LONTAR KOMPUTER VOL. 8, NO.1, APRIL 2017 **p-ISSN 2088-1541**
DOI : 10.24843/LKJITI.2017.v08.i01.p04 **e-ISSN 2541-5832**

Perencanaan *Search Engine E-commerce* dengan Metode *Latent Semantic Indexing* Berbasis *Multiplatform*

Ni Made Ari Lestari[1], **Made Sudarma**[2]

Universitas Udayana
Jl. Kampus Bukit Jimbaran, Bali-Indonesia
[1]nm.arilestari@gmail.com
[2]msudarma@unud.ac.id

Abstrak

E-commerce merupakan sebuah transaksi jual beli yang terjadi melalui sistem elektronik seperti internet, WWW, ataupun jaringan komputer lainnya. E-commerce melibatkan pertukaran data elektronik dan sistem pengumpulan data otomatis. Sebuah kolom search engine untuk pencarian barang yang diinginkan oleh user disediakan di semua e-commerce. Search engine yang disediakan hanya menggunakan teknologi search engine biasa pada e-commerce seperti Tokopedia, Lazada, MatahariMall, Amazon, dan lainnya. Semakin panjang kalimat dari inputnya maka output atau hasil pencarian barangnya akan semakin luas dan banyak pada search engine biasa. Pemanfaatan teknologi semantic indexing, memungkinkan semakin panjang dan jelas input barang yang diinginkan maka jumlah pencarian sedikit dan akurat sesuai dengan input sehingga membantu user dalam pengambilan keputusan. Bagaimana membangun sebuah search engine dalam web e-commerce dengan menggunakan metode Latent Semantic Indexing dibahas pada penelitian ini. Metode yang digunakan yaitu metode teks mining untuk pengolahan kata, metode Levenshtein Distance untuk perbaikan kata otomatis dan Latent Semantic Indexing untuk pemrosesan informasi dan pengeluaran input. Tingkat akurasi untuk *search engine* yang dihasilkan sekitar 96,7%.

Kata kunci: *e-commerce, search engine, Latent Semantic Indexing, Text Mining, Levenshtein Distance.*

Abstract

E-commerce is a sale and purchase transactions that occur through electronic systems such as the Internet, WWW, or other computer networks. E-commerce involves electronic data interchange and automated data collection systems. In all e-commerce search engine provided a column for the search items desired by the user. In e-commerce such as Tokopedia, Lazada, MatahariMall, Amazon, and other search engines that provided just use a regular search engine technology. In the usual search engines getting longer sentences from the input or output of goods search results will be more extensive and more. However, by utilizing the semantic indexing technology, the longer and clear input desired goods, the number of searches will be few and accurately in accordance with the input that helps the user in decision making.

In this study discussed how to build a search engine on the web e-commerce by using Latent Semantic Indexing. The first starts from the use of Text Mining methods for word processing, and the method Levenshtein Distance to repair automatic word and the last Latent Semantic Indexing for information processing and input expenditure.

Keywords: *e-commerce, search engine, Latent Semantic Indexing, Text Mining, Levenshtein Distance.*

1. Pendahuluan

Kebanyakan *e-commerce* yang ada saat ini seperti *Lazada, MatahariMall, Tokopedia,* dan lainnya, masih menggunakan *search engine* yang sedikit saja memiliki kesalahan penulisan

dalam pencarian barang akan membuat hasil yang diinginkan tidak keluar sesuai dengan keinginan user. Metode *Levenhstein Distance* merupakan suatu pengukuran (*metrik*) yang dihasilkan melalui perhitungan jumlah perbedaan yang terdapat pada dua *string.* Diharapkan dengan menggunakan metode ini pada *search engine* di *web e-commerce* ini akan membuat jumlah akurasi dan kepuasan user bertambah.

Beberapa *search engine* yang tersedia saat ini masih menggunakan metode *keyword matching* untuk melakukan pencarian. Metode ini berkerja dengan cara melakukan pencarian pada setiap dokumen untuk kata yang sesuai dengan *keyword* yang diberikan kemudian menampilkan dokumen-dokumen yang sesuai tanpa mempedulikan dokumen lainnya yang tidak terdapat *keyword* yang diinginkan. Dengan metode ini, setiap dokumen tidak saling berhubungan satu dengan lainnya selama proses pencarian karena proses pencarian hanya terbatas pada isi dokumen per dokumen. Metode *Latent Semantic Indexing (LSI)* dapat digunakan dalam proses *indexing* suatu dokumen dalam *database*, sehingga dapat diperoleh keterkaitan antara setiap dokumen yang ada. Metode ini bekerja dengan prinsip yang cukup sederhana, dimana selain melakukan penyimpanan kata-kata ke dalam *database*, metode ini juga memeriksa keseluruhan koleksi dokumen dalam *database* untuk menentukan kemiripan antara satu dokumen dengan dokumen lainnya. LSI menganggap dokumen-dokumen yang memiliki banyak kata-kata yang sama memiliki kemiripan secara semantik dan sebaliknya dokumen-dokumen yang tidak banyak memiliki kesamaan kata-kata sebagai *semantically distant.* Ketika dilakukan proses pencarian pada LSI *database*, *search engine* memperhitungkan bobot kemiripan untuk setiap kata-kata yang merupakan isi dari koleksi dokumen dalam *database*. Nilai ini, yakni *similarity values* dari kata-kata tersebut menentukan kemiripan antara dokumen, dan dua dokumen dapat saja mirip secara semantik meskipun keduanya tidak memiliki *keyword* tertentu, sehingga pencarian tidak memerlukan keberadaan kata yang sama untuk mendapatkan hasil yang berguna. Dengan demikian kelebihan dari penggunaan LSI *database* memungkinkan hasil pencarian berupa dokumen-dokumen yang relevan meskipun tidak terdapat keyword sama sekali [1].

Berdasarkan permasalahan diatas, peneliti mencoba untuk menyelesaikan permasalahan tersebut dan membuat kesimpulan untuk membuat dan meneliti Perencanaan *Search engine E-commerce* dengan Metode *Latent Semantic Indexing* Berbasis Multiplatform dimana untuk proses autocorrect menggunakan Metode *Levenshtein* dan perangkingan *output* pada *search engine* akan menggunakan metode *Latent Semantic Indexing*.

2. Tinjauan Pustaka

Ada beberapa tinjauan pustaka yang digunakan oleh penulis sebagai referensi dalam jurnal ini. Penelitian pertama berjudul *Perspectives of Semantic Web in E-Commerce*. Dalam penelitian ini membahas arsitektur semantik untuk *e-commerce* menggunakan bahasa ontologi seperti RDF [2]. Di penelitian ini digunakan *Jena Framework* untuk membangun ontologi seperti RDF. Kesimpulannya penelitian ini memperkenalkan sebuah aplikasi *e-commerce* berbasis *web semantic* yang cocok untuk mendapatkan data tanpa adanya data yang tidak konsisten.

Penelitian kedua berjudul *Mining Text using Levenshtein Distance in Hierarchical Clustering* dimana pada penelitian ini berisi tentang teks *mining* yang menjadi subjek yang sangat diperhatikan banyak peneliti data tetapi adanya kesalahan pengejaan dan kesalahan tata bahasa dapat membuat data teks melihatnya sebagai sebuah *noise* dan menyebabkan hilangnya informasi yang penting yang seharusnya bisa didapatkan dari input teks [3]. Penelitian ini bertujuan untuk menyajikan algoritma yang efektif memperbaiki jumlah kesalahan dalam teks. Dokumen teks yang digunakan dalam penelitian ini diambil dari Wikipedia dan beberapa kesalahan yang sudah dilakukan pada penelitian untuk algoritma sebelumnya. Untuk algoritma koreksi pengejaan yang digunakan dalam penelitian ini adalah algoritma *Levenshtein Distance* yang dapat menghitung jumlah jarak minimum antara dua kata berbeda.

Penelitian keempat berjudul *Investigation of Latent Semantic Analysis for Clustering of Czech News Articles* ditulis oleh Michal Rott dan Petr Cerva tahun 2014. Penelitian ini mempelajari penggunaan *Latent Semantic Analysis* (LSA) untuk *clustering* otomatis artikel berita Ceko. Penelitian ini menunjukkan bahwa LSA mampu menghasilkan hasil yang baik di permasalahan ini karena memungkinkan untuk mengurangi masalah sinonim. Ini adalah faktor yang sangat

penting terutama untuk Ceko, yang termasuk dalam kelompok yang sangat inflektif dan bahasa yang penuh dengan morfologi [4]. Evaluasi eksperimental skema clustering dan penyelidikan LSA dilakukan pada *query* dan berbasis kategori tes set. Hasil yang diperoleh menunjukkan bahwa sistem otomatis menghasilkan nilai indeks *Rand* yang benar-benar lebih rendah 20% dibandingkan akurasi anotasi *cluster* manusia. Penelitian ini juga menunjukkan kemiripan yang metrik harus digunakan untuk *cluster* penggabungan dan efek pengurangan dimensi akurasi *clustering.*

3. Metodologi Penelitian

3.1 Identifikasi Masalah

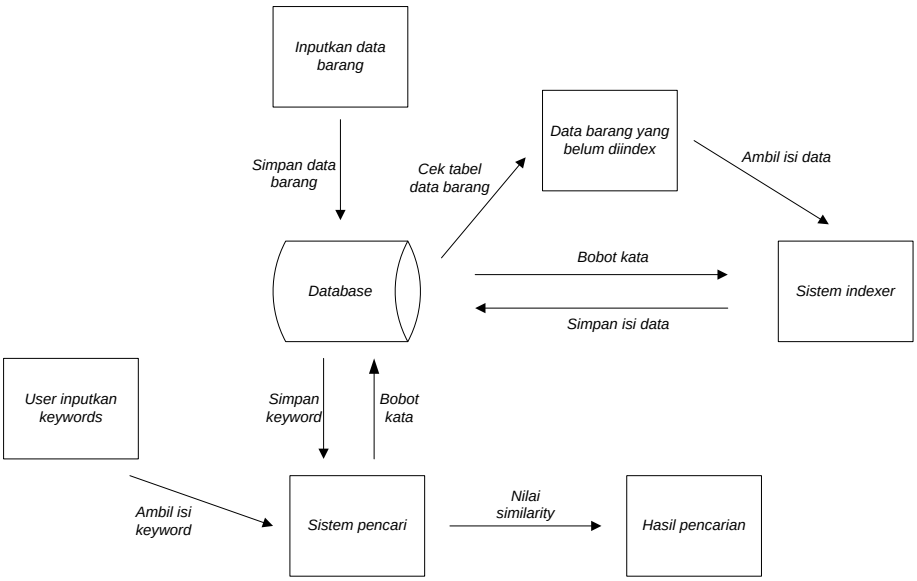
Masalah-masalah yang dihadapi dalam perencanaan *search engine e-commerce* dengan metode *Latent Semantic Indexing* ini adalah:

a. Bagaimana merencanakan sebuah aplikasi *e-commerce* yang menggunakan *search engine* dengan metode *Latent Semantic Indexing* yang mampu memberikan hasil yang lebih relevan.
b. Bagaimana membangun sebuah *search engine* yang dapat berjalan di berbagai platform (*multi platform*).
c. Bagaimana hasil nilai *usability* dari setiap aspek dalam *search engine* web *e-commerce* ini.

4. Teori Penunjang

4.1. Perancangan Aplikasi *Search Engine Web Semantic*

Perancangan *search engine* sistem ini dibagi atas sistem *indexing* dan *searching*. Kedua sistem ini berjalan secara terpisah. Sistem *searching* dapat digunakan oleh setiap *user web* untuk mencari produk yang diinginkan sedangkan sistem *indexing* hanya dapat diakses oleh *admin* sistem untuk menambah koleksi produk yang disimpan dan memprosesnya agar dapat ditampilkan sebagai hasil pencarian. Diagram kedua sistem tersebut adalah sebagai berikut:



Gambar 1. Diagram Proses Indexing dan Searching

Proses *indexing* berawal dari input yang diberikan user yaitu berupa *file* yang akan ditambahkan kemudian setelah itu dilakukan segmentasi untuk mendapatkan keterangan tentang benda tersebut seperti nama, merk, kapasitas, tipe, dan lainnya secara otomatis. Kemudian semua input disimpan dalam *database*. *Input* yang dimasukkan tidak langsung muncul sebagai hasil pencarian karena belum dilakukan proses perhitungan *latent semantic structure* dari file tersebut. Prosesnya dilakukan secara terpisah untuk melakukan perhitungan pada beberapa *file* sekaligus.

Untuk proses *indexing* terdiri dari dua proses yaitu *input file*, tahap penyimpanan data *file* ke *database* dan proses perhitungan untuk menghitung *latent semantic structure* dari *file-file* yang telah diinputkan. Pada tahap indexing, sistem akan mencari dokumen dalam *database* yang belum di*index* (status = 0). Kemudian sistem membuka isi file yang tersimpan di *database* dan memecah isinya menjadi kata, kemudian setelah melalui proses *preprocessing*maka akan didapatkan *content words* yaitu kata-kata yang dapat menjelaskan isi dokumen. *Content words* yang didapat akan disimpan ke dalam tabel kata, kemudian dilakukan perhitungan bobot kata tersebut. Selanjutnya sistem akan men-*generate term document matrix* dari tabel kata untuk disimpan dalam file matriks. Perhitungan SVD terhadap matriks tersebut akan dilakukan dengan menggunakan program SVD *calculator*, yang akan menghasilkan 3 matriks hasil dekomposisi yang masing-masing akan disimpan dalam file matriks_u, matriks_v, dan *singularvalue*. Nilai-nilai dalam matriks tersebut (kecuali *singularvalue)* akan disimpan dalam *database* sebagai tabel nilai_u dan nilai_v.

Proses *search* diawali dengan menginputkan *keywords* yang akan dicari oleh *user. Keywords* dapat terdiri atas beberapa kata. Pada metode *latent semantic indexing, keywords* diperlakukan seperti sebuah dokumen, maka terhadap *keywords* dilakukan proses *preprocessing* dan perhitungan bobot tiap kata yang ada di dalamnya. Perhitungan vektor koordinat dari *keywords* dilakukan dengan menggunakan nila-nilai dari tabel nilai_u dan file *singularvalue* kemudian untuk perhitungan *similarity keyword* dengan dokumen, digunakan nilai-nilai pada tabel nilai_v yang merupakan vektor koordinat dari tiap-tiap dokumen dan digunakan metode *cosine similarity* untuk mendapatkan nilai *similarity* antara vektor *keywords* dengan dokumen

4.2. *Multiplatform*

Platform bisa diartikan sebagai tipe *processor* (CPU) atau *hardware* lainnya yang memberi sistem operasi atau aplikasi jalan. Dalam komputasi, *multi platform* adalah *software* komputer yang dapat dijalankan di berbagai *platform* komputasi. *Software multi platform* dibagi mejadi dua tipe; yang pertama membutuhkan bangunan sendiri atau kompilasi dari tiap *platform* yang mendukung, dan yang kedua dapat secara langsung dijalankan pada setiap *platform* tanpa persiapan khusus, contohnya *software* ditulis dalam bahasa *interpreted* atau *pre-compiled portable bytecode* [5].

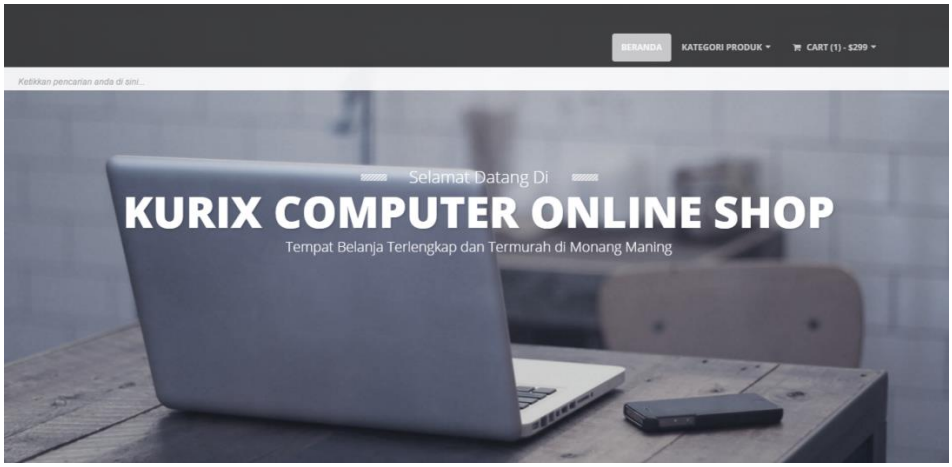
4.3. SUMI *(Software Usability Measurement Inventory)*

SUMI digunakan untuk mengetahui nilai kelima aspek *usability* dalam suatu perangkat lunak yang dikembangkan. SUMI menyediakan metode pengujian yang valid dan dapat diandalkan untuk membandingkan produk sejenis serta memberikan informasi diagnostik untuk perkembangan aplikasi ke depan. SUMI menyediakan cara yang objektif dalam menilai kepuasan pengguna dalam menggunakan perangkat lunak dimana terdapat 50 pertanyaan untuk responden jawab dengan tiga pilihan yaitu (setuju, ragu-ragu, tidak setuju). Setelah partisipan menyelesaikan menjawab kuisioner, jawabannya kemudian akan dinilai menggunakan SUMISCO yang kemudian membandingkan skor tersebut dengan skor pada basis data standarisasi (*standardization database*) dimana nilai *mean* dari *standardization database* adalah 50 dengan nilai standar *deviation* 10. *Standardization database* sendiri dikembangkan dari berbagai produk komersial yang telah sukses [6].

5. Hasil dan Pembahasan

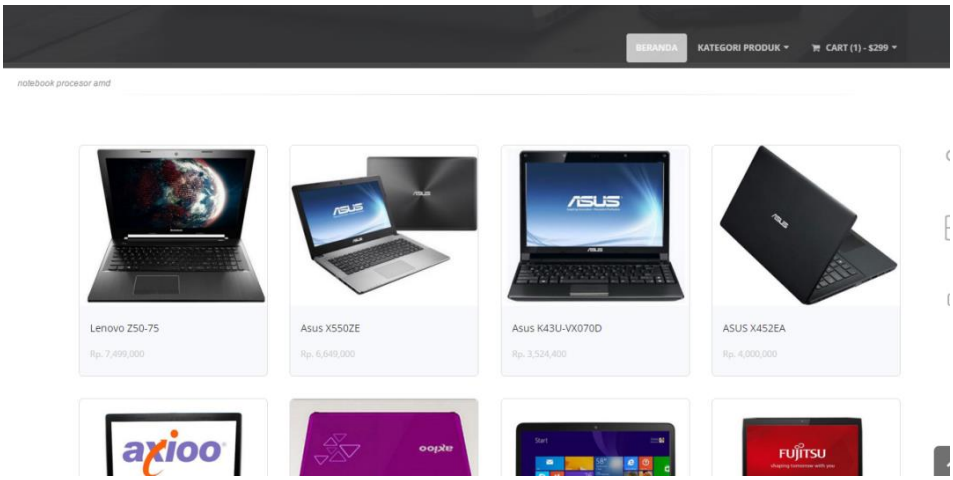
5.1. Implementasi Sistem

Tampilan yang akan terlihat pada *browser* pertama kali ketika alamat sistem dibuka yaitu halaman tampilan utama. Tampilan halaman utama ini di desain sederhana dengan warna utama adalah hitam dan biru muda. Pada header terdapat nama situs yaitu “Kurix Komputer Online Shop” dengan background gambar laptop dan handphone. Diatasnya terdapat link utama dalam *web* yaitu link beranda*,*dan Kategori produk.Jika kursor diarahkan ke tombol beranda maka halaman akan menuju ke beranda atau halaman utama.



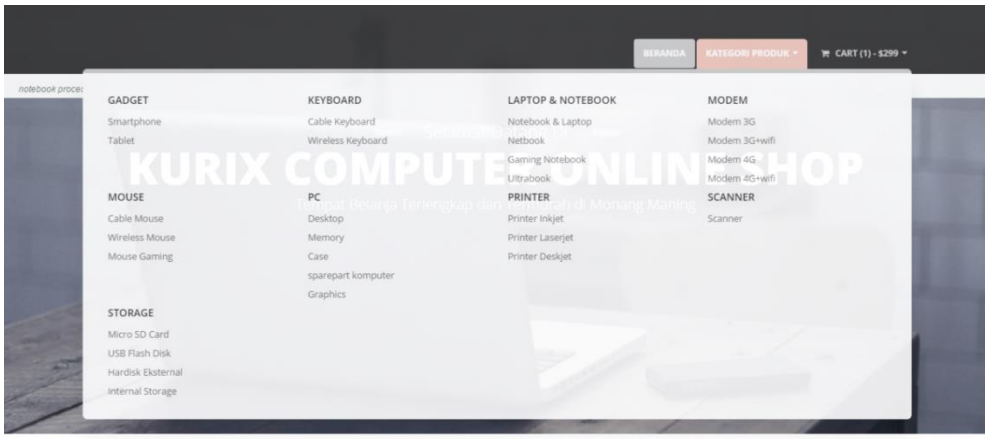
Gambar 2. Tampilan Header

Di bawah header terdapat *text box* untuk melakukan input pencarian. Di bawah *text box* terdapat pilihan semua produk yang ada di *database.*



Gambar 3. Tampilan Halaman Utama

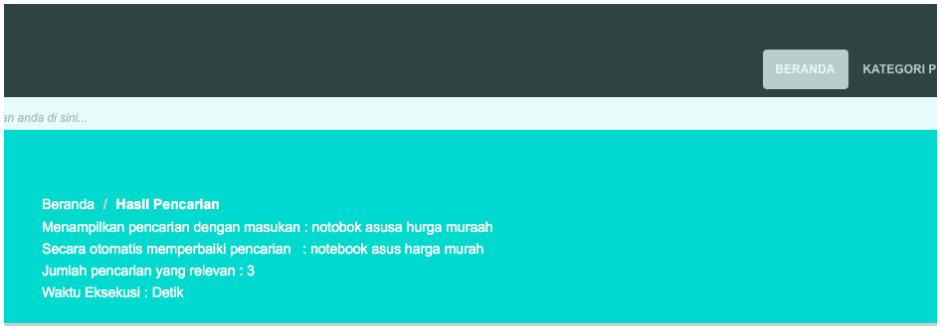
Jika kursor diarahkan pada tombol kategori produk maka akan muncul pilihan kategori produk yang ada di web ini seperti pada Gambar 4. Jika dklik pada salah satu pilihan kategori maka akan muncul barang yang termasuk pada kategori tersebut.



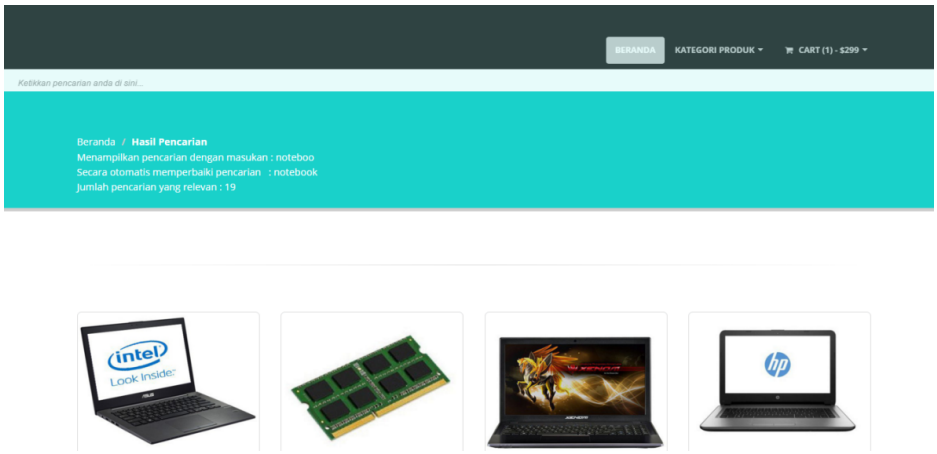
Gambar 4. Tampilan kategori produk

5.2. Implementasi Metode *Levenshtein*

Metode *Levenshtein* disebut juga sebagai *edit distance* merupakan metode pengukuran (metrik) yang dihasilkan melalui perhitungan jumlah perbedaan yang terdapat pada dua *string.* Pada web *e-commerce* "Kurix Computer Online Shop" untuk kolom pencarian sudah menggunakan metode *Levenshtein* untuk *automatic correction* pada input. Operasi yang dapat dilakukan pada adalah penambahan, pengubahan dan pengurangan karakter. Pada aplikasi web search engine ini, penggunaan levenshtein sebagai perbaikan kata dilakukan perkata ataupun perkalimat. Gambar 5 dengan contoh kalimat "notobok asusa hurga muraah" setelah diperbaiki menjadi "laptop asus harga murah".



Gambar 5. Operasi *Levenhstein*



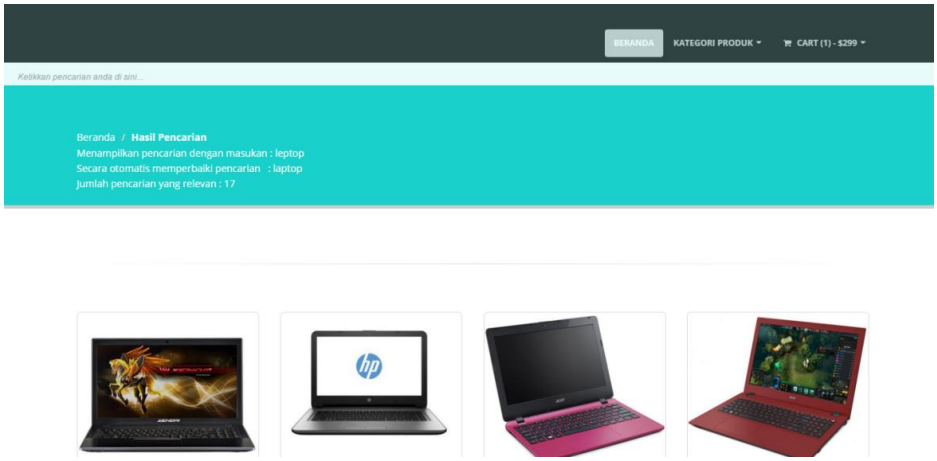
Gambar 6. Tampilan penambahan karakter

5.2.1. Operasi penambahan karakter

Contoh operasi penambahan karakter misalkan pada input kata "noteboo" kata yang seharusnya adalah "notebook" tetapi user melakukan kesalahan input sehingga input kekurangan satu karakter pada akhir kata. Metode *levenhstein* akan melakukan *looping* pengecekan pada tiap karakter hingga akhirnya ditemukan satu kekurangan karakter pada akhir kata. Tampilan pada *web* nya seperti Gambar 6.

5.2.2. Operasi pengubahan karakter

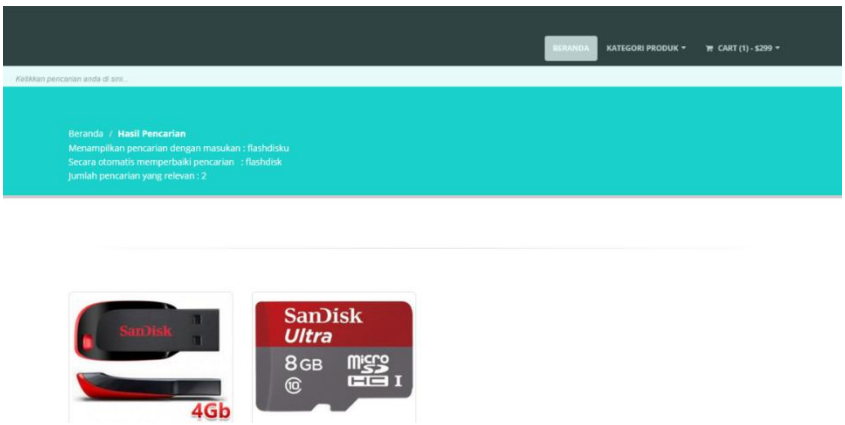
Contoh operasi pengubahan karakter misalkan pada input kata "leptop" yang seharusnya "laptop" tetapi terdapat satu karakter yang berbeda yaitu "e" dan "a" maka metode *levenhstein* akan melakukan looping pengecekan pada karakterdan mencari kata di *database* yang terdekat kemiripannya dengan kata yang dimaksud oleh user. Setelah ditemukan maka kata "leptop" akan otomatis berubah menjadi "laptop". Contoh pada tampilan seperti pada Gambar 7.



Gambar 7. Operasi pengubahan karakter

5.2.3 Operasi pengurangan karakter

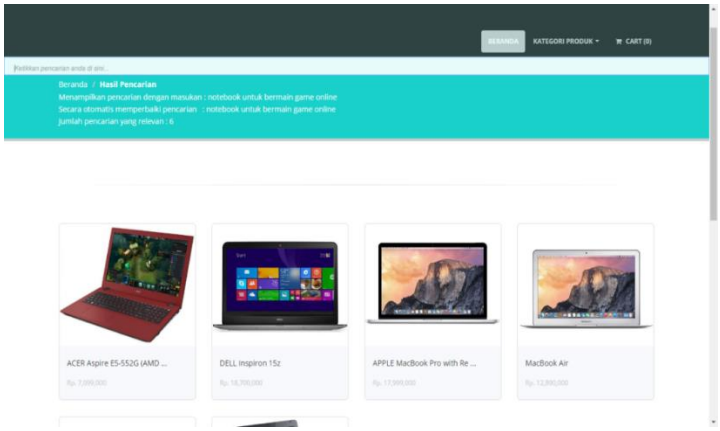
Pada input kata “flashdisku” yang seharusnya “flashdisk” terdapat satu kelebihan karakter “u” pada akhir kata. Metode *levenshtein* akan melakukan pengecekan pada *database* kata yang terdekat kemiripannya dengan *input.* Setelah dilakukan *looping* akan ditemukan kelebihan karakter dibelakang kata input yang akhirnya akan dikurangi sehingga input yang keluar sesuai. Tampilan pengurangan karakter pada web seperti pada Gambar 8.



Gambar 8. Tampilan pengurangan/penghapusan karakter

5.3 Implementasi LSI (Latent Semantic Indexing)

Latent Semantic Indexing adalah metode *indexing* dan *retrieval* yang mengunakan teknik matematika bernama *Singular Value Decomposition* (SVD) untuk mengidentifikasi pola dalam hubungan antara *term* dan konsep terkandung dalam kumpulan teks tidak terstruktur. LSI didasarkan pada prinsip bahwa kata-kata yang digunakan dalam konteks yang sama cenderung memiliki makna yang sama. Fitur utama dari LSI adalah kemampuannya untuk mengekstrak isi konseptual dari tubuh teks dengan mendirikan asosiasi antara hal yang terjadi dalam konteks yang serupa. Contoh ketika user memasukkan input “notebook untuk bermain game online”, maka outputnya seperti Gambar 9.

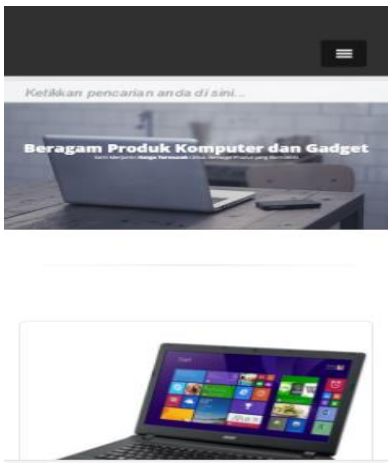


Gambar 9. Hasil searching dengan LSI

Bisa dilihat hasil output dari searching “notebook untuk bermain game online” yang pertama adalah laptop ACER Aspire E5-552G (AMD A10-8700P). N*otebook* ACER tersebut dapat digunakan untuk bermain game *online* maupun *offline*. Jadi, hasil pencarian yang dilakukan akurat dan sesuai dengan input dari pengguna.

5.4 Implementasi Multiplatform

Percobaan ini akan dilakukan dengan mengakses melalui *web Mobile*. Disini dilakukan percobaan sederhana untuk melihat tampilannya jika diakses melalui mobile dengan *platform Android*. Contoh *handphone* yang digunakan adalah *Nexus 5.* Membangun sebuah *web multiplatform* dibutuhkan aplikasi *CSS 3, HTML 5, Javascript* dan *bootstrap.* CSS 3 merupakan versi terbaru dari CSS yang ada. CSS merupakan kepanjangan dari *Cascading Style Sheets* merupakan bahasa yang menjelaskan gaya dari sebuah dokumen HTML. CSS menjelaskan bagaimana elemen HTML harus ditampilkan.HTML 5 merupakan versi terbaru dari HTML. HTML merupakan singkatan dari *Hyper Text Markup Language* merupakan bahasa markah standar untuk membuat halaman *web. Javascript* merupakan bahasa pemrograman dinamis tingkat tinggi yang popular di internet dan dapat bekerja di banyak *browser* seperti *Firefox, Opera, Chrome*, dan *Netscape.* Kode *javascript* disisipkan ke web dengan tag SCRIPT. Keempat teknologi tersebut digabungkan akan membentuk *Responsive Web Design. Responsive Web Design* dapat membuat halaman web terlihat bagus dalam semua perangkat (*desktop, tablet, phone*).



Gambar 10. Tampilan *Banner* pada *Mobile (Android*)

5.5 Implementasi SUMI

Jumlah sampel untuk pengujian aplikasi yang melibatkan uji *usability* dan uji statistik adalah sama. Sampel dipilih berdasarkan kriteria sebelumnya. Dari populasi mahasiswa Teknologi Informasi Universitas Udayana Bali yang memenuhi persyaratan diatas yaitu 35 mahasiswa.

Dari 35 populasi dicari sampel berdasarkan tabel *Krejcie-Morgan* yaitu dimana jika terdapat 35 populasi maka yang bisa dijadikan sampel sejumlah 32. Sekitar 10% dari jumlah sampel akan ditambahkan untuk menjadi cadangan sampel yang berarti akan ditambahkan sekitar 2 sampel, sehingga sampel akhir akan menjadi sebanyak 34 sampel. Asumsi tingkat kehandalan yang akan dicapai adalah 95% dengan jumlah galat maksimum 5%. Adapun sumber data yang akan menjadi sampel pengujian memiliki syarat:

a. Dapat mengoperasikan komputer
b. Tidak asing dengan penggunaan web dan *search engine*
c. Pernah melakukan menggunakan layanan *e-commerce* seperti Bhinneka, Lazada, Tokopedia dan lainnya.
d. Sudah mengikuti mata kuliah Pemrograman Internet

Kuesioner SUMI memiliki 50 pertanyaan yang setiap pertanyaan terdiri dari lima kategori. Kategori tersebut menggambarkan dimensi pertimbangan pengguna saat menggambarkan usability perangkat lunak. Lima kategori pernyataan tersebut adalah *Efficiency, Affect, Helpfulness, Control, Learnability.* Tiap-tiap pertanyaan dari kuisioner tersebut bertujuan untuk menunjukkan tingkat *usability* menurut penerimaan user. Skor dari setiap tanggapan diberi bobot yang berbeda. Pernyataan yang mengarah ke positif terhadap sistem diberi nilai 4,2,0 untuk tanggapan setuju, tidak tahu dan tidak setuju. Contoh pernyataan positif: *I would recommend this software to my colleagues*. Pernyataan yang mengarah negatif akan diberi nilai sebaliknya yaitu 0,2,4 untuk tanggapan setuju, tidak tahu, dan tidak setuju. Contoh pernyataan*: This software responds too slowly to inputs* [7]. Isi kuesioner SUMI yang lengkap dapat diakses di http://sumi.uxp.ie/en/index.php

Tabel 1. Tabel nilai hasil evaluasi SUMI

No	Atribut	Nilai
1	*Efficiency*	85,75%
2	*Affect*	77,95%
3	*Helpfulness*	76,76%
4	*Control*	82,34%
5	*Learn*	85%

Tabel 1 menunjukkan tabel skor hasil evaluasi SUMI berdasarkan aspeknya. Dilihat dari nilainya, setiap aspek sudah cukup baik karena mencapai nilai diatas 50%. Aspek dengan nilai terendah adalah *Helpfulness* dengan nilai 76,76%. Aspek tertinggi adalah *Efficiency* dengan nilai 85,75%. Berdasarkan nilai hasil evaluasi SUMI untuk setiap aspeknya, bisa disimpulkan bahwa web ini sudah memenuhi kelima aspek *usability* dari SUMI.

5.6 Uji Akurasi

Database kata dasar akan menggunakan sekitar 28.000 kata dasar bahasa Indonesia terdiri dari kata sifat, benda, dan kata kerja. *Database* kata dasar ini sudah berdasarkan dari Kamus Besar Bahasa Indonesia. Untuk *database* barang akan diisi dengan *item*/barang yang akan dijualkan di *web e-commerce* toko komputer ini seperti *notebook, ultrabook, netbook, smartphone, tablet*, dan *notebook gaming.* Data latih yang digunakan dalam penelitian ini berjumlah 100 atau lebih karena semakin banyak data latih akan semakin mempengaruhi dari hasil pencarian yang dilakukan. Tingkat akurasi yang ingin dicapai diharapkan bisa mencapai hingga 95%.

Uji akurasi dilakukan untuk menguji keakuratan dari *search engine* di *web e-commerce* toko komputer yang telah dibuat. Uji akurasi menggunakan ukuran *Precision, Recall*, dan *Accuracy* yang sudah sejak dulu digunakan sebagai penghitung relevansi dalam pengembangan system IR [8]. Uji akurasi ini menggunakan data ujicoba yaitu tabel barang dengan isi 100 barang, 253 *terms*, dan tabel *query* yang akan diujicobakan sebanyak 34 *query* dimana setiap responden yang berjumlah 34 orang memasukkan satu keyword yang diinginkan. Input yang dimasukkan adalah barang atau item tentang laptop atau smartphone. Uji akurasi pada sebuah *retrieval system* biasanya menggunakan perhitungan *precision* dan *recall.*

Rumus untuk perhitungan precision adalah:

Precision: $\frac{jumlah\ output\ benar\ menurut\ user}{jumlah\ ouput\ benar\ hasil\ eksekusi\ komputer}$ x 100% (1)

Rumus perhitungan recall adalah:

Recall: $\frac{jumlah\ output\ benar\ menurut\ user}{jumlah\ yang\ benar\ ada\ di\ database}$ x 100% (2)

Rumus perhitungan untuk *accuracy* adalah:

Accuracy: $\frac{(jumlah\ produk\ benar + jumlah\ produk\ salah\ di\ database) yang\ dipisahkan\ dengan\ benar}{total\ produk\ di\ database}$ x 100% (3)

Hasil *precision* untuk uji akurasi ini adalah sebesar 78,97%, *recall* sebesar 56,02%, dan *accuracy* sebesar 96,7%.

6. Kesimpulan

Membangun sebuah *web multiplatform* dibutuhkan aplikasi *CSS 3, HTML 5, Javascript* dan *bootstrap*. Keempat teknologi tersebut digabungkan akan membentuk *Responsive Web Design. Responsive Web Design* dapat membuat halaman web terlihat bagus dalam semua perangkat (*desktop, tablet, phone*). Hasil nilai *usability* untuk *search engine* LSI ini mendapatkan nilai 85,75% untuk aspek *Efficiency*, 77,95% untuk aspek *Affect*, 76,76% untuk aspek *Helpfulness*, 82,34% untuk *Control*, dan 85% untuk *Learn*. Hasil di setiap aspek sudah cukup baik karena sudah melebihi 50%. Hasil perencanaan *search engine* di *web e-commerce* dengan *metode Latent Semantic Indexing* ini dapat memperoleh nilai akurasi yang baik yaitu sekitar 96,7%. Untuk pengembangan aplikasi selanjutnya dapat dilakukan penambahan dari *processor*, RAM, dan HDD komputer sehingga untuk proses perhitungan matriksnya bisa berjalan lebih cepat dan penggunaan bahasa pemrograman PHP bisa diganti dengan bahasa pemrograman berbasis desktop seperti C++, C#, atau Phyton untuk pemrosesan SVD sehingga diharapkan bisa membuat waktu eksekusi lebih cepat.

Daftar Pustaka

[1] O. N. Oyelade, S. B. Junaidu, and A. A. Obiniyi, “Semantic Web Framework for E-Commerce Based on OWL,” vol. 11, no. 3, pp. 145–154, 2014.
[2] B. VijayaLakshmi, A. GauthamiLatha, Y. Srinivas, and K. Rajesh, “Perspectives of Semantic Web in E-Commerce,” *International Journal of Computer Application*, vol. 25, no. 10, pp. 52–56, 2011.
[3] S. Kaur and P. Kiranjyoti, “Mining Text using Levenshtein Distance in Hierarchical Clusteing,” vol. 2, no. 1, pp. 92–97, 2015.
[4] M. Rott and P. Cerva, “Investigation of Latent Semantic Analysis for Clustering of Czech News Articles,” in *Database and Expert Systems Applications (DEXA)*, 2014.
[5] P. Smutný, “Mobile Development Tools and Cross-Platform Solutions,” in *Proceedings of the 2012 13th International Carpathian Control Conference, ICCC 2012*, 2012.
[6] T. Arh and B. J. Blažič, “A Case Study of Usability Testing - the SUMI Evaluation Approach of the EducaNext Portal,” *WSEAS Trans. Inf. Sci. Appl.*, 2008.
[7] Hamidah, “Pengembangan Situs PTN Menggunakan Usability Engineering dan Evaluasi Usability dengan Koesioner SUMI,” p. 29, 2013.
[8] S. A. Alvarez, “An Exact Analytical Relation Among Recall, Precision, and Classification Accuracy in Information Retrieval.’Boston College,” *Boston, Tech. Rep. BCCS-02-01*, 2002.