BAB III
ANALISIS DAN PERANCANGAN

Analisis dan perancangan berfungsi untuk mempermudah, memahami dan menyusun perancangan pada bab selanjutnya. Selain itu juga berfungsi untuk memberikan gambaran dan solusi pada pembuatan solusi ini.

3.1 Analisis Masalah

Bahasa Indonesia adalah bahasa resmi Republik Indonesia. Meskipun beratus-ratus bahasa daerah digunakan di negara ini, Bahasa Indonesia dipergunakan oleh lebih dari 200 juta jiwa, belum termasuk 20 juta pengguna Bahasa Melayu yang juga dapat memahami Bahasa Indonesia. *Internet, e-mail*, forum diskusi *on-line*, dan *world wide web* merupakan wahana yang tak terduga bagi negara yang terdiri atas ribuan pulau dengan jutaan penduduk yang tersebar di dalamnya bagi interaksi, pengembangan, dan pelestarian budaya yang beraneka ragam dan tiada taranya. Meskipun demikian, belum banyak tersedia mesin pencari yang efektif bagi pengguna Bahasa Indonesia untuk menggali informasi dari halaman-halaman web berbahasa Indonesia. Portal-portal *Internet* yang menyediakan sarana pencarian dokumen pada umumnya menggunakan teknologi komersial yang tersedia di pasaran dan diperuntukkan bagi Bahasa Inggris. Tanpa disadari, tidak adanya mesin pencari yang dapat mengolah dokumen dan informasi berbahasa Indonesia secara efektif telah menimbulkan dampak tersendiri bagi pengguna *Internet* yang mahir berbahasa Indonesia. Tanpa adanya fasilitas-fasilitas yang mendukung Bahasa Indonesia, perkembangan web Indonesia (situs-situs, dokumen-dokumen, dan informasi-informasi di dalam web

yang menggunakan Bahasa Indonesia) sangatlah minimal. Keterbatasan fasilitas tersebut turut pula menurunkan minat pengguna Bahasa Indonesia untuk meningkatkan kemahiran berbahasa dan penggunaan Bahasa Indonesia baik dalam proses penuangan maupun pencarian informasi.

3.2 Analisis Sistem *Search Engine*

Dalam membangun search engine yang paling diperhatikan adalah komponen-komponen penting mesin pencari yang memerlukan perhatian khusus agar dapat mendukung pemrosesan dokumen-dokumen berbahasa Indonesia. Satu-persatu, dokumen-dokumen yang diinginkan akan diproses lebih lanjut oleh modul pengindeks, yang terlebih dahulu akan mem-*parsing* atau mensegmentasi dokumen itu sehingga diperoleh daftar kata-kata yang ada didalamnya. Daftar kata itu kemudian disaring dengan membuang kata-kata yang ada di daftar *stop-word*. Kata-kata yang tersisa itu kemudian dihilangkan imbuhan-imbuhannya melalui proses *stemming* sehingga didapatkan daftar kata dasar yang dapat mewakili dokumen tersebut. Daftar kata dasar inilah yang kemudian diasosiasikan dengan dokumen dan URL (*Universal Resource Locator*) dari dokumen tersebut. *Query* juga diproses dengan cara yang hampir sama.

Modul temu-kembali akan membentuk daftar dokumen-dokumen yang diperkirakan relevan dengan *query* yang diberikan pengguna. Dokumen-dokumen tersebut kemudian diurutkan berdasarkan bobot kemiripan masing-masing dokumen dengan *query* pengguna. Model proses temu-kembali informasi yang penulis gunakan adalah *Vektor Space Model*.

Dalam vektor space model, dokumen dan query direpresentasikan sebagai vektor dalam ruang vektor yang disusun dalam indeks *term*, kemudian dimodelkan dengan persamaan geometri. Ada beberapa yang perlu diperhatikan pada model ruang vektor yaitu:

1. Menggunakan bobot index *term*
2. Adanya vektor dokumen dari kueri
3. Perhitungan *Cosine* menentukan kesamaan dokumen kueri

3.3 Teknik Temu Kembali Informasi

Secara garis besar, program ini terdiri dari 6 proses yaitu :

1. Proses pengenalan *file* berbasis hiperteks yang sudah ada.

2. Proses *Tokenizer*, yaitu unit pemrosesan dokumen menghasilkan token dan proses parsing dokumen untuk pengenalan token, yang terdapat di dalam file hyperteks yang sudah diinputkan.

3. Proses *Stoplist,* yaitu proses menghilangkan kata-kata buang yang didapat dari file hyperteks.

4. Proses *Stemming*, yaitu proses untuk menghilangkan imbuhan, awalan dan akhiran dari hasil *stoplist*.

5. Proses Pembobotan Istilah (*Term Weighting*) dan Pengindeksan, yaitu proses untuk tingkat kepentingan berbeda-beda suatu istilah kata dasar untuk menentukan hasil temu kembali yang hasilnya berupa indeks.

6. Proses Pembobotan Kueri (*Query Term Weighting*) dan Pembalikan *File* (*Inverted File*), yaitu proses pembobotan pada kueri user yang digunakan untuk mengukur kesamaan dengan bobot istilah, dan dibalikan kembali.

3.4 Tahap Memasukan File Hyperteks yang akan Diproses untuk Masukan Pengindeksan

File-file Hiperteks yang diambil dari piranti berupa kumpulan koleksi file-file hiperteks yang sudah.

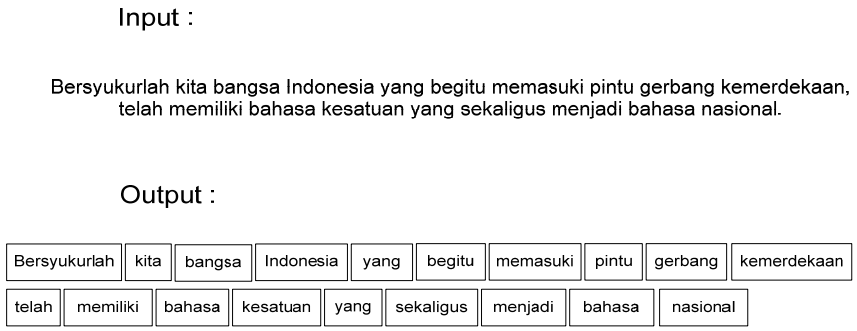
3.4.1 Tokenizer

Tokenizer adalah pemrosesan suatu unit dokumen yang mempunyai hasil akhir berupa *Tokens* unik dan banyaknya frekuensi *Tokens* yang terdapat dari suatu unit dokumen. Didalam proses *Tokenizer* terdapat dua proses yaitu proses *Tokenization* dan proses *Parsing*.

3.4.1.1 Tokenization

Dengan satu urutan karakter dan satu unit dokumen yang didefinisikan, tokenization adalah pekerjaan pemotongan satu urutan karakter menjadi beberapa bagian yang dinamakan *tokens* yang biasanya adalah kata, pada saat bersamaan proses tokenization membuang karakter tertentu, seperti pemberian tanda baca. *Tokens* ini sering dengan bebas menunjuk sebagai istilah atau kata, tetapi tanda baca ini kadang-kadang penting untuk membuat satu perbedaan *Tokens*. *Tokens* adalah satu contoh (instance) dari satu urutan karakter didalam beberapa dokumen tertentu.

Salah satu contoh dari *Tokenization*:



Contoh disini dilakukan pemotongan setiap satu kata dan menghilangkan semua karakter tanda baca.

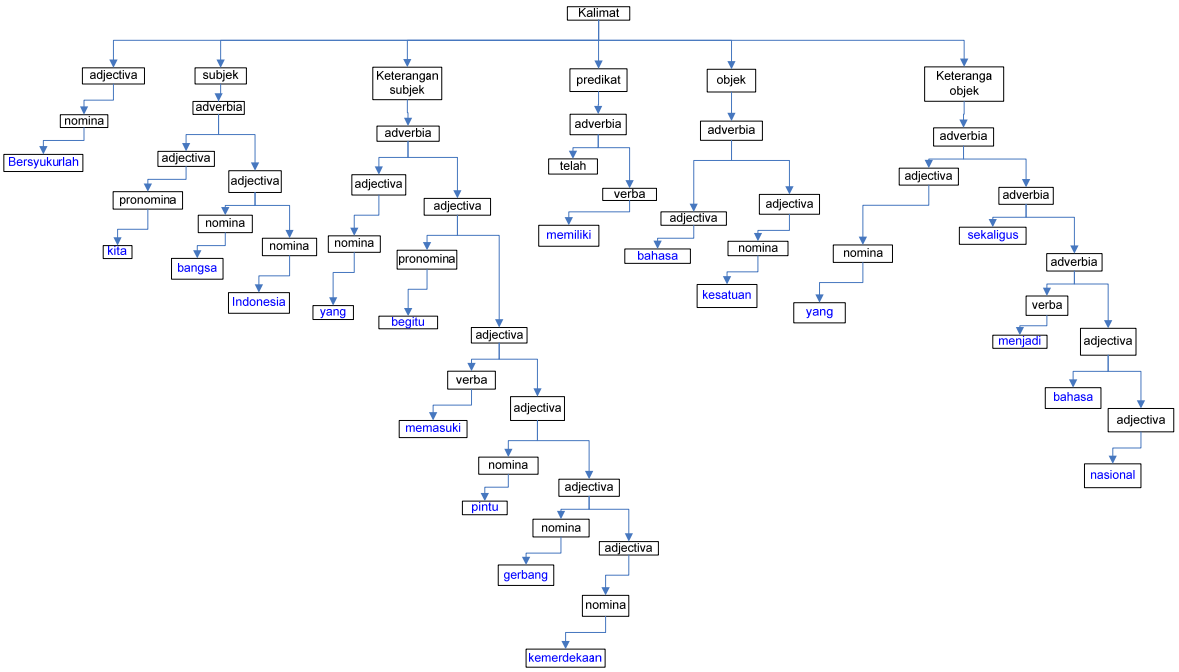
3.4.2. Parsing

Parsing adalah proses pengenalan dan pengambilan *Token* hasil Tokenization dari sekumpulan unit dokumen. Yang biasanya kata – kata. Proses parsing tidak hanya dapat dilakukan dalam proses *Information retrieval*, melainkan juga pada bidang lain seperti pada pembuatan sebuah *compiler* dan *Bahasa Alami*. Sebelumnya perlu diketahui arti dari istilah *parser* yaitu program yang melakukan proses parsing.

3.4.2.1 Parser

Parser dapat diibaratkan sebagai “otak” dari sebuah kompiler: komponen inilah yang menginferensikan makna dari bahasa dalam string input berdasarkan grammar/tatabahasa yang telah ditentukan sebelumnya oleh pemrogram. Dari makna atau nilai semantik sebuah rangkaian token, parser dapat segera memproses (interpretasi, translasi) struktur implisit dalam rangkaian token tersebut, atau menunda pemrosesan sampai didapat struktur utuh dari string input.

Contoh :



Gambar 3.1. Pohon Parsing untuk Kalimat *"Bersyukurlah kita bangsa Indonesia yang begitu memasuki pintu gerbang kemerdekaan, telah memiliki bahasa kesatuan yang sekaligus menjadi bahasa nasional"*

Keterangan :

1. Adjectiva, yaitu kata yang menjelaskan Nomina atau Pronomina.
2. Adverbia, yaitu kata yang menjelaskan Verba, Adjectiva, Adverbia lain, atau Kalimat.
3. Nomina, yaitu kata benda.
4. Numeralia, yaitu kata bilangan.
5. Partikel, yaitu kelas kata seperti kata depan, kata sambung, kata seru, kata sandang, ucapan salam.
6. Pronomina, yaitu meliputi kata ganti, kata rujuk, dan kata tanya.
7. Verba, yaitu kata kerja.

3.4.3 Stoplist Bahasa Indonesia

Stoplist Adalah proses pembuangan atau menghilangkan kata-kata buang, yaitu : Kata depan, kata sambung, kata ganti, dll. seperti : di, dan, tetapi, dia, yaitu, sedangkan, dan sebagainya.

Contoh :

Bersyukurlah kita bangsa Indonesia yang begitu memasuki pintu gerbang kemerdekaan, telah memiliki bahasa kesatuan yang sekaligus menjadi bahasa nasional.

.

Menjadi :

Bersyukurlah bangsa Indonesia memasuki pintu gerbang kemerdekaan, memiliki bahasa kesatuan sekaligus menjadi bahasa nasional.

3.5 Stemming Bahasa Indonesia

3.5.1 Algoritma Stemming Bahasa Indonesia Mirna Adriani dan Bobby Nazief

Berdasarkan pembentukan kata-kata yang sudah dibahas didalam BAB II Algoritma Stemming Bahasa Indonesia M. Adriani dan B Nazief ini mempunyai aturan imbuhan sendiri dengan model, seperti :

[[[AW+]AW+]AW+] Kata-Dasar [[+AK][+KK][+P]]

AW : Awalan

AK : Akhiran

KK : Kata Ganti kepunyaan

P : Partikel

Tanda kurung besar menandakan bahwa imbuhan adalah opsional.

Tabel 3.1, kombinasi Awalan dan Akhiran yang tidak dibenarkan. Kecuali kata dasar “tahu” dibolehkan dengan awalan “ke-“ dan Akhiran “-i” yang menjadi “ketahui”.

Awalan Prefiks)	Pelarangan Akhiran (Suffiks)
be-	-i
di-	-an
ke-	-i, -kan
me-	-an
se-	-i, -kan
te-	-an

Definisi sebelumnya membentuk aturan yang digunakan, namun ada perkecualian dan batasan yang disatukan dalam aturan.

1. tiga kata atau sedikit karakter yang tidak mempunyai imbuhan, maka tidak akan dilakukan proses stemming pada kata tersebut.
2. Imbuhan yang sama tidak pernah diulangi, sebagai contoh, setelah Awalan “te-“ atau dengan variasinya. Maka tidak akan mungkin Awalan “te-“ tersebut akan diulang kembali
3. Kita bisa menggunakan pembatasan konfiks dalam proses stemming untuk menghindari kombinasi imbuhan yang salah berdasarkan Tabel 3.I. Sebagai contoh, kata dasar yang mempunyai awalan “di-“, maka kata tersebut tidak akan diikuti dengan akhiran “-an”.
4. Menambahkan satu awalan dapat mengubah kata dasar atau sebelumnya sudah mempunyai awalan; dengan mendiskusikan ini lebih lanjut dalam uraian dari aturan untuk menggambarkan, mempertimbangkan “meng-" yang mempunyai variasi “mem-", “meng-", “meny-", dan “men-". Salah satu imbuhan ini dapat mengubah satu kata, sebagai contoh, untuk kata dasar “sapu", variasi yang diterapkan adalah “meny-" untuk menghasilkan kata “menyapu" dimana “s" dihilangkan

5. Karakter akan dikembalikan setelah proses penghilangan awalan.

Algoritma ini mempunyai tiga komponen : pengelompokan Imbuhan, urutan menggunakan aturan (dan perkecualiannya), dan kamus. Kamus di cek setelah semua aturan stemming berhasil. Jika kata yang dimaksud ditemukan dalam kamus, dan proses stemming berhasil menemukan kata dasarnya, algoritma kembali mencek kamus, dan algoritma berhenti.selain itu, beberapa langkah mencek kata yang kurang dari dua karakter dalam panjangnya, jika demikian tidak akan dilakukan proses stemming pada kata tersebut.

3.5.2 Algoritma ini Dijabarkan untuk setiap kata yang akan di Stemming :

1. Kata yang belum di Stemming dicari dalam kamus. Jika ketemu maka diasumsikan kata tersebut adalah kata dasar, maka kata tersebut dikembalikan dan algoritma berhenti.
2. Hilangkan sufiks untuk (“-lah", “-kah", “-ku", “-mu", atau “-nya"). Pertama hilangkan (P) (“-lah",”-kah”,”-tah”,“-pun"). Setelah itu hilangkan juga (KK) sufiks (“-ku", “-mu", atau “-nya"), contoh : kata “bajumulah”, proses stemming pertama menjadi “bajumu” dan proses stemming kedua menjadi “baju”.jika kata “baju” ada didalam kamus maka algoritma berhenti.

   Sesuai dengan model imbuhan, menjadi :

   [[[AW+]AW+]AW+] Kata-Dasar [+AK]
3. Hilangkan juga (AK) sufiks (“-i",”-an”, dan “-kan"), jika berhasil maka jalankan langkah 4.

Dengan model :

[[[AW+]AW+]AW+] Kata-Dasar

Contoh : kata “membelikan” distemming menjadi “membeli”, jika tidak ada dalam kamus maka dilakukan proses penghilangan prefiks pada langkah 4.

4. Penghilangan prefiks mengikuti langkah-langkah berikut ini:
    a. Jika sufiks telah dihilangkan pada langkah 3 maka aturan pelarangan kombinasi prefiks-sufiks dicek sesuai dengan aturan pada tabel 3.1. Jika aturan sesuai maka algoritma kembali
    b. Jika prefiks yang sekarang sesuai dengan prefiks sebelumnya maka algoritma kembali.
    c. Jika 3 prefiks telah sebelumnya dihapus maka algoritma kembali.
    d. Jenis prefiks ditentukan dengan salah satu langkah berikut ini:
        i. Jika prefiks dari kata adalah “di-", “ke-", atau “se-", maka maka dapat langsung dihilangkan.
        ii. Jika prefiks adalah {“te-", “be-", “me-", atau “pe-"}, mempunyai variasi yang berbeda. Yang menggunakan aturan peluruhan yang akan dijelaskan pada tabel 3.2. Dengan contoh, prefiks “me-“ dapat meluruh menjadi “mem-“, “men-“, “meny-“, atau “meng-“ tergantung pada huruf awal dari kata dasar tersebut. Langkah sebelumnya menstemming kata “membelikan” menjadi “membeli”.

Sekarang menstemming prefiks “mem-“ menjadi “beli”. Ini adalah kata yang terdapat dalam kamus, maka proses berhenti. Dan jika tidak ada prefiks yang cocok, maka proses berhenti, dan algoritma menunjukan bahwa kata dasar tidak ditemukan.

iii. Jika pencarian kata dalam kamus saat ini gagal, algoritma mengulang kembali pada langkah 4 (ini adalah proses reckursif). Jika kata tersebut ditemukan dalam kamus, maka proses berhenti.

5. Jika setelah rekursif penghilangan prefiks, kata dasar tetap tidak ditemukan. Maka *recoding* menguji kolom aturan dari aturan tabel 3.2. kolom ini menunjukan variasi prefiks dan *recoding* karakter untuk digunakan pada saat kata dasar diawali dengan huruf tertentu, atau pada suku kata pertama dari akhir kata dasar dengan huruf tertentu. *Recoding* karakter ditunjukan sebagai huruf kecil tertentu diikuti dengan tanda hubung sebagai kaitan. Tidak semua prefiks mempunyai karakter *recoding*.

   Sebagai contoh, kata “menangkap” mengikuti aturan 15 untuk prefiks “me-“ (diawali dengan prefiks “men-” diikuti dengan huruf hidup/vokal “-a”). Setelah menghilangkan “men-“ seperti pada langkah 4, didapatkan “angkap”, ini bukan kata dasar.

   Untuk aturan 15, terdapat dua aturan recoding karakter yang memungkinkan, “n” (sebagai “men-*n*V...”) dan “t” (sebagai “men-*t*V...”).

ini adalah suatu pengecualian; dimana hanya satu recoding karakter dari banyak kasus. Algoritma mengenali ”n” menjadi “angkap” menghasilkan “nangkap”, dan kembali pada langkah 4. Karena bukan kata dasar yang benar, sebagai gantinya “t” mendapatkan “tangkap”, dan kembali pada langkah 4. Dan “tangkap” sebagai kata dasar yang benar. Algoritma berhenti.

6. jika semua langkah gagal, algoritma mengembalikan kata asli yang tidak distemming.

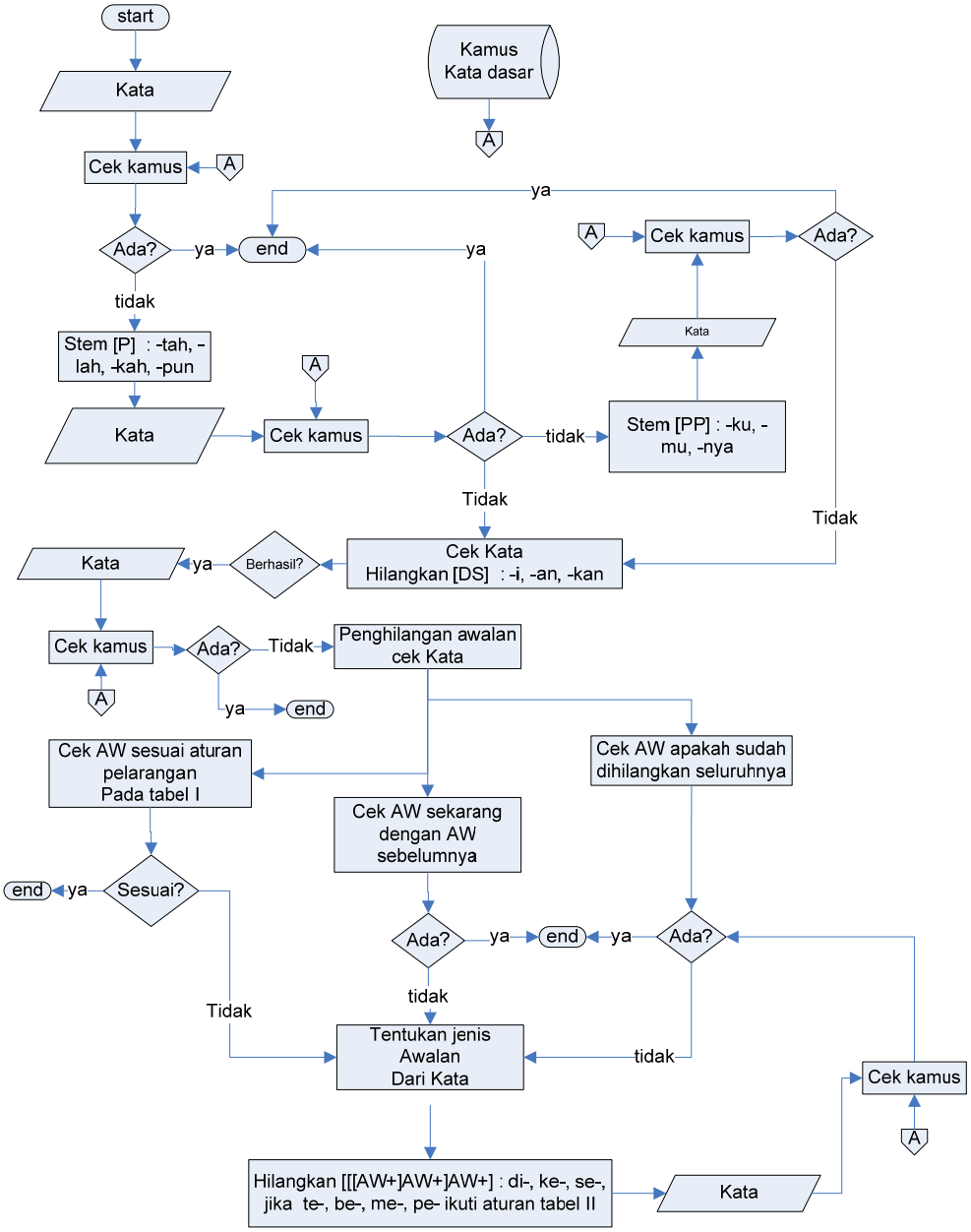
3.5.3 Peluruhan Awalan (prefiks)

Saat kita menemukan awalan yang kompleks, kita menentukan batas menurut aturan yang ditunjukkan pada tabel 3.2. seperti kata “menangkap”. Dengan melihat aturan untuk prefiks “me-“, kita melihat huruf ketiga dari kata tersebut adalah “n” sebagai ganti dari “m”, dan tidak akan menggunakan aturan 10, aturan 11, aturan 12, aturan 13, dan aturan 14 dengan huruf ke empatnya dari kata adalah “a” bukan “c”, “d”, “j”, atau “z”. Dan akhirnya yang dipilih adalah aturan 15, dengan menunjukkan prefiks yang dihilangkan adalah “me-“. Dengan salah satu hasil stem “nangkap”, yang tidak terdapat didalam kamus. Dan “tangkap” yang terdapat didalam kamus.

Beberapa sisa aturan peluruhan, contoh : seperti aturan 17 untuk prefiks “me-“, dengan kata “mengaku” dapat menjadi “meng-aku” dengan kata dasar “aku” atau menjadi “meng-kaku” dengan kata dasar “kaku”. Keduanya adalah kata yang benar, dan kita menentukan kata dasar yang benar tergantung dari konteks.

Tabel 3.2. Formula untuk aturan turunan prefiks. Huruf “V” menunjukkan vokal, huruf “C” menunjukkan konsonan, huruf “A” menunjukkan huruf apapun, dan huruf “P”menunjukkan pecahan kata pendek seperti “er”. Prefiks dipisahkan sisa dari kata pada posisi ditunjukkan dengan tanda penghubung. Huruf kecil diikuti tanda penghubung dan keterkaitan luar adalah karakter recoding. Jika karakter awal dari kata tidak cocok dengan aturan yang ada ini, maka prefiks tidak akan dihilangkan. Aturan imbuhan ini tidak menitik beratkan seperti didefinisikan dalam buku tata bahasa Moeliono dan Dardjowidjojo [1988] dan Sneddon [1996].

Rule	Construct	Return
1	berV...	ber-V... | be-rV...
2	berCAP...	ber-CAP... where C!='r' and P!='er'
3	berCAerV...	ber-CAerV... where C!='r'
4	belajar...	bel-ajar...
5	beC_1erC...	be-C_1erC ... where C_1!={'r'| 'l'}
6	terV...	ter-V... | te-rV...
7	terCP...	ter-CP... where C!='r' and P!='er'
8	terCer...	ter-Cer... where C!='r'
9	teC_1erC_2...	te-C_1erC_2... where C_1!='r'
10	me{l|r|w|y}V...	me-{l|r|w|y}V...
11	mem{b|f|v}...	mem-{b|f|v}...
12	mempe...	mem-pe...
13	mem{rV|V}...	me-m{rV|V}... | me-p{rV|V}...
14	men{c|d|j|z}...	men-{c|d|j|z}...
15	menV...	me-nV... | me-tV...
16	meng{g|h|q|k}...	meng-{g|h|q|k}...
17	mengV...	meng-V... | meng-kV...
18	menyV...	meny-sV...
19	mempV...	mem-pV... where V!='e'
20	pe{w|y}V...	pe-{w|y}V...
21	perV...	per-V... | pe-rV...
22	perCAP...	per-CAP... where C!='r' and P!='er'
23	perCAerV...	per-CAerV... where C!='r'
24	pem{b|f|v}...	pem-{b|f|v}...
25	pem{rV|V}...	pe-m{rV|V}... | pe-p{rV|V}...
26	pen{c|d|j|z}...	pen-{c|d|j|z}...
27	penV...	pe-nV... | pe-tV...
28	peng{g|h|q}...	peng-{g|h|q}...
29	pengV...	peng-V... | peng-kV...
30	penyV...	peny-sV...
31	pelV...	pe-lV...; Exception: for “pelajar”, return ajar
32	peCP...	pe-CP... where C!={r|w|y|l|m|n} and P!='er'
33	peCerV...	per-CerV... where C!={r|w|y|l|m|n}



Gambar 3.2 Flowchart dari algoritma stemming bahasa Indonesia M. Adriani dan B. Nazief

Contoh Stemming Algoritma M. Adriani dan B. Nazief,

Input :

Bersyukurlah bangsa Indonesia memasuki pintu gerbang kemerdekaan,
memiliki bahasa kesatuan sekaligus menjadi bahasa nasional.

Output :

syukur bangsa Indonesia masuk pintu gerbang merdeka,
milik bahasa satu sekaligus jadi bahasa nasional.

3.6 Algoritma Model Ruang Vektor (*Vector Space Model*)

3.6.1 Pembobotan Istilah (*Term Weighting*) dan Pengindeksan

Pembobotan Istilah (*Term Weighting*) dan Pengindeksan adalah proses pemberian bobot kemunculan istilah yang terdapat pada suatu dokumen dalam kumpulan dokumen. Yang hasilnya berupa indeks beserta dengan bobot istilahnya.

Seperti yang sudah di bahas didalam BAB II, penulis menerapkan rumus pembobotan istilah *Vektor Space Model* Sebagai berikut:

Contoh :

Tabel 3.3 Pembobotan istilah model vektor berdasarkan $w_i = tf_i * IDF_i$

ISTILAH MODEL VEKTOR BERDASARKAN $w_i = tf_i * IDF_i$									
Query, Q : "pintu bahasa" D_1 : "syukur bangsa Indonesia masuk pintu gerbang merdeka" D_2 : "milik bahasa satu sekaligus jadi bahasa nasional" D = 2; IDF = log(D/df_i)									
		tf_i yang terhitung					**Bobot $w_i = tf_i * IDF_i$**		
Istilah	**Q**	**D_1**	**D_2**	**df_i**	**D/df_i**	**IDF_i**	**Q**	**D_1**	**D_2**
Bahasa	1	0	2	1	2/1=2	0.3010	0.3010	0	0.6020
Bangsa	0	1	0	1	2/1=2	0.3010	0	0.3010	0
Gerbang	0	1	0	1	2/1=2	0.3010	0	0.3010	0
Indonesia	0	1	0	1	2/1=2	0.3010	0	0.3010	0
Jadi	0	0	1	1	2/1=2	0.3010	0	0	0.3010
Masuk	0	1	0	1	2/1=2	0.3010	0	0.3010	0
Merdeka	0	1	0	1	2/1=2	0.3010	0	0.3010	0
Milik	0	0	1	1	2/1=2	0.3010	0	0	0.3010
Nasional	0	0	1	1	2/1=2	0.3010	0	0	0.3010
Pintu	1	1	0	1	2/1=2	0.3010	0.3010	0.3010	0
Satu	0	0	1	1	2/1=2	0.3010	0	0	0.3010
Syukur	0	1	0	1	2/1=2	0.3010	0	0.3010	0

3.6.1.1 Pengindeksan

Sebuah bahasa indeks adalah bahasa yang digunakan untuk menggambarkan dokumen-dokumen dan permintaan-permintaan.

Contoh :

Tabel 3.4 Pengindeksan berdasarkan dokumen

Doc1

Bersyukurlah kita bangsa Indonesia yang begitu memasuki pintu gerbang kemerdekaan,

Doc 2

telah memiliki bahasa kesatuan yang sekaligus menjadi bahasa nasional.

→

Term	Doc#
Bersyukurlah	1
kita	1
bangsa	1
Indonesia	1
yang	1
begitu	1
memasuki	1
pintu	1
gerbang	1
kemerdekaan	1
telah	2
memiliki	2
bahasa	2
kesatuan	2
yang	2
sekaligus	2
menjadi	2
bahasa	2
nasional.	2

Tabel 3.5 Pengindeksan berdasarkan dokumen yang diurutkan berdasarkan abjad

Term	Doc#
bersyukurlah	1
kita	1
bangsa	1
Indonesia	1
yang	1
begitu	1
memasuki	1
pintu	1
gerbang	1
kemerdekaan	1
telah	2
memiliki	2
bahasa	2
kesatuan	2
yang	2
sekaligus	2
menjadi	2
bahasa	2
nasional.	2

→

Term	Doc#
bersyukurlah	1
bahasa	2
bahasa	2
bangsa	1
begitu	1
gerbang	1
Indonesia	1
kemerdekaan	1
kesatuan	2
kita	1
memasuki	1
memiliki	2
menjadi	2
nasional.	2
pintu	1
sekaligus	2
telah	2
yang	1
yang	2

Tabel 3.6 Pengindeksan berdasarkan Kemunculan istilah yang sama dan frekuensi kemunculan istilah

Term	Doc#
bersyukurlah	1
bahasa	2
bahasa	2
bangsa	1
begitu	1
gerbang	1
Indonesia	1
kemerdekaan	1
kesatuan	2
kita	1
memasuki	1
memiliki	2
menjadi	2
nasional.	2
pintu	1
sekaligus	2
telah	2
yang	1
yang	2

→

Term	Doc#	*Term* Frekuensi
bersyukurlah	1	1
bahasa	2	2
bangsa	1	1
begitu	1	1
gerbang	1	1
Indonesia	1	1
kemerdekaan	1	1
kesatuan	2	1
kita	1	1
memasuki	1	1
memiliki	2	1
menjadi	2	1
nasional.	2	1
pintu	1	1
sekaligus	2	1
telah	2	1
yang	1	1
yang	2	1

Tabel 3.7 Pengindeksan berdasarkan Kemunculan istilah yang sama dan frekuensi kemunculan istilah, beserta bobotnya.

Term	Doc#	Frekuensi	Bobot Kata
Bersyukurlah	1	1	0.3010
bahasa	2	2	0.6020
bangsa	1	1	0.3010
begitu	1	1	0.3010
gerbang	1	1	0.3010
Indonesia	1	1	0.3010
kemerdekaan	1	1	0.3010
kesatuan	2	1	0.3010
kita	1	1	0.3010
memasuki	1	1	0.3010
memiliki	2	1	0.3010
menjadi	2	1	0.3010
nasional.	2	1	0.3010
pintu	1	1	0.3010
sekaligus	2	1	0.3010
telah	2	1	0.3010
yang	1	1	0.3010
yang	2	1	0.3010

3.6.2 Proses Pembobotan Kueri (*Query Term Weighting*) dan Ukuran Kesamaan (*Similarity Measurement*)

3.6.2.1 Pembobotan Kueri (*Query Term Weighting*)

Pembobotan Kueri (*Query Term Weighting*) dan Pembalikan *File* (*Inverted File*) adalah proses pemberian bobot pada kueri masukan, pemberian bobot pada kueri dilakukan unutuk mempermudah dalam mencari istilah yang sama dalam hasil indeks.

3.6.2.1.1 Pembalikan *File* (*Inverted File*)

Proses pembalikan file ini dilakukan untuk membandingkan antara bobot kueri dengan indeks bobot istilah, dokumen yang dicari oleh user melalui inputan kueri akan di balikan kepada user.

3.6.2.2 Ukuran Kesamaan (*Similarity Measurement*)

Ukuran kesamaan (*similarity*) istilah dalam model ruang vektor ditentukan berdasarkan *assosiative coefficient* berdasarkan *inner product* dari dokumen vektor dan kueri vektor, dimana *word overlap* menunnjukkan kesamaan istilah. *Inner product* umumnya sudah dinormalisasi. Metode Ukuran kesamaan yang paling populer adalah *cosine coefficient*, yang menghitung sudut antara vektor dokumen dengan vektor kueri.

1. Query Term Weight (tf-idf)

Contoh *Keyterm* dokumen matriks:

	k1	k2	k3	k4	k5	k6	k7	k8	k9	k10
d1	w1,1	w1,2				...				w1,10
d2	w2,1	w2,2				...				w2,10
d3	w3,1	w3,2				...				w3,10
d4	w4,1	w4,2				...				w4,10

Contoh *Keyterm* dokumen matriks:

1) syukur bangsa Indonesia masuk pintu gerbang merdeka
2) milik bahasa satu sekaligus jadi bahasa nasional

Key-term

a. syukur (1)
b. bangsa (1)
c. indonesia (1)
d. masuk (1)
e. pintu (1)
f. gerbang (1)
g. merdeka (1)
h. milik (1)
i. bahasa (2)
j. satu (1)
k. sekaligus (1)
l. jadi (1)
m. nasional (1)

Berdasarkan dokumen *keyterm* matriks :

	syukur	bangsa	indonesia	masuk	pintu	gerbang	merdeka	milik	bahasa	Satu	sekaligus	jadi	nasional
d1	w1,1	w1,2	w1, 3	w1,4	w1, 5	w1, 6	w1,7	w1,8	w1,9	w1,10	w1,11	w1,12	w1,13
d2	w2,1	w2,2	w2,3	w2, 4	w2,5	w2,6	w2,7	w2,8	w2,9	w2,10	w2,11	w2,12	w2,13

Dengan menggunakan bobot yang sudah dihitung :

	syukur	bangsa	indonesia	masuk	pintu	gerbang	merdeka	milik	bahasa	Satu	sekaligus	jadi	nasional
d1	0.3010	0.3010	0.3010	0.3010	0.3010	0.3010	0	0	0	0	0	0	0
d2	0	0	0	0	0	0	0.3010	0.3010	0.6020	0.3010	0.3010	0.3010	0.3010

Contoh bobot *query*: keyterm : {pintu, bahasa} maka bobot *querynya* dapat dihitung sebagai berikut:

$$w_{i,q} = \left(0.5 + \frac{0.5 freq_{i,q}}{max_l freq_{i,q}}\right) X \log \frac{N}{n_i}$$

"pintu" $w_{5,q} = \left(0.5 + \frac{0.5x1}{1}\right) X \log \frac{2}{1} = 0.3010$

"bahasa" $w_{9,q} = \left(0.5 + \frac{0.5x1}{1}\right) X \log \frac{2}{1} = 0.3010$

Maka vektor querynya adalah sebagai berikut:

$\vec{q}$ = (0, 0, 0, 0, 0.3010, 0, 0, 0, 0.3010, 0, 0, 0, 0)

2. Normalisasi

Normalisasi dari vektor *query* :

$$\sqrt{||\overrightarrow{q_1}||.||\overrightarrow{q_t}||} = \sqrt{(0)^2 + (0)^2} + (0)^2 + (0)^2 + (0.3010)^2 + (0)^2 + (0)^2 + (0)^2 +$$

$$(0.3010)^2 + (0)^2 + (0)^2 + (0)^2 + (0)^2 = \underline{0.4257}$$

Normalisasi dari vektor dokumen :

$\|\overrightarrow{d_1}\| = \sqrt{(0.3010)^2 + (0.3010)^2 + (0.3010)^2} + (0.3010)^2 + (0.3010)^2 + (0.3010)^2 + (0)^2 + (0)^2 + (0)^2 + (0)^2 + (0)^2 + (0)^2 + (0)^2 + (0)^2 =$ 0.7372

$\|\overrightarrow{d_2}\| = \sqrt{(0)^2 + (0)^2} + (0)^2 + (0)^2 + (0)^2 + (0)^2 + (0.3010)^2 + (0.3010)^2 + (0.6020)^2 + (0.3010)^2 + (0.3010)^2 + (0.3010)^2 + (0.3010)^2 =$ 0.9518

3. cosine

$$sim(d_{j,q}) = \frac{\overrightarrow{d_j}.\overrightarrow{q}}{\|\overrightarrow{d_j}\| x \|\overrightarrow{q}\|}$$

Untuk $\overrightarrow{d_j}.\overrightarrow{q_1}$ direpresentasikan terhadap matriks, dengan query manjadi kolom :

Contoh:

$$\begin{pmatrix} 0.3010 & 0.3010 & 0.2010 & 0.3010 & 0.3010 & 0.3010 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0.3010 & 0.3010 & 0.6020 & 0.3010 & 0.3010 & 0.3010 & 0.3010 \end{pmatrix} X \begin{pmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0.3010 \\ 0 \\ 0 \\ 0 \\ 0.3010 \\ 0 \\ 0 \\ 0 \\ 0 \end{pmatrix} =$$

$$\begin{pmatrix} 0.0906 \\ 0.1312 \end{pmatrix}$$

Hasilnya seperti dibawab ini :

$\begin{pmatrix} 0.0906 \\ 0.1812 \end{pmatrix}$ → normalisasi → $\begin{pmatrix} 0.2007 \\ 0.4480 \end{pmatrix}$ → rank →

Rank		Dokumen
1	0.4480	D_2
2	0.2887	D_1

3.7 Analisis Kebutuhan Non Fungsional

3.7.1 Analisis Kebutuhan Perangkat Keras

Analisis perangkat keras (*Hardware*) yang digunakan untuk membangun mesin pencari antara lain :

Beberapa komputer dengan spesifikasi minimal sebagai berikut :

a. Processor : Minimal (2.4 GHz).
b. HardDisk : Minimal 80 Gb.
c. Memory : Minimal 2 Gb.
d. Monitor : SVGA 15”
e. VGA Card : Minimal 64 MB
f. Keyboard dan Mouse Standar

3.7.2 Analisis Kebutuhan Perangkat Lunak

Perangkat lunak (*Software*) yang digunakan antara lain :

1. *Windows* XP *Profesional*.
2. Browser *Mozila Firefox 3.0.*
3. *XAMPP local server*

Perangkat lunak (*Software)* pendukung mesin pencari yang akan di bangun adalah menggunakan bahasa pemrograman *PHP* dan Macromedia Dreamweaver sebagai software pembangun, dan di *Online* kan pada jaringan internet.

3.7.3 Analisis Kebutuhan User

Analisis kebutuhan *user* dimaksudkan untuk mengetahui siapa saja *user* yang terlibat dalam sistem temu kembali informasi. Sistem yang sedang berjalan melibatkan semua *user*, siapa saja, kapan dan dimana, seperti di contohkan pada gambar berikut.

3.8 Perancangan Fungsional

Setelah mengetahui berdasarkan masalah diatas, kemudian akan dilanjutkan pada tahap perancangan sistem yang membahas mengenai pembuatan *Flow Chart*, *Basis Data*, dan Struktur Tabel.

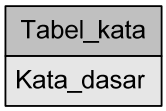
3.9 *Flowchart* Mesin Pencari (*Search Engine*)



Gambar 3.3 *Flowchart* Mesin Pencari (*Search Engine*) File Hyperteks Berbahasa Indonesia

3.10 Basis Data

Pada tahapan ini, membuat rancangan tabel-tabel yang digunakan pada basis data (*database*) beserta atribut-atribut pada setiap tabel nya. Berikut ini adalah gambar rancangan tabel-tabel pada basis data (*database*) mesin pencari (*search engine*) file hyperteks berbahasa Indonesia :



Gambar 3.4 Tabel kata dasar

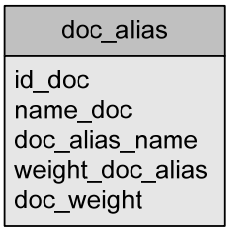
Tabel kata digunakan untuk menyimpan daftar kata dasar yang dibutuhkan pada saat proses proses *stemming* berlangsung.

Pembuatan *database doc_alias dan database term_index* untuk pengindeksan dilakukan dengan menggunakan listing program (*coding*), listing program tersebut adalah sebagai berikut :

```
//------------------ tabel doc_alias
$str_sql="CREATE TABLE doc_alias(id_doc int unsigned NOT NULL PRIMARY KEY ,
name_doc varchar( 255 ) NOT NULL,doc_alias_name varchar( 25 ) NOT NULL,weight_doc_alias varchar( 25) NOT NULL,
doc_weight decimal(10,5) not null default 0 )"
$result=mysql_query($str_sql) or die("Ada Error Pada Pembuatan Tabel doc_alias dengan Pesan : " . mysql_error());
while($file=readdir($dir))
{
  if($file!="." && $file!="..")
  {
    $ekstensi=strtolower(substr($file,-4));
        if($ekstensi=="html" || $ekstensi==".htm")
        {
          $str_sql="insert into doc_alias values($counter,\"$file\",\"doc$counter\",\"w_doc$counter\",0)";
          $result=mysql_query($str_sql) or die ("Gagal Dalam Menyimpan Nama Alias File - Pesan Error : " . mysql_error() );
          $counter++;
        }
  }
}
$doc_alias=array();
$doc_weight_name=array();
cek_exist_table("term_index");

//---------------- algo II relasi tabel
$str_sql="select * from doc_alias";
$result=mysql_query($str_sql) or die("Error Dalam Algo II");
if($result)
{
  $i=0;
  while ($rec=mysql_fetch_row($result))
  {
    $doc_alias[$i]=$rec[2];
        $doc_weight_name[$i]=$rec[3];
        $i++;
  }
}
else
  echo "File HTML Tidak Terdapat Pada Host Tempat Search Engine";

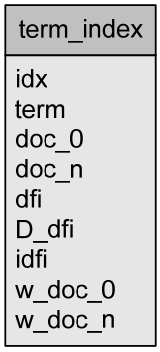
//------------------ tabel term_index
$str_sql="create table term_index(idx int auto_increment primary key, term varchar(30) not null ";
for($i=0;$i<count($doc_alias);$i++)
{
  $str_sql=$str_sql . ", $doc_alias[$i] int unsigned not null default 0 " ;
}
$str_sql=$str_sql . ", dfi int unsigned not null default 0, D_dfi decimal(8,2) not null default 0,idfi decimal(10,5) not null default 0 ";
for($i=0;$i<count($doc_weight_name);$i++)
{
  $str_sql=$str_sql . ", $doc_weight_name[$i] decimal(10,5) not null default 0 " ;
}
$str_sql=$str_sql . ")";
$result=mysql_query($str_sql) or die ("Error Dalam Algo III - Pesan Error : " . mysql_error());
```



Gambar 3.5 Tabel *doc_alias*

Tabel *doc_alias* digunakan untuk menyimpan daftar file-file hiperteks yang terdapat pada media penyimpanan, dan menyimpan hasil perhitungan vektor setiap dokumen. Yang akan dijelaskan setiap atributnya sebagai berikut :

1. Atribut *Id_doc* : kolom pemberian penomoran pada setiap dokumen yang ditemukan dan penomoran dimulai dari 0.
2. Atribut *name_doc* : kolom yang menampung nama-nama dokumen hiperteks yang berhasil terbaca.
3. Atribut *doc_alias_name* : kolom pemberian alias *id_doc* dan *name_doc*, yang nantinya akan digunakan oleh tabel *term_index* untuk menyimpan banyaknya frekuensi istilah yang terkandung dalam satu dokumen (tfi).
4. Atribut *weight_doc_alias* : kolom pemberian nama untuk menyimpan bobot istilah yang nanti digunakan pada tabel *term_index*.
5. Atribut *doc_weight* : kolom yang digunakan untuk menyimpan hasil perhitungan bobot setiap dokumen atau normalisasi vektor dokumen.



Gambar 3.6 Tabel *term_index*

Tabel *term_index* digunakan untuk menyimpan istilah-istilah yang terkandung pada seluruh file-file hiperteks, dan hasil perhitungan bobot istilah. Yang akan dijelaskan setiap atributnya sebagai berikut :

1. Atribut *idx* : kolom penomoran untuk semua istilah atau kata dari kumpulan dokumen.
2. Atribut *term* : kolom istilah atau kata dari kumpulan dokumen
3. Atribut *doc_0…doc_n* : kolom yang diambil dari *record* tabel *doc_alias*, digunakan untuk menyimpan banyaknya frekuensi istilah yang terkandung dalam satu dokumen (tfi).
4. Atribut *dfi* : kolom untuk menyimpan banyaknya dokumen dimana istilah atau kata muncul di dalamnya (dfi).
5. Atribut *D_dfi* : kolom untuk menyimpan hasil pembagian jumlah banyaknya dokumen (D) dengan banyaknya dokumen dimana istilah atau kata muncul di dalamnya (dfi).
6. Atribut *idfi* : kolom untuk menyimpan hasil perhitungan $idf_i = \log_{10}(D/df_i)$.

7. Atribut *w_doc_0…w_doc_n* : kolom yang diambil dari *record* tabel *doc_alias*, digunakan untuk menyimpan bobot istilah atau kata pada setiap dokumen yang mengandung kata tersebut.

3.11 Struktur Tabel

Dalam membangun database yang baik, tabel yang akan dibuat harus didefinisikan terlebih dahulu struktur tabelnya. Dengan memperhatikan hal diatas maka dalam merancang Aplikasi Sistem Temu Kembali Informasi File Hiperteks Berbahasa Indonesia, tabel yang akan digunakan kedalam program aplikasi ini harus terlebih dahulu didefinisikan struktur tabelnya. Pada tahap perancangan struktur tabel, tabel dibuat berdasarkan kumpulan atribut atau entitas-entitas. Berikut ini merupakan struktur tabel yang akan digunakan dalam Program Aplikasi Sistem Temu Kembali Informasi File Hiperteks Berbahasa Indonesia :

Tabel 3.8 Tabel Kata dasar

No	Nama Field	Tipe	Ukuran	Keterangan
1	kata_dasar	varchar	30	

Tabel 3.9 Tabel *doc_alias*

No	Nama Field	Tipe	Ukuran	Keterangan
1	id_doc	int	10	
2	name_doc	varchar	255	
3	doc_alias_name	varchar	25	
4	weight_doc_alias	varchar	25	
5	doc_weight	Decimal	10,5	

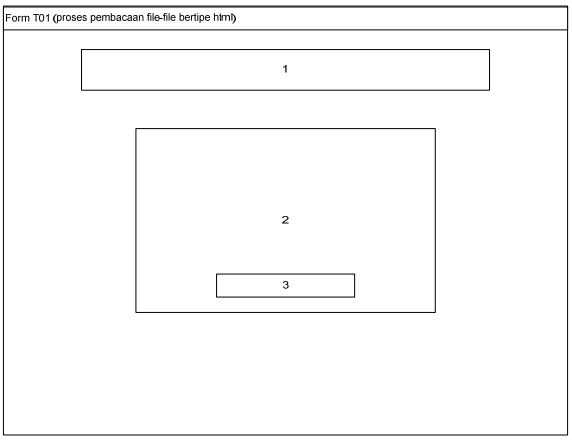
Tabel 3.10 Tabel *term_index*

No	Nama Field	Tipe	Ukuran	Keterangan
1	idx	int	11	
2	term	varchar	30	
3	doc_0...doc_n	int	10	
4	dfi	int	10	
5	D_dfi	decimal	8,2	
6	idfi	decimal	10,5	
7	w_doc_0…w_doc_n	decimal	10,5	

3.12 Perancangan Antar Muka

3.12.1 Perancangan Form

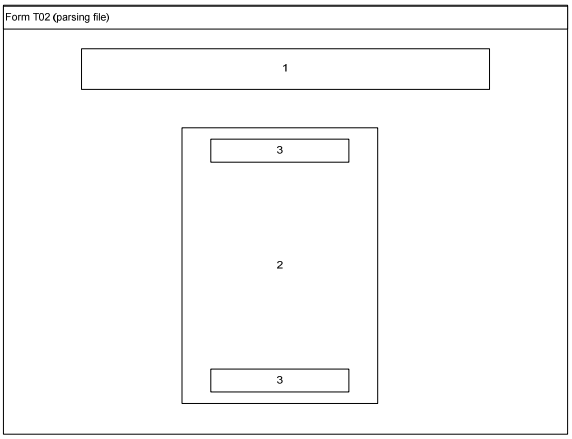
Berikut ini merupakan tampilan beberapa perancangan antar muka yang akan digunakan kedalam Sistem Temu Kembali Informasi dalam pengindeksan dan Pencarian File Hiperteks Berbahasa Indonesia, antara lain :



Gambar 3.7 form T01(pembacaan file-file bertipe html)

Dari gambar 3.7 dapat dijelaskan sebagai berikut :

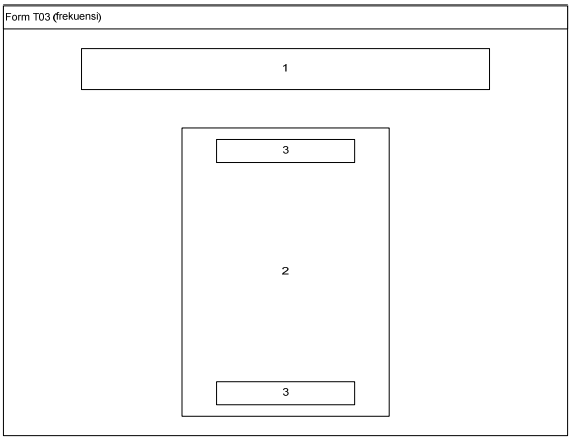
1. *Label*, proses pembacaan isi direktori yang bertipe hiperteks
2. *Textbox*, yang menunjukkan file-file yang bertipe hiperteks
3. Tombol, untuk melakukan proses *parsing* dan menuju ke form T02 (*parsing* file).



Gambar 3.8 form T02 (*parsing file*)

Dari gambar 3.8 dapat dijelaskan sebagai berikut :

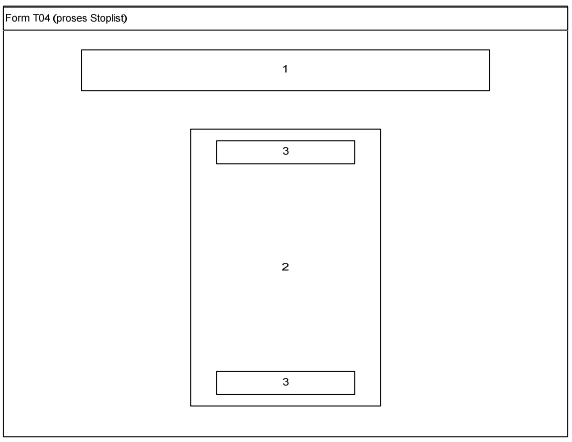
1. *Label*, *parsing* file html
2. *Textbox*, yang menunjukkan kata atau *term* yang sudah diparsing
3. Tombol, untuk melakukan proses penghilangan kata atau *term* yang sama berikut frekuensi kemunculannya dan menuju ke form T03 (frekuensi).



Gambar 3.9 T03 (frekuensi)

Dari gambar 3.9 dapat dijelaskan sebagai berikut :

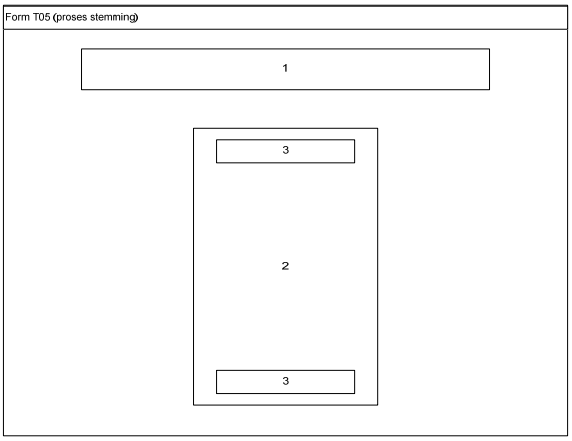
1. *Label*, perhitungan frekuensi setiap kata atau *term* yang sama dan penghilangan kata atau *term* yang sama.
2. *Textbox*, yang menunjukkan kata atau *term* yang sama dan frekuensi kemunculan setiap katanya.
3. Tombol, untuk melakukan proses *Stoplist* dan menuju form T04 (proses *Stoplist*).



Gambar 3.10 T04 (proses *Stoplist*)

Dari gambar 3.10 dapat dijelaskan sebagai berikut :

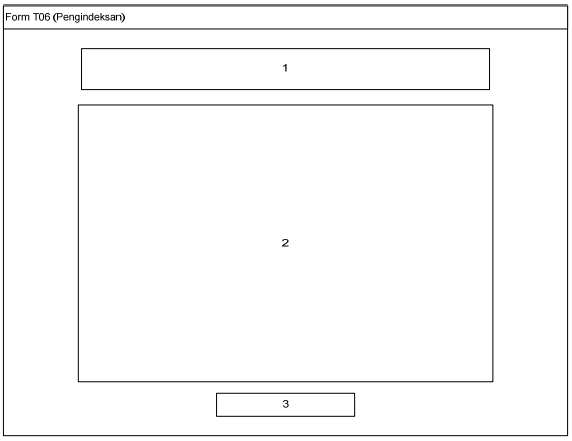
1. *Label*, penghilangan kata atau *term* yang termasuk *Stoplist*.
2. *Textbox*, yang menunjukkan kata atau *term* yang dihilangkan berdasarkan daftar *Stoplist* dan frekuensi kemunculannya.
3. Tombol, untuk melakukan proses *Stemming* dan menuju form T05 (proses *Stemming*).



Gambar 3.11 T05 (proses *Stemming*)

Dari gambar 3.11 dapat dijelaskan sebagai berikut :

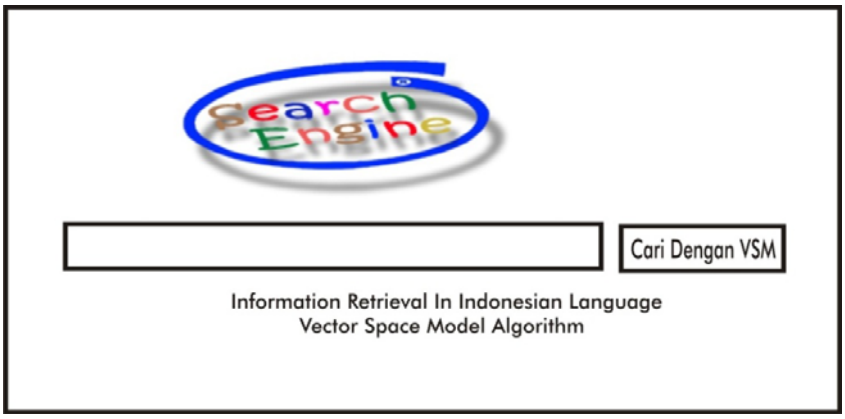
1. *Label*, Proses *Stemming*
2. *Textbox*, yang menunjukkan kata atau *term* yang dihilangkan imbuhannya dan menjadi kata dasar.
3. Tombol, untuk melakukan proses pengindeksan dan menuju form T06 (Pengindeksan).



Gambar 3.12 T06 (Pengindeksan)

Dari gambar 3.12 dapat dijelaskan sebagai berikut :

1. *Label*, Pengindeksan
2. Hasil akhir berupa indeks yang menampilkan kata dasar, banyaknya dokumen, kemunculan kata atau *term* dari setiap dokumen, bobot dokumen, dan bobot setiap kata atau *term* dari setiap dokumen.
3. Tombol *navigator* untuk melihat hasil selanjutnya..

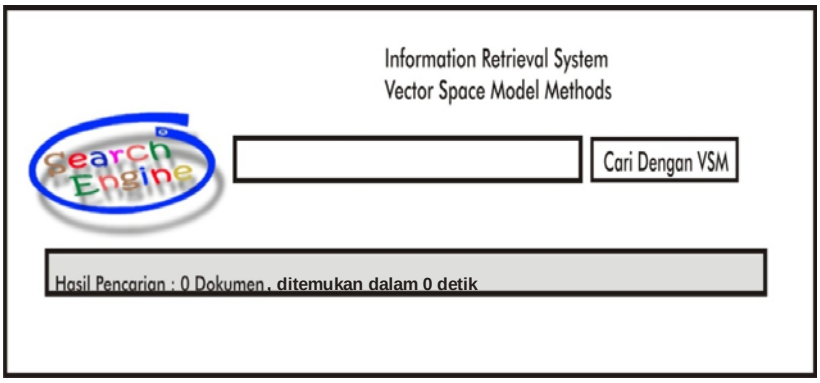


Gambar 3.13 Form Utama *Search Engine*
Sistem Temu Kembali Informasi

Dari gambar 3.13 dapat dijelaskan sebagai berikut :

***Textbox* Enter Keyword** : Untuk memasukkan keyword yang akan kita cari.

Tombol Cari Dengan VSM : Tombol untuk melakukan pencarian, dimana pencarian dilakukan dengan teknik vektor space model dan hasil pencarian akan diperlihatkan di form hasil.



Gambar 3.14 Form Hasil Search Engine
Sistem Temu Kembali Informasi

Dari gambar 3.14 dapat dijelaskan sebagai berikut :

***Textbox* Request User** : Untuk memasukkan keyword yang akan kita cari.

Tombol Cari Dengan VSM : Tombol untuk melakukan pencarian, dimana pencarian dilakukan dengan teknik vektor space model

Hasil Pencarian : Program penutup dan sekaligus berperan untuk memperlihatkan hasil pencarian berdasarkan kata kunci yang dimasukkan oleh *user*. Program ini adalah bagian terpenting dalam *Search Engine*.