Versi Online tersedia di :
https://jurnal.buddhidharma.ac.id/index.php/algor/index

JURNAL ALGOR

|2715-0577 (Online)| 2715-0569 (Print)

Jurnal

PERANCANGAN APLIKASI KAMUS ISTILAH KOMPUTER DENGAN METODE ALGORITMA BOYER MOORE BERBASIS ANDROID

Billy Gozali[1], Yo Ceng Giap[2] Desiyanna Lasut[3]

[1,2,3] Universitas Buddhi Dharma, Teknik Informatika, Banten, Indonesia

SUBMISSION TRACK

Recieved: Agustus, 25, 2021
Final Revision: Sept, 18, 2021
Available Online: September, 2021

KEYWORD

Istilah Komputer, Pencocokan String, *Boyer-Moore, SQLite, Android*

KORESPONDENSI

Phone: 0895-3265-24581

Email: billygozaly123@gmail.com

A B S T R A K

Kebutuhan masyarakat terhadap perkembangan teknologi informasi telah memberikan peranan penting bagi kegiatan manusia. Perkembangan teknologi informasi itupun sudah dapat dirasakan pemanfaatannya dalam kehidupan manusia, khususnya komputer dan *smartphone* telah mempengaruhi perkembangannya dan kini telah diminati oleh kehidupan masyarakat. Proses pencarian pun sangatlah penting untuk melakukan proses pencarian pada aplikasi kamus, walaupun saat melakukan proses pencarian pun bisa saja lambat. Untuk mempermudah proses pencarian, diperlukan suatu *algoritma* yang nantinya akan melakukan suatu pencarian suata kata. Metode yang digunakan oleh peneliti adalah *algoritma boyer-moore* merupakan metode *algoritma* pencarian dengan membandingkan tipe data *string* pada proses pencarian *pattern* dari penyimpanan basis data. Penelitian ini berfokus terhadap implementasi pada sebuah aplikasi kamus yang nantinya akan membantu pengguna dalam menerjemahkan istilah komputer yang dapat disimpan secara *database* dalam *smartphone*. Aplikasi ini dijalankan pada *platform* android secara *offline* yang dapat diakses kapanpun dan dimanapun. Untuk membuat aplikasi ini peneliti menggunakan *android studio* dengan *database SQLite* dan analisis berorientasi objek melalui sistem *flowchart*. Aplikasi ini nantinya dapat memberikan tampilan hasil pencarian sebuah istilah berupa media terhadap pencarian istilah komputer pada ponsel *smartphone*. Berdasarkan hasil pengujian dari 17 responden, aplikasi kamus isitlah komputer memperoleh tingkat presentase sebesar 91,29% yang artinya aplikasi ini tergolong dalam kategori baik. Sehingga dengan adanya aplikasi kamus istilah komputer diharapkan dapat membantu bagi khususnya mahasiswa ilmu komputer ingin memahami sebuah istilah-istilah komputer yang biasa sering ditemui.

PENDAHULUAN

Latar Belakang

Kebutuhan masyarakat terhadap perkembangan teknologi informasi telah memberikan peranan penting bagi kegiatan manusia. Perangkat digitalisasi otomatis telah mengambil peran dalam perubahan kegiatan manusia diberbagai aktivitas teknologi. Perkembangan teknologi informasi itupun sudah dapat dirasakan pemanfaatannya dalam kehidupan manusia, khususnya komputer dan *smartphone* telah dampak mempengaruhi perkembangannya dan kini telah diminati oleh kehidupan masyarakat. Saat ini penggunaan komputer tidak hanya dilakukan oleh para pengembang saja tetapi dapat juga dilakukan oleh seorang pengguna maupun orang awam sekalipun[1]. Namun bertepatan dengan perkembangan teknologi, istilah-istilah yang digunakan pun semakin bertambah dan berkembang yang terkadang kurang dimengerti oleh pengguna komputer pada umumnya.

Peningkatan teknologi saat kini telah memberikan dampak pengaruh sangat besar bagi dunia digital teknologi informasi[2]. Munculnya beragam aplikasi yang mengedepankan peningkatan kinerja suatu proses kerja, baik yang bersifat aplikasi *desktop*, *web*, adanya perkembangan teknologi informasi terdiri dari aplikasi-aplikasi yang dapat dijalankan dalam *mobile device* maupun *smartphone* [3].

Pemilihan *mobile device* selain untuk pengembangan aplikasi, dapat dengan mudah dalam pengoperasiannya, fleksibel dan dapat dengan mudah di bawa saat perlu digunakan. Berbagai jenis sistem operasi banyak bermunculan baik dalam komputer maupun pada perangkat *mobile device*.

Buku kamus istilah komputer beredar luas di pasaran namun buku terlebih jauh meyulitkan. Belum lagi juga cara untuk mencari informasinya cukup lama karena klien perlu membaca setiap huruf dari halaman buku untuk mencari arti istilah komputer secara manual. Namun tidak banyak dari kita yang miliki koleksi buku tersebut bisa lengkap, selain harganya yang relatif mahal, biasanya ukuran buku cukup besar dan tebal sehingga merasakan cukup sulit untuk dibawa kemana-mana, namun kekurangan lainnya adalah buku untuk tetap mewaspadai kemajuan inovasi data yang berkembang pesat hingga saat ini.

Algoritma pencarian dianggap memberikan hasil paling akurasi dalam melakukan pencarian, yaitu *algoritma* yang melakukan pergerakan proses tipe data *string*. Salah satu metode peneliti yang digunakan adalah *algoritma boyer-moore* merupakan metode *algoritma* pencarian dengan membandingkan tipe data string pada proses pencarian *pattern* dari penyimpanan basis data[4].

I. METHODS

1.1 Algoritma *Boyer-Moore*

Algoritma *boyer-moore* merupakan suatu algoritma pergeseran proses pencarian tipe data string, yang pertama kali didistribusikan pada tahun 1977. Penggunaannya banyak digunakan karena banyaknya penerapan pada saat melakukan proses terhadap penerapan string (*multiple pattern*). Hal yang dapat diketahui pada *algoritma boyer-moore* sebenarnya dengan sederhananya sama *algoritma brute force* menggunakan penggunaan isi tabel perpindahan pattern yang nantinya akan dilakukan ke kanan saat melakukan pergeseran antara kecocokan dan ketidakcocokan karakter[5].

Pada algoritma ini pencocokkan dilakukan saat mengubah pergeseran arah pencocokkan, yaitu mencocokkan karakter dimulai dari kanan (karakter terakhir pada *pattern*) hingga ke kiri. Sementara *pattern* tetap bergeser kearah kanan.

Apabila menemui ketidakcocokkan, maka pattern akan bergeser kearah kanan dan memungkinkan untuk melakukan kecocokkan antara karakter pada teks yang sedang dilakukan proses pengecekkan.

Secara efisien, cara yang diperoleh dari penggunaan algoritma *boyer-moore* terdapat langkah-langkah saat melakukan pencocokan string, yaitu: [6].

1. Algoritma memulai pencocokan

string antara pattern dan awal text.

2. Algoritma nanti akan melakukan pencocokan karakter dari arah kanan ke kiri *pattern* dengan karakter *teks* yang beruntun, sampai proses kondisinya tersebut dapat dipenuhi.
3. Jika dalam proses karakter antara *pattern* dan *text* yang dibandingan tidak cocok, maka pergeseran dilakukan dengan memilih salah satu nilai pergeseran dari tabel *string* yang memiliki nilai pergeseran tertinggi.
4. Apabila semua karakter dan *pattern* sesuai, maka *algoritma* akan memberikan proses sebuah posisi dengan persyaratan istilah komputer.
5. Algoritma akan melakukan pergeseran pattern dengan memproses sebuah nilai antara pergeseran *good-suffix* dan pergeseran *bad-character*, kemudian mengulanginya lagi sampai pencocokan istilah tersebut dapat ditemukan.

1.2 *Black Box Testing*

Black box testing diperlukan pada saat melakukan pengetesan daleman struktur dari sistem ataupun komponen bagian sistem yang nantinya akan diuji. *Black box testing* berpusat pada kebutuhan fungsional terhadap perangkat lunak, hal tersebut berdasarkan pada kebutuhan sistem[7]. *Black box testing* biasanya akan melakukan untuk menemukan kesesuaian pada kategori dari istilah komputer berikut, yaitu:

1. Untuk mengetahui jalannya fungsi aplikasi yang salah atau tidak benar.
2. Mengindentifikasi permasalahan pada kinerja dan *interface*
3. Kesalahan dalam mengatur proses struktur data dari akses basis data secara terpusat secara *real-time*.
4. Untuk melakukan inisialisasi pada variabel dalam menentukan nilai objek terhadap pergeseran dalam penentuan oleh *algortima boyer-moore*.

Pengetesan dalam *Black box* dirancang untuk mendapatkan responden terhadap menjawab pertanyaan dibawah ini, yaitu: [8].

1. Bagaimana validitas terhadap fungsionalitas akan diuji ?
2. Jenis penginputan hal apa yang akan memberikan sistem fungsional baik ?
3. Apakah sistem secara khusus dapat responsif terhadap nilai input tertentu ?
4. Berapakah presentase rasio dan junlah data yang dapat diterapkan oleh sistem ?
5. Apa reaksi yang terjadi saat melakukan kombinasi data pada aplikasi ?

1.3 Skala *Likert*

Dengan Skala *likert*, faktor yang akan diestimasi diubah menjadi pointer variabel. Kemudian penanda tersebut digunakan sebagai tahap awal untuk mengurutkan hal-hal instrumen yang dapat berupa pertanyaan [9].

Tanda skala ini adalah bahwa jenis jawaban atas pertanyaan menggunakan skala likert memiliki tingkatan untuk menunjukan sikap dan pendapat. Skala *likert* yang dipakai penulis dalam pembuatan angket, yaitu:

Sangat Baik (SB) = 5
Baik (B) = 4
Cukup (C) = 3
Tidak Baik (TB) = 2
Sangat Tidak Baik (STB) = 1

II. PERANCANGAN

2.1 Metode Perancangan *Algoritma Boyer -Moore*

Konstruksi yang akan dibangun adalah aplikasi kamus digital. Kamus ini akan memberikan informasi tentang makna kata

maupun contoh penggunaan dalam kalimat yang mengandung kata – kata tersebut yaitu, kamus khusus istilah komputer. Kamus yang dibangun merupakan kamus ekabahasa, yang menggunakan hanya satu bahasa saja pada saat kata dimasukan kemudian menjelaskan defenisinya, seperti halnya di bahasa Inggris. Kamus ini akan ditambahkan dengan fitur *auto complete* yang akan memudahkan pengguna dalam mengetik kata yang tepat[11]. Fitur ini menunjukan prediksi komposisi penggunaan kata tanpa harus menuliskan kata atau frasa tersebut secara lengkap pada kolom pencarian.

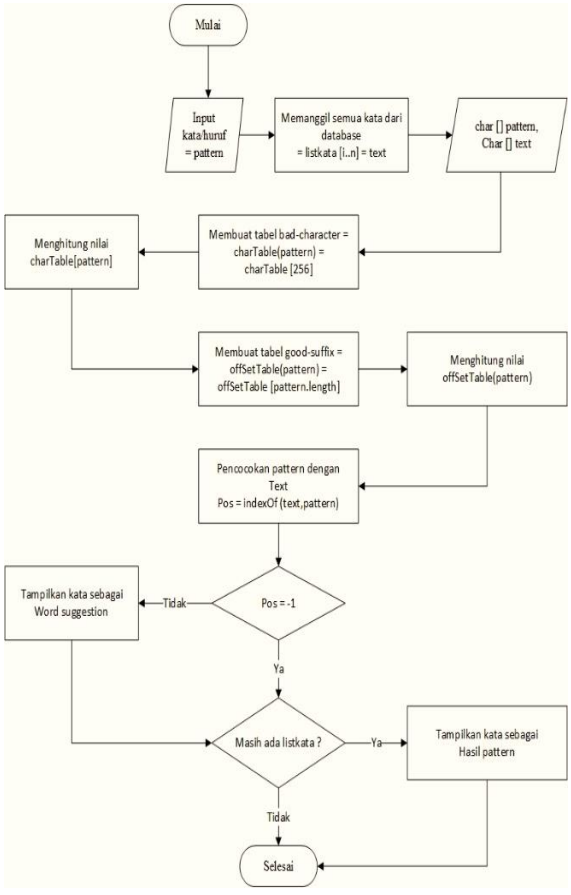
Perancangan metode ini berisi penggunaan terhadap metode *algoritma boyer-moore* untuk menunjukan pencarian *auto-complete* yang dikembangkan dengan sistem *flowchart*, berikut adalah perhitungan *algoritma boyer-moore*, dan perancangan antarmuka sistem.

Hal pertama yang dilakukan oleh pengguna adalah melakukan penginputan kata atau huruf pada pencarian. Kemudian user akan mengetik kata/huruf yang dicari, secara langsung sistem akan menganggap kata tersebut sebagai sebuah *string input* yang nantinya dicocokkan dengan *string target* yang merupakan kata yang diambil dari *database*.

Pada awalnya sistem akan mengambil setiap kata yang terkandung dalam kumpulan data atau basis data. Dari siklus ini akan memperoleh rangkaian kata yang kemudian selanjutnya akan dianggap sebagai *string* yang akan dicocokan sesuai dengan informasi. Setelah sistem mulai pencocokan diantara satu persatu dari *list* kata dengan *string input*, maka prosesnya akan dimulai dari kata pertama pada *list* pergeseran awal.

Pada proses pencocokan, setiap kata akan diubah menjadi urutan karakter. Proses yang diimplementasikan adalah menggunakan metode *algoritma boyer-moore*. Algoritma akan melakukan proses pergeseran string input sebanyak pada *string target*, dimana besar pergeseran ditentukan banyaknya nilai pergeseran sebesar diantara tabel *bad-character* sama *good-suffix*. Kemudian, variabel pos akan perkenalkan sebagai variabel yang dapat menyimpan nilai akan dilakukan saat pencocokan secara langsung.

Nilai variabel pos akan melakukan pergeseran sampai pencocokan telah selesai. Apabila *string input* ditemukan dalam string target, variabel pos akan memiliki nilai yang ditunjukan oleh indeks posisi ditemukan string input, kemudian kata yang merupakan *string target* akan ditampilkan sebagai *word suggestion*, meskipun jika string input tidak ditemukan, variabel pos bernilai -1 dan karakter tersebut tidak dapat menampilkan bila *word suggestion*. Pencocokan akan terus dilanjutkan sampai kata terakhir pada *list*. Alur proses sistem dapat ditemukan dalam bentuk alur sebagai sistem *flowchart*.



Gambar 3.2 Sistem *Flowchart*

2.5 Penggunaan Metode *Algoritma Boyer - Moore*

Algoritma *boyer-moore* melakukan pencocokan string melalui dua pendekatan adalah pendekatan *bad-character* dan pendekatan *good-suffix*. Metodologi ini nantinya akan menentukan banyaknya pergeseran yang dilakukan pada pencocokan hingga dapat ditemukan *string input* dengan *string target* dan apabila prosesnya string input belum ditemukan pada *string target*, maka string input akan lanjut tergeser sepanjang banyaknya posisi terhadap *string target*.

Berikut ini adalah tahapan-tahapan proses perhitungan *algoritma boyer-moore* :

1. **Mengambil list kata dari *database***

Setelah sistem menerima input, sistem akan mengambil semua kata pada database untuk dicocokkan dengan kata diinputkan user. Beberapa list kata yang akan dicocokkan diantaranya adalah '*drive'*, pencocokan dilakukan antara string input dengan kata pertama pada list kata yaitu '*drive'*.

2. **Menentukan nilai pergeseran dengan menggunakan pendekatan *bad-character* dan *good-suffix***

1. **Menghitung tabel *bad-character***

Langkah perhitungan *bad-character* yaitu dengan melakukan penkondisian yang mana dimulai dari karakter paling kanan string input sampai karakter paling kiri, dimulai dengan 0. Nilai *bad-character* diperoleh proses perhitungan sebagai berikut :

BmBc[y[i]] = n – 1 – i
BmBc[y[4]] = 5 – 1 – 4 = 0
BmBc[y[3]] = 5 – 1 – 3 = 1
BmBc[y[2]] = 5 – 1 – 2 = 2
BmBc[y[1]] = 5 – 1 – 1 = 3
BmBc[y[0]] = 5 – 1 – 0 = 4

Dan untuk semua karakter yang tidak muncul pada *string input*, bernilai sesuai jumlah banyaknya karakter pada string input yaitu 5.

i	0	1	2	3	4	
c	d	r	i	v	e	*
BmBc [c]	4	3	2	1	0	5

Gambar 3.3 Pergeseran *bad-character*

2. **Menghitung tabel *good-suffix***

Nilai pergeseran *good-suffix* digunakan ketika ketidakcocokan ditemukan pada karakter yang menyebabkan ketidakcocokan dilihat berdasarkan posisi karakter tersebut. Nilai setiap karakter didalam *pattern* bergantung pada adanya atau tidaknya perulangan akhiran (*suffix*) v pada *pattern*. Semakin banyaknya literasi, maka akan semakin kecil juga nilai pada pergeseran. Untuk menentukan nilai-nilai ini, pertama-tama tentukan nilai dari tabel *suffix* yang nantinya akan menunjukan dengan memberi tanda adanya perulangan akhiran.

i	0	1	2	3	4
c	d	r	i	v	e
Suff [i]	0	0	0	0	4
BmGs [i]	4	4	4	4	1

Gambar 3.4 Pergeseran *good-suffix*

Pada gambar diatas menjelaskan tahap *preprocessing*, merupakan tahapan sebelum melakukan sebuah pencarian. Proses tersebut dibentuk 2 tabel dengan menggunakan tabel *BmGs* dan tabel *BmBc*. Dari proses perhitungan dari algoritma tersebut mempertoleh nilai pergeseran antara *bad character* dan *good suffix* untuk setiap masing-masing tipe karakter dari *pattern*.

Index	0	1	2	3	4
Pattern	D	R	I	V	E
BmBc	4	3	2	1	0
BmGs	4	4	4	4	1

Gambar 3.5 Pencarian Tipe Data Karakter pada *Pattern*

3. Pencocokan String

Setelah mendapatkan posisi pergeseran *bad-character* dan *good-suffix*, maka selanjutnya akan membandingkan proses dari arah kanan ke kiri. Ukuran pergeseran ini tergantung pada pencocokan karakter terbesar terhadap pada pergeseran *bad-character* dan *good-suffix*. Setelah *string input* ditemukan pada string target, maka akan diketahui indeks posisinya. Proses pencocokan dapat dijelaskan dalam contoh, sebagai berikut:

String input : 'drive
String target : 'harddisk drive'

H	A	R	D	D	I	S	K		D	R	I	V	E
				1									
D	R	I	V	E									

BmBc[d] = 5
BmGs[4] = 1
Maka pergeserannya, max(5,1) = 5

H	A	R	D	D	I	S	K		D	R	I	V	E
										1			
						D	R	I	V	E			

BmBc[d] = 1
BmGs[4] = 1
Maka pergeserannya, max(1,1) = 1

H	A	R	D	D	I	S	K		D	R	I	V	E
											1		
							D	R	I	V	E		

BmBc[d] = 1
BmGs[4] = 1
Maka pergeserannya, max(1,1) = 1

H	A	R	D	D	I	S	K		D	R	I	V	E
												1	
								D	R	I	V	E	

BmBc[d] = 1
BmGs[4] = 1
Maka pergeserannya, max(1,1) = 1

H	A	R	D	D	I	S	K		D	R	I	V	E
									-	-	-	-	-
									D	R	I	V	E

Pada pola ini tidak perlu melakukan proses pergeseran lagi karena *pattern* sudah sampai pada indeks terakhir.

4. Menampilkan *Word Suggestion*

Setelah sistem menerima input, sistem akan mengambil semua kata pada *database* untuk dicocokkan dengan kata diinputkan user. Beberapa list kata yang akan dicocokkan diantaranya adalah '*drive'*, pencocokan dilakukan antara *string input* dengan kata pertama pada list kata yaitu '*drive'*.

2.6 Perancangan Database

Tujuan dari Perancangan database adalah untuk dapat menampung penyimpanan data istilah kamus yang akan dipanggil oleh sistem saat mencari istilah komputer pada aplikasi. Perancangan *database* pada sistem ini menggunakan *SQLite*. Berikut ini adalah tabel dan gambar yang digunakan untuk melakukan implementasi pada aplikasi.

Tabel 1 Struktur Tabel Istilah Komputer

No	Elemen Data	Akronim	Tipe	Panjang	Keterangan
1	Id	Id	Text	6	*Primary Key*
2	Kosakata	Kosa kata	Text	30	
3	Kategori	Kategori	Text	10	
4	Artikata	Arti Kata	Text	255	

III. PEMBAHASAN DAN HASIL

3.1 Tampilan Program

a. Halaman *Splash Screen*

Gambar 4.1 Tampilan halaman *Spalash Screen*

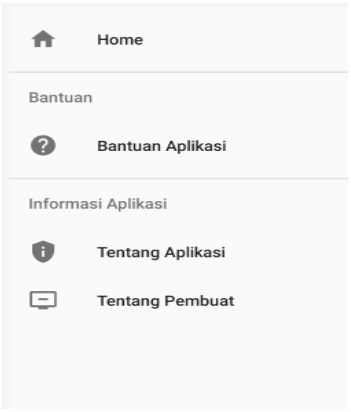
Pada saat aplikasi ini pertama kali dijalankan, maka akan muncul halaman *splash screen* sebagai halaman awal aplikasi. Jadi ketika aplikasi lain dijalankan itu menunjukkan satu halaman dalam 3 detik dengan latar belakang gambar menunjukkan nama aplikasi tersebut.

b. Halaman Menu Utama

Gambar 4.2 Tampilan Halaman Utama

Kemudian setelah itu akan tampil halaman menu utama dari aplikasi yang menampilkan *cardview* istilah komputer pada masing-masing kategori untuk pindah ke halaman-halaman jenis istilah, menu navigasi untuk menjalankan fitur-fitur disetiap masing pilihan, dan menu toolbar untuk melakukan pengaturan yang ada didalam pada aplikasi.

c. Halaman List Menu Navigasi



Gambar 4.3 Tampilan Halaman Menu Navigasi

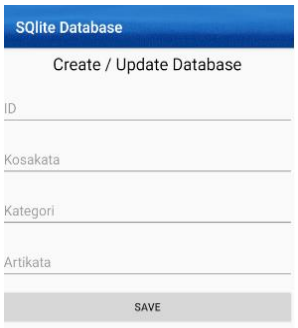
Jika pengguna memilih fitur menu navigasi, maka akan tampilan menu-menu halaman navigasi. Halaman ini menjelaskan mengenai menu yang digunakan oleh pengguna aplikasi.

d. Halaman Cari Daftar Istilah Komputer

Gambar 4.4 Tampilan Halaman Menu Navigasi

Apabila pengguna memilih salah satu jenis istilah komputer, maka akan tampil halaman daftar dari setiap kategori. Halaman ini akan menampilkan seluruh daftar istilah dari jenis istilah yang sudah di pilih.

e. Halaman Membuat *Database* Istilah Komputer



Gambar 4.5 Tampilan Membuat Tabel Basis Data

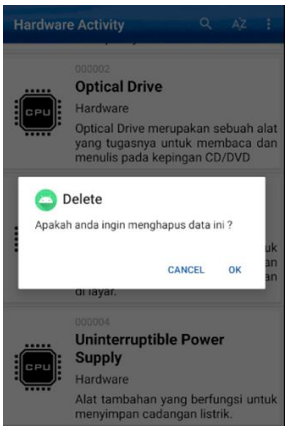
Jika pada halaman jenis istilah komputer pengguna memilih salah satu list jenis istilah komputer maka akan tampil halaman membuat dan menambahkan kata istilah sebuah kamus. Halaman ini akan memberikan list untuk menambahkan kata istilah didalam aplikasi.

f. Halaman Update Database Istilah Komputer

Gambar 4.6 Tampilan Update Tabel Basis Data

Kemudian halaman ini akan memberikan keterangan untuk mengubah atau *update* kata-kata istilah komputer didalam aplikasi.

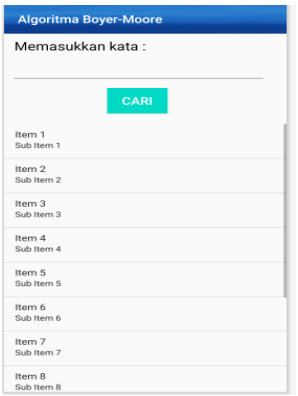
g. Halaman Delete Database Istilah Komputer



Gambar 4.7 Tampilan Delete Tabel Basis Data

Jika pengguna mengklik dengan cukup lama, maka akan tampil keterangan utnuk menghapus data didalam tabel.

h. Halaman Algoritma *Boyer-Moore*



Gambar 4.8 Tampilan Algoritma *Boyer-Moore*

Halaman menu algoritma *boyer-moore* merupakan bagian dalam pada button pencarian yang fungsinya untuk melakukan pencarian didalam database. Halaman menu algoritma *boyer-moore* akan memberikan deskripsi terhadap pencarian istilah.

i. Halaman Cari Kata Istilah Komputer

Gambar 4.9 Tampilan Pencarian kata *Boyer-Moore*

3.2 *Black Box Testing*

Pengujian *black box* dilakukan dengan menguji perangkat lunak dari segi fungsionalitas terhadap perangkat lunak. Fungsionalitas perangkat lunak yang diuji sesuai dengan *flowchart* pada tahap desain. Setiap bagian diuji sesuai dengan skenario *flowchart* terhadap pada tahap desain. Hasil pengujian *black box* adalah sebagai berikut :

Tabel 2 Hasil Fungsionalitas *Splash Screen*

Aksi Pengguna	Reaksi Sistem	Hasil Pengujian
1.) Memulai Aplikasi		
	2.) Dapat menjalankan fitur splash screen saat masuk ke menu utama istilah komputer	**Sesuai**

Tabel 3 Hasil Fungsionalitas Menu Utama

Aksi Pengguna	Reaksi Sistem	Hasil Pengujian
1.) Memulai Aplikasi		
	2.) Menampilkan menu utama aplikasi beserta menu - menu kategori dan button menu algoritma *boyer-moore*.	**Sesuai**

Tabel 4 Hasil Fungsionalitas *Navigation Drawer*

Aksi Pengguna	Reaksi Sistem	Hasil Pengujian
1.) Menggunakan menu *navigation drawer* di kamus istilah komputer		
	2.) Menampilkan halaman menu *navigation drawer* pada kamus istilah komputer	Sesuai
3.) Memilih atau klik bantuan aplikasi di menu *navigation drawer*		
	4.) Menampilkan *activity menu* di bantuan aplikasi istilah komputer	Sesuai
5.) Memilih atau klik tentang aplikasi di menu *navigation drawer*		
	6.) Menampilkan *activity menu* tentang aplikasi istilah komputer	Sesuai
7.) Memilih atau klik tentang pembuat di menu *navigation drawer*		
	8.) Menampilkan *activity menu* tentang pembuat istilah komputer	Sesuai

Tabel 5 Hasil Fungsionalitas Kategori Program

Aksi Pengguna	Reaksi Sistem	Hasil Pengujian
1.) Pengguna menjalankan menu kategori program.		
	2.) Menampilkan halaman menu-menu istilah pada kategori program.	Sesuai
3.) Pengguna mampu dapat menambahkan kata-kata istilah di menu istilah kategori program.		
	4.) Menampilkan form list data untuk menambahkan kata-kata istilah di kategori program, kemudian save.	Sesuai
5.) Pengguna dapat juga melakukan update dan perubahan data di menu kategori program.		
	6.) Menampilkan form list data untuk mengupdate kata-kata istilah di kategori program, kemudian save.	Sesuai
7.) Pengguna dapat melakukan penghapus data ataupun istilah komputer di menu kategori program.		
	8.) Sistem memberikan peringatan sebuah pernyataan untuk menghapus sebuah data istilah komputer.	Sesuai

Tabel 6 Hasil Fungsionalitas Kategori Android

Aksi Pengguna	Reaksi Sistem	Hasil Pengujian
1.) Pengguna menjalankan menu kategori *android*.		
	2.) Menampilkan halaman menu-menu istilah pada kategori *android*.	Sesuai
3.) Pengguna mampu dapat menambahkan kata-kata istilah di menu istilah kategori *android*.		
	4.) Menampilkan form list data untuk menambahkan kata-kata istilah di kategori *android*, kemudian save.	Sesuai
5.) Pengguna dapat juga melakukan update dan perubahan data di menu kategori *android*.		
	6.) Menampilkan form list data untuk mengupdate kata-kata istilah di kategori *android*, kemudian save.	Sesuai
7.) Pengguna dapat melakukan penghapus data ataupun istilah komputer di menu kategori android.		
	8.) Sistem memberikan peringatan sebuah pernyataan untuk menghapus sebuah data istilah komputer.	Sesuai

Tabel 7 Hasil Fungsionalitas Kategori *Problem*

Aksi Pengguna	Reaksi Sistem	Hasil Pengujian
1.) Pengguna menjalankan menu kategori *problem*.		
	2.) Menampilkan halaman menu-menu istilah pada kategori *problem*.	Sesuai
3.) Pengguna mampu dapat menambahkan kata-kata istilah di menu istilah kategori *problem*.		
	4.) Menampilkan form list data untuk menambahkan kata-kata istilah di kategori *problem*, kemudian save.	Sesuai
5.) Pengguna dapat juga melakukan update dan perubahan data di menu kategori *problem*.		
	6.) Menampilkan form list data untuk mengupdate kata-kata istilah di kategori *problem*, kemudian save.	Sesuai
7.) Pengguna dapat melakukan penghapus data ataupun istilah komputer di menu kategori *problem*.		
	8.) Sistem memberikan peringatan sebuah pernyataan untuk menghapus sebuah data istilah komputer.	Sesuai

Tabel 8 Hasil Fungsionalitas Algoritma *Boyer Moore*

Aksi Pengguna	Reaksi Sistem	Hasil Pengujian
1.) Menemukan menu pencarian istilah komputer		
	2.) Menampilkan halaman button pencarian algoritma *boyer-moore* pada kamus istilah komputer	Sesuai
3.) Memasukkan kata atau istilah komputer dengan menekan tombol *button* cari		
	4.) Menampilkan data pencarian apabila tersimpan pada database	Sesuai

3.3 Pengujian Kelayakan

Tahap ini merupakan tahapan penelitian terhadap suatu implementasi sistem atau aplikasi yang telah dibuat dengan menggunakan kusioner. Pengujian kelayakan ini didasarkan oleh penilaian responden. Total responden dalam pada aplikasi kamus istilah komputer ini berjumlah 17 orang dengan profesi mahasiswa jurusan komputer. Penilaian ini dilakukan dengan pengisian kuesioner setelah peneliti mempresentasikan program kepada pengguna. Skala pengukuran untuk menguji kelayakan sistem yang digunakan oleh peneliti adalah dengan menggunakan skala likert. Penelitian yang digunakan dari skala likert menentukan presentase kepuasan user terhadap aplikasi yang akan dibuat. Berikut ini adalah elemen-elemen yang dijadikan penilian pada kusioner yang telah dibuat adalah :

Tabel 9 Halaman Kuesioner Penelitian

No.	Elemen Penilaian	SB	B	C	K	SK
1.	Cepat dan responsif saat diakses pada mobile smartphone					
2.	Memberikan kemudahan untuk mengakses beserta dapat mudah dipahami berdasarkan penggunaan					
3.	Memberikan fitur-fitur menu navigasi pada aplikasi					
4.	Menampilkan informasi mengenai tentang profil aplikasi					
5.	Memberikan fitur-fitur berupa abjad, sortir dan menu-menu lainnya pada toolbar menu					
6.	Memberikan fitur menu utama untuk akses ke setiap kategori					
7.	Menampilkan hasil pencarian berdasarkan algoritma pencarian					
8.	Mendapatkan hasil tampilan yang menarik pada menu disetiap masing-masing kategori istilah komputer					
9.	Kamus dan database tersimpan dalam bentuk digital					
10.	Dapat diakses dimana saja melalui mobile smartphone					

Pembuatan komponen evaluasi penilian pada kuesioner diatas tegantung pada aspek sudut data yang dibutuhkan oleh klien, baik sejauh rencana desain UI (*User Interface*), desain UX (*User Experience*), dan maupun secara teknis. Survei diatas memiliki lima skala penilaian, yaitu SB (Sangat Baik) dengan skor 5, B (Baik) dengan skor 4, C (Cukup) dengan skor 3, K (Kurang) dengan skor 2, SK (Sangat Kurang) dengan skor 1. Pada penelitian ini menggunakan skala likert dalam menentukan presentase kepuasan user terhadap aplikasi yang telah dibuat. Berikut ini adalah proses perhitungan hasil kuesioner dengan menggunakan skala likert:

1. Penentuan Hasil Kuesioner

Proses penentuan nilai pada skala jawaban yang terdapat pada kuesioner penelitian.

Tabel 10 Penentuan Hasil Kuesioner

Skala Jawaban	Nilai
Sangat Baik	5
Baik	4

Cukup	3
Kurang	2
Sangat Kurang	1

2. Penentuan Skor Ideal

Proses penentuan nilai pada skala jawaban yang terdapat pada kuesioner penelitian.

Tabel 11 Penentuan Skor Ideal

Rumus (Nilai Skala Jawaban * Jumlah Responden)	Skala
5 x 17 = 85	SB
4 x 17 = 68	B
3 x 17 = 51	C
2 x 17 = 34	K
1 x 17 = 17	SK

3. Penentuan *Rating Scale*

Skala ini digunakan untuk mengukur nilai pada tiap jawaban kuesioner. Rating Scale ini berdasarkan pada skor ideal yang telah dibuat sebelumnya.

Tabel 12 Penentuan Hasil Kuesioner

Nilai Jawaban	Skala
69-85	SB
52-68	B
35-51	C
18-34	K
1-17	SK

4. Menghitung Presentase Seluruh Jawaban

Untuk menghitung persentase seluruh jawaban dari hasil kuesioner adalah dengan melakukan mencari nilai rata-rata dari seluruh jawaban terlebih dahulu dengan rumus (P1 + P2 + P3 + P4 + P5 + P6 + P7 + P8 + P9 + P10) / jumlah pertanyaan. Nilai rata-rata seluruh pertanyaan adalah sebesar 77.6, dengan perhitungan sebagai berikut (80 + 83 + 78 + 67 + 70 + 82 + 83 + 73 + 85+75) / 10 = 77,6. Perhitungan persentase seluruh jawaban kuesioner adalah (77,6 / 85) * 100% = 91,29%. Dapat disimpulkan bahwa dari persentase seluruh jawaban, tingkat kepuasan responden terhadap aplikasi yang telah dibuat adalah sebesar 91,29% dari 100% yang diharapkan.

IV. KESIMPULAN DAN SARAN

4.1 Kesimpulan

Berdasarkan hasil penelitian, pengujian, implementasi dan hasil yang telah dilakukan, maka dapat diperoleh masukan sebagai kesimpulan, yaitu:

1. Aplikasi kamus istilah komputer dapat menampilkan hasil pencarian istilah yang dapat tersimpan didalam database dengan menggunakan algoritma *boyer-moore* dan digunakan untuk dapat melakukan pencarian istilah komputer berbasis android.
2. Berdasarkan analisis penilaian yang dilakukan oleh responden, penilaian terhadap aplikasi kamus istilah komputer mendapatkan nilai rata-rata sebesar 91,29% dari 100%. Maka dapat disimpulkan, aplikasi ini dapat diimplementasikan kepada khususnya mahasiswa komputer sebagai user yang dapat menerjemahkan istilah komputer lebih cepat dari pada kamus cetak.

DAFTAR PUSTAKA

[1]. Hidayatul Qomariyah, S. M. ‘‘Aplikasi Kamus Istilah Komputer Pada Perangkat Mobile Berbasis Android.’’ *Prosiding SENTIA 2016 – Politeknik Negeri Malang*, vol 8 no.1, hh. 278-283. 2016

[2]. Santoso, H. 2010. *Aplikasi Web/asp.net + cd. Jakarta*. Jakarta: Indonesia.

[3]. Mulyana, H., & Maimunah. ‘‘Aplikasi Mobile Kamus Istilah Komputer Berbasis Android.’’ *Jurnal Penelitian Ilmu Komputer, System Embedded & Logic*, vol 1 no.2, hh. 28. 2014

[4]. Prayitno, A., Johar, A., & Setiawan, Y. ‘Implementasi Algoritma Turbo Boyer Moore Pada Aplikasi Kamus Istilah Biologi Berbasis Android’ Jurnal Rekursif, vol 6 no.1, hh. 13-23. 2018

[5]. Ardi, Y., Andreswari, D., & Setiawan, Y. ‘‘Rancang Bangun Aplikasi Kamus Istilah Kedokteran Dengan Menggunakan Algoritma Boyer-Moore Berbasis Android.’’ *Jurnal Rekursif,* vol 5 no.3, hh. 346-359. 2017

[6]. Soleh, M. Y. ‘‘Implementasi Algoritma KMP dan Boyer-Moore dalam Aplikasi Search Engine Sederhana.’’ *Sekolah Teknik Elektro Informatika ITB Bandung,* hh. 3. 2011

[7]. Pudjadi Tri. 2008, *Testing dan Implementasi sistem informasi*. (Online), updated 29 Agustus 2021, <http://pksm.mercubuana.ac.id/new/elearning/files_modul/>.

[8]. Ayuliana. 2009, *Testing dan Implementasi*. (Online), updated 29 Agustus 2021, <http://rifiana.staff.gunadarma.ac.id/Downloads/files/26083/Teknik+Pengujian+perangkat+Lunak+-+Black+Box.pdf/>.

[9]. Sugiyono. 2011. *Metode Penelitian Kuantitatif Kualitatif dan R&D*. Bandung: Alfabeta.

[10]. Rosa, A. S., & Salahuddin, M. 2011. *Modul Pembelajaran rekayasa perangkat lunak* (*Terstruktur dan Berorientasi Objek*). Bandung: Informatika.

[11]. Mulyana, H., & Maimunah. ‘Aplikasi Mobile Kamus Istilah Komputer Berbasis Android.’ *Jurnal Penelitian Ilmu Komputer, System Embedded & Logic*, vol 1 no.2, hh. 27-34. 2014

[12]. Safaat, N. H.. 2011. *Android (Pemograman Aplikasi Mobile Smartphone dan Tablet PC Berbasis Android)*. Bandung: Informatika.

[13]. Erlinda, & Masriadi. ‘Perancangan Aplikasi Mobile Kamus Istilah Komputer Untuk Mahasiswa Baru Bidang Komputer Berbasis Android.’ *Jurnal Teknologi dan Open Source*, vol 3 no.1, hh. 30-43. 2020

[14]. Febrian. 2007. *Kamus Komputer dan Teknologi Informasi*. Bandung: Informatika.

[15]. Ananda, D. d 2011. *Algoritma dan Pemrograman.* Politeknik Telkom.

BIOGRAPHY

Billy Gozali, dilahirkan di Tangerang, 1 Oktober 1998. Sekolah Dasar dilaksanakan di SD Maria Mediatrix, Kabupaten Tangerang, SMP Kartika II-2, Bandar Lampung, SMK Bina Latih Karya, Bandar Lampung.

Yo Ceng Giap, Saat ini bekerja sebagai dosen Tetap pada Program Studi Teknik Informatika di Universitas Buddhi Dharma

Desiyanna Lasut, Saat ini bekerja sebagai dosen Tetap pada Program Studi Teknik Informatika di Universitas Buddhi Dharma