JURNAL TEKNISI **E-ISSN: 2775-0965**
(Jurnal Teknologi Komputer dan Sistem Informasi)
Februari 2021, Vol 1, No 1, Hlm. 46 - 53
Available online at http://jurnal.goretanpena.com/index.php/teknisi

SISTEM TEMU KEMBALI INFORMASI DENGAN *LATENT SEMANTIC ANALISYS* PADA KESAMAAN TUGAS AKHIR MAHASISWA

Meri Sri Wahyuni, Dedi Setiawan, Trinanda Syahputra
STMIK Triguna Dharma, Medan
e-mail: Meri.sriwahyuni@gmail.com, setiawandedi07@gmail.com,
trinandasyahputra@gmail.com

Abstract: The Final Project is a scientific work that is made as a report on the results of student research for graduation requirements and obtaining a bachelor's degree. Each final project produced must not be the same from one work to another. If there is the same, the author will be considered plagiarism. Plagiarism is an act intentionally or unintentionally in obtaining or trying to obtain credit or value for a scientific work, by quoting part or all of the scientific work of another party which is recognized as his scientific work, without stating the source accurately and adequately. The Latent Semantic Analysis (LSA) method is used to find documents that have similar text to other documents in the database, with using TF/IDF Algoritms, to check the degree of similarity term using Vector Space Models (VSM).

Keywords: Final Project, Plagiarism, Latent Semantic Analysis,TF/IDF, Vector Space Model

Abstrak: Tugas Akhir (Skripsi) merupakas sebuah karya ilmiah yang diabuat sebagai laporan hasil penelitian mahasiswa untuk syarat lulus dan meraih gelar sarjana. Setiap Tugas Akhir yang dihasilkan tidak boleh sama antara satu karya dengan karya lainnya. Apabila ada yang sama maka penulis akan dianggap plagiat. Plagiat adalah perbuatan secara sengaja atau tidak sengaja dalam memperoleh atau mencoba memperoleh kredit atau nilai untuk suatu karya ilmiah, dengan mengutip sebagian atau seluruh karya ilmiah pihak lain yang diakui sebagai karya ilmiahnya, tanpa menyatakan sumber secara tepat dan memadai. Metode Latent Semantic Analysis (LSA) berfungsi untuk mencari dokumen yang memiliki kesamaan teks terhadap dokumen lain yang ada di dalam database. Algoritma yang digunakan adalah Algoritma TF/ID, dan untuk melihat tingkat kedekatan atau kesamaan (smilarity) kata (term) dengan cara pembobotan term digunakan Vector Space Models (VSM).

Kata kunci: Tugas Akhir, Plagiat, Latent Semantic Analysis,TF/IDF, Vector Space Model.

PENDAHULUAN

Tugas Akhir Mahasiswa merupakan laporan hasil penelitian mahasiswa yang didokumentasikan untuk salah satu syarat meraih gelar sarjana pada sebuah perguruan tinggi. Laporan penelitian merupakan penelitian ilmiah terhadap suatu gejala. Dengan adanya Internet memberikan kemudahan bagi orang dalam berbagi ataupun melakukan pencarian dan pengambilan dokumen. Ada banyak publikasi ilmiah secara online dapat membantu mahasiswa dalam mencari referensi maupun bahan pertimbangan judul dalam pembuatan laporan karya ilmiahnya. Tetapi dengan mudahnya juga melakukan plagiat terhadap karya ilmiah lainnya. Plagiat

JURNAL TEKNISI **E-ISSN: 2775-0965**
(Jurnal Teknologi Komputer dan Sistem Informasi)
Februari 2021, Vol 1, No 1, Hlm. 46 - 53
Available online at http://jurnal.goretanpena.com/index.php/teknisi

merupakan perbuatan secara sengaja atau tidak sengaja dalam memperoleh atau mencoba memperoleh kredit atau nilai untuk suatu karya ilmiah, dengan mengutip sebagian atau seluruh karya ilmiah pihak lain yang diakui sebagai karya ilmiahnya, tanpa menyatakan sumber secara tepat. Untuk meminimalisir atau menghindari plagiat pada laporan tugas akhir mahasiswa maka dilakukan pendeteksi terhadap kesamaan dokumen.

Melalui pembahasan dalam tulisan ini, metode *Latent Semantic Analysis* (LSA) digunakan untuk mencari dokumen yang memiliki kesamaan teks terhadap dokumen lain yang ada di dalam database. Metode ini memiliki kemampuan untuk menemukan hubungan tersembunyi (*latent*) antara semua *term* (kata) yang memiliki kedekatan makna secara kontekstual. Metode ini memberikan solusi untuk masalah kata-kata sinonim dan polisemi yang sering terjadi dalam temu-kembali informasi .

Latent Semantic Analysis (LSA) atau disebut juga dengan *Latent Semantic Indexing (LSI)*, merupakan sebuah metode *automatic indexing* dan *retrival* yang memanfaatkan *semantic structure* (struktur asosiasi term dengan dokumen) yang secara implisit terdapat dalam suatu dokumen untuk digunakan dalam pencarian dokumen yang relevan dengan *term* dalam *query*.

Untuk mengukur derajat kemiripan antara suatu dokumen dalam suatu *query* menggunakan *Vector Space Model*. Dalam *Vector Space Model*, koleksi dokumen ditampilkan dalam bentuk matrik term dokumen atau matrik term frekuensi.

Untuk membuktikan hasil yang didapat oleh metode *Latent Semantic Analysis (LSA)* maka digunakan Algoritma *Term Frequency-Inverse Document Frequency (TF-IDF)*. Algoritma TF-IDF hasil perkalian nilai TF dengan IDF untuk sebuah term dalam dokumen.

Term Frequency/Inverse Document Frequency

TF adalah jumlah munculnya suatu term dalam suatu dokumen, IDF adalah perhitungan logaritma pembagian jumlah dokumen dengan frekuensi dokumen yang memuat suatu term, dan TF/IDF adalah hasil perkalian nilai TF dengan IDF untuk sebuah term dalam dokumen (Ardytha Luthfiarta, dkk, 2013). Persamaan IDF dan TFIDF dapat dilihat pada persamaan dibawah ini:

$$W_{d,t} = tf_{d,t} * IDF_t$$

$$IDF = \log \frac{D}{DF}$$

$$TFIDF_t = TF * \log \frac{D}{DF}$$

Dimana:

d = dokumen ke-d
t = kata ke-t dari kata kunci
W = bobot dokumen ke-d terhadap kata ke-t
tf/TF = banyaknya kata yang dicari pada sebuah dokumen
IDF = log (D/df)
D = total dokumen
df = banyak dokumen yang mengandung kata yang dicari

Latent Semantic Analysis

LSA (*Latent Semantic Analysis*) adalah metode statistik aljabar yang mengekstrak struktur semantik yang tersembunyi dari kata dan kalimat. LSA ini menggunakan konteks yaitu memasukkan dokumen dan mengekstrak informasi dari kata yang digunakan bersama dan kata-kata umum yang sering dilihat pada kalimat yang berbeda. Jika

JURNAL TEKNISI **E-ISSN: 2775-0965**
(Jurnal Teknologi Komputer dan Sistem Informasi)
Februari 2021, Vol 1, No 1, Hlm. 46 - 53
Available online at http://jurnal.goretanpena.com/index.php/teknisi

jumlah dari kata-kata umum pada kalimat dalam jumlah banyak, itu berarti kalimat tersebut lebih banyak bersifat semantik. Algoritma peringkas dokumen teks yang berbasis pada LSA ini biasanya terdiri dari tiga tahap, yaitu pembentukan matrik input, dokumen yang diinput ditunjukkan dengan matrik untuk menampilkan kalkulasi, dan penyeleksian kalimat (Muhammad Jamhari et al, 2014).

Vector Space Model

Vector Space Model (VSM) adalah metode untuk melihat tingkat kedekatan atau kesamaan (*smilarity*) kata (*term*) dengan cara pembobotan term. Dokumen dipandang sebagi sebuah vektor yang memiliki jarak (*magnitude*) dan arah (direction). Pada Vector Space Model, sebuah istilah direpresentasikan dengan sebuah dimensi dari ruang vektor. Relevansi sebuah dokumen ke sebuah query didasarkan pada similaritas diantara vektor dokumen dan vektor query.

Cara kerja dari vector space model adalah dengan menghitung nilai cosines sudut dari dua vector, yaitu vektor kata kunci terhadap vektor tiap dokumen. Perhitungan vektor space model menggunakan persamaan (1), (2) dan (3).

$$Co\sin e\theta_{D_i} = Sim(Q, D_i) \quad \text{..... (1)}$$

Dimana :
Q = kata kunci (query)
Di = dokumen ke-i

$$Sim Q, D_i = \frac{\sum_j W_{i,j} W_{q,j}}{\sqrt{\sum_j W_{i,j}^2}\sqrt{\sum_j W_{q,j}^2}} \quad \text{..... (2)}$$

Dimana :
Di = dokumen ke-i
q = kata kunci (query)
j = Kata diseluruh dokumen

$$Co\sin e\theta_{D_i} = \frac{Q, D_i}{|Q| * |D_i|} \quad \text{..... (3)}$$

Dimana :
Di = dokumen ke-i
Q = kata kunci (*query*)
|Q| = Vektor Q
|Di| = Vektor Di

METODE

Pada tahap ini akan dilakukan penganalisaan pada Latent Semantic Analisys, yang mana tahap-tahapnya adalah sebagai berikut:

1. Menghitung nilai term frequency (tf).
2. Menghitung nilai document frequency (df).
3. Menghitung invers document frequency (idf).
4. Menghitung bobot dokumen (W).
5. Menghitung perkalian nilai bobot kata kunci (query) dokumen (WD) dengan nilai bobot dokumen ke-i (Wdi), jumlahkan hasil dari perkalian nilai bobot tersebut.
6. Menghitung panjang vektor tiap dokumen dan kata kunci (query).
7. Menghitung similaritas.

Hasil akhir selanjutnya dihitung tingkat kemiripannya dengan kata kunci (query) menggunakan perhitungan vektor space model. Tahapan implementasi TF/IDF agar lebih jelas dibuat contoh kata kunci (Q=query) dan dokumen (D=4), seperti dibawah ini:

Kata kunci (Q) =Analisis pada dokumen otomatis menggunakan metode LSA
Dokumen 1 (D1) = Sistem analisis dokumen teks
Dokumen 2 (D2) = Penilaian sistem dokumen teks terhadap dokumen otomatis
Dokumen 3 (D3) = dalam sistem dokumen teks pada algoritma TF/IDF dan LSA
Dokumen 4 (D4) = Analisis sistem pada dokumen otomatis untuk proses clustering dokumen menggunakan LSA dan TF/IDF

JURNAL TEKNISI **E-ISSN: 2775-0965**
(Jurnal Teknologi Komputer dan Sistem Informasi)
Februari 2021, Vol 1, No 1, Hlm. 46 - 53
Available online at http://jurnal.goretanpena.com/index.php/teknisi

Melalui proses tokenizing selanjutnya masuk pada proses filtering (stopword dan stoplist), maka kata "pada" pada Q, kata "terhadap" pada D2, kata "dalam", "pada" dan tanda "/" pada D3, kata "pada", "untuk" dan "dan" pada D4 dihapus.

Selanjutnya, kumpulan kata yang telah terpilih dilakukan proses pembobotan dokumen melalui beberapa perhitungan dibawah ini.

1. Menghitung nilai term frequency (tf).

Tabel 1. **Nilai tf**

NO	Kata (*Term*)	Tf				
		Q	D1	D2	D3	D4
1	Analisis	1	1	0	0	1
2	Dokumen	1	1	2	1	2
3	Otomatis	1	0	1	0	1
4	Menggunakan	1	0	0	0	1
5	LSA	1	0	0	1	1
6	Sistem	0	1	1	1	1
7	Teks	0	1	1	1	0
8	Penilaian	0	0	1	0	0
9	Algoritma	0	0	0	1	0
10	Tf	0	0	0	1	0
11	Idf	0	0	0	1	0
12	Proses	0	0	0	0	1
13	Clustering	0	0	0	0	1

Setelah proses tokenizing dilakukan, maka dapat hasil nilai term frequency (tf) dari masing-masing kata pada kalimat dan terbentuk sebuah matriks pada setiap dokumen.

2. Menghiung nilai document frequency (df).

Document frequency (df) adalah banyaknya dokumen dimana suatu kata (term) muncul.

Tabel.2. Nilai df

NO	Kata (*Term*)	Df
1	Analisis	2
2	Dokumen	4
3	Otomatis	2
4	Menggunakan	1
5	LSA	2
6	Sistem	4
7	Teks	3
8	Penilaian	1
9	Algoritma	1
10	Tf	1
11	Idf	1
12	Proses	1
13	Clustering	1

Setelah hasil perhitungan tf dan df didapatkan, langkah selanjutnya dilakukan perhitungan inverse document frequency (idf) tiap kata (term) untuk menghitung bobot kata (term).

3. Menghitung inverse document frequency (idf).

Tabel 3. Nilai IDF

NO	Kata (*Term*)	D/df	IDF=log (D/df)
1	Analisis	4/2	0,301
2	Dokumen	4/4	0
3	Otomatis	4/2	0,301
4	Menggunakan	4/1	0,602
5	LSA	4/2	0,301
6	Sistem	4/4	0
7	Teks	4/3	0,124
8	Penilaian	4/1	0,602
9	Algoritma	4/1	0,602
10	Tf	4/1	0,602
11	Idf	4/1	0,602
12	Proses	4/1	0,602
13	Clustering	4/1	0,602

Selanjutnya, setelah nilai tf dan idf telah didapatkan, kemudian dimasukkan dalam perhitungan tf-idf weighting untuk menghitung bobot hubungan suatu kata (term) di dalam dokumen.

4. Menghitung bobot dokumen (W).

Tabel 4 Nilai W (bobot dokumen)

NO	Kata (*Term*)	$W_{dt}=tf_{dt}*IDF$				
		Q	D1	D2	D3	D4
1	Analisis	0,301	0,301	0	0	0,301
2	Dokumen	0	0	0	0	0
3	Otomatis	0,301	0	0,301	0	0,301
4	Menggunakan	0,602	0	0	0	0,602
5	LSA	0,301	0	0	0,301	0,301

JURNAL TEKNISI **E-ISSN: 2775-0965**
(Jurnal Teknologi Komputer dan Sistem Informasi)
Februari 2021, Vol 1, No 1, Hlm. 46 - 53
Available online at http://jurnal.goretanpena.com/index.php/teknisi

6	Sistem	0	0	0	0	0
7	Teks	0	0,124	0,124	0,124	0
8	Penilaian	0	0	0,602	0	0
9	Algoritma	0	0	0	0,602	0
10	Tf	0	0	0	0,602	0
11	Idf	0	0	0	0,602	0
12	Proses	0	0	0	0	0,602
13	Clustering	0	0	0	0	0,602

5. Menghitung perkalian nilai bobot kata kunci (query) dokumen (WD) dengan nilai bobot dokumen ke-i (Wdi).

Tabel 5. Perkalian WD*Wd_i

WD*Wd_i			
D1	**D2**	**D3**	**D4**
0,091	0	0	0,091
0	0	0	0
0	0,091	0	0,091
0	0	0	0,362
0	0	0,091	0,091
0	0	0	0
0	0	0	0
0	0	0	0
0	0	0	0
0	0	0	0
0	0	0	0
0	0	0	0
0	0	0	0

Selanjutnya, setelah menghitung perkalian WD*Wdi, dilakukan penjumlahan hasil (∑)dari perkalian nilai bobot tersebut. Dengan menggunakan rumus

$$\mathrm{SUM}(\mathrm{WD} * \mathrm{Wd_i}) = \sum_{j=1}^{n} \mathrm{WD_j Wd_{i,j}}$$

Maka di dapa nilai:
D1 = 0,091 D2 = 0,091
D3 = 0,091 D4 = 0,634

Setelah perkalian WD*Wdi dan penjumlahan WD*Wdi tersebut didapatkan, maka hitung panjang vektor tiap dokumen dan kata kunci (query).

6. Menghitung panjang vektor tiap dokumen dan kata kunci (query).
   a.Perhitungan vector dari query dengan menggunakan rumus

$$|Q| = \sqrt{\sum_i W_{q,j}^2}$$

Sehingga didapat hasil Q = 0,796

   b. Perhitungan vector dari dokumen dengan menggunakan rumus

$$|D_i| = \sqrt{\sum_i W_{i,j}^2}$$

Sehingga didapat hasil;
D1 = 0,326 D2 = 0,684
D3 = 1,092 D4 = 1,166

Setelah didapatkan perhitungan vector query dan dokumen, maka didapatkan nilai panjang vektor, seperti berikut.

Tabel 6. Nilai Panjang Vektor

PANJANG VEKTOR				
Q	**D1**	**D2**	**D3**	**D4**
0,091	0,091	0	0	0,091
0	0	0	0	0
0,091	0	0,091	0	0,091
0,362	0	0	0	0,362
0,091	0	0	0,091	0,091
0	0	0	0	0
0	0,015	0,015	0,015	0
0	0	0,362	0	0
0	0	0	0,362	0
0	0	0	0,362	0
0	0	0	0,362	0
0	0	0	0	0,362
0	0	0	0	0,362

Langkah selanjutnya adalah menghitung Similaritas antara vektor kata kunci (*query*) dengan tiap dokumen.

7. Menghitung Similaritas.
   Dalam menghitung similaritas dokumen digunakan rumus

$$Cosine\theta_{D_i} = \frac{Q, D_i}{|Q| * |D_i|}$$

JURNAL TEKNISI **E-ISSN: 2775-0965**
(Jurnal Teknologi Komputer dan Sistem Informasi)
Februari 2021, Vol 1, No 1, Hlm. 46 - 53
Available online at http://jurnal.goretanpena.com/index.php/teknisi

Berdasarkan rumus tersebut sehingga diperoleh:

Tabel 7 Perhitungan Similaritas

DOKUMEN	HASIL
D1	0,349
D2	0,166
D3	0,104
D4	0,683

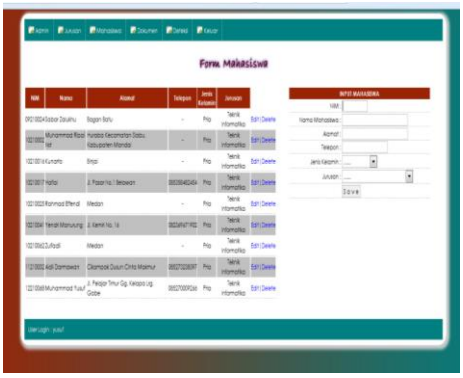
Dari hasil perhitungan di atas diketahui bahwa Dokumen 4 (D4) memilki tingkat similaritas tertinggi disusul dengan D1, D2 dan D3

HASIL DAN PEMBAHASAN

Pada pembahasan ini akan dilakukan pengujian system.

1. Halaman Input Mahasiswa

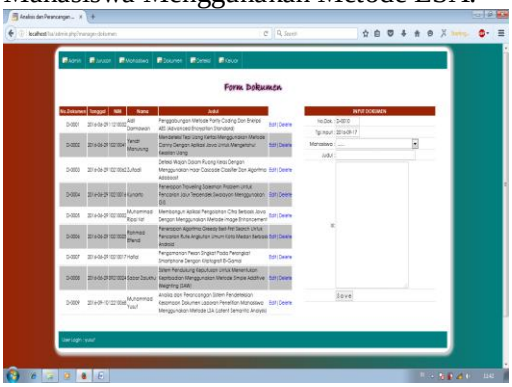
Halaman ini akan tampil jika user memilih form mahasiswa pada menu yang ada di halaman administrator, pada halaman ini seorang user dapat menambah, mengedit atau menghapus data mahasiswa.



Gambar 1. Halaman Input Mahasiswa

2. Halaman Input Dokumen

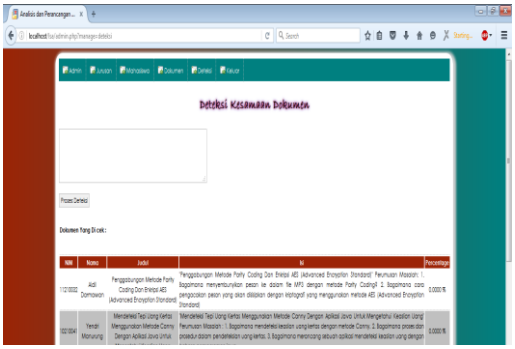
Halaman ini digunakan untuk menambah, mengedit atau menghapus data dokumen, di dalam data dokumen inilah nantinya akan dimasukkan dokumen-dokumen dari tugas akhir mahasiswa yang digunakan sebagai perbandingan untuk mencari data yang akan dicari untuk dideteksi pada Kesamaan Laporan Tugas Akhir Mahasiswa Menggunakan Metode LSA.



Gambar 2. Halaman Input Dokumen

3. Halaman Form Deteksi

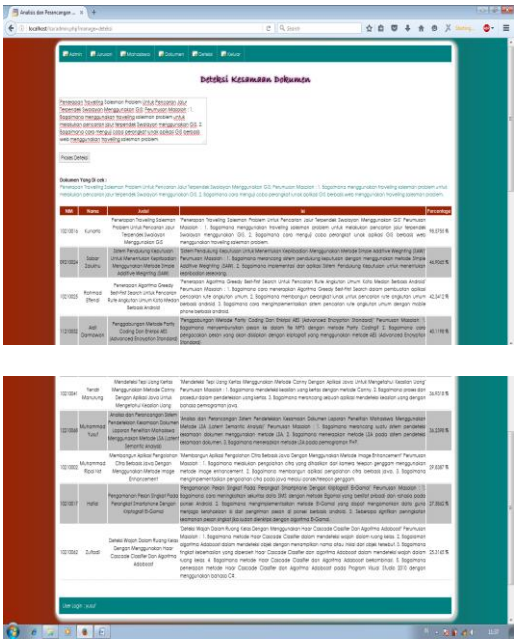
Halaman ini dapat mendeteksi string yang mau dicocokkan dengan dokumen-dokumen yang sudah terdaftar di database, cara pemakaian form deteksi ini user akan memasukkan string yang mau dideteksi ke dalam textarea bisa diketikkan manual bisa juga di salin dan tempelkan ke dalam textarea yang ada di dalam form deteksi ini, setelah string dimasukkan ke dalam textarea kemudian user mengklik tombol "Proses Deteksi" maka untuk hasilnya persentase kesamaan dokumen akan tampil di tiap-tiap baris record perbandingan, dimana nantinya persentase yang paling tinggi dianggap sistem sebagai string yang paling mendekati dari string yang mau dicari kesamaannya. Untuk pencocokkan string di dalam sistem digunakan metode LSA. Berikut adalah tampilan dari form deteksi dokumen.



Gambar 3. Halaman Form Deteksi

4. Pengujian Kasus

Pada pengujian kasus ini string yang akan di cari kesamaannya dengan record yang sudah terdaftar di database. Melalui pengujian ini maka akan didapatkan persentase kesamaan pada masing-masing record, dimana persentase yang paling tinggi merupakan dokumen yang paling mendekati ke string yang mau dicari, berikut adalah hasil dari deteksi kesamaan dokumen.



Gambar 4. Pengujian Kasus

SIMPULAN

Berdasarkan pembahasan dan evaluasi maka dapat diambil kesimpulan bahwa dalam menentukan kata kunci sebagai acuan pendeteksian dekumen-dokumen yang ada di database. Sistem pendeteksi plagiat yang dibangun menggunakan Latent Semantic Analysis (LSA) dengan bantuan preprocessing dapat menghasilkan nilai similarity yang lebih akurat dan mendeteksi beberapa tipe plagiat dengan baik. Penerapan metode LSA adalah dengan menghitung nilai term freuency (tf) pada dokumen database.

DAFTAR PUSTAKA

[1] Ali Ridho Barakbah, Tita Karlita, Ahmad Syauqi Ahsan, (2013), Logika dan Algoritma.

[2] Ardytha Luthfiarta, Junta Zeniarja, dan Abu Salam (2013) Algoritma Laten Semantic Analysis (LSA) Pada Peringkas Dokumen Otomatis Untuk Proses Clustering Dokumen, ISBN: 979-260266-6.

[3] Bunafit Nugroho, (2013), Dasar Pemrograman Web PHP-MySQL dengan Dreamweaver, Cetakan I, Yogyakarta : GAVA MEDIA.

[4] Fatkhul Amin (2011) Implementasi Search Engine (Mesin Pencari) Menggunakan Metode Vector Space Model, vol.V, No.1.

[5] G. Susanto, H. L. Purwanto, "Information Retival Menggunakan Latent Semantic Indexing", SMANTIKA Jurnal, Vol 8, no. 2, Okt 2018. pp. 74 - 79

[6] Kusrini, (2007), Konsep dan Aplikasi Sistem Pendukung Keputusan, Yogyakarta : C.V Andi.

[7] Indra Warman, M.Kom, Keni Novandri Saputra (2012) Sistem Informasi Alumni ITP Menggunakan PHP Dan My SQL, vol.12, No.1.

[8] Landauer, T. K., Foltz, P. W., & Laham, D. (1998) An Introduction to Latent Semantic Analysis.

[9] Muhammad, S. N Endah, and B. Noranita, "Sistem Temu Kembali Informasi dalam Dokumen Menggunakan Metode Latent Semantic Indexing" Jurnal

Masyarakat Informatika, Vol 3, No. 5, 2012

[10] Muhammad Jamhari, Edi Noersasongko, dan Hendro Subagyo (2014) Pengklusteran Dokumen Teks Hasil Peringkas Dokumen Otomatis Yang Menggunakan Metode Seleksi Fitur Dan Latent Semantic Analysis (LSA), vol.10, No.1.

[11] Noor Sahib Maricar, (2005), Oracle SQL Simplified, Cetakan I, Jakarta : Ekuator Digital Publishing.

[12] Rachmat Hidayat (2014) Sistem Informasi Ekspedisi Barang Dengan Metode E-CRM Untuk Meningkatkan Pelayanan Pelanggan, vol.4, No.2.

[13] Syaifudin Ramadhani, Urifatun Anis, dan Siti Tazkiyatul Masruro (2013) Rancang Bangun Sistem Informasi Geografis Layanan Kesehatan Di Kecamatan Lamongan Dengan PHP MySQL, vol.5, No.2.