PENERAPAN ALGORITMA LEVENSHTEIN DISTANCE DALAM APLIKASI PENGOLAHAN SURAT

Isbalaikana Larasati, Fitri Marisa
[1]Fakultas Teknik, Teknik Informatika, Universitas Widyagama Malang
email: isbalaikana.laras11@gmail.com
[2]Fakultas Teknik, Teknik Informatika, Universitas Widyagama Malang
email: fitrimarisa@widyagama.ac.id

Abstract

The application of a letter processing system is expected to provide an advantage for a company. In designing a letter processing application, you can use the Levenshtein Distance method, where two strings of Levenshtein distance are the minimum number of operations needed to convert a string to another string. Levenshtein distance is also used to measure the value of similarity or similarity between two words (strings). So that it can simplify the process of finding mail archives. An example of the word used is sirat as the source word and letter as the target word, selling as the source word and submission as the target word, trading as the source and travel word as the target word. The results of the comparison of the word sirat and letters get the results of the distance comparison 1. While the word sales and submission has a distance comparison result 4. And in the word trade and travel has a distance comparison result 3.

Keywords*: Information Systems, Letter Archiving, Levenshtein Distance Algorithms*

1. PENDAHULUAN

Pada zaman sekarang teknologi berkembang sangat pesat sehingga untuk saat ini teknologi sangat berpengaruh kegunaannya pada berbagai bidang, salah satunya di bidang informasi. Pada setiap instansi atau perusahaan sebaiknya selalu mengikuti perkembangan teknologi informasi (TI) yang berhubungan dengan instansi tersebut.

Perkembangan teknologi informasi kini dapat memberi manfaat bagi perusahaan, yaitu untuk meningkatkan kegiatan usaha khususnya untuk pengolahan data. Faktanya manusia tidak bisa dengan sempurna dalam mengolah berbagai data yang ada dalam perusahaan. Sehingga peran teknologi informasi sangat penting, salah satunya dalam informasi pengolahan surat. Penerapan sistem pengolahan surat dapat memberikan keunggulan bagi suatu perusahaan.

Pada perusahaan *software house* CV. Khasanah Konsultama ini belum mempunyai sistem yang baik untuk pengolahan surat, baik itu surat masuk maupun surat keluar . Karena selama ini karyawan masih menggunakan cara manual dalam pengarsipan surat tersebut, sehingga terdapat beberapa kendala yaitu hilangnya beberapa surat. Kelemahan yang terdapat pada perusahaan *software house* dalam pengolahan surat tersebut maka dibutuhkan sistem pengolahan surat yang tepat.

Pada penelitian sebelumnya sudah banyak dari peneliti yang mengembangkan metode Algoritma Levensthein String salah satunya adalah implementasi Algoritma Levensthein String pada sistem pencarian judul skripsi atau tugas akhir. Sistem yang dibuat berbasis website. Dengan menggunakan Algoritma Levensthein String dipercaya dapat membantu mengatasi permasalahan pada kesalahan ejaan kata kunci secara optimal. (Arnawa, 2017).

Metode lain yang telah dikembangkan dalam sistem pengolahan surat adalah metode *Naïve Bayes Clasifier* (NBC) salah satu contohya aplikasi pengarsipan surat dan data kelurahan menggunakan metode *Naïve Bayes Clasifier* berbasis *E-Document*. Dalam sistem ini mampu menyiapkan dokumen yang ada dikantor pelayanan kelurahan antara lain dokumen surat dan program kegiatan (Kustianingsih & Rahmanita, 2015)

Dengan latar belakang yang telah diuraikan diatas, maka akan dibuat sebuah aplikasi pengolahan surat di perusahaan *software house* CV. Khasanah Konsultama dengan menggunakan metode Algoritma Levenshtein Distance untuk mempermudah proses pencarian dan pengarsipan surat yang akurat.

2. KAJIAN LITERATUR

2.1 SURAT

Surat merupakan suatu sarana komunikasi tertulis untuk menyampaikan suatu informasi, pernyataan, atau pesan kepada pihak lain yang mempunyai keperluan kegiatan dengan bentuk tertentu. Apabila dilihat dari sifat isinya, surat adalah jenis karangan paparan,

Sebab pengirim surat mengungkapkan maksud dan tujuannya, menjelaskan apa yang dipikirkannya dan dirasakannya melalui surat.

Berbeda halnya jika dilihat dari wujud penurunannya, surat merupakan percakapan tertulis, dari seseorang kepada seseorang, dari seseorang kepada lembaga, dari Lembaga kepada seseorang, atau dari lembaga kepada lembaga. Apabila dilihat dari fungsinya, surat merupakan sarana komunikasi tertulis. Komunikasi tersebut dapat berupa pengumuman, pemberitahuan, keterangan, dan sebagainya (luqman, 2013).

2.2 ALGORITMA LEVENSHTEIN DISTANCE

Dalam teori informasi, Levenshtein distance string adalah jumlah minimal operasi yang dibutuhkan untuk mengubah suatu string ke string yang lain, di mana operasi-operasi tersebut adalah operasi penyisipan, penghapusan, atau penyubstitusian sebuah karakter. Algoritma ini. dinamakan berdasarkan Vladimir Levenshtein yang ditemukannya pada tahun 1965. Pada makalah ini, Levenshtein distance dirujuk dengan menggunakan kata jarak saja agar lebih singkat. (Ilmy, et al., n.d.).

Algoritma penentuan jarak dua string ini dapat dibentuk melalui hubungan rekursif.

Basis:

levDis(“”, “”) = 0

levDis(s , “”) = levDis(“” , s) = |s|

Di atas terdapat dua basis. Baris pertama menyatakan dengan jelas bahwa dua string kosong tidak memiliki jarak, berarti untuk mengubah string yang satu ke yang lain tidak diperlukan operasi apapun. Baris kedua menyatakan bahwa jarak antara suatu string tidak kosong dengan string kosong adalah sebesar panjang (jumlah karakter) di dalam string yang tidak kosong.

```
Rekurens:
  levDis( s1+c1 , s2+c2 )
  =
  min ( ( levDis( s1 , s2 ) +
  ( if (c1 = c2) then
     0
   else
     1
   endif ) ),
   (levDis( s1 + c1 , s2 ) + 1) ,
   (levDis( s1 , s2 + c2 ) + 1)
  )
```

Di atas tertulis bahwa kedua string yang dibandingkan tidak kosong. Keduanya memiliki karakter terakhir c1 dan c2. Di sini dijelaskan bahwa terdapat tiga alternatif untuk menentukan jarak/ kedua string. Pertama, Jika c1 dan c2 sama, maka c1 dan c2 tidak perlu dipertukarkan berarti jaraknya levDis (s1,s2) + 0, jika berbeda berarti hanya tinggal mengubah c1 menjadi c2 saja, berarti jaraknya levDis (s1,s2) + 1. Dalam hal ini operasi yang dilakukan adalah operasi substitusi. Kedua, dapat juga dilakukan operasi penghapusan c1 dari s1 dan mengubahnya menjadi s2 + c2 sehingga jaraknya menjadi levDis (s1,s2+c2) + 1. Ketiga, mirip seperti yang kedua dapat dilakukan juga operasi penyisipan c2 pada s1 + c1 yang telah diubah menjadi s2 sehingga jaraknya adalah levDis (s1+c1,c2) + 1. Ketiga alternatif di atas adalah semua kemungkinan perubahan yang ada, dan dari antara ketiganya dicari yang mana yang paling sedikit jaraknya dengan fungsi min yang mencari nilai paling minimum di antara tiga nilai.

Dalam implementasi algoritma rekursif ini, terdapat tiga kali pemanggilan rekursif untuk setiap rekurens. Hal ini membuat algoritma ini menjadi sangat lambat dan hanya baik digunakan pada string yang terdiri dari karakter-karakter yang sedikit saja. Oleh karena itu, pemeriksaan lebih lanjut menunjukkan bahwa jarak s1 dan s2 bergantung pada jarak s1’ dan s2’ saja di mana s1’ lebih pendek dari s1, dan s2’ lebih pendek dari s2. Sementara jarak s1’ dan s2’ bergantung pada jarak s1” dan s2” di mana

keduanya lebih pendek dari yang sebelumnya. Hal ini menunjukkan bahwa teknik pemrograman dinamis dapat digunakan.

Untuk menghitung jaraknya tanpa menggunakan proses rekursif, digunakan matriks (n + 1) × (m + 1) di mana n adalah panjang string s1 dan m adalah panjang string s2. Berikut dua string yang akan digunakan sebagai contoh:

RONALDINHO ROLANDO

Jika kita melihat sekilas, kedua string tersebut memiliki jarak 6. Berarti untuk mengubah string RONALDINHO menjadi string ROLANDO diperlukan 6 operasi, yaitu :

1. Mensubstitusikan N dengan L
   RONALDINHO = ROLALDINHO
2. Mensubstitusikan L dengan N
   ROLALDINHO = ROLANDINHO
3. Mensubstitusikan 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 ditunjukkan tabel berikut:

Tabel 1 Representasi Matriks

		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 ini, elemen baris 1 kolom 1 (M[1,1]) adalah jumlah operasi yang diperlukan untuk mengubah substring dari kata ROLANDO yang diambil mulai dari karakter awal sebanyak 1 (R) ke substring kata RONALDINHO yang diambil mulai dari karakter awal sebanyak 1 (R). Sementara elemen M [3,5] adalah jumlah operasi antara ROL (substring yang diambil mulai dari karakter awal sebanyak 3) dengan RONAL (substring yang diambil mulai dari karakter awal sebanyak 5).Berarti elemen M [p,q] adalah jumlah dari operasi antara substring kata pertama yang diambil mulai dari awal sebanyak p dengan substring kata kedua yang diambil dari awal sebanyak q. Sehingga dengan peraturan ini matriks dapat diisi, menghasilkan:

Tabel 2 Matriks Hasil Perhitungan

		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 (yang paling kanan bawah) adalah elemen yang nilainya menyatakan jarak kedua string yang dibandingkan. Sehingga algoritma untuk mengisi matriks sesuai dengan yang di atas adalah:

```
function levDis (s1 : string, s2 : string) : integer
kamus
  i, j, cost : integer
  m : array [0 .. s1.length, 0 .. s2.length] of integer
algoritma
  for i ← 0 to s1.length do
    for j ← 0 to s2.length do
      if i = 0 then
        m[i,j] ← j {perbandingan dengan kosong}
      else if j = 0 then
        m[i,j] ← i {perbandingan dengan kosong}
      else {implementeasi pemrograman dinamis}
        if s1[i] = s2[j] then
          cost ← 0
        else
          cost ← 1

        m[i,j] = minimum (
                    m[i-1, j-1] + cost, {substitusi}
                    m[i-1,j] + 1, {penghapusan}
                    m[i ,j-1] + 1, {penambahan}
                  )

  return  m[s1.length, s2.length]
```

Gambar 1 Algoritma Untuk Matriks

Algoritma ini memiliki kompleksitas waktu O(mn), dengan m dan n adalah panjang masing-masing string yang diperbandingkan. Dengan kata lain, jika kedua string memiliki panjang yang sama kompleksitas waktunya adalah O(n2). Kompleksitas ruang untuk algoritma ini adalah juga O(mn) atau O(n2) jika kedua string memiliki panjang sama. Namun, jika kita melihat bagian implementasi rekurens pada algoritma ini, sebenarnya suatu baris matriks hanya membutuhkan data dari baris sebelumnya saja, berarti hanya dibutuhkan 2 × n ruang memori untuk menyimpannya. Oleh karena itu, dapat dibuat lagi perbaikan algoritma sehingga kompleksitas ruangnya menjadi O(n). Algoritma perbaikannya adalah:

```
function levDis (s1 : string, s2 : string) : integer
kamus
  i, cost : integer
  mBefore : array [0 .. s1.length] of integer
  mCurrent : array [0 .. s1.length] of integer
algoritma
  for i ← 0 to s1.length do
    {inisialisasi baris awal dengan nilai jarak
    perbandingan dengan kosong}
    mBefore[i] ← i

  for i ← 0 to s1.length do
    for j ← 1 to s2.length do
      if i = 0 then  {perbandingan dgn kosong}
        mCurrent[i] ← j
      else
        if s1[i] = s2[j] then
          cost ← 0
        else
          cost ← 1

        mCurrent[i] = minimum (
              mBefore[i-1] + cost, {substitusi}
              mCurrent[i-1] + 1, {penghapusan}
              mBefore[i] + 1, {penambahan}
          )

    mBefore ← mCurrent

  return  mCurrent[s1.length]
```

Gambar 2 Kompleksitas Algoritma Untuk Matriks.

Perlu diketahui bahwa pada algoritma ini, tipe string dianggap memiliki properti length yang menyatakan jumlah karakter di dalam string tersebut dan dianggap bahwa string terdiri dari array of characters yang berindeks mulai dari 1 sampai jumlah karakter. (Ilmy, et al., n.d.)

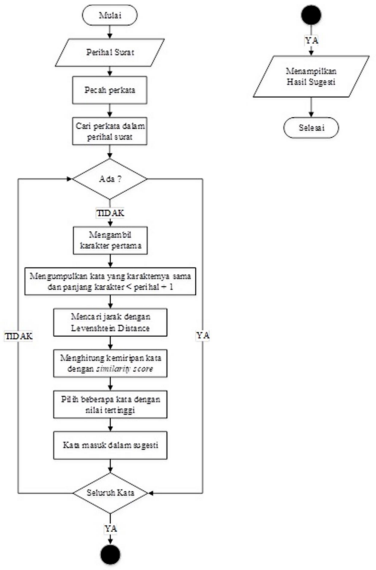
3. METODE PENELITIAN

3.1 ANALISA KEBUTUHAN

Analisis kebutuhan bertujuan untuk menganalisis dan mendapatkan semua kebutuhan yang diperlukan dalam perancangan rekomendasi tempat kerja praktek menggunakan Algoritma *Levenshtein Distance*. Analisa kebutuhan disesuaikan dengan lokasi dan variabel penelitian, menentukan kebutuhan data yang akan digunakan, dan mempersiapkan alat dan bahan penelitian. Metode analisis yang digunakan adalah *Procedural Analysis* dengan menggunakan bahasa pemodelan prosedural. Pemrograman berbasis prosedur merupakan teknik pemrograman yang dikembangkan berdasarkan algoritma untuk memecahkan suatu masalah. Algoritma merupakan cara-cara yang ditempuh dalam memanipulasi data sehingga masalah yang dihadapi bisa dipecahkan. Dalam hal ini, menggunakan metode Algoritma *Levenshtein Distance* dalam pengimplementasiannya. Secara keseluruhan, kebutuhan yang digunakan dalam pembuatan perancangan rekomendasi tempat kerja praktek ini meliputi:

1. Kebutuhan Hardware, meliputi:
   a. Laptop
2. Kebutuhan Software, meliputi:
   a. Windows 7 sebagai sistem operasi
   b. Microsoft Visio 2013 sebagai aplikasi untuk pembuatan perancangan sistem.
   c. Balsamiq Mockups 3 sebagai aplikasi untuk perancangan user interface.
3. Data yang dibutuhkan meliputi:
   a. Data surat masuk dan surat keluar.

3.2 FLOWCHART PROSES ALGORITMA



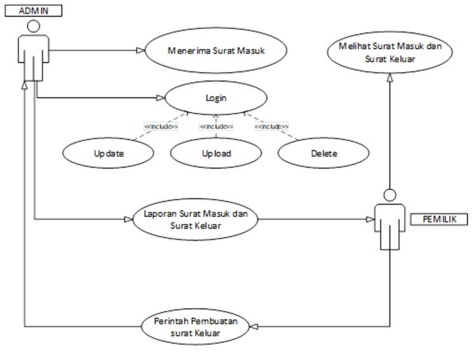
Gambar 3 Flowchart Proses Pencarian.

Alur Penelitian ini berisi tentang bagaimana prosedur-prosedur yang berjalan pada perancangan aplikasi pengolahan surat ini. Dalam perancangan *flowchart* ini peneliti akan menjelaskan alur sistem yang berjalan yaitu sistem yang dirancang untuk analisis yang menggunakan Algoritma *Levenshtein Distance*.

3.3 RANCANGAN PROSES

Rancangan proses (design process) menjelaskan kapan dan bagaimana sesuatu harus dilakukan untuk mendukung kebutuhan pemakai aplikasi pengolahan surat. Untuk menggambarkan rancangan proses dalam perancangan aplikasi pengolahan surat yaitu:

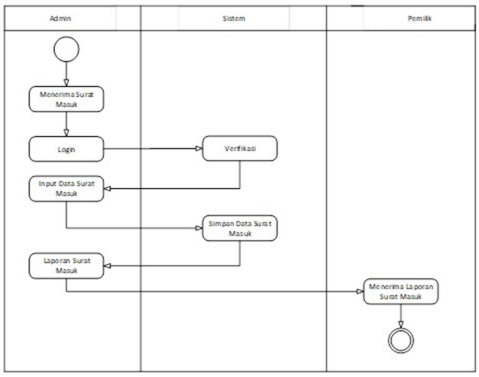
3.3.1 USE CASE DIAGRAM



Gambar 4 *Use Case* Diagram.

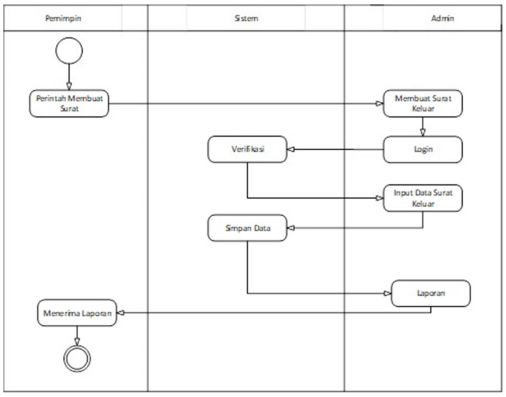
Pembuatan use case diagram menggambarkan alur proses pada suatu sistem serta yang akan menentukan sebuah kebutuhan yang akan di pakai dalam aplikasi pengolahan surat di CV. Khasanah Konsultama Malang.

3.3.2 ACTIVITY DIAGRAM SURAT MASUK



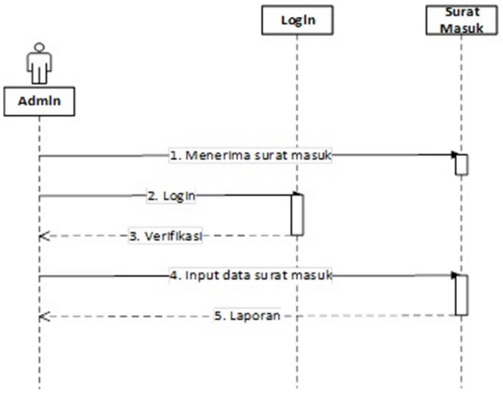
Gambar 5 *Activity* Diagram Surat Masuk.

3.3.3 ACTIVITY DIAGRAM SURAT KELUAR



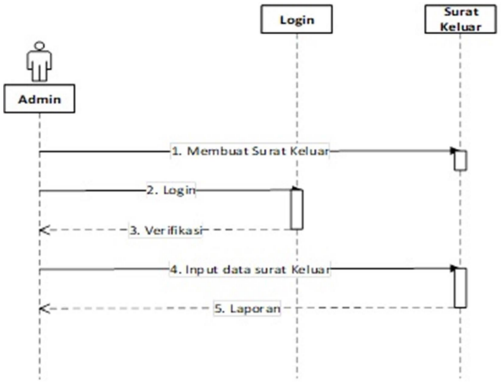
Gambar 6 *Activity* Diagram Surat Keluar.

3.3.4 SEQUANCE DIAGRAM SURAT MASUK



Gambar 7 *Sequance* Diagram Surat Masuk.

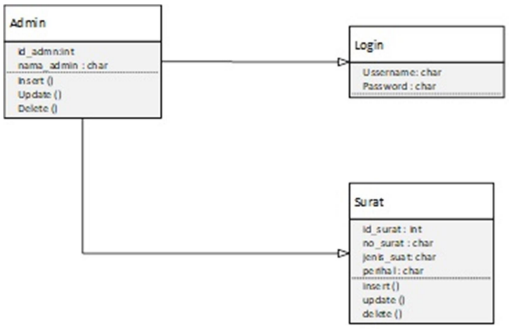
3.3.5 SEQUANCE DIAGRAM SURAT KELUAR



Gambar 8 *Sequance* Diagram Surat Keluar.

3.3.6 CLASS DIAGRAM

Class Diagram menggambarkan strukur sistem yang akan dibuat untuk membangun Aplikasi Pengolahan Surat di CV. Khasanah Konsultama Malang.



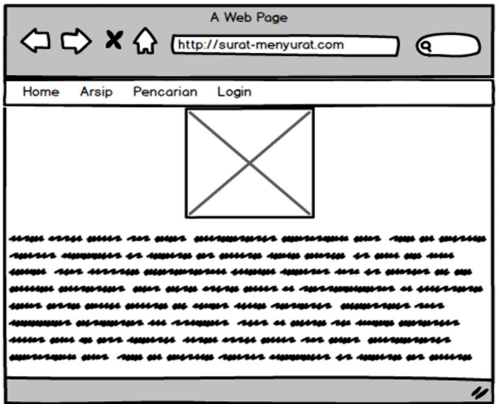
Gambar 9 *Class* Diagram.

3.3.7 RANCANGAN ANTAR MUKA

Rancangan antar muka memberikan gambaran kepada user tentang struktur menu dan perancangan tampilan pada tampilan user. Rancangan antar muka yang akan dibuat adalah :

3.3.7.1 MENU HOME

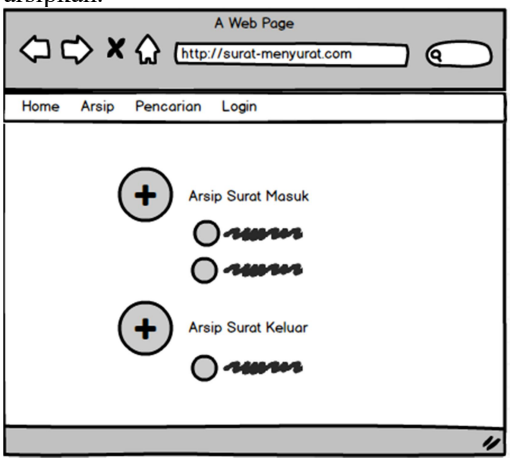
Menu Home berisi beberapa informasi mengenai CV. Khasanah Konsultama Malang dan terdapat logo dari CV. Khasanah Konsultama Malang.



Gambar 10 Menu Home.

3.3.7.2 MENU ARSIP

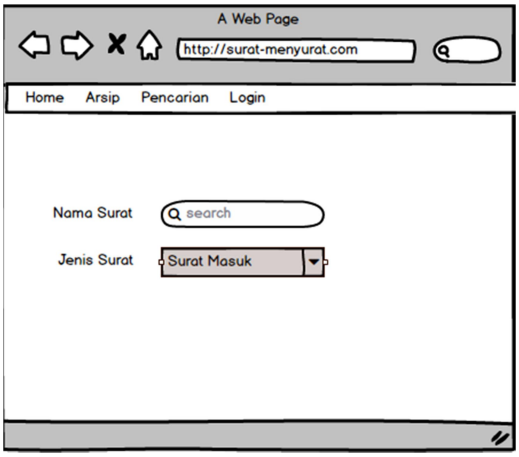
Menu Arsip berisikan jenis surat, yaitu surat masuk dan surat keluar. Didalam jenis surat tersebut berisi surat-surat yang telah di arsipkan.



Gambar 11 Menu Arsip.

3.3.7.3 MENU PENCARIAN

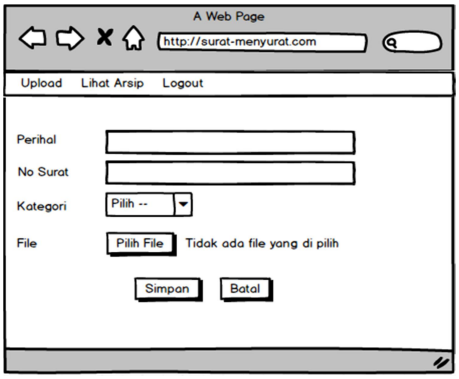
Menu Pencarian merupakan menu untuk pengelompokan atau root lokasi file berdasarkan kelompok yang telah ditentukan. Ada 2 (dua) root, yaitu : 1) Surat Masuk, 2) Surat Keluar.



Gambar 12 Menu Pencarian.

3.3.7.4 MENU UPLOAD

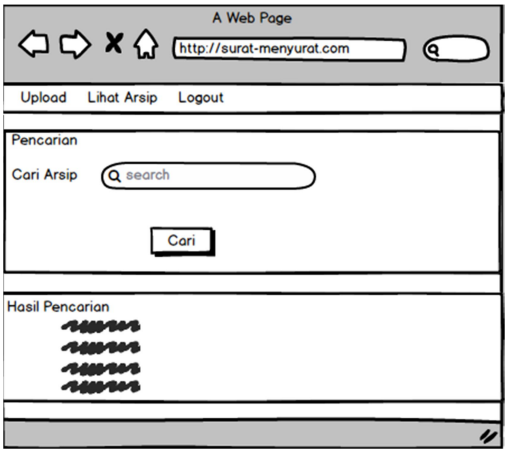
Menu Upload ini digunakan untuk admin menginputkan data-data surat masuk dan surat keluar.



Gambar 13 Menu Upload.

3.3.7.5 MENU LIHAT ARSIP

Menu Lihat Arsip adalah salah satu menu di dalam login. Menu lihat arsip ini bisa digunakan admin untuk mengecek data-data surat masuk dan surat keluar.



Gambar 14 Menu Lihat Arsip.

4. HASIL DAN PEMBAHASAN

4.1 INISIALISASI

Langkah pertama yang dilakukan dalam perhitungan ini adalah inisialisasi. Inisialisasi ini merupakan sebagai dasar perhitungan, dimana kita harus menentukan sumber kata dan target kata untuk menghitung distance atau jarak dari kedua kata tersebut. Berikut adalah inisialisasi sumber kata dan target kata :

a. S (Source) = Sirat, T (Target) = Surat

Tabel 3 Inisialisasi Sirat (S), Surat (T)

		S	I	R	A	T
S						
U						
R						
A						
T						

b. S (Source) = Penjualan , T (Target) = Pengajuan

Tabel 4 Inisialisasi Penjualan (S), Pengajuan (T)

		P	E	N	J	U	A	L	A	N
P										
E										
N										
G										
A										
J										
U										
A										
N										

c. S (Source) = Perdagangan, T (Target) = Perjalanan

Tabel 5 Inisialisasi Perdagangan (S), Perjalanan (T)

		P	E	R	D	A	G	A	N	G	A	N
P												
E												
R												
J												
A												
L												
A												
N												
A												
N												

4.2 INISIALISASI BARIS DAN KOLOM

Langkah kedua adalah inisialisasi baris dan kolom. Dalam langkah ini membuat matriks dengan menggunakan sumber kata (S) dan target kata (T) yang nanti diisikan dengan nilai 0-N (Source) dan 0-M (Target).

a. S (Source) = Sirat, T (Target) = Surat

Tabel 6 Inisialisasi Baris dan Kolom Sirat (S), Surat (T)

		S	I	R	A	T
	0	1	2	3	4	5
S	1					
U	2					
R	3					
A	4					
T	5					

b. S (Source) = Penjualan , T (Target) = Pengajuan

Tabel 7 Inisialisasi Baris dan Kolom Penjualan (S), Pengajuan (T)

		P	E	N	J	U	A	L	A	N
	0	1	2	3	4	5	6	7	8	9
P	1									
E	2									
N	3									
G	4									
A	5									
J	6									
U	7									
A	8									
N	9									

c. S (Source) = Perdagangan, T (Target) = Perjalanan

Tabel 8 Inisialisasi Baris dan Kolom Perdagangan (S), Perjalanan (T)

		P	E	R	D	A	G	A	N	G	A	N
	0	1	2	3	4	5	6	7	8	9	10	11
P	1											
E	2											
R	3											
J	4											
A	5											
L	6											
A	7											
N	8											
A	9											
N	10											

4.3 PROSES PERHITUNGAN

Ini adalah langkah terakhir dalam mencari distance atau jarak antara dua string yang telah di bandingkan sebelumya. Untuk menentukan hasilnya bias dilihat pada bagian pojok kanan bawah tabel.

	(j)	S	I	R	A	T
(i)	0 +edit	1 +1	2	3	4	5
S	1 +1	0				
U	2					
R	3					
A	4					
T	5					

		S (j	I	R	A	T
(i)	0	1 +edit	2+1	3	4	5
S	1	0+1	1			
U	2					
R	3					
A	4					
T	5					

		S	I(j)	R	A	T
i	0	1	2+edit	3+1	4	5
S	1	0	1+1	2		
U	2					
R	3					
A	4					
T	5					

		S	I	R(j)	A	T
i	0	1	2	3+edit	4+1	5
S	1	0	1	2+1	3	
U	2					
R	3					
A	4					
T	5					

		S	I	R	A(j)	T
i	0	1	2	3	4	5
S	1	0	1	2	3 +1	4
U	2					
R	3					
A	4					
T	5					

	(j)	S	I	R	A	T
	0	1	2	3	4	5
S(i)	1+edit	0+1	1	2	3	4
U	2+1	1				
R	3					
A	4					
T	5					

Untuk Mendapatkan hasil "0" pada kotak merah menggunakan rumus [j+1], [i+1],[+edit]. Dimana yag berwarna hijau adalah "j" dan yang berwarna kuning adalah "i".Untuk "+edit" itu dipergunakan sebagai patokan untuk menghitung kesamaan huruf antara i dan j. Menentukan "+edit" itu sendiri juga ada aturannya, yaitu ketika pada huruf yang bertempatan antara "j" dan "i" sama maka nilai "+edit" nya adalah "0". Tetapi jika huruf yang bertemu antara "j" dan "i" berbeda maka nilai "+edit" nya adalah "1". Ketika sudah mendapatkan hasil dari perhitungan tersebut, cari nilai minimumnya. Karena nilai minimum itulah yang akan menjadi hasilnya. Ulangi perhitungan tersebut pada semua kolom. Hasil akhir keseluruhan terdapat pada kolom pojok kanan bawah.

a. S (Source) = Sirat, T (Target) = Surat

Tabel 9 Hasil Perhitungan Sirat (S), Surat (T)

		S	I	R	A	T
	0	1	2	3	4	5
S	1	0	1	2	3	4
U	2	1	0	1	2	3
R	3	2	1	1	2	3
A	4	3	2	2	1	2
T	5	4	3	3	2	1

b. S (Source) = Penjualan , T (Target) = Pengajuan

Tabel 10 Hasil Perhitungan Penjualan (S), Pengajuan (T)

И	9	8	Л	Л	Р	С	Т	З	Т	Т
У	8	Л	Р	Р	С	Т	З	Т	С	Т
Г	Л	Р	С	С	Т	З	Т	Т	Т	Т
Т	Р	С	Т	Т	З	З	З	З	Т	С
У	С	Т	З	З	З	З	З	Т	С	С
С	Т	З	З	З	І	З	З	Т	С	Р
И	З	З	І	І	З	З	Т	С	Р	Р
Е	З	І	0	І	З	З	Т	С	Р	Л
Ь	І	0	І	З	З	Т	С	Р	Л	8
	0	І	З	З	Т	С	Р	Л	8	9
		Ь	Е	И	Т	Г	У	Г	У	И

c. S (Source) = Perdagangan, T (Target) = Perjalanan

Tabel 11 Hasil Perhitungan Perdagangan (S), Perjalanan (T)

И	10	9	8	Л	Р	Р	С	С	Т	З	З
У	9	8	Л	Р	С	С	Т	Т	З	З	Т
И	8	Л	Р	С	Т	Т	З	З	З	Т	С
У	Л	Р	С	Т	З	З	З	З	Т	С	С
Г	Р	С	Т	З	З	З	І	З	З	Т	С	Р
У	С	Т	З	З	І	І	З	З	Т	С	Р	Р
Т	Т	З	З	І	0	І	З	З	Т	С	Р	Л
К	З	З	І	0	І	З	З	Т	С	Р	Л	8
Е	З	І	0	І	З	З	Т	С	Р	Л	8	9
Ь	І	0	І	З	З	Т	С	Р	Л	8	9	10
	0	І	З	З	Т	С	Р	Л	8	9	10	11
		Ь	Е	К	D	У	С	У	И	С	У	И

Hasil perbandingan kata pada tabel 9 memiliki hasil jarak pada kata sirat dan surat mendapatkan hasil jarak perbandingan 1. Sedangkan pada tabel 10 pada kata penjualan dan pengajuan memiliki hasil jarak perbandingan 4. Dan pada tabel 11 memiliki hasil jarak perbandingan 3 dari kata perdagangan dan perjalanan.

5. KESIMPULAN

1. Dapat membantu dalam mempermudah proses pencarian arsip surat pada aplikasi pengolahan surat di CV. Khasanah Konsultama.
2. Metode algoritma levenshtein distance menggunakan matrik perbandingan dengan rumus [j+1],[i+1],[+edit].
3. Dari hasil perhitungan dengan menggunakan metode algoritma levenshtein distance memiliki hasil perbandingan kata yang benar akan ditampilkan.

6. REFERENSI

Adriyani, N. M. M., Santiasa, I. W. & Muliantara, A., N.D. Implementasi Algoritma Levenshtein Distance Dan Metode Empiris Untuk Menampilkan Saran Perbaikan Kesalahan Pengetikan Dokumen Berbahasa Indonesia.

Ilmy, M. B., Rahmi, N. & Bu'ulolo, R. L., N.D. Penerapan Algoritma Levenshtein Distance Untuk Mengoreksi Kesalahan Pengejaan Pada Editor Teks.

Jogianto, 2008. Terstruktur Teori Dan Praktek Aplikasi Bisnis. *Analisis Dan Desain Sistem Informasi.*

Kadir, A., 2014. Pengenalan Sistem Informasi Dan Revisi.

Luqman, M., 2013. Pembangunan Sistem Informasi Manajemen Surat Masuk Dan Surat Keluar Pada Bagian Umum Sekretariat Daerah Kabupaten Pacitan. *Sentra Penelitian Engineering Dan Edukasi,* Volume 5, P. 22.