BAB 3

ANALISIS DAN PERANCANGAN

3.1 Analisis Metode

Analisis metode bertujuan untuk mendapatkan hasil analisis yang sesuai antara bidang ilmu yang diteliti dan kebutuhan dalam pembangunan aplikasi. Hasil analisis ini berupa hasil uraian dari suatu informasi yang utuh ke dalam bagian-bagian komponen yang nantinya akan menjadi masukan bagi proses perancangan.

Arsitektur global dalam pembangunan aplikasi ini adalah dimulai dari tahap menerima input dari *user* berupa kalimat bahasa Arab tanpa harakat, kemudian dilakukan proses pemberian harakat pada setiap kata, setelah itu menghasilkan output berupa kalimat bahasa Arab yang telah diharakati.

3.1.1 Analisis Masalah

Berdasarkan hasil penelitian mengenai cara membaca kalimat bahasa Arab tanpa harakat yang dilakukan terhadap para santri di PPM *Miftahul Khoir* dan beberapa sumber lainnya , maka masalah yang dihadapi dalam membaca kalimat bahasa Arab tanpa harakat adalah sebagai berikut :

Gramatika bahasa Arab terkenal dengan kerumitannya dan untuk mempelajari secara teoritis butuh waktu yang lama, sehingga banyak dari para pelajar Islam yang kurang semangat dalam mempelajarinya.

3.1.2 Analisis Penyelesaian Masalah

Berdasarkan permasalahan-permasalahan yang telah dijabarkan di atas serta sebagai modal awal simplikasi untuk mempelajari ilmu dasar keislaman, akan dibangun aplikasi pemberian harakat pada kalimat bahasa Arab berbasis *web*. Aplikasi ini dapat menangani pemberian harakat pada kalimat sederhana bahasa Arab berdasarkan gramatikanya, dan memberikan informasi mengenai kedudukan setiap kata pada suatu kalimat serta memberikan informasi kata asal dari setiap perubahan kata.

Pada proses pemberian harakat, aplikasi ini menggunakan metode pencocokan string dengan algoritma *levenshtein distance* yang akan menyelesaikan sebagian permasalahan pada penentuan bentuk kata dan metode klasifikasi menggunakan algoritma *bayesian classification* yang akan menyelesaikan permasalahan pada penerapan ilmu *Nahwu*.

3.1.3 Analisis Data Masukan

Pada aplikasi pemberian harakat ini dibutuhkan data masukan berupa kalimat bahasa Arab yang tersusun dari beberapa kata yang terdiri dari huruf-huruf *hijaiyyah* (ا, ب, ت, ث dan seterusnya) tanpa harakat. Dalam kaidah bahasa Arab setiap kata dikelompokkan menjadi beberapa kategori berdasarkan fungsinya masing-masing.

Pada penelitian ini sebelum masuk kepada tahap klasifikasi kata berdasarkan gramatikanya dalam suatu kalimat, akan dilakukan pengkategorian kosakata dengan mencocokkan kata masukan dengan kata yang terdapat di dalam basis data. Sebagai contoh berikut tabel 3.1 tentang bentuk kata dan kategorinya.

Tabel 3.1 Kategori Kata

No	Bentuk Kata	Kategori Kata
1	من, الى, عن, على, فى	*Huruf Jar/Khofadz*
2	هو, هما, هم, هي, هنّ, انت, انتما, انتم,انا,نحن	*Isim Dhomir* (kata ganti)
3	هذا, هذه, هؤلاء, ذلك, تلك	*Isim Isyaroh* (kata tunjuk)
4	الّذى, الّذين, الّتى, اللّاتى	*Isim Maushul* (kata penghubung)
5	ان, لن, كي,حتّي	*Amil Nawasib*
6	لم, لمّا, الم, المّا, إن	*Amil Jawazim*
7	كان, اصبح, ليس,امسى	*Kaana* dan kawanannya

No	Bentuk Kata	Kategori Kata
8	انّ, أنّ, لعّل, ليت, لكنّ	*Inna* dan kawanannya
9	استفعل, فعل, افعل	*Fi'il madhi*
10	يستفعل ,يفعّل ,يفعل	*Fi'il mudhore*
11	آستفعل ,آفعل	*Fi'il amr*
12	البيت ,الرجل	*Isim* dengan *alif lam*
13	مدرسة ,مسجد	*Isim* tanpa *alif lam*

3.1.4 Analisis Algoritma Pada Proses Pemberian Harakat

Proses pemberian harakat pada sistem ini terbagi menjadi beberapa tahapan proses, adapun tahap-tahap proses yang akan dilakukan adalah sebagai berikut :

1. Proses Pemecahan kalimat

Pada proses ini dilakukan pemecahan kata per kata dari suatu kalimat dan disimpan dalam sebuah variable *array*. Berikut proses pemecahan kata per kata menggunakan fungsi yang telah disediakan oleh PHP.

```
$kalimat  = " يكون زيد في المكتبة";
$pecah = explode(" ", $kalimat);
```

Setelah kalimat dipecah menjadi perkata, selanjutnya sebelum memasuki tahap identifikasi kata akan melewati validasi kata yang berawalan dengan *alif lam*, berikut contoh penerapannya :

```
//apabila kalimat menggunakan alif lam
     if(substr($pisah[$i],0,14)=='&#1575;&#1604;'){
          $kata=substr($pisah[$i],14);
          $asal[$i]=aslu($kata);
          $kalimat=levi($kata);
          $kata=$kalimat;
          $kata=substr($kata,2);
          $pisah[$i]='al&#1575;&#1604;'.$kata;
     }
     //kalimat tanpa alif lam
     else {
          $asal[$i]=aslu($pisah[$i]);
          $kalimat=levi($pisah[$i]);
          $pisah[$i]=$kalimat;
     }
```

Karena kode *unicode* karakter Arab setelah diproses akan dikonfersi ke dalam bentuk desimal, jadi penerapan pada koding karakter Arab dipresentasikan dengan angka desimal.

2. Identifikasi Kata

Setelah proses pemecahan kalimat menjadi per kata ke dalam variabel *array*, langkah selanjutnya adalah melakukan proses identifikasi kata. Dalam proses ini setiap kata akan diidentifikasi dengan diberikan kode unik di awal kata yang sesuai dengan identitas masing-masing. Berikut kode unik yang telah disusun berdasarkan analisa pada kalimat-kalimat sederhana.

Tabel 3.2 Kode Identifikasi Kata

No	Identitas Kata	Kode	No	Identitas Kata	Kode
1	*Huruf jar/khofadz*	Hj	13	*Fi'il amr*	fa
2	*Isim dhomir*	Id	14	*Isim* dengan *alif lam*	al
3	*Isim isyaroh*	Ii	15	*Isim* tanpa *alif lam*	no
4	*Isim maushul*	Im			

No	Identitas Kata	Kode	No	Identitas Kata	Kode
5	*Amil nawasib*	An			
6	*Amil jawazim*	Aj			
7	*Amil kaana*	Kn			
8	*Amil inna*	In			
9	*Fi'il madhi*	Fm			
10	*Fi'il madhi* disertai *dhomir (fa'il)*	Fd			
11	*Fi'il mudhore*	Fr			
12	*Fi'il mudhore mukhathab*	Mm			

Kemudian setiap kata akan diberikan harakat sesuai bentuk kata yang teridentifikasi. Pada proses identifikasi ini dilakukan proses pencocokan bentuk kata masukan dengan bentuk kata yang telah disediakan di dalam basis data menggunakan fungsi algoritma *levenstein distance*.

Akibat kaidah bahasa Arab yang sedemikian kompleks tidak semua identifikasi dapat dilakukan menggunakan algoritma *levenstein distance*. Pada beberapa jenis kata seperti *fi'il*(kata kerja) dan kata yang mengalami perubahan dari bentuk asalnya akan diidentifikasi berdasarkan pola kata masing-masing, identifikasi dilakukan menggunakan fungsi *regular expression functions* yang terdapat pada pemrograman PHP. Berikut ini contoh beberapa jenis kata yang diidentifkasi berdasarkan pola kata masing-masing beserta makna umumnya :

Tabel 3.3 Jenis Kata Berdasarkan Pola

No	Bentuk Kata	Pola Kata	Jenis	Makna
1	أفعل	أ…	*Fi'il Madhi*	Telah me…kan
2	فعّل	..ّ.	*Fi'il Madhi*	Telah me…kan
3	تفاعل	ت..ا..	*Fi'il Madhi*	Telah saling …
4	إفتعل	إ..ت..	*Fi'il Madhi*	Telah ter…

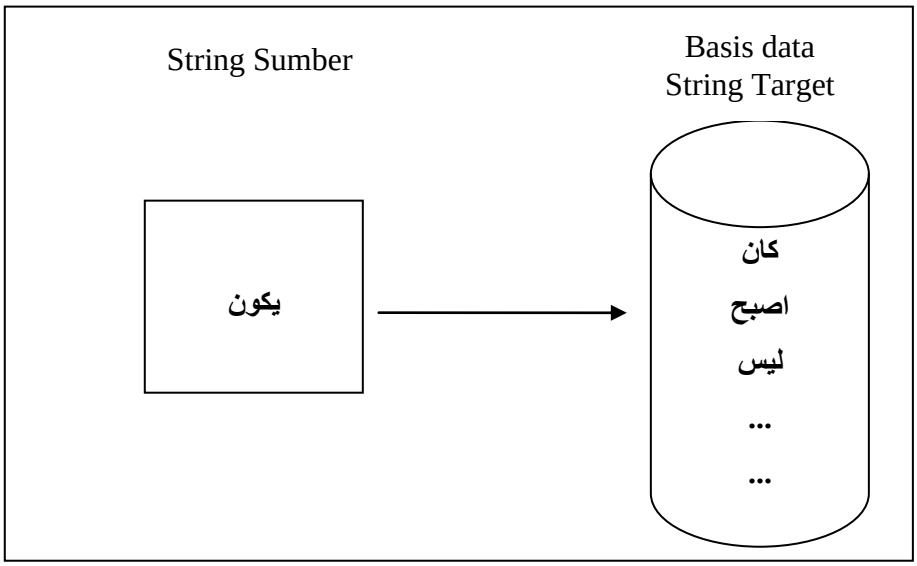
No	Bentuk Kata	Pola Kata	Jenis	Makna
5	إنفعل	إن...	*Fi'il Madhi*	Telah ter…
6	تفعّل	ت..ّ.	*Fi'il Madhi*	Telah ter/me…
7	استفعل	است...	*Fi'il Madhi*	Telah minta …
8	مفعلة	م...ة	*Isim makan*	Makna tempat
9	فعول	..و.	*Isim fa'il*	Makna subjek
10	مفعول	م..و.	*Isim maf'ul*	Yang di…
11	فاعل	..ا.	*Isim fa'il*	Yang me…
12	يفعل	ي...	*Fi'il mudhore*	Sedang me…
13	يفتعل	ي..ت.	*Fi'il mudhore*	Sedang ter…
14	افتعال	ا.ت.ا.	*Isim masdar*	Makna sifat
15	تفعيل	ت..ي.	*Isim masdar*	Makna sifat

3. Proses Pencocokkan Kata Menggunakan Algoritma *Levenstein Distance*

Selanjutnya akan dilakukan analisis bagaimana penerapan algoritma *levenshtein distance* dalam pencocokkan pola kata.

Misalkan terdapat kalimat masukan "*يكون زيد في المكتبة*", setelah melewati proses pemecahan kata per kata dan disimpan dalam suatu variabel *array* sebagai berikut :

$pecah[0] = "*يكون*", $pecah[1] =" *زيد*", $pecah[2] =" *في*", $pecah[3] =" *المكتبة*". Kemudian untuk mengetahui jarak kedekatan antara bentuk kata pada *array* masukan (string sumber) dengan bentuk kata pada *array* yang tersedia di dalam basis data (string target) dilakukan proses algoritma *levenshtein distance* terhadap karakter kata dari *array* masukan dengan karakter kata dari *array* yang tersedia di dalam basis data.



Gambar 3.1 Proses Pencocokkan kata

String sumber ($pecah[0]) = "***يكون***"

String target = "***كان***"

Proses algoritma *levenshtein distance* adalah sebagai berikut :

1) Buat matriks s[x], t[y] dengan ukuran s[x] = (sumber+1) X t[y] = (target+1).
2) Inisialisasi s[x] pertama (0..n).
3) Inisialisasi t[y] pertama (0..m).

		ي	ك	و	ن
	0	1	2	3	4
ك	1				
ا	2				
ن	3				

Gambar 3.2 Inisialisasi Matriks (n+1) x (m+1)

4) Melakukan proses pencocokan dengan membandingkan setiap karakter pada string target dengan setiap karakter pada string sumber.
5) Jika nilai n = 0 maka nilai cell [n,m] = m.
6) Jika nilai m = 0 maka nilai cell [n,m] = n.
7) Jika karakter s[x] = t[y] maka beri nilai 0.
8) Jika karakter s[x] ≠ t[y] maka beri nilai 1.
9) Tentukan nilai *cell* [x , y] dengan masukkan nilai terkecil dari :
   a. Nilai *cell* [x-1, y-1] + nilai
   b. Nilai *cell* [x-1, y] + 1
   c. Nilai *cell* [x, y-1] + 1

		ي	ك	و	ن
	0	1	2	3	4
ك	1	1			
ا	2	2			
ن	3	3			

Gambar 3.3 Matriks Proses *Levenstein Distance* Iterasi (1,1),(1,2),(1,3)

		ي	ك	و	ن
	0	1	2	3	4
ك	1	1	1		
ا	2	2	2		
ن	3	3	3		

Gambar 3.4 Matriks Proses *Levenstein Distance* Iterasi (2,1),(2,2),(2,3)

		ي	ك	و	ن
	0	1	2	3	4
ك	1	1	1	2	
ا	2	2	2	2	
ن	3	3	3	3	

Gambar 3.5 Matriks Proses *Levenstein Distance* Iterasi (3,1),(3,2),(3,3)

		ي	ك	و	ن
	0	1	2	3	4
ك	1	1	1	2	3
ا	2	2	2	2	3
ن	3	3	3	3	2

Gambar 3.6 Matriks Proses *Levenstein Distance* Iterasi (4,1),(4,2),(4,3)

Hingga tahap ini dapat disimpulkan bahwa jarak antara string sumber yang terdapat pada variabel $pecah[0] mendekati string target yang merupakan jenis kata *amil kaana* dengan jarak 2. Kemudian pada string sumber $pecah[2] =" **في** " memiliki jarak 0 dengan salah satu string target yang terdapat pada basis data yang merupakan jenis kata *huruf jar* karena kesamaan pola dan setiap karakternya. Sedangkan pada string sumber $pecah[1] =" **زيد** " dan $pecah[3] =" **المكتبة** " tidak memiliki kriteria kemiripan dengan salah satu string sumber yang terdapat pada data kata sehingga harus melalui proses *regular expression* untuk pemberian harakat secara umum, kode unik dan kata asal, berikut contoh penerapan fungsi *regular expression* :

```
$arr = /^(.{7})(.{7})(.{7})(.{7})(.{7})$/
$kata = مكتبة
$hr = $1◌$2◌$3◌$4◌$5◌
$kata= preg_replace($arr,$kode.$hr,$kata);
$kalimat = $kata;
```

Dalam bahasa pemrograman PHP telah tersedia fungsi *levenstein distance*, jadi dalam aplikasinya proses pencocokkan string dilakukan menggunakan fungsi tersebut.

Pseudocode algoritma *levenstein distance* :

```
Procedure levenshtein distance;
kamus
i, j, s1 , s2, m, cost : integer
algoritma
for i ← 0 to s1.length do
 for j ← 0 to s2.length do
  if i = 0 then
    m[i,j] ← j {perbandingan dengan kosong}
  else if j = 0 then
    m[i,j] ← i {perbandingan dengan kosong}
  else
  if s1[i] = s2[j] then
    cost ← 0
  else
    cost ← 1
m[i,j] = minimum (
m[i-1, j-1] + cost, {substitusi}
m[i-1,j] + 1, {penghapusan}
m[i ,j-1] + 1, {penambahan}
)
return m[s1.length, s2.length]
```

4. Proses Pemberian Harakat Akhir Menggunakan Algoritma *Bayesian Classification*

Setelah semua kata memiliki kode unik dan telah diharakati secara umum berdasarkan identitas kata masing-masing, tahap selanjutnya yang akan dilalui adalah pemberian harakat akhir kata sesuai dengan gramatika setiap kata pada

suatu kalimat. Pada tahap ini akan digunakan metode *bayesian classification* untuk mengklasifikasi gramatika setiap kata berdasarkan parameter tertentu yang terdapat pada masing-masing kata. Metode ini terdiri dari dua proses, yaitu proses pembelajaran dan proses klasifikasi.

a. Proses Pembelajaran Algoritma *Bayesian Classification*

Sebelum melakukan proses klasifikasi akan dilakukan pembelajaran terhadap aplikasi dengan memasukkan data latihan berupa kode unik pada setiap kata yang didapatkan melalui tahap identifikasi. Kemudian menentukan kelas masing-masing berdasarkan hasil hipotesis terhadap kalimat bahasa Arab sederhana. Berikut ini keterangan mengenai kode unik yang akan dijadikan data latihan :

1. KODE = kode yang terdapat pada kata itu sendiri
2. KD1 = kode yang terdapat pada kata setelahnya
3. KD2 = kode yang terdapat pada kata sebelumnya
4. KD3 = kode yang terdapat pada kata dua langkah sebelumnya.

Berikut data latihan yang sudah diklasifikasi menjadi 11 kelas :

Tabel 3.4 Data Latihan

ID	KODE	KD1	KD2	KD3	Kelas : Gramatika
1	Al	Al	fr	An	*Fa'il*
2	Al	Hj	fr	Aj	*Fa'il*
3	Al	0	fr	0	*Fa'il*
4	No	0	hj	Al	*Isim majrur*
5	Al	0	no	Kn	*Khobar amil kaana*
6	No	0	al	0	*Khobar*
7	Al	No	0	0	*Mubtada*
8	No	No	fm	0	*Fa'il*
9	No	0	no	Fm	*Maf'ul*
10	Al	0	hj	No	*Isim majrur*

ID	KODE	KD1	KD2	KD3	Kelas : Gramatika
11	Al	0	no	In	*Khobar amil inna*
12	Al	0	fd	0	*Maf'ul*
13	No	0	Fd	0	*Maf'ul*
14	Al	0	Hj	Fd	*Isim majrur*
15	Al	0	No	Fm	*Maf'ul*
16	Al	Hj	0	0	*Mubtada*
17	No	No	0	0	*Mubtada*
18	No	0	Hj	Mm	*Isim majrur*
19	Al	No	Hj	0	*Isim majrur*
20	Al	0	Fa	0	*Maf'ul*
21	No	0	Fa	0	*Maf'ul*
22	Fr	Al	An	Fr	*Fi'il mansub*
23	Fr	No	Aj	0	*Fi'il majzum*
24	Id	No	0	0	*Mubtada*
25	Fr	No	An	0	*Fi'il mansub*
26	Al	Al	Kn	0	*Isim amil kaana*
27	No	0	Al	Kn	*Khobar amil kaana*
28	Fr	Al	Aj	0	*Fi'il majzum*
29	No	Al	In	0	*Isim amil inna*
30	No	0	Al	In	*Khobar amil inna*
31	No	Hj	No	0	*Khobar*
32	No	0	Id	0	*Khobar*
33	Al	0	Mm	0	*Maf'ul*
34	Al	0	Al	Fr	*Maf'ul*
35	No	No	Kn	0	*Isim amil kaana*
36	Al	No	In	0	*Isim amil inna*
37	Al	Hj	Kn	0	*Isim amil kaana*

ID	KODE	KD1	KD2	KD3	Kelas : Gramatika
38	No	Hj	In	0	*Isim amil inna*

Pada setiap kelas akan ditentukan nilai P(Ci) berdasarkan data latihan, untuk i=1.. n.

P(Ci) = nilai *prior probability* untuk setiap kelas berdasarkan data latihan.

Berikut ini adalah daftar kelas gramatika dasar yang akan diterapkan di dalam aplikasi :

Tabel 3.5 Nilai *Prior Probability* untuk setiap kelas

No	P(Ci)	Nilai P(Ci)
1	P(gramatika=”*Mubtada*”)	4/38 = 0.105
2	P(gramatika=”*Khobar*”)	3/38 = 0.078
3	P(gramatika=”*Fa’il*”)	4/38 = 0.105
4	P(gramatika=”*Maf’ul*”)	8/38 = 0.210
5	P(gramatika=”*Isim amil kaana*”)	3/38 = 0.078
6	P(gramatika=”*Khobar amil kaana*”)	2/38 = 0.053
7	P(gramatika=”*isim amil inna*”)	3/38 = 0.078
8	P(gramatika=”*Khobar amil inna*”)	2/38 = 0.053
9	P(gramatika=”*Isim Majrur*”)	5/38 = 0.131
10	P(gramatika=”*Fi’il Mansub*”)	2/38 = 0.053
11	P(gramatika=”*Fi’il Majzum*”)	2/38 = 0.053

b. Proses Klasifikasi Algoritma *Bayesian Classification*

Pada analisis proses klasifikasi digunakan data tes. Data tes merupakan data masukan yang belum diklasifikasi. Data tes adalah data yang telah melewati proses pemberian kode unik dan telah diberi harakat sesuai bentuk kata.

Sebelum data tes memasuki tahap klasifikasi menggunakan algoritma *bayesian classification*, data tes akan diperiksa menggunakan percabangan dengan kondisi-kondisi sebagai berikut :

1. Untuk kata yang memiliki kode unik berupa kode-kode untuk *amil* dan *fi'il* diberikan kedudukan gramatika berdasarkan nama kategori kata pada kode unik yang dimilikinya.
2. Kata yang memiliki data tes sama dengan data latihan diberikan kedudukan gramatika sesuai kelas data latihan yang sama.
3. Kata yang memiliki data tes berbeda dengan data latihan dan kode unik yang bukan berupa kode untuk *amil* dan *fi'il*, akan melakukan klasifikasi terhadap kelas yang tersedia dengan menggunakan perhitungan :

$$P(X \mid C_i) = \prod_{k=1}^{n} P(X_k \mid C_i)$$

Sebagai contoh, kalimat masukan "***يكون زيد في المكتبة***", setelah melewati proses pemberian kode unik dan harakat sesuai bentuk kata sebagai berikut :

$pecah[0] = "**يَكُوْنُkn**", $pecah[1] =" **زَيْدٌno**", $pecah[2] =" **فِيhj**", $pecah[3] =" **المَكْتَبَةُal**".

Pada kata dalam variabel "$pecah[0]" memiliki kode "kn" yang merupakan kode unik untuk "*amil kaana*" dan kata dalam variabel "$pecah[2]" memiliki kode "hj" yang merupakan kode unik untuk "*huruf jar*". Keduanya merupakan *amil* yang merubah harakat akhir kata setelahnya, jadi keduanya tidak perlu melewati proses klasifikasi dan langsung diberikan kedudukan gramatika sesuai nama *amil* tersebut.

Selanjutnya kata dalam variabel "$pecah[3]" memiliki data sebagai berikut : KODE="al" KD1="0" KD2="hj" KD3="no" . apabila dicocokkan data tersebut sama dengan data latihan dengan ID="10", jadi kata tersebut tidak perlu melewati proses klasifikasi dan langsung diberikan kedudukan gramatika sesuai dengan gramatika data latihan pada ID="10".

Selanjutnya kata dalam variabel "$pecah[1]" memiliki data sebagai berikut : KODE="no" KD1="hj" KD2="kn" KD3="0". Kata tersebut belum diketahui

kelasnya, sehingga harus melalui tahap klasifikasi menggunakan algoritma *bayesian classification*, proses klasifikasi kata dalam variabel "$pecah[1]" adalah sebagai berikut :

1) Diketahui data X = (KODE="no" KD1="hj" KD2="kn" KD3="0")
2) Memaksimalkan P(X | Ci) P(Ci) untuk i= 1..11. P(Ci) merupakan *prior probability* untuk setiap kelas berdasarkan data pada tabel 3.5.
   Untuk mencari nilai P(X | Ci) dapat menggunakan pendekatan sebagai berikut :

$$P(X \mid C_i) = \prod_{k=1}^{n} P(X_k \mid C_i)$$

3) Hitung P(X=KODE | Ci), untuk i= 1..11
   - P(KODE="no" | gramatika="mubtada") = 1/4 = 0,25
   - P(KODE="no" | gramatika="khobar") = 3/3 = 1
   - P(KODE="no" | gramatika="fa'il") = 1/4 = 0,25
   - P(KODE="no" | gramatika="maf'ul") = 3/8 = 0,375
   - P(KODE="no" | gramatika="isim amil kaana") = 1/3 = 0,33
   - P(KODE="no" | gramatika="khobar amil kaana") = 1/2 = 0,5
   - P(KODE="no" | gramatika="isim amil inna") = 2/3 = 0,66
   - P(KODE="no" | gramatika="khobar amil inna") = 1/2 = 0,5
   - P(KODE="no" | gramatika="isim majrur") = 2/5 = 0,4
   - P(KODE="no" | gramatika="fi'il mansub") = 0/2 = 0
   - P(KODE="no" | gramatika="fi'il majzum") = 0/2 = 0
4) Hitung P(X=KD1 | Ci), untuk i= 1..11
   - P(KD1="hj" | gramatika="mubtada") = 1/4 = 0,25
   - P(KD1="hj" | gramatika="khobar") = 1/3 = 0,33
   - P(KD1="hj" | gramatika="fa'il") = 1/4 = 0,25
   - P(KD1="hj" | gramatika="maf'ul") = 0/8 = 0
   - P(KD1="hj" | gramatika="isim amil kaana") = 1/3 = 0,33
   - P(KD1="hj" | gramatika="khobar amil kaana") = 0/2 = 0
   - P(KD1="hj" | gramatika="isim amil inna") = 1/3 = 0,33
   - P(KD1="hj" | gramatika="khobar amil inna") = 0/2 = 0

- P(KD1=”hj” │gramatika=”isim majrur”) = 0/5 = 0
- P(KD1=”hj” │gramatika=”fi’il mansub”) = 0/2 = 0
- P(KD1=”hj” │gramatika=”fi’il majzum”) = 0/2 = 0

5) Hitung P(X=KD2 │Ci), untuk i= 1..11
   - P(KD2=”kn” │gramatika=”mubtada”) = 0/4 = 0
   - P(KD2=”kn” │gramatika=”khobar”) = 0/3 = 0
   - P(KD2=”kn” │gramatika=”fa’il”) = 0/4 = 0
   - P(KD2=”kn” │gramatika=”maf’ul”) = 0/8 = 0
   - P(KD2=”kn” │gramatika=”isim amil kaana”) = 3/3 = 1
   - P(KD2=”kn” │gramatika=”khobar amil kaana”) = 0/2 = 0
   - P(KD2=”kn” │gramatika=”isim amil inna”) = 0/3 = 0
   - P(KD2=”kn” │gramatika=”khobar amil inna”) = 0/2 = 0
   - P(KD2=”kn” │gramatika=”isim majrur”) = 0/5 = 0
   - P(KD2=”kn” │gramatika=”fi’il mansub”) = 0/2 = 0
   - P(KD2=”kn” │gramatika=”fi’il majzum”) = 0/2 = 0
6) Hitung P(X=KD3 │Ci), untuk i= 1..11
   - P(KD3=”0” │gramatika=”mubtada”) = 4/4 = 1
   - P(KD3=”0” │gramatika=”khobar”) = 3/3 = 1
   - P(KD3=”0” │gramatika=”fa’il”) = 2/4 = 0,5
   - P(KD3=”0” │gramatika=”maf’ul”) = 5/8 = 0,625
   - P(KD3=”0” │gramatika=”isim amil kaana”) = 3/3 = 1
   - P(KD3=”0” │gramatika=”khobar amil kaana”) = 0/2 = 0
   - P(KD3=”0” │gramatika=”isim amil inna”) = 3/3 = 1
   - P(KD3=”0” │gramatika=”khobar amil inna”) = 0/2 = 0
   - P(KD3=”0” │gramatika=”isim majrur”) = 1/5 = 0,2
   - P(KD3=”0” │gramatika=”fi’il mansub”) = 1/2 = 0,5
   - P(KD3=”0” │gramatika=”fi’il majzum”) = 2/2 = 1
7) Hitung P(X │Ci), untuk i= 1..11
   - P(X │gramatika=”mubtada”) = 0,25 x 0,25 x 0 x 1 = 0
   - P(X │gramatika=”khobar”) = 1 x 0,33 x 0 x 1 = 0
   - P(X │gramatika=”fa’il”) = 0,25 x 0,25 x 0 x 0,5 = 0

- P(X | gramatika=”maf’ul”) = 0,375 x 0 x 0 x 0,625 = 0
- P(X | gramatika=”isim amil kaana”) = 0,33 x 0,33 x 1 x 1 = 0,1089
- P(X | gramatika=”khobar amil kaana”) = 0,5 x 0 x 0 x 0 = 0
- P(X | gramatika=”isim amil inna”) = 0,66 x 0,33 x 0 x 1 = 0
- P(X | gramatika=”khobar amil inna”) = 0,5 x 0 x 0 x 0 = 0
- P(X | gramatika=”isim majrur”) = 0,4 x 0 x 0 x 0,2 = 0
- P(X | gramatika=”fi’il mansub”) = 0 x 0 x 0 x 0,5 = 0
- P(X | gramatika=”fi’il majzum”) = 0 x 0 x 0 x 1 = 0

8) Setelah $P(X|C_i)$ maksimum diketahui maka *class* dari data sampel X dapat didekati dengan menghitung $P(X|C_i)*P(C_i)$. *Class* Ci dimana $P(X|C_i)*P(C_i)$ maksimum adalah *class* dari sampel X.

9) Hitung P(X | Ci) P(Ci), untuk i= 1..11
   - P(X | gramatika=”mubtada”) P(gramatika=”mubtada”) = 0 x 0,105 = 0
   - P(X | gramatika=”khobar”) P(gramatika=”khobar”) = 0 x 0,078 = 0
   - P(X | gramatika=”fa’il”) P(gramatika=”fa’il”) = 0 x 0,105 = 0
   - P(X | gramatika=”maf’ul”) P(gramatika=”maf’ul”) = 0 x 0,210 = 0
   - P(X | gramatika=”isim amil kaana”) P(gramatika=”isim amil kaana”) = 0,1089 x 0,078 = 0,0084
   - P(X | gramatika=”khobar amil kaana”) P(gramatika=”khobar amil kaana”) = 0 x 0,053 = 0
   - P(X | gramatika=”isim amil inna”) P(gramatika=”isim amil inna”) = 0 x 0,078 = 0
   - P(X | gramatika=”khobar amil inna”) P(gramatika=”khobar amil inna”) = 0 x 0,053 = 0
   - P(X | gramatika=”isim majrur”) P(gramatika=”isim majrur”) = 0 x 0,131= 0
   - P(X | gramatika=”fi’il mansub”) P(gramatika=”fi’il mansub”) = 0 x 0,053 = 0
   - P(X | gramatika=”fi’il majzum”) P(gramatika=”fi’il majzum”) = 0 x 0,053 = 0

10) Nilai terbesar dari P(X | Ci) P(Ci), untuk i= 1..11 adalah gramatika=”isim amil kaana”.

Kesimpulannya adalah bahwa kata “ **زَيْدٌ** ” di dalam kalimat “ **يكون زيد في المكتبة** “ mempunyai kedudukan dalam gramatika berupa “*isim amil kaana*”.

3.1.5 Analisis Keluaran

Dalam pembangunan aplikasi ini diharapkan dapat memberikan keluaran berupa kalimat bahasa Arab yang telah diharakati sesuai gramatikanya dan keterangan-keterangan berupa kata asal dari setiap kata yang mengalami perubahan bentuk, serta gramatika setiap kata pada suatu kalimat.

3.2 Analisis Kebutuhan Non Fungsional

Analisis kebutuhan non fungsional menggambarkan keadaan luar sistem yang akan digunakan untuk mengolah aplikasi ini, diantaranya adalah perangkat keras dan perangkat lunak.

3.2.1 Analisis Kebutuhan Perangkat Keras

Perangkat keras yang digunakan selama pembangunan aplikasi ini memiliki spesifikasi sebagai berikut :

1. *Processor* 2,27 GHz
2. *Hard Disk* 320 GB
3. *Memory* 2 GB
4. *Monitor*
5. *Keyboard*
6. *Mouse*

3.2.2 Analisis Kebutuhan Perangkat Lunak

Dalam pembangunan suatu aplikasi komputer perangkat lunak merupakan salah satu faktor yang dibutuhkan untuk merancang dan membuat aplikasi sehingga maksud dan tujuan dari pembangunan aplikasi tercapai. Adapun

perangkat lunak yang dibutuhkan untuk membangun aplikasi ini adalah sebagai berikut :

1. Sistem Operasi Windows 7
2. Wamp Server 2.0
3. MySQL 5.0.51b
4. phpMyAdmin 2.11.6
5. Notepad++ v5.9
6. Adobe Dreamweaver CS3
7. Browser Mozilla Firefox 15.0.1

3.2.3 Analisis Pengguna

Dalam menggunakan aplikasi ini dibutuhkan *user* yang telah terbiasa mengoperasikan komputer dan memiliki karakteristik sebagai berikut :

Tabel 3.6 Spesifikasi Pengguna

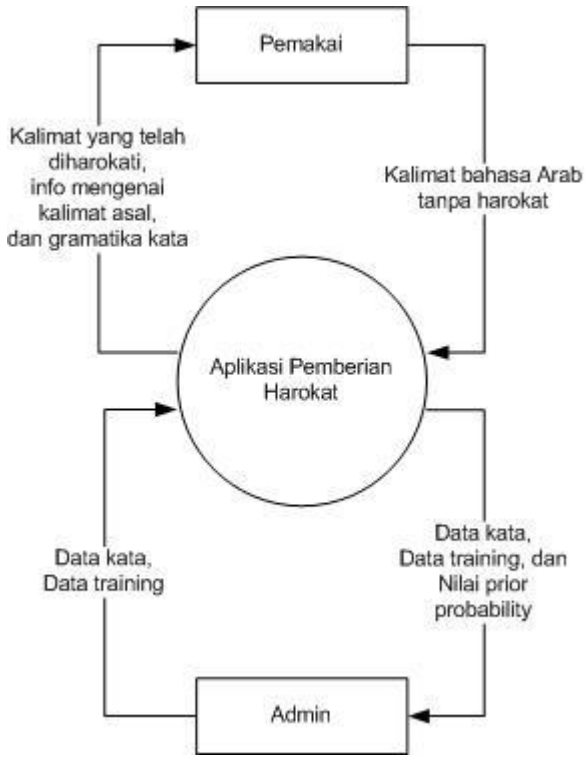
Pengguna	Keterampilan	Fisikal
Pengguna umum	- Dapat menggunakan aplikasi *web browser* - Sedang mempelajari bahasa Arab - Mampu membaca kalimat bahasa Arab dengan harakat	Normal (memiliki jari tangan lengkap, dapat melihat)
Admin	- Dapat menggunakan aplikasi *web browser* - Mengerti kaidah-kaidah dasar bahasa Arab	Normal (memiliki jari tangan lengkap, dapat melihat)

3.3 Analisis Kebutuhan Fungsional

Kebutuhan fungsional merupakan proses kegiatan yang dibutuhkan dalam merancang suatu sistem untuk menjelaskan kebutuhan yang diperlukan sistem agar dapat berjalan sesuai dengan konsep yang telah direncanakan. Pada pembangunan aplikasi ini sistem akan dimodelkan menggunakan model struktural.

3.3.1 Diagram Konteks

Diagram konteks adalah diagram yang terdiri dari suatu proses dan menggambarkan ruang lingkup suatu sistem. Perancangan ini melibatkan dua entitas luar, yaitu admin dan pemakai. Admin adalah entitas luar yang melakukan proses masukan berupa data kata dan data latihan. Pemakai adalah entitas luar yang memberikan masukan berupa kalimat bahasa Arab tanpa harakat yang akan diberikan harakat melalui proses yang ada pada aplikasi. Lebih jelasnya akan digambarkan dalam diagram konteks sebagai berikut :



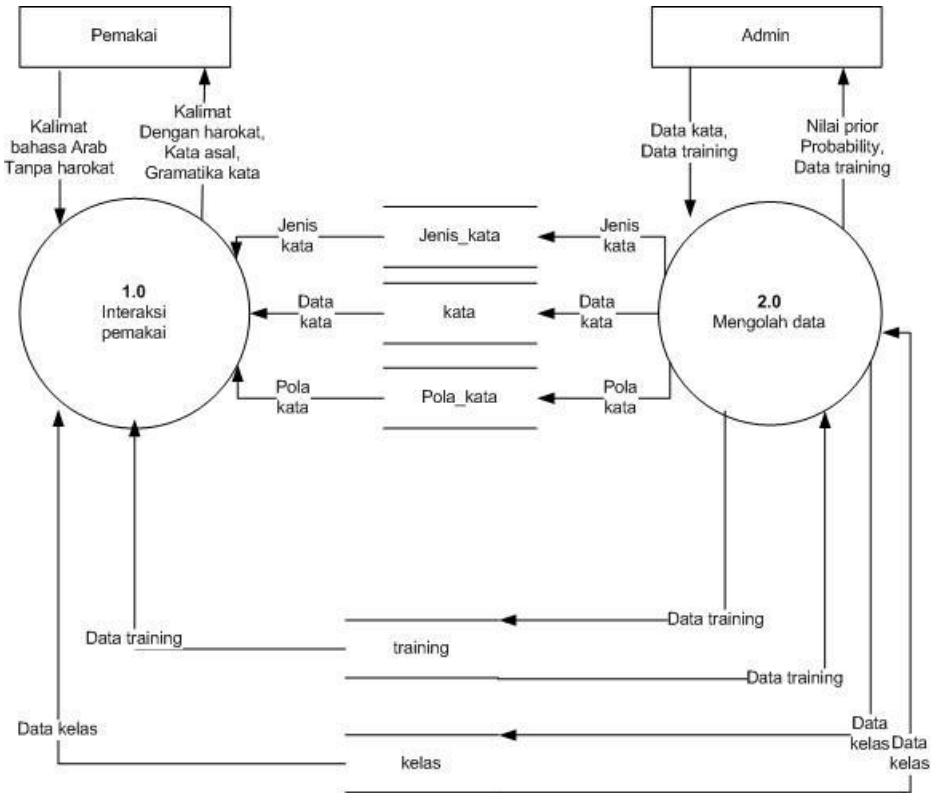
Gambar 3.7 Diagram Konteks pemberian harakat kalimat bahasa Arab

3.3.2 *Data Flow Diagram* (DFD)

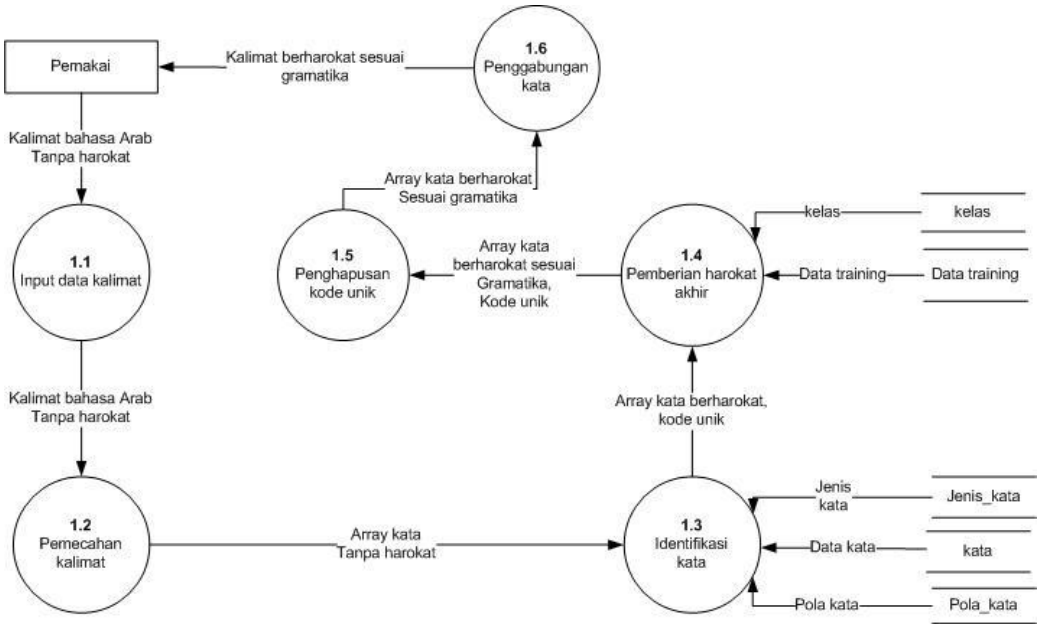
DFD merupakan model dari sistem untuk menggambarkan pembagian sistem ke modul yang lebih kecil. DFD juga merupakan alat yang digunakan

dalam menggambarkan arus data yang mengatur dari suatu proses ke proses selanjutnya.

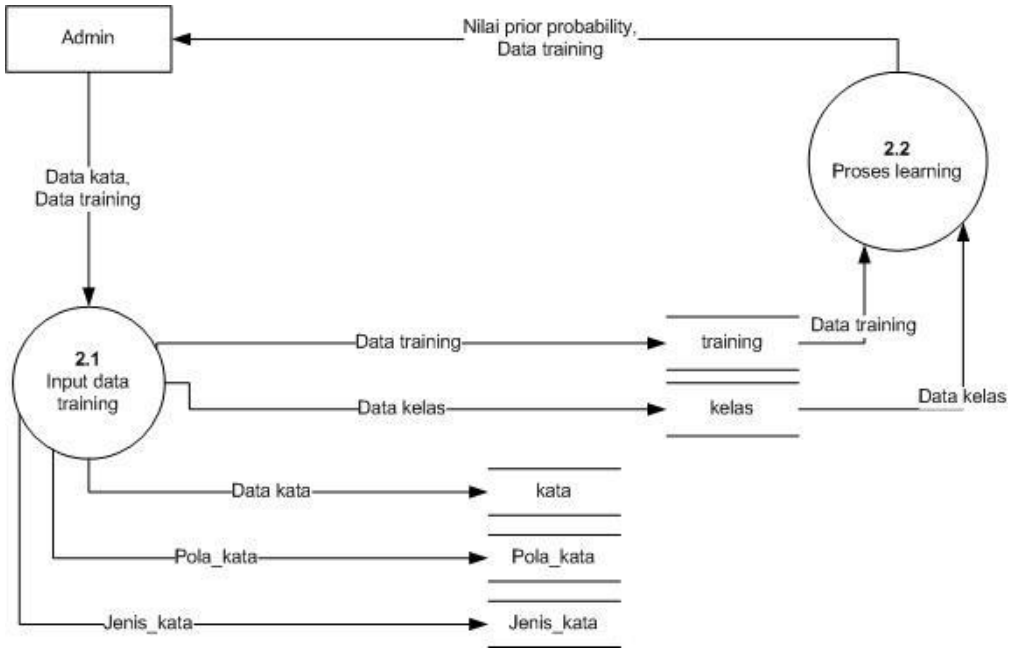
Dibawah ini merupakan DFD aplikasi pemberian harakat pada kalimat bahasa Arab yang akan dibuat, pada proses aplikasi ini memiliki dua interaksi eksternal yang akan mengoperasikan aplikasi, yaitu admin dan pemakai, dimana keduanya memiliki interaksi masing-masing terhadap aplikasi yang akan dibuat. Proses-proses utama dalam aplikasi ini akan berinteraksi dengan pemakai. Untuk lebih jelasnya dapat dilihat pada gambar DFD berikut ini :



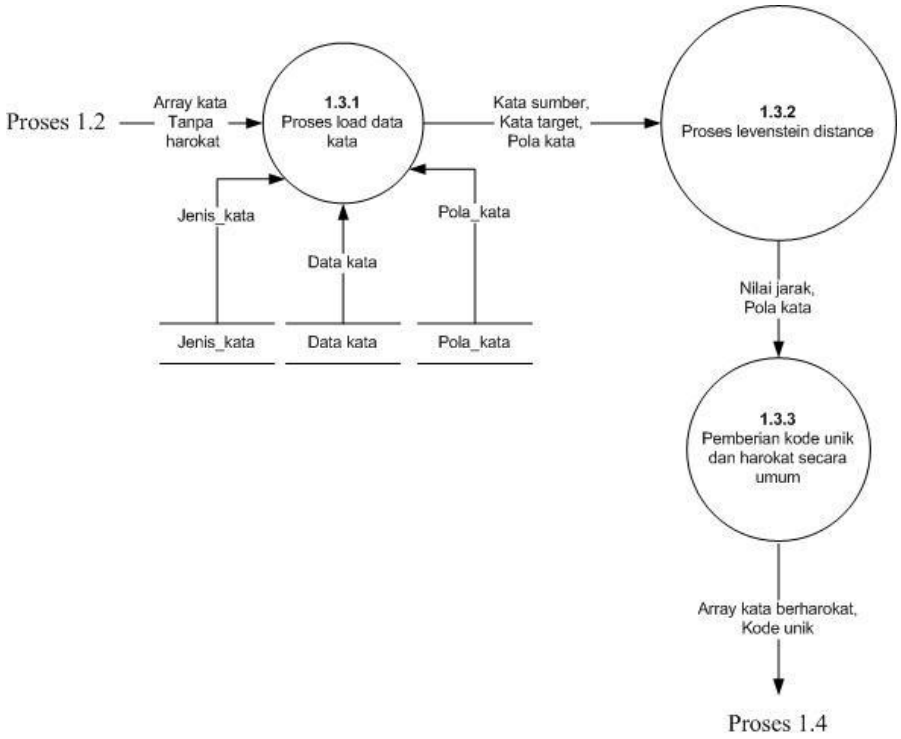
Gambar 3.8 DFD level 1 aktivitas pemberian harakat pada kalimat bahasa Arab



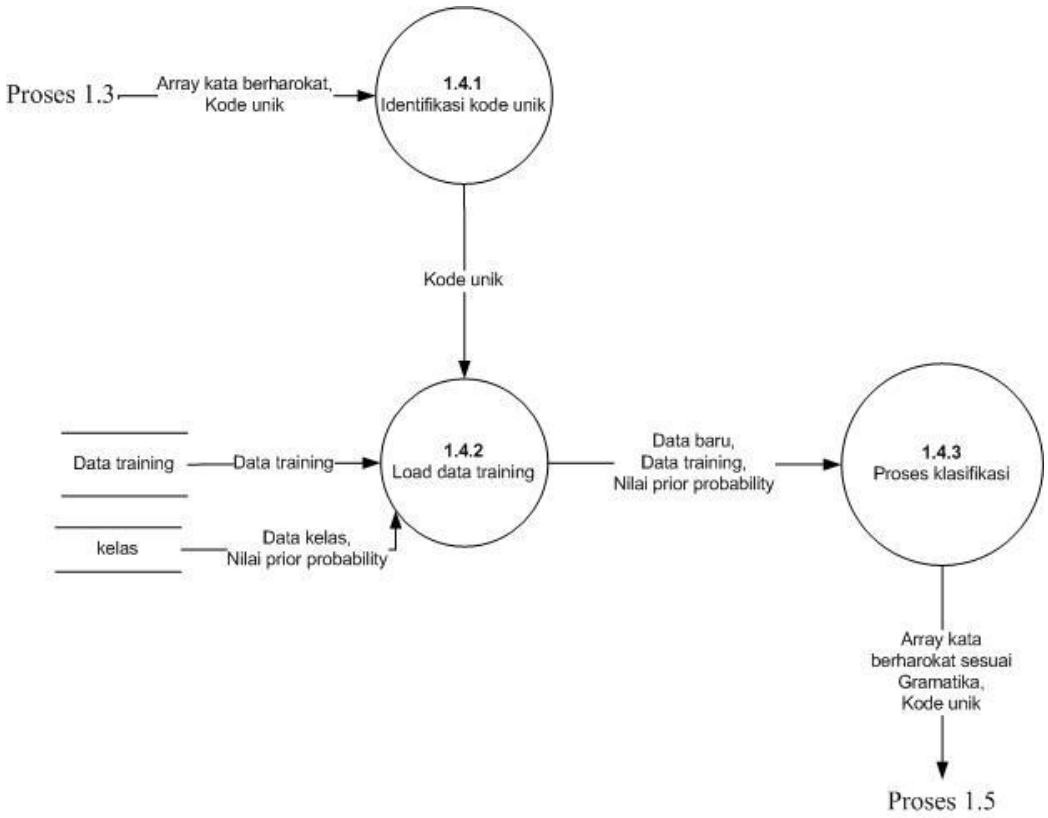
Gambar 3.9 DFD level 2 proses 1.0 interaksi pemakai



Gambar 3.10 DFD level 2 proses 2.0 interaksi admin



Gambar 3.11 DFD level 3 proses 1.3 identifikasi kata



Gambar 3.12 DFD level 3 proses 1.4 pemberian harakat akhir dan gramatika kata

3.3.3 Spesifikasi Proses

Proses-proses yang terdapat pada DFD diatas dapat dijelaskan dengan tabel spesifikasi proses berikut :

Tabel 3.7 Spesifikasi Proses

No	**Proses**	**Keterangan**
1.	No Proses	1.1
	Nama Proses	Input data kalimat
	Deskripsi	*User* memasukkan kalimat bahasa Arab tanpa harakat
	Source (Sumber)	Pemakai
	Input	Kalimat bahasa Arab tanpa harakat
	Output	Kalimat bahasa Arab tanpa harakat
	Destination (Tujuan)	Proses 1.2
	Logika Proses	*Begin* {User masukkan kalimat bahasa Arab tanpa harakat} If input kalimat then read kalimat *End*
No	**Proses**	**Keterangan**
2.	No Proses	1.2
	Nama Proses	Pemecahan kalimat
	Deskripsi	Kalimat dipecah menjadi bentuk array kata
	Source (Sumber)	Proses 1.1
	Input	Kalimat bahasa Arab tanpa harakat
	Output	*Array* kata tanpa harakat
	Destination (Tujuan)	Proses 1.3
	Logika Proses	*Begin*

		{read kalimat bahasa Arab tanpa harakat} If pecah kalimat then array kata *End*
No	**Proses**	**Keterangan**
3.	No Proses	1.3
	Nama Proses	Identifikasi kata
	Deskripsi	Mengidentifikasi bentuk kata dan memberikan kode unik sesuai dengan kategori kata
	Source (Sumber)	Proses 1.2
	Input	*Array* kata tanpa harakat, data kata
	Output	Array kata berharakat, kode unik
	Destination (Tujuan)	Proses 1.4
	Logika Proses	*Begin* {read *array* kata} *If array* kata sama dengan data target then beri kode unik dan harakat secara umum *Else* bandingkan dengan data pola *then* beri kode unik dan harakat secara umum *End*
No	**Proses**	**Keterangan**
4.	No Proses	1.4
	Nama Proses	Pemberian harakat akhir
	Deskripsi	Menentukan harakat akhir setiap kata berdasarkan gramatika masing-masing kata menggunakan algoritma *Bayesian classification*
	Source (Sumber)	Proses 1.3, data training
	Input	*Array* kata berharakat, kode unik
	Output	*Array* kata berharakat sesuai gramatika, kode unik
	Destination	Proses 1.5

	(Tujuan)	
	Logika Proses	*Begin* {read array kata berharakat secara umum, kode unik} *If* berikan harakat akhir menggunakan algoritma *bayesian classification then* array kata berharakat sesuai gramatika, kode unik *End*
No	**Proses**	**Keterangan**
5.	No Proses	1.5
	Nama Proses	Penghapusan kode unik
	Deskripsi	Menghapus kode unik yang terdapat di awal kata
	Source (Sumber)	Proses 1.4
	Input	*Array* kata berharakat sesuai gramatika, kode unik
	Output	*Array* kata berharakat sesuai gramatika
	Destination (Tujuan)	Proses 1.6
	Logika Proses	*Begin* {read array kata berharakat sesuai gramatika, kode unik} *If* hapus kode unik *then* array kata sesuai gramatika *End*
No	**Proses**	**Keterangan**
6.	No Proses	1.6
	Nama Proses	Penggabungan kata
	Deskripsi	Menggabungkan kembali kata yang semula dipecah menjadi *array* menjadi kalimat bahasa Arab berharakat
	Source (Sumber)	Proses 1.5
	Input	*Array* kata berharakat sesuai gramatika

	Output	Kalimat bahasa arab berharakat sesuai gramatika
	Destination (Tujuan)	Pemakai
	Logika Proses	*Begin* {*read* array kata berharakat sesuai gramatika} *If* gabungkan array kata *then* output kalimat bahasa Arab sesuai gramatika *End*
No	**Proses**	**Keterangan**
7.	No Proses	1.3.1
	Nama Proses	*Load* data kata
	Deskripsi	Me-*load* data kata dari *database*
	Source (Sumber)	Proses 1.2, *database*
	Input	*Array* kata tanpa harakat
	Output	Kata sumber, kata target(dari *database*), pola kata
	Destination (Tujuan)	Proses 1.3.2
	Logika Proses	*Begin* *Load* kata target, pola regex *from database* *End*
No	**Proses**	**Keterangan**
8.	No Proses	1.3.2
	Nama Proses	Proses *levenstein distance*
	Deskripsi	Menentukan nilai jarak kata sumber dengan kata target menggunakan fungsi *levenstein distance* dan/atau menentukan pola menggunakan fungsi *regular expression*
	Source (Sumber)	Proses 1.3.1

	Input	Kata sumber, kata target, pola kata
	Output	Nilai jarak, pola kata
	Destination (Tujuan)	Proses 1.3.3
	Logika Proses	*Begin* {*read* kata sumber, kata target, pola regex} Cari nilai jarak kata sumber dengan kata target *If* nilai jarak <= 3 *then* tentukan bentuk kata *Else* cek pola *regular expression then* tentukan pola kata *End*
No	**Proses**	**Keterangan**
9.	No Proses	1.3.3
	Nama Proses	Pemberian kode unik dan harakat secara umum
	Deskripsi	Memberikan kode unik dan mengharakati kata secara umum berdasarkan kategori kata
	Source (Sumber)	Proses 1.3.2
	Input	Kata sumber, kata target, pola kata, nilai jarak
	Output	*Array* kata berharakat, kode unik
	Destination (Tujuan)	Proses 1.3.3
	Logika Proses	*Begin* {*read* kata sumber, kata target, pola regex} *If* nilai jarak = 0 *then* beri harakat kata sumber dan kode unik sesuai kata target yang terdapat pada *database* *Else* cek dengan pola kata *then* beri harakat kata sumber dan kode unik sesuai pola yang terdapat pada *database* *End*

No	Proses	Keterangan
10.	No Proses	1.4.1
	Nama Proses	Identifikasi kode unik
	Deskripsi	Mengidentifikasi kode unik yang terdapat pada awal kata
	Source (Sumber)	Proses 1.3
	Input	*Array* kata berharakat, kode unik
	Output	Kode unik
	Destination (Tujuan)	Proses 1.4.2
	Logika Proses	*Begin* {*read* kode unik pada setiap kata} *If* identifikasi kode unik pada setiap kata *then* output kode unik yang teridentifikasi *End*
No	**Proses**	**Keterangan**
11.	No Proses	1.4.2
	Nama Proses	*Load* data latihan
	Deskripsi	Me-*load* data latihan dari *database*
	Source (Sumber)	Proses 1.4.1, *database*
	Input	Kode unik, data *training*, nilai *prior probability*
	Output	Data baru, data *training*, nilai *prior probability*
	Destination (Tujuan)	Proses 1.4.3
	Logika Proses	*Begin* {*read* kode unik} *Load* data latihan *End*
No	**Proses**	**Keterangan**
12.	No Proses	1.4.3

	Nama Proses	Proses klasifikasi
	Deskripsi	Menentukan gramatika setiap kata dan memberikan harakat akhir kata yang sesuai dengan gramatika menggunakan algoritma *bayesian classification*
	Source (Sumber)	Proses 1.4.2
	Input	Data baru, data *training*, nilai *prior probability*
	Output	*Array* kata dengan harakat sesuai gramatika, kode unik
	Destination (Tujuan)	Proses 1.5
	Logika Proses	*Begin* {*read* data baru, data *training*} *If* data baru sama dengan data *training* *then* beri gramatika dan harakat sesuai data *training* *Else* tentukan gramatika menggunakan algoritma *bayesian classification* *End*
No	**Proses**	**Keterangan**
13.	*No Proses*	2.1
	Nama Proses	*Input* data *training*
	Deskripsi	Memasukkan data *training* dan kata ke dalam *database*
	Source (Sumber)	Admin
	Input	Data kata, data *training*
	Output	Info Data kata dan data *training*
	Destination (Tujuan)	2.2
	Logika Proses	*Begin* {*input* data kata, data *training*} *If Input* data kata, data *training* *then output* data

		kata, data *training* *End*
No	**Proses**	**Keterangan**
14.	No Proses	2.2
	Nama Proses	Proses *learning*
	Deskripsi	Menghitung nilai *prior probability* dari masing-masing kelas
	Source (Sumber)	2.1
	Input	data *training*
	Output	data *training*, nilai *prior probability*
	Destination (Tujuan)	*Database*
	Logika Proses	*Begin* {*read* data *training*} *if* input data *training then* hitung nilai *prior probability AND insert into database* *End*

Tabel 3.8 Kamus Data

No	**Kamus**	**Keterangan**
1.	Nama	Kalimat bahasa Arab tanpa harakat
	Deskripsi	Kalimat yang di*input*kan oleh *user* berupa kalimat lengkap dengan tulisan Arab
	Penggunaan	Proses 1.1 *input* data kalimat Proses 1.2 pemecahan kalimat
No	**Kamus**	**Keterangan**
2.	Nama	*Array* kata tanpa harakat
	Deskripsi	Kumpulan beberapa kata hasil pemecahan kalimat
	Penggunaan	Proses 1.3 identifikasi kata

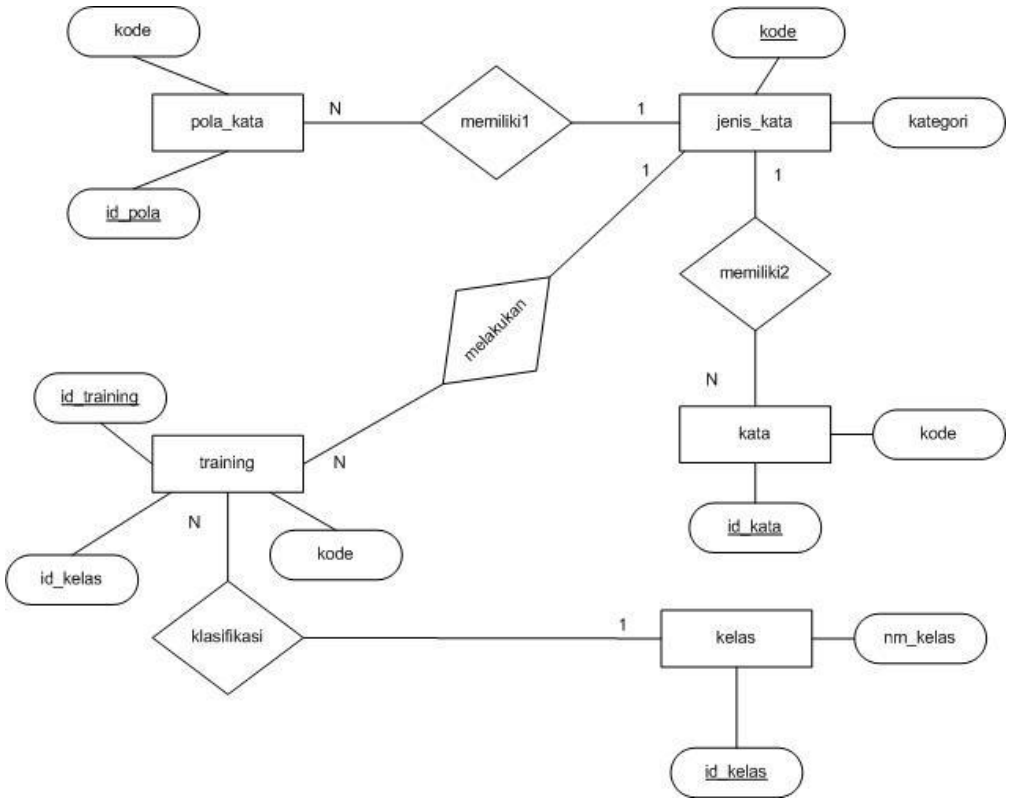
No	**Kamus**	**Keterangan**
3.	Nama	Data kata
	Deskripsi	Daftar bentuk kata yang tersimpan pada *database*
	Penggunaan	Proses 1.3.1 *load* data kata
No	**Kamus**	**Keterangan**
4.	Nama	*Array* kata berharakat
	Deskripsi	Kumpulan beberapa kata yang telah diharakati secara umum berdasarkan jenis kata
	Penggunaan	Proses 1.4.1 identifikasi kode unik
No	**Kamus**	**Keterangan**
5.	Nama	Kode unik
	Deskripsi	Kode berbentuk dua *digit* yang terdapat di awal kata untuk membedakan setiap jenis kata
	Penggunaan	Proses 1.4.1 identifikasi kode unik Proses 1.4.2 *load* data *training* Proses 1.4.3 proses klasifikasi Proses 1.5 penghapusan kode unik
No	**Kamus**	**Keterangan**
6.	Nama	Data *training*
	Deskripsi	Merupakan data yang tersimpan pada basis data untuk menentukan nilai probabilitas setiap kelas
	Penggunaan	Proses 1.4.2 *load* data *training* Proses 1.4.3 proses klasifikasi
No	**Kamus**	**Keterangan**
7.	Nama	*Array* kata berharakat sesuai gramatika
	Deskripsi	Kumpulan kata yang telah diharakati sesuai dengan gramatika masing-masing
	Penggunaan	Proses 1.5 penghapusan kode unik Proses 1.6 penggabungan kata
No	**Kamus**	**Keterangan**

8.	Nama	Kalimat berharakat sesuai gramatika
	Deskripsi	Hasil dari penggabungan kata yang telah diharakati sesuai gramatika masing-masing
	Penggunaan	*User*
No	**Kamus**	**Keterangan**
9.	Nama	Nilai *prior probability*
	Deskripsi	Nilai probabilitas utama dari setiap kelas
	Penggunaan	Proses 1.4.2 *load* data *training* Proses 1.4.3 proses klasifikasi
No	**Kamus**	**Keterangan**
10.	Nama	Kata sumber
	Deskripsi	Merupakan *array* kata yang akan melewati proses *levenstein distance*
	Penggunaan	Proses 1.3.2 proses *levenstein distance*
No	**Kamus**	**Keterangan**
11.	Nama	Kata target
	Deskripsi	Merupakan data kata yang akan melewati proses *levenstein distance*
	Penggunaan	Proses 1.3.2 proses *levenstein distance*
No	**Kamus**	**Keterangan**
12.	Nama	Nilai jarak (*distance*)
	Deskripsi	Merupakan hasil proses perbandingan kata sumber dengan kata target pada proses *levenstein distance*
	Penggunaan	Proses 1.3.3 pemberian kode unik dan harakat secara umum
No	**Kamus**	**Keterangan**
13.	Nama	Pola kata
	Deskripsi	Merupakan data kata yang menggunakan pola *regular expression* untuk membantu identifikasi kata

	Penggunaan	Proses 1.3.3 pemberian kode unik dan harakat secara umum
No	**Kamus**	**Keterangan**
14.	Nama	Data baru
	Deskripsi	Merupakan data yang berlum diketahui kelasnya
	Penggunaan	Proses 1.4.3 proses klasifikasi

3.4 Analisis Basis Data

Dalam memodelkan data dan menggambarkan keterhubungan antar data yang ada pada aplikasi digunakan alat bantu yaitu *Entity Relational Diagram* (ERD). Berikut adalah ERD yang terdapat pada aplikasi :



Gambar 3.13 ERD pada aplikasi pemberian harakat

3.5 Perancangan Aplikasi

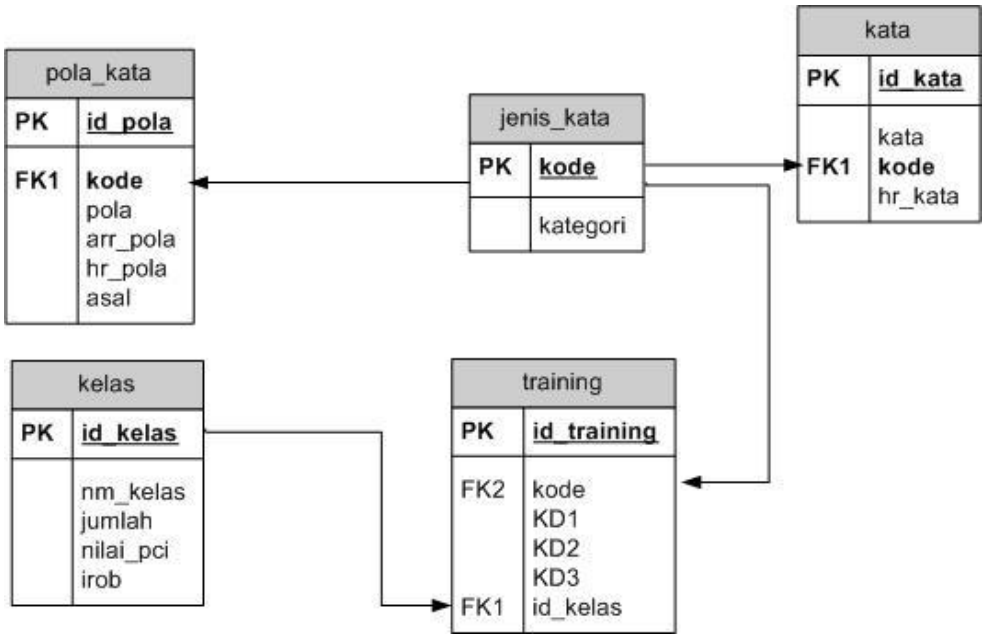
Perancangan aplikasi dapat didefinisikan sebagai penggambaran/ pembuatan sketsa perencanaan dari beberapa elemen yang terpisah ke dalam satu kesatuan yang utuh dan memiliki fungsi yang saling berkaitan.

3.5.1 Perancangan Basis Data

Perancangan basis data merupakan tahapan untuk memodelkan model konseptual ke dalam model basis data yang akan dipakai. Perancangan basis data terbagi dua, yaitu perancangan skema relasi dan perancangan struktur tabel.

3.5.1.1 Skema Relasi

Skema relasi berfungsi menggambarkan relasi atau keterkaitan antar tabel yang mempunyai atribut kunci utama yang sama, sehingga tabel-tabel tersebut menjadi satu kesatuan yang dihubungkan oleh atribut kunci tersebut. Berikut ini merupakan skema relasi basis data pada aplikasi pemberian harakat pada kalimat bahasa Arab :



Gambar 3.14 Skema relasi basis data pemberian harakat pada kalimat bahasa Arab

3.5.1.2 Perancangan Struktur Tabel

Perancangan struktur tabel adalah perancangan atribut tabel yang akan digunakan pada basis data. Tabel-tabel yang terdapat pada basis data yang digunakan dalam aplikasi ini adalah :

1. Tabel jenis_kata

Nama Field	Tipe	Ukuran	Kunci
Kode	Varchar	10	*Primary Key*
Kategori	Varchar	50	

2. Tabel kata

Nama Field	Tipe	Ukuran	Kunci
Id_kata	Integer	11	*Primary Key*
Kata	Varchar	100	
Kode	Varchar	10	*Foreign Key*
Hr_kata	Varchar	200	

3. Tabel pola_kata

Nama Field	Tipe	Ukuran	Kunci
Id_pola	Integer	11	*Primary Key*
Pola	Varchar	100	
Arr_pola	Varchar	100	
Hr_pola	Varchar	200	
Kode	Varchar	10	*Foreign Key*
Asal	Varchar	30	

4. Tabel kelas

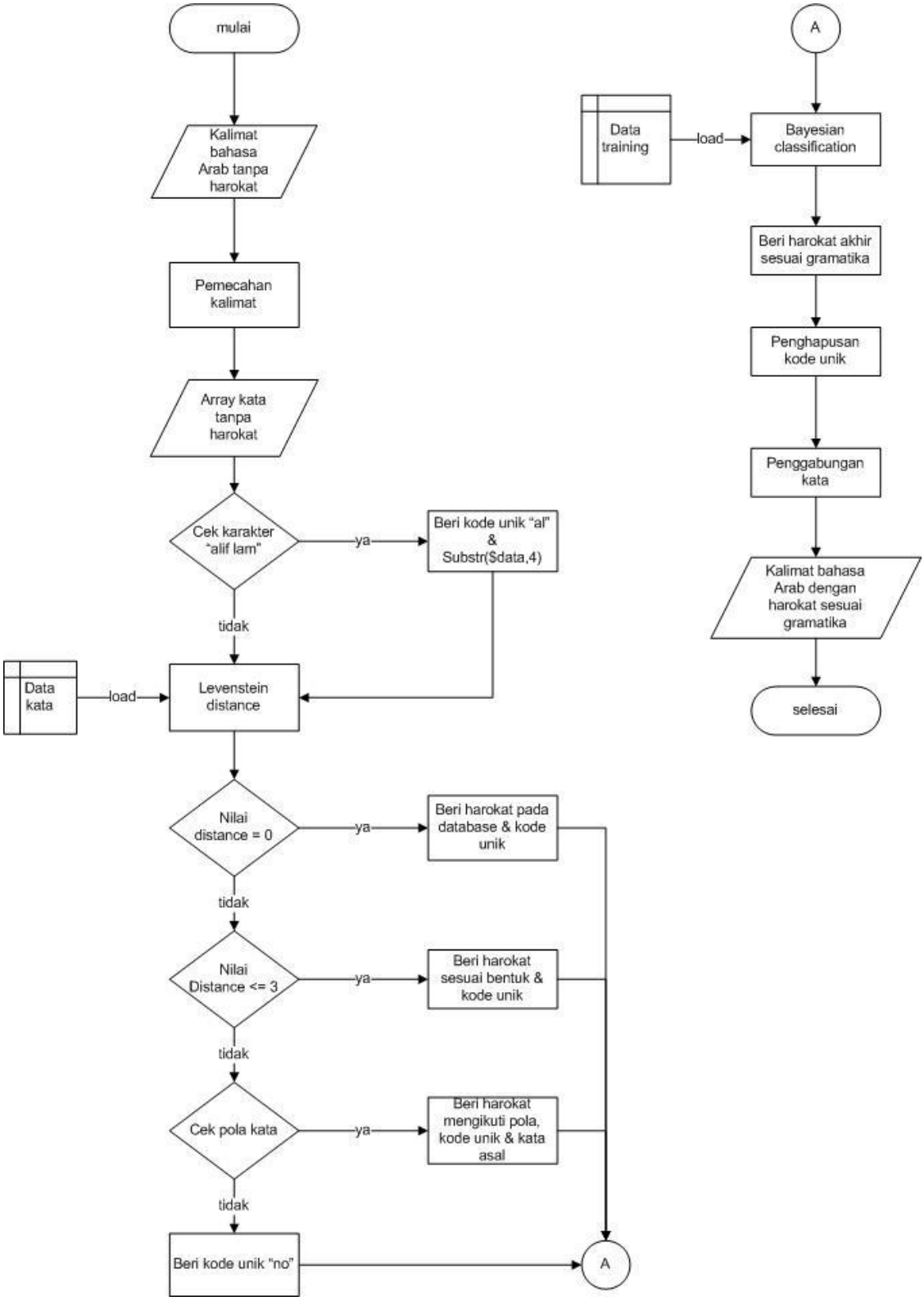
Nama Field	Tipe	Ukuran	Kunci
Id_kelas	Integer	11	*Primary Key*
Nm_kelas	Varchar	50	
Jumlah	Integer	11	
Nilai_pci	Float	11	
Irob	Varchar	30	

5. Tabel *training*

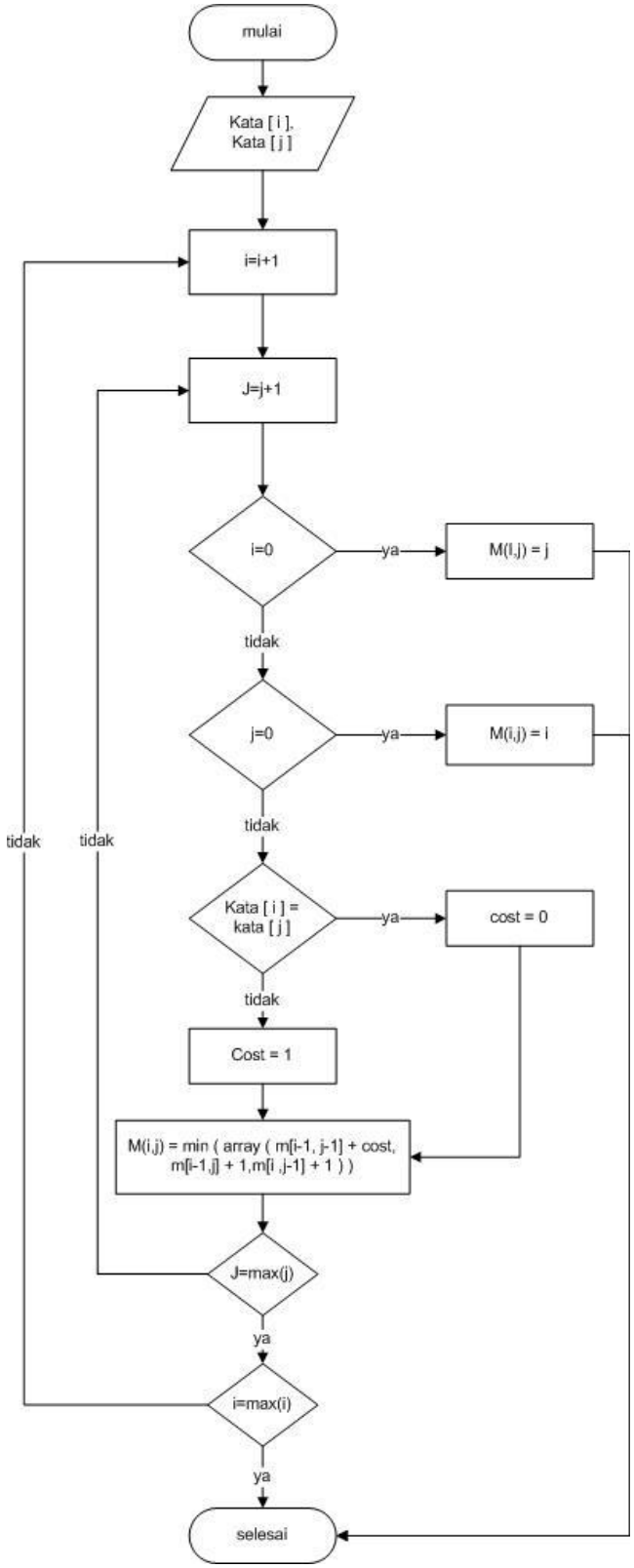
Nama Field	Tipe	Ukuran	Kunci
Id_training	Varchar	10	*Primary Key*
Kode	Varchar	10	*Foreign Key*
KD1	Varchar	10	
KD2	Varchar	10	
KD3	Varchar	10	
Id_kelas	Integer	11	*Foreign Key*

3.5.2 Perancangan Prosedural

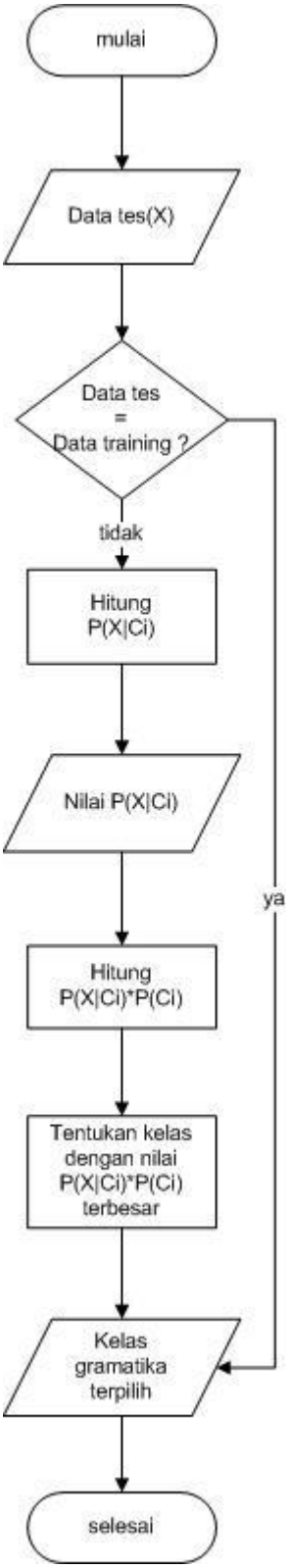
Pada bagian ini akan dibuat deskripsi rinci dari aplikasi, yaitu algoritma yang akan digunakan oleh masing-masing prosedur dan fungsi dengan menggunakan *flowchart*. *Flowchart* adalah bagan-bagan yang mempunyai arus yang menggambarkan langkah-langkah penyelesaian suatu masalah. *Flowchart* merupakan cara penyajian dari suatu algoritma. Berikut ini *flowchart* kerja aplikasi pemberian harakat kalimat bahasa Arab :



Gambar 3.15 *Flowchart* kerja aplikasi pemberian harakat kalimat bahasa Arab



Gambar 3.16 *Flowchart* Proses *Levenshtein Distance*



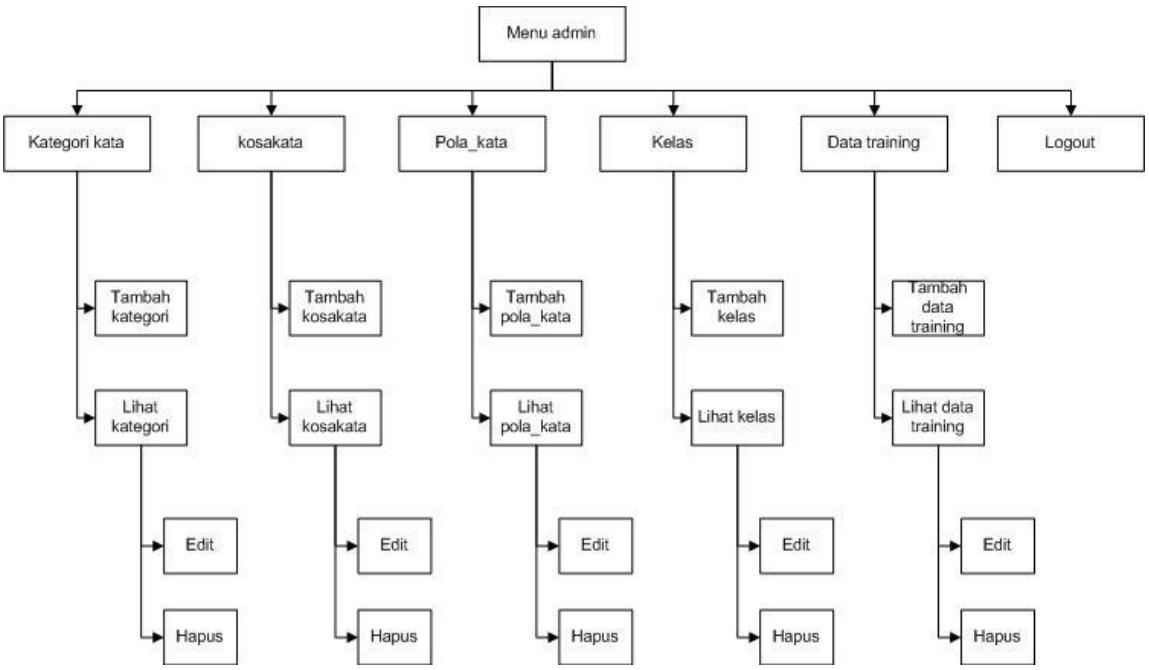
Gambar 3.17 *Flowchart* proses *bayesian classification*

3.5.3 Perancangan Antarmuka

Perancangan antarmuka menjelaskan sekumpulan objek-objek dan operasi-operasi yang bisa digunakan untuk memanipulasi objek. Perancangan antarmuka ini mencakup beberapa perancangan, diantaranya perancangan struktur menu, perancangan antarmuka halaman, perancangan pesan dan jaringan semantik.

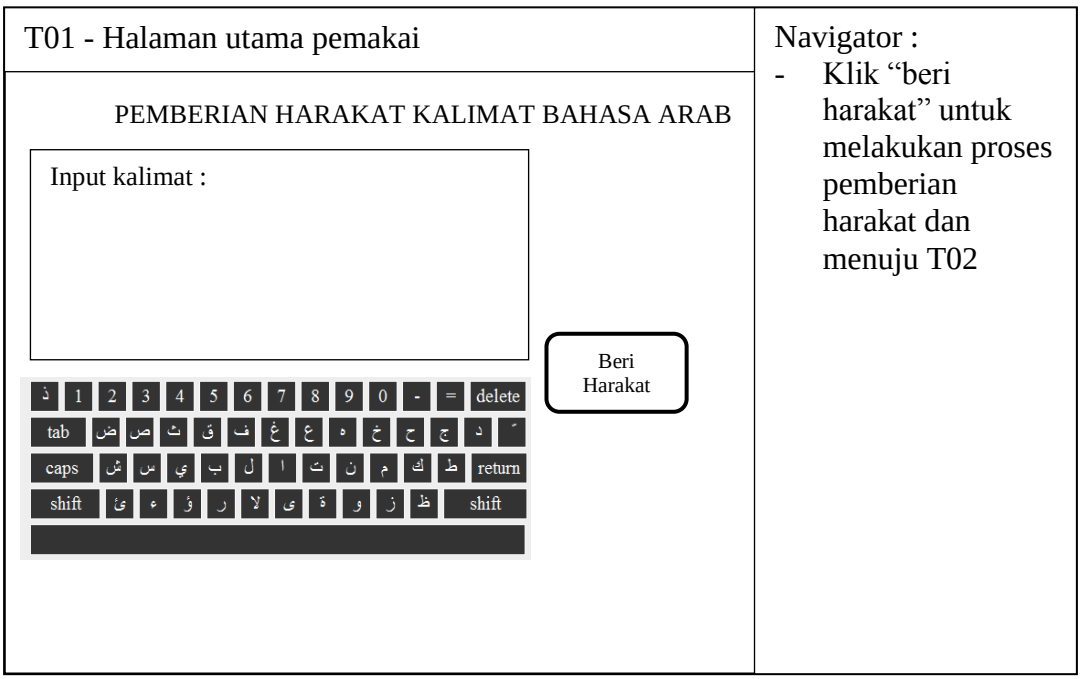
3.5.3.1 Perancangan Struktur Menu

Berikut adalah perancangan struktur menu aplikasi pemberian harakat kalimat bahasa Arab pada halaman admin :

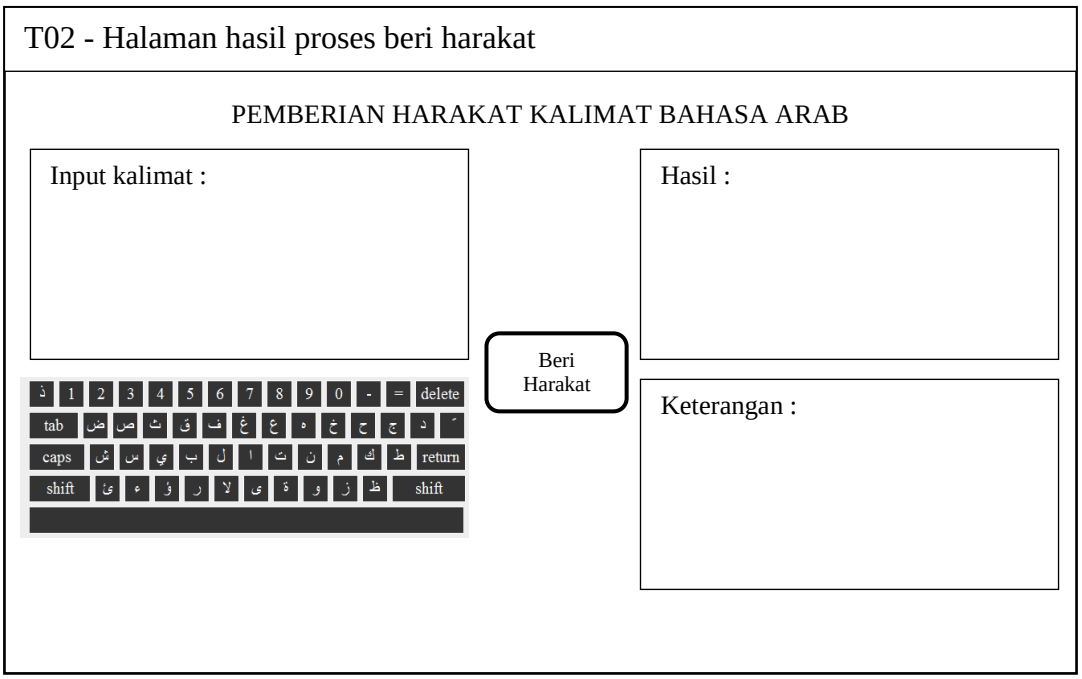


Gambar 3.18 Struktur menu admin

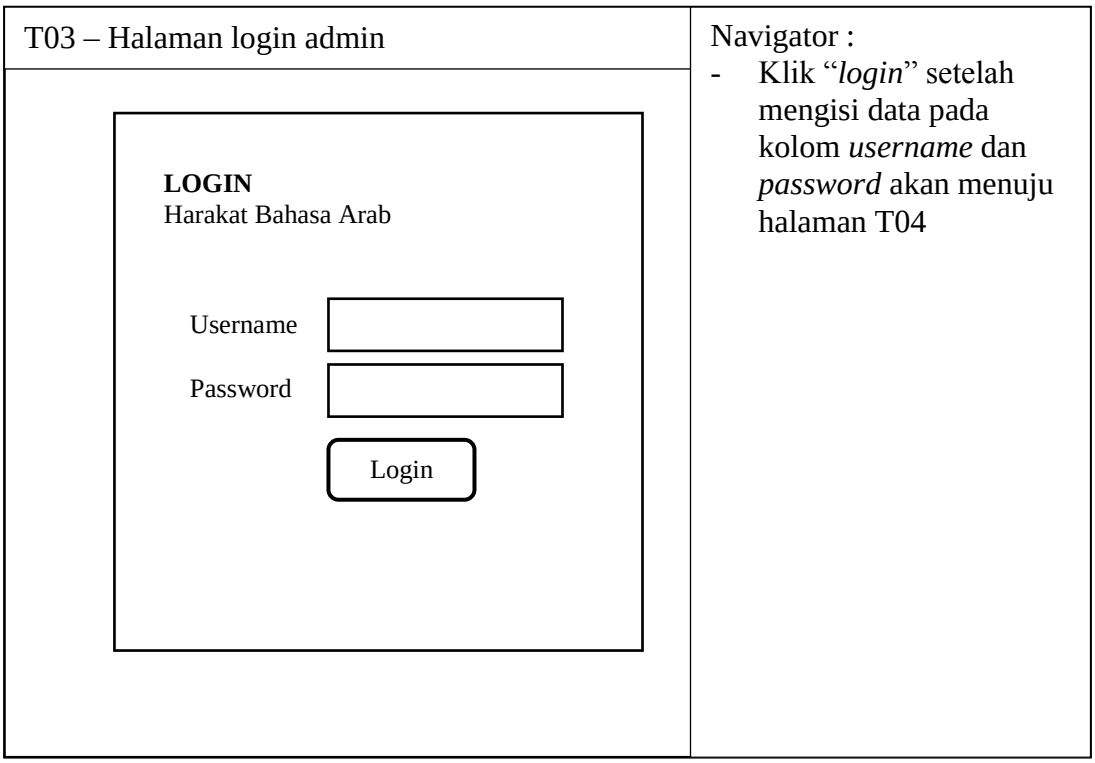
3.5.3.2 Perancangan Antarmuka Halaman



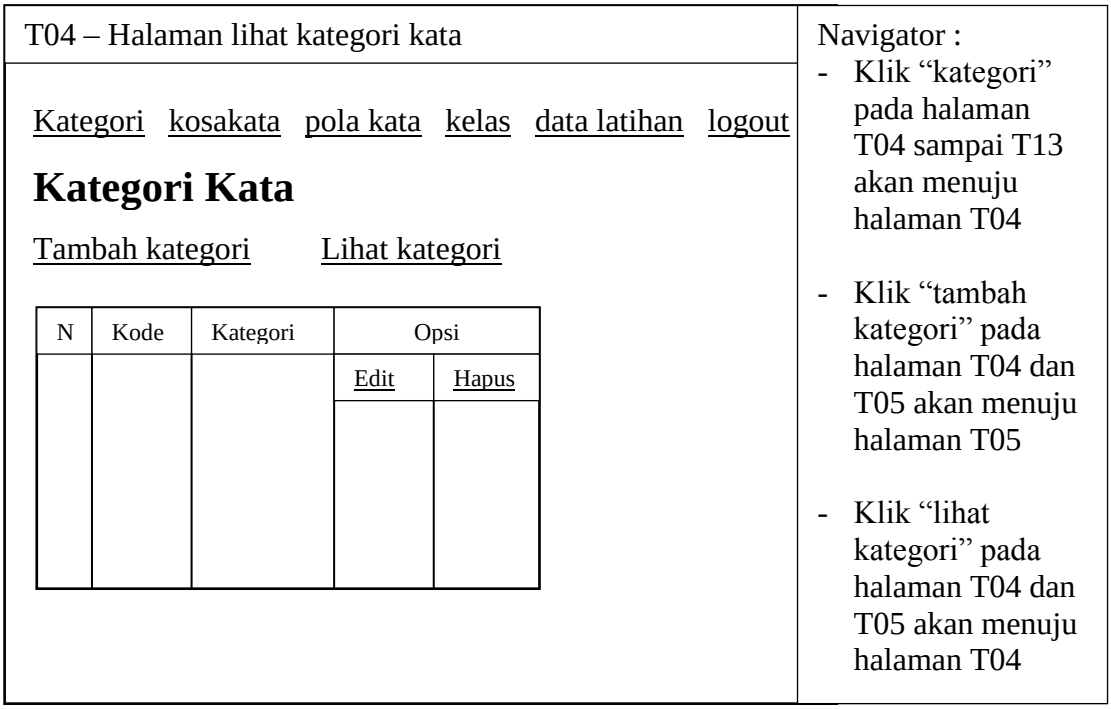
Gambar 3.19 Perancangan halaman utama antarmuka pemakai



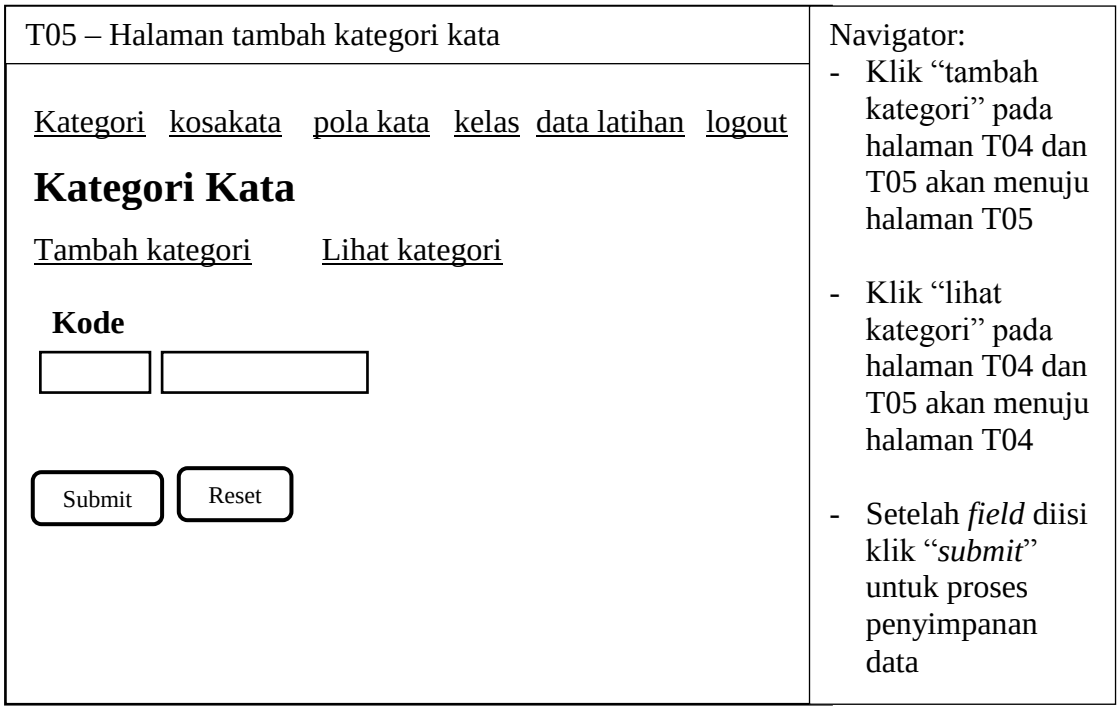
Gambar 3.20 Perancangan halaman hasil proses beri harakat



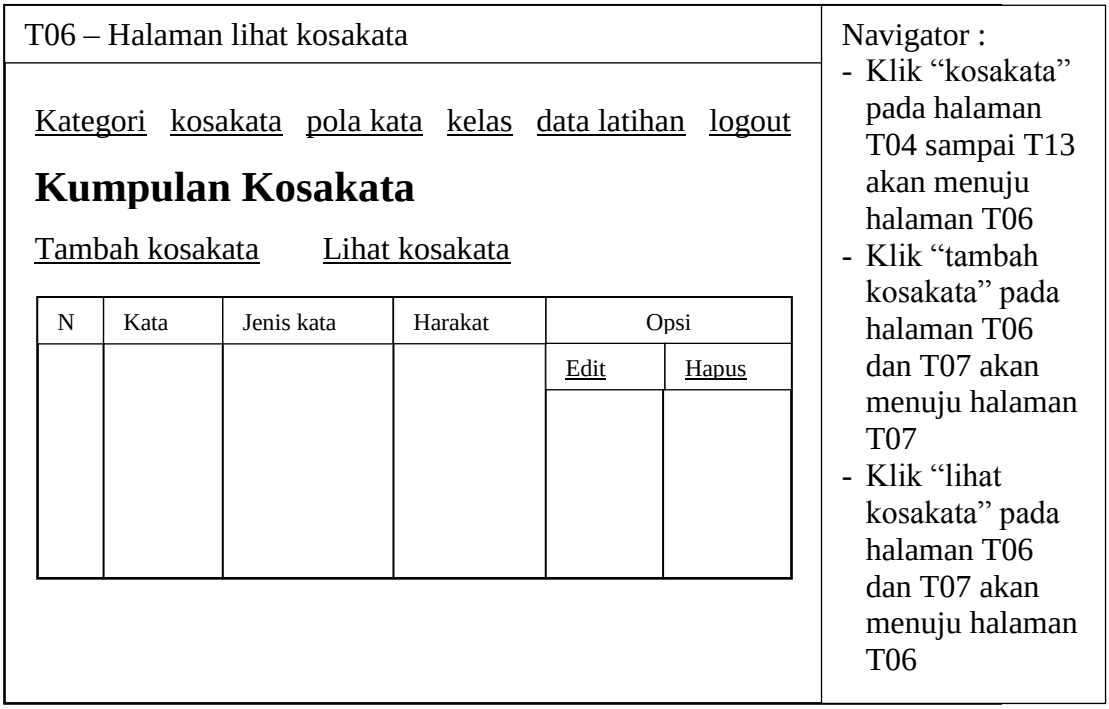
Gambar 3.21 Perancangan halaman *login* admin



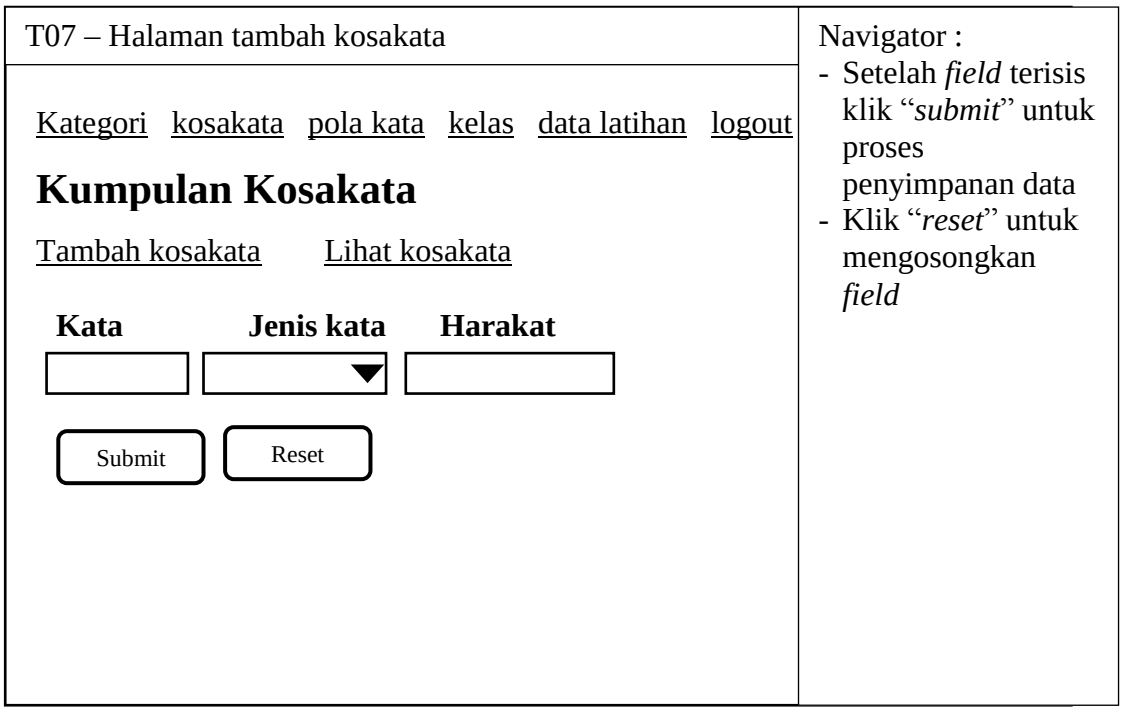
Gambar 3.22 Perancangan halaman lihat kategori kata



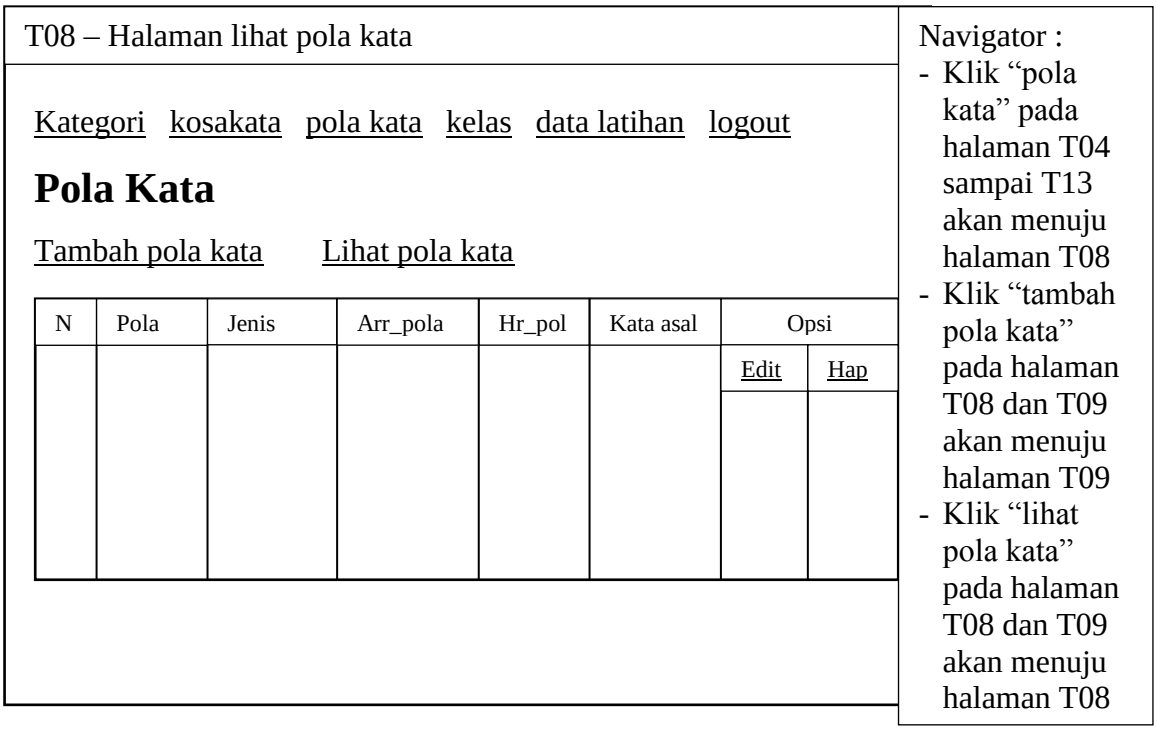
Gambar 3.23 Perancangan halaman tambah kategori kata



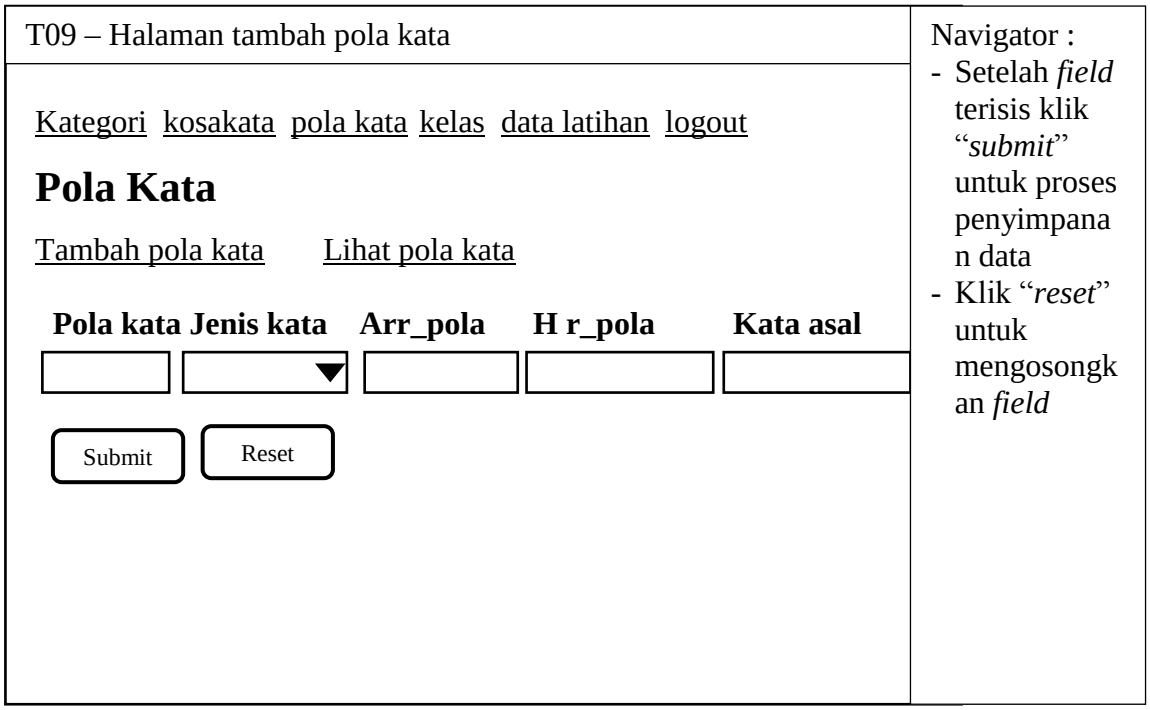
Gambar 3.24 Perancangan halaman lihat kosakata



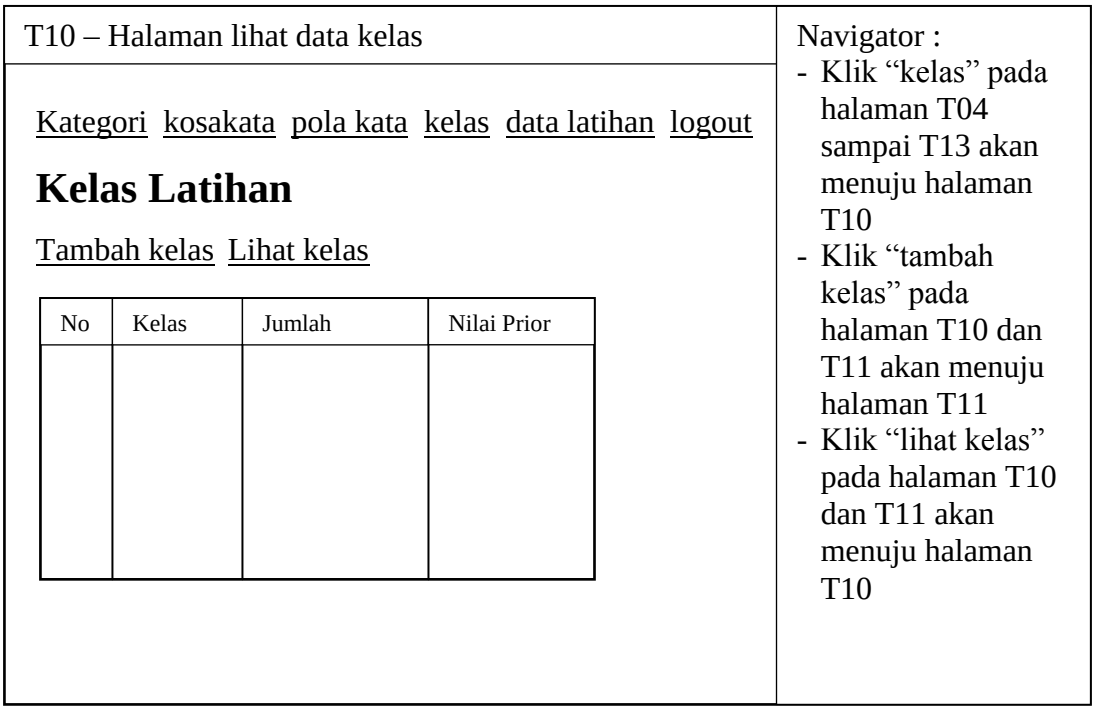
Gambar 3.25 Perancangan halaman tambah kosakata



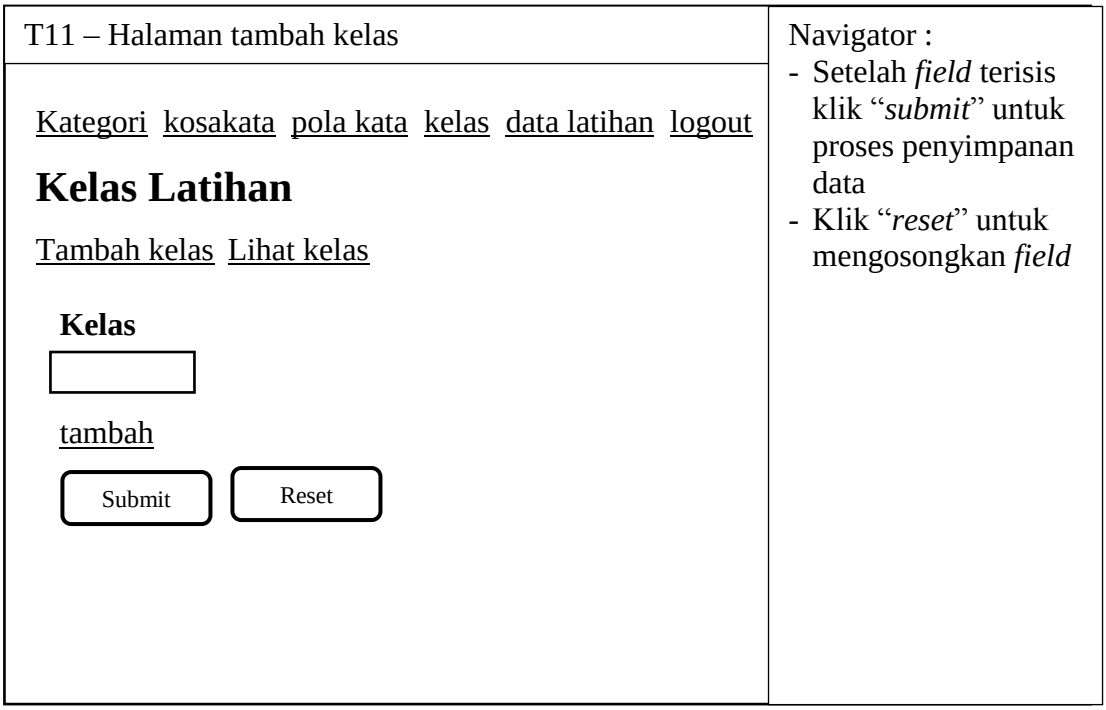
Gambar 3.26 Perancangan halaman lihat pola kata



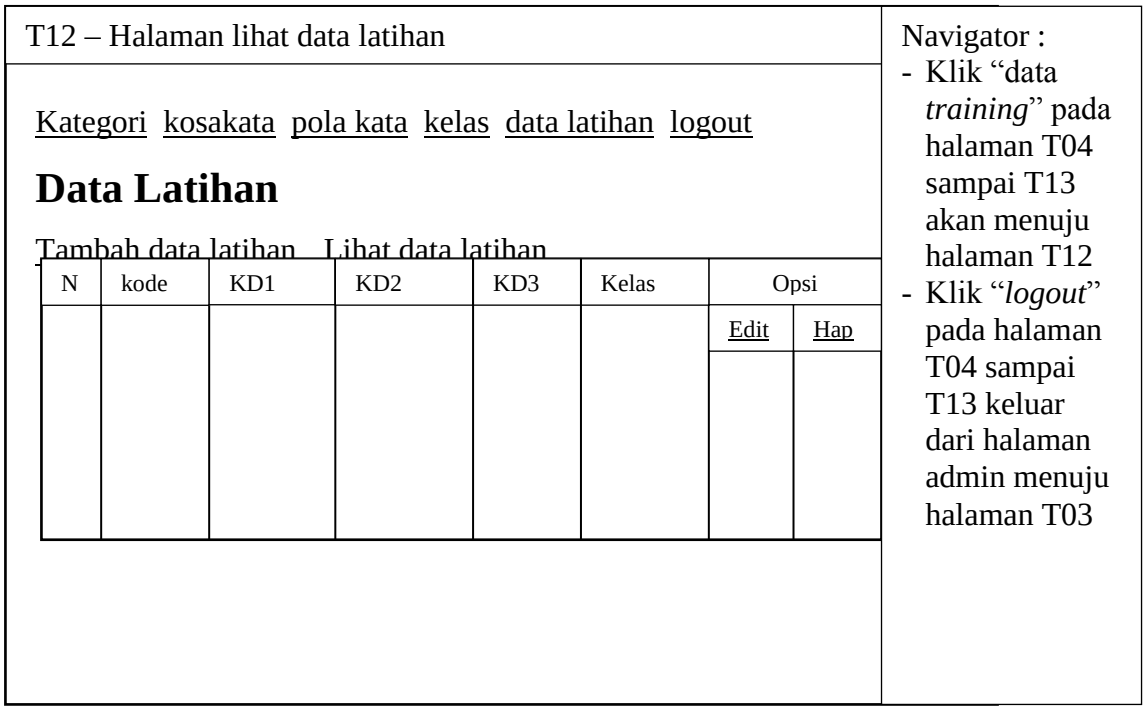
Gambar 3.27 Perancangan halaman tambah pola kata



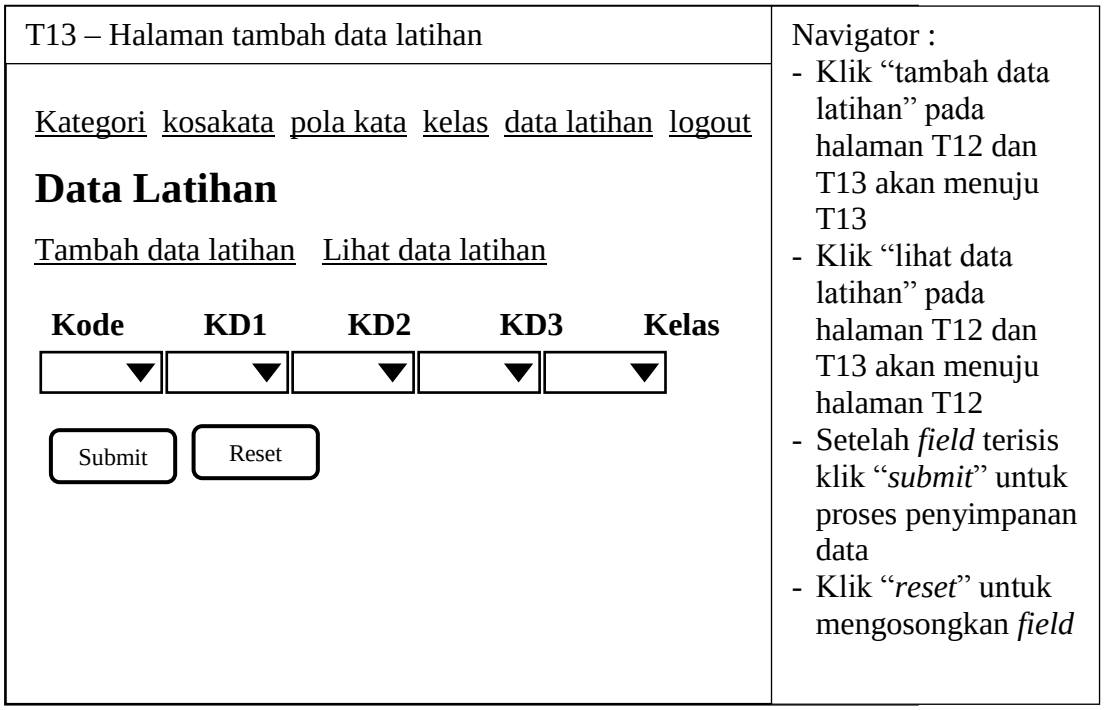
Gambar 3.28 Perancangan halaman lihat kelas latihan



Gambar 3.29 Perancangan halaman tambah kelas

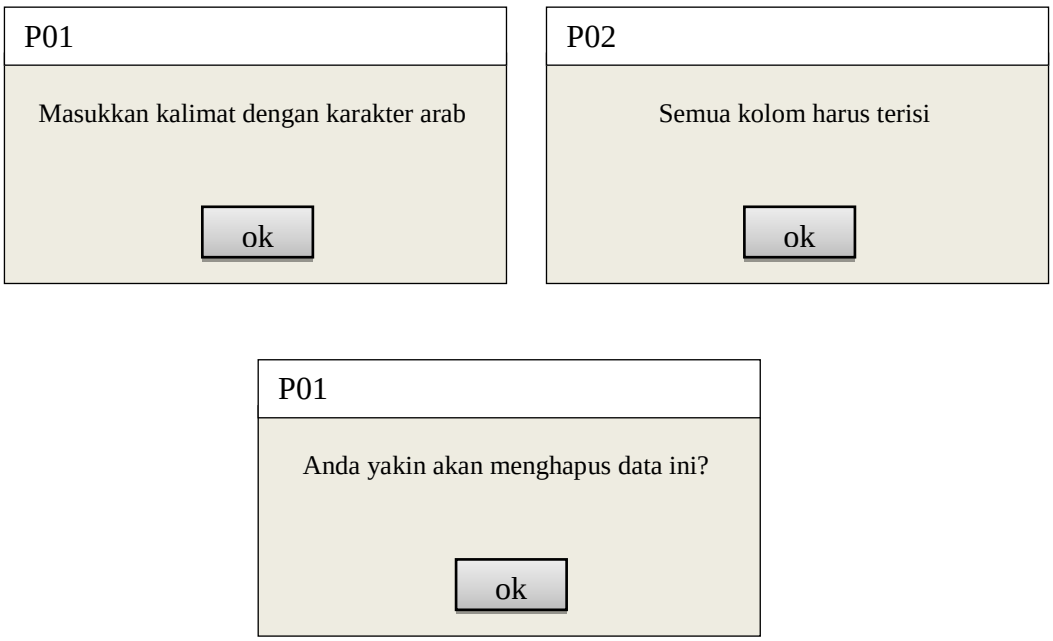


Gambar 3.30 Perancangan halaman lihat data latihan



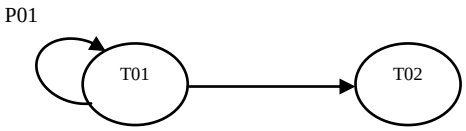
Gambar 3.31 Perancangan halaman tambah data latihan

3.5.3.3 Perancangan Pesan

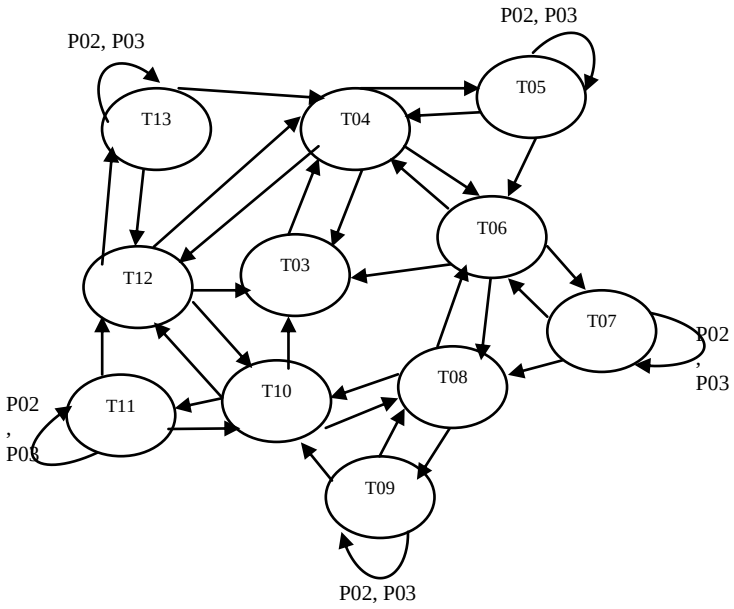


Gambar 3.32 Perancangan pesan

3.5.3.4 Jaringan Semantik



Gambar 3.33 Jaringan semantik untuk halaman pengguna umum



Gambar 3.34 Jaringan semantik untuk halaman admin