diagram Preprocessing Chord Recognition***



Gambar 3. 22 ***Sequence diagram*** **Pelatihan** ***Fuzzy C-Means***



Gambar 3. 23 *Sequence Diagram* Menyimpan *Center* Akhir



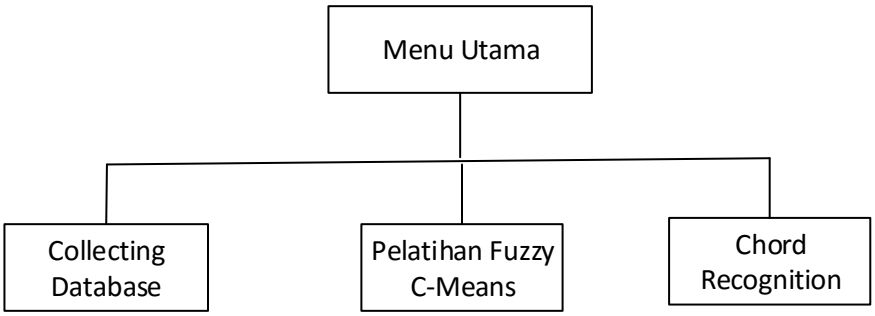
Gambar 3. 24 *Sequence Diagram* Chord Recognition

3.7 Perancangan Sistem

Perancangan sistem merupakan penggambaran dan perencanaan sistem untuk mengimplementasikan hasil analisis yang dilakukan sebelumnya. Perancangan sistem yang dibuat terdiri dari perancangan struktur menu, perancangan antarmuka, perancangan pesan dan jaringan semantik .

3.7.1 Perancangan Struktur Menu

Struktur menu merepresentasikan organisasi atau modul secara hirarki. Adapun struktur menu untuk perangkat lunak ini sebagai berikut.



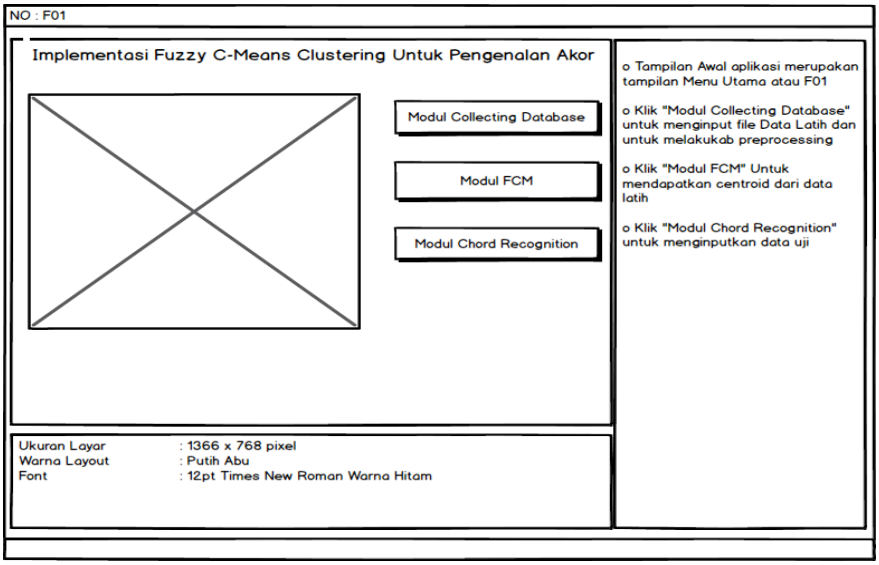
Gambar 3. 25 Perancangan Struktur Menu

3.7.2 Perancangan Antarmuka

Perancangan antarmuka mendeskripsikan rencana tampilan yang akan digunakan pada aplikasi yang akan dibangun. Berikut adalah perancangan antarmuka yang akan dibangun :

1. F01 – Menu Utama

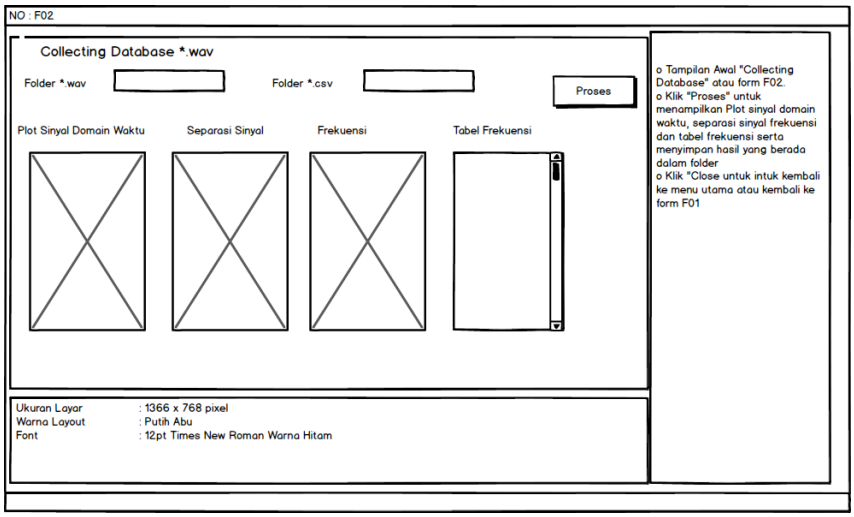
Perancangan F01 merupakan halaman utama pada saat pertama kali menjalankan program. Perancangan F01 dapat dilihat pada gambar berikut.



Gambar 3. 26 F01 Menu Utama

2. F02 –*Collecting Database*

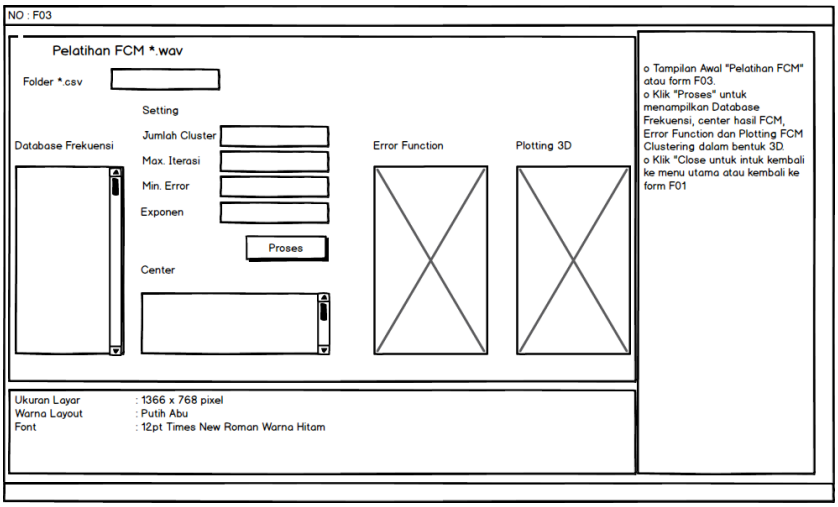
Perancangan F02 merupakan halaman untuk melakukan input data latih yang akan melalui *preprocessing*. Perancangan F02 dapat dilihat pada gambar berikut.



Gambar 3. 27 F02 *Collecting Database*

3. F03 – Pelatihan *Fuzzy C-Means*

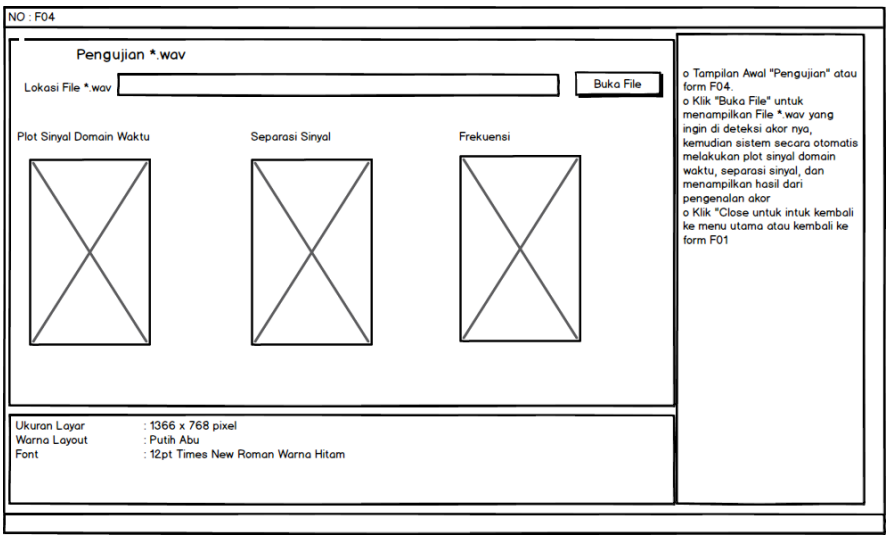
Perancangan F03 merupakan form pelatihan data latih menggunakan *Fuzzy c-means*. Perancangan F03 dapat dilihat pada gambar berikut.



Gambar 3. 28 F03 Pelatihan *Fuzzy C-Means*

4. F04 – *Chord Recognition*

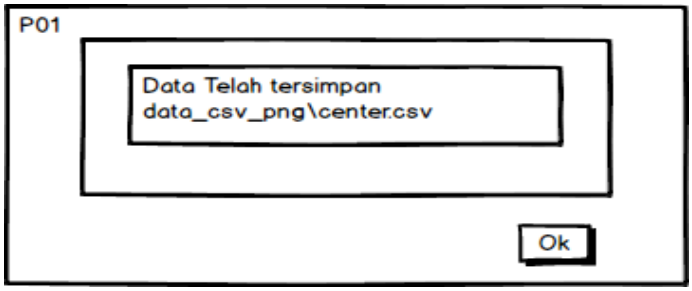
Perancangan F04 merupakan halaman untuk melakukan proses Input Data latih. Perancangan F04 dapat dilihat pada gambar berikut.



Gambar 3. 29 F04 *Chord Recognition*

3.7.3 Perancangan Pesan

Perancangan pesan merupakan validasi dan pesan yang akan disampaikan oleh dalam keadaan-keadaan tertentu. Adapun perancangan pesan dapat dilihat pada gambar 3.30 berikut.

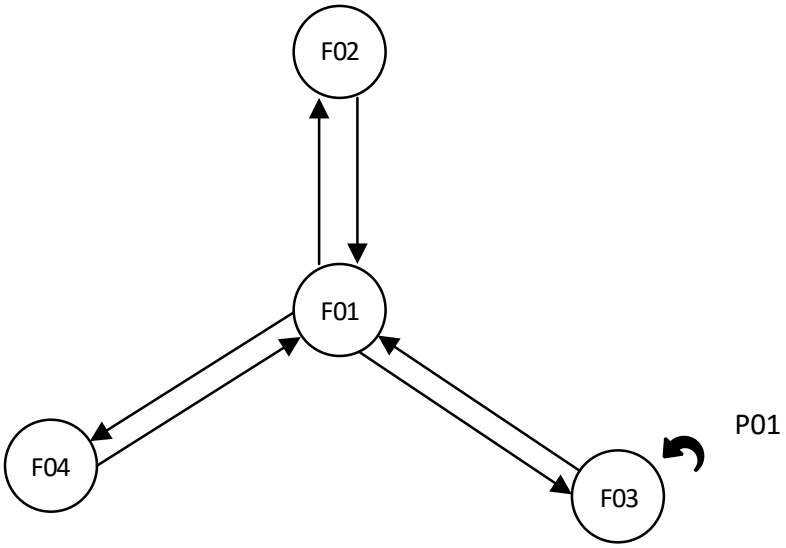


Gambar 3. 30 Penyimpanan Center FCM

Pesan pada gambar 3.30 muncul apabila data latih telah tersimpan.

3.7.4 Jaringan Semantik

Jaringan semantik menggambarkan keterhubungan navigasi menu dari satu antarmuka ke antarmuka lain. Jaringan semantik yang terbentuk pada sistem ini dapat dilihat pada gambar 3.31 berikut:



Gambar 3. 31 Jaringan Semantik