IMPLEMENTASI SKEMA PEMBOBOTAN PADA APLIKASI PENILAIAN ESAI OTOMATIS METODE *LATENT SEMANTIC ANALYSIS*

SKRIPSI

OLEH

DIEGO OCTARIA
04 04 03 0296

DEPARTEMEN TEKNIK ELEKTRO
FAKULTAS TEKNIK UNIVERSITAS INDONESIA
DEPOK 2008

IMPLEMENTASI SKEMA PEMBOBOTAN PADA APLIKASI PENILAIAN ESAI OTOMATIS METODE *LATENT SEMANTIC ANALYSIS*

SKRIPSI

OLEH

DIEGO OCTARIA
04 04 03 0296

SKRIPSI INI DIAJUKAN UNTUK MELENGKAPI SEBAGIAN PERSYARATAN MENJADI SARJANA TEKNIK

DEPARTEMEN TEKNIK ELEKTRO
FAKULTAS TEKNIK UNIVERSITAS INDONESIA
DEPOK 2008

PERNYATAAN KEASLIAN SKRIPSI

Saya menyatakan dengan sesungguhnya bahwa skripsi dengan judul **:**

IMPLEMENTASI SKEMA PEMBOBOTAN PADA APLIKASI PENILAIAN ESAI OTOMATIS METODE *LATENT SEMANTIC ANALYSIS*

yang dibuat untuk melengkapi sebagian persyaratan menjadi Sarjana Teknik pada program studi Teknik Elektro, Departemen Elektro Fakultas Teknik Universitas Indonesia, sejauh yang saya ketahui bukan merupakan tiruan atau duplikasi dari skripsi yang sudah dipublikasikan dan atau pernah dipakai untuk mendapatkan gelar kesarjanaan di lingkungan Universitas Indonesia maupun di Perguruan Tinggi atau instansi manapun, kecuali bagian yang sumber informasinya dicantumkan sebagaimana mestinya.

Jakarta,20 Juni 2008

Diego Octaria

NPM 04 04 03 0296

PERSETUJUAN

Skripsi dengan judul:

IMPLEMENTASI SKEMA PEMBOBOTAN PADA APLIKASI PENILAIAN ESAI OTOMATIS METODE *LATENT SEMANTIC ANALYSIS*

Dibuat untuk melengkapi sebagian persyaratan menjadi Sarjana Teknik pada program studi Teknik Elektro Departemen Teknik Elektro Fakultas Teknik Universitas Indonesia dan disetujui untuk diajukan pada sidang skripsi.

Jakarta, 24 Juni 2008

Dosen Pembimbing

Dr. Ir. Anak Agung Putri Ratna, M.Eng

NIP. 131 865 234

UCAPAN TERIMA KASIH

Puji syukur hanya kepada ALLAH SWT, Yang Maha Berkuasa, shalawat dan salam terlimpah kepada Muhammad SAW. Alhamdulillah, tugas skripsi ini dapat diselesaikan dengan baik. Penulis mengucapkan terima kasih kepada :

Dr. Ir. Anak Agung Putri Ratna, M.Eng

selaku dosen pembimbing yang telah bersedia meluangkan waktu untuk memberi pengarahan, diskusi dan bimbingan serta persetujuan sehingga skripsi ini dapat selesai dengan baik.

Diego Octaria
NPM 04 04 03 0296
Departemen Teknik Elektro

Dosen Pembimbing
Ir. Anak Agung Putri Ratna, M.

IMPLEMENTASI SKEMA PEMBOBOTAN PADA APLIKASI PENILAIAN ESAI OTOMATIS METODE *LATENT SEMANTIC ANALYSIS*

ABSTRAK

Setiap proses pembelajaran memerlukan suatu evaluasi berupa ujian, begitu pula dengan e-learning. Pada proses e-learning jenis ujian yang banyak digunakan adalah jenis ujian pilihan ganda dan isian singkat. Alasannya adalah kemudahan dalam proses penilaian, komputer yang menjadi komponen penting dalam proses e-learning lebih mudah dalam melakukan penilaian ujian pilihan ganda dan isian singkat secara akurat karena jawaban yang ada harus sama baik pilihan maupun kata-katanya, dibandingkan dengan melakukan penilaian jenis ujian esai yang lebih kearah pemahaman bukan hafalan. Padahal jenis ujian pilihan ganda dan isian singkat memiliki banyak kekurangan bila dibandingkan dengan jenis ujian esai. Hal inilah yang mendasari lahirnya penilaian jawaban esai secara otomatis untuk mempersingkat pemeriksaan jawaban esai.

Ada banyak metode yang telah dikembangkan untuk penilai jawaban esai secara otomatis, salah satunya adalah *Latent Semantic Analysis* (LSA). Metode ini mempunyai ciri khas hanya mementingkan kata-kata kunci yang terkandung dalam sebuah kalimat tanpa memperhatikan karakteristik linguistiknya. Pada LSA, kata-kata direpresentasikan dalam sebuah matriks semantik dan kemudian diolah secara matematis menggunakan teknik aljabar linier *Singular Value Decomposition* (SVD).

Implementasi pembobotan pada sistem penilaian esay otomatis dilakukan dengan menggunakan bahasa php, pada percobaan menggunakan jawaban esay dari *quiz* jaringan komputer. Hasil ujicoba menunjukkan hal-hal yang mempengaruhi kecepatan proses aplikasi adalah banyaknya jawaban mahasiswa dan banyaknya user yang mengakses aplikasi. Dari percobaan juga menunjukkan bahwa skema yang paling mendekati dengan *human rater* adalah skema 4 yaitu dengan pembobotan lokal jawaban mahasiswa untuk Square Root dan pembobotan dosen Binary dan tidak menggunakan pembobotan global.

Kata kunci : E-Learning, LSA,Penilaian Esai Otomatis, Pembobotan.

Diego Octaria
NPM 04 04 03 0296
Electrical Engineering Departement

Counsellor
Ir. Anak Agung Putri Ratna, M.

IMPLEMENTATION WEIGHTING SCHEME IN AUTOMATED ESSAY GRADING LATENT SEMANTIC ANALYSIS

ABSTRACT

Every learning process needs an evaluation in the form of test. At e-learning process the test type many used is multiple choice and short answer test type. Its reason is amenity in course of assessment, the computer become the important component in course of e-learning easier in doing assessment of multiple choice and short anwer test in accurate because the answer have to be same exactly, compared to do assessment test of essay type more toward understanding and not memorizing. Though multiple choice and short answer test type have many insuffiencies if compared to the test type esai. These matters constitute the creation of automatically assessment of answer esai to take a short cut inspection of essay answer.

There are many methods which have been developed for the automatically essay assessor, one of them is Latent Semantic Analysis (LSA). This Method has the unique method only making account of the key words implied in a sentence regardless of his linguistics characteristic. In LSA, words represented in a semantic matrix and then mathematicaly proceed to usely linear algebra technique Singular Value Decomposition (SVD).

Wight implementation at automatically esay assessment system is done by using language php, In experiment the esay answer are from quiz computer network. Result of experiment show the things influence speed of application process is the number of student answers and to the number of user accessing application. Of attempt is also indicate that the scheme very come near with human rater is scheme of 4 that is with local wight [of] student answer to Square Root and lecturer wight Binary and don't use any global wight.

Keywords : E-Learning, LSA,Penilaian Esai Otomatis, Pembobotan.

DAFTAR ISI

Halaman

DAFTAR GAMBAR

Halaman

DAFTAR TABEL

DAFTAR SINGKATAN

APACHE	A Patchy Server
FTP	File Transfer Protocol
GUI	Graphical User Interface
HTML	Hyper Text Markup Languange
HTTP	Hypertext Tranfer Protocol
LAN	Local Area Network
LSA	Latent Semantic Analysis
MySQL	My Structure Query Languange
PC	Personal Computer
PEG	Project Essay Grade
PERL	Practical Extraction and Report Languange
PHP	Hypertext Preprocessor
SQL	Structure Query Languange
SVD	Singular Value Decomposition
UML	Unified Modeling Languange
URL	Uniform Resource Locator
XML	eXtended Markup Languange
WWW	World Wide Web

BAB I
PENDAHULUAN

1.1. LATAR BELAKANG

Berkembangnya teknologi internet membuat berbagai macam dampak dalam berbagai bidang, salah satunya adalah bidang pendidikan. Saat ini metode pembelajaran *e-learning* sudah banyak dikembangkan didunia. Aplikasi yang banyak dikembangkan dalam *e-learning* antara lain konfrensi video, forum diskusi serta ujian secara online.

Sistem ujian online dapat berupa berbagai macam seperti ujian pilihan ganda, jawaban singkat ataupun ujian esai . Pada ujian esai dibutuhkan pemahaman konsep dan analisis yang tepat untuk menjawab pertanyaannya, karena itu ujian esai dianggap ujian yang paling baik untuk melihat tingkat pemahaman seseorang tehadap materi yang diujikan.

Untuk memeriksa ujian esai secara manual diperlukan waktu yang cukup lama Hal ini menjadi masalah jika waktu untuk memeriksanya terbatas sehingga pemeriksa akan memaksakan diri dalam memberi penilaian, selain itu jika diperiksa oleh banyak orang maka akan membuat adanya subjektivitas dalam penilaian ujian yang semuanya akan berakibat pada nilai –nilai peserta ujian. Untuk mengatasi masalah ini maka diperlukan suatu system penilaian esai otomatis karena yang memeriksa adalah computer akan lebih objektif dan konsisten dalam memberikan penilaian. Selain itu juga dengan penilaian esai otomatis maka pemeriksaan ujian dapat menjadi lebih cepat.

Ada banyak metode yang dikembangkan untuk penilaian esai otomatis. Masing-masing metode menggunakan pendekatan yang berbeda dan memiliki keunggulan dan kelemahan yang berbeda-beda juga. Salah satu metode penilaian esai otomatis adalah metode Latent Semantic Analysis (LSA). Dalam pemeriksaan ujian esai, metode LSA mementingkan kata-kata yang terkandung di dalam jawabannya tanpa memperhatikan karakteristik linguistiknya. Dalam metode ini kata-kata akan direpresentasikan dengan matriks dan kemudian akan

dilakukan perhitungan matematis. Metode ini cukup sederhana tetapi memiliki korelasi yang tinggi dengan jawaban sebenarnya. LSA merepresentasikan isi kata dalam matriks dua dimensi yang besar. Bagian pemrosesan penting dari LSA adalah komponen penganalisis bernama SVD (*Singular Value Decomposition*) yang mengkompresi informasi yang berkaitan dalam jumlah besar ke dalam ruang yang lebih kecil tetapi mewakili arti sebenarnya.

Ada beberapa parameter yang mempengaruhi penilaian suatu penilaian esai otomatis. Diantaranya adalah pembersihan kata-kata jawaban seperti tanda baca, huruf besar, dan lain-lain, *weighting* (pembobotan), *singular value dimensionality* dan *type of similarity measure* [1]. Parameter-parameter itu tentu saja mempengaruhi peningkatan performansi penilaian esai. Pada skripsi ini akan dibahas penerapan teknik pembobotan pada sistem penilaian esai metode *Latent Semantic Analysis* dengan teknik *Singular Value Decomposition*.

1.2. TUJUAN

Tujuan dari penulisan skripsi adalah merancang dan melakukan implementasi teknik pembobotan pada aplikasi sistem penilaian esai otomatis berbahasa Indonesia.

1.3. PEMBATASAN MASALAH

Permasalahan difokuskan dan dibatasi pada implementasi teknik pembobotan pada sistem penilaian esai otomatis dengan bahasa pemrograman PHP. Pembahasan mengenai aplikasi meliputi algoritma, prinsip kerja, analisis kecepatan sistem, serta perbandingan tiga skema pembobotan dan tanpa pembobotan dengan penilaian manual.

1.4. SISTEMATIKA PENULISAN

Secara keseluruhan, skripsi ini terdiri dari lima bab. Bab pertama adalah pendahuluan yang membahas mengenai latar belakang, tujuan penulisan, batasan

masalah dan sistematika penulisan untuk memberikan gambaran umum mengenai penulisan skripsi ini.

Bab kedua menjelaskan landasan teori. Diawali penjelasan mengenai *e-learning.* beberapa metode essay grading, LSA, selain itu dijelaskan pula program pendukung serta teori pembobotan.

Bab ketiga bab ini berisikan langkah-langkah konsep aplikasi yang akan dibuat. Pada bab ini juga akan diperlihatkan kapan pembobotan digunakan serta skema pembobotan yang dilakukan.

Bab empat merupakan analisis dari implementasi dan hasil kerja aplikasi sistem penilaian esai otomatis berbahasa Indonesia. Analisis meliputi kecepatan aplikasi dan tingkat akurasi aplikasi terhadap penilaian manual serta bagaimana pengaruhnya pada berbagai metode pembobotan yang diuji.

Bab terakhir, yakni bab kelima, merupakan penutup berupa kesimpulan dari hasil dan analisis bab sebelumnya.

BAB II
SISTEM PENILAIAN ESAI OTOMATIS DENGAN METODE LSA

2.1. *E-LEARNING*

Saat ini perkembangan e-lerning sudah sangat banyak. Selain itu penerapannya di bidang pendidikan juga sudah sangat banyak di seluruh dunia. Pada bab ini pembahasan mengenai pembahasan *e-learning* hanya mengenai pengertian, manfaat, dan aplikasinya.

2.1.1. Pengertian *E-Learning*

E-learning (Electronic Learning) merupakan konsep pendidikan dengan menggunakan teknologi jaringan komputer seperti internet dan LAN sebagai metode untuk penyampaian aktifitas belajar mengajarnya. *E-learning* menggunakan alat-alat elektronik seperti komputer ataupun handphone untuk mengakses internet karena *e-learning* berbasis web. *E-learning* juga merupakan pembelajaran jarak jauh yang menggunakan computer dan umumnya menggunakan internet. Hal ini membuat para pelajarnya dapat belajar dengan menggunakan komputer ditempatnya masing-masing tanpa harus pergi ke tempat belajarnya.

2.1.2. Manfaat E-Learning

Dengan adanya *e-learning* akan membuat proses belajar mengajar menjadi lebih mudah. Ada beberapan manfaat dari kehadiran *e-learning* diantaranya adalah sebagai berikut

a. Fleksibilitas tempat dan waktu belajar

   Dengan adanya *e-learning* maka proses belajar mengajar dapat dilakukan dari mana dan kapan saja. Karena sumber bahan pelajarannya dapat diakses oleh pelajarnya melalui internet, maka para pesertanya dapat mengakses bahan pembelajarannya kapan dan dimana saja. Selain itu

untuk tugas-tugas dan ujian yang dilakukan tidak perlu menunggu untuk para pengajar dan muridnya karena dapat dilakukan untuk periode tertentu.

b. Interaksi antara pengajar dan muridnya yang lebih baik.

   Jika proses *e-learning* dapat berjalan sesuai maka interaksi antara pengajar dan murid dapat lebih baik karena jika murid mempunyai pertanyaan dapat disampaikan langsung kepada pengajarnya tanpa batasan waktu. Hal ini berbeda dengan system pembelajaran yang konvensional dimana para muridnya hanya bisa mengajukan pertanyaan jika ada tatap muka yang biasanya dilakukan di kelas yang waktunya sempit. Selain itu untuk beberapa peserta yang malu untuk bertanya atau menyampaikan pendapatnya dapat lebih bebas untuk menyampaikannya.

c. Peserta yang banyak dan beragam

   E-learning yang membuat proses belajar mengajar dapat dilakukan kapan dan dimana saja membuat dapat jangkauan pesertanya lebih luas. Selain itu pesertanya juga dapat lebih banyak karena system ujian dan penyampaian materinya dapat dilakukan secara otomatis.

2.1.3. Aplikasi E-Learning

Dalam penerapannya, *e-learning* mempunyai berbagai macam bentuk aplikasi yang ada untuk mendukung proses belajar mengajar seperti: pendistribusian bahan ajar, konferensi video, forum tanya jawab, dan ujian online. Saat ini beberapa aplikasi yang paling sering digunakan dalam *e-learning* adalah pendistribusian bahan ajar, forum diskusi, forum tanya jawab serta ujian online tetapi biasanya ujian pilihan ganda atau jawaban singkat, tetapi karena system ujian pilihan ganda maupun jawaban singkat memiliki beberapa kelemahan maka saat ini banyak dikembangkan ujian online metode esai dengan penilaiannya secara otomatis.

2.2. METODE-METODE PENILAIAN JAWABAN ESAI OTOMATIS

2.2.1. PEG (*Project Essay Grader*)

Sistem ini merupakan sistem penilaian esai yang pertama, dikenalkan oleh Ellis Page pada tahun 1966 [1]. Sistem ini dikembangkan karena beratnya beban pengajar untuk melakukan pemeriksaan esai secara manual dalam waktu yang singkat. Pemeriksaan yang dilakukan tim penilai beranggota beberapa orang mungkin mengalami regresi dalam konsistensi penilaian.

Cara kerja metode ini, sekumpulan tugas yang telah dinilai diambil sebagai sampel dan dibandingkan untuk mencari karakteristik tertentu. Kemudian elemen-elemen ini disarikan secara otomatis dari jawaban, juga diterapkan regresi linear ganda untuk menentukan kombinasi optimal dari pembobotan masing-masing karakter. Ini dilakukan untuk memprediksi nilai yang akan diberikan. Sistem kemudian melakukan penilaian pada jawaban murid menggunakan pombobotan yang sama karakteristiknya.

2.2.2. *E-RATER (Electronic Essay Rater)*

Sistem ini dikembangkan oleh ETS (*Educational Testing Service*) yang telah memulai riset di bidang pengujian tulisan sejak tahun 1947. Beberapa riset terdahulu dalam pengujian tulisan meletakkan dasar bagi penilaian "holistik" yang terus digunakan ETS untuk pengujian berskala besar. Beberapa program pengujian berskala besar yang mengandung pengukuran dari tulisan adalah GMAT (*Graduate Management Admissions Test*), TOEFL (*Test of English as a Foreign Language*), GRE (*Graduate Record Examination*), dan banyak lagi [1].

Pada Februari 1999 ETS mulai menggunakan *e-rater* untuk melakukan penilaian terhadap GMAT AWA (*Analytical Writing Assessment*) – Ujian Analisis Penulisan. Penilaian dilakukan oleh dua orang dalam skala holistik enam poin. Penilaian orang ketiga dibutuhkan bila terjadi perbedaan penilaian lebih dari satu poin. Sejak penggunaan *e-rater* hanya dibutuhkan satu orang penilai pembanding dengan hasil sistem. Bila terdapat perbedaan lebih dari satu poin,

satu orang penilai lagi akan digunakan, namun bila tidak kedua penilaian ini digunakan untuk menghitung nilai akhir dari esai.[4]

2.3. *LATENT SEMANTIC ANALYSIS* (LSA)

Latent Semantic Analysis (LSA) adalah suatu cara untuk mengekstrak sebuah tulisan dan kemudian merepresentasikan tulisan tersebut dengan perhitungan matematis. Penilaian dengan metode LSA lebih kepada kata-kata yang ada dalam tulisan tanpa memperhatikan urutan kata dan tata bahasa dalam tulisan tersebut, sehingga suatu kalimat yang dinilai adalah berdasarkan kata-kata kunci yang ada pada kalimat tersebut.[5]

Untuk menilai sebuah jawaban esai menggunakan metode LSA, caranya adalah pemeriksa harus terlebih dahulu membuat suatu esai yang menjadi acuan jawaban dan kemudian esai yang akan dinilai dibandingkan dengan esai acuan jawaban, semakin banayak kemiripannya maka semakin besar nilai jawabannya.

Langkah-langkah pemeriksaaan esai dengan metode LSA adalah pertama-tama tulisan direpresentasikan ke dalam matriks dimana baris matriks menunjukkan kata-kata kunci pada tulisan tersebut dan setiap kolom menunjukkan suatu kalimat. Setiap sel menunjukkan banyaknya peminculan kata kunci yang berada pada matriks pada kalimat yang ada di kolom matriks. Kemudian isi dari sel tersebut terlebih dahulu ditransformasikan dimana setiap frekuensi kata dibobotkan dengan sebuah fungsi yang menunjukkan pentingnya sebuah kalimat dalam paragraph dan juga derajat yang menunjukkan seberapa pentingnya tipe kata didalam suatu kalimat.

Selanjutnya adalah mengaplikasikan Singular Value Decomposition (SVD), yaitu suatu bentuk analisa faktor pada matriks. Pada SVD matriks memuat frekuensi pemunculan kata kunci di dekomposisi menjadi tiga buah matriks yang jika tiga buah matriks tersebut dikalikan maka akan muncul kembali matriks asalnya. Matriks pertamanya mendeskripsikan entitas kolom sebagai nilai vektor orthogonal matriks. Matriks keduanya berupa matriks diagonal yang memuat nilai scalar matriks. Secara matematis, faktor yang paling baik adalah menggunakan

dimensi terkecil dari matriks awalnya, sehingga rekonstruksi matriks terbaik dihasilkan pada saat nilai faktor lebih kecil dari jumlah faktor yang digunakan.

Dimensi dari matriks yang telah disederhanakan dengan menghapus koefisien pada matriks diagonal sebanyak yang diinginkan sampai tersisa koefisien sebanyak dimensi yang terpilih. Tujuan penyederhanaan ini adalah agar terbentuk matriks yang memuat nilai korelasi yang diinginkan ketika tiga buah matriks direkonstruksi. Kemudian penilaian akan dilakukan dengan membandingkan matriks korelasi dari jawaban penguji dengan matriks korelasi dari jawaban peserta ujian dengan menggunakan perhitungan cosinus α

Untuk lebih memahami proses ini lebih jelas, berikut ini adalah contoh yang biasa digunakan. Pada gambar 2.1 dapat dilihat beberapa kalimat yang terdiri dari dua tema.

Technical Memo Example

Titles:

c1: *Human* machine *interface* for Lab ABC *computer* applications

c2: A *survey* of *user* opinion of *computer system response time*

c3: The *EPS user interface* management *system*

c4: *System* and *human system* engineering testing of *EPS*

c5: Relation of *user*-perceived *response time* to error measurement

m1: The generation of random, binary, unordered *trees*

m2: The intersection *graph* of paths in *trees*

m3: *Graph minors* IV: Widths of *trees* and well-quasi-ordering

m4: *Graph minors*: A *survey*

Gambar 2.1 Contoh Kalimat dan Kata Kunci[2]

Dari contoh pada gambar 2.1 ada sembilan kalimat dengan lima kalimat pertama bertema *human computer interaction* (kalimat c1 sampi c5), dan empat kalimat selanjutnya bertema mathematical *graph theory* (kalimat m1 sampai m4). Syarat sebuah kata menjadi kata kunci pada paragraf ini adalah setidaknya sebuah kata

harus muncul minimal 2 kali. Pentransformasian kata-kata diatas menjadi sebuah matriks A ditunjukkan pada tabel 2.1.

Tabel 2-1. Transformasi Matriks Kemunculan Kata Kunci Pada Susunan Kalimat dalam Gambar 2.1

	C1	C2	C3	C4	C5	M1	M2	M3	M4
Human	1	0	0	1	0	0	0	0	0
interface	1	0	1	0	0	0	0	0	0
computer	1	1	0	0	0	0	0	0	0
User	0	1	1	0	1	0	0	0	0
System	0	1	1	2	0	0	0	0	0
response	0	1	0	0	1	0	0	0	0
Time	0	1	0	0	1	0	0	0	0
EPS	0	0	1	1	0	0	0	0	0
Survey	0	1	0	0	0	0	0	0	0
Trees	0	0	0	0	0	1	1	1	0
Graph	0	0	0	0	0	0	1	1	1
Minors	0	0	0	0	0	0	0	1	1

Sumber :[2]

Pendekomposisian matriks pada tabel 2.1 menjadi bentuk SVD dilkukan sebagai berikut . Matriks pertama sebagai nilai vector orthogonal matriks didapat dengan mencari matriks A^TA dan kemudian dicari nilai *eigen* dari matriks A^TA. Nilai eigennya kemudian di normalisasi untuk membentuk matriks yang disebut sebagai matriks U. Matriks U dari hasil perhitungan untuk contoh pada gambar 2.1 ditunjukkan pada tabel 2-2.

Tabel 2-2. Matriks U sebagai komponen matriks A

0.22	-0.11	0.29	-0.41	-0.11	-0.34	0.52	-0.06	-0.41
0.2	-0.07	0.14	-0.55	0.28	0.5	-0.07	-0.01	-0.11
0.24	0.04	-0.16	-0.59	-0.11	-0.25	-0.3	0.06	0.49
0.4	0.06	-0.34	0.1	0.33	0.38	0	0	0.01
0.64	-0.17	0.36	0.33	-0.16	-0.21	-0.17	0.03	0.27
0.27	0.11	-0.43	0.07	0.08	-0.17	0.28	-0.02	-0.05
0.27	0.11	-0.43	0.07	0.08	-0.17	0.28	-0.02	-0.05
0.3	-0.14	0.33	0.19	0.11	0.27	0.03	-0.02	-0.17
0.21	0.27	-0.18	-0.03	-0.54	0.08	-0.47	-0.04	-0.58
0.01	0.49	0.23	0.03	0.59	-0.39	-0.29	0.25	-0.23
0.04	0.62	0.22	0	-0.07	0.11	0.16	-0.68	0.23
0.03	0.45	0.14	-0.01	-0.3	0.28	0.34	0.68	0.18

Sumber :[2]

Sedangkan matriks kedua berupa matriks diagonal yang memuat nilai scalar matriks didapat dari akar kuadrat dari nilai eigen matriks A^TA dan disebut sebagai matriks S. Matriks S dari hasil perhitungan untuk contoh pada gambar 2.1 ditunjukkan pada tabel 2-3. Untuk matriks terakhir cara mencarinya sama dengan matriks pertama, yang menjadi perbedaan adalah yang dicari matriks AA^T bukan A^TA, dan matriks ini disebut sebagai matriks V Matriks V dari hasil perhitungan untuk contoh pada gambar 2.1 ditunjukkan pada tabel 2-4

Tabel 2-3. Matriks S sebagai komponen dari matriks A

3.34	0	0	0	0	0	0	0	0
0	**2.54**	0	0	0	0	0	0	0
0	0	2.35	0	0	0	0	0	0
0	0	0	1.64	0	0	0	0	0
0	0	0	0	1.5	0	0	0	0
0	0	0	0	0	1.31	0	0	0
0	0	0	0	0	0	0.85	0	0
0	0	0	0	0	0	0	0.56	0
0	0	0	0	0	0	0	0	0.36

Sumber :[2]

Tabel 2-4. Matriks V sebagai komponen vektor orthogonal transpose matriks A

0.2	-0.06	0.11	-0.95	0.05	-0.08	0.18	-0.01	-0.06
0.61	0.17	-0.5	-0.03	-0.21	-0.26	-0.43	0.05	0.24
0.46	-0.13	0.21	0.04	0.38	0.72	-0.24	0.01	0.02
0.54	-0.23	0.57	0.29	-0.21	-0.37	0.26	-0.02	-0.08
0.28	0.11	-0.51	0.15	0.33	0.03	0.67	-0.06	-0.26
0	0.19	0.1	0.02	0.39	-0.3	-0.34	0.45	-0.62
0.01	0.44	0.19	0.01	0.35	-0.21	-0.15	-0.76	0.02
0.02	0.62	0.25	0.01	0.15	0	0.25	0.48	0.52
0.08	0.53	0.08	-0.03	-0.6	0.36	0.04	-0.07	-0.45

Sumber :[2]

Penyederhanaan matriks dipilih sebesar *k* dimensi dalam contoh ini k = 2, sehingga matriks komponen S hanya menyisakan nilai seperti yang di tebalkan pada Tabel 2.3, kemudian komponen U,S yang disederhanakan , dan V dikalikan

kembali untuk mendapat matriks A_k yang merupakan matriks rekonstruksi A Hasilnya adalah seperti tabel yang ditunjukkan oleh tabel 2-5.

Tabel 2-5. Matriks A_k sebagai matriks rekonstruksi A

0.16	0.4	0.38	0.47	0.18	-0.05	-0.12	-0.16	-0.09
0.14	0.37	0.33	0.4	0.16	-0.03	-0.07	-0.1	-0.04
0.15	0.51	0.36	0.41	0.24	0.02	0.06	0.09	0.12
0.26	0.84	0.61	0.7	0.39	0.03	0.08	0.12	0.19
0.45	1.23	1.05	1.27	0.56	-0.07	-0.15	-0.21	-0.05
0.16	0.58	0.38	0.42	0.28	0.06	0.13	0.19	0.22
0.16	0.58	0.38	0.42	0.28	0.06	0.13	0.19	0.22
0.22	0.55	0.51	0.63	0.24	-0.07	0.14	-0.2	-0.11
0.1	0.53	0.23	0.21	0.27	0.14	0.31	0.44	0.42
-0.06	0.23	-0.14	-0.27	0.14	0.24	0.55	0.77	0.66
-0.06	0.34	-0.15	-0.3	0.2	0.31	0.69	0.98	0.85
-0.04	0.25	-0.1	-0.21	0.15	0.22	0.5	0.71	0.62

Sumber :[2]

Matriks A_k yang tidak sama dengan matriks *A* yang asli. Matriks A_k hanyalah sebuah pendekatan atau aproksimasi *A* pada faktor *k* . SVD yang dilakukan setelah penyederhanaan mengambil sebagian besar struktur penting yang terdapat pada hubungan kata kunci dan kalimat. Dan, pada saat yang sama juga menghilangkan variabilitas penggunaan kata yang menjadi gangguan utama. Selama nilai dari *k* jauh lebih kecil dari banyaknya kata kunci maka perbedaan minor dalam terminologi dapat diabaikan. Kata-kata kunci yang terdapat dalam kalimat yang sama, akan berdekatan satu sama lain dalam ruang *k* walaupun kata-kata kunci itu tidak pernah hadir bersamaan lagi pada kalimat yang sama. Hal ini berarti beberapa kalimat yang tidak memiliki satupun kata kunci yang sama, maka dia tidak akan mendekati saling berhubungan dalam ruang-*k*. Pengertian yang

berhubungan dengan asosiasi kata-kata kunci dalam kalimat ini kemudian digunakan untuk penilaian jawaban selanjutnya.

2.4. PENILAIAN ESAI MAHASISWA (DOKUMEN)

Query yang merupakan kata kunci dari jawaban dosen dapat direpresentasikan sebagai vektor dalam ruang-*k*. Vektor inilah yang kemudian di bandingkan dengan vektor-vektor jawaban esai mahasiswa (dokumen) untuk selanjutkan dinilai mana yang paling mendekati. Sebuah *query* seperti halnya dokumen, merupakan kumpulan dari kata-kata. *Query* pengguna dapat representasikan sebagai

$$\bar{q} = q^T U_k \Sigma_k^{-1}$$

. . . (2-1)

Seperti vektor *query*, vektor dokumen direpresentasikan sebagai

$$\bar{d} = d^T U_k \Sigma_k^{-1}$$

. . . (2-2)

Matriks *q* adalah matriks satu kolom yang elemennya berisi jumlah kata kunci dalam *query*. Sementara matriks *d* adalah matriks satu kolom. Elemennya berisi nilai kehadiran kata kunci dalam dokumen. Matriks *d* sama dengan kolom matriks *A*. $\bar{q}$ adalah vektor *query* dan $\bar{d}$ adalah vektor dokumen. Vektor *query* dapat dibandingkan atau dikorelasikan dengan semua vektor dokumen yang ada. Teknik korelasi yang umum digunakan adalah dengan mencari nilai kosinus sudut yang dibentuk antara vektor *query* dan vektor dokumen. Korelasi kosinus antara vektor *query* dan vektor dokumen diberikan oleh persamaan

$$\cos\alpha = \frac{\bar{q} \cdot \bar{d}}{\|\bar{q}\|\|\bar{d}\|}$$

. . . (2-3)

α adalah sudut diantara kedua vektor tersebut. Jadi, nilai korelasi adalah perhitungan sudut berdasarkan kosinus antara $\overline{q}$ dan $\overline{d}$. Jika dilakukan penilaian dari jawaban mahasiswa yang paling besar ke paling kecil nilainya, maka jawaban yang paling besar nilainya adalah yang memiliki sudut α dengan yang paling kecil.

2.5. PEMBOBOTAN

Teknik pembobotan yang tepat dapat meningkatkan performansi dari model vektor LSA. Sebuah metode pembobotan merupakan susunan dari tiga buah pembobotan: pembobotan lokal *(local weighting)*, pembobotan global *(global weighting)* dan normalisasi *(normalization)* [3]. Pembobotan dikenakan pada tiap elemen matriks A . Pembobotan dirumuskan melalui persamaan :

$$a_{ij} = L(i, j) \times G(i) \times N(j) \quad \textbf{... (2-4)}$$

Dengan *L(i,j)* merupakan bobot lokal untuk kata kunci *i* dalam dokumen *j. G(i)* adalah bobot global untuk kata kunci *i,* dan *N(j)* adalah faktor normalisasi dokumen *j.* Bobot lokal adalah fungsi dari berapa banyak setiap kata kunci muncul dalam suatu dokumen. Bobot global adalah fungsi dari berapa banyak setiap kunci muncul dalam semua dokumen atau koleksi. Faktor normalisasi digunakan untuk mengkompensasi perbedaan panjang dokumen-dokumen dalam koleksi.

Vektor dokumen (matriks kolom A) dan vektor *query* dikenakan pembobotan dengan metode yang berbeda. Bobot lokal dihitung berhubungan dengan kata kunci pada dokumen atau *query*. Bobot global lebih didasarkan pada sejumlah koleksi dokumen yang ada tanpa memperhatikan apakah itu pembobotan pada dokumen atau *query.* Normalisasi vektor *query* sebenarnya tidak perlu karena tidak mempengaruhi urutan relevansi akhir terhadap dokumen.

2.5.1. PEMBOBOTAN LOKAL

Pembobotan lokal merupakan pembobotan kata dalam satu dokumen dengan prinsip jika ada *query* yang kata-kata kuncinya jumlah di dokumen tersebut ada banyak maka dokumen tersebut akan semakin berhubungan dengan

query tersebut. Sejumlah pembobotan lokal yang umum digunakan diberikan dalam Tabel 2.2.

Tabel 2-6. Macam-macam pembobotan lokal

Formula	Nama Metode	Kependekan
$1 \quad \text{jika } f_{ij} > 0$ $0 \quad \text{jika } f_{ij} = 0$	Biner	BNRY
f_{ij}	Frekuensi intra-dokumen	FREQ
$1+\log f_{ij} \quad \text{jika } f_{ij} > 0$ $0 \quad \text{jika } f_{ij} = 0$	Log	LOGA
$\frac{1+\log f_{ij}}{1+\log a_j} \quad \text{jika } f_{ij} > 0$ $0 \quad \text{jika } f_{ij} = 0$	Normalisasi log	LOGN
$\sqrt{f_{ij} - 0,5} + 1 \quad \text{jika } f_{ij} > 0$ $0 \quad \text{jika } f_{ij} = 0$	Akar pangkat dua	SQRT

Sumber : [3]

Pembobotan lokal yang paling sederhana adalah pembobotan biner (BNRY) dan frekuensi intra-dokumen (FREQ). Dari Tabel 2.2, f_{ij} adalah frekuensi kemunculan kata kunci *i* dalam dokumen *j.* Pembobotan ini biasanya digunakan untuk pembobotan pada *query*, dimana kata kunci hanya muncul satu-dua kali saja. Untuk pembobotan dokumen, metode ini umumnya bukan yang terbaik. Hal ini karena BNRY tidak membedakan antara kata kunci yang muncul beberapa kali

dengan kata kunci yang muncul hanya sekali. Selain itu metode FREQ dinilai memberikan bobot terlalu besar untuk kata kunci yang muncul beberapa kali.

Metode logaritma digunakan untuk menyesuaikan frekuensi intra dokumen. Karena sebuah kata kunci yang muncul sepuluh kali dalam sebuah dokumen tidak berarti sepuluh kali lebih penting dibandingkan kata kunci yang muncul sekali dalam dokumen tersebut. Dua dari sejumlah metode pembobotan lokal dalam Tabel 2.2 bisa dikatakan mirip karena metode tersebut menggunakan logaritma. Dua metode itu adalah LOGA dan LOGN. Semua logaritma pada metode pembobotan berbasis 2. a_j adalah frekuensi rata-rata dari kemunculan kata kunci dalam dokumen *j*. Karena dalam LOGN terdapat normalisasi yakni $(1+\log a_j)$, maka hasil pembobotan yang diberikan oleh LOGN akan selalu lebih kecil nilainya dibandingkan LOGA untuk kata kunci dan dokumen yang sama.

Pembobotan lokal lainnya, yang menjadi penengah antara metode biner dan frekuensi intra dokumen dalah metode normalisasi frekuensi diperlebar *(augmented normalized term frequency)* atau ATF1 Pada Tabel 2.2 x_j merupakan frekuensi maksimum dari kata kunci dalam dokumen *j*. ATF1 memberikan bobot pada sebuah kata yang muncul pada dokumen dan memberikan tambahan bobot bila kata tersebut muncul beberapa kali. Dengan formula ini, *L(i,j)* bervariasi hanya antara 0,5 sampai 1 untuk kata yang muncul dalam dokumen.

2.5.2. PEMBOBOTAN GLOBAL

Pembobotan global ditujukan untuk memberikan nilai yang berbeda antara setiap kata kunci. Pembobotan global berdasarkan semakin kecil nilai frekuensi kemunculan kata dalam seluruh koleksi dokumen, maka makin berbedalah kata tersebut [3].

Tabel 2-7. Macam-macam pembobotan global

Formula	Nama Metode	Kependekan
$\log\left(\frac{N}{n_i}\right)$	Invers frekuensi dokumen	IDFB
$\log\left(\frac{N-n_i}{n_i}\right)$	Invers probabilistik	IDFP
$1+\sum_{j=1}^{N}\frac{\frac{f_{ij}}{F_i}\log\left(\frac{f_{ij}}{F_i}\right)}{\log N}$	Entropi	ENPY
$\frac{F_i}{n_i}$	Frekuensi global IDF	IGFF
$\sqrt{\frac{F_i}{n_i}-0,9}$	Akar pangkat dua global IDF	IGFS
1	Tidak ada bobot global	NONE

Sumber: [3]

Sebuah pembobotan global yang umum digunakan adalah *inverted document frequency* atau IDF. Dalam Tabel 2.3 diberikan dua variasi yakni IDFB dan IDFP. *N* adalah jumlah dokumen dalam koleksi dan n_i merupakan jumlah dokumen dimana kata kunci *i* muncul didalamnya. IDFB adalah logaritma dari invers dari probabilitas kata kunci *i* muncul dalam dokumen acak. IDFP adalah logaritma dari invers dari probabilitas ketidak-hadiran kata kunci *i* dalam dokumen acak. IDFB dan IDFP adalah sama dalam artian keduanya memberikan bobot yang lebih besar untuk kata yang tampil pada beberapa dokumen saja dan memberikan bobot yang lebih kecil untuk kata yang muncul pada banyak dokumen dalam koleksi. IDFP memberikan bobot negatif untuk kata yang muncul pada lebih dari separuh jumlah seluruh dokumen. Sementara pada IDFB, nilai terendah pembobotan adalah 1.

Pada metode entropi (ENPY), F_i merupakan frekuensi kemunculan kata kunci *i* di seluruh koleksi dokumen. Jika sebuah kata kunci muncul sekali pada setiap dokumen, maka kata tersebut diberikan bobot bernilai nol. Jika sebuah kata kunci muncul sekali pada satu dokumen, maka kata tersebut diberi bobot satu. Kombinasi dan variasi lain dari frekuensi kemunculan akan menghasilkan bobot yang nilainya antara nol dan satu. Entropi adalah teknik pembobotan yang sangat berguna karena ia memberikan bobot yang lebih besar untuk kata yang frekuensi kemunculannya kecil pada sejumlah kecil dokumen.

Dalam Tabel 2.3 juga disebutkan pembobotan frekuensi global IDF (IGFF). Jika sebuah kata kunci muncul sekali pada setiap dokumen atau sekali pada satu dokumen, maka kata tersebut diberikan bobot sebesar satu, yang merupakan bobot terkecil. Sebuah kata yang muncul beberapa kali pada sejumlah dokumen akan mendapat bobot yang besar. Pembobotan ini bekerja dengan baik jika dikombinasikan dengan pembobotan global yang berbeda pada vektor *query*.

2.5.3. NORMALISASI

Bagian ketiga dari sebuah pembobotan adalah faktor normalisasi atau *N(j)*, yang mana digunakan untuk mengkompensasi perbedaan panjang dokumen-dokumen dalam koleksi. Bagian ini berguna untuk menormalkan vektor dokumen sehingga dokumen-dokumen tersebut independen terhadap panjangnya. Dalam Tabel 2.4 diperlihatkan dua buah metode normalisasi.

Tabel 2-8. Macam-macam Normalisasi

Formula	Nama Metode	Kependekan
$\frac{1}{\sqrt{\sum_{i=0}^{m}\left(G_i L_{ij}\right)^2}}$	Normalisasi kosinus	COSN
$\frac{1}{(1-slope)\,pivot + slope \times l_j}$	Normalisasi pivot	PUQN
1	Tidak ada normalisasi	NONE

Sumber : [3]

Normalisasi yang paling umum digunakan dalam model ruang vektor adalah normalisasi kosinus (COSN). Normalisasi ini memiliki faktor pembagi magnitude dari dokumen yang dibobotkan, sehingga hal ini menyebabkan magnitude dari vektor dokumen selalu bernilai satu. Dalam metode COSN, dokumen yang lebih panjang diberikan bobot lebih kecil untuk kata kunci, sehingga dokumen yang lebih pendek akan lebih panjang dalam proses perolehan informasi.

Metode *pivoted unique normalization* (PUQN) mencoba untuk mengatasi masalah dalam penanganan dokumen-dokumen yang pendek. Dalam Tabel 2.4 l_j adalah banyaknya kata kunci yang berbeda dalam dokumen *j*. Nilai *slope* dapat diset sebesar 0,2 dan *pivot* adalah rata-rata banyaknya kata kunci yang berbeda per dokumen dalam seluruh koleksi. Prinsip dasar dari normalisasi pivot adalah untuk mengatasi perbedaan panjang dokumen diantara dokumen yang memiliki probabilitas relevan dan dokumen yang memiliki probabilitas akan diperoleh atau di-*retrieve*. Dengan faktor normalisasi ini, kurva relevansi dan kurva *retrieval* digambarkan berdasarkan panjang dokumen. Titik dimana kedua kurva ini bersinggungan atau memotong disebut pivot. Dokumen pada sebelah kiri dari pivot umumnya memiliki probabilitas yang lebih besar untuk di-*retrieve* daripada tingkat relevansinya. Dan, dokumen yang ada di sebelah kanan pivot memiliki probabilitas lebih relevan daripada untuk diterima. Melalui penggeseran pivot ini, faktor normalisasi dapat diubah-ubah sedemikian rupa untuk mendapatkan hasil kombinasi yang lebih baik antara probabilitas relevansi dan *retrieval*.

2.6. PROGRAM-PROGRAM PENDUKUNG

2.6.1. PHP

PHP merupakan singkatan dari Hypertext Preprocessor, yang merupakan sebuah bahasa *scripting* yang dipasang dan menyatu pada halaman HTML (Hypertext Markup Language). PHP dibuat oleh Rasmus Lerdorf dan bersifat *open source* yang ditulis menggunakan sintaks bahasa C, Java, dan Perl.

PHP hampir dapat berjalan di semua sistem operasi seperti Windows, Unix, Linux dan variannya, Mac OS X, RISC OS dan sistem operasi lainnya. PHP juga kompatibel dengan *web server* yang banyak digunakan sekarang seperti

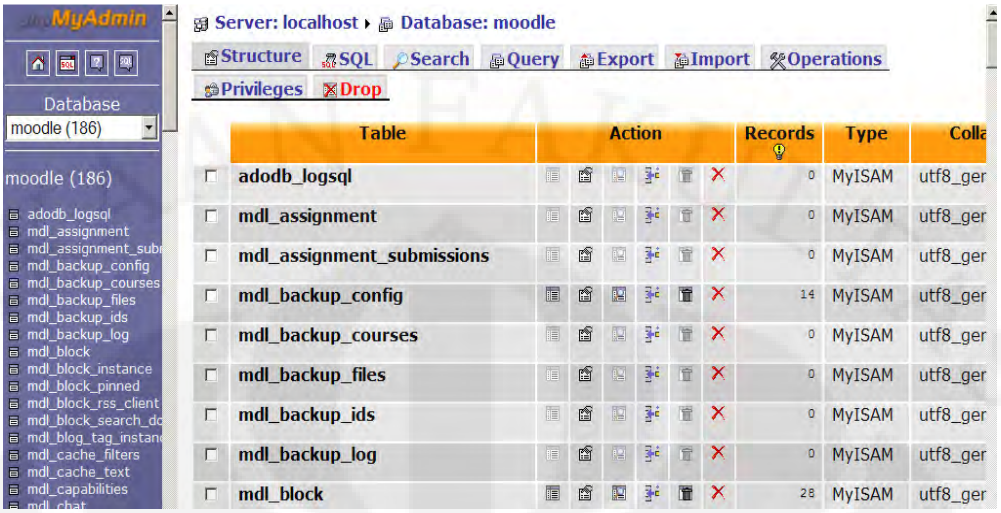
Apache, IIS (*Internet Information Service*), Caudium, Xitami, Omni dan *web server* lainnya. PHP juga mampu berkomunikasi hampir dengan semua sistem *database* yang ada sekarang, seperti MySQL, PostgreSQL, Oracle dan lain-lain.

Script PHP dieksekusi di komputer *server* dimana *script* tersebut dijalankan, kemudian hasilnya dikirim ke *web browser client*. PHP membuat sebuah halaman web menjadi lebih dinamis, lebih interaktif dan halaman yang ditampilkan dibuat saat *client* melakukan *request* halaman tersebut sehingga informasi yang diterima oleh *client* adalah informasi yang baru.

2.6.2. MYSQL

MySQL (*My Structure Query Language*) adalah salah satu *database* dari sekian banyak *database* lain seperti Oracle, MySQL, PostgresSQL dan banyak lagi. Kesemuanya itu mempunyai fungsi dan manfaat yang hampir sama namun dalam pengerjaanya sedikit berbeda tetapi MySQL adalah penggunaan yang paling mudah.

Sifat MySQL adalah *database* manajemen sistem (DBMS). DBMS (*Database Manajemen System*) merupakan salah satu system dalam mengakses *database* yang menggunakan bahasa SQL, MySQL menggunakan bahasa SQL dan dapat dikatakan sebagai DBMS. Salah satu sifat unik MySQL yang membedakan dari *database* seperti Oracle adalah *open source*. Artinya memungkinkan untuk semua orang untuk menggunakan dan memodifikasi *software*. Setiap orang dapat men-download MySQL dari internet dan menggunakannya tanpa membayar apapun. Hal lainnya mengapa *database* MySQL umum digunakan karena sifat *database* MySQL sangat cepat, *reliable*, dan mudah untuk digunakan, selain itu MySQL telah banyak menangani pembuatan software besar. Akan tetapi pada saat ini yang umum berkembang adalah MySQL diintegrasikan dengan pembuatan web dimana pada umumnya bahasa pemrograman web yang digunakan adalah PHP.



Gambar 2.2. GUI PHPMyAdmin

Selain MySQL banyak *database server* lain yang beredar di pasaran, namun selain menyediakan dukungan *open source*, MySQL memiliki beberapa keunggulan lain. Pertama adalah kemampuannya menangani jutaan user dalam waktu yang bersamaan. Kelebihan ini tentu cocok untuk dimanfaatkan pada sebuah program penilaian esai yang ujiannya mungkin diikuti ratusan bahkan ribuan murid.

2.6.3. Apache

Apache pada umumnya digunakan oleh *web server Web server* digunakan untuk menyediakan *web pages* yang diminta oleh computer klien. Klien biasanya meminta dan melihat *web pages* menggunakan aplikasi *web browser* seperti firefox, opera atau mozilla. Pengguna memasukkan sebuah *Uniform Resource Locator (URL)* untuk menunjuk pada sebuah *web server* dengan alat *Fully Qualified Domain Name (FQDN)* dan sebuah path untuk sumber yang diberikan.

Protokol yang paling sering digunakan untuk mengirim *web pages* adalah *Hyper Text Transfer Protocol (HTTP)*. Protokol lain seperti *Hyper Text Transfer Protocol over Secure Sockets Layer (HTTPS)*, dan *File Transfer Protocol (FTP)*, protokol untuk meng-*upload* dan men-*download* file, juga didukung. Apache *web server* seringkali digunakan dengan kombinasi *MySQL database engine,* bahasa tulisan *HyperText Preprocessor (PHP)* dan bahasa tulisan lainnya seperti *phyton* dan *perl*. Konfigurasi ini disebut *WAMP (Windows, Apache, MySQL and PHP)*

dan membentuk sebuah program yang kuat untuk pengembangan dan penyebaran dari aplikasi berbasis *web*.

2.6.4. Jama

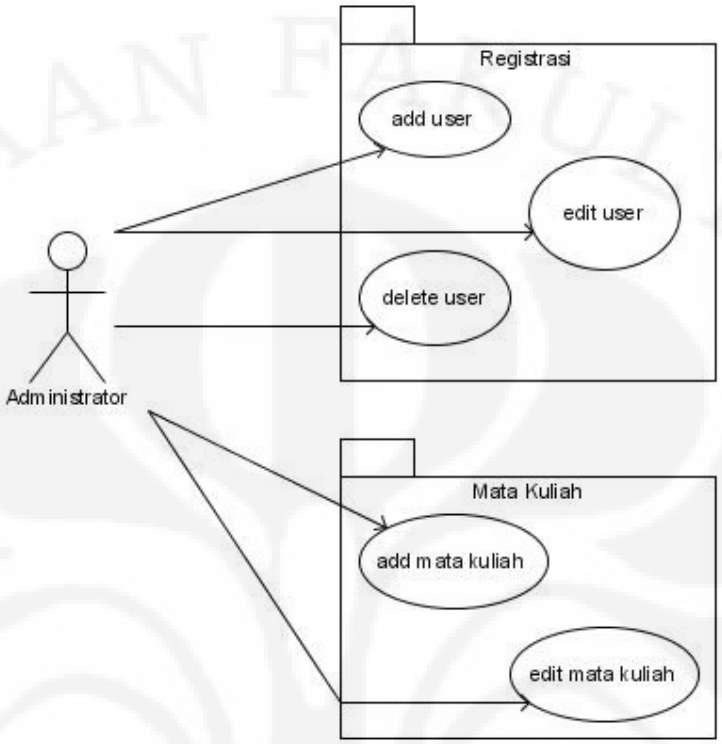
Selain MATLAB, aplikasi matematis *web based* yang bisa digunakan untuk penghitungan SVD adalah JAMA. JAMA adalah singkatan dari *Java Matrix.* JAMA merupakan *script* php untuk perhitungan matriks kompleks. Class-class dari package JAMA akan sering digunakan dalam operasi matriks seperti perkalian matriks, transpose, dan inverse. Karena JAMA hanya merupakan script PHP bukan merupakan aplikasi maka kinerja server-pun tidak akan terlalu terbebani.

BAB III
PERANCANGAN PENGEMBANGAN APLIKASI DAN MEKANISME PEMBOBOTAN PADA PENILAIAN ESAI OTOMATIS

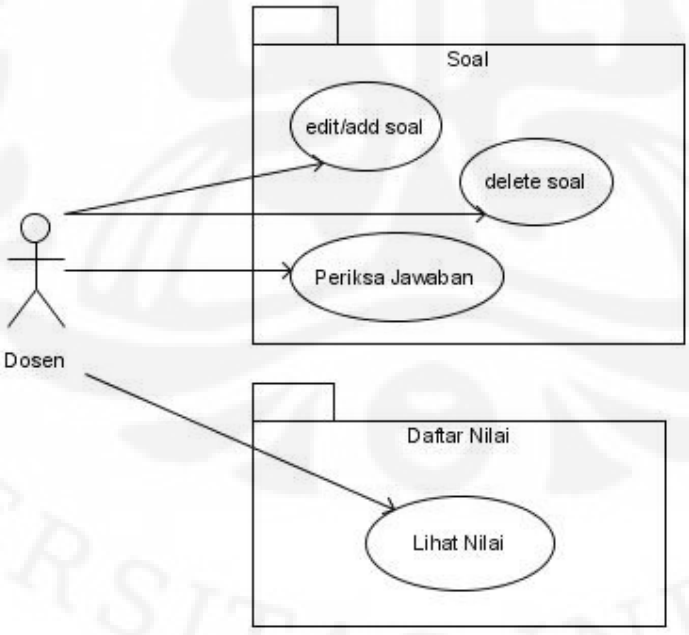
3.1. KONSEP APLIKASI

Aplikasi dirancang sebagai *web-based application,* sehingga mahasiswa, dosen dan administrator bisa berada dimana saja. Aplikasi ini merupakan modifikasi dari aplikasi penilaian esai otomatis yang sudah ada di Departemen Teknik Elektro Universitas Indonesia. Secara struktur, aplikasi ini terbagi menjadi tiga *interface,* yang pertama adalah antarmuka untuk administrator yang akan melakukan administrasi mengenai peserta dan mata kuliah yang ada, yang kedua adalah antarmuka untuk dosen untuk memasukkan soal dan jawaban serta untuk menjalankan penilaian esai otomatis, yang ketiga adalah antarmuka mahasiswa untuk menginput jawabannya. Aplikasi berjalan melalui beberapa tahapan yaitu proses pemasukan data, pemrosesan data, penyimpanan data dan penampilan hasil bagi pengguna. Pemasukan dan pemrosesan data menggunakan skrip PHP dan untuk penyimpanannya menggunakan database MySQL.

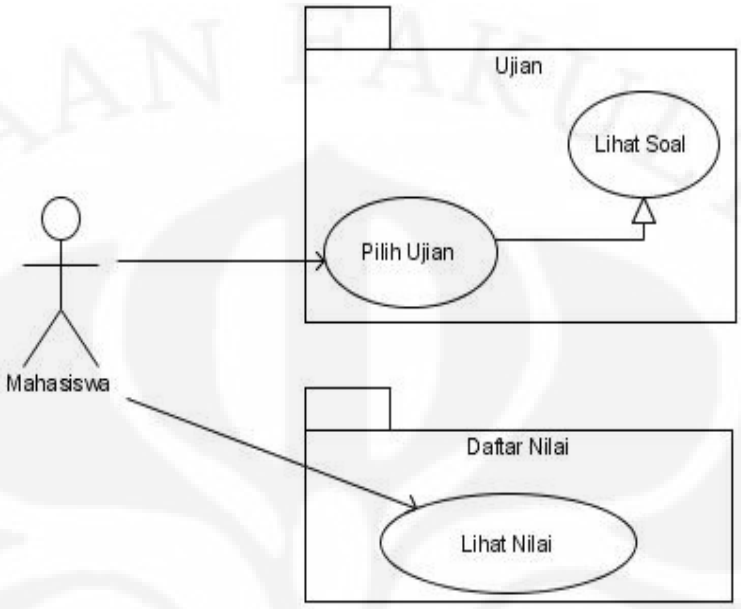
Untuk mengimplementasikan pembobotan seperti yang dijelaskan pada subbab 2.5 maka pada SIMPLE-O dilakukan perubahan pada bagian inti dari proses yaitu dengan memasukan jawaban mahasiswa dan dosen hanya masuk ke database tanpa dilakukan perhitungan nilai, dan proses penilaianya ada pada menu tersendiri. Use Case Diagram dari SIMPLE-O yang dimodifikasi untuk masing-masing *user* ditunjukan pada Gambar 3.1 untuk user administrator, Gambar 3.2. untuk user dosen, dan Gambar 3.3 untuk user mahasiswa



Gambar 3.1. Use Case Diagram Administrator

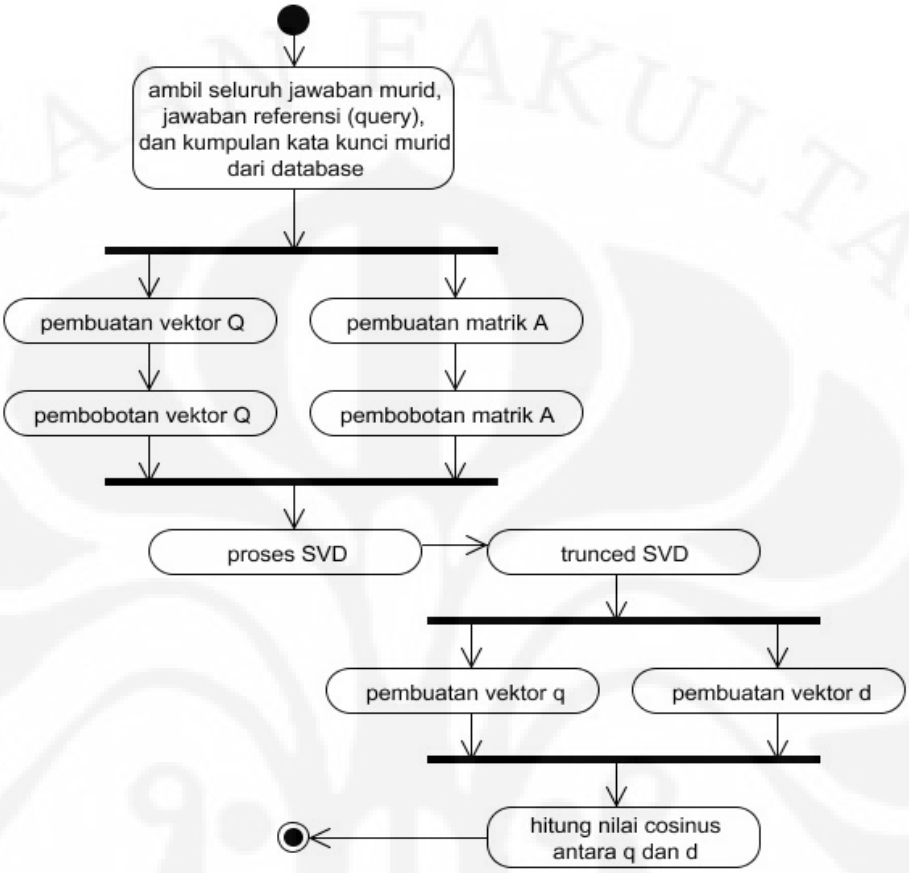


Gambar 3.2. Use Case Diagram Dosen



Gambar 3.3.Use Case Diagram Mahasiswa

Seperti telah diuraikan pada subbab 2.5, ada banyak metode penilaian yang bisa dilakukan. dengan mengkombinasikan pembobotan lokal dan global yang berbeda. Pada skripsi ini hanya tiga metode yang digunakan. Penilaian dilakukan dengan menjadikan kumpulan jawaban mahasiswa sebagai kumpulan dokumen dan jawaban referensi sebagai query. Kumpulan jawaban mahasiswa tersebut nantinya akan menjadi *semantic space* atau disebut juga Matriks A sedangkan jawaban referensi akan menjadi vector Q. Pada gambar 3.4 menunjukkan activity diagram dari proses penilaian otomatis jawaban mahasiswa.



Gambar 3.4. Activity Diagram proses penilaian otomatis jawaban mahasiswa

Semua data jawaban baik mahasiswa maupun dosen yang akan diolah sebelumnya mengalami *preprocessing text*. *Preprocessing text* dilakukan untuk tujuan penyeragaman dan kemudahan pembacaan serta proses LSA selanjutnya. *Preprocessing* jawaban mahasiswa ataupun jawaban meliputi:

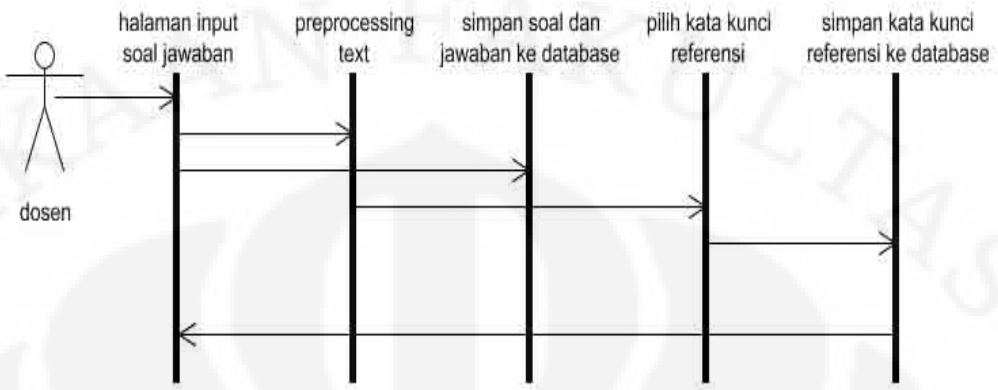
- Semua huruf dalam jawaban dijadikan huruf kecil atau *lower case*
- Penghilangan *white space* dan karakter null seperti spasi kosong panjang, dan tab. Dari sini diharapkan antar kata dalam hanya dipisahkan oleh 1 buah karakter spasi saja

- Penghilangan sejumlah karakter angka yang membentuk numerik sebuah bilangan (karakter numerik yang dicampur dengan alfabet tidak dihilangkan. Contohnya : ”ipv6”, ”CDMA2000”)
- Penghilangan karakter-karakter diluar alfabet terbaca seperti titik, koma, tanda kurung, #, $, %, &, *, !, ? dan sejenisnya

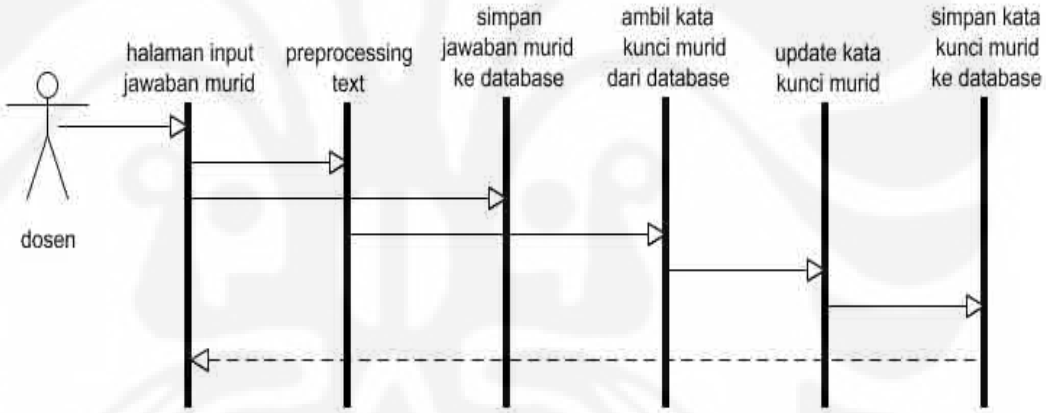
Setelah melalui tahap *preprocessing*, jawaban mahasiswa akan disimpan kedalam database sesuai *field*-nya. Langkah berikutnya adalah proses *update* kata_kunci_mahasiswa. Dalam LSA dikenal istilah *term* yakni sekumpulan kata kunci yang terdapat dalam dokumen-dokumen (jawaban mahasiswa). Dalam hal ini, dokumen berarti jawaban mahasiswa dan term adalah kata_kunci_mahasiswa. Kombinasi kemunculan sejumlah kata_kunci_mahasiswa terhadap jawaban-jawaban mahasiswa akan membangun matriks A yang diperlukan dalam proses LSA nantinya. Kumpulan kata_kunci_mahasiswa dipilih secara otomatis oleh program. Berikut ketentuan pemilihan dan pemrosesan kata kunci :

- Didalam daftar kata kunci tidak boleh ada dua kata kunci yang sama
- Daftar kata kunci urut sesuai abjad

Proses *updating* kata kunci dilakukan dengan cara mengambil daftar kata kunci yang ada di *database*, kemudian dibandingkan dengan kata kunci dari jawaban mahasiswa yang baru masuk. Setelah melalui ketiga aturan diatas, daftar kata kunci baru diperoleh dan disimpan *(update)* ke *database*. Proses input jawaban mahasiswa berakhir disini dengan menampilkan pesan sukses. *Sequence diagram* yang menunjukkan peralihan dari satu proses ke proses lain dalam sisi mahasiswa dan dosen diperlihatkan pada gambar 3.5 dan gambar 3.6.



Gambar 3.5. Sequence Diagram input soal jawaban sisi dosen



Gambar 3.6. Sequence Diagram input jawaban sisi mahasiswa

3.2. PROSES PENILAIAN

3.2.1. Implementasi Pembobotan

Setelah matriks A dan vektor Q selesai dibuat maka pembobotan baru bisa dilakukan. Pembobotan bertujuan untuk meningkatkan performa LSA seperti yang telah dijelaskan dalam subbab 2.5. Pada dasarnya pembobotan dibagi menjadi tiga yakni pembobotan lokal, pembobotan global dan normalisasi. Dengan mengacu pada tabel 2-6, 2-7, dan 2-8 maka terdapat banyak skema

pembobotan yang mungkin dilakukan. Karena pembobotan dibagi menjadi tiga, maka setiap skema akan melakukan tiga kali perhitungan. Untuk vektor *query*, tidak perlu dilakukan normalisasi. Hal ini karena vektor *query* hanya terdiri dari satu buah esai yaitu jawaban dari dosen. Hasil yang diharapkan adalah matriks A dan vektor Q yang sudah dibobotkan. Masing-masing disimpan dalam bentuk variabel Array_A dan Array_Q.

Pembobotan bertujuan untuk meningkatkan performa LSA seperti yang telah dijelaskan dalam subbab 2.5 . Skema pembobotan dapat dipilih oleh dosen sendiri di halaman periksa jawaban mahasiswa. Skema pembobotan yang disediakan ditunjukkan pada Tabel 3-1.

Tabel 3-1. Skema Pembobotan

No.	Pembobotan Mahasiswa			Pembobotan Dosen	
	Lokal	Global	Normalisasi	Lokal	Global
1	FREQ	NONE	NONE	FREQ	NONE
2	BNRY	IGFF	COSN	BNRY	IDFB
3	LOGA	IGFF	COSN	LOGA	IDFP
4	SQRT	NONE	COSN	BINARY	NONE

Variabel pilihan pembobotan dari user disimpan dalam variabel Pembobotan. Kemudian, mekanisme pemilihan proses pembobotan sesuai yang diminta dosen ditunjukkan pada Gambar 3.7

```
Proc Pilih_Pembobotan()

switch (Pembobotan) {

      case Pembobotan=1 :
            Bobot_lokal_A = bobot_lokal_FREQ();
            Bobot_global_A = bobot_global_NONE();
            Bobot_normalisasi_A = bobot_normalisasi_NONE();

            Bobot_lokal_Q = bobot_lokal_FREQ();
            Bobot_global_Q = bobot_global_NONE();

      case Pembobotan=2:
            Bobot_lokal_A = bobot_lokal_BNRY();
            Bobot_global_A = bobot_global_IGFF();
            Bobot_normalisasi_A = bobot_normalisasi_COSN();

            Bobot_lokal_Q = bobot_lokal_BNRY();
            Bobot_global_Q = bobot_global_IDFB();

      case Pembobotan=3:
            Bobot_lokal_A = bobot_lokal_LOGA();
            Bobot_global_A = bobot_global_IGFF();
            Bobot_normalisasi_A = bobot_normalisasi_COSN();

            Bobot_lokal_Q = bobot_lokal_FREQ();
            Bobot_global_Q = bobot_global_IDFP();

      case Pembobotan=4:
            Bobot_lokal_A = bobot_lokal_LOGA();
            Bobot_global_A = bobot_global_IGFS();
            Bobot_normalisasi_A = bobot_normalisasi_COSN();

            Bobot_lokal_Q = bobot_lokal_FREQ();
            Bobot_global_Q = bobot_global_IDFB();
}

Eproc
```

Gambar 3.7. Algoritma Proses Pemilihan Pembobotan

Pada dasarnya pembobotan dibagi menjadi tiga yakni pembobotan lokal, pembobotan global dan normalisasi, maka setiap skema akan melakukan tiga perhitungan pembobotan. Untuk vektor *query*, tidak perlu dilakukan normalisasi. Seperti yang terlihat dalam cuplikan kode pada gambar 3.7, perhitungan pembobotan dilakukan oleh fungsi tersendiri. Misalnya perhitungan pembobotan lokal LOGA yang rumusnya ada di Tabel 2.6, dikerjakan oleh fungsi bobot_lokal_LOGA(). Fungsi ini memiliki nilai *return* atau output berupa matriks (dalam bentuk *array*) dengan dimensi $i \times j$.

Keluaran dari setiap fungsi merupakan matriks, dimana dimensinya sesuai dengan definisi setiap pembobotan dalam Tabel 2.6, 2.7 dan 2.8 . Jadi semua fungsi pembobotan lokal akan memiliki output berupa matriks berdimensi $i \times j$, fungsi pembobotan global akan mengeluarkan output berupa matriks berdimensi $i \times 1$ dan fungsi normalisasi memberikan keluaran matriks $1 \times j$.

Proses berlanjut dengan mengalikan semua elemen pembobotan untuk membentuk skema pembobotan. Seperti yang ditunjukkan gambar 3.8.

```
Proc Pembobotan()
for(i=0; i<jumlah_Keywords ; i++) {
   for(j=0; j<jumlah_Dokumen; j++) {
       Array_A[i][j]=Bobot_Lokal_A[i][j]*Bobot_Global_A[i]*Bobot_Normal
isasi_A[j];
       Array_Q[i]   =Bobot_Lokal_Q[i]*Bobot_Global_Q[$i];
   }
}
Eproc
```

Gambar 3.8. Algoritma Pembobotan

Proses *looping* dilakukan untuk melakukan perkalian tiap elemen matriks. Disini, *looping* dalam bertugas menelusuri seluruh dokumen (jawaban mahasiswa) sementara *looping* luar menelusuri kata kunci. Dengan kata lain,

proses perkalian matriks dilakukan dengan pola menyelesaikan sebuah baris dulu baru kemudian menelusuri semua kolom. Sampai kolom terakhir, barulah pindah baris. Hasil yang diharapkan adalah matriks A dan vektor Q yang sudah dibobotkan. Masing-masing disimpan dalam bentuk variabel *array* Array_A dan Array_Q.

3.2.2. Operasi SVD

Operasi SVD dijalankan setelah matriks A dan vektor Q selesai dibobotkan. Kalkulasi SVD yang telah dijelaskan tahap demi tahap dalam Subbab 2.3. Proses ini akan menghasilkan U, Σ dan V.

Setelah U, Σ dan V diperoleh, dilakukan truncated SVD. Kali ini triplet SVD yakni U, Σ dan V akan mengalami pengecilan dimensi baris dan kolom, jika ditentukan faktor k=2 maka :

Tabel 3-2. Dimensi matriks SVD

Hasil SVD Awal	Dimensi	Truncated SVD	Dimensi
U	$i \times j$	Uk	$i \times 2$
Σ	$j \times j$	Σk	2×2
V	$j \times j$	Vk	$2 \times j$

Matriks-matriks hasil truncated yakni U_k, Σ_k dan V_k bertujuan untuk membentuk ruang vektor sesuai faktor-*k* disini, faktor *k* adalah 2 sehingga vektor dokumen dan vektor *query* berdimensi 2. Hasil ini bisa dilihat melalui persamaan (2-1) dan (2-2).

Truncated SVD dilakukan dengan bantuan *library* JAMA, dari *class* Matrix. Berikut algoritmanya yang ditunjukkan gambar 3.9 :

```
Proc TruncatedSVD()

k = 2;
Matriks_Sk = Matriks_S.getMatrix(1, k, 1, k);
Matriks_Uk = Matriks_U.getMatrix(1, Baris_U, 1, k);
Matriks_Vk = Matriks_V.getMatrix(1, k, 1, Kolom_V);

Eproc
```

Gambar 3.9. Algoritma Truncated SVD

Variabel Baris_U dan Kolom_V berturut-turut adalah baris matriks U dan V hasil SVD awal. Yang perlu diperhatikan adalah dalam *object* matrix, indeks awal adalah 1. Proses *truncated* intinya dilakukan oleh tiga baris terakhir dalam prosedur ini. Ketiga baris itu sama-sama menjalankan method getMatrix dan hasilnya berupa object, disimpan ke Matriks_Sk, Matriks_Uk dan Matriks_Vk. getMatrix sendiri merupakan *method* yang bertujuan untuk meng-*capture* atau memecah matriks dari object sesuai indeks baris dan kolom (awal dan akhir) dengan pola sintaks :

getMatrix(baris_awal, baris_akhir, kolom_awal, kolom_akhir)

Matriks-matriks hasil truncated yakni U_k, Σ_k dan V_k bertujuan untuk membentuk ruang vektor sesuai faktor-*k* disini, faktor *k* adalah 2 sehingga vektor dokumen dan vektor *query* berdimensi 2.

3.2.3. Penghitungan Nilai Cosinus

Sebelum dilakukan penghitungan nilai cosinus terlebih dahulu jawaban dosen (query) dan jawaban mahasiswa (dokumen) dibentuk vektornya. Untuk membentuk vektor *query* urutan langkahnya adalah sebagai berikut. Pertama, Matriks Σ_k di-*inverse* setelah itu matriks q di-*transpose*, kemudian lakukan

perkalian matriks sesuai persamaan (2-4). Dalam blok kode diatas, perkalian matriks dilakukan dengan method times.

Untuk membentuk vektor d, urutannya mirip seperti pembentukan vektor *query* hanya saja, yang di-*transpose* adalah matriks d. Matriks d sama dengan kolom matriks A. Oleh karena itu persamaan (2-2) dapat dikembangkan menjadi $D = A^T U_k \Sigma_k^{-1}$. Dengan D merupakan matriks yang kolomnya merupakan vektor 2 dimensi setiap dokumen (jawaban mahasiswa) yang ada. Pembentukan vektor dokumen dan *query* yang lebih detail dapat mengikuti cuplikan kode pada gambar 3.12 berikut ini.

```
Proc Buat_vektor_q()

Matriks_Sk_invers = Matriks_Sk.inverse();
UkSki = Matriks_Uk.times(Matriks_Sk_invers);
Matriks_Q_transpose = Matriks_Q.transpose();
Vektor_Q = Matriks_Q.times(UkSki);

Eproc
```

```
Proc Buat_vektor_dokumen()

Matriks_A_transpose = Matriks_A.transpose();
Matriks_D = Matriks_A_transpose.times(UkSki);

Eproc
```

Gambar 3.10. Algoritma Pembentukan Vektor Query dan Vektor Dokumen

Untuk penilaian antara esai mahasiswa dilihat dari hasil perhitungan cosinus antara vektor jawaban mahasiswa dan jawaban dosen sesuai persamaan (2-3). Namun sebelumnya perlu dilakukan penghitungan Frobenius Norm untuk vektor *query* dan vektor dokumen agar didapatkan panjang vektor *query* dan vektor dokumen. Rangkaian algoritmanya ditunjukkan pada Gambar 3.11:

```
Proc Hitung_Frobenius_vektor_Q()

Frobenius_Q = Vektor_Q.normF();

Eproc
```

```
Proc Hitung_Frobenius_vektor_D()

for(i=0; i<Baris_D; i++) {
      Frobenius_D[i] = Vektor_D[i].normF();
}

Eproc
```

Gambar 3.11. Algoritma Perhitungan Frobenius

Dari blok kode diatas didapatkan besarnya | d | dan | q |. Norm dari d disimpan dalam bentuk array di variabel Frobenius_D dengan indeks mengikuti indeks dokumen. Kemudian perhitungan nilai kosinus antara vektor *query* dan dokumen dapat dilakukan dengan *looping* sebagai berikut :

```
Proc Hitung_Kosinus_vektor_D()

Baris=Jumlah_Dokumen
for(i=0; i<Baris; i++) {

      // Menghitung Q . D

      Vektor_Dt = $Vektor_D.transpose();
      QD[i] = Vektor_Q.times(Vektor_Dt);

      // Menghitung Q . D / |Q| |D|

      Cosine[i] = QD[i]/(Frobenius_Q*Frobenius_D[i]);

      }
Eproc
```

Gambar 3.12. Algoritma Proses Kalkulasi Kosinus

Hasil perhitungan kosinus setiap dokumen (jawaban mahasiswa) terhadap *query* (jawaban dosen) disimpan dalam array nilai. Pada array ini berisi nilai dari seluruh mahasiswa untuk satu soal. Berarti jika ada sebanyak n soal maka akan ada sebanyak n array nilai. Nilai yang berada pada indeks awal database ini merupakan nilai dari mahasiswa dengan absensi awal (ID teratas), sehingga untuk mengetahui nilai keseluruhan untuk seorang mahasiswa maka yang perlu dilakukan adalah menjumlahkan nilai-nilai yang ada pada array sesuai dengan indeksnya.

BAB IV
IMPLEMENTASI DAN ANALISIS APLIKASI

4.1. IMPLEMENTASI SISTEM

Aplikasi dijalankan secara *web based*, sehingga diperlukan sebuah *web server* . Instalasi PHP *engine* dan *mysql* Sebagai *database* juga mutlak diperlukan sebelum dilakukan pengetesan. Ujicoba dilakukan pada sebuah perangkat komputer dengan spesifikasi sebagai berikut:

Prosesor	: Intel Core2Duo 2,2 GHz
RAM	: 2 Gb
Mainboard	: Biostar GF7050V-M07

Pada ujicoba pengambilan data keadaan *server* dan *client* berada dalam satu sistem (*localhost*). Setelah *server* dijalankan, *client* dapat langsung mengakses aplikasi dengan aplikasi *browser.* Semua perangkat lunak yang digunakan dalam perancangan dan ujicoba aplikasi sistem penilaian esai otomatis ini adalah sebagai berikut

Sistem Operasi	: Microsoft Windows XP 64 Professional SP2
Pengedit teks dan kode	: Notepad++
Browser	: Mozilla Firefox 2.0
Web Server	: Apache 2.0.49
Bahasa Pemrograman	: PHP 4.3.9
Database	: MySQL 4.1.20

4.1.1. Perubahan Setting di PHP

Untuk menjalankan aplikasi ini sebelumnya harus dilakukan perubahan setting di php bagian yang harus di ubah adalah *"resource limit"* dan *"data handling"*. Seperti yang ditunjukkan gambar 4-1.

```
;;;;;;;;;;;;;;;;;;
; Resource Limits ;
;;;;;;;;;;;;;;;;;;

max_execution_time = -1
max_input_time = -1
memory_limit = -1


;;;;;;;;;;;;;;;;
; Data Handling ;
;;;;;;;;;;;;;;;;
;
register_globals = On
```

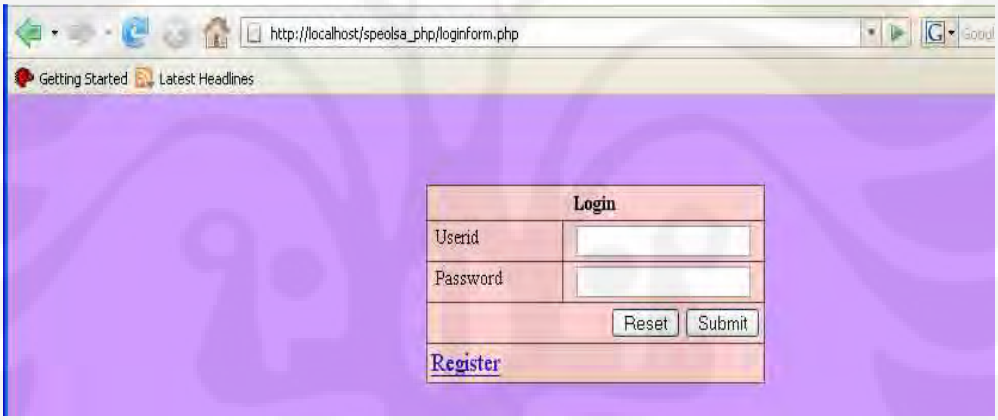
Gambar 4.1 Perubahan Setting di PHP

Seperti yang diketahui bahwa aplikasi ini membawa lebih banyak kalkulasi matematis dibandingkan proses pengumpulan data dan manajemen *database*. Karena itu diperlukan beberapa perlakuan khusus untuk menghindari terjadinya *hang* atau *failed attempt* dari PHP. Karena PHP didesain sebagai bahasa pemrograman berbasis *web* yang mudah dipakai, dan bukan bertujuan untuk melakukan proses *scientific,* maka ada *setting* alokasi memori dan waktu proses yang perlu diubah. Buka *file* ”php.ini” yang ada didalam *folder* aplikasi PHP atau *folder web server* menggunakan *script editor.* Cari bagian ”*resource limits*” kemudian lakukan beberapa perubahan seperti yang ditunjukkan gambar 4.1. Max_execution_time merupakan variabel yang menetapkan berapa lama waktu eksekusi (waktu proses) maksimal sebuah skrip PHP. Variabel max_input_time berisi nilai yang menetapkan waktu maksimal sebuah skrip PHP mendapatkan input, dalam hal ini melakukan *fetching* data dari luar, misalnya dari *database*. Sementara memory_limit menjelaskan besarnya memori maksimal untuk alokasi skrip. Ketiga variabel ini memiliki nilai *default* dari PHP berturut-turut 30 detik, 30 detik dan 8 *MegaByte*. Ketiga buah nilai ini ternyata tidak cukup untuk menjalankan skrip aplikasi *information retrieval,* untuk itu dilakukan perubahan seperti yang terlihat dalam blok. Perubahan menjadi -1 untuk

max_execution_time yang (-1 berarti alokasi waktunya tidak dibatasi), -1 sekon untuk max_input_time dan -1 untuk memory_limit (-1 berarti alokasi memori tak dibatasi sesuai dengan kebutuhan). Selain bagian "*resource limits*" yang perlu di ubah adalah bagian adalah bagian *"data handling"* pada variabel register_globals diubah nilainya dari off menjadi on.

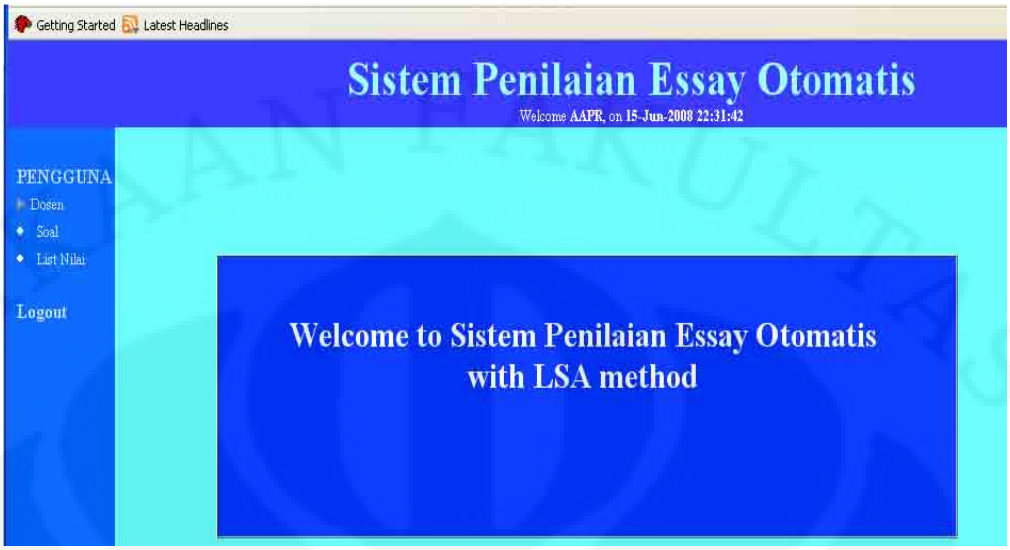
4.1.2. Program Utama

Penjelasan mengenai jalannya program harus didahului dari halaman paling muka yaitu pintu masuk untuk *login* pengguna seperti yang ditunjukkan oleh gambar 4.1. Untuk membuat sebuah ujian harus ada soal yang dimasukkan maka pertama-tama yang dimasukkan pada login adalah dosen. Bila pengguna belum terdaftar maka bisa melakukan register sebagai pengguna baru.



Gambar 4.2 Halaman *login*

Halaman muka setelah dosen melakukan login ditunjukkan pada Gambar 4.2. Pada panel sebelah kiri terlihat menu-menu yang bisa dipilih dosen mulai dari List Nilai dan Soal.

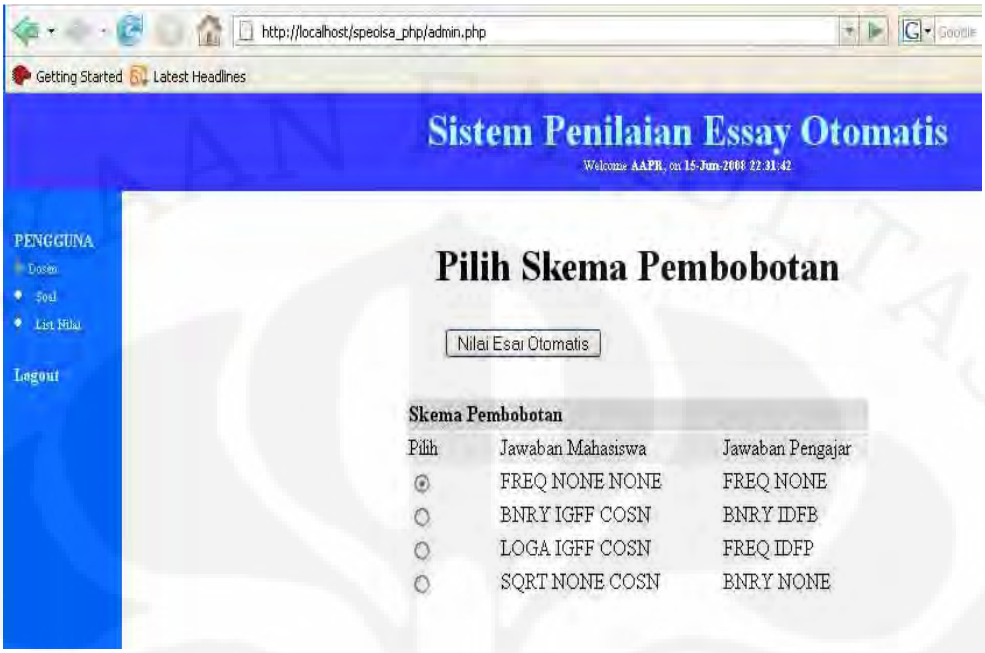


Gambar 4.3 Halaman Depan untuk Dosen

Setelah *login* maka dosen bisa memilih menu "Soal" pada bagian kiri tengah untuk membuat soal baru. Muncul halaman yang menunjukkan mata kuliah apa saja yang diajarkannya dan pilihan untuk mengedit maupun melakukan input soal terhadap mata kuliah tersebut. Selain menu menambah soal, pada halaman khusus dosen juga ada pilihan menu lain seperti melihat daftar nilai dari seluruh muridnya yang telah mengikuti ujian pada menu "List Nilai".

4.1.3. Fitur Pemilihan Pembobotan

Fitur pemilihan pembobotan ini dapat digunakan untuk memilih skema pembobotan penilaian jawaban esai otomatis. Sebelum masuk ke fitur ini harus sudah ada jawaban mahasiswa yang disimpan di database terlebih dahulu. Untuk dapat menggunakan fitur ini user harus login sebagai dosen. Setelah memilih skema pembobotan maka dosen dapat melihat hasil penilaiannya pada menu list nilai. Pada aplikasi ini hanya disediakan 3 menu pembobotan dan 1 menu tanpa pembobotan seperti yang ditunjukkan oleh gambar 4.3. Setelah dipilih pembobotannya maka untuk menilai secara otomatis dilakukan dengan meng-klik "Nilai Esai Otomatis" yang terdapat diatas menu skema pembobotan.



Gambar 4.4 Pemilihan Skema Pembobotan

4.2. ANALISIS KECEPATAN PROSES

Pada bagian ini akan dijelaskan beberapa data dan analisis beberapa hal yang dapat mempengaruhi kecepatan proses penilaian esai otomatis. Untuk bahan pengamatan, dilakukan beberapa penilaian esai otomatis dengan pemilihan pembobotan yang berbeda-beda secara acak. Dari beberapa kali percobaan dengan kombinasi pembobotan didapatkan beberapa hal yang perlu diperhatikan untuk mendapatkan peningkatan kecepatan proses penilian esai otomatis secara signifikan, diantaranya :

1. **Banyaknya Jawaban Mahasiswa**

Banyaknya jawaban mahasiswa merupakan salah satu faktor yang mempengaruhi kecepatan proses dari sistem penilaian esai otomatis. Pada ujicoba ada perbedaan waktu pemrosesan ketika hanya ada 5 jawaban dari mahasiswa dibandingkan lebih dari itu. Ujicoba dilakukan dengan jawaban

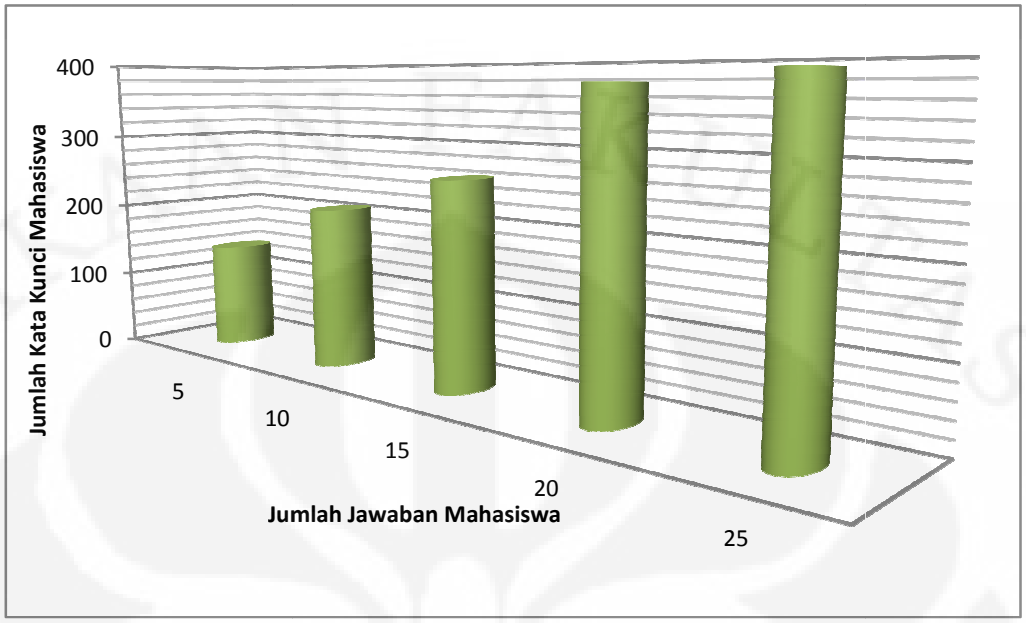
mahasiswa dengan 1 soal dan untuk penilaiannya menggunakan skema pembobotan 2. Hasil pencatatan waktu ditunjukkan tabel 4-1.

Tabel 4-1. Banyaknya Jawaban Mahasiswa Berpengaruh kepada Waktu Pemrosesan

Jumlah jawaban	Rata-rata Waktu Pemrosesan Yang dibutuhkan
5	0.0996
10	0.283307
15	0.661444
20	1.202278
25	2.342805

Hasil yang ditunjukkan tabel 4-1 disebabkan jika jawaban mahasiswa semakin banyak maka akan membuat ukuran dari matriks A yang dibuat dimensinya akan semakin besar. Baris dari matriks A merupakan semua kata-kata yang ada pada jawaban mahasiswa sedangkan baris dari matriks A merupakan jawaban dari mahasiswanya dan kedua hal ini saling berhubungan. Semakin banyak jawaban mahasiswa maka kata kunci mahasiswanya juga semakin banyak. Jumlah baris dan kolom ini akhirnya akan sangat mempengaruhi kecepatan proses karena elemen matriks yang akan melalui operasi matematis akan semakin banyak.

Dari hasil pengamatan dapat disimpulkan bahwa hubungan antara banyaknya kata kunci dan dokumen adalah linier. Diagram yang ditunjukkan gambar 4-4 memaparkan hubungan antara penambahan jumlah jawaban mahasiswa dengan banyaknya kata kunci mahasiswa:



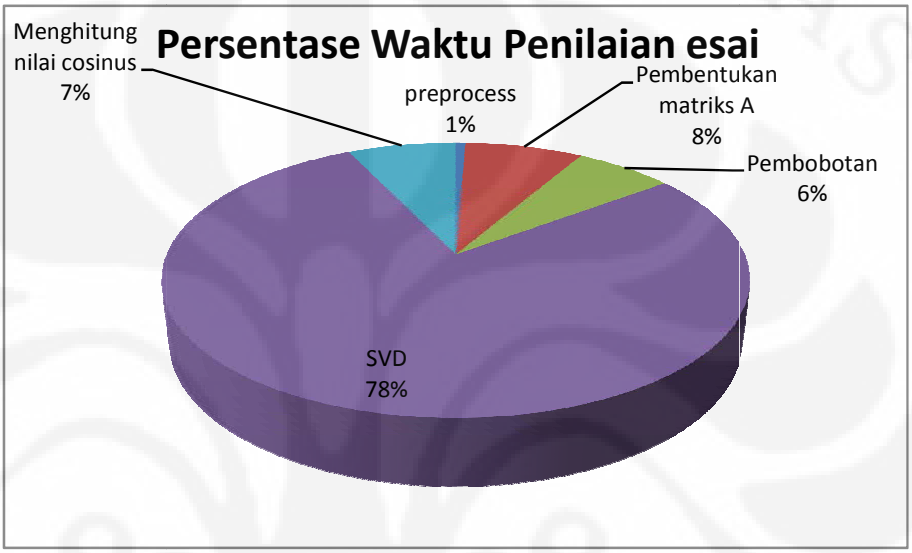
Gambar 4.5. Diagram hubungan banyaknya kata kunci mahasiswa yang dibentuk dengan jumlah jawaban mahasiswa

Solusi yang dapat dilakukan adalah menggunakan mekanisme pembentukan kata kunci yang lebih efektif. Perlu dilakukan penelitian lebih lanjut untuk menebak kata-kata apa saja yang benar-benar merupakan kata kunci dari jawaban mahasiswa, tetapi jika mahasiswa dalam ujian hanya menebak-nebak kata kunci maka tidak akan seperti ujian esay yang sebenarnya. Metode lain yang adalah dengan penghilangan kata umum pada kata kunci jawaban mahasiswa kata umum seperti hanya, sebuah, merupakan, dll tetapi akan menjadi suatu kekurangan jika dosen menggunakan kata umum sebagai kata kunci jawabannya.

2. Algoritma Pemrograman

Algoritma menjadi faktor yang sangat dasar yang dapat meningkatkan performansi kecepatan proses. Satu hal yang masih mengganjal dalam aplikasi ini adalah masalah kalkulasi SVD. Dalam aplikasi ini kalkulasi SVD dilakukan setiap akan melakukan penilaian esay otomatis. Kalkulasi SVD memakan sebagian besar waktu proses perolehan informasi. Melalui suatu pengamatan didapatkan jika proses perolehan informasi memakan waktu total

sampai 0.627 sekon, maka kalkulasi SVD sendiri memakan waktu 0.50787 sekon, sekitar 78% dari waktu total. Beberapa pengamatan menghasilkan fakta bahwa kalkulasi SVD akan selalu memakan waktu lebih dari 75% waktu total.



Gambar 4.6. Diagram Proporsi Waktu Tiap Bagian Pemrosesan

3. Spesifikasi *Hardware* dan Banyaknya *User Agent*

Spesifikasi *hardware*, terutama prosesor dan RAM yang dipakai oleh *server,* akan berpengaruh pada waktu pemrosesan. Hal ini dikarenakan PHP merupakan bahasa pemrograman *server-side* yang artinya seluruhnya dikerjakan di *server.* Selain itu, dari beberapa ujicoba yang dilakukan, banyaknya aplikasi yang sedang aktif dan proses yang dilakukan di server juga dapat menurunkan performa kecepatan proses.

Banyaknya *user* dosen yang memakai penilaian esay otomatis ini dalam waktu yang bersamaan akan menyita waktu pemrosesan di *server.* Dalam ujicoba, ada perbedaan waktu pemrosesan ketika hanya ada 1 *user agent* dibandingkan lebih dari satu yang mengakses. Ujicoba dilakukan dengan cara membuka *tab/window browser* baru sebagai *user* lain yang mengakses aplikasi. Kali ini, dilakukan untuk soal, jumlah soal, dan

jumlah mahasiswa yang sama dan dengan skema pembobotan 4. Hasil pencatatan waktu dan banyaknya user terdapat dalam Tabel 4-2.

Tabel 4-2. Banyaknya *User Agent* Berpengaruh kepada Waktu Pemrosesan

Banyaknya *user agent*	Waktu Pemrosesan rata-rata tiap *user agent* (sekon)
1	30.10836
2	30.36695
3	30.35917
4	30.4772

4.3. ANALISIS PERBANDINGAN PENILAIAN OTOMATIS DENGAN *HUMAN RATER*

Untuk melihat pengaruh pembobotan terhadap penilaian jawaban yang dilakukan secara manual maka dilakukan pengujian untuk satu soal ujian. Skema pembobotan yang dibandingkan antara lain ditunjukkan oleh tabel 4-3.

Tabel 4-3. Skema Pembobotan yang akan dibandingkan

No.	Pembobotan Jawaban Mahasiswa			Pembobotan Dosen	
	Lokal	Global	Normalisasi	Lokal	Global
1	FREQ	NONE	NONE	FREQ	NONE
2	BNRY	IGFF	COSN	BNRY	IDFB
3	LOGA	IGFF	COSN	FREQ	IDFP
4	SQRT	NONE	COSN	BNRY	NONE

Soal yang di ujikan adalah quiz mata kuliah jaringan komputer. Jawaban dari mahasiswa diambil 20 mahasiswa secara acak dari peserta quiz tersebut. Untuk mengukur data dari skema pembobotan mana yang paling mendekati dengan

human rater maka dihitung korelasi antara data dari human rater. Nilai korelasi yang paling bagus adalah 1 sedangkan jika data yang diperoleh sangat tidak berhubungan maka nilai korelasinya adalah 0. Hasil perbandingan korelasi antara penilaian manual penilaian otomatis dengan skema pembobotan yang berbeda-beda ditunjukkan pada tabel 4-4.

Tabel 4-4. Hasil Korelasi antara data human rater dengan skema pembobotan

Skema Pembobotan	Nilai Korelasi dengan Human Rater
(1) FREQ NONE NONE . FREQ NONE	0.003
(2) BNRY IGFF COSN . BNRY IDFB	0.1
(3) LOGA NONE COSN . FREQ NONE	0.2
(4) SQRT NONE COSN . BNRY NONE	0.39

Dari data hasil percobaan dan perhitungan pada tabel 4-4 menunjukkan skema pembobotan 4 merupakan skema yang paling mendekati perhitungan manusia dengan nilai korelasi yang paling besar.

Ada beberapa hal yang dapat mempengaruhi hasil dari penilaian otomatis yang merupakan kekurangan dari aplikasi ini antara lain adalah imbuhan kata karena pada aplikasi ini hanya memeriksa persamaan dari kata yang ada sehingga jika kata kuncinya adalah “mengirim” dan pada jawaban mahasiswa ada kata “mengirimkan” maka kata yang ada di mahasiswa tidak mendapat nilai karena berbeda. Hal lainnya adalah persamaan kata atau sinonim karena ada banyak sinonim dalam bahasa Indonesia contohnya adalah kata mengawasi dan memonitor memiliki arti yang sama tetapi pada aplikasi ini tentu saja dianggap berbeda.

KESIMPULAN

Beberapa kesimpulan yang dapat diambil dari hasil ujicoba dan analisis yang telah dilakukan antara lain:

1. Implementasi pembobotan pada penilaian esay otomatis dengan menggunakan teknik LSA telah berhasil di implementasikan dengan bahasa pemrograman PHP. Kalkulasi waktu yang mengambil porsi yang paling besar dalam waktu pemrosesan aplikasi adalah kalkulasi SVD yang memakan waktu 78% dari total proses.

2. Beberapa hal lain yang berpengaruh pada kecepatan proses perolehan dokumen antara lain banyaknya kata kunci mahasiswa dan jumlah jawaban mahasiswa, spesifikasi perangkat keras yang digunakan *server*, banyaknya *user dosen* yang sedang melakukan penilaian otomatis secara bersamaan dan masalah fundamental dalam algoritma pemrograman.

3. Berdasarkan hasil pengamatan dan perhitungan, skema pembobotan yang tingkat korelasinya paling mendekati *human rater* adalah skema yang ke 4 dalam percobaan.

4. Hal-hal yang mempengaruhi hasil penilaian esai otomatis adalah adanya kata berimbuhan dan persamaan arti kata.

DAFTAR ACUAN

[1] Kukich, K., "Beyond Automated Essay Scoring", IEEE Intelligent System, pp 22-23, September/October 2000

[2] Deerwester, Scott. Et al, "Indexing by Latent Semantic Analysis", University of Chicago, October 2002

[3] Erica Chisholm, Tamara G. Kolda, *New Term Weighting Formulas for the Vector Space Method in Information Retrieval,* Oak Ridge National Laboratory, US 1999

[4] Valenti, S., Neri, F., Cucchiarelli, A., "An Overview of Current Research on Automated Essay Grading", Journal of Information Technology Education, vol. 2, pp 321-330, Ancona, Italy, 2003

[5] Landauer, T.K.,Foltz, P.W., Laham, D., "Introduction to Latent Semantic Analysis", Discourse Processes,1998

DAFTAR PUSTAKA

Foltz, P., Laham, D., Landauer, T., "Automated Essay Scoring: Application to Educational Technology", 1999.

Valenti, S., Neri, F., Cucchiarelli, A., "An Overview of Current Research on Automated Essay Grading", Journal of Information Technology Education, vol.2, 2003.

Foltz, P., Laham, D., Landauer, T., "The Intelligent Essay Assessor: Applications to Educational Technology", Colorado, 2002

Hasdianto, Bayu., "Penerapan Fitur Pendistribusian Bahan Ajar dan Feedback Penyajian Grafik Data dan Statistik Pada Sistem Penilaian Esai Otomatis Metode Latent Semantic Analysis", Skripsi, Teknik Elektro FTUI, Januari 2006

Pinandito, Nareswara A., "Pengembangan Sistem Ujian dan Keamanan Pada Sistem Penilaian Esai Otomatis Metode Latent Semantic Analysis ", Skripsi, Teknik Elektro FTUI, Januari 2006

Cosentino, Chritoper., Advanced PHP for Web Professionals, CV. Prentice Hall PTR, October 2002

Lampiran-1 : PENGHITUNGAN NILAI KORELASI

	human rater	SKEMA 1	SKEMA 2	SKEMA 3	SKEMA 4
1	30	27.28	91.70	44.80	58.46
2	10	29.28	75.69	97.97	54.69
3	100	60.97	57.68	60.42	98.49
4	70	12.86	88.64	35.11	77.80
5	60	0.71	11.98	69.05	17.06
6	70	38.13	78.29	57.16	94.80
7	50	60.38	97.02	99.16	20.55
8	100	18.13	63.70	12.42	98.67
9	60	58.95	76.45	83.58	96.55
10	60	69.89	92.75	80.50	99.47
11	100	56.58	60.78	52.57	99.88
12	80	70.75	72.66	71.62	95.83
13	20	21.56	86.74	8.95	99.58
14	60	56.67	85.44	41.62	56.72
15	10	3.58	24.71	29.78	14.77
16	10	77.01	30.98	99.89	95.47
17	20	88.41	60.84	99.99	40.96
18	20	91.72	68.53	68.55	100.00
19	40	23.35	88.73	5.26	71.35
20	40	17.46	81.84	42.11	58.63
NILAI KORELASI		-0.0032	0.0965	0.1903	0.3897

Lampiran-2 : PENGHITUNGAN WAKTU PEMROSESAN

Jumlah User	Waktu Yang di butuhkan (s)				
	User 1	User 2	User 3	User 4	Rata-rata
1user	30.00295				30.00295
	30.08711				
	30.08016				
	30.26323				
	30.00295				
2 user	30.44195	30.13312			30.36695
	30.22863	30.44647			
	30.37948	30.46755			
	30.13158	30.70684			
	30.36695	30.36695			
3 user	30.26881	30.3974	30.29644		30.35917
	30.06417	31.04475	30.01844		
	30.14712	30.63801	30.05851		
	30.23521	30.89254	30.24861		
	30.35917	30.35917	30.35917		
4 user	30.6006	29.93406	30.05502	30.38448	30.4772
	29.60414	30.46619	30.74493	31.45559	
	30.40187	30.77739	30.34829	30.76827	
	30.28929	30.75636	30.31367	30.73508	
	30.4772	30.4772	30.4772	30.4772	