BAB III
ANALISIS DAN PERANCANGAN SISTEM

3.1 Analisis Sistem

Pada bagian analisis sistem akan dilakukan proses analisis yang bertujuan untuk mempelajari dan menganalisa kebutuhan sistem yang akan dibuat sehingga dapat dilakukan perancangan sistem dengan kriteria dan perangkat-perangkat yang ditentukan. Analisis dan perancangan berfungsi untuk mempermudah, memahami dan menyusun perancangan pada bab selanjutnya. Dari proses analisis akan didapat alur proses dari sistem temu kembali informasi yang akan di buat.

3.1.1 Analisis Masalah

Penelitian ini menggunakan data surat sekretariat dinas kesehatan kabupaten gresik untuk digunakan sebagai objek yang akan dilakukan *preprocessing*. Data surat ini berupa catatan yang nantinya akan di lakukan *text-preprocessing* pada deskripsi. Untuk pencarian surat di sekretariat dinas kesehatan kabupaten gresik ini sebelumnya sudah dibuatkan sistem informasi yang di gunakan untuk menyimpan surat dan dapat di gunakan untuk mencari kembali surat yang sudah dimasukan ke *database*. Pencarian surat itu masih memiliki kelemahan, yaitu surat yang ditampilkan dari hasil pencarian tidak sesuai dengan yang di inginkan pengguna ketika pengguna salah kurang tepat dalam menginputkan *query*. Oleh karena itu permasalahan yang akan di teliti adalah membuat sistem pencarian yang dapat menampilkan surat yang sesuai dengan inputan *query* dan dokumen yang memiliki keterkaitan dengan *query* yang dimasukan pengguna.

3.1.2 Kebutuhan Pembuatan Sistem

A. Kebutuhan Perangkat Lunak

Perangkat Lunak (*Software*) adalah program-program yang digunakan untuk menjalankan sistem perangkat keras, diantaranya adalah sistem operasi, bahasa pemrograman dan program aplikasi. Dalam pembuatan sistem temu kembali informasi pencarian surat berbasis *dekstop* di sekretariat dinas kesehatan kabupaten gresik diperlukan perangkat-perangkat lunak yang sangat mendukung agar dapat mencapai hasil yang sempurna dari sistem tersebut.

Adapun perangkat lunak yang dibutuhkan dalam pembangunan sistem tersebut adalah sebagai berikut :

1. Sistem Operasi Windows 7/10

   Windows 7/10 sebagai sistem operasi yang digunakan untuk menerapkan sistem temu kembali informasi pencarian surat di kesekretariatan dinas kesehatan kabupaten gresik sebagai bahasa pemrograman berbasis *dekstop* dan sekaligus *compiler*nya.
2. MySQL yog Server 9.0.2

   Sebagai *database* server
3. Sublime Text 3.0

   Sublime merupakan *software text editor* yang akan digunakan untuk pembuatan sistem informasi ini.

B. Kebutuhan Perangkat Keras

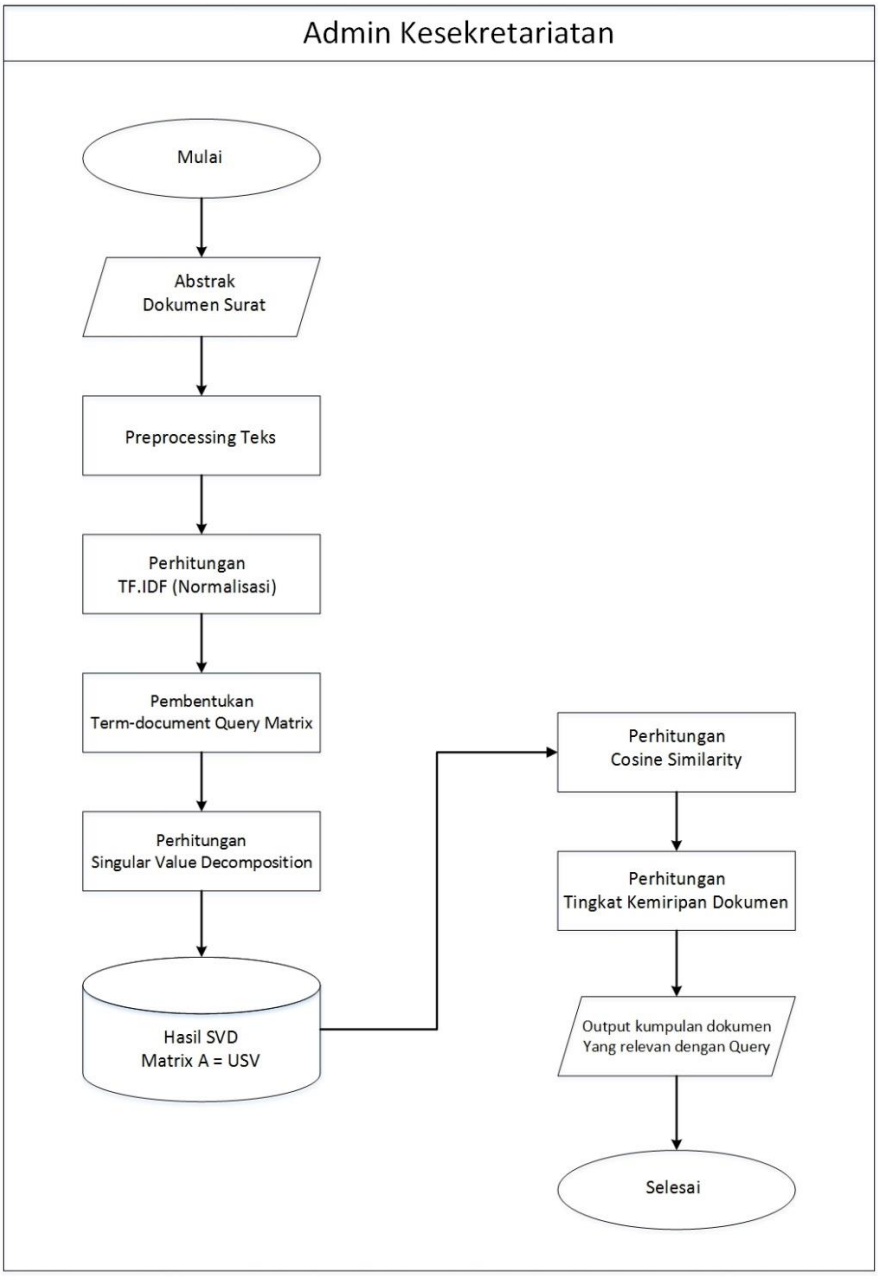
Sistem perangkat keras (*Hardware*) adalah komponen-komponen pendukung kinerja dari sistem komputer. Komponen-komponen yang dapat dipakai untuk menjalankan sistem informasi pengolahan surat ini adalah sebagai berikut.

1. Processor
2. RAM 2 GB
3. Monitor
4. Hardisk minimal 40 GB atau lebih
5. Mouse
6. Keyboard.

3.1.3 Perancangan Sistem

Sistem yang dibangun adalah sistem pencarian surat menggunakan *topic modeling*. Tujuan dari sistem ini mengukur kemiripan dan relevansi untuk menemukan surat yang sesuai dengan *query*. Secara umum sistem ini melalui tahapan *preprocessing atau indexing*, yaitu pada tahapan *pre-processing* mencakup *stop words removal , tokenizing, filtering*, dan *stemming*. Kemudian dilanjutkan dengan membangun *term*-document matrix dengan menempatkan hasil pre-processing ke dalam baris. hasil *Term*-document matrix akan di lakukan reduksi

dimensi dengan menggunakan *singular value decomposition* yang bertujuan untuk memperkecil nilai kompleksitas dalam pemrosesan *term-document matrix*. Dari hasil *singular value decomposition* tadi akan di bandingkan dengan vektor dokumen dan *query* yang telah di inputkan oleh pengguna, antara vektor *query* dan dokumen dapat di hitung koefisien *similarity* dengan menggunakan metode *cosine similarity*. Dari hasil perhitungan koefisien similarity dapat di tuliskan urutan dokumen yang paling dekat dengan *query* yang di inputkan oleh pengguna. Berikut adalah *flowchart* sistem pencarian surat yang dijelaskan pada Gambar 3.1



Gambar 3.1 *Flowchart* Sistem Pencarian Surat

Berikut penjelasan flowchart sistem pencarian surat Gambar 3.1 :

1. Sistem mengambil daftar abstrak dokumen yang ada dalam *database* dan meneriman inputan *query* dari pengguna.
2. Teks abstrak dan *query* yang di dapat di lakukan *preprocessing* teks.
3. *Term* atau teks hasil *preprocessing* di lakukan di beri bobot kemunculan pada tiap dokumen.
4. Sistem membentuk *term-document matrix* dari hasil *term* yang sudah di beri bobot.
5. Matriks yang terbentuk di dekomposisi dengan menggunakan algoritma *singular value decomposition*.
6. Hasil dari perhitungan SVD di simpan dalam *database* untuk di gunakan dalam perhitungan *cosine similarity.*
7. Sistem melakukan perhitungan kemiripan antara dokumen dengan *query* yang di inputkan pengguna.
8. Sistem menghasilkan *output* berupa kumpulan dokumen yang relevan dengan *query.*

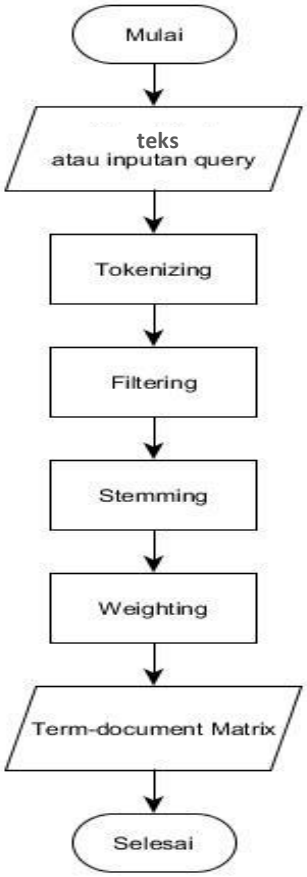
3.2 Preprocessing

Di dalam sistem dilakukan *preprosesing* terhadap teks abstrak diantaranya *tokenizing, filtering, stemming dan weighting*. Berikut diagram alir proses *preprocessing* dalam proses pengolahan isi abstrak dan inputan *query* dapat di lihat pada gambar 3.2.

Proses yang terjadi dalam tahapan *perprocessing* yaitu:

1. Teks abstrak dan *query* di baca oleh sistem dan dilakukan *preprocessing*
2. Tahapan selanjutnya yaitu proses *tokenizing* yaitu memecah kalimat menjadi *term*, tahapan ini juga dilakukan proses untuk menghilangkan karakter-karakter tertentu seperti tanda baca dan mengubah semua token ke bentuk huruf kecil (*lower case*).
3. Proses *filtering* adalah pemilihan kata dan pencocokan kata terhadap *stopword* kemudian menghilangkan kata yang ada dalam *stopword*.
4. *Stemming* adalah proses pemisahan kata-kata yang berimbuhan sehingga berubah menjadi kata dasar.

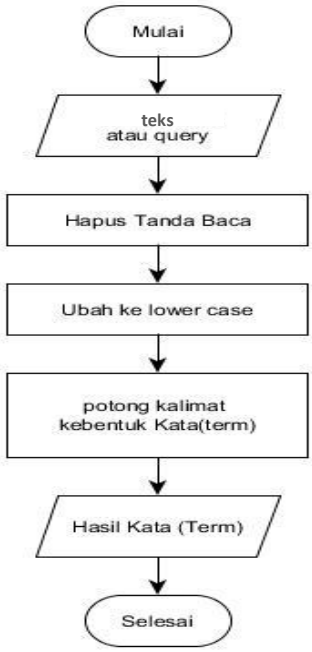
5. Tahapan selanjutnya adalah *weighting* yaitu memberi nilai atau bobot pada setiap *term* dan membentuk *term* menjadi *term-document matrix* yang nantinya di gunakan untuk tahapan selanjutnya.



Gambar 3.2 Diagram alir Preprocessing

3.2.1 *Tokenizing*

Tokenizing adalah proses untuk memisahkan deretan kata di dalam kalimat, paragraf atau halaman menjadi token atau potongan kata. Tahapan ini juga menghilangkan karakter-karakter tertentu seperti tanda baca dan mengubah semua token ke bentuk huruf kecil.



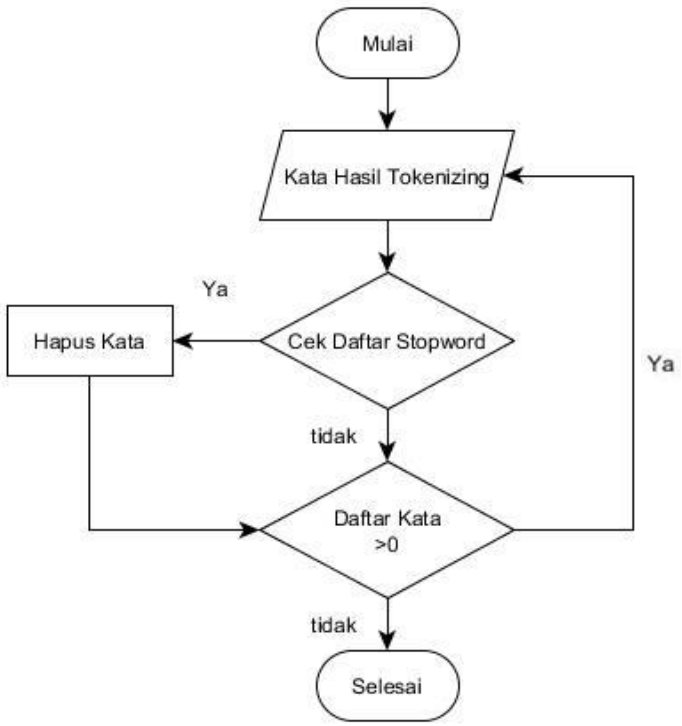
Gambar 3.3 Diagram alir Tokenizing

Proses yang terjadi dalam proses *tokenizing* yaitu :

1. Sistem mengambil teks abstrak atau *query* dalam *database.*
2. Tanda baca dan karakter yang tidak diperlukan pada teks yang diambil dari *database* dihapus.
3. Semua kata diubah ke huruf kecil.
4. Semua kalimat di pisahakan menjadi perkata dengan menggunakan spasi sebagai indikator pemisah.
5. Sistem menghasilkan kumpulan kata yang telah terpisah dari teks awal dan akan disimpan didalam *database* untuk diproses pada tahap selanjutnya.

3.2.3 *Filtering*

Filtering adalah pemilihan kata yang tidak diperlukan dan pencocokan kata terhadap *stopword* untuk di hilangkan dari *term.* Pada proses ini kata-kata yang dianggap tidak mempunyai makna seperti kata sambung akan dihilangkan. Pada proses *filtering* biasanya digunakan daftar *stopword* yang tersimpan dalam suatu basis data, yang nantinya digunakan sebagai acuan penghilangan kata.



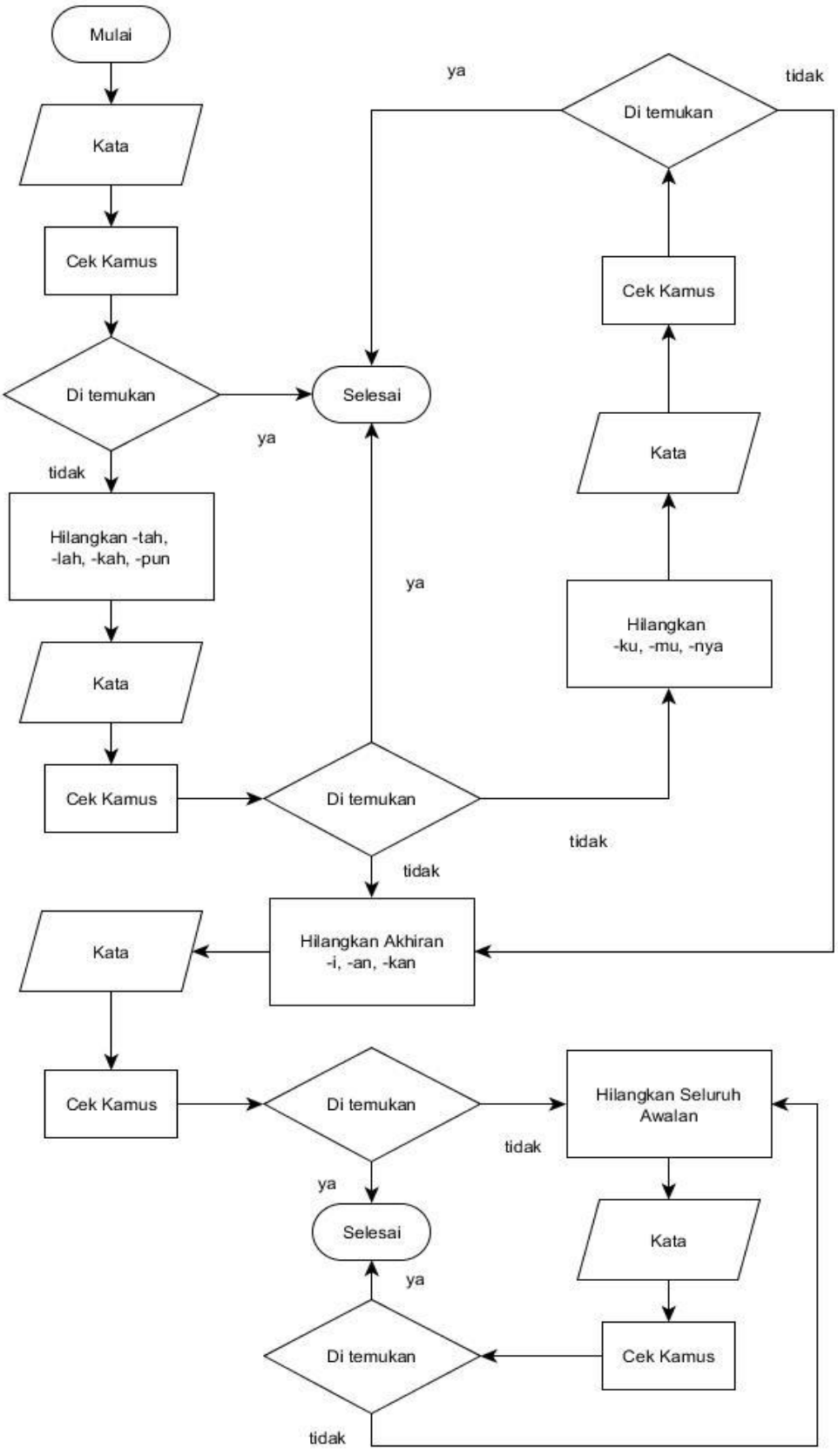
Gambar 3.4 Diagram alir Filtering

Proses yang terjadi pada gambar 3.4 dalam tahapan filtering yaitu :

1. Sistem akan mengambil kata hasil *tokenizing*
2. Sistem melakukan pengecekan kata yang ada dalam *stopword* dalam *database* apakah ada kata yang sama atau tidak, jika ada yang sama maka sistem akan melakukan penghapusan, jika tidak ada maka kata tersebut akan disimpan untuk proses *stemming*.
3. Sistem akan melakukan pengecekan berulang-ulang sampai tidak ada kata yang sama.

3.2.4 Stemming

Stemming adalah pengubahan kata ke bentuk kata dasar atau penghapusan imbuhan. Dalam sistem ini digunakan algoritma *stemming* Nazief Andriani yang merupakan pembentukan kata dasar yang dapat memperkecil hasil indeks tanpa menghilangkan makna. Diagram alir algoritma *stemming* bahasa indonesia Nazief-Adriani dapat di lihat pada gambar 3.5 berikut.



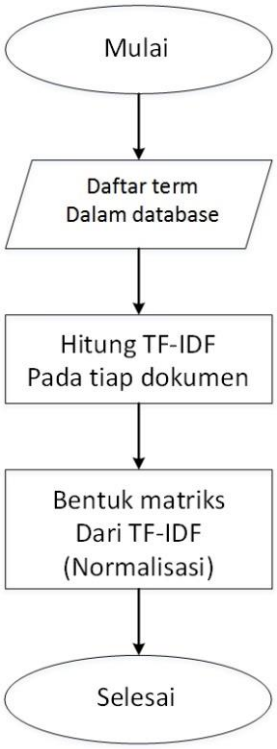
Gambar 3.5 Diagram Alir Algoritma Stemming Nazief-Adriani

Proses tahapan stemming Nazief dan Adriani yang terjadi pada gambar 3.4 yaitu :

1. Kata dicari di dalam daftar kamus, bila kata tersebut ditemukan di dalam kamus maka dapat diasumsikan kata tersebut adalah kata dasar sehingga algoritma dihentikan.
2. Bila kata di dalam langkah pertama tidak ditemukan di dalam kamus, maka diperiksa apakah sufiks tersebut yaitu sebuah partikel (“-tah”,”lah”,”-pun” atau “-kah”). Bila ditemukan maka partikel tersebut dihilangkan.
3. Pemeriksaan dilanjutkan pada kata ganti milik(“-ku”,”-mu”,”nya”).bila ditemukan maka kata ganti tersebut dihilangkan.
4. Memriksa akhiran (“-i”,”-an”,”-kan”). Bila ditemukan maka akhiran tersebut dihilangkan. Hingga langkah ke-4 dibutuhkan ketelitian untuk memeriksa apakah akhiran “-an” merupakan hanya bagian dari akhiran “-kan” dan memeriksa lagi apakah partikel (“-tah”,”-lah”,”-pun” atau “-kah”) dan kata ganti milik (“-ku”,”-mu”,”-nya”) yang telah dihilangkan pada langkah 2 dan 3 bukan merupakan bagian dari kata dasar.
5. Memeriksa awalan (“se-“,”ke-“,”di-“,”te-“,”be-“,”pe-“,”me-“). Bila ditemukan, maka awalan tersebut dihilangkan. Pemeriksaan dilakukan dengan berulang mengingat adanya kemungkinan multi *prefix*. Langkah ke-5 ini juga membutuhkan ketelitian untuk memeriksa kemungkinan peluluhan awalan, perubahan *prefix* yang disesuaikan dengan huruf awal kata dan aturan kombinasi *prefix-sufix* yang diperbolehkan.
6. Setelah menyelesaikan semua langkah diatas dengan sukses, maka algoritma akan mengembalikan kata dasar yang ditemukan.

3.2.5 Weighting *Term*-document Matrix

Pembobotan pada *latent semantic analysis* menggunakan *term frequency*(TF) dan *inverse document frequency*(IDF) untuk memberi bobot atau nilai pada tiap kata yang sudah di lakukan *preprocessing* pada teks abstrak dan *query*. Dari hasil *term frequency* yang sudah dilakukan pembobotan dapat di bentuk *term-document matrix* yang nantinya akan digunakan untuk proses selanjutnya. Proses *weihgting term-documen matrix* dapat di lihat pada gambar 3.6 berikut.



Gambar 3.6 Diagram Alir Weighting *Term*-document Matrix

Proses pembobotan dan pembentukan *term-document matrix* yang terjadi pada gambar 3.6 adalah sebagai berikut.

1. Mengambil *term* yang telah tersimpan dalam *database*.
2. Menghitung jumlah kemunculan suatu kata atau *term* pada tiap dokumen yang ada dalam *database*.
3. Membentuk matriks dari kumpulan dokumen dan *term* yang sudah memiliki nilai frekuensi dan ternormalisasi.

Contoh kasus terdapat 2 dokumen abstrak yang sudah di lakukan *preprocessing* :

Isi D1 :

Berdasarkan surat dari kepala Dinas Kesehatan Di Surabaya No. 443/212/II.1 tanggal 11 Maret 2018 perihal undangan dalam “Seminar Kesehatan Masyarakat” untuk menjadikan masyarakat yang lebih peduli terhadap kesehatan.

Yang bertanda tangan : Kepala Dinas Kesehatan Kabupaten Gresik menugaskan kepada :

Nama : Novi Emawati, Sp.Ok

NIP : 198711162009022003

Jabatan : Spesialis Kedokteran Okupasi

Instansi : Rs. Ibnu Sina

Untuk menjadi nara sumber dalam “Seminar Kesehatan Masyarakat”.

Dari isi dokumen surat 1 dapat di ambil inti sari dari isi surat tersebut kemudian menjadikannya Deskripsi.

Deskripsi D1 : Kurangnya kesadaran masyarakat terhadap kesehatan.

Isi D2 :

Sehubungan dengan adanya penyakit Demam Berdarah Dengue (DBD) yang menjakiti warga desa sasaran kerja Puskesmas Cerme dan agar tidak terjadi wabah atau penyebaran penyakit kami mohon bantuannya Kepada Dinas Kesehatan Kabupaten Gresik untuk dapat memberikan penyemprotan (*Fogging*) di daerah desa sasaran kerja yaitu Desa Cerme Kidul dan Desa Cerme Lor yang berjumlah 2.762 rumah. Untuk mengantisipasi penyebarluasan penyakit pihak Puskesmas telah melakukan penyuluhan tentang penyakit DBD, PSN 3 Plus, dan Penyelidikan Epidemiologi Penyakit DBD.

Demikian surat permohonan ini disampaikan dan besar harapan kami agar permohonan ini dapat direalisasikan.

Dari isi dokumen surat 2 dapat di ambil inti sari dari isi surat tersebut kemudian menjadikannya Deskripsi.

Deskripsi D2 : Pencegahan penyakit demam berdarah.

Dari kumpulan deskripsi dokumen abstrak tersebut akan di bentuk *term-document matrix*, contoh *term-document matrix* dapat dilihat pada tabel 3.1.

Tabel 3.1 Tabel *Term*-document Matrix

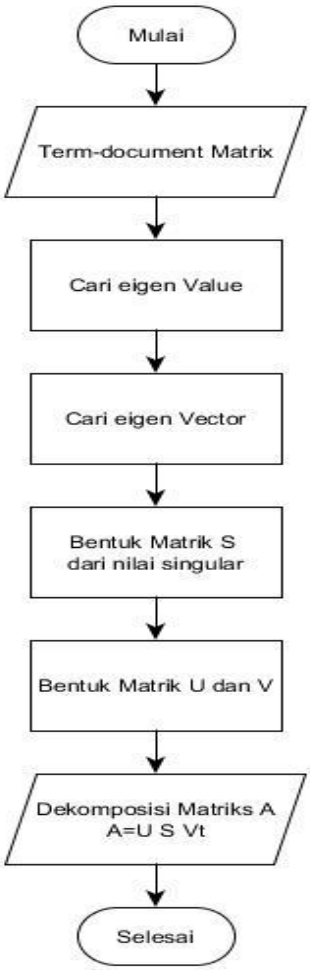
***Term*-Document**	D1	D2	df	idf	tf,idf		Normalisasi	
Kurang	1		1	0,30103	0,30103	0	0,2	0
Sadar	1		1	0,30103	0,30103	0	0,2	0
Masyarakat	1		1	0,30103	0,30103	0	0,2	0
Hadap	1		1	0,30103	0,30103	0	0,2	0
Sehat	1		1	0,30103	0,30103	0	0,2	0
Cegah		1	1	0,30103	0	0,30103	0	0,25
Sakit		1	1	0,30103	0	0,30103	0	0,25
Demam		1	1	0,30103	0	0,30103	0	0,25
Darah		1	1	0,30103	0	0,30103	0	0,25
				Total	1,50515	1,20412		

3.3 Algoritma *Singular Value Decomposition*

Singular Value Decomposition (SVD) merupakan teknik komputasi numerik yang melakukan faktorisasi terhadap sebuah matriks tak nol sehingga diperoleh tiga matriks tak nol. Salah satu matriks yang diperoleh dari proses SVD akan memuat nilai-nilai singular dari matriks asal. Pada penelitian ini di gunakan algoritma *singular value decomposition* untuk mendekomposisi *term-document matrix* yang telah di peroleh dari hasil *preprocessing*. Proses pada SVD bisa di lihat pada gambar 3.6.

Proses dekomposisi matriks awal yang terjadi pada gambar 3.7 adalah sebagai berikut :

1. Mengambil matriks dari *term-document matrix* yang ada pada *database*.
2. Cari nilai eigen dari matriks awal.
3. Mencari vektor eigen.
4. Membentuk matriks S dengan cara mengambil *singular valu*e dari perhitungan nilai eigen.
5. Membentuk matriks U dan V dari hasil kalkulasi normalisasi vektor eigen.
6. Terbentuk 3(tiga) Matriks baru dari hasil dekomposisi matriks awal.



Gambar 3.7 Diagram alir Singular Value Decomposition

Contoh kasus akan diambil dari *term-document matrix* hasil dari *preprocessing* yang ada pada tabel 3.1. langkah utama pada algoritma *singular value decomposition* adalah mendekomposisikan matriks menjadi 3(tiga) matriks lainnya. Contoh perhitungan algoritma *singular value decomposition* adalah sebagai berikut :

$$\text{Matriks A} = \begin{bmatrix} 0{,}2 & 0 \\ 0{,}2 & 0 \\ 0{,}2 & 0 \\ 0{,}2 & 0 \\ 0{,}2 & 0 \\ 0 & 0{,}25 \\ 0 & 0{,}25 \\ 0 & 0{,}25 \\ 0 & 0{,}25 \end{bmatrix}$$

Matriks A di ambil dari hasi pembentukan *term-matrix document*. Matriks A akan di dekomposisi untuk membentuk 3(tiga) matriks lainnya dengan menggunakan SVD. Tahapan perhitungan SVD adalah sebagai berikut :

Hitung A^TA

Matriks $A^TA = \begin{bmatrix} 0,2 & 0 \\ 0 & 0,25 \end{bmatrix}$

Mencari determinan dan persamaan linier dari Matriks A^TA dengan menggunakan rumus :

$$det(A^TA - \lambda I) = 0$$

$$det\begin{bmatrix} 0,2-\lambda & 0 \\ 0 & 0,25-\lambda \end{bmatrix} = 0$$

$$\lambda^2 - 0,45.\lambda + 0,05 = 0$$

$$\lambda_1 = 0,2$$

$$\lambda_2 = 0,25$$

Dari perhitungan di atas di peroleh nilai eigen 0,2 dan 0,25 dari perhitungan determinan dan persamaan linier. Dari perhitungan di atas nilai eigen akan di gunakan untuk menghitung vektor eigen. Contoh perhitungan vektor eigen adalah sebagai berikut :

1. $\lambda_1 = 0,2$

$$(A^TA - \lambda I).X = 0$$

$$\begin{bmatrix} 0 & 0 \\ 0 & 0,05 \end{bmatrix}\begin{bmatrix} X1 \\ X2 \end{bmatrix}=0$$

$$0.X1 + 0.X2 = 0$$
$$0.X1 + 0,05.X2 = 0$$

$$X = \begin{bmatrix} 0,4472 \\ 0 \end{bmatrix}$$

2. $\lambda_2 = 0{,}25$

$$(A^T A - \lambda \mathrm{I}).X = 0$$

$$\begin{bmatrix} -0{,}05 & 0 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} X1 \\ X2 \end{bmatrix} = 0$$

$$-0{,}05.X1 + 0.X2 = 0$$

$$0.X1 + 0.X2 = 0$$

$$X = \begin{bmatrix} 0 \\ 0{,}5 \end{bmatrix}$$

Dari perhitungan di atas di dapat nilai dari panjang setiap vektor dan di gunakan untuk membagi setiap nilai dari vektor eigen. Dari perhitungan tersebut di dapatkan matriks V sebagai berikut :

$$\text{Matriks V} = \begin{bmatrix} 0{,}4472 & 0 \\ 0 & 0{,}5 \end{bmatrix}$$

Matriks U di dapatkan dengan menggunakan rumus AA^T dan perhitungan yang sama dengan perhitungan yang di gunakan matriks V. Dari perhitungan tersebut di dapatkan matriks U sebagai berikut :

$$\text{Matriks U} = \begin{bmatrix} 0{,}4472 & 0{,}0000 \\ 0{,}4472 & 0{,}0000 \\ 0{,}4472 & 0{,}0000 \\ 0{,}4472 & 0{,}0000 \\ 0{,}4472 & 0{,}0000 \\ 0{,}0000 & 0{,}5 \\ 0{,}0000 & 0{,}5 \\ 0{,}0000 & 0{,}5 \\ 0{,}0000 & 0{,}5 \end{bmatrix}$$

Proses selanjutnya adalah mencari matriks S yang telah di peroleh dari perhitungan nilai eigen. Matriks S sendiri merupakan vektor S diagonal dari akar nilai eigen. Berikut ini adalah matriks S dari akar nilai eigen :

$$\text{Matriks S} = \begin{bmatrix} \sqrt{0,2} & 0 \\ 0 & \sqrt{0,25} \end{bmatrix} = \begin{bmatrix} 0,4472 & 0 \\ 0 & 0,5 \end{bmatrix}$$

Kemudian akan dilanjutkan perhitungan untuk mencari bobot setiap *term* yang nanti akan digunakan untuk menghitung bobot *Query* pada masing-masing topik, maka dilakukan perkalian terhadap matriks U dan matriks S, dari perhitungan tersebut didapatkan matriks Uk, Sk sebagai berikut.

term	**Uk x Sk**	
Kurang	**0,223607**	**0**
Sadar	**0,223607**	**0**
Masyarakat	**0,223607**	**0**
Hadap	**0,223607**	**0**
Sehat	**0,223607**	**0**
Cegah	**0**	**0,223607**
Sakit	**0**	**0,223607**
Demam	**0**	**0,223607**
Darah	**0**	**0,223607**

Selanjutnya dilakukan perhitungan dari bobot-bobot setiap *term* untuk merepresentasikan *query* terhadap topik dengan *query* pertama "Kurangnya kesadaran terhadap kesehatan" lalu di *stemming* per-kata menjadi kurang,sadar,hadap,sehat. Kemuadian *query* kedua "Pencegahan penyakit" lalu di *stemming* per-kata menjadi cegah,sakit. Dan didapatkan nilai yang merepresentasikan terhadap topik Kesehatan Masyarakat(KM) dan Pencegahan dan Pengendalian Penyakit(PPP) sebagai berikut.

	Kurang	Sadar	Hadap	Sehat		
KM	0,4472	0,4472	0,4472	0,4472	=	1,788856
PPP	0	0	0	0	=	0

	Cegah	Sakit		
KM	0	0	=	0
PPP	0,5	0,5	=	1

3.4 Metode Cosine Similarity

Metode *cosine similarity* merupakan metode untuk menghitung kesamaan antara dua buah objek yang dinyatakan dalam dua buah vector dengan menggunakan *keywords* (kata kunci) dari sebuah dokumen sebagai ukuran. Pada penelitian ini di gunakan metode *cosine similarity* untuk menentukan tingkat kemiripan dari setiap dokumen berdasarkan pada *query* yang di masukan. Dengan menggunakan Matriks V yang telah dilakukan perhitungan pada tahap SVD sebagai berikut.

$$\text{Matriks V} = \begin{bmatrix} 0{,}4472 & 0 \\ 0 & 0{,}5 \end{bmatrix}$$

Vektor *query* dan vektor dokumen yang di dapat akan di gunakan untuk menghitung tingkat kemiripan *query* terhadap dokumen dengan menggunakan *cosine similarity*. Proses perhitungan *cosine similarity* dengan menggunakan *query* "kurangnya kesadaran terhadap kesehatan" dan "pencegahan penyakit" yang akan di bentuk vektor *query*, proses perhitungan *cosine similarity* adalah sebagai berikut:

$$similarity \ = \ cos(\theta) \ = \frac{A + B}{A \bullet B}$$

Deskripsi d1 : kurangnya kesadaran masyarakat terhadap kesehatan

Query : kurang sadar hadap sehat

$$Sim\ (d1) = \frac{0{,}4472 + 0{,}4472 + 0{,}4472 + 0{,}4472}{1{,}7888 \ \ x \ \ 2{,}23607} = \ \frac{1{,}7888}{4{,}0000} = 0{,}4472$$

$$Sim\ (d2) = \frac{0 + 0 + 0 + 0}{0 \ \ x \ \ 0} = \ \frac{0}{0} = 0$$

Deskripsi d2 : Pencegahan penyakit demam berdarah

Query : cegah sakit

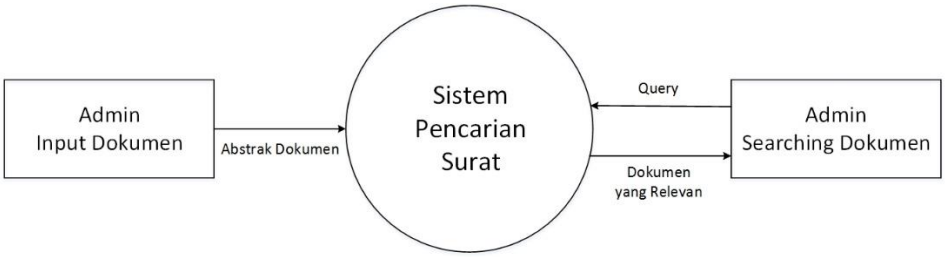
$$Sim\ (d1) = \frac{0 + 0 + 0 + 0}{0 \ \ x \ \ 0} = \ \frac{0}{0} = 0$$

$$Sim\ (d2) = \frac{0{,}2236 + 0{,}2236}{0{,}9000 \ \ x \ \ 0{,}8944} = \ \frac{0{,}4472}{0{,}8049} = 0{,}5$$

Dari perhitungan *cosine similarity* di atas, dapat di tuliskan urutan dokumen terdekat terhadap *query* “kurangnya kesadaran terhadap kesehatan” adalah d1 dan dokumen terdekat terhadap *query* “pencegahan penyakit” adalah d2.

3.5 Diagram Konteks

Berdasarkan folwchart yang telah dijelaskan diatas, maka sistem dapat dijelaskan dengan diagram konteks sebagai berikut:



Gambar 3.8 Diagram Konteks Sistem Pencarian Dokumen Surat

Proses yang terjadi pada gambar 3.8 dalam tahapan diagram konteks sistem pencarian dokumen surat yaitu :

1. Admin menginputkan abstrak dokumen sertifikasi ke *database* system pencarian dokumen surat.
2. Admin menginputkan *query* kedalam sistem, kemudian sistem memberikan *output* berupa kumpulan dokumen yang relevan dengan *query* yang di inputkan admin.

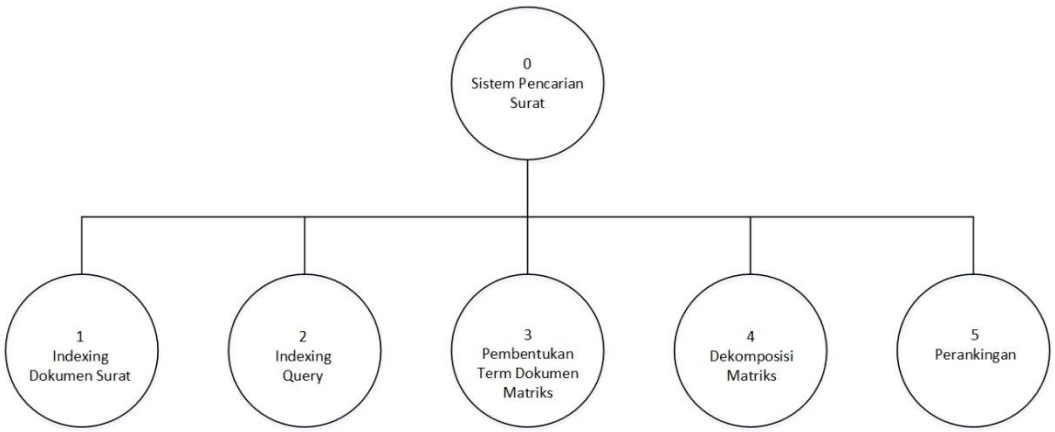
3.6 Diagram Berjenjang

Pembuatan diagram berjenjang pada sistem pencarian dokumen surat berfungsi untuk menggambarkan proses berjalannya sistem. Pada sistem pencarian dokumen surat terdapat 2(dua) level seperti yang terlihat pada gambar 3.9.

Proses yang terjadi pada gambar 3.9 dalam tahapan diagram berjenjang sistem pencarian dokumen surat yaitu :

1. Top Level : Sistem pencarian dokumen surat
2. Level 1 : Merupakan turunan dari top level yang terdiri dari beberapa sub proses
   - Indexing Dokumen

- Indexing *Query*
- Pembentukan *Term*-document Matrix
- Dekomposisi Matrix
- Perankingan

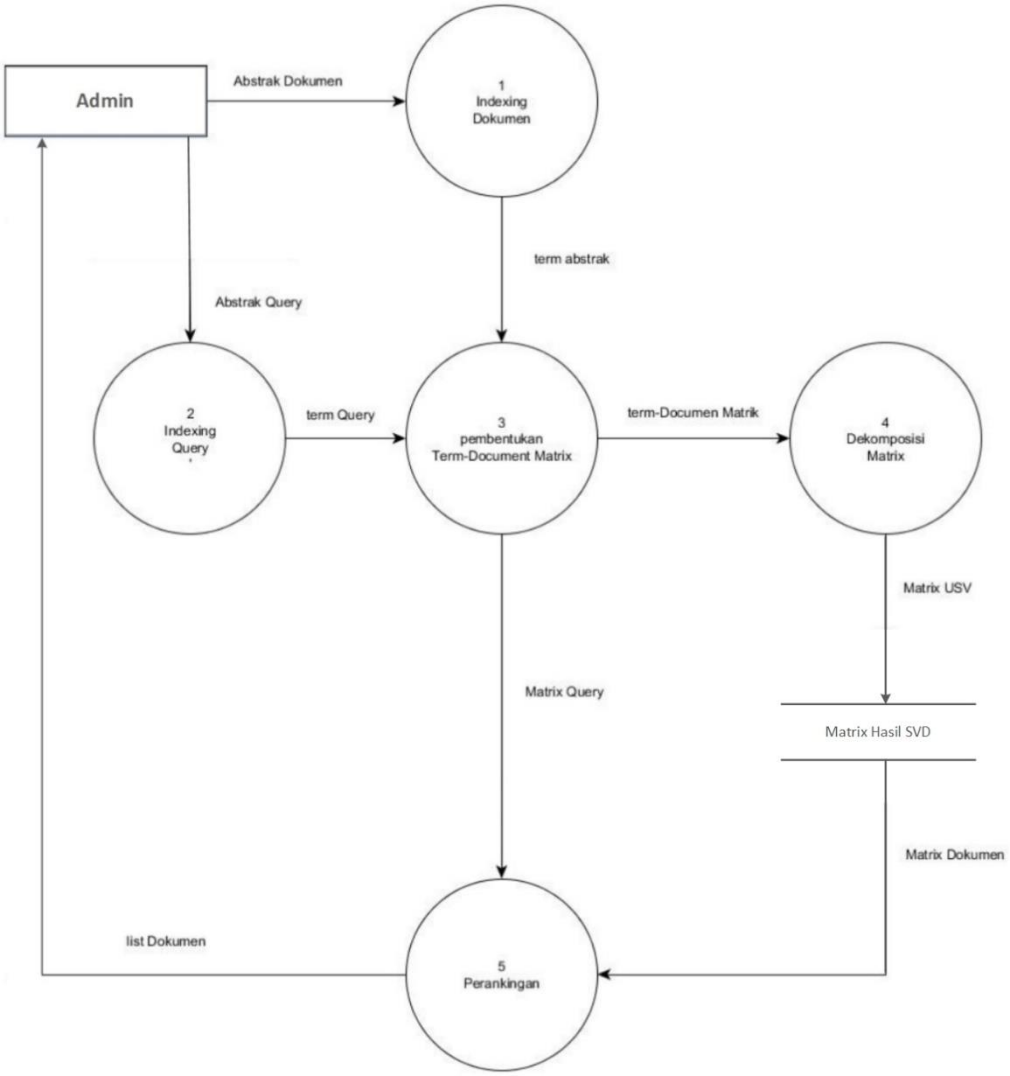


Gambar 3.9 Diagram Berjenjang Sistem Pencarian Dokumen Surat

3.7 Data Flow Diagram (DFD)

Proses yang terjadi pada DFD level 1 yaitu :

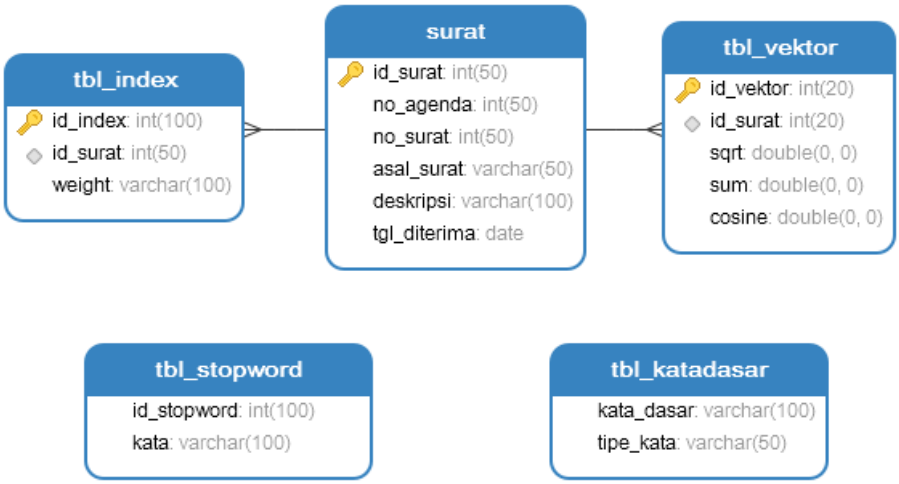
1. Sistem melakukan proses indexing terhadap abstrak dokumen yang telah di masukan oleh admin ke dalam *database* menjadi *term* abstrak.
2. Sistem melakukan proses indexing terhadap abstrak *query* yang telah di inputkan.
3. Sistem membentuk sebuah *term-document matrix* berdasarkan dari proses indexing abstrak dokumen dan *query*.
4. Sistem melakukan proses dekomposisi pada *term-document matrix*, matriks tersebut kemudian di simpan dalam *database*.
5. Sistem melakukan proses ranking dokumen terhadap *query* yang di masukan oleh pengguna. Kemudian memberikan hasil list dokumen yang relevan kepada pengguna.



Gambar 3.10 DFD Sistem Pencarian Dokumen Surat level 1

3.8 Desain *Database*

Database (Basis Data) adalah kumpulan dari data yang berhubungan antara satu dengan yang lainya, tersimpan di perangkat keras komputer dan menggunakan perangkat lunak untuk memanipulasinya. *Database* merupakan salah satu Komponen yang penting dalam sistem komputerisasi, karena *database* merupakan data dalam menyediakan informasi bagi para pengguna.



Gambar 3.11 ERD Sistem Pencarian Dokumen Surat

Berdasarkan gambar 3.11 berikut penjelasan table-tabel yang digunakan dalam pembangunan sistem pencarian dokumen surat :

1. Tabel Surat

   Berfungsi Untuk menyimpan dokumen-dokumen surat yang telah di beri deskripsi pada dokumen untuk di gunakan dalam *preprocessing* teks pada deskripsi dokumen tersebut.

   Tabel 3.2 Struktur tabel surat

Nama_Field	**Tipe**	**Keterangan**
Id_surat	Int(50)	
No_agenda	Int(50)	Nomer agenda surat
No_surat	Int(50)	Nomer surat
Asal_surat	Varchar(50)	Asal surat
Deskripsi	Varchar(100)	Deskripsi surat
Tgl_diterima	date	Tanggal surat diterima

2. Tabel *Stopword*

   Tabel stopword berfungsi untuk menyimpan daftar *stoplist* seperti: “yang”,”di”,”dan” yang nantinya digunakan untuk proses *filtering* pada tahapan *preprocessing*.

Tabel 3.3 Struktur Tabel Stopword

Nama_Field	**Tipe**	**Keterangan**
Id_stopword	Int(100)	
kata	Varchar(100)	Isi kata

3. Tabel Kata Dasar

   Tabel kata dasar berisi kumpulan kata dasar berdasarkan kamus bahasa Indonesia untuk digunakan dalam proses *stemming* setelah proses filtering pada tahapan *preprocessing.*

   Tabel 3.4 Struktur Tabel Kata Dasar

Nama_Field	**Tipe**	**Keterangan**
Kata_dasar	Varchar(100)	
Tipe_kata	Varchar(50)	Jenis kata dasar

4. Tabel Index

   Tabel index berfungsi untuk penyimpanan daftar *term* abstrak hasil *Preprocessing.*

   Tabel 3.5 Struktur Tabel Index

Nama_Field	**Tipe**	**Keterangan**
Id_index	Int(100)	
Id_surat	Int(50)	Id surat
weight	Varchar(100)	Bobot kemunculan kata

5. Tabel Vektor

   Tabel vektor berfungsi dalam perhitungan *similarity* tiap dokumen terhadap *query*, untuk mempermudah dalam perhitungan *cosine similarity* dan perankingan dalam mendapatkan hasil dokumen yang memliki nilai *similarity* tertingggi.

   Tabel 3.6 Struktur Tabel Vektor

Nama_Field	**Tipe**	**Keterangan**
Id_vektor	Int(20)	
Id_surat	Int(20)	Id surat

Lanjutan **Tabel 3.6** Struktur Tabel Vektor

Nama_Field	**Tipe**	**Keterangan**
Sqrt	Varchar(50)	Panjang vector surat
Sum	Varchar(50)	Jumlah dari perkalian *term query* dan dokumen
Cosine	Varchar(50)	Hasil perhitungan

3.9 Desain *Interface*

3.9.1 Halaman Awal

Halaman awal ini merupakan tampilan awal ketika *user* mulai menjalankan sistem pencarian. Terdapat beberap menu pada halaman awal. *Desain interface* halaman ini adalah seperti pada gambar 3.12.

Gambar 3.12 Rancangan Halaman Awal

Keterangan :

1. Logo Dinas dan nama sistem.
2. Label *Home* Menu.
3. Tombol *input* surat.
4. Tombol cari/kelola surat.
5. Tombol menu admin.

3.9.2 Halaman Input Dokumen Surat

Halaman *input* dokumen surat merupakan halaman untuk memasukan kumpulan dokumen abstrak yang nantinya di gunakan untuk *preprocessing* teks. *desain interface* halaman ini adalah seperti pada gambar 3.13.

Gambar 3.13 Rancangan Halaman Input Dokumen Surat

Keterangan :

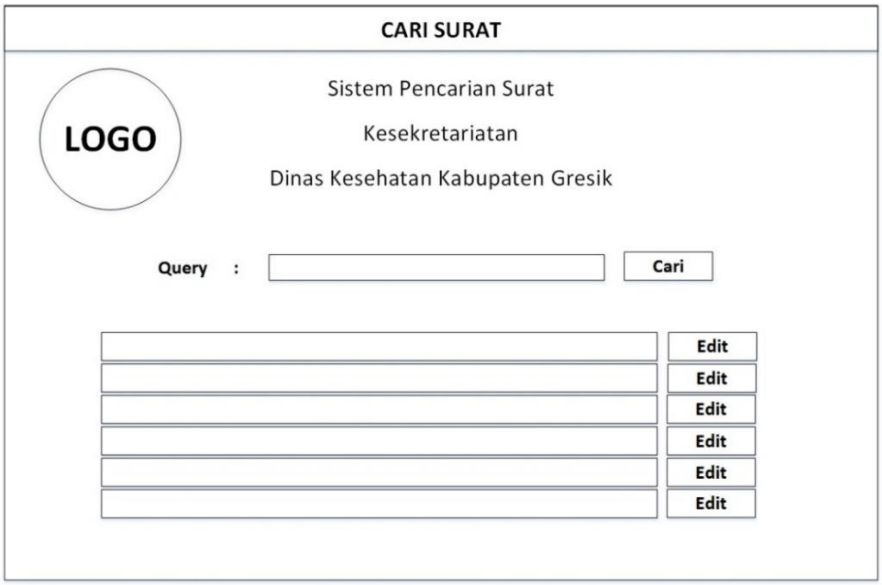
1. Logo Dinas dan nama sistem.
2. *Field* untuk mengisi data sistem.

3.9.3 Halaman Pencarian

Halaman pencarian merupakan halaman yang berisi menu untuk menampilkan hasil dari pencarian dokumen surat. *Desain interface* halaman ini adalah seperti pada gambar 3.14.

Keterangan :

1. Logo Dinas dan nama sistem.
2. Tombol dan *field* cari surat.
3. Tampilan dan isi deskripsi dokumen surat.



Gambar 3.14 Rancangan Halaman Pencarian Dokumen Surat

3.9.4 Halaman Admin

Halaman *admin* merupakan halaman yang berisi menu-menu khusus yang hanya bisa di gunakan dan di akses oleh admin. *Desain interface* halaman ini adalah seperti pada gambar 3.15.

Gambar 3.15 Rancangan Halaman Admin Menu

Keterangan :

1. Logo Dinas dan nama sistem.
2. Tombol Admin Setting.
3. Tombol Manajemen Surat.

3.10 Skenario Pengujian

Untuk mengukur evaluasi kinerja sistem temu kembali informasi, yaitu menggunakan *Recall* dan *Precission*. *Recall* adalah rasio antara dokumen yang ditemukembalikan relevan dari seluruh dokumen yang seharusnya ditemukembalikan relevan yang ada di dalam sistem, sedangkan *precision* adalah rasio dokumen relevan yang berhasil ditemukembalikan dari seluruh dokumen yang berhasil ditemu-kembalikan

Tabel 3.7 Parameter Penghitungan *Recall* dan *Precision*

Keterangan	**TRUE**	**FALSE**
Terambil	True Positive (TP)	False Positive (FP)
Tidak Terambil	False Negative (FN)	True Negative (TN)

$$precision = \frac{TP}{TP+FP}$$

$$recall = \frac{TP}{TP+FN}$$

$$accuracy = \frac{TP + TN}{TP+TN+FP+FN}$$

$$F - measure = \frac{2 \times precision \times recall}{precision + recall}$$

Nilai *precision*, *recall*, *accuracy* dan *f-measure* dinyatakan dalam persen. Semakin tinggi keempat nilai tersebut menunjukkan semakin baiknya kinerja sistem. Evaluasi yang akan dilakukan dalam penelitian ini adalah menghitung nilai dari *precision*, *recall*, *accuracy* dan *f-measure* berdasarkan dokumen yang ditemukan oleh sistem. Sedangkan untuk menentukan nilai dari *precision*, *recall*, *accuracy* dan *f-measure* harus didapatkan jumlah dokumen yang relevan terhadap suatu topik dokumen abstrak.