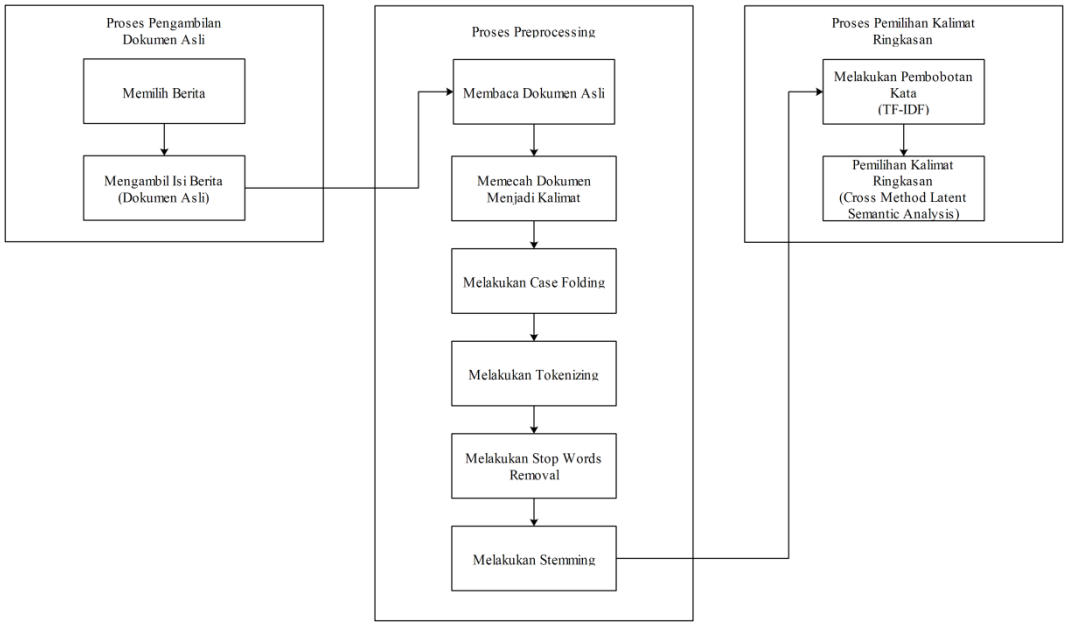
# BAB 3
# ANALISIS KEBUTUHAN ALGORITMA

## 3.1. Analisis Masalah

Analisis masalah adalah suatu gambaran masalah yang diangkat dalam penelitian mengenai simulasi peringkas teks otomatis untuk dokumen tunggal berbahasa Indonesia. Analisis masalah ini menjelaskan proses identifikasi masalah serta evaluasi mengenai sistem peringkas teks otomatis dalam dokumen tunggal berbahasa Indonesia. Berdasarkan uraian pada latar belakang masalah, pada penelitian ini diketahui bahwa penerapan metode LSA yang telah dikembangkan pada peringkas teks otomatis belum di uji untuk meringkas dokumen tunggal berbahsa Indonesia. Oleh karena itu pada penelitian ini akan dibuat sebuah peringkas teks otomatis sebagai alat simulasi dari penerapan metode LSA pada peringkas teks otomatis untuk meringkas dokumen tunggal berbahasa Indonesia.

## 3.2. Analisis Proses

Peringkas teks otomatis yang dibangun mempunyai tiga tahapan proses yang dikerjakan yaitu tahap pengambilan dokumen asli, tahap *preprocessing* dan tahap pemilihan kalimat ringkasan. Tahapan pengambilan dokumen asli merupakan tahapan yang dilakukan untuk mengambil berita dari portal berita yang telah ditentukan untuk dijadikan sebagai berita yang akan di ringkas, tahapan *preprocessing* dilakukan pada keseluruhan isi dari dokumen yang akan di ringkas untuk menghilangkan *noise* yang masih terdapat di dalamnya, sedangkan tahap pemilihan kalimat ringkasan merupakan tahapan pemberian bobot pada hasil dari tahapan *preprocessing* kemudian setalah itu dilakukan proses penyusunan kalimat – kalimat yang di anggap penting menggunakan metode peringkasan yang digunakan. Untuk lebih jelasnya dapat dilihat pada blok diagram pada gambar 3.1.




**Gambar 3.1 Gambaran Umum Peringkas Teks Otomatis**

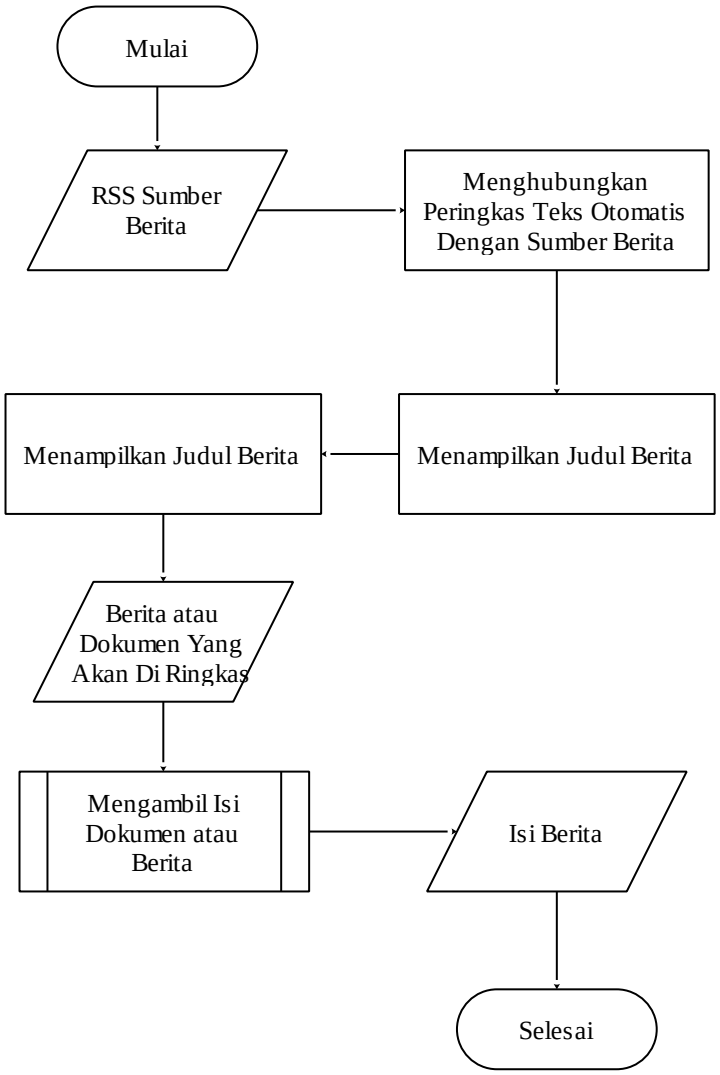
### 3.2.1. Analisis Pengambilan Dokumen Asli

Analisis yang dilakukan pada pengambilan dokumen asli atau dokumen yang akan di ringkas dari peringkas teks otomatis yang dibangun ini menjelaskan proses - proses yang dilakukan untuk mendapatkan dokumen yang akan di ringkas. Data masukan yang digunakan dalam peringkas teks otomatis adalah data dari dokumen tunggal berita berbahasa Indonesia yang di ambil dari portal berita.

Gambaran umum dari proses pemilihan dokumen asli atau dokumen yang akan di ringkas pada peringkas teks otomatis yang akan di bangun adalah :

1. *User* atau pengguna memilih judul berita yang akan di ringkas melalui daftar berita yang ditampilkan oleh peringkas teks otomatis.
2. Peringkas Teks Otomatis melakukan proses pengambilan isi atau konten dari berita yang telah dipilih oleh pengguna kemudian setelah itu peringkas teks otomatis akan melakukan tahapan *preprocessing* pada isi dari berita tersebut untuk menghilangkan noise didalamnya. Tahapan *preprocessing* terdiri dari tahap pemecahan kalimat, *case folding*, *tokenizing*, *stop words removal* dan *stemming* yang akan dibahas pada analisis berikutnya.

Alur atau gambaran dari proses pemilihan dokumen asli pada peringkas teks otomatis dapat dilihat pada gambar 3.2 dibawah :




**Gambar 3.2 Flowchart Pemilihan Dokumen Asli**

Penjelasan dari gambar 3.2 diatas adalah

a. RSS Sumber Berita

   Alamat dari sumber berita atau portal berita yang akan dijadikan sebagai masukan untuk mengambil berita – berita yang terdapat pada portal berita tersebut.

b. Menghubungkan Peringkas Teks Otomatis Dengan Sumber Berita

   Pada tahapan ini peringkas teks otomatis akan meminta daftar dari berita kepada portal berita atau sumber berita sesuai dengan alamat RSS sumber berita yang digunakan.

c. Menampilkan Judul Berita

Peringkas teks otomatis akan menampilkan daftar dari berita yang tersedia pada portal berita yang dapat pengguna pilih untuk di ambil ringkasannya.

d. Memilih Berita Yang Akan Di Ringkas

Pada tahapan ini Pengguna memilih judul berita yang telah disediakan oleh peringkas teks otomatis dari sumber atau portal berita sebagai berita atau dokumen yang akan di ringkas.

e. Berita atau Dokumen Yang Akan Di Ringkas

Pada tahapan ini berita atau dokumen yang telah di pilih oleh pengguna akan dijadikan sebagai *input* atau masukan bagi peringkas teks otomatis. Masukan ini berupa keseluruhan dari halaman berita yang telah di pilih oleh pengguna.

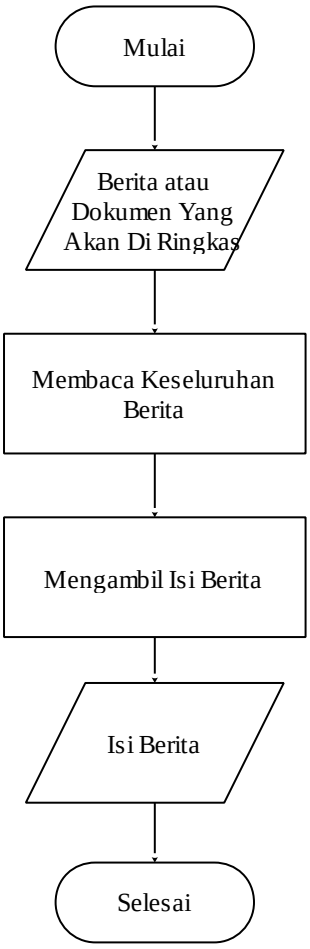
f. Mengambil Isi Dari Berita atau Dokumen

Dokumen yang telah di masukan ke dalam peringkas teks otomatis akan di ambil isi nya yaitu isi berita yang terdapat pada dokumen tersebut karena pada saat dokumen dimasukan kedalam peringkas teks otomatis bukan hanya isi berita saja yang dimasukan tetapi keseluruhan dari halaman berita tersebut.

g. Isi Berita

Isi dari berita yang telah dipilih oleh pengguna.

Sedangkan detil dari *predefined process* Mengambil Isi Dokumen atau Berita pada *flowchart* gambar 3.2 di atas ditunjukan oleh gambar 3.3 dibawah :




**Gambar 3.3 Detil Proses Mengambil Isi Dokumen Atau Berita**

Penjelasan dari gambar 3.2 diatas adalah

a. Berita Atau Dokumen Yang Akan Di Ringkas

   Dokumen yang telah di pilih oleh pengguna akan dijadikan sebagai *input* atau masukan bagi peringkas teks otomatis. Masukan ini berupa keseluruhan dari halaman berita yang telah di pilih oleh pengguna.

b. Membaca Keseluruhan Isi Berita

   Pada tahapan ini berita yang telah dipilih oleh pengguna seelumnya di baca secara keseluruhan oleh peringkas teks otomatis dan di simpan ke dalam sebuah variabel sebagai penampung isi berita tersebut.

c. Mengambil Isi Berita

Dokumen berita yang dipilih oleh pengguna tidak hanya berisi artikel berita saja tetapi seluruh informasi yang dimiliki oleh dokumen tersebut sama seperti pada saat dokumen tersebut ditampilkan pada portal berita seperti daftar rekomendasi berita lain, iklan dan sebagainya. Oleh karena itu pada saat pengambilan isi berita, informasi atau isi berita diambil berdasarkan tag yang menandakan bahwa tah tersebut memiliki artikel berita yang akan di ambil, sehingga keluaran dari proses ini hanya berupa isi dari artikel berita yang dipilih saja

d. Isi Berita

Isi dari berita yang telah dipilih oleh pengguna.

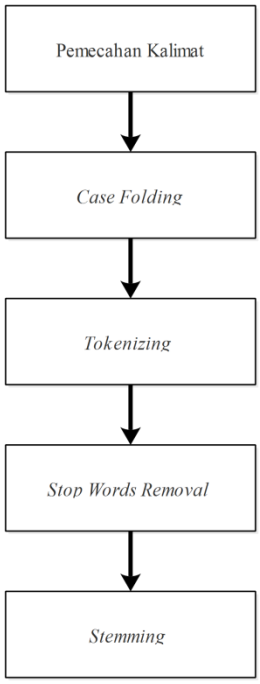
Sedangkan data atau kalimat yang akan digunakan untuk pengujian metode *preprocessing* dan metode pengambilan ringkasan ditunjukan oleh tabel 3.1 di bawah :

**Tabel 3.1 Tabel Kalimat Pengujian**

| Kalimat Uji |
|---|
| Tindak Pidana Korupsi. |
| Korupsi merupakan tindakan kejahatan. |
| Pelaku korupsi harus dihukum. |

### 3.2.2. Analisis Preprocessing

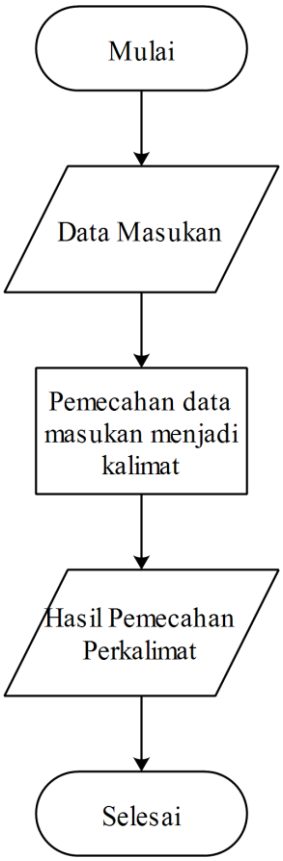
Analisis preprocessing ini menjelaskan praproses yang dilakukan pada data masukan untuk mendapatkan kata-kata atau masukan yang bersih sehingga tidak akan mengandung noise yang akan menganggu hasil untuk proses selanjutnya. Tahapan dari praproses sistem peringkas teks berita dalam bahasa Indonesia dapat dilihat pada gambar 3.4.




**Gambar 3.4 Tahapan Preprocessing**

### 3.2.2.1.Pemecahan Data Masukan menjadi Kalimat

Pada tahap pemecahan kalimat adalah memecah data dari dokumen masukan menjadi kumpulan kalimat-kalimat berdasarkan tanda titik " . ",tanda tanya " ? ", dan tanda seru "!" sebagai pemisah *(delimiter)* untuk memotong string dokumen. Untuk alur dari proses pemecahan kalimat yang diterapkan pada saat pembangunan peringkas teks otomatis dapat dilihat pada gambar 3.5 dibawah:




**Gambar 3.5 Alur Pemecahan Kalimat**

Penjelasan dari alur pemecahan data masukan menjadi kumpulan kalimat – kalimat dari alur gambar 3.5 diatas adalah sebagai berikut :

a. Data Masukan

   Pada tahap ini dilkaukan proses pembacaan data RSS yang diambil dari portal sumber oleh peringkas teks otomatis.

b. Pemecahan data masukan menjadi kalimat

   Pada tahap pemecahan data masukan menjadi kalimat – kalimat digunakan fungsi preg_split("*delimiter*", data_yang_akan_dipecah) dimana fungsi ini akan memecah atau memtong data yang dimasukan berdasarkan dengan *delimiter* yang disertakan. Pada fungsi ini *delimiter* berfungsi sebagai tanda yang akan digunakan untuk memotong dokumen masukan. Delimiter yang digunakan adalah tanda titik “ . ”,tanda tanya “ ? ”, dan tanda seru “!” yang di

tuliskan dalam format *regular expression* pada saat pembangunan aplikasi. Format regular expression yang digunakan untuk *delimiter* adalah /[.?!].

c. Hasil Pemecahan kalimat

Keluaran dari proses pemecahan data masukan menjadi perkalimat dimasukan kedalam array.

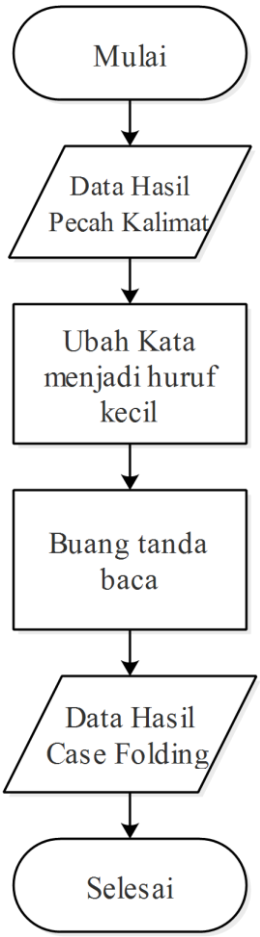
Tabel 3.2 menampilkan hasil proses pemecahan dokumen menjadi kumpulan kalimat-kalimat sebagai berikut :

**Tabel 3.2 Hasil Pemecahan Kalimat**

| No | Kalimat |
|---|---|
| D1 | Tindak Pidana Korupsi |
| D2 | Korupsi merupakan tindakan kejahatan |
| D3 | Pelaku korupsi harus dihukum |

### 3.2.2.2. Case Folding

Pada tahap ini dilakukan proses *case folding* atau penyamaan besar kecilnya huruf pada setiap kata serta pembuangan tanda baca yang masih terdapat pada kalimat. Untuk alur dari proses *case folding* yang diterapkan pada saat pembangunan peringkas teks otomatis dapat dilihat pada gambar 3.6 dibawah:




**Gambar 3.6 Alur Case Folding**

Penjelasan dari alur *case folding* gambar 3.6 diatas adalah sebagai berikut :

a. Data Hasil Pecah Kalimat

   Pada proses ini dilakukan pembacaan terhadap array dari hasil proses pemecahan kalimat untuk dilakukan proses case folding.

b. Ubah kata menjadi huruf kecil

   Pada tahap ini dilakukan proses *case folding* untuk mengubah data yang dihasilkan dari proses pemecahan kalimat menjadi data dengan besar huruf yang sama. Pada proses ini semua string yang terdapat pada data hasil pemecahan kalimat di ubah menjadi huruf kecil menggunakan fungsi strtolower(“variabel _pemecahan_kaliamat”) tetapi karena variabel yang digunakan pada fungsin strtolower bertipe array maka untuk proses *case folding* digunakan fungsi array_map untuk mengakses setiap data yang

terdapat pada aray yang digunakan. Fungsi akhir yang digunakan dalam proses *case folding* adalah array_map("strtolower", variabel_pemecahan_kalimat);

c. Buang tanda baca

Pada tahap ini hasil dari proses penyamaan huruf menjadi huruf kecil diolah kembali untuk menghilangkan *noise* berupa tanda baca dan angka yang masih terdapat pada huruf. Pada proses ini digunakan fungsi preg_replace(“delimiter”, “ pengganti”,kata), delimiter yang digunakan adalah \d yang menandakan digit[0-9] dan \W yang menandakan simbol atau karakter yang tidak termasuk kedalam huruf dan pengganti merupakan karakter yang digunakan untuk mengganti karakter yang sesuai dengan delimiter, pada kasus ini digunakan spasi sebagai karakter pengganti. Dengan delimiter tersebut fungsi akan menghapus angka dan simbol yang terdapat pada kata yang dimasukan sebagai parameter.

d. Data Hasil Case Folding

Output atau keluaran dari proses *case folding* di masukan kedalam array.

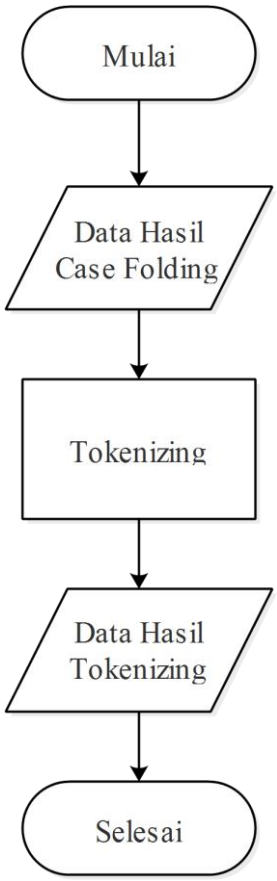
Hasil dari proses *case folding* yang dilakukan dapat dilihat pada tabel 3.3 di bawah:

**Tabel 3.3 Hasil Case Folding**

| No | Kalimat |
|---|---|
| D1 | tindak pidana korupsi |
| D2 | korupsi merupakan tindakan kejahatan |
| D3 | pelaku korupsi harus dihukum |

### 3.2.2.3. Tokenizing

Proses tokenizing kata yaitu menghilangkan karakter pemisah (delimiter) yang menyusunnya berupa karakter spasi. Proses ini merupakan proses pemecahan kalimat menjadi kata. Untuk alur dari proses *case folding* yang diterapkan pada saat pembangunan peringkas teks otomatis dapat dilihat pada gambar 3.7 dibawah:




**Gambar 3.7 Alur Tokenizing**

Penjelasan dari alur *tokenizing* gambar 3.7 diatas adalah sebagai berikut :

a. Data Hasil *Case Folding*

Pada proses ini dilakukan pembacaan terhadap array dari hasil proses *case folding* untuk dilakukan proses *tokenizing.*

b. *Tokenizing*

Pada proses ini digunakan fungsi preg_split untuk memisah data masukan menjadi kata – kata dan menghapus tanda bacaan secara bersamaan sebagai berikut : preg_split("delimiter", implode(variabel_hasil_casefolding)). Pada tahap ini delimiter yang digunakan dalam format regular expression adalah /[\s] dimana \s adalah format yang menandakan spasi. Fungsi implode digunakan pada variabel_hasil_casefolding untuk mengubah variabel yang tadinya array menjadi string karena fungsi preg_split tidak menerima variabel bertipe array sebagai parameter.

c. Data Hasil *Tokenizing*

Output atau keluaran dari proses *tokenizing* di masukan kedalam array.

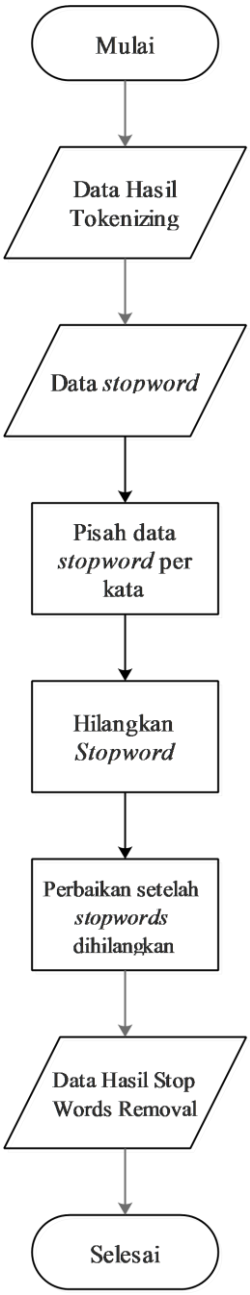
Hasil dari proses tokenizing dapat dilihat pada tabel 3.4 dibawah ini :

**Tabel 3.4 Hasil Proses Tokenizing**

| **Kata Hasil Proses Tokenizing** | | | |
|---|---|---|---|
| dihukum | harus | kejahatan | korupsi |
| korupsi | korupsi | merupakan | pelaku |
| pidana | tindak | tindakan | |

### 3.2.2.4. Stop Words Removal

Proses *Stop Word Removal* dilakukan untuk menghapus kata-kata yang tidak penting yang masih terdapat pada data yang akan di olah. Setelah melalui proses *tokenizing data* kemudian akan melalui proses *stop words removal.* Alur dari proses *stop words removal* yang dilakukan oleh peringkas teks otomatis ditampilkan pada gambar 3.8 dibawah ini:




**Gambar 3.8 Alur Stop Words Removal**

Penejelasan dari alur *stop words removal* yang ditunjukan pada gambar 3.8 diatas adalah sebagai berikut :

a. Data Hasil Tokenizing

   Pada proses ini dilakukan pembacaan terhadap array dari hasil proses *tokenizing* untuk dilakukan proses *stop words removal.*

b. Baca data stopwords

Pada tahap ini dilkaukan proses pembacaan data dengan format *file* .txt yang yang mengandung *stop words* yang akan digunakan dalam simulasi. Untuk proses pembacaan *file* pada peringkas teks otomatis digunakan fungsi file_get_contents("nama_file.txt") untuk membaca *file* bertipe txt yang akan digunakan peringkas teks otomatis.

c. Pisah data stopwords per kata

Pada tahap ini dilakukan proses pemisahan stopwords menjadi kata – kata dan memasukan stopwords kedalam variabel penampung bertipe array. Untuk pemisahan stopwrods menjadi kata – kata digunakan fungsi preg_split("delimiter", file_stopwords). Dimana fungsi ini akan memotong data yang terdapat pada *file stopwords* menjadi *array* berdasarkan *delimiter* yang digunakan. Delimiter yang digunakan dalam *format regular* expression adalah \s yang menandakan karakter spasi.

d. Hilangkan Stop Words

Pada tahap ini dilakuka proses penghilangan *stop words* yang masih terdapat pada data hasil *tokenizing.* Fungsi yang digunakan dalam proses ini adalah array_diff(array1,array2), fungsi ini akan membandingkan array 1 dan array 2 yang telah dimasukan pada parameter fungsi tersebut. Perbandingan yang dilakukan adalah membandingkan kata / nilai yang sama yang terdapat pada kedua array tersebut kemudian fungsi ini akan menghapus kata/nilai yang terdapat pada array 1 yang mempunyai nilai yang sama dengan nilai yang terdapat pada array 2, kemudian nilai yang tidak dihapus pada array 1 akan dijadikan sebagai keluaran. Sehingga parameter yang digunakan dalam fungsi tersebut adalah array_diff(hasil_tokenizing,data_stopwords).

e. Perbaikan Setelah Stopwords Dihilangkan

Karenan pada proses sebelumnya dilakukan penghapusan pada variabel array hasil *tokenizing* maka kunci pada variabel array tersebut akan berlubang. Sehingga perlu dilakukan proses perbaikan pada array tersebut untuk memudahkan proses simulasi. Fungsi yang digunakan adalah array_values(hasil_hapus_stopwords). Fungsi ini akan memberikan nilai

kunci baru pada array yang terdapat pada parameter fungsi ini sehingga array tersebut akan memiliki nilai kunci yang tidak berlubang.

f. Data Hasil *Stop Words Removal*

*Output* atau keluaran dari proses *stop words removal* di masukan kedalam array.

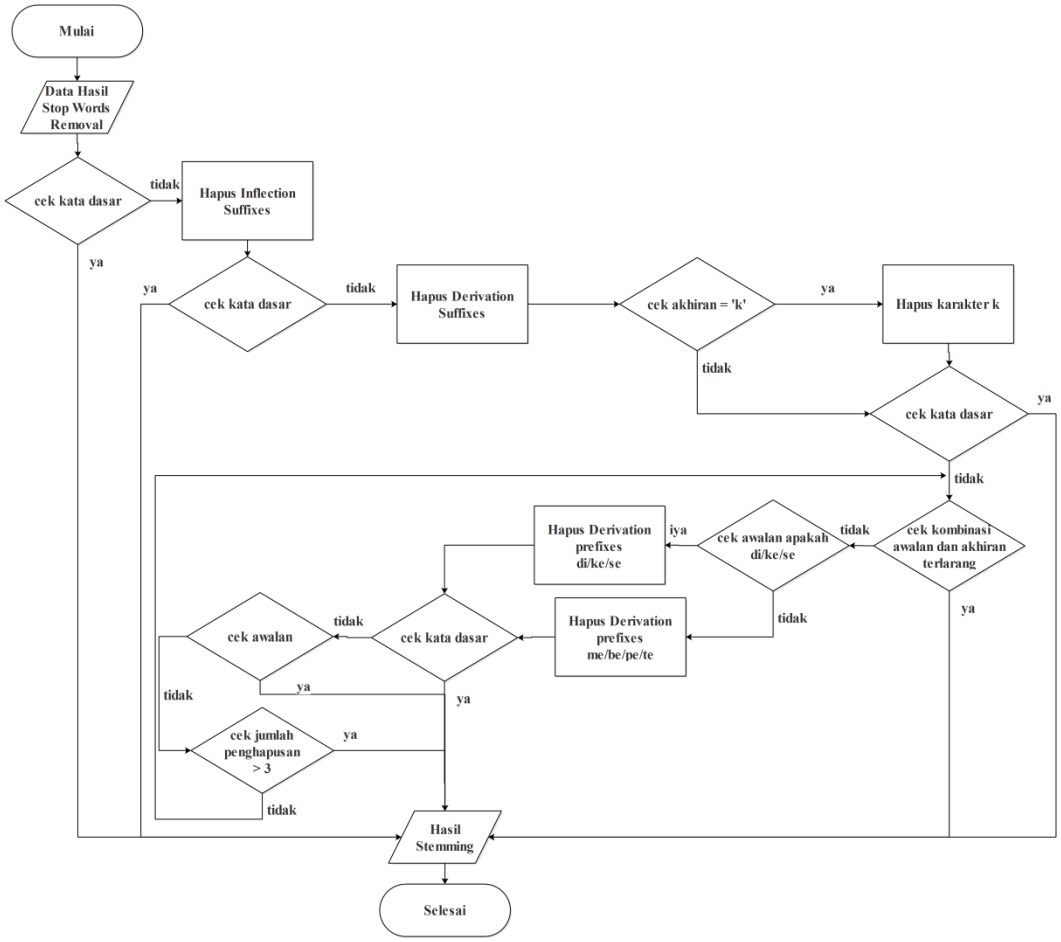
Hasil dari proses stop words removal dapat dilihat pada tabel 3.5 dibawah :

**Tabel 3.5 Hasil Stop Words Removal**

| **Kata Hasil Stop Words Removal** | | | |
|---|---|---|---|
| dihukum | kejahatan | korupsi | korupsi |
| korupsi | pelaku | pidana | tindak |
| tindakan | | | |

### 3.2.2.5.Stemming

Proses stemming yang dilakukan adalah untuk menentukan kata dasar yang terdapat dari sebuah kata yang telah melalui tahap paproses sebelumnya. Algoritma *stemming* yang digunakan adalah algoritma Nazief dan Ardiani yang merupakan algoritma *stemming* yang dikhusukan untuk diterapkan pada bahasa Indonesia. Untuk alur dari proses *stemming* dapat dilihat pada gambar 3.9 dibawah :




**Gambar 3.9 Alur Stemming**

Sedangkan penjelasan dari algoritma stemming Nazief dan Adriani yang digunakan adalah sebagai berikut:

a. Data Hasil Proses Stop Words Removal

   Pada proses ini dilakukan pembacaan terhadap array dari hasil proses *stop words removal* untuk dilakukan proses *stemming.*

b. Cek Kata Dasar

   Di awal proses *stemming* dan setiap langkah yang selanjutnya dilakukan, lakukan pengecekan hasil proses stemming kata yang di-input-kan pada langkah tersebut ke kamus kata dasar. Jika kata ditemukan, berarti kata tersebut sudah berbentuk kata dasar dan proses stemming dihentikan. Jika tidak ditemukan, maka langkah selanjutnya dilakukan.

c. Hapus Inflection Suffiixes

Hilangkan *Inflection Suffixes* (“-lah”, “-kah”, “-ku”, “-mu”, atau “-nya”). Pada tahap ini digunakan fungsi preg_match untuk cek apakah kata masukan mempunyai akhiran lah,kah,ku,mu, atau nya. Jika kata masukan mempunyai akhiran tersebut selanjutnya akhiran tersebut di hilangkan dengan menggunakan fungsi preg_replace.

d. Hapus Derivation Suffixes

Hilangkan *Derivation Suffixes* (“-i”, “-an”, “-kan). Pada tahap ini digunakan fungsi preg_match untuk cek apakah kata masukan mempunyai akhiran I,an atau kan. Jika kata masukan mempunyai akhiran tersebut selanjutnya akhiran tersebut di hilangkan dengan menggunakan fungsi preg_replace.

e. Hapus Derivation Prefixes

Hilangkan *Derivation Prefixes* (“-di”, “-ke”, “-se”, “-me”, “-be”, “-pe”,”-te”). Pada tahap ini digunakan fungsi preg_match untuk cek apakah kata masukan mempunyai awalan di,ke,se,me,be,pe atau te. Jika kata masukan mempunyai awalan tersebut selanjutnya akhiran tersebut di penggal dari kata asli dengan menggunakan fungsi preg_replace. Tetapi untuk kasus awalan me,be,pe,te pemenggalan kata yang dilakukan mengikuti aturan pemenggalan yang sudah ditentukan.

f. Hasil *Stemming*

*Output* atau keluaran dari proses *stemming* adalah kata yang telah melalui tahapan – tahapan di atas. Tetapi jika semua langkah gagal, maka input kata yang diuji pada algoritma ini dianggap sebagai kata dasar. Keluaran yang dihasilkan di masukan kedalam array.

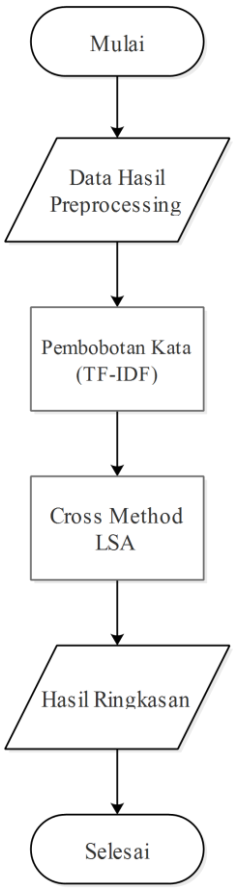
Hasil dari proses stemming dapat dilihat pada tabel 3.6 dibawah

**Tabel 3.6 Hasil Proses Stemming**

| **Kata Hasil Stemming** | | | |
|---|---|---|---|
| hukum | jahat | korupsi | korupsi |
| korupsi | pelaku | pidana | tindak |
| tindak | | | |

### 3.2.3. Analisis Metode Peringkasan

Pada analisis metode peringkasan akan dijelaskan proses peringkasan teks dengan metode pembobotan kata *Term Frequency – Inverse Document Frequency* (TF-IDF) dan metode *Latent Semantic Analysis* untuk menghasilkan ringkasan. Alur dari proses peringkasan teks pada peringkas teks otomatis dapat dilihat pada gambar 3.10.




**Gambar 3.10 Alur Peringkasan Teks**

Sedangkan penjelasan dari gambar 3.10 diatas adalah sebagai berikut:

a. Data hasil preprocessing

   Tahapan ini merupakan tahapan membaca data dari artikel yang telah melalui tahapan *preprocessing* untuk kemudian akan diolah pada tahap selanjutnya.

b. TF-IDF

   Tahapan ini merupakan tahapan pembobotan kata dari data yang didapatkan pada tahap *preprocessing*. Untuk metode pembobotan kata yang digunakan adalah metode *Term Frequency – Inverse Document Frequency* (TF-IDF).

c. Pemilihan kalimat ringkasan (LSA)

   Pada tahapan ini bobot dari kata yang telah di dapatkan akan diolah menggunakan *Cross Method Latent Semantic Analysis* untuk menentukan kalimat mana yang akan dijadikan sebagai ringkasan

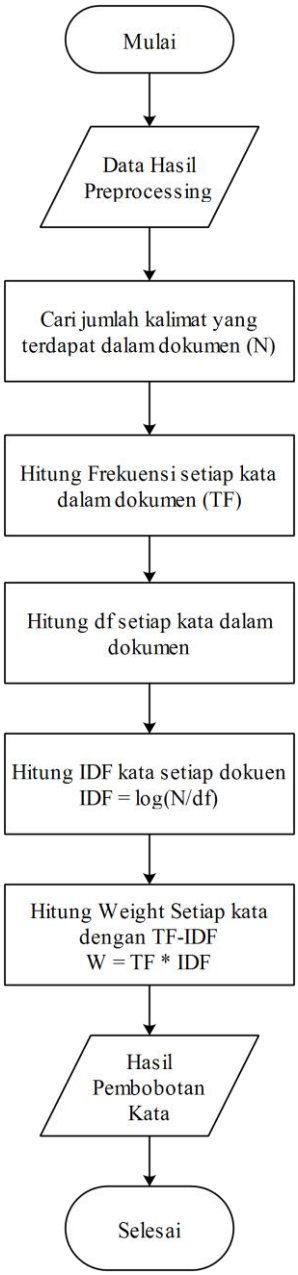
d. Hasil Ringkasan

   Merupakan tahap akhir atau *output* yang menghasilkan sebuah ringkasan dari artikel sumber

**3.2.3.1.Algoritma TF-IDF**

Pada tahapan ini akan dilakukan proses pembobotan pada setiap kata yang terdapat pada artikel yang akan diringkas. Adapun metode yang akan digunakan untuk menghasilkan bobot dari setiap kata adalah metode TF-IDF. Metode ini menghasilkan bobot dari sebuah kata berdasarkan dengan frekuensi kemunculan kata tersebut pada artikel dan berdasarkan kemunculan dari suatu kata pada dokumen lain [**17**].

Pemberian bobot pada masing-masing *term* didefinisikan melalui penghitungan frekuensi kemunculan *term* di dalam dokumen yaitu TF (*term frequency),* perhitungan frekuensi kemunculan dokumen yang mengandung sebuah *term* yaitu DF (*document frequency*) dan perhitungan jumlah dokumen yang mengandung sebuah *term* yang dicari dari kumpulan dokumen yang ada yaitu IDF *(Inverse Document Frequency).*Alur dari metode TD-IDF dapat dilihat pada gambar 3.11 dibawah:




**Gambar 3.11 Alur Pembobotan Kata**

Penjelasan dari gambar 3.11 di atas adalah sebagai berikut :

a. Data Hasil Preprocessing

   Pada proses ini dilakukan pembacaan terhadap array dari hasil tahap *preprocessing* untuk dilakukan proses pembobotan kata.

b. Cari jumlah kalimat yang terdapat dalam dokumen

Pada tahap ini akan dicari jumlah dari kalimat yang terdapat dalam dokumen. Jumlah dokumen dilambangkan dengan variabel N dimana variabel ini menandakan jumlah dokumen yang akan diringkas. Tetapi karena proses peringkasan dilakukan pada dokumen tunggal maka nilai dari variabel N berubah menjadi jumlah kata yang akan diringkas yang terdapat pada dokumen tunggal. Untuk mendapatkan nilai dari N digunakan fungsi count() untuk menghitung jumlah dari kalimat yang telah dimasukan kedalam array, array yang digunakan adalah array hasil dari proses pemecahan kalimat sehingga pemanggilan fungsi akan seperti ini : count(variabel_hasil_pisah_kalimat).

c. Hitung Frekuensi kata dalam setiap dokumen (TF)

Pada tahap ini akan dilakukan perhitungan TF (*Term Frequency*) untuk menghitung frekuensi kemunculan dari suatu kata di dalam sebuah dokumen. Pada tahap ini digunakan fungsi array_count_values() dengan parameter masukan berupa array hasil proses stemming, sehingga pemanggilan fungsi akan seperti ini : array_count_values (variabel_hasil_stemming).

d. Hitung df setiap kata

Pada tahap ini akan dilakukan proses perhitungan df dari setiap kata. Df merupakan banyak dokumen yang mengandung kata yang akan di bobotkan. Tetapi karena dokumen yang akan hitung bobotnya merupakan dokumen tunggal maka df adalah banyak kalimat yang mengandung kata yang akan dibobotkan.

e. Hitung IDF setiap kata

IDF merupakan perhitungan jumlah dokumen yang mengandung sebuah *term* yang dicari dari kumpulan dokumen yang ada. Untuk mencari nilai IDF digunakan persamaan (2.2).

f. Hitung weight atau bobot setiap kata

Untuk mendapatkan bobot atau W dari setiap kata digunakan persamaan (2.1). Rumus ini akan menghasilkan bobot dari setiap kata yang terdapat dalam dokumen tunggal yang akan di ringkas.

g. Hasil Pembobotan Kata

Keluaran yang dihasilkan dari tahap ini dimasukan kedalam array.

Hasil perhitungan nilai TF-IDF pada setiap *term* atau kata yang terdapat dalam dokumen uji yang digunakan dapat dilihat pada tabel 3.7 dan tabel 3.8.

**Tabel 3.7 Perhitungan Nilai TF-IDF**

| Kata yang dianggap penting | Frekuensi kata di setiap kalimat | | | df | D/df | IDF = log(D/df) |
|---|---|---|---|---|---|---|
| | D1 | D2 | D3 | | | |
| hukum | 0 | 0 | 1 | 1 | 3/1 = 3 | Log(3) = 0,477 |
| Jahat | 0 | 1 | 0 | 1 | 3/1 = 3 | Log(3) = 0,477 |
| Korupsi | 1 | 1 | 1 | 3 | 3/3 = 1 | Log(1) = 0 |
| Pelaku | 0 | 0 | 1 | 1 | 3/1 = 3 | Log(3) = 0,477 |
| Pidana | 1 | 0 | 0 | 1 | 3/1 = 3 | Log(3) = 0,477 |
| tindak | 1 | 1 | 0 | 2 | 3/2 = 1,5 | Log(1,5) = 0,176 |

**Tabel 3.8 Perhitungan Bobot kata (W)**

| Kata yang dianggap penting | tf | IDF = Log(D/df) | Bobot(W) setiap kata pada dokumen W = Tf * IDF |
|---|---|---|---|
| hukum | 1 | Log(3) = 0,477 | 1 * 0,477 = 0,477 |
| Jahat | 1 | Log(3) = 0,477 | 1 * 0,477 = 0,477 |
| Korupsi | 3 | Log(1) = 0 | 3 * 0 = 0 |
| Pelaku | 1 | Log(3) = 0,477 | 1 * 0,477 = 0,477 |
| Pidana | 1 | Log(3) = 0,477 | 1 * 0,477 = 0,477 |
| tindak | 2 | Log(1,5) = 0,176 | 2 * 0,176 = 0,352 |

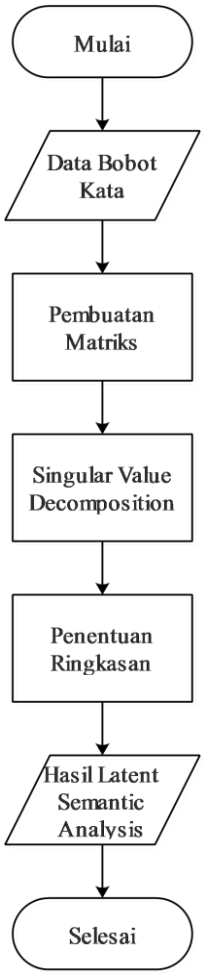
Hasil dari pembobotan kata pada tabel 3.8 diatas kemudian akan diolah kembali untuk menjadi sebuah ringkasan menggunakan metode LSA.

### 3.3.3.2.Cross Method Latent Semantic Analysis

Setelah didapatkan bobot dari setiap kata menggunakan metode TF-IDF maka bobot yang didapatkan tersebut akan diolah untuk pemilihan kalimat ringkasan. Metode yang digunakan untuk proses ekstraksi ringkasan adalah *Cross Method Latent Semantic Analysis*. Perbedaan *Cross* Method *Latent Semantic Analysis* dengan Metode *Latent Semantic Analysis* pada umumnya terdapat pada tahapan ekstraksi kalimat ringkasan.Pada tahapan ekstraksi hasil ringkasan artikel diukur berdasarkan panjang/*length* vector baris yang dihasillkan dari perkalian antara matriks V dan matriks S . Tetapi nilai matriks V seelumnya di olah terlebih dahulu dengan cara mencari setiap nilai rata-rata yang terdapat pada setiap baris di matriks V. Kemudian setelah di dapatkan nilai rata-rata tersebut maka nilai setiap kolom didalam setiap baris yang mempunyai nilai lebih kecil dari pada nilai rata-rata dari setiap baris di ubah menjadi nol (0). Dalam metode *Cross Latent Semantic Analysis* terdapat 3 tahapan utama untuk menghasilkan ringkasan, yaitu:

1. Pembuatan matriks
2. *Singular Value Decomposition*
3. Penentuan Ringkasan

Alur dari metode LSA dapat dilihat pada gambar 3.12 dibawah.




**Gambar 3.12 Alur Metode Latent Semantic Analysis**

a. Data Bobot Kata

Dalam tahap ini dilakukan pembacaan terhadap data dari hasil proses pembobotan kata. Data tersebut akan digunakan pada proses LSA.

b. Pembuatan Matriks

Dalam metode LSA untuk menghasilkan sebuah ringkasan kata yang terdapat dalam dokumen di representasikan kedalam matriks Amxn dimana n merupakan jumlah kata yang terdapat dalam dokumen dan n merupakan jumlah kalimat yang terdapat dalam dokumen. Untuk memudahkan proses pembentukan matriks A kata yang terdapat dalam dokumen ditampilkan dalam bentuk tabel 3.9 dibawah dengan nilai 1 menandakan bahwa kata tersebut terdapat dalam kalimat dan 0 menandakan kata tidak terdapat dalam kalimat.

Keterangan :

Di = Kalimat ke-i dalam dokumen

**Tabel 3.9 Lokasi Kata Dalam Kalimat**

| Kata | D1 | D2 | D3 |
|---|---|---|---|
| hukum | 0 | 0 | 1 |
| Jahat | 0 | 1 | 0 |
| Korupsi | 1 | 1 | 1 |
| Pelaku | 0 | 0 | 1 |
| Pidana | 1 | 0 | 0 |
| tindak | 1 | 1 | 0 |

Setelah diketahui lokasi kata dalam kalimat, selanjutnya masukan weight atau bobot dari setiap kata sesuai dengan bobot yang dihasilkan dari tahap TF-IDF. Hasilnya dapat dilihat pada tabel 3.10 di bawah.

**Tabel 3.10 Bobot Kata Dalam Kalimat**

| Kata | D1 | D2 | D3 |
|---|---|---|---|
| hukum | 0 | 0 | 0,477 |
| Jahat | 0 | 0,477 | 0 |
| Korupsi | 0 | 0 | 0 |
| Pelaku | 0 | 0 | 0,477 |
| Pidana | 0,477 | 0 | 0 |
| tindak | 0,352 | 0,352 | 0 |

Maka di dapatkan matriks $A_{mn}$ sebagai berikut :

$$Amn = \begin{bmatrix} 0 & 0 & 0.477 \\ 0 & 0.477 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0.477 \\ 0.477 & 0 & 0 \\ 0.352 & 0.352 & 0 \end{bmatrix}$$

*c. Singular Value Decomposition*

Pada tahap ini dilakukan proses reduksi matriks menggunakan metode *Singular Value Decomposition*. Dengan metode ini matriks A akan dipecah menjadi tiga buah matriks baru yaitu matriks U,S dan V. untuk memecah matriks A digunakan persamaan (2.3).Berikut ini adalah proses pembentukan matriks U,S dan V.

i. Membuat matriks V dan Transpose matriks V

Untuk mendapatkan nilai dari matriks V pertama cari nilai dari matriks $A^TA$ kemudian cari nilai eigenvector dari hasil perkalian tersebut karena kolom dari matriks V merupakan eigenvector dari matriks orthogonal $A^TA$. Matriks V juga digunakan sebagai koordinat dari dokumen.

$N = A^TA.$

$$N = \begin{bmatrix} 0 & 0 & 0 & 0 & 0{,}477 & 0{,}352 \\ 0 & 0{,}477 & 0 & 0 & 0 & 0{,}352 \\ 0{,}477 & 0 & 0 & 0{,}477 & 0 & 0 \end{bmatrix} \begin{bmatrix} 0 & 0 & 0{,}477 \\ 0 & 0{,}477 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0{,}477 \\ 0{,}477 & 0 & 0 \\ 0{,}352 & 0{,}352 & 0 \end{bmatrix}$$

$$N = \begin{bmatrix} 0.351 & 0.124 & 0 \\ 0.124 & 0.351 & 0 \\ 0 & 0 & 0.455 \end{bmatrix}$$

Setelah itu cari determinan dari matriks V untuk menentukan nilai eigen untuk membangun matriks S dan *eigenvector* untuk membangun matriks V. Determinan dicari dengan rumus $\det(N-\lambda I) = 0$, dimana I merupakan matriks identitas. Berikut proses pencarian determinan dari matriks N:

$\det(N - \lambda I)=0$

$$\det\left(\begin{bmatrix} 0.351 & 0.124 & 0 \\ 0.124 & 0.351 & 0 \\ 0 & 0 & 0.455 \end{bmatrix} - \lambda \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}\right)$$

$$\det\left(\begin{bmatrix} 0.351 & 0.124 & 0 \\ 0.124 & 0.351 & 0 \\ 0 & 0 & 0.455 \end{bmatrix} - \begin{bmatrix} \lambda & 0 & 0 \\ 0 & \lambda & 0 \\ 0 & 0 & \lambda \end{bmatrix}\right)$$

$$\text{Det}\begin{bmatrix} 0.351-\lambda & 0.124 & 0 \\ 0.124 & 0.351-\lambda & 0 \\ 0 & 0 & 0.455-\lambda \end{bmatrix}$$

$$\text{Det}(N-\lambda I) = 0.351-\lambda \begin{vmatrix} 0.351-\lambda & 0 \\ 0 & 0.455-\lambda \end{vmatrix} -$$

$$0.124 \begin{vmatrix} 0.124 & 0 \\ 0 & 0.455-\lambda \end{vmatrix}$$

$$= (0.351-\lambda)((0.351-\lambda)(0.455-\lambda))- \quad (0.124)((0.124)(0.455-\lambda))$$

$$= (0.351-\lambda)(0.160-0.351\lambda-0.455\lambda+\lambda^2)-(0.124)(0.056-0.124\lambda)$$

$$= 0.056 - 0.443\,\lambda + 1.157\lambda^2 - \lambda^3 - 0.007 + 0.015\,\lambda$$

$$= 0.049 - 0.428\,\lambda + 1.157\lambda^2 - \lambda^3$$

Setelah itu lakukan pemfaktoran pada hasil yang didapat. Hasil dari pemfaktoran adalah $(\lambda-0.475)(\lambda-0.455)(\lambda-0.227)$ sehingga di dapatkan nilai eigen berupa $\lambda = 0.475, \lambda = 0.455$ dan $\lambda = 0.227$.

Kemudian cari nilai dari eigenvector dari matriks N. eigenvector(x) merupakan solusi dari matriks (N-$\lambda I$) untuk setiap nilai $\lambda$ yang ada di mana $x \neq 0$. Karena pada matriks N tadi mempunyai tiga nilai eigen, maka vektor eigennya juga ada tiga.

Nilai *eigen vector* untuk $\lambda = 0.475$

$$\begin{bmatrix} 0.351-0.475 & 0.124 & 0 \\ 0.124 & 0.351-0.475 & 0 \\ 0 & 0 & 0.455-0.475 \end{bmatrix}\begin{bmatrix} x1 \\ x2 \\ x3 \end{bmatrix} = 0$$

$$\begin{bmatrix} -0.124 & 0.124 & 0 \\ 0.124 & -0.124 & 0 \\ 0 & 0 & -0.02 \end{bmatrix}\begin{bmatrix} x1 \\ x2 \\ x3 \end{bmatrix} = 0$$

Lakukan operasi eliminasi gauss pada matriks diatas seperti berikut

$$\begin{bmatrix} -0.124 & 0.124 & 0 \\ 0.124 & -0.124 & 0 \\ 0 & 0 & -0.02 \end{bmatrix} \text{R2+R1} \rightarrow \text{R2}$$

$$\begin{bmatrix} -0.124 & 0.124 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & -0.02 \end{bmatrix}$$

Dari matriks diatas dapat dihasilkan persamaan

-0.124x1 + 0.124x2 = 0

-0.021X3 = 0

Dari persamaan terseut diketahui bahwa nilai x1 = x2 dan nilai x3 = 0, maka dapat diperoleh nilai x1 = 1 ,x2 = 1 dan x3 = 0, sehingga menghasilkan *vector* [1,1,0]

Nilai *eigen vector* untuk $\lambda = 0.455$

$$\begin{bmatrix} 0.351 - 0.455 & 0.124 & 0 \\ 0.124 & 0.351 - 0.455 & 0 \\ 0 & 0 & 0.455 - 0.455 \end{bmatrix} \begin{bmatrix} x1 \\ x2 \\ x3 \end{bmatrix} = 0$$

$$\begin{bmatrix} -0.104 & 0.124 & 0 \\ 0.124 & -0.104 & 0 \\ 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} x1 \\ x2 \\ x3 \end{bmatrix} = 0$$

Lakukan operasi eliminasi gauss pada matriks diatas seperti berikut

$$\begin{bmatrix} -0.104 & 0.124 & 0 \\ 0.124 & -0.104 & 0 \\ 0 & 0 & 0 \end{bmatrix}$$ -0.104R2 – 0.124R1 → R2

$$\begin{bmatrix} -0.104 & 0.124 & 0 \\ 0 & 0.005 & 0 \\ 0 & 0 & 0 \end{bmatrix}$$ R1 – 0.124R2 → R1

$$\begin{bmatrix} -0.104 & 0 & 0 \\ 0 & 0.005 & 0 \\ 0 & 0 & 0 \end{bmatrix}$$

Dari matriks diatas dapat dihasilkan persamaan

-0.104x1 = 0

0.005x2 = 0

0*X3 = 0

Dari persamaan tersebut diketahui bahwa nilai x1=0,x2=0 dan x3=1, sehingga menghasilkan *vector* [0,0,1]

Nilai *eigen vector* untuk $\lambda = 0.227$

$$\begin{bmatrix} 0.351-0.227 & 0.124 & 0 \\ 0.124 & 0.351-0.227 & 0 \\ 0 & 0 & 0.455-0.227 \end{bmatrix} \begin{bmatrix} x1 \\ x2 \\ x3 \end{bmatrix} = 0$$

$$\begin{bmatrix} 0.124 & 0.124 & 0 \\ 0.124 & 0.124 & 0 \\ 0 & 0 & 0.228 \end{bmatrix} \begin{bmatrix} x1 \\ x2 \\ x3 \end{bmatrix} = 0$$

Lakukan operasi eliminasi gauss pada matriks diatas seperti berikut

$$\begin{bmatrix} 0.124 & 0.124 & 0 \\ 0.124 & 0.124 & 0 \\ 0 & 0 & -0.228 \end{bmatrix}$$ R2-R1 → R2

$$\begin{bmatrix} 0.124 & 0.124 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & -0.228 \end{bmatrix}$$

Dari matriks diatas dapat dihasilkan persamaan

0.124x1 + 0.124x2 = 0

0.228X3 = 0

Dari persamaan tersebut diketahui bahwa nilai x1 = -x2 dan nilai x3 = 0, maka dapat diperoleh nilai x1 = 1 ,x2 = -1 dan x3 = 0, sehingga menghasilkan vector [1,-1,0]

nilai *eiegen vector* dari hasil -diatas yang akan digunakan sebagai nilai dari kolom matriks V.

$\overrightarrow{v1}$ = [1, 1, 0] sebagai kolom 1

$\overrightarrow{v2}$ = [0 , 0 , 1] sebagai kolom 2

$\overrightarrow{v3}$ = [1, -1, 0] sebagai kolom 3

Sehingga dihasilkan matriks V :

$$V = \begin{bmatrix} 1 & 0 & 1 \\ 1 & 0 & -1 \\ 0 & 1 & 0 \end{bmatrix}$$

Setelah itu konversi matriks yang telah dibuat menjadi sebuah matriks orthogonal dengan menggunakan metode otornomalisasi gram-schmidt pada setiap kolom vektor matriks V.

Pertama normalisasi $\overrightarrow{v1}$ = [1, 1, 0]

$|\overrightarrow{v1}| = \sqrt{1^2 + 1^2 + 0^2} = \sqrt{2}$

Sehingga dihasilkan $\overrightarrow{u1} = \frac{\overrightarrow{v1}}{|\overrightarrow{v1}|} = [\frac{1}{\sqrt{2}}, \frac{1}{\sqrt{2}}, 0]$ atau

$\overrightarrow{u1}$ = [0.707107 , 0.707107 , 0]

Kemudian hitung $\overrightarrow{w2}$ dengan cara :

$\overrightarrow{w2} = \overrightarrow{v2} - \overrightarrow{u1} . \overrightarrow{v2} * \overrightarrow{u1}$

= [0 , 0 , 1] - [0.707107 , 0.707107 , 0] . [0 , 0 , 1] *
[0.707107, 0.707107 , 0]

= [0,0,1] – 0 * [0.707107, 0.707107 , 0]

$\overrightarrow{w2}$ = [0,0,1]

Kemudian normalisasikan $\overrightarrow{w2}$

$\overrightarrow{|w2|} = \sqrt{0^2 + 0^2 + 1^2} = 1$

Sehingga dihasilkan $\overrightarrow{u2} = \frac{\overrightarrow{v2}}{|\overrightarrow{w1}|} = [\frac{0}{1}, \frac{0}{1}, \frac{1}{1}]$ atau $\overrightarrow{u2}$ = [0,0,1]

Terakhir hitung $\overrightarrow{w3}$ dengan vector $\overrightarrow{u1}$ dan $\overrightarrow{u2}$ yang telah diperoleh dengan cara:

$\overrightarrow{w3} = \overrightarrow{v3} - \overrightarrow{u1} . \overrightarrow{v3} * \overrightarrow{u1} - \overrightarrow{u2} . \overrightarrow{v3} * \overrightarrow{u3}$

= [1, -1, 0] - [0.707106, 0.707107, 0] . [1, -1, 0] *

[0.707107,0.707107,0] – [0,0,1] . [1, -1, 0] *

[0,0,1]

= [1, -1, 0] - 0 * [0.707107, 0.707107, 0] – 0

= [1,-1,0] - 0

$\overrightarrow{w3}$ = [1 , -1 , 0]

Kemudian normalisasikan $\overrightarrow{w3}$

$\overrightarrow{|w3|}$ = $\sqrt{1^2 + (-q^2) + 0^2}$ = $\sqrt{2}$

Sehingga dihasilkan $\overrightarrow{u3}$ = $\frac{\overrightarrow{v3}}{\overrightarrow{|w3|}}$ = $[\frac{1}{\sqrt{2}}, \frac{-1}{\sqrt{2}}, \frac{0}{\sqrt{2}}]$ atau

$\overrightarrow{u3}$ = [0.707107 , -0.707107 , 0]

Sehingga dari perhitungan di atas diketahui nilai vektor sebagai berikut

$\overrightarrow{u1}$ = [0.707107 , 0.707107 , 0]

$\overrightarrow{u3}$ = [0, 0, 1]

$\overrightarrow{u3}$ = [0.707107 , -0.707107 , 0]

Sehingga dapat dibentuk matrik V :

$$V = \begin{bmatrix} 0.707107 & 0 & 0.707107 \\ 0.707107 & 0 & -0.707107 \\ 0 & 1 & 0 \end{bmatrix}$$

Kemudian transposes matriks V sehingga didapatkan $V^T$ :

$$V^T = \begin{bmatrix} 0.707107 & 0.707107 & 0 \\ 0 & 0 & 1 \\ 0.707107 & -0.707107 & 0 \end{bmatrix}$$

ii. Membuat matriks S

Matriks S merupakan matriks diagonal dimana nilai dari matriks S didapatkan dari nilai singular value dari matriks $A^TA$ dengan urutan *descending.* Untuk mendapatkan nilai singular value pertama cari dulu *eigenvalue* dari matriks $A^TA$. dari proses pencarian matriks V diatas diketahui bahwa matriks $A^TA$ memiliki *eigenvalue* yaitu 0.475,0.455 dan 0.227, kemudian cari akar dari *eigenvalue* tersebut untuk menghasilkan nilai untuk matriks S.

S1 = $\sqrt{0.475}$ = 0.689

S2 = $\sqrt{0.455}$ = 0.675

S3 = $\sqrt{0.227}$ = 0.476

Sehingga didapatkan matriks S sebagai berikut :

$$S = \begin{bmatrix} 0.689 & 0 & 0 \\ 0 & 0.675 & 0 \\ 0 & 0 & 0.476 \end{bmatrix}$$

iii. Membuat matriks U

Matriks U dapat dicari dengan menggunakan rumus $U = AVS^{-1}$

$$U = \begin{bmatrix} 0 & 0 & 0{,}477 \\ 0 & 0{,}477 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0{,}477 \\ 0{,}477 & 0 & 0 \\ 0{,}352 & 0{,}352 & 0 \end{bmatrix} \begin{bmatrix} 0.707107 & 0 & 0.707107 \\ 0.707107 & 0 & -0.707107 \\ 0 & 1 & 0 \end{bmatrix}$$

$$\begin{bmatrix} 1.451 & 0 & 0 \\ 0 & 1.481 & 0 \\ 0 & 0 & 2.101 \end{bmatrix}$$

$$U = \begin{bmatrix} 0 & 0.706 & 0 \\ 0.489 & 0 & -0.708 \\ 0 & 0 & 0 \\ 0 & 0.706 & 0 \\ 0.489 & 0 & 0.708 \\ 0.723 & 0 & 0 \end{bmatrix}$$

Setelah langkah diatas dilakukan maka didaptkan hasil dari SVD matriks A yaitu $A = USV^T$ :

$$\begin{bmatrix} 0 & 0 & 0{,}477 \\ 0 & 0{,}477 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0{,}477 \\ 0{,}477 & 0 & 0 \\ 0{,}352 & 0{,}352 & 0 \end{bmatrix} = \begin{bmatrix} 0.489 & -0.499 & 0 \\ 0 & 0 & 1.002 \\ 0 & 0 & 0 \\ 0.489 & -0.499 & 0 \\ 0.489 & 0.499 & 0 \\ 0.361 & 0.369 & 0.740 \end{bmatrix}$$

$$\begin{bmatrix} 0.689 & 0 & 0 \\ 0 & 0.675 & 0 \\ 0 & 0 & 0.476 \end{bmatrix}$$

$$\begin{bmatrix} 0.707107 & 0.707107 & 0 \\ 0 & 0 & 1 \\ 0.707107 & -0.707107 & 0 \end{bmatrix}$$

iv. Penentuan Ringkasan

Perbedaan dari penggunaan metode LSA umum dengan dengan metode LSA yang menggunakan metode *cross* terdapat pada tahapan ini. Jika pada metode LSA biasanya hasil ringkasan diambil dari nilai kesamaan antara query dengan koordinat dokumen (matriks $V^T$),kemudian kalimat yang mempunyai nilai tingkat kesamaan terbesar dengan query dijadikan sebagai ringkasan. Tetapi pada metode *Cross Latent Semantic Analysis* parameter yang dijadikan penentu hasil ringkasan adalah *length* atau panjang dari baris kalimat dalam dokumen yang dihasilkan dari *vector* matriks $V^T$ dikalikan dengan baris *vector* matriks S dengan persamaan (2.4).

Tetapi sebelum dilakukan proses perhitungan *length*, dicari dahulu nilai rata – rata dari setiap kata yang terdapat pada setiap baris matriks $V^T$. Kemudian kata yang mempunyai nilai lebih kecil daripada nilai rata – rata yang dihasilkan maka nilai dari kata tersebut diubah menjadi nol, jika nilai lebih besar atau sama dengan nilai matriks maka nilai dari kata tersebut tidak diubah.

Nilai rata – rata dari setiap baris adalah :

avgD1 = $\frac{0.707107+0.707107+0}{3} = 0.471$

avgD2 = $\frac{0+0+1}{3} = 0{,}333$

avgD3 = $\frac{0.707107-0.707107+0}{3} = 0$

Dari nilai rata – rata dari setiap baris matriks diatas terdapat nilai pada matriks $V^T$ yang memiliki nilai lebih kecil dari nilai rata-rata yang telah didapat, sehingga nilai yang lebih kecil tersebut di ubah menjadi 0. Hasil dari perubahan nlai pada matriks $V^T$ adalah sebagai berikut

$V^T = \begin{bmatrix} 0.707107 & 0.707107 & 0 \\ 0 & 0 & 1 \\ 0.707107 & -0.707107 & 0 \end{bmatrix}$

Kemudian setelah proses pengecekan nilai kata dengan nilai rata – rata dilakukan lanjutkan dengan proses perhitungan length dari setiap kolom matriks yang merepresentasikan kalimat dalam dokumen yang akan diringkas. Perhitungan dari length setiap kolom menggunakan rumus (2.5) adalah :

Length 1 $= \sqrt{0.707107^2 * 0.689^2 + 0 * 0.675^2 + 0.707107^2 * 0.476^2}$

$= \sqrt{0.350} = 0.592$

Length 2 $= \sqrt{0.707107^2 * 0.689^2 + 0 * 0.675^2 - 0.707107^2 * 0.476^2}$

$= \sqrt{0.124} = 0.352$

Length 3 $= \sqrt{0 * 0.689^2 + 1^2 * 0.675^2 + 0 * 0.476^2}$

$= \sqrt{0.456} = 0.675$

Jika nilai length diurutkan dengan *descending* maka urutan berupa length 3,length 1 dan length 2. Sehingga susunan kalimat yang dijadikan sebagai hasil ringkasan dari contoh diatas adalah kalimat 3, kalimat 1 dan kalimat 2.

## 3.3. Spesifikasi Kebutuhan Perangkat Lunak

Spesifikasi kebutuhan perangkat lunak yang dibangun dibagi menjadi dua kebutuhan, yaitu kebutuhan nonfungsional dan fungsional. Spesifikasi kebutuhan fungsional dan nonfungsional dapat dilihat pada tabel 3.11dan 3.12 di bawah.

**Tabel 3.11 Kebutuhan Fungsional**

| Kode | Keterangan |
|---|---|
| **SKPL-F-001** | Aplikasi mampu membaca sebuah teks berita dengan tipe file .txt |
| **SKPL-F-002** | Aplikasi mampu melakukan proses *preprocessing* pemecahan kalimat pada data masukan. |
| **SKPL-F-003** | Aplikasi mampu melakukan proses *preprocessing case folding* pada data masukan. |
| **SKPL-F-004** | Aplikasi mampu melakukan proses *preprocessing tokenizing* pada data masukan. |
| **SKPL-F-005** | Aplikasi mampu melakukan proses *preprocessing* menghapus *stop words* pada data masukan. |

| | |
|---|---|
| **SKPL-F-006** | Aplikasi mampu melakukan proses *preprocessing stemming* pada data masukan. |
| **SKPL-F-007** | Aplikasi mampu menghasilkan ringkasan dari dokumen berita tunggal berbahasa Indonesia |

**Tabel 3.12 Kebutuhan Non Fungsional**

| Kode | Keterangan |
|---|---|
| **SKPL-NF-001** | Peringkas teks otomatis dibangun berbasis web |
| **SKPL-NF-002** | Aplikasi yang dibangun dapat berjalan optimal pada *browser* google chrome dan mozila firefox |
| **SKPL-NF-003** | Aplikasi ini dibangun dengan spesifikasi *hardware* yang memenuhi standar minimum kebutuhan. |
| **SKPL-NF-004** | Data yang digunakan untuk ringkasan dokumen adalah artikel berita yang diambil dari portal berita viva.co.id. |
| **SKPL-NF-005** | Data masukan yang diambil dengan menggunakan bantuan RSS yang disediakan oleh viva.co.id |

## 3.4. Analisis Kebutuhan Non Fungsional

Analisis kebutuhan Non Fungsional menggambarkan kebutuhan dari luar yang diperlukan untuk menjalankan perangkat lunak yang dibangun. Analisis kebutuhan non fungsional yang dilakukan adalah Analisis Kebutuhan Perangkat Keras (*Hardware*), Analisis Kebutuhan Perangkat Lunak (*Software*) dan Analisis Perangkat Pikir (*Brainware*).

### 3.4.1. Analisis Kebutuhan Perangkat Keras (Hardware)

Kebutuhan Perangkat keras yang mendukung dalam pembangunan sistem peringkas teks otomatis ini adalah sebagai berikut :

1. Processor *dual core* atau lebih tinggi
2. RAM 2048MB atau lebih tinggi
3. Monitor dengan resolusi 1024 x 768 atau lebih tinggi
4. Harddisk 10GB *Free Space* atau lebih

5. Keyboard
6. Mouse

### 3.4.2. Analisis Kebutuhan Perangkat Lunak (Software)

Analisis kebutuhan perangkat lunak (*software*) pada penelitian ini yang mendukung dalam sistem peringkas teks otomatis ini adalah sebagai berikut :

1. Sistem Operasi Window atau linux.
2. Bahasa Pemrograman PHP.
3. *Web server* XAMPP.
4. *Code Editor* SublimeText 3.
5. *Web Browser* Mozilla Firefox.

### 3.4.3. Analisis Kebutuhan Perangkat Pikir (Brainware)

Kebutuhan perangkat pikir (*Brainware)* yang dibahas dalam penelitian ini dikelompokkan menjadi tiga kategori yaitu *user knowledge and experience*, *user jobs and tasks*, dan *user physical characteristic*. Penjelasan untuk masing-masing kategori sebagai berikut:

1. Pengetahuan dan Pengalaman Pengguna (*User Knowledge and Experience)*
   Pengetahuan dan pengalaman pengguna yang di targetkan dalam pembangunan perangkat lunak ini dapat dilihat pada tabel 3.12 dibawah :

**Tabel 3.13 Pengetahuan dan Pengalaman Pengguna**

| Aspek | Pengguna |
|---|---|
| *Education Level* | SMA sampai perguruan tinggi (D3,S1,S2,S3) |
| *Reading Level* | Sedang sampai tinggi |
| *Computer Literacy* | Mempunyai pengalaman dalam menggunakan komputer |
| *Task Experience* | Rendah sampai tinggi |
| *System Experience* | Rendah sampai tinggi |
| *Application Experience* | Rendah sampai tinggi |

| | |
|---|---|
| *Use Of Other System* | tidak |
| *Native Language or Culture* | Dominan bahasa Indonesia |

2. Tugas dan Kebutuhan Pengguna (*User Jobs and Tasks*)

   Tugas dan pekerjaan pengguna yang ditargetkan dalam pembangunan perangkat lunak ini dapat dilihat pada tabel 3.13 dibawah :

**Tabel 3.14 Tugas dan Pekerjaan Pengguna**

| Aspek | Pengguna |
|---|---|
| *Frequency Of Use* | Continual |
| *Primary Training* | Disediakan menu petunjuk |
| *Task or Need Importance* | Sedang sampai tinggi |
| *Task Structure* | Sedang sampai tinggi |
| *Job Categories* | Tidak |
| *Life Style* | tidak |

3. Karakteristik Pengguna (*User Physical Characteristic*)

   Karakter fisik pengguna yang ditargetkan dalam pembangunan perangkat lunak ini dapat dilihat pada tabel 3.14 dibawah:

**Tabel 3.15 Karakteristik Pengguna**

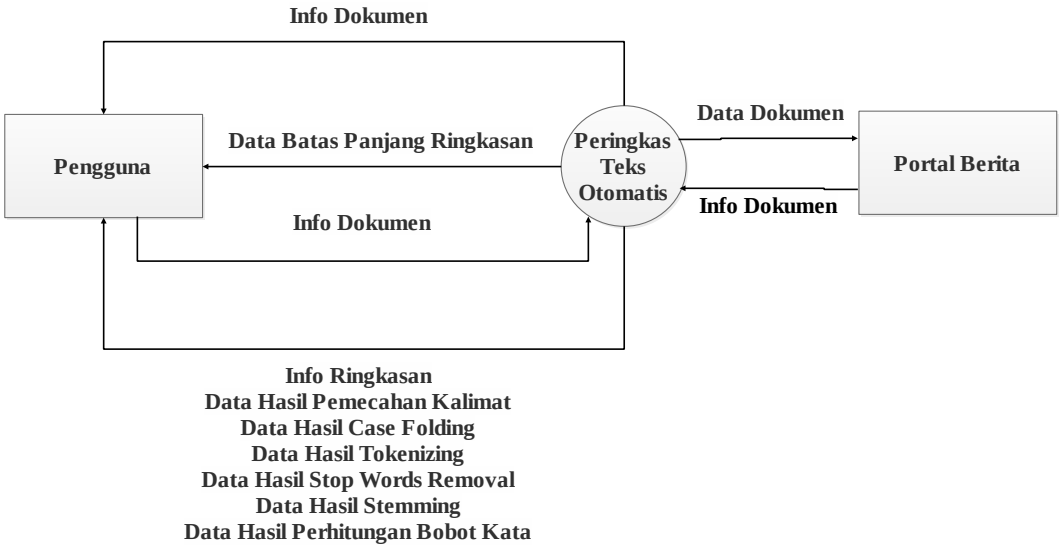
| Aspek | Pengguna |
|---|---|
| Handedness | Keduanya dalam kondisi baik |
| Gender | Laki – laki dan perempuan |
| Age | 20 tahu ke atas |
| Disabilities | Tidak cacat tubuh |

## 3.5. Analisis Kebutuhan Fungsional

Analisis kebutuhan fungsional bertujuan untuk menganalisis proses yang akan diterapkan dalam peringkas teks otomatis yang akan dibangun. Analisis kebutuhan fungsional pada aplikasi ini menggunakan diagram konteks, pembuatan *Data Flow Diagram* (DFD), dan spesifikasi proses.

### 3.5.1. Diagram Konteks

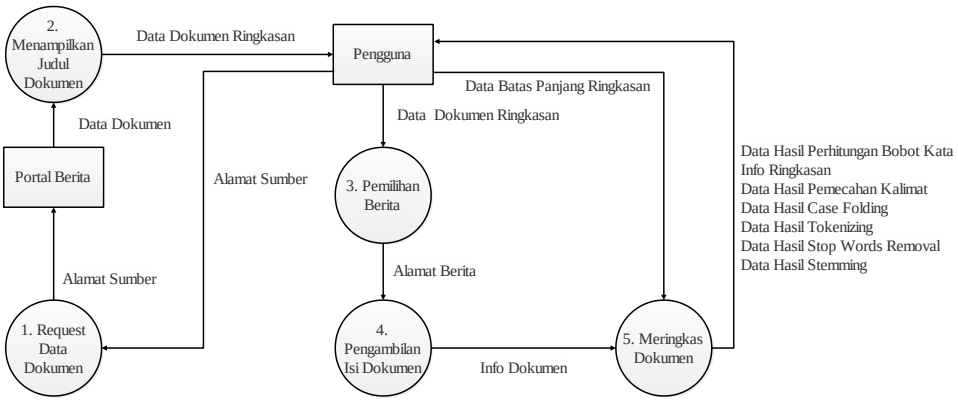
Diagram konteks adalah suatu diagram yang menggambarkan ruang lingkup sistem peringkas teks dalam berita bahasa indonesia. Peringkas teks otomatis digambarkan dengan bulatan, sedangkan lingkungan diwakili oleh entitas luar yang digambarkan dengan persegi. Gambar 3.13 dibawah menggambarkan diagram konteks yang terdapat peringkas teks otomatis.




**Gambar 3.13 Diagram Konteks**

### 3.5.2. Data Flow Diagram (DFD) LV 1

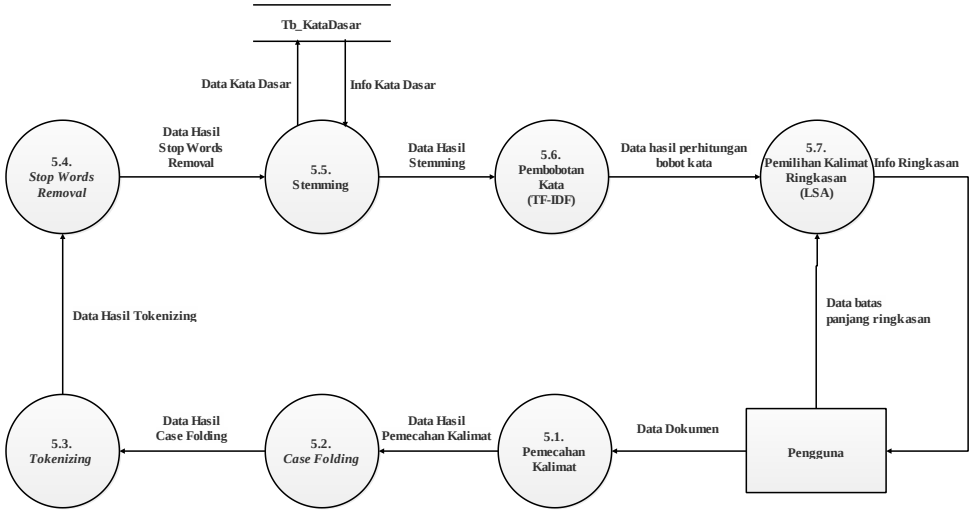
*Data flow diagram* (DFD) *level* 1 berfungsi untuk menjelaskan aliran data yang terdapat di peringkas teks otomatis. Untuk *data flow diagram* (DFD) *level* 1 pada peringkas teks otomatis digambarkan pada gambar 3.14 dibawah :




**Gambar 3.14 Data Flow Diagram Level 1**

### 3.5.3. Data Flow Diagram(DFD) LV 2

*Data flow diagram* (DFD) *Level* 2 berfungsi untuk menjelaskan aliran data yang terdapat di peringkas teks otomatis. Untuk *data flow diagram* (DFD) *level* 2 pada peringkas teks otomatis digambarkan pada gambar 3.15 dibawah :




**Gambar 3.15 Data Flow Diagram Level 2**

### 3.5.4. Deskripsi Proses

Deskripsi proses digunakan untuk menggambarkan proses model aliran yang terdapat pada DFD. Spesifikasi proses dari DFD level 1 dan DFD level 2 yang telah dibuat dapat dijelaskan pada tabel 3.16 sampai tabel 3.26 dibawah:

**Tabel 3.16 Deksripsi Proses Request Data Dokumen**

| Proses | Keterangan |
|---|---|
| **No Proses** | 1 |
| **Nama Proses** | Request Data Dokumen |
| **Sumber** | Pengguna |
| **Input** | Alamat Sumber |
| **Output** | Alamat Sumber |
| **Tujuan** | Meminta data berita yang dimiliki oleh portal berita |
| **Logika Proses** | 1. Pengguna menekan tombol pilih berita pada peringkas teks otomatis<br>2. Peringkas teks otomatis akan meminta hubungan atau koneksi dengan portal berita<br>3. Portal berita akan memberikan alamat dari sumber berita tersebut ke proses selanjutnya |

**Tabel 3.17 Deksripsi Proses Menampilkan Judul Dokumen**

| Proses | Keterangan |
|---|---|
| **No Proses** | 2 |
| **Nama Proses** | Menampilkan Judul Dokumen |
| **Sumber** | Portal Berita |
| **Input** | Data Dokumen |
| **Output** | Judul Dokumen Ringkasan<br>Alamat Dokumen Ringkasan |
| **Tujuan** | Menampilkan daftar dari berita yang dapat di ringkas |

| Proses | Keterangan |
|---|---|
| **Logika Proses** | 1. Portal berita akan memberikan alamat dari sumber berita<br>2. Peringkas teks otomatis akan mengolah alamat sumber yang diberikan dan untuk mendapatkan data dokumen<br>3. Data dokumen yang didapatkan ditampilkan dalam bentuk judul dokumen tersebut |

**Tabel 3.18 Deskripsi Porses Pemilihan Berita**

| Proses | Keterangan |
|---|---|
| **No Proses** | 3 |
| **Nama Proses** | Pemilihan Berita |
| **Sumber** | Pengguna |
| **Input** | Data Dokumen Ringkasan |
| **Output** | Judul Berita<br>Alamat Berita |
| **Tujuan** | Menampilkan daftar berita yang dapat di ringkas oleh peringkas teks otomatis kepada pengguna |

| Proses | Keterangan |
|---|---|
| **Logika Proses** | 1. Pengguna memilih atau menekan tombol memilih berita pada peringkas teks otomatis<br>2. Peringkas teks otomatis menampilkan daftar berita yang dapat di pilih oleh pengguna untuk di ringkas<br>3. Pengguna memilih berita yang akan di ringkas<br>4. Peringkas teks otomatis menerima permintaan berita yang dipilih oleh pengguna berupa judul berita<br>5. Peringkas teks otomatis memberikan alamat dari berita dari judul berita yang telah dipilih oleh pengguna ke proses selanjutnya untuk di ambil isi dari judul berita yang telah dipilih tersebut |

**Tabel 3.19 Deksripsi Proses Pengambilan Isi Dokumen**

| Proses | Keterangan |
|---|---|
| **No Proses** | 4 |
| **Nama Proses** | Pengambilan Isi Dokumen |
| **Sumber** | Pengguna |
| **Input** | Alamat Berita |
| **Output** | Info Dokumen |
| **Tujuan** | Mengambil isi dari berita yang telah di pilih oleh pengguna |

| Proses | Keterangan |
|---|---|
| **Logika Proses** | 1. Peringkas teks otomatis menerima masukan berupa alamat dari dokumen yang akan di ringkas<br>2. Dari alamat tersebut peringkas teks otomatis mengambil keseluruhan isi dari berita<br>3. Data yang telah diambil kemudian akan diolah kembali sehingga hanya isi artikel saja yang akan dibaca oleh peringkas teks otomatis (hal lain selain isi artikel seperti gambar dan tautan lain tidak dibaca) |

**Tabel 3.20 Deksripsi Proses Pemecahan Kalimat**

| Proses | Keterangan |
|---|---|
| **No Proses** | 5.1 |
| **Nama Proses** | Pemecahan kalimat |
| **Sumber** | Pengguna |
| **Input** | Data Dokumen |
| **Output** | Data Hasil Pemecahan Kalimat |
| **Tujuan** | Memecah isi dari berita yang telah didapat menjadi beberapa kalimat |
| **Logika Proses** | 1. Memilih proses peringkasan teks otomatis<br>2. Data dokumen masuk ke dalam tahap *preprocessing* pemecahan kalimat<br>3. Peringkas teks otomatis melakukan *preprocessing* berupa pemecahan kalimat pada dokumen yang dimasukan oleh pengguna. |

**Tabel 3.21 Deksripsi Proses Case Folding**

| Proses | Keterangan |
|---|---|
| **No Proses** | 5.2 |
| **Nama Proses** | Case Folding |
| **Sumber** | Peringkas teks otomatis |
| **Input** | Data Hasil Pemecahan Kalimat |
| **Output** | Data Hasil Case Folding |
| **Tujuan** | Merubah keseluruhan huruf yang terdapat pada data hasil pemecahan kalimat menjadi huruf kecil |
| **Logika Proses** | 1. Memilih proses peringkasan teks otomatis<br>2. Data pemecahan kalimat masuk ke dalam tahap *preprocessing* pemecahan kalimat<br>3. Peringkas teks otomatis melakukan *preprocessing* berupa *Case Folding* pada hasil pemecahan kalimat. |

**Tabel 3.22 Deksripsi Proses Tokenizing**

| Proses | Keterangan |
|---|---|
| **No Proses** | 5.3 |
| **Nama Proses** | Tokenizing |
| **Sumber** | Peringkas teks otomatis |
| **Input** | Data Hasil Case Folding |
| **Output** | Data Hasil Tokenizing |
| **Tujuan** | Memecah data hasil tahapan *case folding* menjadi kata |

| Proses | Keterangan |
|---|---|
| **Logika Proses** | 1. Memilih proses peringkasan teks otomatis<br>2. Data hasil *case folding* masuk ke dalam tahap *preprocessing tokenizing*<br>3. Peringkas teks otomatis melakukan *preprocessing* berupa *tokenizing* pada data hasil *case folding.* |

**Tabel 3.23 Deksripsi Proses Stop Words Removal**

| Proses | Keterangan |
|---|---|
| **No Proses** | 5.4 |
| **Nama Proses** | Stop Words Removal |
| **Sumber** | Peringkas teks otomatis |
| **Input** | Data Hasil Tokenizing |
| **Output** | Data Hasil Stop Words Removal |
| **Tujuan** | Menghilangkan kata - kata *stop words* yang terdapat pada hasil tahap *tokenizing* |
| **Logika Proses** | 1. Memilih proses peringkasan teks otomatis<br>2. Data hasil *tokenizing* masuk ke dalam tahap *preprocessing stop words removal*<br>3. Peringkas teks otomatis melakukan *preprocessing* berupa *stop words removal* pada data hasil *tokenizing.* |

**Tabel 3.24 Deksripsi Proses Stemming**

| Proses | Keterangan |
|---|---|
| **No Proses** | 5.5 |
| **Nama Proses** | stemming |
| **Sumber** | Peringkas teks otomatis |
| **Input** | Data Hasil Stop Words Removal |
| **Output** | Data Hasil Stemming |
| **Tujuan** | Merubah setiap kata dari data yag dihasilkan pada tahap *tokenizing* ke kata dasar setiap kata |
| **Logika Proses** | 1. Memilih proses peringkasan teks otomatis<br>2. Data hasil *stop words removal* masuk ke dalam tahap *preprocessing stemming*<br>3. Peringkas teks otomatis melakukan *preprocessing* berupa *stemming* pada dokumen hasil *stop words removal.* |

**Tabel 3.25 Deksripsi Proses Pembobotan kata (TF-IDF)**

| Proses | Keterangan |
|---|---|
| **No Proses** | 5.6 |
| **Nama Proses** | Pembobotan kata (TF-IDF) |
| **Sumber** | Peringkas teks otomatis |
| **Input** | Data Hasil Stemming |
| **Output** | Data Hasil perhitungan bobot kata |
| **Tujuan** | Memberikan bobot pada data atau kata yang telah melalui tahapan *preprocessing* |

| Proses | Keterangan |
|---|---|
| **Logika User** | 1. User Memilih proses peringkasan teks otomatis<br>2. Data hasil stemming masuk ke dalam tahap perhitungan bobot kata<br>3. Peringkas teks otomatis melakukan perhitungan bobot kata |

**Tabel 3.26 Deksripsi Proses Pemilihan Kalimat Ringkasan (LSA)**

| Proses | Keterangan |
|---|---|
| **No Proses** | 5.7 |
| **Nama Proses** | Pemilihan Kalimat Ringkasan (LSA) |
| **Sumber** | Peringkas teks otomatis |
| **Input** | Data Hasil Perhitungan Bobot Kata<br>Data Batas Panjang Ringkasan |
| **Output** | Info Ringkasan |
| **Tujuan** | Menghitung bobot yang dimilki oleh setiap kalimat berita yang dijadikan sebagai penentu ringkasan |
| **Logika User** | 1. User Memilih proses peringkasan teks otomatis<br>2. Data Hasil perhitungan bobot kata masuk ke dalam tahap perhitungan bobot kalimat<br>3. Peringkas teks otomatis melakukan proses pemilihan kalimat ringkasan menggunakan metode LSA<br>4. Peringkas teks otomatis menampilkan ringkasan yang didapatkan sesuai dengan batas yang telah dimasukan |

### 3.5.5. Kamus Data

Kamus data merupakan deskripsi formal dari seluruh elemen atau aliran data yang terdapat dalam data flow diagram. Kamus data dari data flow diagram sistem peringkas teks dapat dilihat pada tabel 3.27 sampai tabel 3.41 dibawah :

**Tabel 3.27 Kamus Data Data Dokumen**

| | |
|---|---|
| **Nama** | Data Dokumen |
| **Kapan digunakan** | Menampilkan Judul Dokumen (2), |
| **Deksripsi** | Data dari dokumen – dokumen yang diberikan oleh portal berita yang akan di pilih oleh pengguna untuk di ringkas |
| **Struktur data** | Teks |
| **Penjelasan per struktur** | Teks[A-Z][a-z][0-9][simbol] |

**Tabel 3.28 Kamus Data Info Dokumen**

| | |
|---|---|
| **Nama** | Info Dokumen |
| **Kapan digunakan** | Meringkas Dokumen (4) |
| **Deksripsi** | Data yang digunakan dalam proses peringkasan teks yang didapat dari portal berita sesuai dengan judul berita yang telah dipilih oleh pengguna |
| **Struktur data** | Teks |
| **Penjelasan per struktur** | Teks[A-Z][a-z][0-9][simbol] |

**Tabel 3.29 Kamus Data Batas Panjang Ringkasan**

| | |
|---|---|
| **Nama** | Batas Panjang Ringkasan |
| **Kapan digunakan** | Meringkas Dokumen (4) |
| **Deksripsi** | Data yang digunakan dalam proses meringkas dokumen ditampilkan dalam bentur persentase batas panjang ringkasan dokumen |
| **Struktur data** | angka |
| **Penjelasan per struktur** | Angka[0-9] |

**Tabel 3.30 Kamus Data Info Ringkasan**

| | |
|---|---|
| **Nama** | Info Ringkasan |
| **Kapan digunakan** | Meringkas Dokumen(4) |
| **Deksripsi** | Data yang akan di proses adalah data masukan yang akan melalui tahapan *preprocessing*,perhitungan bobot kata dan tahapan pemilihan kalimat yang akan digunakan sebagai hasil ringkasan |
| **Struktur data** | Teks |
| **Penjelasan per struktur** | Teks[A-Z][a-z][0-9][simbol] |

**Tabel 3.31 Kamus Data Data Hasil Pemecahan Kalimat**

| | |
|---|---|
| **Nama** | Data Hasil Pemecahan Kalimat |
| **Kapan digunakan** | Case Folding(4.2) |
| **Deksripsi** | Merupakan data hasil dari tahap *preprocessing* pemecahan kalimat yang kemudian akan digunakan pada tahap *preprocessing case folding* |
| **Struktur data** | Teks |
| **Penjelasan per struktur** | Teks[A-Z][a-z][0-9][simbol] |

**Tabel 3.32 Kamus Data Data Hasil Case Folding**

| | |
|---|---|
| **Nama** | Data Hasil Case Folding |
| **Kapan digunakan** | Tokenizing(4.3) |
| **Deksripsi** | Merupakan data hasil dari tahap *preprocessing case folding* yang kemudian akan digunakan pada tahap *preprocessing tokenizing* |
| **Struktur data** | Teks |
| **Penjelasan per struktur** | Teks[A-Z][a-z] |

**Tabel 3.33 Kamus Data Data Hasil Tokenizing**

| | |
|---|---|
| **Nama** | Data Hasil Tokenizing |
| **Kapan digunakan** | Stop Words Removal(4.4) |
| **Deksripsi** | Merupakan data hasil dari tahap *preprocessing tokenizing* yang kemudian akan digunakan pada tahap *preprocessing stop words removal* |
| **Struktur data** | Teks |
| **Penjelasan per struktur** | Teks[A-Z][a-z] |

**Tabel 3.34 Kamus Data Data Hasil Stop Words Removal**

| | |
|---|---|
| **Nama** | Data Hasil Stop Words Removal |
| **Kapan digunakan** | Stemming(4.5) |
| **Deksripsi** | Merupakan data hasil dari tahap *preprocessing tokenizing* yang kemudian akan digunakan pada tahap *preprocessing stemming* |
| **Struktur data** | Teks |
| **Penjelasan per struktur** | Teks[A-Z][a-z] |

**Tabel 3.35 Kamus Data Data Hasil Stemming**

| | |
|---|---|
| **Nama** | Data Hasil Stemming |
| **Kapan digunakan** | Pembobotan Kata (TF-IDF)(4.6) |
| **Deksripsi** | Merupakan data hasil dari tahap *preprocessing stemming* yang kemudian akan digunakan pada tahap pembobotan kata (TF-IDF) |
| **Struktur data** | Teks |
| **Penjelasan per struktur** | Teks[A-Z][a-z] |

**Tabel 3.36 Kamus Data Data Kata Dasar**

| | |
|---|---|
| **Nama** | Data Kata Dasar |
| **Kapan digunakan** | Stemming (4.5) |
| **Deksripsi** | Merupakan kumpulan data kata dasar bahasa Indonesia yang tersimpan pada *database* |
| **Struktur data** | Teks |
| **Penjelasan per struktur** | Teks[A-Z][a-z] |

**Tabel 3.37 Kamus Data Info Kata Dasar**

| | |
|---|---|
| **Nama** | Info Kata Dasar |
| **Kapan digunakan** | Stemming (4.5) |
| **Deksripsi** | Merupakan kumpulan data kata dasar bahasa Indonesia yang tersimpan pada *database* |
| **Struktur data** | Teks |
| **Penjelasan per struktur** | Teks[A-Z][a-z] |

**Tabel 3.38 Kamus Data Data Hasil Perhitungan Bobot Kata**

| | |
|---|---|
| **Nama** | Data Hasil Perhitungan Bobot Kata |
| **Kapan digunakan** | Pemilihan Kalimat Ringkasan (LSA) (4.7) |
| **Deksripsi** | Data yang digunakan ketika proses perhitungan bobot kata adalah data hasil dari tahapan *preprocessing* sebelumnya yang akan di gunakan pada proses pemilihan kalimat ringkasan |
| **Struktur data** | Teks |
| **Penjelasan per struktur** | Teks[A-Z][a-z] |

**Tabel 3.39 Kamus Data Alamat Sumber**

| | |
|---|---|
| **Nama** | Alamat Sumber |
| **Kapan digunakan** | Request Data Dokumen (1) |

| | |
|---|---|
| **Deksripsi** | Alamat dari sumber berita yang akan digunakan |
| **Struktur data** | Teks |
| **Penjelasan per struktur** | Teks[A-Z][a-z][0-9][simbol] |

**Tabel 3.40 Kamus Data Alamat Berita**

| | |
|---|---|
| **Nama** | Alamat Berita |
| **Kapan digunakan** | Pengambilan Isi Dokumen (4) |
| **Deksripsi** | Alamat dari artikel berita yang telah dipilih oleh pengguna untuk di ringkas |
| **Struktur data** | Teks |
| **Penjelasan per struktur** | Teks[A-Z][a-z][0-9][simbol] |

**Tabel 3.41 Kamus Data Data Dokumen Ringkasan**

| | |
|---|---|
| **Nama** | Data Dokumen Ringkasan |
| **Kapan digunakan** | Pengambilan Isi Ringkasan (3) |
| **Deksripsi** | Merupakan data yang diperoleh dari portal berita yang berisi alamat dan judul dari berita yang akan di ringkas |
| **Struktur data** | Teks |
| **Penjelasan per struktur** | Teks[A-Z][a-z][0-9][simbol] |

## 3.6. Perancangan Peringkas Teks Otomatis

Perancangan adalah suatu bagian dari metodologi pengembangan suatu perangkat lunak yang dilakukan setelah tahapan analisis untuk memberikan gambaran secara terperinci dari perangkat lunak yang akan dibangun. Dalam tahap ini digambarkan rancangan peringkas teks otomatis yang akan dibangun sebelum dilakukan pengkodean ke dalam suatu bahasa pemrograman.

### 3.6.1. Perancangan Basis Data

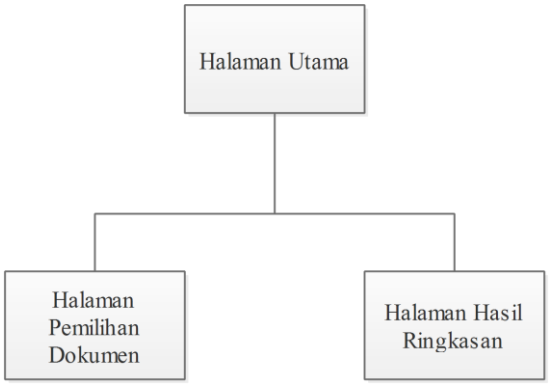
Perancangan basis data pada aplikasi yang akan dibangun meliputi struktur tabel dari database yang ditampilkan oleh tabel 3.42 dibawah :

**Tabel 3.42 Tabel Kata Dasar**

| No | Atribut | Tipe Data | Keterangan |
|---|---|---|---|
| 1 | Id_katadasar | int | *Primary key* |
| 2 | KataDasar | Varchar(70) | |
| 3 | Tipe_katadasar | Varchar(25) | |

### 3.6.2. Perancangan Arsitektur

Perancangan arsitektur yang telah dibuat meliputi beberapa perancangan diantaranya, perancangan antarmuka, perancangan pesan, jaringan semantik dan perancangan prosedural. Berikut adalah perancangan arsitektur dapat dilihat pada gambar 3.16 dibawah:



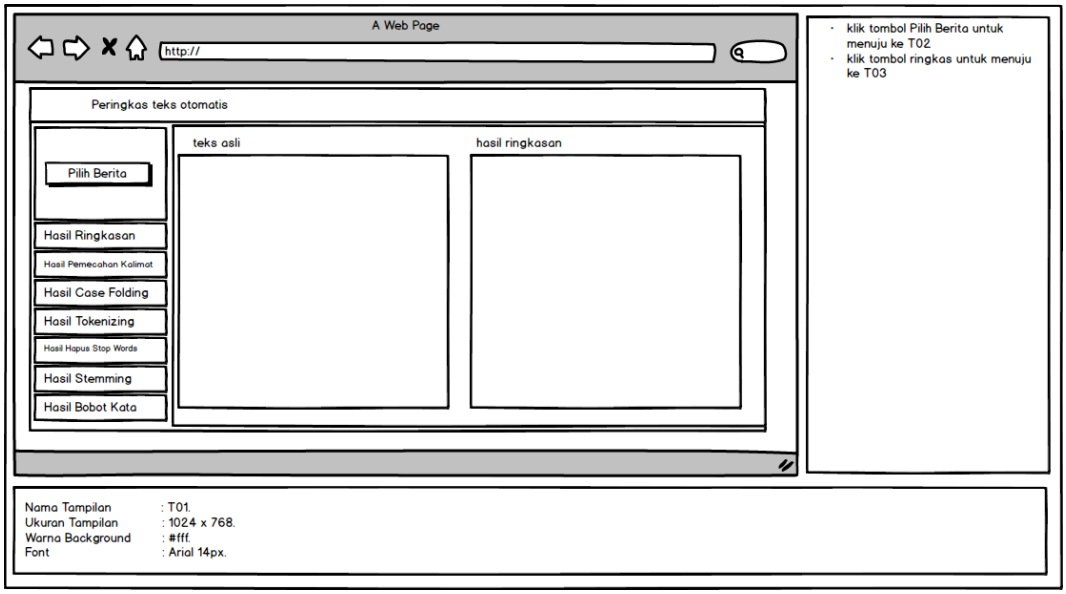

**Gambar 3.16 Perancangan Arsitektur**

### 3.6.3. Perancangan Antar Muka

Perancangan antarmuka yang telah dibuat meliputi beberapa perancangan, diantaranya perancangan antarmuka halaman utama, perancangan antarmuka pencarian dokumen dan perancangan antarmuka hasil ringkasan.

#### 3.6.3.1.Antarmuka Halaman Utama

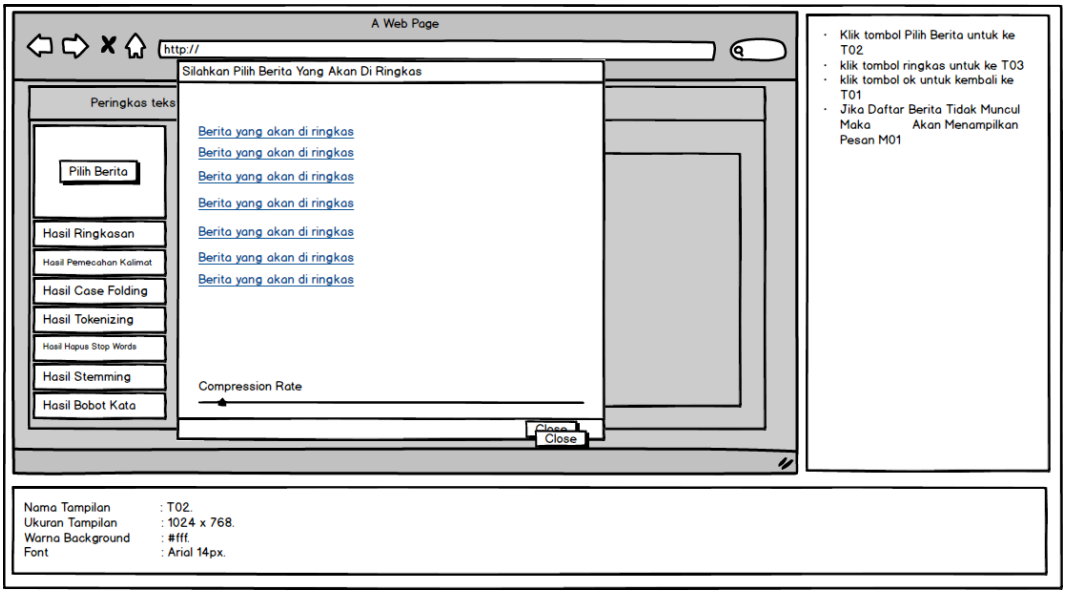
Perancangan antarmuka halaman utama yang ditampilkan peringkas teks otomatis dapat dilihat pada gambar 3.17 dibawah




**Gambar 3.17 Halaman Utama Peringkas Teks Otomatis**

### 3.6.3.2.Antarmuka Pemilihan Berita Ringkasan

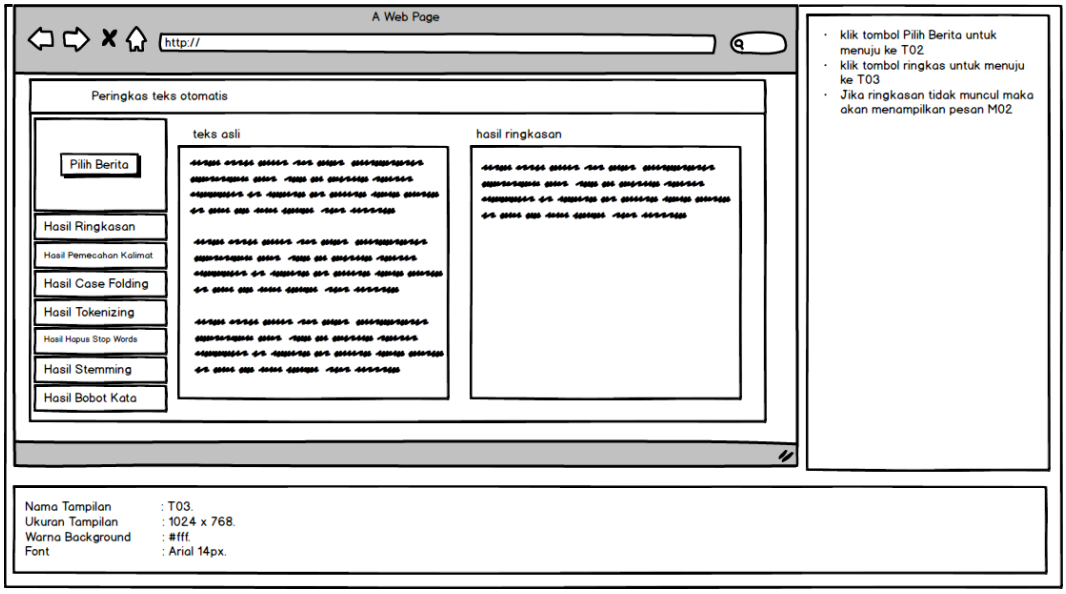
Perancangan antarmuka pemilihan berita yang ditampilkan peringkas teks otomatis dapat dilihat pada gambar 3.18 dibawah :




**Gambar 3.18 Antarmuka Pencarian Dokumen**

### 3.6.3.3.Antarmuka Hasil Ringkasan

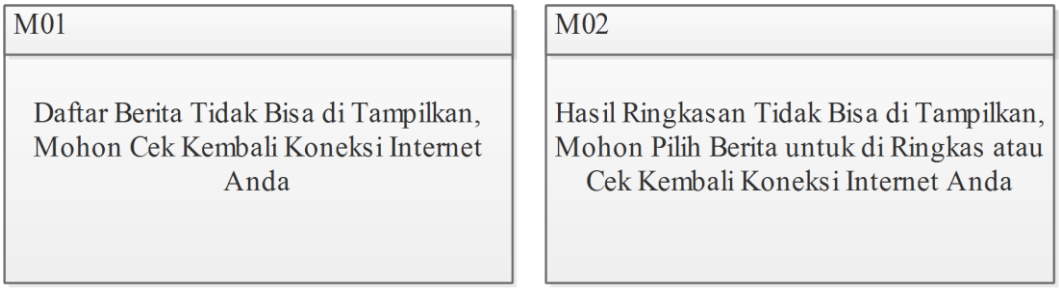
Perancangan antarmuka hasil ringkasan yang ditampilkan peringkas teks otomatis dapat dilihat pada gambar 3.19 dibawah:




**Gambar 3.19 Antarmuka Hasil Ringkasan**

### 3.6.4. Perancangan Pesan

Pada bagian ini akan di tampilkan gambar dari perancangan pesan pada peringkas teks otomatis yang dibangun. Pesan yang dimunculkan peringkas teks otomatis dapat dilihat pada gambar 3.20 dibawah :

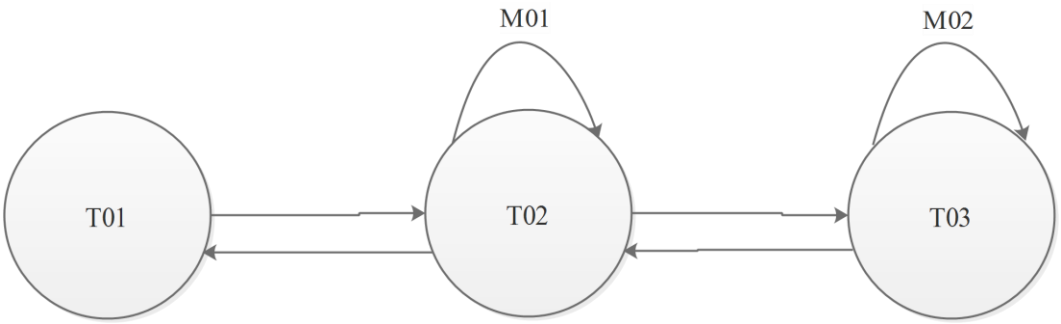



**Gambar 3.20 Tampilan Pesan Yang Di Berikan**

### 3.6.5. Jaringan Semantik

Pada bagian ini akan di tampilkan gambar dari jaringan semantik perancangan peringkas teks otomatis . Jaringan semantik akan menggambarkan hubungan antara navigasi dari menu suatu form terhadap form yang lainnya.

Jaringan semantic untuk peringkas teks otomatis bahasa Indonesia dapat dilihat pada gambar 3.21 dibawah.



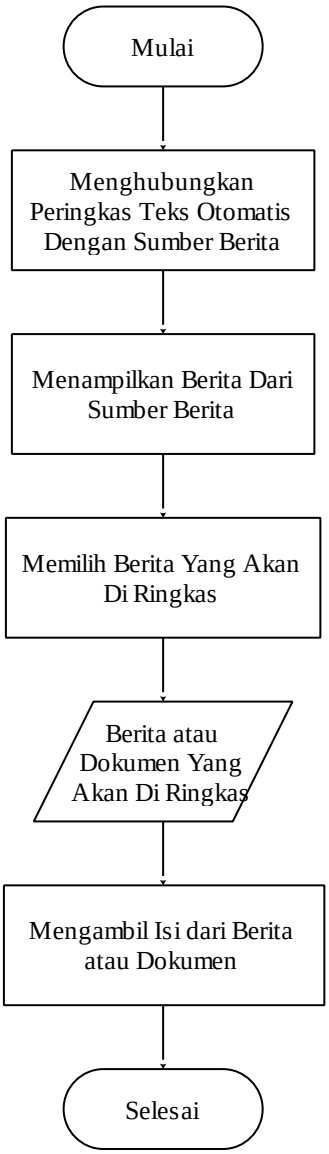

**Gambar 3.21 Jaringan Semantik**

## 3.7. Analisis Prosedural

Pada Bagian ini akan di tampilkan analisis prosedural dari peringkas teks otomatis yang dibangun dalam bentuk *flowchart*.

### 3.7.1. Analisis Tahapan Memilih Dokumen

Tahap ini merupakan tahapan pemilihan dokumen yang akan di ringkas menggunakan peringkas teks otomatis oleh user. Tahap pemilihan dokumen oleh user dapat di lihat pada gambar 3.22 dibawah




**Gambar 3.22 Flowchart Pemilihan Dokumen**

Penjelasan dari gambar 3.22 di atas adalah:

a. Menghubungkan Peringkas Teks Otomatis Dengan Sumber Berita

Pada tahap awal peringkas teks otomatis secara otomatis akan meminta data atau hubungan dari lokasi atau sumber berita yang akan di ambil ringkasan nya dengan memberikan alamat RSS yang dimiliki oleh portal berita tersebut kepada peringkas teks otomatis. Sumber berita yang dimaksud adalah portal berita yang menyediakan data berita bertema politik.

b. Menampilkan Berita Dari Sumber Berita

   Setelah alamat sumber atau RSS diberikan, maka peringkas teks otomatis akan mengambil data berita yang disediakan oleh sumber berita. Peringkas teks otomatis akan menampilkan berita – berita yang tersedia dari sumber berita atau portal berita yang telah dihubungkan dengan peringkas teks otomatis.

c. Memilih Berita yang akan di Ringkas

   Pengguna dapat memilih berita dari daftar berita yang ditampilkan oleh peringkas teks otomatis. Berita yang dipilih oleh pengguna selanjutnya akan diolah untuk mendapatkan ringkasan dari berita tersebut.

d. Berita atau Dokumen yang akan di Ringkas

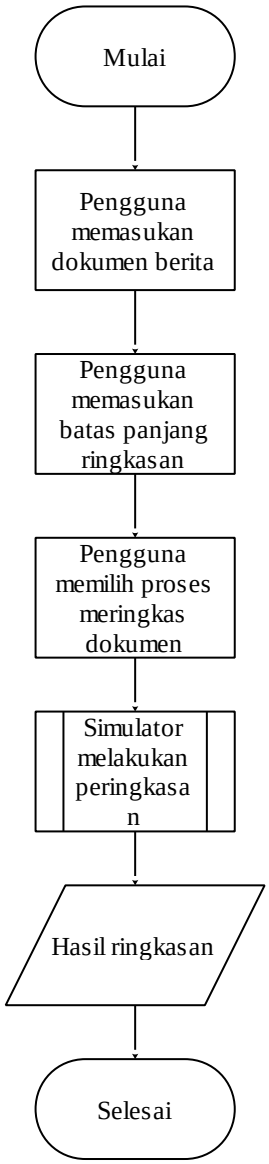
   Setelah pengguna memilih judul dari berita yang akan di ringkas maka peringkas teks otomatis akan menjadikan berita pilihan tersebut sebagai masukan yang kemudian akan di olah untuk memperoleh ringkasan dari berita tersebut.

e. Mengambil Isi dari Berita atau Dokumen

   Berita yang *user* masukan pada saat pemilihan berita berisi alamat dari sumber berita tersebut, kemudian peringkas teks akan menuju ke alamat tersebut dan mengambil isi dari berita yang telah di pilih oleh pengguna.

### 3.7.2. Analisis Tahapan Meringkas Dokumen

Pada tahap ini akan di lakukan analisis dari tahap peringkasan dokumen yang telah di pilih oleh user. Untuk leih jelasnya bisa dilihat pada gambar 3.23 dibawah:




**Gambar 3.23 Flowchart Meringkas Dokumen**

Penjelasan dari gambar 3.23 di atas adalah :

a. Pengguna memasukan dokumen berita

   Pengguna memasukan dokumen berita yang akan di ringkas menggunakan peringkas teks otomatis.

b. Pengguna memasukan batas panjang ringkasan

   Pengguna memasukan batas panjang ringkasan sebagai batas jumlah ringakasan yang akan dihasilkan oleh peringkas teks otomatis.

c. Pengguna memilih proses meringkas dokumen

   Pengguna menekan tombol ringkas yang terdapat pada peringkas teks otomatis untuk memberikan instruksi kepada peringkas teks otomatis untuk meringkas teks atau dokumen berita yang telah dimasukan oleh Pengguna.

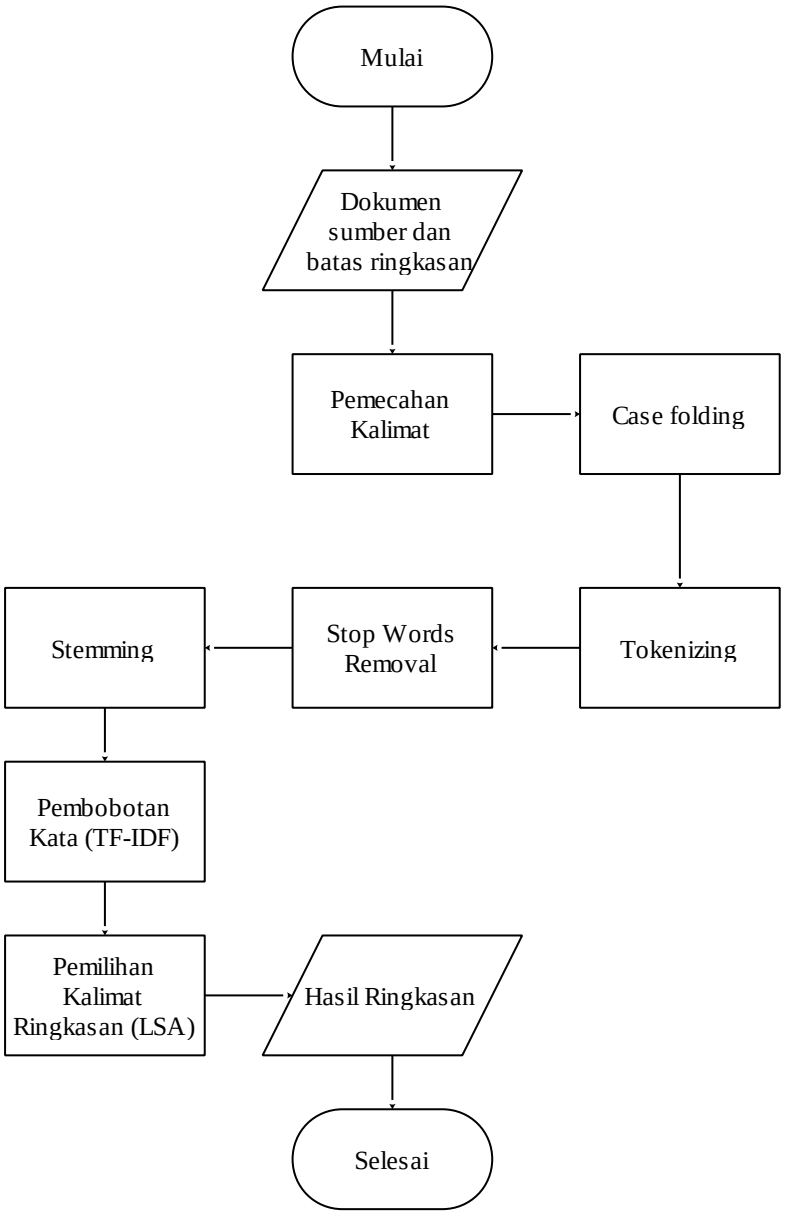
d. Peringkas teks otomatis melakukan perigkasan

   Peringkas teks otomatis melakukan proses peringkasan terhadap dokumen berita yang dimasukan oleh Pengguna.

e. Hasil Ringkasan

   Peringkas teks otomatis menampilkan hasil ringkasan dari hasil pengolahan dokumen sumber yang telah dimasukan oleh Pengguna. Jumlah ringkasan yang ditampilkan tergantung dari batas ringkasan yang telah dimasukan oleh Pengguna.

Sedangkan detil dari *predefined process* peringkas teks otomatis melakukan peringkasan yang terdapat pada *flowchart* gambar 3.23 di atas ditunjukan oleh gambar 3.24 dibawah :




**Gambar 3.24 Detil Meringkas Dokumen**

Penjelasan dari gambar 3.24 di atas adalah :

a. Dokumen sumber dan batas ringkasan

   Pada tahap ini peringkas teks otomatis membaca dokumen sumber dan batas hasil ringkasan yang telah dimasukan oleh Pengguna sebelumnya.

b. Pemecahan kalimat

   Tahapan *preprocessing* pemecahan dokumen sumber menjadi beberapa kalimat.

c. Case folding

Tahapan *preprocessing* hasil pemecahan kalimat untuk menyamakan *case* data tersebut menjadi huruf kecil.

d. Tokenizing

Tahapan *preprocessing* untuk memecah data hasil *Case Folding* menjadi per kata.

e. Stop words removal

Tahapan *preprocessing* untuk menghapus *stop words* yang terdapat pada data hasil *Tokenizing*

f. Steming

Tahapan *preprocessing* untuk mengubah kata dari hasil proses *Stop Words Removal* kedalam bentuk kata dasar nya.

g. Pembobotan kata (TF-IDF)

Tahapan pemberian bobot pada data yang telah melalui tahapan-tahapan *preprocessing* sebelumnya. Metode yang digunakan adalah metode *Term Frequency – Inverse Document Frequency* (TF-IDF).

h. Pemilihan kalimat ringkasan (LSA)

Tahapan pemilihan kalimat yang akan di jadikan ringkasan berdasarkan data yang didapat dari hasil proses pembobotan kata. Metode peringkasan yang digunakan adalah metode *Cross Latent Semantic Analysis*.

i. Hasil Ringkasan

Hasil dari proses pemilihan kalimat ringkasan berupa ringkasan dari dokumen sumber yang telah dimasukan. Jumlah ringkasan yang dihasilkan berdasarkan batas ringkasan yang telah ditentukan oleh Pengguna.