PEMERIKSA EJAAN UNTUK DOKUMEN BERBAHASA INDONESIA DENGAN ALGORITMA *LEVENSHTEIN DISTANCE*

[1]Chorin Maulina (1210651280), [2]Deni Arifianto, S.Kom., M.Kom
Jurusan Teknik Informatika Fakultas Teknik Universitas Muhammadiyah Jember
Email : qorinmaulina@gmail.com

ABSTRAK

Bahasa adalah salah satu komponen yang paling penting dalam kehidupan manusia. Dalam bentuk tulisan, bahasa menyimpan pengetahuan dari satu generasi ke generasi lain. Sedangkan dalam bentuk lisan, bahasa berperan dalam mengarahkan tingkah laku manusia sehari-hari dalam berhubungan dengan orang lain. Bahasa menjadi faktor penting pula dalam penulisan dokumen. Apabila dalam penulisan dokumen terdapat tulisan yang salah, maka artinya akan menjadi berbeda. Kesalahan pengetikan dokumen memang sering sekali terjadi.apalagi belakangan ini kesadaran masyarakat untuk menuangkan idenya ke dalam artikel, jurnal ilmiah, tugas kuliah ataupun dokumen lainnya mengalami peningkatan. Tentu saja dalam penulisan dokumen tersebut adanya kesalahan pengetikan yang disebabkan oleh beberapa faktor seperti, letak huruf pada *keyboard* yang berdekatan, dan kesalahan karena kegagalan mekanis atau slip dari tangan atau jari. Oleh karena itu, dibutuhkan sebuah program yang dapat mendeteksi kesalahan penulisan bahasa. Fasilitas pengolah kata (pemeriksa ejaan / *spell checker*) adalah sebuah fasilitas yang memungkinkan pengguna aplikasi pengolah kata memeriksa ejaan pada semua dokumen yang diketik serta memberikan usulan kata-kata untuk yang salah ejaannya. Di negara-negara maju penggunaaan fasilitas pengolah kata elektronis sangat umum,sehingga menjadi salah satu indikator pemilihan terhadap pengolah kata yang hendak dipakai. Dari beberapa algoritma dapat diimplementasikan dalam memberikan kata saran yang paling mendekati dari kata yang salah pengetikannya. Salah satunya adalah Algoritma *Levenshtein Distance* yang dapat menghitung jarak keterbedaan antara 2 (dua) String. Dalam tugas akhir ini penulis akan mengimplementasikan algoritma *Levenshtein Distance* yang dapat membantu memeriksa kesalahan penulisan pada sebuah dokumen berbahasa Indonesia.

Kata kunci : *bahasa, ejaan, kata , Levenshtein Distance, algoritma*

1. PENDAHULUAN

1.1 Latar Belakang

Bahasa adalah salah satu komponen yang paling penting dalam kehidupan manusia. Dalam bentuk tulisan, bahasa menyimpan pengetahuan dari satu generasi ke generasi lain. Sedangkan dalam bentuk lisan, bahasa berperan dalam mengarahkan tingkah laku manusia sehari-hari dalam berhubungan dengan orang lain. Bahasa menjadi faktor penting pula dalam penulisan dokumen. Apabila dalam penulisan dokumen terdapat tulisan yang salah, maka artinya akan menjadi berbeda. Pada era sekarang ini penulisan dokumen banyak menggunakan komputer, dan pada komputer tidak selalu ada program yang dapat mendeteksi kesalahan pada penulisan bahasa. Kesalahan pengetikan dokumen memang sering sekali terjadi.apalagi belakangan ini kesadaran masyarakat untuk menuangkan idenya ke dalam artikel, jurnal ilmiah, tugas kuliah ataupun dokumen lainnya mengalami peningkatan. Tentu saja dalam penulisan dokumen tersebut adanya kesalahan pengetikan yang disebabkan oleh beberapa faktor seperti, letak huruf pada keyboard yang berdekatan, dan kesalahan karena kegagalan mekanis atau slip dari tangan atau jari. Oleh karena itu, dibutuhkan sebuah program yang dapat mendeteksi kesalahan penulisan bahasa.

Fasilitas pengolah kata (pemeriksa ejaan / spell checker) adalah sebuah fasilitas yang memungkinkan pengguna aplikasi pengolah kata memeriksa ejaan pada semua dokumen yang diketik serta memberikan usulan kata-kata untuk yang salah ejaannya. Hal ini dapat meminimumkan kemungkinan salah eja atau salah ketik. Di negara-negara maju penggunaaan fasilitas pengolah kata elektronis sangat umum,sehingga menjadi salah satu indikator pemilihan terhadap pengolah kata yang hendak dipakai. Pengetikan dengan cara manual akan menghabiskan banyak waktu dan membutuhkan suatu sumber pasti sebagai acuan bahwa kata tersebut memang salah dalam proses penulisannya. Efisiensi waktu yang

dibutuhkan jika dilakukan dengan manual tentunya tidak akan optimal dan cukup membosankan sehingga kemungkinan adanya human error dapat mengakibatkan proses pengecekan kata menjadi tidak optimal. Dari beberapa algoritma dapat diimplementasikan dalam memberikan kata saran yang paling mendekati dari kata yang salah pengetikannya. Salah satunya adalah Algoritma Levenshtein Distance yang dapat menghitung jarak keterbedaan antara 2 (dua) String (Andhika,2010).

Levenshtein Distance, atau sering disebut juga sebagai edit distance, adalah suatu pengukuran (metrik) yang dihasilkan melalui perhitungan jumlah perbedaan yang terdapat pada dua string. Levenshtein distance antara dua string didefinisikan sebagai jumlah minimum perubahan yang diperlukan untuk mengganti suatu string dengan string lain, dengan operasi penambahan (insert), penghapusan (delete), atau penggantian karakter (substitute) pada suatu karakter. Yang dimaksud dengan distance adalah jumlah perubahan yang diperlukan untuk mengubah suatu bentuk string ke bentuk string yang lain. Contohnya, string "hasil" dan "hasal" memiliki distance 1 karena diperlukan satu operasi untuk mengubah string "hasal" menjadi "hasil".

Hasil Levenshtein distance yang diperoleh sebenarnya tidak dapat langsung dimanfaatkan, namun perlu diolah untuk memenuhi kebutuhan aplikasi tersebut. Banyak aplikasi yang menggunakan algoritma ini, seperti pengecek ejaan, pemandu penerjemahan, perkiraan dari pengucapan dialek, mesin pencari, pemberi revisi file dengan membandingkan perbedaan dua buah file, pendeteksi pemalsuan, pengenal percakapan (speech recognition), dan sebagainya.

Dalam tugas akhir ini penulis akan mengimplementasikan algoritma Levenshtein Distance yang dapat membantu memeriksa kesalahan penulisan pada sebuah dokumen berbahasa Indonesia.

1.2 Rumusan Masalah

Berdasarkan latar belakang yang telah dikemukan di atas dapat dirumuskan permasalahan sebagai berikut:

1. Bagaimana Mengimplementasikan algoritma *Levenshtein Distance* untuk kata dari bahasa indonesia dengan membuat aturan transformasinya ?
2. Bagaimana membuat sebuah program *Spell Checker* dengan menerapkan algoritma *Levenshtein Distance* untuk kata dari bahasa indonesia ?
3. Bagaimana mengukur kinerja algoritma *Levenshtein Distance* untuk *spelling checker* pada sebuah dokumen berbahasa indonesia ?

1.3 Batasan Masalah

Beberapa batasan masalah dalam penelitian ini adalah sebagai berikut :

1. Bahasa yang ditetapkan sebagai dasar evaluasi penulisan dalam program yang dibangun adalah bahasa indonesia.
2. Format dokumen yang di-*input*-kan adalah format *.txt.*

1.4 Tujuan Penelitian

Tujuan dari penelitian ini adalah :

1. Memberikan pemahaman mengenai proses transformasi dari kata berbahasa Indonesia yang tidak mempunyai arti.
2. Mengimplementasikan algoritma *Levenshtein Distance* pada sebuah program spelling checker.
3. Mengukur kinerja algoritma *Levenshtein Distance* untuk spelling checker pada dokumen berbahasa Indonesia.

1.5 Manfaat Penelitian

Membantu dalam mengurangi salah pengetikan pada sebuah dokumen berbahasa Indonesia yang dikarenakan salah pengetikan agar sesuai dengan kaidah penulisan Bahasa Indonesia dengan ketelitan yang cukup tinggi, serta dapat membantu dalam pembuatan naskah karya ilmiah maupun skripsi.

2. TINJAUAN PUSTAKA

2.1 Dokumen

Dokumen menurut bahasa inggris berasal dari kata *document* yang memiliki arti suatu yang tertulis atau tercetak dan segala benda yang mempunyai keterangan-keterangan dipilih untuk di kumpulkan, disusun, disediakan atau untuk disebarkan. Dibawah ini ada pendapat dari beberapa ahli mengenai pengertian dokumen, diantaranya pengertian dokumen menurut Louis Gottschalk (1986;38) Dokumen merupakan sumber tertulis bagi informasi

sejarah sebagai kebalikan daripada kesaksian lisan, artefak, peninggalan-peninggalan terlukis dan petilasan-petilasan arkeologis.

2.2 Teks

Ada beberapa pegertian yang dikemukakan oleh para ahli terkait dengan teks. Berdasarkan beberapa pengertian yang dikemukan ahli tersebut secara keseluruhan hampir sama. Luxemburg (1989) yang dikutip Tedi dalam makalahnya menyatakan bahwa teks ialah ungkapan bahasa yang menurut isi, sintaksis, dan pragmatik merupakan satu kesatuan. Teks dalam hal ini tidak hanya dipandang dari sisi tata bahasa yang sifatnya tertulis atau unsur-unsur kebahasan yang dituliskan, lebih dari itu, suatu teks juga dilihat dari segi maksud dan makna yang diujarankan. Teks memiliki kesatuan dan kepaduan antara isi yang ingin disampaikan dengan bentuk ujaran, dan situasi kondisi yang ada. Dengan kata lain, bahwa teks itu berupa ungkapan berupa bahasa yang di dalamnya terdiri dari satu kesatuan antar isi, bentuk, dan situasi kondisi penggunaannya.

2.3 Bahasa Indonesia

Secara umum bahasa adalah alat untuk beriteraksi atau berkomunikasi berupa lambang bunyi yang dihasilkan alat ucap manusia, untuk menyampaikan pikiran, gagasan, konsep atau perasaan seseorang. Bahasa terdiri atas kumpulan kata yang apabila di gabungkan akan memiliki makna tersendiri. Bahasa diciptakan sebagai alat komunikasi universal yang diharapkan dapat dimengerti oleh setiap manusia untuk melakukan suatu interaksi sosial dengan manusia lainnya.

2.4 Tokenisasi

Tokenisasi merupakan proses pemotongan kumpulan karakter menjadi sebuah kata tunggal atau *token*. Terkadang *token* dapat dikatakan juga sebagai *term* atau kata. Pemotongan kumpulan karakter biasanya berdasarkan karakter spasi, namun beberapa permasalahan yang terjadi dalam proses *tokenisasi* yaitu terdapat beberapa kata yang akan berbeda arti bila dipotong berdasarkan spasi seperti *San Fransisco* akan memiliki arti yang berbeda bila dipotong menjadi San dan Fransisco.

2.5 Algoritma *Levenshtein Distance*

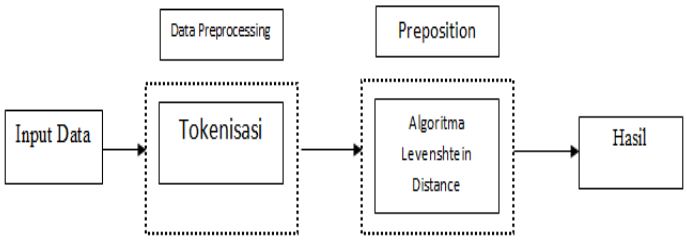
Algoritma *Levenshtein distance* dibuat oleh Vladimir Levenshtein pada tahun 1965. Algoritma *levenshtein distance* merupakan metrik yang digunakan untuk mengukur perbedaan jarak antara dua sekuens. Perhitungan *edit distance* didapatkan dari matriks yang digunakan untuk menghitung jumlah perbedaan kata antara dua kata. Perhitungan jarak antara dua kata ini ditentukan dari jumlah minimum operasi perubahan untuk membuat kata A menjadi kata B. *Levenshtein distance* antara dua kata ditentukan berdasarkan jumlah minimum perubahan/pengeditan yang dibutuhkan untuk melakukan transformasi dari satubentuk kata ke kata yang lain.

3. METODOLOGI PENELITIAN

3.1 Perancangan Sistem

Dalam tahap implementasi dilakukan pencarian dan pengumpulan informasi yang dibutuhkan selama perancangan sistem. Metode yang digunakan untuk mengumpulkan informasi dan data adalah metode studi literatur, yaitu dengan mempelajari literature yang terkait dengan penelitian, antara lain mengenai Algoritma *Levenshtein Distanse* dan penerapannya pada sistem pengecekan ejaan berbahasa Indonesia.

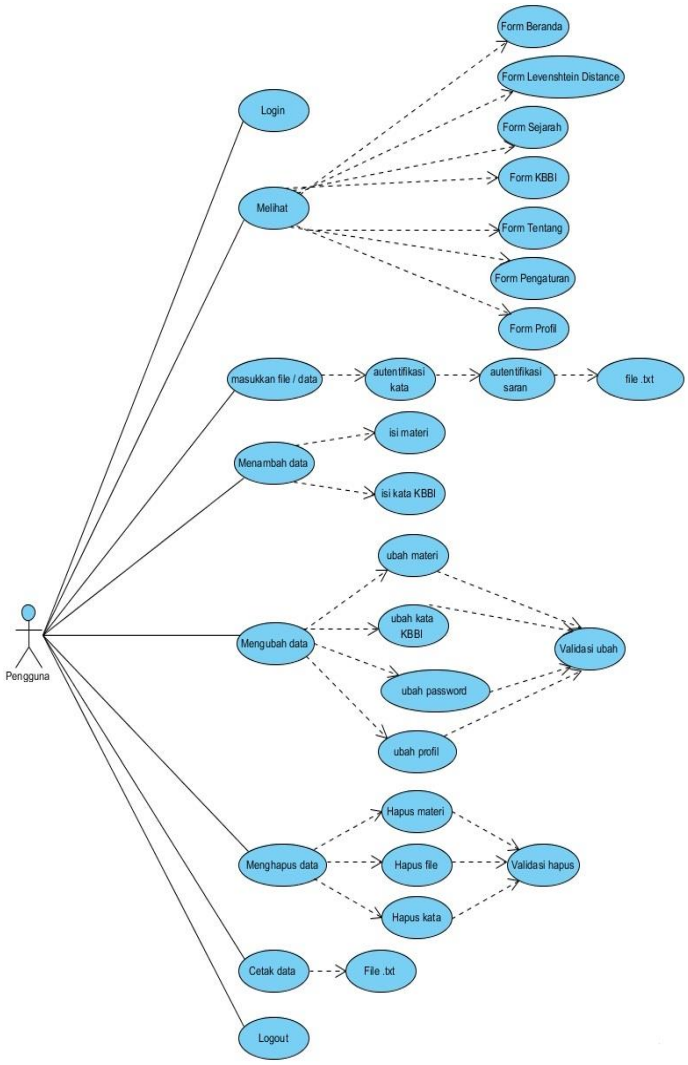
Pengecekan dilakukan per kata, kalimat masukan akan masuk ke dalam tahap *preprocessing* terlebih dahulu, sebelum diproses lebih lanjut. Proses *preprocessing*-nya meliputi penghilangan tanda baca, *tokenisasi* terhadap masing-masing kata. Kemudian setiap token akan dicocokkan ke basis data menggunakan Algoritma *Levenshtein Distance*. Setelah dilakukan perhitungan maka sistem akan menampilkan kata saran yang mendekati kesalahan penulisan. Diagram alir Sistem pengecekan ejaan kata adalah sebagai berikut.



Gambar 3.1 *Diagram Blok Sistem*

3.2 *Use Case Diagram*

Use case diagram merupakan identifikasi fungsi atau fitur yang ada pada sistem digambarkan berinteraksi dengan pengguna sebagai akses fitur yang bisa digunakan oleh pengguna tersebut. *Usecase Diagram* menggambarkan fungsi atau tugas yang dilakukan oleh pengguna, baik manusia maupun sistem. Dibawah ini merupakan *use case diagram* dari sistem pemeriksa ejaan.



Gambar 3.2 *Use case diagram pemeriksa ejaan*

3.3 *Activity Diagram*

Dari *usecase* yang di jelaskan pada sub-bab sebelumnya bahwa setiap menu yang dapat dilakukan oleh pengguna adalah *log-in*, Melihat Data, Menginput Data, Menambah Data, Mengubah Data, Menghapus Data, Cetak data dan *log-out*. Di setiap menu yang terdapat pada *usecase* diatas dapat digambarkan dengan menggunakan sebuah diagram aktifitas.

3.4 *Sequence Diagram*

Sequence diagram merupakan penggambaran interaksi dari masing-masing komponen pada satu fungsi. Interaksi tersebut dilakukan oleh user pada sistem. Di dalam sistem sendiri juga terdapat interaksi yaitu antara view, controller dan model. Pada proses perancangan ini setiap fitur akan digambarkan ke dalam *sequence diagram*. Dibawah ini merupakan *sequence* diagram yang menjelaskan tiap-tiap proses pada sistem pemeriksa ejaan.

3.5 Perancangan Basis Data

Perancangan basis data meliputi perancangan tabel yang akan digunakan untuk menyimpan data kata Berbahasa Indonesia yang dapat dilihat dari Kamus Besar Bahasa Indonesia (KBBI). Referensi kata yang tersedia cukup banyak yaitu 41,057 kata Berbahasa Indonesia.

3.6 Implementasi Algoritma *Levenshtein Distance*

Levenshtein distance digunakan untuk mengukur nilai kesamaan atau kemiripan antara dua buah kata (kata). Jarak *Levenshtein* diperoleh dengan mencari cara termudah untuk mengubah suatu kata. Secara umum, operasi mengubah yang diperbolehkan untuk keperluan ini adalah:

1. Memasukkan karakter ke dalam kata,
2. Menghapus sebuah karakter dari suatu kata,
3. Mengganti karakter kata dengan karakter lain.

Untuk menghitung jarak, digunakan matriks (n+1) x (m+1) di mana n adalah panjang kata s1 dan m adalah panjang kata s2. Dua buah kata yang akan digunakan sebagai contoh adalah RONALDINHO dengan ROLANDO. Jika dilihat sekilas, kedua kata tersebut memiliki jarak 6. Berarti untuk mengubah kata RONALDINHO menjadi ROLANDO diperlukan 6 operasi, yaitu:

1. Mensubtitusikan N dengan L (RONALDINHO -> ROLALDINHO)
2. Mensubtitusikan L dengan N (ROLALDINHO -> ROLANDINHO)
3. Mensubtitusikan I dengan O (ROLANDINHO -> ROLANDONHO)
4. Menghapus O (ROLANDONHO -> ROLANDONH)

5. Menghapus H (ROLANDONH -> ROLANDON)
6. Menghapus N (ROLANDON -> ROLANDO)

Dengan menggunakan representasi matriks dapat dilihat pada tabel di bawah.

		R	O	N	A	L	D	I	N	H	O
	0	1	2	3	4	5	6	7	8	9	10
R	1										
O	2										
L	3										
A	4										
N	5										
D	6										
O	7										

Pada Tabel 1, elemen baris 1 kolom 1 (M[1,1]) adalah jumlah operasi yang diperlukan untuk mengubah huruf dari kata ROLANDO yang diambil mulai dari karakter awal sebanyak 1 (R) ke huruf dari kata RONALDINHO yang diambil mulai dari karakter awal sebanyak 1 (R). Sementara M[3,5] adalah jumlah operasi antara ROL (huruf yang diambil mulai dari karakter awal sebanyak 3) dengan RONAL (huruf yang diambil mulai dari karakter awal sebanyak 5). Hal ini disimpulkan bahwa elemen M[p,q] adalah jumlah operasi antara huruf kata pertama yang diambil mulai dari awal sebanyak p dengan huruf kata kedua yang diambil dari awal sebanyak q. Dengan demikian, matriks pada Tabel diatas dapat diisi seperti pada Tabel dibawah ini.

		R	O	N	A	L	D	I	N	H	O
	0	1	2	3	4	5	6	7	8	9	10
R	1	0	1	2	3	4	5	6	7	8	9
O	2	1	0	1	2	3	4	5	6	7	8
L	3	2	1	1	2	3	4	5	6	7	8
A	4	3	2	2	1	2	3	4	5	6	7
N	5	4	3	3	2	2	3	4	5	6	7
D	6	5	4	4	3	3	2	3	4	5	6
O	7	6	5	5	4	4	3	3	4	5	6

Elemen terakhir (kanan bawah) adalah elemen yang nilainya menyatakan jarak kedua *kata* yang dibandingkan. Lalu untuk menghitung nilai kemiripan menggunakan rumus:

$$Sim = 1 - \left(\frac{Dis}{MaxLength}\right) \quad (1)$$

Sim = *Similarity* / nilaikemiripan
Dis = Jarak *Levenshtein*
MaxLength = Nilai string terpanjang

Jika nilai *similarity* adalah 1, maka kedua kata yang dibandingkan sama. Di lain hal, jika *similarity* 0, maka kedua kata yang dibandingkan tidak sama

3.7 Skenario Pengujian dan Analisis

Pengujian dimaksudkan untuk mengetahui apakah perangkat lunak yang dibuat sudah memenuhi krieria yang diharapkan dari tujuan perancangan perangkat lunak tersebut. Skenario Pengujian dilakukan melalui beberapa tahap, yaitu :

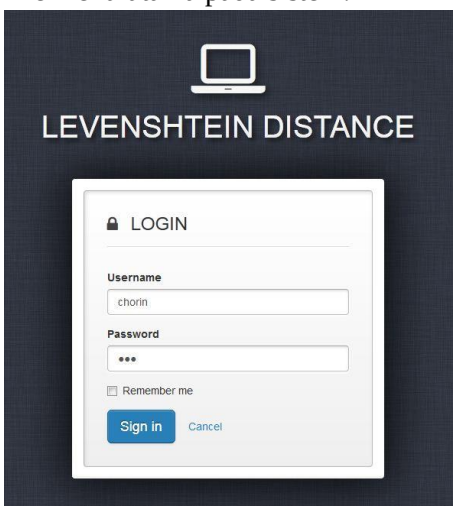
1. Masukan Inputan
2. Proses Tokenisasi
3. Pengecekan Kata
4. Pencarian Saran
5. Hasil Teks Keluaran

4. HASIL dan PEMBAHASAN

Pada bab ini akan dibahas tentang implementasi dan pengujian dari penelitian yang dilakukan. Berdasarkan dari hasil pengujian maka akan dilakukan proses analisis untuk mengetahui beberapa hal yang akan menjadi kesimpulan Tugas Akhir.

4.1 Halaman Login

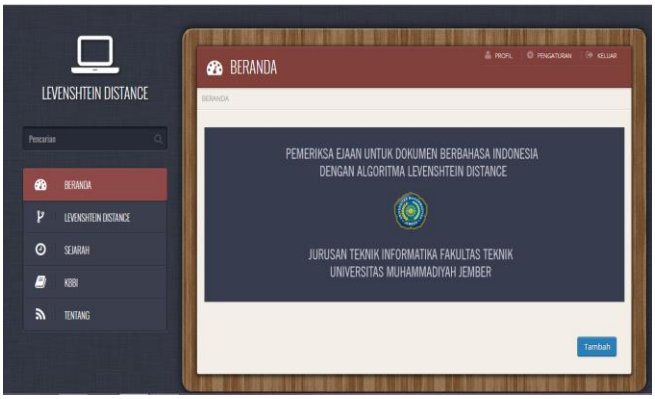
Uji coba sistem dimulai dari pengujian menu *log-in*. Pada menu ini pengguna diharuskan memasukkan user name dan password secara benar sehingga pengguna dapat masuk ke menu utama pada sistem.



Gambar 4.1 Halaman Login

4.2 Tampilan Menu Utama

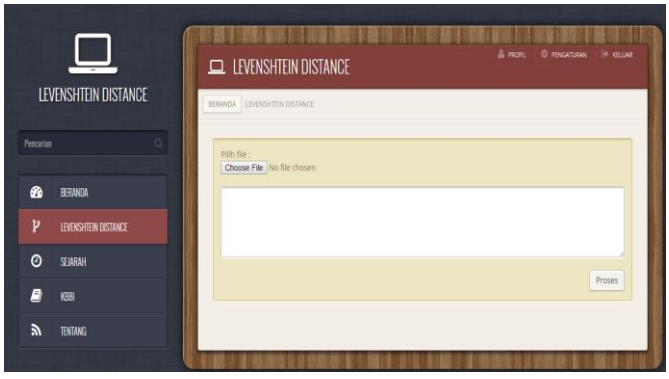
Pada tampilan Menu Utama terdapat beberapa menu, yaitu menu Beranda, *Levenshtein Distance*, Sejarah, KBBI, Tentang, Profil, dan Pengaturan. Pada menu ini terdapat Judul dari sistem yang dibuat serta terdapat tombol tambah pada bagian kanan bawah pada menu utama, tombol ini digunakan untuk menambah beberapa inputan untuk menambah isi dari menu utama tersebut.



Gambar 4.2 Halaman Utama

4.3 Levenshtein Distance

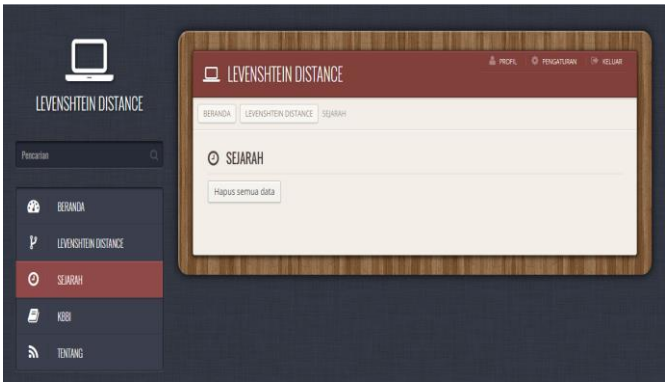
Pada halaman menu Levenshtein Distance pengguna dapat memasukkan inputan kalimat yang akan di proses untuk dicek kebenarannya, baik dengan mengetik pada kolom ataupun memilih file. Pada menu ini pengguna juga dapat menyimpan file yang sudah selesai di proses dalam bentuk format *.txt.*



Gambar 4.3 Halaman Levenshtein Distance

4.4 Sejarah

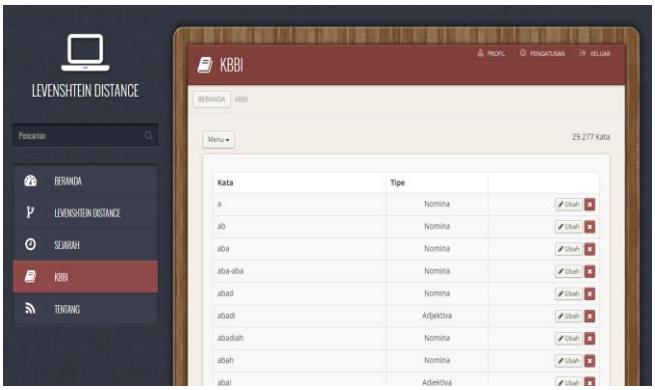
Pada menu Sejarah pengguna dapat melihat file yang sudah di proses pada menu sebelumnya yaitu menu Levenshtein Distance. Pada menu sejarah file yang sudah di proses tersimpan pada menu Sejarah, file yang terdapat pada menu Sejarah juga dapat langsung dicetak oleh user. Pada menu ini juga pengguna juga dapat menghapus file yang sudah tidak diperlukan lagi pada menu Sejarah.



Gambar 4.4 Halaman Sejarah

4.5 Halaman KBBI

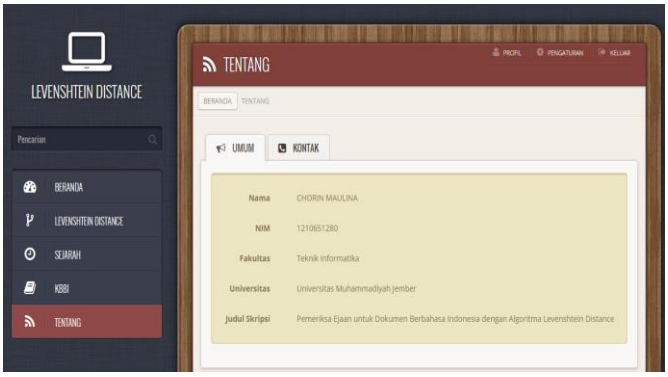
Menu Halaman KBBI adalah menu dimana terdapat kata yang terdapat pada Kamus Besar Bahasa Indonesia. Jumlah kata pada menu ini adalah 30.230 kata. Kata yang terdapat pada KBBI inilah yang menjadi kata dasar dalam penentuan kata saran pada sistem pengecekan *Levenshtein Distance*. Pada menu ini pengguna juga dapat menambah ataupun menghapus kata pada Kamus Besar Bahasa Indonesia.



Gambar 4.5 Halaman KBBI

4.6 Halaman Tentang

Pada menu Tentang terdapat dua sub menu, yang pertama tampilan Umum dan yang kedua adalah tampilan Kontak.



Gambar 4.6 Halaman Tentang

Pada tampilan Umum berisi tentang profil dari pembuat sistem, di dalam tampilan sistem berisi tentang biodata dari pembuat sistem dan judul skripsi. Sedangkan pada tampilan kontak berisi tentang alamat, no telepon dan juga alamat email.

5. KESIMPULAN dan SARAN

5.1 Kesimpulan

Dari hasil penelitian, analisis, perancangan sistem, pembuatan program sampai tahap penyelesaian program, maka penulis dapat mengambil kesimpulan sebagai berikut:

1. Berdasarkan hasil pengujian pada sistem *Spell Checker* (pemeriksa ejaan), sistem sudah dapat memeriksa akurasi sebesar 98.77%, presisi sebesar 96.43%, dan *recall* sebesar 96.43%, hal ini menunjukkan bahwa sistem memiliki tingkat nilai keakuratan (akurasi) lebih tinggi dibandingkan dengan tingkat nilai ketepatan (presisi).
2. Pengujian pencarian saran dilakukan dengan memasukkan 162 kata ke dalam sistem. Dari hasil pengujian menunjukkan bahwa 28 kata terdeteksi terdapat kesalahan ejaan yang kemudian diuji dalam pencarian saran. Dalam pencarian saran didapat 27 kata memberikan saran yang sesuai. Meskipun demikian, tidak semua kesalahan memiliki saran yang sesuai. Hal ini disebabkan karena tidak ada kata dalam kamus yang memiliki jarak maksimal 2 dengan kata yang dimasukkan pengguna. Penyebab lain dari terjadinya eror adalah karena kata yang dimasukkan terlalu kompleks.

5.2 SARAN

Berdasarkan pada pengujian yang telah dilakukan pada perangkat lunak yang dibuat, masih banyak kekurangan dan kelemahan sehingga perlu dikembangkan lagi agar kinerjanya lebih baik, oleh karena itu disarankan :

1. Untuk membuat pengoreksian dalam sistem pengoreksi kesalahan ejaan ini lebih tepat, kosa kata dalam kamus sebaiknya dilengkapi kembali.
2. Pengecekan pada sistem masih sebatas pengecekan kesalahan pengetikan, untuk kedepannya sebaiknya sistem harus bisa melakukan pengecekan pola kalimat Bahasa Indonesia
3. Selain itu, agar penggunaan sistem bisa lebih luas, pada masukan dan keluaran sistem bisa digunakan jenis-jenis ekstensi file selain *.txt.*.

Daftar Pustaka

Andhika, Fatardhi Rizky. 2010. "Penerapan *String Suggestion* dengan Algoritma *Levenshtein Distance* dan Alternatif Algoritma Lain dalam Aplikasi". Bandung : Institut Teknologi Bandung.

Gottschalk, Louis. 1986. *Understanding History; A Primer of Historical Method* (terjemahan Nugroho Notosusanto). Jakarta: UI Press.

Harimurti, Kridalaksana. 2008. *Kamus Linguistik* Jakarta: P.T. Gramedia

Renier, G.J. 1997. *History its Purpose and Method* (terjemahan Muin Umar). Yogyakarta: Pustaka Pelajar.

Singla, Nimisha. 2012. *"String Mathcing Algorithms and their Applicability in various Applications"*.