REPOSITOR, Vol. 2, No. 9, September 2020, Pp. 1183-1194
ISSN : 2714-7975
E-ISSN : 2716-1382

Aplikasi Tanya Jawab Otomatis Seputar Rekayasa Perangkat Lunak Dengan Menggunakan Metode Cosine Similarity Berbasis Android

Rizky Heriawan P.T[*1], Maskur [2], Nur Hayatin, S.[3]
[1,2,3]Teknik Informatika/Universitas Muhammadiyah Malang
rizkyheriawan2e@gmail.com[*1], maskur@umm.ac.id[2], noor.hayatin@umm.ac.id[3]

Abstrak

Jawaban pertanyaan aplikasi penjawab pertanyaan yang tersedia saat ini masih menggunakan metode pencocokan kata kunci untuk melakukan pencarian atas jawaban. Sistem penjawab pertanyaan otomatis adalah sistem yang secara otomatis mencoba menemukan kembali informasi yang benar untuk pertanyaan diajukan oleh user. Pertanyaan dapat dikembangkan untuk membantu dan membuat lebih mudah untuk menjawab pertanyaan tentang rekayasa perangkat lunak. Aplikasi ini menggunakan metode Cosine Similarity yang merupakan salah satu solusi untuk membantu mencari jawaban pertanyaan yang diinginkan dengan tepat, yang bermanfaat untuk sistem pengolah kata. Karena dengan metode ini, tanya jawab otomatis dapat mencari data yang diinginkan oleh penanya, dengan menampilkan jawaban dengan bobot tertinggi sebagai jawaban yang paling tepat. Jawaban pertama atau bobot tertinggi yang dihasilkan oleh sistem adalah jawaban yang benar menurut penilaian sistem dan pakar. Jawaban pertama atau bobot tertinggi yang dihasilkan oleh sistem adalah jawaban yang benar menurut penilaian sistem, pakar dan pengujian Kappa. Hasil pengujian menggunakan kappa statistik memberikan nilai terbaik Kappa pada jawaban pertama (jawaban dengan bobot terbesar). Nilai tersebut membuktikan bahwa sistem yang telah dibangun dapat digunakan untuk mengetahui kemiripan antar kasus penggunaan pertanyaan dan jawaban.

Kata Kunci: *Cosine Similarity, Rekayasa Perangkat Lunak,Tanya Jawab*

Abstract

The Answers of question answering applications that are available today are still using keyword matching method to perform a search for answering. Automatic question answering system is a automatically system used to find information that might correspond to the questions asked by the user. Questions can be developed to help and make it easier to answer questions about software engineering. This application uses the method of Cosine Similarity which is one solution to help searching for the desired answer of questions correctly, that is useful for word processing system. By this method, Automatic Question Answering can looking for desired data of user by showing the the highest weights answer as the best answer. The first or the highest answer resulted by system is the right answer based on system, expert and Kappa Testing. The result of Kappa testing giving the best Kappa value on the first answer (the highest weights answer). It proves that the system can be used to know the similarity between question and answer for between cases of using quetions and answers.

Keywords: *Cosine Similarity, Software engineering, Question Answering*

1. Pendahuluan

Rekayasa perangkat lunak merupakan satu disiplin ilmu yang bertujuan mengembangkan sistem perangkat lunak yang efektif dari segi biaya. Perangkat lunak bersifat abstrak dan tidak nyata. Perangkat lunak tidak terbuat dari unsur, mengikuti hukum fisika atau proses manufaktur. Dalam beberapa hal, kenyataan ini menyederhanakan rekayasa perangkat lunak karena tidak ada pembatasan fisis terhadap potensi perangkat lunak. Akan tetapi, dalam hal lain, tidak adanya batasan natural ini berarti bahwa perangkat lunak dengan mudah dapat menjadi sangat kompleks dan dengan demikian sangat sulit dipahami.

Rekayasa perangkat lunak masih merupakan disiplin yang relatif muda. Istilah rekayasa perangkat lunak pertama kali diajukan pada tahun 1968 pada konferensi yang diselenggarakan untuk membahas apa yang pada waktu itu disebut ‘krisis perangkat lunak’. Krisis perangkat lunak

ini merupakan akibat langsung dari lahirmya perangkat keras komputer generasi ketiga yang canggih (pada waktu itu). Kecanggihannya membuat aplikasi komputer yang belum terealisasi pada saat itu menjadi proposisi yang layak. Perangkat lunak yang dihasilkan menjadi beberapa kali lipat lebih besar dan lebih kompleks dari sistem perangkat lunak sebelumnya.

Rekayasa perangkat lunak di Indonesia dijadikan disiplin ilmu yang dipelajari mulai tingkat sekolah menengah kejuruan sampai tingkat perguruan tinggi. Di tingkat perguruan tinggi, jurusan ini sudah memiliki kurikulum materi pelajaran sendiri yang sudah ditentukan oleh jurusan. Rekayasa Perangkat Lunak di tingkat perguruan tinggi biasanya mempelajari materi seperti bahasa pemprograman, desain web, dan sebagainya, tergantung dari kurikulum tiap tahunnya.

Smartphone android merupakan perangkat mobile yang sering di bawa oleh masyarakat umum sehingga mempermudah pengguna dalam mencari informasi hanya dengan menggunakan smartphone user dapat menggali informasi melalui media internet, atau melalui aplikasi lain yang tertanam dalam perangkat tersebut [1].

Melihat dari permasalahan yang telah dipaparkan di atas, pada tugas akhir ini akan dibuat sebuah sistem dengan memanfaatkan metode cosine similarity dalam menentukan jawaban atas pertanyaan-pertanyaan seputar *software engineering* berbasis android.

2. Landasan Teori

2.1 *Question And Answering System*

Question answering system merupakan sebuah sistem yang mengizinkan *user* menyatakan kebutuhan informasinya dalam bentuk yang spesifik dan alami, yaitu dalam bentuk *natural language question* dan tidak mengembalikan daftar dokumen yang harus disaring oleh *user* untuk menentukan apakah document-dokumen tersebut mengandung jawaban atas pertanyaan, tetapi mengembalikan kutipan teks singkat atau frasa sebagai jawaban [2].

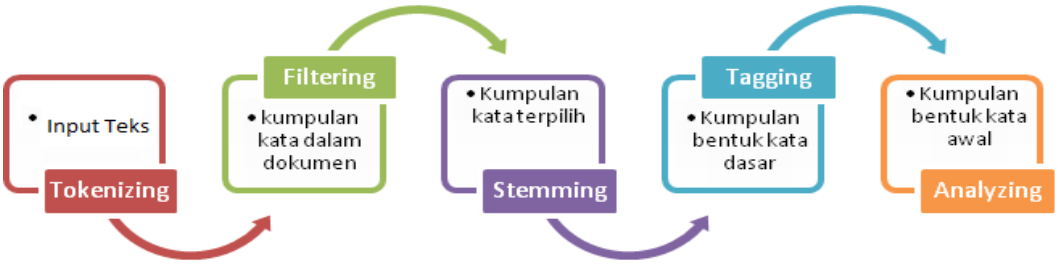
Question answering adalah bentuk khusus dari pencarian informasi. Mengingat koleksi dokumen, *Question answering* sistem adalah sistem yang mencoba menemukan kembali informasi yang benar untuk pertanyaan diajukan dalam bahasa alami.

Question answering adalah sebuah bentuk dari *information retrieval* yang berkaitan dengan jawaban yang tepat yang diberikan oleh pertanyaan dengan bahasa alami. Sebuah *question-answering system (QAS)* mencoba untuk menemukan kembali jawaban eksplisit dalam sebuah bentuk jawaban tunggal, potongan teks dari sebuah dokumen atau kumpulan dari dokumen. Tantangan terbesar di dalam *QAS* adalah bagaimana cara mengelompokkan sebuah pertanyaan ke dalam kategori tertentu yang selanjutnya akan akan digunakan untuk menemukan jawaban yang tepat dari sebuah dokumen yang besar.

2.2 *Text Mining*

Text Mining merupakan proses otomatis atau sebagian proses otomatis untuk teks. Ini melibatkan pembentukan text yang lebih terstruktur dan penggalian informasi yang relevan dari teks [3].

Text Mining selalu berurusan dengan kata – kata, jutaan kata – kata yang disimpan dalam bentuk file elektronik. File elektronik ini biasa berbentuk beberapa dokumen yang akan diproses, namun tentu saja dokumen – dokumen ini belum dalam bentuk yang terstruktur. Butuh mekanisme untuk menambang teks - teks yang ada dalam koleksi dokumen sehingga didapatkan informasi – informasi yang lebih bernilai dan terstruktur. Mekanisme tersebut dibagi dalam beberapa tahapan (fase *pre-processing*). Tahapan-tahapan seperti pada Gambar 1 yang dilakukan secara umum dalam text mining, yaitu: T*okenizing, Filtering, Stemming, Tagging,* dan *Analyzing* [4].



Gambar 1. Tahapan Text Mining [4]

2.3 *Text Preprocesing*

Proses ekstraksi ini bertujuan untuk menghasilkan *term-term* yang akan digunakan sebagai *prototype* bagi setiap dokumen. Tiap *term* tersebut dicari bentuk kata dasar-nya berdasarkan kamus kata dasar Bahasa Indonesia. Hal ini untuk menghindari tersimpannya kata-kata yang memiliki kata dasar yang sama namun berimbuhan berbeda. Disamping itu dilakukan penyaringan (*filtering*) terhadap kata-kata yang tidak layak untuk dijadikan sebagai pembeda. Kelompok kata ini biasanya disebut sebagai stoplist. Oleh karena belum tersedia maka penelitian ini juga berusaha mencari stoplist tersebut secara manual.

2.4 Text Transformation

Pada tahap ini dilakukan penyaringan (*filtration*). Penyaringan dilakukan dengan menentukan *term* mana yang akan digunakan untuk merepresentasikan dokumen sehingga dapat mendiskripsikan isi dokumen dan membedakan dokumen tersebut dengan dokumen lain dalam koleksi. *Term* yang sering dipakai tidak dapat digunakan untuk tujuan ini, setidaknya karena dua hal. Pertama, jumlah dokumen yang relevan terhadap suatu *query* kemungkinan besar merupakan bagian kecil dari koleksi. *Term* yang efektif dalam pemisahan dokumen yang relevan dari dokumen tidak relevan kemungkinan besar adalah *term* yang muncul pada sedikit dokumen. Ini berarti bahwa term dengan frekuensi kemunculan tinggi bersifat *poor descriminator*. Kedua, *term* yang muncul dalam banyak dokumen tidak mencerminkan definisi dan topik atau sub-topik dokumen. Karena itu, *term* yang sering digunakan dianggap sebagai *stop-word* dan dihapus.

Stop-word didefinisikan sebagai term yang tidak berhubungan *(irrelevant*) dengan subjek utama dari database meskipun kata tersebut sering kali hadir di dalam dokumen [5]. *Stopword* merupakan kata-kata yang bukan merupakan ciri (kata unik) sehingga dengan menghilangkannya dari suatu teks maka sistem hanya akan memperhitungkan kata-kata yang dianggap penting. Penghapusan *stop-word* dari dalam suatu koleksi dokumen pada satu waktu membutuhkan banyak waktu. Solusinya adalah dengan menyusun suatu pustaka *stop-word* atau *stop-list* dari *term* yang akan dihapus.

Konversi term ke bentuk akar (*stemming*) juga merupakan tindakan yang dapat dilakukan pada tahap ini*. Stemming* merupakan proses untuk mereduksi kata ke bentuk dasarnya. Kata-kata yang muncul di dalam dokumen sering mempunyai banyak varian morfologik. Karena itu, setiap kata yang bukan *stop-words* direduksi ke bentuk *stemmed word* yang cocok. Dengan cara ini, diperoleh kelompok kata yang mempunyai makna serupa tetapi berbeda wujud sintaksis daru dengan lainnya. Kelompok tersebut dapat direpresentasikan oleh satu kata tertentu. Pembahasan lanjut tentang *stemming* dipaparkan di pembahasan sebelumnya [6].

2.5 *Stemming* Bahasa Indonesia

Stemming dapat dikatakan sebagai proses membentuk suatu kata menjadi kata dasarnya. Misalnya:
berkata → kata
mengakatakan → kata
perkataan → kata
Beberapa algoritma dasar dalam *stemming* antara lain:

1) *Brute force stemming*. Algoritma ini adalah algoritma yang paling sederhana. Bermodalkan *database* kata dengan kata dasarnya, komputer dengan mudah mencari kata dasar. Namun metode ini mempunyai kelemahan yaitu jumlah *database* kata dan kata dasarnya harus besar. Kesalahan terjadi bila kata tidak ditemukan di *database* dan kemudian dianggap kata dasar, padahal bukan.
2) Menghilangkan imbuhan (awalan, akhiran, sisipan). Untuk menggunakan metode ini harus tahu terlebih dahulu aturan bahasanya. Kata akan dipotong imbuhannya berdasar aturan bahasanya. Kesalahan terjadi bila kata tersebut adalah kata dasar yang dipotong, misalnya: perawan → awan.
3) *Porter Stemmer*. Algoritma ini terkenal digunakan sebagai *stemmer* untuk bahasa Inggris. *Porter Stemmer* dalam bahasa Indonesia akan menghasilkan keambiguan karena aturan morfologi bahasa Indonesia [6][7].
4) *Nazief & Adriani Stemmer*. Algoritma ini paling sering dibicarakan dalam *stemming* bahasa Indonesia. Algoritma ini merupakan hasil penelitian internal UI (Universitas Indonesia) dan tidak dipublish secara umum [8]. Algoritma ini merupakan gabungan antara algoritma

menghilangkan imbuhan dan *brute force stemming*. Namun algoritma ini mempunyai dua masalah, yang pertama kemampuannya tergantung dari besarnya *database* kata dasar, dan yang kedua, hasil stemming tidak selalu optimal untuk aplikasi *information retrieval* (Tala, 2003).

5) Dan masih banyak algoritma-algoritma dasar lainnya, seperti gabungan algoritma di atas, *stokastik, lematasi*, dll.

Bila dibandingkan, untuk teks berbahasa Indonesia, *Porter stemmer* lebih cepat prosesnya daripada *Nazief & Adriani stemmer* namun algoritma *Nazief & Adriani* memilki tingkat keakuratan lebih tinggi daripada *Porter stemmer* [9].

2.7 Pembobotan *Tf-Idf*

Metode pembobotan yang paling sederhana terhadap suatu *term* (*term wighting*) adalah dengan menggunakan frekuensi kemunculan *term* (kata) / *term frequency* (*TF*) yang bersangkutan pada suatu dokumen. Eksperimen-eksperimen *pre-processing* dokumen berbasiskan frekuensi *term*, telah banyak dilakukan dalam bidang *information retrieval*. Namun, dalam kaitannya dengan performa *recall* dan *precision,* penggunaan frekuensi *term* saja ternyata hanya dapat memenuhi fungsi *recall*. Fungsi *precision* yang baik sayangnya tidak dapat dicapai dengan representasi frekuensi *term* saja pada suatu dokumen. *Precision* yang tinggi mengisaratkan kemampuan untuk membedakan suatu dokumen dengan dokumen yang lain untuk mencegah *retrieval* yang tidak diinginkan. Frekuensi *term* yang tinggi dapat digunakan dalam *pre-processing*, hanya jika frekuensi kemunculan *term* bersangkutan tidaklah tinggi pada dokumen – dokumen yang lainnya. Nilai *precision* yang baik pada kenyataannya dihasilkan oleh *term-term* yang kemunculannya tergolong jarang pada suatu dokumen, karena *term-term* bersangkutan seringkali menjadi pembeda signifikan antara dokumen-dokumen yang memiliki *term-term* tersebut dengan dokumen yang tidak memiliki *term-term* bersangkutan. [10]. Persamaan 1 menyatakan bobot (*w*) masing-masing dokumen terhadap kata kunci.

$$\boldsymbol{Wd,t} = \boldsymbol{tf\ d,t} \times \mathbf{IDF}\boldsymbol{t} \quad (1)$$

Dimana:
d = dokumen ke-d
t = kata ke-t dari kata kunci
$W_{d,t}$ = bobot dokumen ke-d terhadap kata ke-t

2.8 Penghitungan Tingkat Kemiripan *(Cosine Similarity)*

Perbandingan kemiripan (*similarity*) yang digunakan disini adalah standard *cosine similarity* dengan Persamaan 2 [10].

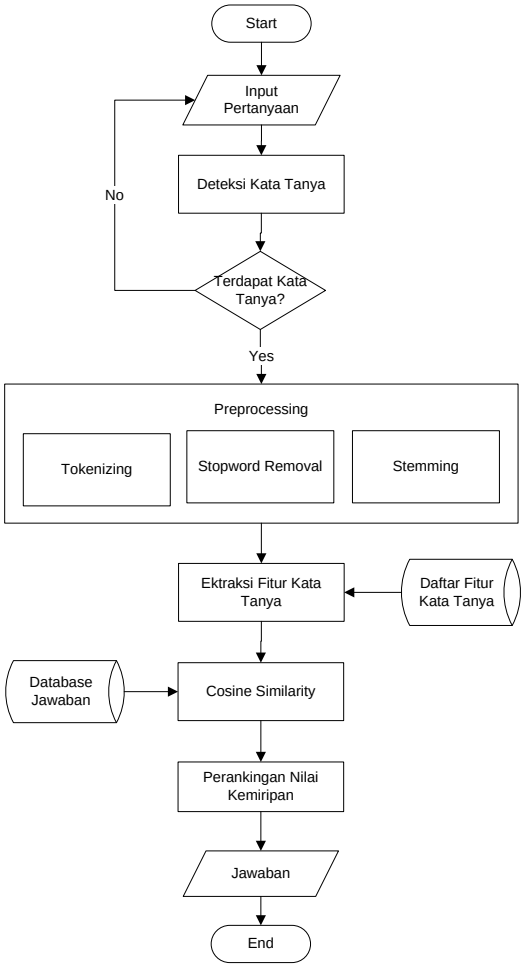
$$S_{D_i D_j} = \frac{\sum_{k=1}^{L} (weight_{ik} weight_{jk})}{\sqrt{\sum_{k=1}^{L} weight_{ik}^2 \sum_{k=1}^{L} weight_{jk}^2}} \quad (2)$$

S_{DiDj}: Similarity Dokumen ke I dan ke j

3. Analisa dan Perancangan Sistem

3.1 Desain Sistem

Perancangan sistem merupakan tahap awal dari perancangan aplikasi yang meluputi desain proses yang digambarkan dalam diagram alur atau flowchart, desain database yang di gambarkan dalam ERD dan desain *interface*. perancangan ini dilakukan untuk mengetahui kondisi system secara umum. Dalam prancangan sistem ini akan membahas mengenai Kerangka Sistem, tahapan *preprocessing* yang meliputi proses tokenizing, filtering / stopword removal dan stemming*, Cosine Similary* yang di mulai dari perhitungan bobot term dengan menggunakan *tf-idf,* dan Perangkingan Nilai Kemiripan*.* Gambar 2 berikut merupakan kerangka sistem yang akan dibuat.



Gambar 2. Flowchart Sistem

Dari Gambar 2 dapat dijelaskan awal mula system berjalan yaitu dengan cara user menginput pertanyaan kedalam system lalu pertanyaan tersebut akan di proses melalui tahapan *preprocessing* yang terdiri dari tiga bagian yaitu *Tokenizing* yang berfungsi untuk memparsing dan menghilangkan tanda baca, *Stopword Removal* yang berfungsi untuk menghilangkan kata tidak penting, dan *Stemming* yang berfungsi untuk menghilangkan awalan atau akhiran dari sebuah kata. Kemudian hasil *preprocessing* akan di proses kedalam proses selanjutnya yaitu proses pencarian kemiripan antara pertanyaan dengan data jawaban yang ada pada database dengan menggunakan metode cosine similarity lalu hasil kemiripan akan di urutkan dari nilai terbesar hingga terkecil, nilai terbesar akan di jadikan jawaban yang paling tepat yang akan di sajikan kepada *user*.

Berikut merupakan contoh perhitungan pencarian jawaban menggunakan metode cosine similarity:

Sebelum melakukan proses perhitungan perlu di ketahui bahwa setiap kata tanya memiliki fitur masing-masing berikut merupakan fitur setiap kata tanya:

1. Apa
    a. Adalah
    b. Yaitu
    c. Merupakan
2. Kapan
    a. Waktu
    b. Saat
    c. Tanggal
    d. Jam
3. Dimana
    a. Di

b. Tempat
c. Lokasi

4. Berapa
   a. Jumlah
   b. 0-9
   c. Angka

Contoh pertanyaan:
"Kapan waktu perbaikan atau restart?"
Contoh dokumen:
D1. Banyaknya kegagalan system untuk sejumlah permintaan layanan system tertentu
D2. Waktu antara kegagalan system
D3. Waktu perbaikan atau waktu restart yang dibutuhkan ketika terjadi kegagalan
Penyelesaian:
1. Lakukan proses preprocessing, Tabel 1 berikut merupakan hasil preprocessing

Tabel 1. Hasil Preprocessing

Term
kapan
waktu
perbaikan
restart
saat
tanggal
Jam
banyaknya
gagal
system
jumlah
minta
layan
tentu
butuh
Jadi

2. Hitung tf-idf, Tabel 2 dan Tabel 3 berikut merupakan hasil perhitungan tf-idf

Tabel 2. Hasil idf

Term	Tf					Idf
	Q	D1	D2	D3	Df	
kapan	1	0	0	0	1	0.84509804
waktu	1	0	1	2	4	0.243038049
perbaikan	1	0	0	1	2	0.544068044
Restart	1	0	0	1	2	0.544068044
Saat	1	0			1	0.84509804
tanggal	1	0			1	0.84509804
Jam	1	0			1	0.84509804
banyaknya	0	1	0	0	1	0.84509804
Gagal		1	1	1	3	0.367976785
System		1	1		2	0.544068044
Jumlah		1			1	0.84509804
Minta		1			1	0.84509804

Layan	1		1	0.84509804
Tentu	1		1	0.84509804
Butuh		1	1	0.84509804
Jadi		1	1	0.84509804

Tabel 3. Hasil tf-idf

Kapan	0.84509804	0	0	0
Waktu	0.243038049	0	0.243038	0.486076
perbaikan	0.544068044	0	0	0.544068
Restart	0.544068044	0	0	0.544068
Saat	0.84509804	0	0	0
tanggal	0.84509804	0	0	0
Jam	0.84509804	0	0	0
banyaknya	0	0.845098	0	0
Gagal	0	0.367977	0.367977	0.367977
System	0	0.544068	0.544068	0
Jumlah	0	0.845098	0	0
Minta	0	0.845098	0	0
Layan	0	0.845098	0	0
Tentu	0	0.845098	0	0
Butuh	0	0	0	0.845098
Jadi	0	0	0	0.845098

3. Hitung bobot kemiripan cosine similarity, dengan hasil seperti pada Tabel 4.

Tabel 4. Hasil Cosine Similarity

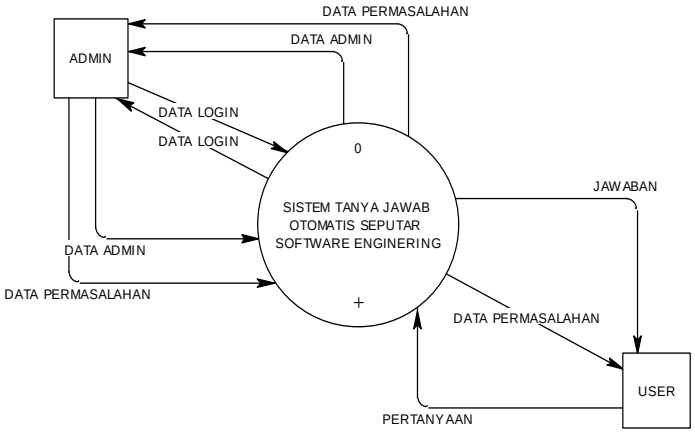
TERM	D1	D2	D3
Kapan	0	0	0
Waktu	0	0.059067	0.118135
Perbaikan	0	0	0.29601
Restart	0	0	0.29601
Saat	0	0	0
Tanggal	0	0	0
Jam	0	0	0
banyaknya	0	0	0
Gagal	0	0	0
System	0	0	0
Jumlah	0	0	0
Minta	0	0	0
Layan	0	0	0
Tentu	0	0	0
Butuh	0	0	0
Jadi	0	0	0
	0	0.059067	0.710155

Dari hasil hitung pembobota kemiripan maka didapat nilai yang paling tingg yaitu pada dokimen tiga (D3) dengan nilai 0.710 sehingga dapat disimpulkan bahwa jawaban yang paling tepat untuk pertanyaan diatas adalah dokumen ke 3.

3.4.1 Data Flow Diagram

1. Data Flow Diagram Level 0 (Diagram Konteks)

Pada Gambar 3, DFD level 0 ini terdapat 2 entitas luar yaitu user sebagai pengguna sistem dan dapat melakukan proses tanya jawab otomatis. dan admin sebagai pengelola sistem, pada user terdapat beberapa alir data yaitu data Data hasil cari, dan data keyword. Pada admin juga terdapat alir data yaitu data login, data admin, dan Data Data Permasalahan.



Gambar 3. Context Diagram

4. Implementasi dan Pengujian

4.1 Implementasi Antar muka

Implementasi antar muka terdiri dari beberapa tampilan pada menu-menu yang ada pada aplikasi. Ada dua antar muka yang dirancang, yaitu *user* antar muka *dan admin* antar muka. Desain dari *user* antar muka yang baik pada suatu sistem dapat mempermudah *user* untuk menggunakan sistem tersebut. Berikut *user* antar muka pada sistem yang sudah dibangun, diantaranya adalah:

4.1.1 Implementasi *User* Antar muka

Desain dari *user* antar muka yang baik pada suatu sistem dapat mempermudah *user* untuk menggunakan sistem tersebut. Berikut *user* antar muka pada sistem yang sudah dibangun, diantaranya adalah:

a. Main Menu

Main Menu pada Gmabar 4 merupakan tampilan awal untuk *user* yang didalamnya terdapat tiga menu utama yaitu menu pertanyaan yang berfungsi untuk menampilkan form pertanyaan, menu data permasahalan yang berfungsi untuk menampilkan data data permasahalan yang telah tersimpan pada database, dan menu *help* yang berfungsi untuk menampilkan cara menggunakan sistem tanya jawab.

Gambar 4. User Antar muka Main Menu

b. Menu Pertanyaan

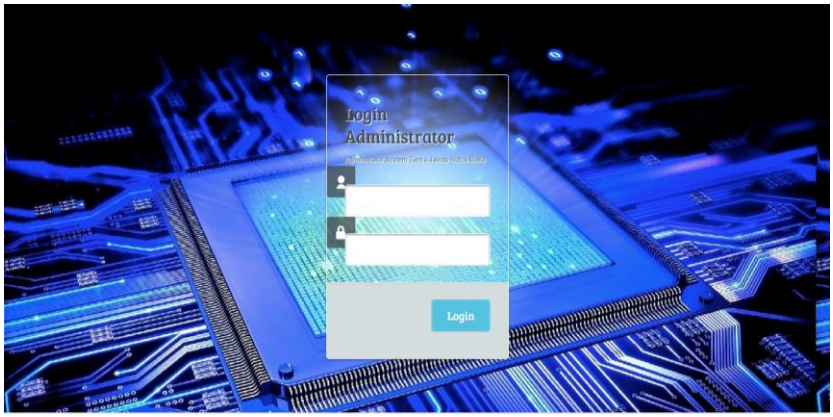
Gambar 5. User Antar muka menu pertanyaan

Pada Gambar 5, menu ini menampilkan antar muka untuk menginputkan pertanyaan, dan akan menampilkan beberapa kandidat jawaban, dan jawaban yang paling benar adalah jawaban yang berada pada posisi pertama.

4.1.2 Implementasi *admin* Antar muka

Implementasi *admin* antar muka terdiri dari beberapa tampilan pada menu-menu yang ada pada aplikasi. Desain dari *admin* antar muka yang baik pada suatu sistem dapat mempermudah *admin* untuk menggunakan sistem tersebut. Berikut *admin* antar muka pada sistem yang sudah dibangun, diantaranya adalah:

a. Login Form



Gambar 6. User Antar muka menu Data permasahalan

Login form pada Gambar 6, merupakan tampilan yang pertamakali muncul ketika admin akan melakukan proses pengolahan data yang berfungsi untuk memverifikasi hak akses admin untuk melakukan manajemen data.

b. Main Menu

Ketika login berhasil maka akan tampil main menu atau halaman *index* yang akan menjelaskan tentang menu apa saja yang terdapat pada admin dalam hal ini yaitu menu home, data permasahalan dan stemming.

Gambar 7. User Antar Muka Main Menu Pengelolaan Data Permasalahan

c. Menu Data permasahalan

Menu Gambar 8 merupakan data permasahalan berfungsi untuk mengelola data data permasahalan yang akan disimpan dalam database, peda menu ini terdapat sub menu yaitu tambah data, edit, dan delete.

Gambar 8. User Antar Muka Menu Data Permasahalan

4.3 Pengujian

4.3.1. Pengujian Akurasi

Data - data permasalahan yang digunakan adalah Data-data permasahalan yang di ambil dari buku berjudul rekayasa perangkat lunak karangan Ian Somerville. Pengujian dilakukan dengan menginputkan 20 pertanyaan. Dengan rincian kata tanya apa, bagaimana, kapan dan berapa. Dalam penelitian ini pengujian akurasi dilakukan dengan membandingkan jawaban yang di uji oleh pakar. Pengujian pakar merupakan pengujian dimana sistem diuji oleh pakar yang merupakan dosen di bidang rekayasa perangkat lunak. Pada pengujian ini pakar yang melakukan pengujin terhadap sistem adalah bapak Yuda Munarko, S.kom. M.Sc. yang merupakan dosen teknik informatika bidang minat rekayasa perangkat lunak di Universitas Muhammadiyah Malang.

Untuk menghitung nilai akurasi pencarian jawaban yang tepat akan di hitung dengan menggunakan rumus berikut:

$$\text{Akurasi} = \frac{\text{jawaban yang benar}}{\text{jumlah pertanyaan}} x\ 100\%$$

$$\text{Akurasi} = \frac{16}{20} x\ 100\%$$

$$\text{Akurasi} = 80.00\%$$

Dari hasil sebanyak 20 pertanyaan diatas, nilai akurasi pencarian jawaban yang benar dapat di simpulkan sebanyak 80.00% akurat. Untuk menghitung nilai precission pencarian jawaban yang tepat akan di hitung dengan menggunakan rumus berikut:

$$\text{Precission} = \frac{\text{jawaban yang relevan}}{\text{jumlah pertanyaan}} x\ 100\%$$

$$\text{Precission} = \frac{13}{20} x\ 100\%$$

$$\text{Precission} = 65.00\%$$

Dari hasil sebanyak 20 pertanyaan diatas, nilai precission pencarian jawaban yang relevan dapat di simpulkan sebanyak 65.00%.

4.3.2. Pengujian Konsistensi Cohen's Kappa

Uji Konsistensi Cohen's Kappa Merupakan ukuran yang menyatakan konsistensi pengukuran yang dilakukan dua penilai (Rater) atau konsistensi antar dua metode pengukuran atau dapat juga mengukur konsistensi antar dua alat pengukuran. Koefiseien Cohen's kappa hanya diterapkan pada hasil pengukuran data kualitatif (Kategorik). Contoh pada penentuan jawaban relevan dan tidak relevan, dimana dua peneliti diminta untuk menentukan jawaban relevan pada 20 data soal dan jawaban. Apakah penentuan jawaban relevan antara dua peneliti tersebut menunjukan hasil yang sama (kosisten)?.

		System	
		Benar	Salah
Pakar	Benar	16	0
	Salah	3	1

Dimana
Hasil pengukuran benar oleh
Pr(Sistem) = $\frac{16+3}{20}$ = 0.95 = 95%
Pr(Pakar) = $\frac{16+0}{20}$ = 0.80 = 80%
Hasil pengukuran salah oleh
Pr(Sistem) = $\frac{0+1}{20}$ = 0.05 = 5%
Pr(Pakar) = $\frac{3+1}{20}$ = 0.2 = 20%
Perubahan kemungkinan hasil pengukuran
Benar = 95% x 80% = 76%
Perubahan kemungkinan hasil pengukuran
Salah =5% x 20% = 1 %
Total perubahan pengukuran antar Rater = 76% + 1% = 77%
Berdasarkan hasil hitung diatas maka di dapat nilai koefisien kappa berikut:
K = $\frac{0.77-0.76}{1-0.76}$ = **0.0416**

Hasil Nilai koefiesien kappa dijadikan sebagai nilai ambang batas dari pengujian yang dilakukan oleh sistem. Apabila nilai bobot yang diuji oleh sistem lebih dari atau sama dengan nilai ambang batas maka dianggap relevan, kemudian apabila nilai dibawah dari nilai ambang batas maka dianggap kurang relevan.

5. Penutup

5.1 Kesimpulan

Berdasarkan pengujian pada bab sebelumnya, didapat kesimpulan dalam penelitian *auto answer* seputar Permasalahan *software engineering* menggunakan metode *cosine similarity* ini, diantaranya adalah:

1. Bobot jawaban tertinggi merupakan jawaban terbaik untuk setiap pertanyaan yang di inputkan kedalam sistem tanya jawab.
2. Dari hasil sebanyak 20 pertanyaan diatas, nilai precission pencarian jawaban yang relevan dapat di simpulkan sebanyak 65.00%.
3. Dari 10 pertannyaan yang menghasilkan jawaban relevan sebanya 8 pertanyaan dan 2 dari pertanyaan tidak relevan sehingga dapat di simpulkan bahwa sistem dapat menjawab pertanyaan dengan nilai 80% relevan.

5.2 Saran

Untuk lebih menyempurnakan dibutuhkan pengembangan untuk menjadikan sistem ini menjadi lebih baik. Adapaun beberapa saran untuk pengembangan sistem ini, diantaranya adalah:

1. Dapat di implementasikan pada beberapa platform smartphone yang lain seperti blackberry dan windows phone, sehingga dapat digunakan oleh lebih banyak orang.
2. Penambahan metode yang di gunakan untuk meningkatkan dan menyempurnakan sistem tanya jawab otomatis sehingga menghasilkan jawaban yang paling akurat.

Referensi

[1] Stephanus Hermawan Susanto, Mudah Membuat Aplikasi Android. Yogyakarta: C.V ANDI OFFSET, 2011.
[2] Gunawan dan Lovina, G., 2006, Question Answering System dan Penerapannya ada Alkitab. Jurnal Informatika. No. 1, Vol 7, hal 1-9.
[3] Miller, K. (2005), Communication Theories: Perspectives, processes, and ontexts, 2nd Ed; New York: McGraw-Hill.
[4] Riza, BAB 11 Text Mining, http://student.eepisits.edu/~risa/files/DataMining /chapter11.pdf, 2008.
[5] Cios, Krzysztof J. Etc.2007.Data Mining A Knowledge Discovery Approach, Springer. (online). (http://www.4shared.com/document/FyVdn5pm/Data_Mining_Knowledge_Discov.html, diakses 8 oktober 2014).
[6] Tala, F.Z., 2003, A Study of Stemming Effects on Information Retrieval in bahasa Indonesia. Master Thesis, Institut for logic, Language and Computation Universiteit van Amsterdam The Netherlands.
[7] Abdul Chaer. (2008). Morfologi Bahasa Indonesia (Pendakatan dan Proses). Jakarta: Rineka Cipta.
[8] Adriani, M., Asian, J., Nazief, B., Tahaghoghi, S. M., and Williams, H. E. (2007). Stemming indonesian: A confix-stripping approach. ACM Transactions on Asian Language Information Processing (TALIP), 6(4):1–33.
[9] Augusta, Ledy. 2009. “Perbandingan Algoritma Stemming Porter Dengan Algoritma Nazief & Adriani Untuk Stemming Dokumen Teks BahasaIndonesia”. Konferensi Nasional Sistem dan Informatika 2009, Bali, November 14, 2009.
[10] Mahendra, Krisnatuti D, Tobing A, Boy. Care Your Self DiabetesMellitus. Jakarta: Penebar Plus. 2008.