ARTIKEL

APLIKASI PENDETEKSI KEMIRIPAN ISI TEKS TUGAS SISWA MENGGUNAKAN ALGORITMA *LEVENSHTEIN DISTANCE*

Oleh :

M. FAHRUR AZHRI

14.1.03.02.0098

Dibimbing Oleh :

1. **Daniel Swanjaya, M.Kom**
2. **Ratih Kumalasari Niswatin, S.ST, M.Kom**

PROGAM STUDI TEKNIK INFORMATIKA

FAKULTAS TEKNIK

UNIVERSITAS NUSANTARA PGRI

2019

SURAT PERNYATAAN
ARTIKEL SKRIPSI TAHUN 2019

Yang bertanda tangan di bawah ini:

Nama Lengkap	:	M. Fahrur Azhri
NPM	:	14.1.03.02.0098
Telepun/HP	:	085856894144
Alamat Surel (Email)	:	amfahrur@gmail.com
Judul Artikel	:	Aplikasi Pendeteksi Kemiripan Isi Teks Tugas Siswa Menggunakan Algoritma *Levenshtein Distance*
Fakultas – Program Studi	:	Teknik / Teknik Informatika
Nama Perguruan Tinggi	:	Universitas Nusantara PGRI Kediri
Alamat Perguruan Tinggi	:	Jln. KH. Ahmad Dahlan No. 76 Mojoroto, Kediri, Jawa Timur

Dengan ini menyatakan bahwa :

a. artikel yang saya tulis merupakan karya saya pribadi (bersama tim penulis) dan bebas plagiarisme;
b. artikel telah diteliti dan disetujui untuk diterbitkan oleh Dosen Pembimbing I dan II.

Demikian surat pernyataan ini saya buat dengan sesungguhnya. Apabila di kemudian hari ditemukan ketidaksesuaian data dengan pernyataan ini dan atau ada tuntutan dari pihak lain, saya bersedia bertanggungjawab dan diproses sesuai dengan ketentuan yang berlaku.

Mengetahui		Kediri, 7 Februari 2019
Pembimbing I	Pembimbing II	Penulis,
Daniel Swanjaya, M.Kom NIDN. 0723098303	Ratih Kumalasari Niswatin, S.ST, M.Kom NIDN. 0710018501	M. Fahrur Azhri NPM. 14.1.03.02.0098

APLIKASI PENDETEKSI KEMIRIPAN ISI TEKS TUGAS SISWA MENGGUNAKAN ALGORITMA *LEVENSHTEIN DISTANCE*

M. Fahrur Azhri
14.1.03.02.0098
Teknik Informatika
amfahrur@gmail.com
Daniel Swanjaya, M.Kom dan Ratih Kumalasari Niswatin, S.ST, M.Kom
UNIVERSITAS NUSANTARA PGRI KEDIRI

ABSTRAK

Banyak terjadinya tindakan plagiarisme di kalangan siswa SMA ketika mengumpulkan tugas serta kuranganya sistem yang mampu mendeteksi kemiripan isi tugas tugas siswa. Dari 35 siswa dalam satu kelas terdapat 24 siswa yang menjiplak dengan berbagai jenis penjiplakan seperti mengambil ide yang sudah ada tanpa menyebut sumber dengan jelas, mengambil data penelitian orang lain, menggunakan kata, kalimat, paragraf yang sama dan bahkan ada yang melakukan penjiplakan kesuluruhan.

Penelitian ini menggunakan algoritma *Levenshtein Distance* untuk melakukan deteksi kemiripan teks. Dalam sistem yang dibuat, terdapat 3 *role* pengguna yakni *role admin, teacher*, dan *student*. *Admin* bertugas menghandel semua sistem yang berhubungan dengan administrasi, *teacher* bertugas untuk memberikan nilai kepada siswa, *student* bertugas melakukan *upload* tugas kedalam sistem.

Dari sistem yang telah dibuat, dokumen sebelum dilakukan proses pengecekan melewati tahapan *text preprocessing* yang terdiri dari *Tokenizing, Purifying, Stopword Removal, Stemming, dan Sortir.* Setelah melewati tahapan *text preprocessing* selanjutnya dilakuan perhitungan menggunakan algoritma *Levenshtein Distance* dengan batasan kemiripan teks (*threshold)* adalah 70%, jika kemiripan teks siswa kurang dari 70%, maka dokumen akan diterima sistem, jika terdapat siswa mendapat kemiripan teks melebihi 70%, maka dokumen tersebut akan ditolak oleh sistem dan siswa tidak bisa melakukan *upload* tugas yang sama. Hal ini dibuat untuk memberikan efek jera kepada siswa agar tidak melakukan tindakan plagiarisme. Kesimpulan hasil dari penelitian ini adalah dengan adanya sistem ini dapat membantu guru khusus nya pihak intansi pendidikan dalam mendapatkan dokumen yang *original* serta mengurangi tindakan plagiarisme sejak dini di mulai dari tingkat SMA.

KATA KUNCI : Kemiripan Teks, Plagiarisme, *Levenshtein Distance*.

I. Latar Belakang

Pengajaran menulis di kalangan siswa kelas menengah atas saat ini sedang gencar dilakukan sebagai persiapan siswa ke jenjang kerja atau perguruan tinggi. Bersamaan dengan itu perkembangan teknologi komunikasi melalui internet baik dari komputer, laptop, atau *gadget* siswa juga mengalami banyak berkembangan dan pemberdayaan. guru dengan menggunakan *blended learning* di era digital ini juga memiliki tugas penting untuk tetap menjaga intelijensi asli siswa dalam menulis terutama masalah plagiarisme.

Penjiplakan atau plagiarisme berarti mencontoh atau meniru atau mencuri tulisan dan karya orang lain yang kemudian diakui sebagai karangannya sendiri dengan ataupun tanpa seizin penulisnya. Penjiplakan dokumen digital bukanlah hal yang susah, cukup dengan menggunakan teknik *copy-paste-modify* pada sebagian isi dokumen dan bahkan keseluruhan isi dokumen sudah bisa dikatakan bahwa dokumen tersebut merupakan hasil duplikasi dari dokumen lain.

Sebagai pendidik, seorang guru harus mampu menepis kebiasaan buruk yang dapat melanggar hukum di kemudian hari di lingkup perguruan tinggi dan lingkungan kerja berkenaan dengan Undang – Undang Hak Cipta dan plagiat. Dalam studi ini, peneliti menggunakan rekayasa perangkat untuk mengurangi tindak plagiarisme di lingkungan sekolahDalam studi ini, peneliti menggunakan rekayasa perangkat untuk mengurangi tindak plagiarisme di lingkungan sekolah. Dengan menggunakan aplikasi pendeteksi kemiripan isi teks tugas siswa menggunakan algoritma *levenshtein distance,* tindak plagiarisme secara perlahan dapat ditekan dan untuk mendukung penekanan tindak plagiarisme ini di lingkungan sekolah. Oleh karena itu, dibutuhkan adanya sistem yang memudahkan dalam mendeteksi dan mengukur kemiripan dokumen. Selain dapat mengetahui tindak plagiarisme, pengukuran kemiripan dokumen ini dapat membantu dalam pengelompokan dokumen. Sebagian besar kasus plagiarisme ditemukan di kalangan siswa, berupa mengambil ide yang sudah ada tanpa menyebut sumber dengan jelas, mengambil data penelitian orang lain, menggunakan kata, kalimat, paragraf yang sama dan bahkan ada yang melakukan penjiplakan kesuluruhan, dan

sebagainya. Deteksi plagiarisme ini dilakukan dengan membandingkan sebuah dokumen dengan dokumen lainnya. Tingkat kesamaan dokumen tersebut akan menjadi dasar pendeteksian plagiarisme dari tugas yang diberikan oleh guru.

Penggunaan metode *levenshtein distance* menjadi sebuah matriks *string* yang digunakan untuk mengukur perbedaan atau jarak (*distance*) antara dua *string*. Nilai *distance* antara dua *string* ini ditentukan oleh jumlah minimum dari operasi - operasi perubahan yang diperlukan untuk melakukan transformasi dari suatu *string* menjadi *string* lainnya. Operasi-operasi tersebut adalah penyisipan (*insertion*), penghapusan (*deletion*), atau penukaran (*subtitution*). *Levenshtein distance* merupakan salah satu algoritma yang dapat digunakan dalam mendeteksi kemiripan antara dua *string* yang berpotensi melakukan tindak plagiarisme.

II. Metode

A. Simulasi Perhitungan Algoritma *Levenshtein Distance*

Algortima *Levenstein Distance* adalah algortima yang mengukur kesamaan antara 2 *string*, nantinya akan dikenal dengan *string* sumber (s) dan *string* target (t). Sebagai contoh dibawah :

Jika (s) = “coba” dan (t) = “coba” => maka jarak perbedaan antara kedua *string* tersebut adalah 0, karena tidak ada perubahan. Jika (s) = “coba” , dan (t) = “cona” => maka nilai *distance* antara kedua *string* diatas adalah (s,t) = 1, karena adanya perubahan antara kedua *string* tersebut di huruf yang ketiga yakni “b” dengan “n”.

Langkah – langkah algoritma *lavenshtein distance sebagai berikut :*

1) Langkah pertama
   a) *Set variabel* N yang menyimpan panjang *string source* (S)
   b) *Set variabel* M yang menyimpan panjang *string* target (T)
   c) Jika N = 0 atau M=0 maka *exit*
   d) Buat matriks ber-ordo [0 – N][0 – M]
2) Langkah kedua
   a) Inisialisasi baris 0 – N
   b) Inisialisasi kolom 0 – M
3) Langkah ketiga

   Periksa setiap karakter dari

string S (*looping* dari 1 ke N -> variabel i)

4) Langkah keempat

   Periksa setiap karakter dari *string* T (*looping* dari 1 ke M -> variabel j)

5) Langah kelima

   a) jika S[i] = T[i] –>*cost* = 0

   b) jika S[i] != T[i] –> *cost* = 1

6) Langkah keenam

   Set value d[i,j] yang diambil dari minimum jumlah :

   d[i-1,j] + 1

   d[i,j-1] + 1

   d[i-1,j-1] + *cost*

7) Langkah ketujuh

   Setelah langkah 3,4,5,6 selesai (tidak ada *looping* lagi), hasilnya akan ketemu di *element* d[N,M].

   Contoh soal :

   Jika ada 2 buah kata x = BARU dengan y = BATU, berapakah perbedaan huruf dari kedua kata tersebut.

   Langkah 1&2

Tabel 2.1 Pembuatan Kolom Dalam Tahap 1 & 2

	X	B	A	R	U
Y	0	1	2	3	4
B	1				
A	2				
T	3				
U	4				

Langkah 3 – 6, ketika i = 1, dan j = 1

*x[i] = B , y[i] = B, *cost* = 0

kemudian ambil nilai minimal dari

d[i-1,j]+1 => d[1-1,1]+1 = 1

d[i,j-1]+1 => d[1,1-1]+1 = 2

d[i-1,j-1]+*cost* => d[1-1,1-1]+0=0

Dari langkah di atas , akan dihasilkan nilai *distance* 0, karena B dibandingkan B adalah sama , dan tidak ada perubahan. Berikut tabel dari perhitunggan di atas :

Tabel 2.2 Pembuatan Kolom Dalam Tahap 3 s/d 6

	X	B	A	R	U
Y	0	1	2	3	4
B	1	**0**			
A	2				
T	3				
U	4				

Lakukan langkah dengan tahap 3 sampai 6 sampai kotak terpenuhi seperti tabel 2.3 :

Tabel 2.3 Pembuatan Kolom Setelah Semua Langkah Selesai					
	X	B	A	R	U
Y	0	1	2	3	4
B	1	0	1	2	3
A	2	1	0	1	2
T	3	2	1	1	2
U	4	3	2	2	1

Tabel 2.3 adalah tabel setelah semua langkah dari langkah 1 sampai 7 dilakukan, ketika semua langkah sudah dilakukan , maka semua kolom akan terisi dengan nilai.

Dari contoh soal perhitungan di atas maka diperoleh nilai *distance* dari kata "BARU" dan "BATU" adalah 1 yakni huruf "r" diganti "t".

Untuk mengukur nilai kemiripan teks menggunakan persamaan di bawah ini :

$$Sim = 1 - \left(\frac{Dis}{MaxLenght}\right) x\ 100\ \%$$

Keterangan :

a) *Sim* = *Similarity value*
b) *Dis* = nilai jarak *distance*
c) *MaxLenght* = nilai terbesar dari kedua *string* dokumen

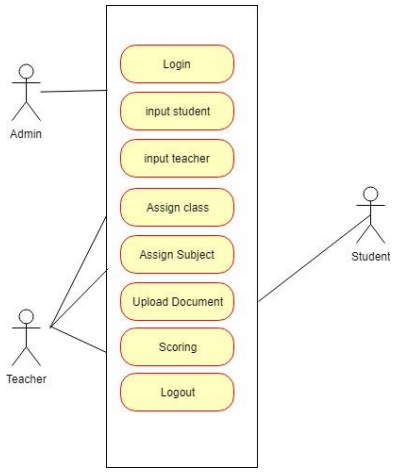
Jadi nilai kemiripan dari kata 'BATU' dan 'BARU' adalah :

$$Sim = 1 - (\frac{1}{4})\ x\ 100\% = 75\%$$

B. Implementasi

a. Use Case Diagram

Use Case Plagiarism Checker



Gambar 2.1. *Use Case* Aplikasi *Plagiarism Checker*

Keterangan *Use Case Plagiarism Checker* pada gambar 2.1 adalah sebagai berikut :

1) *Login* ini digunakan untuk masuk kedalam sistem sesuai *username* dan *password* masing-masing *role*.

2) Input student digunakan oleh *admin* untuk melakukan penambahan data siswa kedalam sistem.

3) *Input teacher* digunakan oleh *admin* untuk melakukan

penambahan data guru kedalam sistem.

4) *Assign class* digunakan oleh *admin* untuk mengatur pembagian kelas mengajar guru.

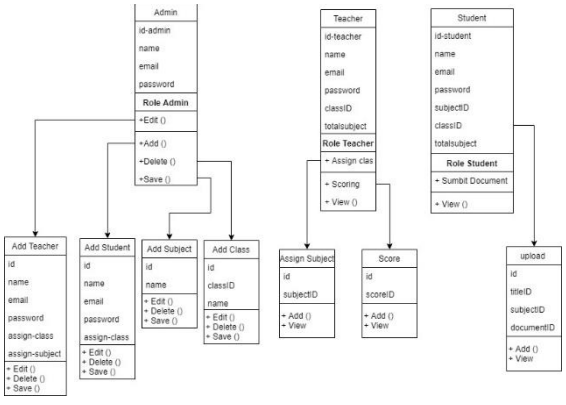
5) *Assign subject* digunakan oleh *admin* untuk memberikan hak akses guru terhadap mata pelajaran yang diampunya.

6) *Upload document* digunakan siswa untuk *upload* tugas yang telah dikerjakan kedalam sistem.

7) *Scoring* merupakan proses untuk memberikan penilaian kepada siswa.

8) *Logout* ini merupakan proses untuk keluar dari aplikasi.

b. *Class Diagram*



Gambar 2.2. Class Diagram Aplikasi *Plagiarism Checker*

Dari gambar 2.2, terdapat beberapa kelas yang saling berhubungan yang ada di dalam sistem, hubungan antar kelas yang ada dijelaskan sebagai berikut :

a. *Role Class Admin*, kelas ini merupakan kelas pengguna sistem yang memiliki tugas sebagai pengelola sistem dengan memiliki wewenang untuk mengedit, simpan, tambah dan menghapus data.
b. *Role Class Teacher*, kelas ini merupakan kelas pengguna yang memiliki tugas untuk memberikan penilaian kepada siswa sesuai kelas yang di ampunya.
c. *Role Class Student*, kelas ini merupakan kelas pegguna yang bertugas untuk melakukan upload tugas yang diberikan oleh guru kedalam sistem.
d. *Class Add Teacher*, kelas ini digunakan oleh *admin* untuk menambahkan data guru baru kedalam sistem
e. *Class Add Student*, kelas ini digunakan oleh *admin* untuk menambkan data siswa kedalam sistem.
f. *Class Add Subject*, kelas ini digunakan oleh *admin* untuk menambkan mata pelajaran kedalam sistem.
g. *Class Add Class*, kelas ini digunakan oleh *admin* untuk menambkan data kelas kedalam sistem.

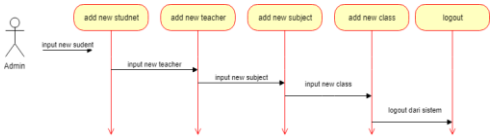
h. *Class Assign Subject*, kelas ini digunakan oleh *guru* untuk melakukan penambahan izin siswa kedalam kelas yang diampu oleh guru.

i. *Class Score*, kelas ini digunakan oleh guru untuk melakukan proses penilian terhadap tugas siswa yang telah dikumpulkan.

j. *Class Upload*, kelas ini digunakan oleh siswa untuk melakukan *upload* tugas kedalam sistem.

c. *Sequence Diagram*

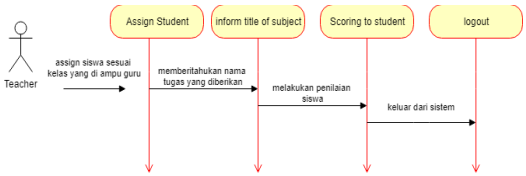
1) Sequence Diagram Admin



Gambar 2.4 Sequence Diagram Admin

Dari gambar 2.4 menggambarkan tugas yang dilakukan *admin* diawali dengan *add new student, add new teacher, add new subject, add new class*, kemudian *logout.*

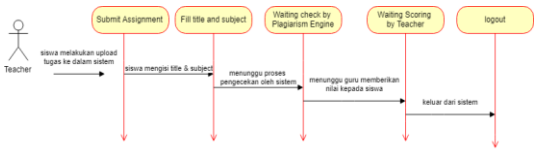
2) Sequence Diagram Teacher



Gambar 2.5 Sequence Diagram Teacher

Dari gambar 2.5 menggambarkan tugas yang dilakukan *teacher* diawali dengan *assign student, inform title of subject, scoring to student,* kemudian *logout.*

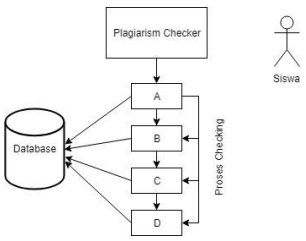
3) Sequence Diagram Student



Gambar 2.6 Sequence Diagram Student

Dari gambar 2.6 menggambarkan tugas yang dilakukan *student* diawali dengan *submit assignment, fill title and subject, waiting check by plagiarism engine, waiting score by teacher,* kemudian *logout.*

d. *Proses Checking*



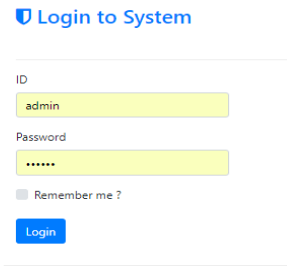
Gambar 2.7 Proses Checking

Gambar 2.7 menampilkan gambaran proses ketika dilakukan proses perbandingan antara dokumen 1 dengan dokumen lainnya. Siswa

yang pertama melakukan *upload* dokumen akan di inisialisasi dengan huruf 'A'. Kemudian jika ada siswa yang kedua melakukan *upload* dokumen ke sistem dalam hal ini di inisialisasi dengan huruf 'B' kemudian dokumen B akan di bandingkan dengan dokumen A, Sistem menerapkan batasan nilai kemiripan (*threshold*) sebesar 70%. Jadi, ketika nilai kemiripan teks siswa melebihi 70% ketika proses pengecekan berlangsung , dokumen akan ditolak oleh sistem dan tidak masuk kedalam *database.* Ketika dokumen B berhasil diterima, dan ada siswa ketiga ingin melakukan *upload* dokumen dalam hal ini diinisialisasi dengan huruf 'C'. Maka proses pengecekannya adalah dokumen 'C' akan dibandingkan dengan huruf 'A', jika nilai kemiripan teks kurang dari 70% , akan dilanjutkan dibandingkan dengan dokumen 'B'. Jika ketika dibandingkan dengan dokumen 'B' nilai kemiripan teks lebih dari 70%, maka dokumen ditolak sistem, ketika kurang dari 70% maka akan diterima oleh sistem dan masuk kedalam *database.* Proses selanjutnya akan sama jika terdapat siswa baru yang berhasil melakukan *upload* kedalam sistem.

III. Hasil

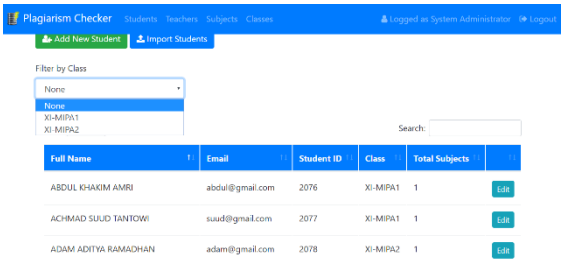
a. Halaman *Login*



Gambar 3.1. Login Aplikasi

Tampilan Login berfungsi untuk pengguna dapat masuk kedalam sistem dengan memasukkan *username* dan *password* yang sesuai terlebih dahulu. Tampilan *login* dapat dilihat pada gambar 3.1.

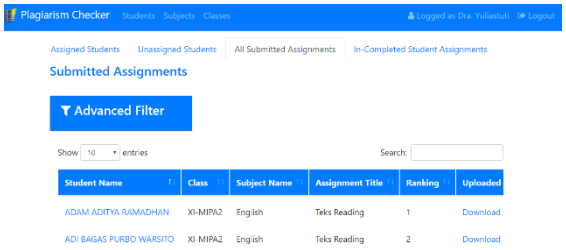
b. Menu Admin



Gambar 3.1. Tampilan *Dasboard Admin*

Dasboard admin terdiri dari tab *menu student, teacher, subject, dan class*. *Admin* mempunyai kedudukan tertinggi didalam sistem karena mempunyai hak untuk melakukan *add, edit, delete, dan save.*

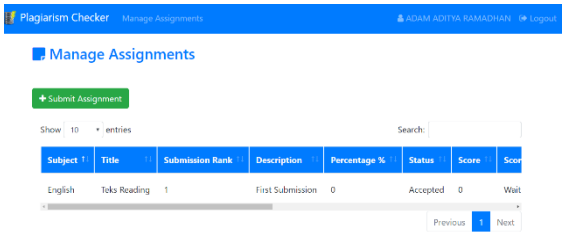
c. Menu Teacher



Gambar 3.2 Tampilan Dasboard Teacher

Dasboard teacher secara keseluruhan terdiri dari *student, subject, dan class.* Sedangakan *Tab subject* terdiri dari *assigned student, unsigned student, all sumbited assignment,* dan *in-completed student assignment.*

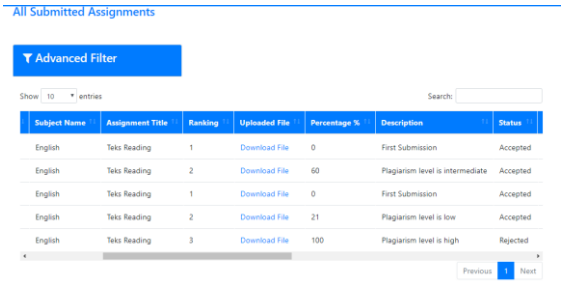
d. Menu Student



Gambar 3.3. Tampilan *Dasboard Student*

Tampilan *dasboard student* terdiri dari *tab manage assignment, submit assignment* , dan *log* informasi aktivitas pengiriman tugas.

e. Result Plagiarism Engine



Gambar 3.4 Hasil Plagiarisme Checker

Gambar 3.4 merupakan monitor hasil dari proses *plagiarism checker.* Ketika tugas siswa nilai kemiripan teksnya kurang dari batasan 70% , makan akan berstatus *accept,* tetapi ketika tugas siswa nilai kemiripan teksnya lebih dari 70%, maka akan berstatus *rejected*.

IV. Penutup

a. Kesimpulan

Setelah melalui beberapa tahapan dalam menyelesaikan aplikasi pendeteksi kemiripan isi teks tugas siswa menggunakan algoritma *levenshtein distance* didapatkan kesimpulan sebagai berikut :

1. Dihasilkan sebuah aplikasi yang mampu mendeteksi kemiripan teks dokumen tugas siswa.
2. Aplikasi menerapkan batasan kemiripan teks (*threshold)* sebesar 70%, ketika dokumen siswa setelah melewati semua tahapan di aplikasi nilai kemiripan teks kurang dari 70%, maka dokumen akan diterima

oleh sistem kemudian masuk kedalam database. Akan tetapi ketika nilai kemiripan teks lebih dari 70%, maka akan di tolak oleh sistem dan tidak masuk kedalam *database*.

3. Aplikasi ini berguna membantu guru khusus nya untuk pihak instansi pendidikan dalam mendapatkan tugas siswa yang *original*.

b. Saran

Pada penulisan skripsi ini tentu masih terdapat kekurangan yang dapat disempurnakan lagi pada pengembangan sistem berikutnya. Beberapa saran yang dapat dipergunakan diantaranya :

1. Inputan aplikasi sebatas teks, belum mengenali file bergambar.
2. Penyempurnaan fitur lain untuk menambah kenyamanan pengguna.

V. Daftar Pustaka

Irianto, WA. 2014. *Penentuan Tingkat Plagiarisme Dokumen Penelitian Menggunakan Centroid Linkage Hierarchical Method (CLHM)*. Jurnal. Program Teknologi Informasi dan Ilmu Komputer, Universitas Brawijaya.Malang.

Pratama, B. P. & Pamungkas, S. A. 2016. *Analisis Kinerja Algoritma Levenshtein Distance dalam Mendeteksi Kemiripan Dokumen Teks*. Jurnal Logika, Jilid 6, No. 2, 2016, Hal. 131-143I.

Obed, K. 2012. *Implementasi Algoritma Winnowing Untuk Mendeteksi Kemiripan Pada Dokumen Teks.* Jurnal Informatika Vol. 9 No.1.

Sukmana,dkk. 2018. *Perbandingan Penggunaan Stemming Pada Deteksi Kemiripan Dokumen Menggunakan Metode Rabin Karp dan Jaccard Similarity.* Seminar Nasional Teknologi Informasi dan Multimedia. Yogyakarta.

Tudesman, 2013. *Sistem Deteksi Plagiarisme Dokumen Bahasa Indonesia Menggunakan Metode Vector Space Model, Skripsi, Program Studi Teknik Informatika*, STIMIK GI MDP.