BAB 3
ANALISIS ALGORITMA

3.1. Analisis Masalah

Penyelesaian dari permainan *word search puzzle* ini adalah menemukan semua kata yang tersembunyi di papan permainan yang berbentuk matriks. Permasalahan yang dihadapi adalah bagaimana sistem dapat menemukan semua kata yang tersembunyi di dalam *puzzle* yang telah tersusun secara random baik secara horizontal, vertikal, maupun diagonal atau sebaliknya. Setelah mendapatkan huruf yang dimaksud, pemain dapat mencari huruf kedua dengan melihat ke delapan arah yang mengelilingi huruf pertama tersebut. Metoda ini terus dilakukan sampai seluruh huruf pada kata ditemukan.

Oleh karena itu dilakukan penelitian dalam pencarian kata dalam papan permainan menggunakan algoritma *string searching.* Ada sekitar 35 algoritma pencarian kata yang bisa digunakan, baik algoritma yang diciptakan dari awal maupun pengembangan dari algoritma yang sudah ada [17]. Dua di antaranya yaitu algoritma *Knuth-Morris-Pratt* dan algoritma *Boyer-Moore*. Dengan melakukan perbandingan performansi dari algoritma *Knuth-Morris-Pratt* dan algoritma *Boyer-Moore* maka akan dapat diketahui cara kerja dan performansi dalam kecepatan dan ketepatan dari kedua algoritma tersebut.

3.2. Analisis Data Masukan

Program yang dibuat untuk pengujian yaitu papan permainan *word search puzzle* yang akan digunakan untuk menguji waktu pencarian kata. Pengujian dilakukan dilakukan terpisah dari dua algoritma pada saat pencarian kata. Untuk setiap pengujian algoritma, akan dilakukan pencatatan waktu pencarian dan posisi pola pada papan permainan. Pencarian kata dilakukan terhadap kata yang terdapat papan permainan. Papan permainan dibuat acak yang disusun random 15x15 dan 20x20, dimana didalam papan tersebut berisi nama-nama hewan indonesia baik mamalia, reptil, serangga,

ikan, burung dan sebagainya secara acak dengan menggunakan 8 sampel dengan panjang kata dari 3-10 karakter yang dapat dilihat pada tabel 3.1. Sedangkan keluaranya adalah :

1. Hasil penemuan pola pada papan permainan.
2. Waktu yang dibutuhkan untuk mengukur kecepatan pencarian yang dilakukan adalah mengambil waktu saat mulai menjalankan algoritma, dan waktu saat selesai, kemudian diambil selisihnya dalam waktu *millisecond*.

Tabel 3.1 Kata Yang Dicari

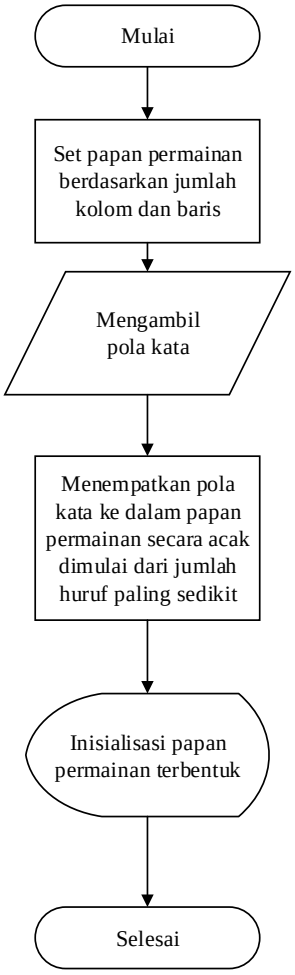
HIU
AYAM
DOMBA
CACING
KAMBING
FLAMINGO
SAPIHUTAN
JALAKPUTIH

Kata-kata yang dicari ditempatkan secara acak pada papan permainan dengan posisi yaitu :

1. Horizontal ke kanan
2. Horizontal ke kiri
3. Vertikal ke atas
4. Vertikal ke bawah
5. Diagonal ke kiri atas
6. Diagonal ke kiri bawah
7. Diagonal ke kanan atas
8. Diagonal ke kanan bawah

3.2.1. Inisialisasi Papan Permainan

Papan permainan disusun secara acak disusun secara horizontal, vertikal maupun diagonal dan dapat ditulis dalam posisi terbalik maupun tidak berdasarkan pola kata yang terdapat pada file exstensi .txt. Papan permainan diacak secara random dari alpabet A-Z dengan jumlah total abjad yaitu 26 huruf. Papan permainan tersebut diacak berdasarkan pola kata pada papan permainan yang kemudian ditempatkan secara berurutan dimulai dengan kata yang mempunyai huruf paling sedikit yaitu 3 huruf sampai dengan jumlah huruf terbanyak yaitu 10 huruf. Berikut adalah gambar 3.1 flowchart inisialisasi papan permainan :



Gambar 3.1 Inisialisasi Papan Permainan

Untuk penerapan permainan pada algoritma *Knuth-Morris-Pratt* dan algoritma *Boyer-Moore*, akan diperlihatkan papan permainan yang akan mencari pola yang sudah ditentukan. Pada gambar 3.2 yaitu gambar papan permainan yang dimana tabel sebelah kanan adalah pola yang sudah teracak pada papan permainan yang terdapat tabel sebelah kiri.

X	M	C	F	H	T	V	G	Y	U	I	H	D	Z	V
M	A	Y	A	L	M	B	T	T	E	B	N	X	Z	H
M	J	U	F	K	B	R	E	T	S	M	A	H	Q	X
T	A	B	E	B	C	Y	B	K	M	B	O	Z	T	D
K	L	I	Z	Z	Q	T	D	L	A	G	B	T	B	E
I	A	R	M	N	Z	T	N	R	N	X	N	H	K	M
R	K	E	W	D	A	H	H	I	I	P	K	I	T	K
K	P	S	B	K	N	T	M	T	F	D	T	A	M	K
G	U	I	B	E	X	A	U	V	L	I	F	T	Z	H
N	T	K	E	B	L	S	M	H	Z	G	U	M	K	R
A	I	A	O	F	M	Y	I	B	I	M	D	B	P	P
J	H	K	B	B	T	N	U	T	B	P	A	C	P	W
J	M	X	M	M	K	H	Z	Z	W	D	W	Q	X	R
S	H	S	D	D	I	X	G	Z	K	G	A	Z	D	C
Y	Z	X	I	I	R	G	X	E	U	W	N	A	M	E

HIU
AYAM
DOMBA
CACING
KAMBING
FLAMINGO
SAPIHUTAN
JALAKPUTIH

Gambar 3.2 Gambar Papan Permainan

Pada tahap pencarian pola kata pada permainan terdapat 8 arah pencarian kata, berikut adalah langkah-langkah yang akan dilakukan:

1. Cari keberadaan kata pada tiap baris, jika tidak ketemu ubah cara baca, lakukan pencocokan kata pada tiap baris. Langkah tersebut dapat dilihat pada gambar 3.3 dibawah ini.

X	M	C	F	H	T	V	G	Y	U
M	A	Y	A	L	M	B	T	T	E
M	J	U	F	K	B	R	E	T	S
T	A	B	E	B	C	Y	B	K	M
K	L	I	Z	Z	Q	T	D	L	A
I	A	R	M	N	Z	T	N	R	V
R	K	E	W	D	A	H	H	K	I
K	P	S	B	K	N	T	D	T	F
G	U	I	B	E	X	P	U	V	L
N	T	K	E	B	M	S	M	H	Z

Gambar 3.3 Pencarian Pada Tiap Baris

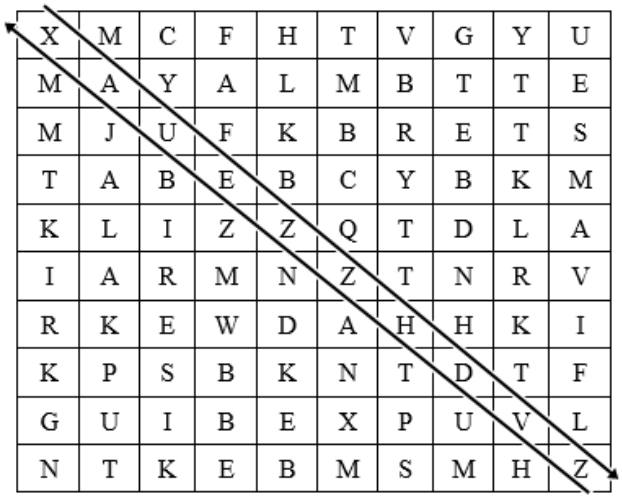
2. Jika tidak ketemu lakukan pencocokan kata per kolom, jika tidak ketemu ubah cara baca, lakukan pencocokan kata pada tiap kolom. Langkah tersebut dapat dilihat pada gambar 3.4 dibawah ini.

X	M	C	F	H	T	V	G	Y	U
M	A	Y	A	L	M	B	T	T	E
M	J	U	F	K	B	R	E	T	S
T	A	B	E	B	C	Y	B	K	M
K	L	I	Z	Z	Q	T	D	L	A
I	A	R	M	N	Z	T	N	R	V
R	K	E	W	D	A	H	H	K	I
K	P	S	B	K	N	T	D	T	F
G	U	I	B	E	X	P	U	V	L
N	T	K	E	B	M	S	M	H	Z

Gambar 3.4 Pencarian Pada Tiap Kolom

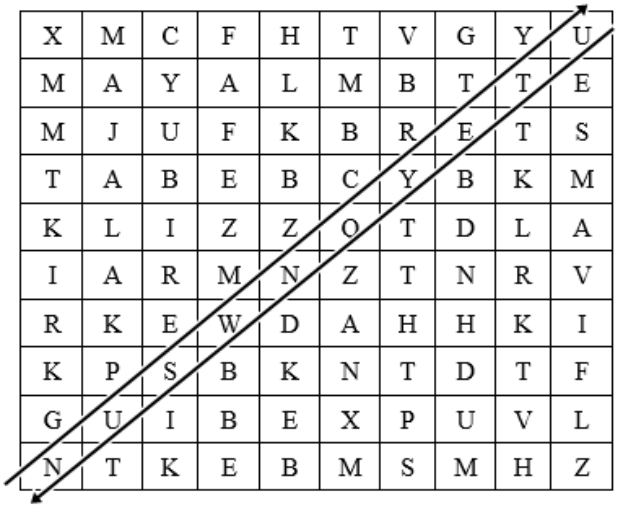
3. Jika tidak ketemu lakukan pencocokan kata secara diagonal dari kiri atas ke kanan bawah yang panjang diagonalnya ≥ panjang kata yang dicari, jika tidak ketemu ubah cara baca, lakukan pencocokan kata secara diagonal dari kanan bawah ke kiri atas yang panjang diagonalnya ≥

panjang kata yang dicari. Langkah tersebut dapat dilihat pada gambar 3.5 dibawah ini.



Gambar 3.5 Pencarian Secara Diagonal Kiri Atas ke Kanan Bawah

4. Jika tidak ketemu lakukan pencocokan kata secara diagonal dari kanan atas ke kiri bawah yang panjang diagonalnya ≥ panjang kata yang dicari, jika tidak ketemu ubah cara baca, lakukan pencocokan kata secara diagonal dari kiri bawah ke kanan atas yang panjang diagonalnya ≥ panjang kata yang dicari. Langkah tersebut dapat dilihat pada gambar 3.6 dibawah ini.

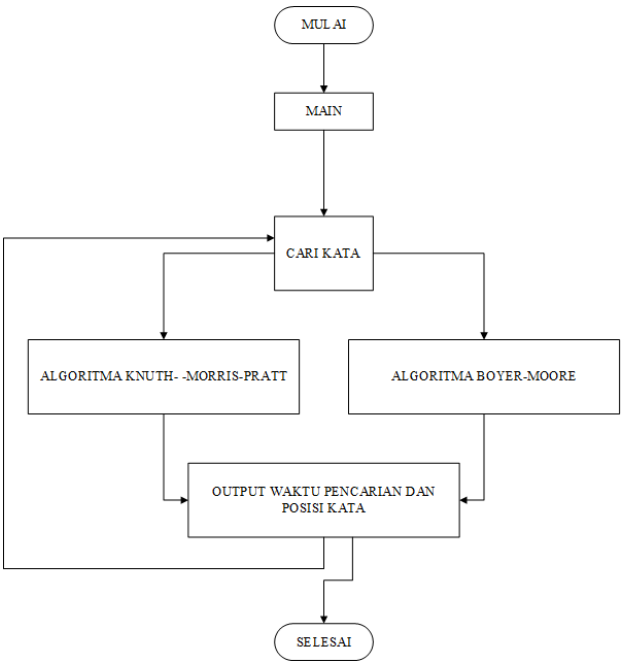


Gambar 3.6 Pencarian Secara Diagonal Kanan atas Ke Kiri Bawah

5. Ulangi hingga tiap kata sudah ditemukan posisinya pada papan permainan.

3.3. Analisis Algoritma

Analisis algoritma yang dilakukan dalam penelitian ini adalah untuk menganalisis cara kerja algoritma *Knuth-Morris-Pratt* dan algoritma *Boyer-Moore* terhadap kasus permainan *word search puzzle* yang dikhususkan pada proses pencarian kata. Bagan pencarian secara utuh dapat dilihat pada gambar 3.7 di bawah ini.

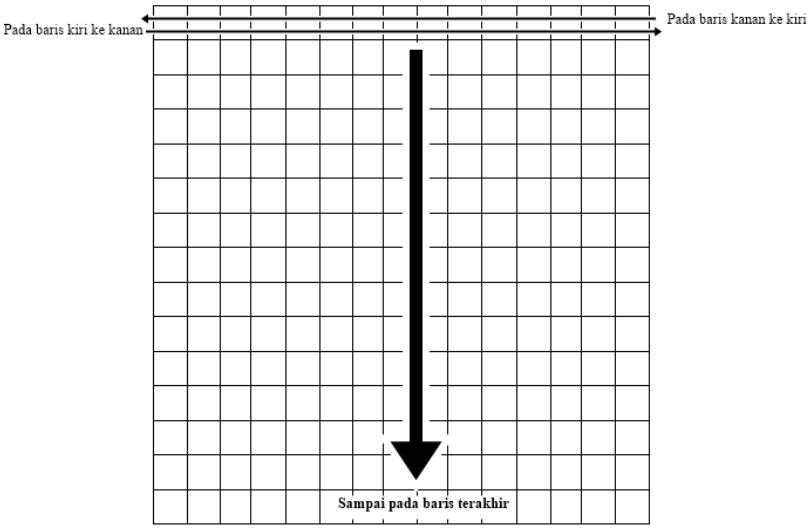


Gambar 3.7 Pencarian Kata

3.3.1. Analisis Pencarian Pola pada Permainan

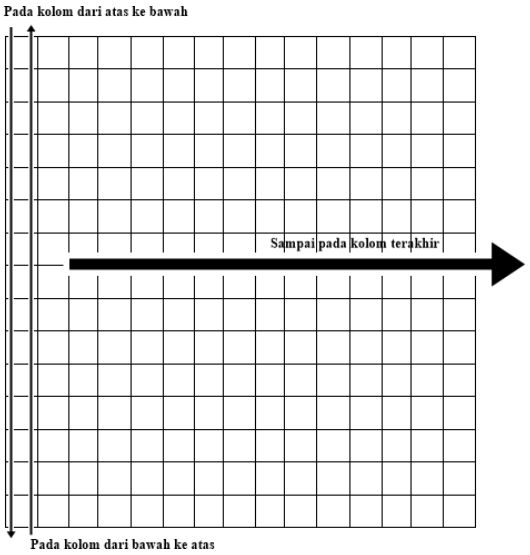
Pada tahap pencarian kata pada papan permainan langkah – langkah yang dilakukan algoritma pada saat pencarian kata adalah sebagai berikut :

1. Pada tahap pertama pada gambar 3.8, papan permainan akan membaca teks dari baris kiri ke kanan dan begitupun sebaliknya untuk mencari pola.



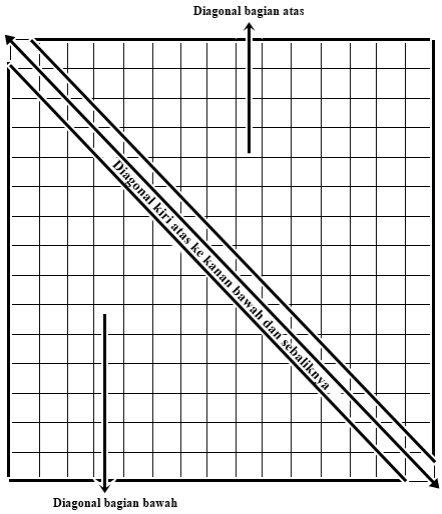
Gambar 3.8 Pencarian Pada Baris Kiri ke Kanan

2. Pada tahap kedua pada gambar 3.9, papan permainan akan membaca teks dari kolom pertama dari atas kebawah dan begitupun sebaliknya untuk mencari pola kata.



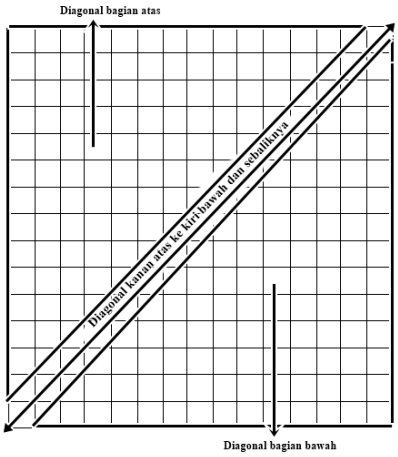
Gambar 3.9 Pencarian Pada Kolom Dari Atas ke Bawah

3. Pada tahap ketiga papan permainan akan membaca teks secara diagonal dari kiri atas ke kanan bawah dan begitupun sebaliknya untuk mencari pola kata, namun pada tahap ini terbagi dua pencarian yaitu memisah pola pencarian menjadi dua bagian yaitu diagonal bagian atas dan diagonal bagian bawah seperti pada gambar 3.10.



Gambar 3.10 Pencarian Secara Diagonal Dari Kiri Atas ke Kanan Bawah dan Sebaliknya

4. Pada tahap keempat sama seperti pada tahap ketiga, papan permainan akan membaca teks secara diagonal dari kanan atas ke kiri bawah dan begitupun sebaliknya untuk mencari pola kata, namun pada tahap ini terbagi dua pencarian yaitu memisah pola pencarian menjadi dua bagian yaitu diagonal bagian atas dan diagonal bagian bawah seperti pada gambar 3.11.



Gambar 3.11 Pencarian Secara Diagonal Dari Kanan Atas ke Kiri Bawah dan Sebaliknya

Pada saat pencarian, seperti pada langkah – langkah diatas, langkah pertama akan menentukan teks yang akan dicari polanya berdasarkan pencarian. Sebagai contoh diberikan sebuah papan permainan dengan panjang papan permainan 15 x 15 pada gambar 3.12 dengan pencarian pola kata yaitu FLAMINGO.

X	M	C	F	H	T	V	G	Y	U	I	H	D	Z	V
M	A	Y	A	L	M	B	T	T	E	B	N	X	Z	H
M	J	U	F	K	B	R	E	T	S	M	A	H	Q	X
T	A	B	E	B	C	Y	B	K	M	B	O	Z	T	D
K	L	I	Z	Z	Q	T	D	L	A	G	B	T	B	E
I	A	R	M	N	Z	T	N	R	N	X	N	H	K	M
R	K	E	W	D	A	H	H	I	I	P	K	I	T	K
K	P	S	B	K	N	T	M	T	F	D	T	A	M	K
G	U	I	B	E	X	A	U	V	L	I	F	T	Z	H
N	T	K	E	B	L	S	M	H	Z	G	U	M	K	R
A	I	A	O	F	M	Y	I	B	I	M	D	B	P	P
J	H	K	B	B	T	N	U	T	B	P	A	C	P	W
J	M	X	M	M	K	H	Z	Z	W	D	W	Q	X	R
S	H	S	D	D	I	X	G	Z	K	G	A	Z	D	C
Y	Z	X	I	I	R	G	X	E	U	W	N	A	M	E

HIU
AYAM
DOMBA
CACING
KAMBING
FLAMINGO
SAPIHUTAN
JALAKPUTIH

Gambar 3.12 Papan permainan 15 x 15

Pada contoh kasus diatas, pada tahap pertama yaitu akan menentukan teks dari papan permainan dengan cara membuat teks secara sejajar seperti pada tabel 3.2 dengan cara membaca teks secara diagonal menjadi vertikal. Dari teks tersebut akan dicari pola kata yang dicari.

Tabel 3.2 Pola Yang Disejajarkan

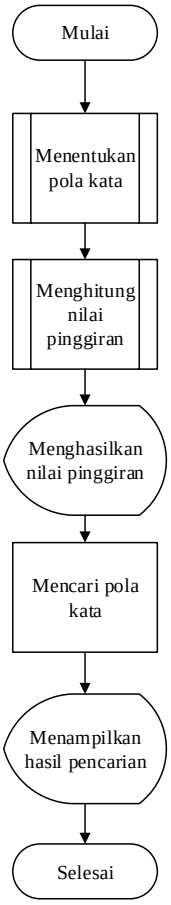
YHXBFLAMINGOHZV
SMKOBXTHRABAXZ
JHAEENHNLMMND
JIKBKATDKSBH
ATIBDZTBTEI
NUSWNQYETU
GPEMZCRTY
KKRZBBBG
RAIEKMV

ILBFLT
KAUAH
TJYF
MAC
MM
X

Dari ekstraksi kata tersebut kemudian kata disejajarkan untuk dicari menggunakan Algoritma *Knuth-Morris-Pratt* dan Algoritma *Boyer-Moore*

3.3.1.1. Tahapan Pencarian Menggunakan Algoritma Knuth-Morris-Pratt

Algoritma Knuth-Morris-Pratt mempunyai tahapan-tahapan dalam proses pencarian kata, seperti yang dapat dilihat pada gambar 3.13 dibawah ini.



Gambar 3.13 Flowchart Algoritma Knuth-Morris-Pratt

Tahapan pencarian pada algoritma Knuth-Morris-Pratt terdapat dua fase yaitu tahap preprocessing dan tahap pencarian. Tahapan pertama yaitu tahap preprocessing akan menghitung nilai pinggiran dari pattern **FLAMINGO**, maka lihat pada tabel 3.3 yaitu tabel nilai pinggiran sebagai berikut :

Tabel 3.3 Tabel Pencarian Nilai Pinggiran

J	1	2	3	4	5	6	7	8
P[j]	F	L	A	M	I	N	G	O
B[j]	0	0	0	0	0	0	0	0

Ket : j = Index
P[j] = Pattern
B[j] = Nilai pinggiran

Tahapan pencarian pada baris pertama yaitu T (teks) = **YHXBFLAMINGOHZV** dan P (*pattern*) = **FLAMINGO**. Setelah didapatkan nilai pinggiran dari *pattern*, maka proses pencarian dapat diproses sebagai berikut :

1. Pada *pattern F* pada tabel 3.4 terjadi ketidakcocokan dengan dengan teks *Y*, maka lihat nilai pinggiran *pattern F* pada tabel 3.3 adalah 0. Karena tidak tejadi ketidakcocokan, maka pattern digeser sejumlah 1 karakter.

Tabel 3.4 Tabel Tahapan Pencarian Kata Iterasi ke-1

Teks :	Y	H	X	B	F	L	A	M	I	N	G	O	H	Z	V
Pattern :	F	L	A	M	I	N	G	O							

Ket :

	Cocok (*match*)
	Tidak cocok (*missmatch*)

2. Pada *pattern F* pada tabel 3.5 terjadi ketidakcocokan dengan dengan teks *H* maka lihat nilai pinggiran *pattern F* pada tabel 3.3 yaitu 0. Karena tidak tejadi ketidakcocokan, maka pattern digeser sejumlah 1 karakter.

Tabel 3.5 Tabel Tahapan Pencarian Kata Iterasi ke-2

Teks :	Y	H	X	B	F	L	A	M	I	N	G	O	H	Z	V
Pattern :		F	L	A	M	I	N	G	O						

Ket :

	Cocok (*match*)

	Tidak cocok (*missmatch*)

3. Pada *pattern F* pada tabel 3.6 terjadi ketidakcocokan dengan dengan teks *X* maka lihat nilai pinggiran *pattern F* pada tabel 3.3 yaitu 0. Karena tidak tejadi ketidakcocokan, maka pattern digeser sejumlah 1 karakter.

Tabel 3.6 Tabel Tahapan Pencarian Kata Iterasi ke-3

Teks :	Y	H	X	B	F	L	A	M	I	N	G	O	H	Z	V
Pattern :			F	L	A	M	I	N	G	O					

Ket :

	Cocok (*match*)
	Tidak cocok (*missmatch*)

4. Pada *pattern B* pada tabel 3.7 terjadi ketidakcocokan dengan dengan teks *O* maka lihat nilai pinggiran *pattern A* pada tabel 3.3 yaitu 0, Karena tidak tejadi ketidakcocokan, maka pattern digeser sejumlah 1 karakter.

Tabel 3.7 Tabel Tahapan Pencarian Kata Iterasi ke-4

Teks :	Y	H	X	B	F	L	A	M	I	N	G	O	H	Z	V
Pattern :				F	L	A	M	I	N	G	O				

Ket :

	Cocok (*match*)
	Tidak cocok (*missmatch*)

5. Pada *pattern F,L,A,M,I,N,G,O* pada tabel 3.8 terjadi kecocokan dengan dengan teks *F,L,A,M,I,N,G,O*. Dari semua *pattern* itu cocok dengan teks. Maka pencarian selesai dilakukan pada algoritma *Knuth-Morris-Pratt* setelah melakukan 5 iterasi.

Tabel 3.8 Tabel Tahapan Pencarian Kata Iterasi ke-6

Teks :	Y	H	X	B	F	L	A	M	I	N	G	O	H	Z	V
Pattern :					F	L	A	M	I	N	G	O			

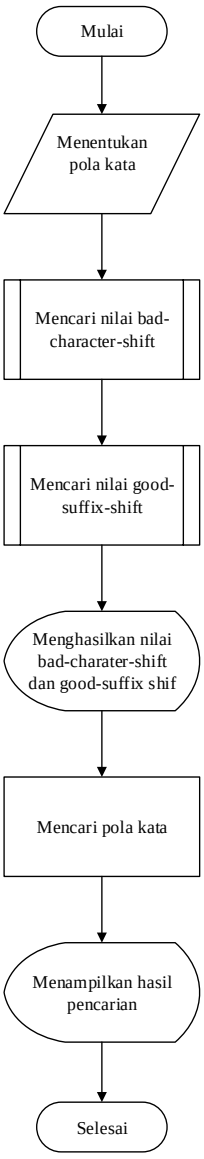
Ket :

	Cocok (*match*)
	Tidak cocok (*missmatch*)

Pada tahap pencarian dengan algoritma *Knuth-Morris-Pratt* diatas pencarian selesai setelah melakukan sebanyak 4 iterasi.

3.3.1.2. Tahapan Pencarian Menggunakan Algoritma Boyer-Moore

Algoritma Boyer-Moore mempunyai tahapan-tahapan dalam proses pencarian kata, seperti yang dapat dilihat pada gambar 3.14 dibawah ini



Gambar 3.14 Flowchart Algoritma Boyer-Moore

Tahapan pencarian pada algoritma *Boyer-Moore* sama saja dengan algoritma *Knuth-Morris-Pratt* yaitu melalui 2 tahap yaitu tahap *preprocessing* dan tahap pencarian pola kata. Namun didalam tahap preprocessing, algoritma *Boyer-Moore* akan mencari nilai *bad-character-*

shift dan *good-suffix shift*. Tahapan pertama yaitu tahap preprocessing algoritma akan menghitung *bad-character shift* yang akan menghasilkan nilai OH (*Occurrence Heuristic*) dari *pattern FLAMINGO*, maka lihat pada tabel 3.9 sebagai berikut :

Tabel 3.9 Tabel Nilai Occurrence Heuristic (OH)

Index	0	1	2	3	4	5	6	7
Karakter	F	L	A	M	I	N	G	O
OH value	7	6	5	4	3	2	1	0

Ket : OH = Occurrence Heuristic

Setelah mendapatkan hasil akhir dari perhitungan *bad-character shift* maka dilanjutkan membuat tabel akhiran/*suffix* dari *pattern* . Tabel *suffix* berfungsi untuk mencari nilai *good-suffix shift* yang akan menghasil nilai MH (*Match Heuristic*). Tabel tersebut dapat dilihat pada tabel 3.10 dibawah ini :

Tabel 3.10 Suffix Kanan ke Kiri Dan Suffix Kiri ke Kanan

Pattern : FLAMINGO (kanan ke kiri)			
idx	Prefix	*Suffix*	Length
7	O	FLAMING	7
6	GO	FLAMIN	6
5	NGO	FLAMI	5
4	INGO	FLAM	4
3	MINGO	FLA	3
2	AMINGO	FL	2
1	LAMINGO	F	1
0	FLAMINGO	NULL	0

Pattern : FLAMINGO (kiri ke kanan)			
Idx	Prefix	*Suffix*	Length
0	F	LAMINGO	7
1	FL	AMINGO	6
2	FLA	MINGO	5
3	FLAM	INGO	4
4	FLAMI	NGO	3
5	FLAMIN	GO	2
6	FLAMING	O	1
7	FLAMINGO	NULL	0

Pembuatan tabel *suffix* dari kanan ke kiri berfungsi untuk dijadikan *suffix comparator* sedangkan pembuatan tabel *suffix* dari kiri ke kanan berfungsi sebagai *suffix* yang nanti akan dibandingkan dan pencarian *good-suffix shift* yang akan menghasilkan nilai MH (*Match Heuristic*), tabel tersebut dapat dilihat pada tabel 3.11 dibawah ini.

Tabel 3.11 Suffix Lengkap Beserta Suffix Comparator

	Suffix								
Index	0	1	2	3	4	5	6	7	Move
Prefix	F	L	A	M	I	N	G	O	
Suffix	LAMINGO	AMINGO	MINGO	INGO	NGO	GO	O	NULL	
Suffix comparator	FLAMING								1
	FLAMIN								2
	FLAMI								3
	FLAM								4
	FLA								5
	FL								6
	F								7
	NULL								8
MH value	8	8	8	8	8	8	8	1	

Setelah proses pada tahap *preprocessing* yaitu perhitungan pencarian nilai *good-suffix shift* didapatkan nilai MH dan *bad-character shift* didapatkan nilai OH yang dapat dilihat pada tabel 3.12.

Tabel 3.12 Tabel BmBc dan BmGs Berserta Nilai OH dan MH

BmBc & BmGs								
Index	0	1	2	3	4	5	6	7
Karakter	F	L	A	M	I	N	G	O
Nilai OH	7	6	5	4	3	2	1	0
Nilai MH	8	8	8	8	8	8	8	1

Ket :

- Nilai OH = Nilai yang dihasilkan *Bad-Character Shift* (BmBc)
- Nilai MH = Nilai yang dihasilkan *Good-Suffix Shift* (BmGs)

Kemudian setelah tahap *preprocessing*, tahap kedua yaitu tahap pencarian pada teks = **YHXBFLAMINGOHZV** dengan menggunakan nilai OH dan MH yang terdapat pada tabel 3.12 yang telah dicari sebelumnya dengan langkah-langkah sebagai berikut :

1. Pada iterasi ke-1, *pattern* akan membandingkan dari posisi terakhir *pattern* yaitu dari kanan, di posisi ke 7 pada tabel 3.13, karakter pada teks *M* tidak cocok dengan *pattern O*, maka lihat besar nilai MH pada tabel *Good-suffix shift* dan OH pada tabel *Bad-character shift* di tabel 3.12.

Kemudian bandingkan nilai MH dan OH berdasarkan nilai maksimal untuk melakukan pergeseran.

BmGs [7] = BmBc[M] -7 + 7 atau Max (1 = 4 - 7 + 7) = 4 (3.1)

Sehingga geser *pattern* sebesar 4 posisi (nilai maksimal dari kedua tabel pergeseran).

Tabel 3.13 Tabel Pencarian Boyer-Moore Iterasi ke-1

Index :	0	1	2	3	4	5	6	7	8	9					
Teks :	Y	H	X	B	F	L	A	M	I	N	G	O	H	Z	V
Pattern :	F	L	A	M	I	N	G	O							

Ket :

	Cocok (*match*)
	Tidak cocok (*missmatch*)

2. Pada iterasi ke-2, *pattern* akan membandingkan dari posisi terakhir *pattern* yaitu dari kanan, di posisi ke 11 pada tabel 3.14, karakter pada teks *O,G,N,I,M,A,L,F* cocok dengan *pattern* *O,G,N,I,M,A,L,F*. karena *pattern* sudah cocok dengan teks maka pencarian selesai.

Tabel 3.14 Tabel Pencarian Boyer-Moore Iterasi ke-2

Index :	0	1	2	3	4	5	6	7	8	9	10	11	13	14	15
Teks :	Y	H	X	B	F	L	A	M	I	N	G	O	H	Z	V
Pattern :					F	L	A	M	I	N	G	O			

Ket :

	Cocok (*match*)
	Tidak cocok (*missmatch*)

Pada tahap pencarian dengan algoritma *Boyer-Moore* diatas pencarian selesai setelah melakukan sebanyak 1 iterasi.

3.4. Analisis Kebutuhan Perangkat Lunak

Analisis kebutuhan perangkat lunak dibagi menjadi dua bagian yaitu analisis kebutuhan non fungsional dan analisis kebutuhan fungsional.

3.5.1. Analisis Kebutuhan Non Fungsional

Analisis kebutuhan non fungsional adalah langkah dimana seorang pembangun perangkat lunak menganilisis sumber daya yang akan digunakan dan menggunakan perangkat lunak yang akan dijadikan simulasi pengujian dibangun.

Analisis kebutuhan non fungsional yang dilakukan dibagi dalam 2 tahap, yaitu:

1. Analisis perangkat keras (*hardware*).
2. Analisis perangkat lunak (*software*).

3.5.1.1. Analisis Kebutuhan Perangkat Keras

Perangkat keras yang digunakan untuk membangun permainan word search puzzle untuk pengujian analisis ini memiliki spesifikasi sebagai berikut :

1. *Processor* 2.5GHz.
2. VGA (*Video Graphics Array*) 1Gb.
3. RAM (*Random Access Memory*) 8Gb.
4. Hardisk 500Gb.

3.5.1.2. Analisis Kebutuhan Perangkat Lunak

Perangkat lunak yang dibutuhkan untuk membangun pengujian permainan word search puzzle ini adalah sebagai berikut :

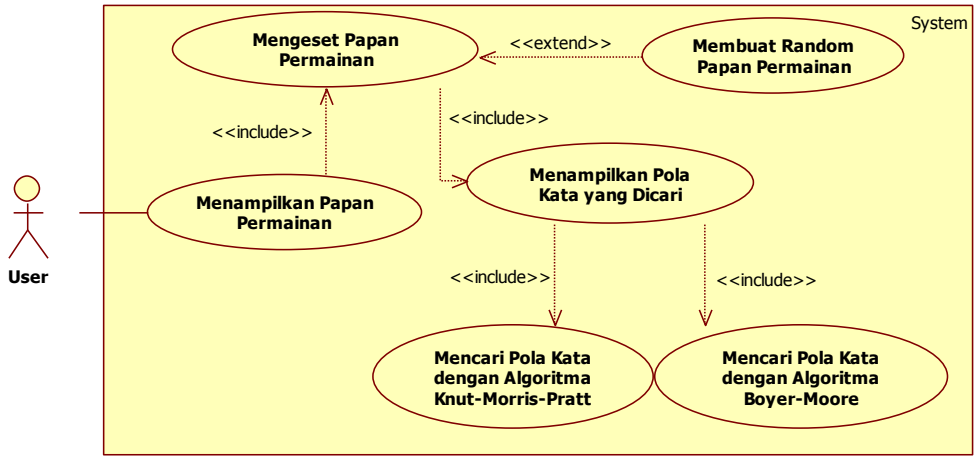
1. Sistem Operasi Linux Mint 17
2. Java SE Development Kit 7
3. Java Runtime Environtment 7
4. Netbeans IDE 8.0

3.5.2. Analisis Kebutuhan Fungsional

Analisis fungsional adalah analisis yang menggambarkan proses kegiatan yang akan diterapkan didalam sebuah sistem dan menjelaskan kebutuhan yang diperlukan oleh sistem agar dapat berjalan dengan baik serta sesuai dengan kebutuhan. Permodelan sistem menggunakan UML (Unified Modeling Language). Tahap – tahap permodelan dalam analisis tersebut antara lain Use case diagram, Class Diagram, Activity Diagram dan Sequence Diagram.

3.5.2.1. Use Case Diagram

Pada sub bab berikut ini dijelaskan bagaimana *User Case* yang terjadi dalam pengujian permainan *word search puzzle* dimodelkan dalam sekumpulan *Use Case* dan *Actor* dan bagaimana hubungan-hubunganya yang akan dipetakan dalam digaram *Use Case*, dilengkapi dengan skenario untuk menjelaskan dari gambaran *Use Case* yang ada dengan menggambarkan skenario per *Use Case*, *Use Case Diagram* dapat dilihat pada gambar 3.15 berikut.



Gambar 3.15 Use Case Diagram

3.5.2.1.1. Identifikasi Aktor

Identifikasi Aktor berfungsi untuk menjelaskan aktor yang terdapat pada *Use Case Diagram* dan menggambarkan pihak – pihak yang berperan dalam sistem. Aktor yang berperan didalam sistem ini dapat dilihat pada tabel 3.15 dibawah ini :

Tabel 3.15 Identifikasi Aktor

No	Aktor	Deskripsi
A-01	User	Merupakan *actor* yang dapat menjalankan aplikasi

3.5.2.1.2. Identifikasi Use Case

Identifikasi *Use Case* berfungsi untuk menjelaskan fungsi *Use Case* yang terdapat pada *Use Case Diagram* dan menjelaskan aktivitas yang disiapkan oleh sistem. Identifikasi *Use Case* yang terdapat pada sistem dapat dilihat pada tabel 3.16 dibawah ini :

Tabel 3.16 Identifikasi Use Case

No.	Use Case	Deskripsi
UC-01	Menampilkan Papan Permainan	Fungsionalitas untuk menampilkan papan permainan
UC-02	Mengeset Panjang Papan Permainan	Fungsionalitas untuk mengeset panjang papan permainan
UC-03	Membuat Random Papan Permainan	Fungsional untuk membuat random karakter pada papan permainan
UC-04	Menampilkan Pola Kata yang dicari	Fungsionalitas untuk menampilkan pola kata yang akan dicari
UC-05	Mencari Pola Kata dengan Algoritma Knuth-Morris-Pratt	Fungsionalitas untuk mencari pola kata dengan algoritma *Knuth-Morris-Pratt*
UC-06	Mencari Pola Kata dengan Algoritma Boyer-Moore	Fungsionalitas untuk mencari pola kata dengan algoritma *Boyer-Moore*

3.5.2.1.3. Use Case Skenario

Use Case Skenario mendeskripsikan urutan langkah – langkah dalam proses, baik yang dilakukan aktor terhadap sistem maupun yang dilakukan oleh sistem terhadap aktor. Berdasarkan *Use Case Diagram* pada gambar 3.15 di atas maka *Use Case* Skenario untuk pengujian aplikasi yang dibangun akan dijelaskan sebagai berikut.

3.5.2.1.3.1. Use Case Skenario Menampilkan Papan Permainan

Use Case Skenario untuk Use Case menampilkan papan permainan dijelaskan pada tabel 3.17 dibawah ini.

Tabel 3.17 Use Case Skenario Menampilkan Papan Permainan

Identifikasi	
Nomor	UC-01
Nama Use Case	Menampilkan Papan Permainan
Aktor	User
Tujuan	Masuk ke dalam pengujian permainan
Kondisi awal	Sistem menampilkan papan permainan kosong
Skenario Utama	
Aksi Aktor	**Reaksi Sistem**
1. User memulai pengujian	
	2. Sistem menampilkan papan permainan kosong
Kondisi Akhir	Sistem menampilkan papan permainan kosong yang akan diuji

3.5.2.1.3.2. Use Case Skenario Mengeset Papan Permainan

Use Case Skenario untuk Use Case mengeset panjang papan permainan dijelaskan pada tabel 3.18 dibawah ini.

Tabel 3.18 Use Case Skenario Mengeset Panjang Papan Permainan

Identifikasi	
Nomor	UC-02
Nama Use Case	Mengeset Panjang Papan Permainan
Aktor	User
Tujuan	Mengeset panjang papan permainan
Kondisi awal	Sistem menampilkan papan permainan kosong
Skenario Utama	
Aksi Aktor	**Reaksi Sistem**
1. Mengisi kolom dan baris papan permainan	
2. Mengklik *button* set papan	
	3. Sistem membaca kolom dan baris yang sudah diset

	4. Sistem menampilkan papan permainan sesuai dengan kolom dan baris
Kondisi Akhir	Sistem menampilkan papan permainan sesuai dengan kolom dan baris yang telah ditentukan

3.5.2.1.3.3. Use Case Skenario Membuat Random Papan Permainan

Use Case Skenario untuk Use Case membuat random papan permainan dijelaskan pada tabel 3.19 dibawah ini.

Tabel 3.19 Use Case Skenario Membuat Random Papan Permainan

Identifikasi	
Nomor	UC-03
Nama Use Case	Membuat Random Papan Permainan
Aktor	User
Tujuan	Mengeset random karakter pada papan permainan
Kondisi awal	Sistem menampilkan papan permainan yang telah diset
Skenario Utama	
Aksi Aktor	**Reaksi Sistem**
1. Mengklik *button* random	
	2. Sistem mengenerate random karakter
	3. Sistem menampilkan papan permainan yang telah dirandom karakternya.
Kondisi Akhir	Sistem menampilkan papan permainan yang telah di random karakternya

3.5.2.1.3.4. Use Case Skenario Menampilkan Pola Kata yang Dicari

Use Case Skenario untuk Use Case menampilkan pola kata yang dicari dijelaskan pada tabel 3.20 dibawah ini.

Tabel 3.20 Use Case Skenario Menampilkan Pola Kata Yang Dicari

Identifikasi	
Nomor	UC-04
Nama Use Case	Menampilkan Pola Kata yang Dicari
Aktor	User
Tujuan	Menampilkan pola kata yang dicari
Kondisi awal	Sistem menampilkan pola kata
Skenario Utama	
Aksi Aktor	**Reaksi Sistem**
	1. Sistem membaca file wordlist.txt yang disimpan pada *package* program
	2. Sistem mengeset pola kata pada list
	3. Sistem menampilkan pola kata pada list
Kondisi Akhir	Sistem menampilkan pola kata yang terdapat pada papan permainan

3.5.2.1.3.5. Use Case Skenario Mencari Pola Kata dengan Algoritma Knuth-Morris-Pratt

Use Case Skenario untuk Use Case menampilkan mencari pola kata dengan algoritma Knuth-Morris-Pratt dijelaskan pada tabel 3.21 dibawah ini.

Tabel 3.21 Use Case Skenario Cari Pola Kata Dengan Algoritma Knuth-Morris-Pratt

Identifikasi	
Nomor	UC-05
Nama Use Case	Mencari Pola Kata dengan Algoritma Knuth-Morris-Pratt
Aktor	User
Tujuan	Menampilkan posisi dan waktu pencarian kata dengan algoritma *Knuth-Morris-Pratt*
Kondisi awal	Sistem menampilkan kolom cari kata kosong
Skenario Utama	
Aksi Aktor	**Reaksi Sistem**
1. Mengklik pola kata pada list yang terdapat pola kata	
	2. Sistem mengambil pola kata pada list
	3. Sistem menampilkan pola kata pada kolom cari kata
4. User mengklik button pencarian dengan menggunakan algoritma *Knuth-Morris-Pratt*	
	5. Sistem melakukan pencarian kata berdasarkan arah pencarian
	6. Mencari pola kata pada teks yang sudah diekstraksi
	7. Jika tidak ditemukan ubah cara
Kondisi Akhir	Sistem menampilkan hasil posisi pola dan waktu pencarian algoritma *Knuth-Morris-Pratt.*

3.5.2.1.3.6. Use Case Skenario Mencari Pola Kata dengan Algoritma Boyer-Moore

Use Case Skenario untuk Use Case mencari pola kata dengan algoritma Boyer-Moore dijelaskan pada tabel 3.22 dibawah ini.

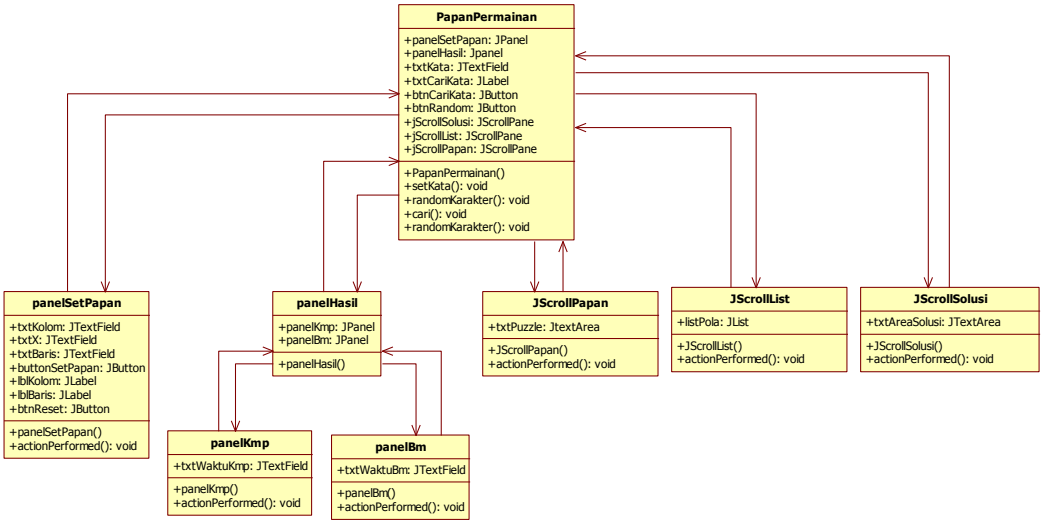
Tabel 3.22 Use Case Skenario Cari Pola Kata Dengan Algoritma Boyer-Moore

Identifikasi	
Nomor	UC-06
Nama Use Case	Mencari Pola Kata dengan Algoritma Boyer-Moore
Aktor	User
Tujuan	Menampilkan posisi dan waktu pencarian kata dengan algoritma *Boyer-Moore*
Kondisi awal	Sistem menampilkan kolom cari kata kosong
Skenario Utama	
Aksi Aktor	**Reaksi Sistem**
1. Mengklik pola kata pada list yang terdapat pola kata	
	2. Sistem mengambil pola kata pada list
	3. Sistem menampilkan pola kata pada kolom cari kata
4. User mengklik button pencarian dengan menggunakan algoritma *Boyer-Moore*	
	5. Sistem melakukan pencarian kata berdasarkan arah pencarian
	6. Mencari pola kata pada teks yang sudah diekstraksi
	7. Jika tidak ditemukan ubah cara
Kondisi Akhir	Sistem menampilkan hasil posisi pola dan waktu pencarian algoritma *Boyer-Moore*

3.5.2.2. Class Diagram

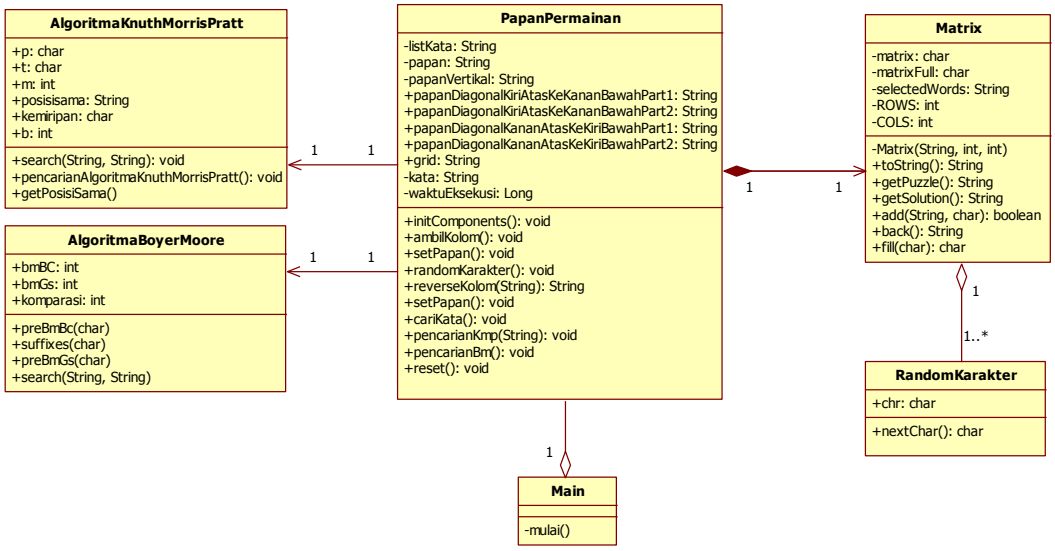
Class Diagram adalah inti dari proses permodelan objek. Class Diagram menggambarkan struktur dan hubungan antar objek-objek yang ada pada sistem. Struktur itu meliputi atribut dan method yang ada pada masing-masing *class*. Pada class diagram disusun menggunakan class diagram low level design yang terdiri dari *user interface layer* dan *backend layer*.

3.5.2.2.1 Class Diagram Graphical User Interface



Gambar 3.16 Class Diagram User Interface Layer

3.5.2.2.2 Class Diagram Backend



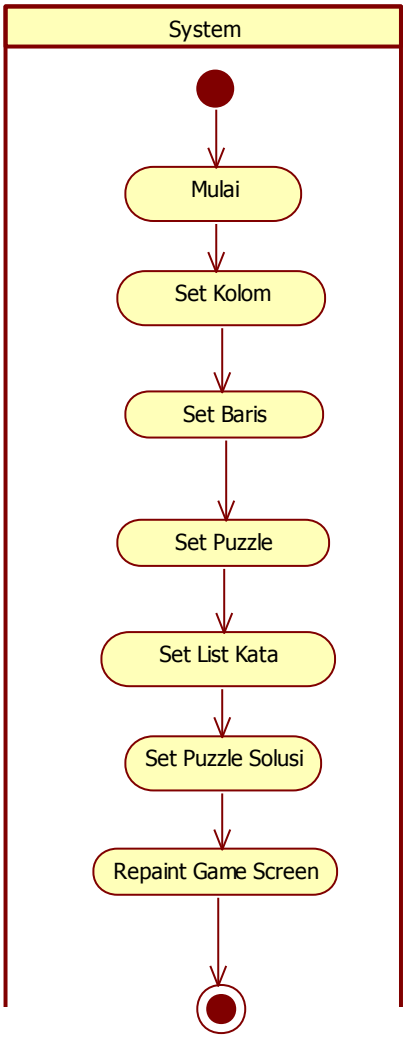
Gambar 3.17 Class Diagram Backend Layer

3.5.2.3. Activity Diagram

Acivity Diagram lebih memfokuskan diri pada eksekusi dan alur sistem dari pada bagaimana sistem itu dibuat. Activity Diagram menunjukan aktivitas sistem dalam bentuk kumpulan aksi-aksi. Berikut ini Activity Diagram pada permainan *word search puzzle*.

3.5.2.3.1. Activity Diagram Papan Permainan

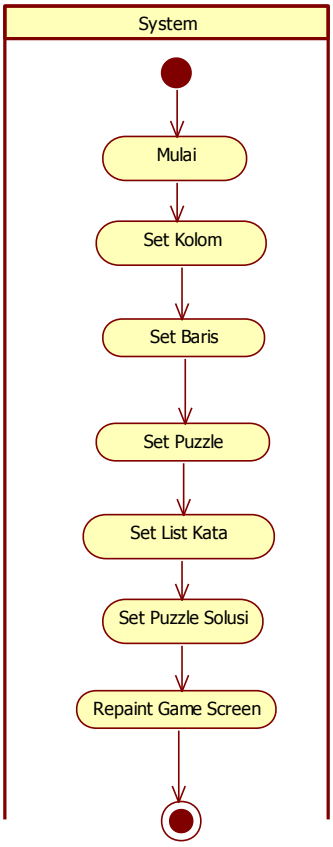
Berikut ini adalah diagram yang menujukan alur aksi pada aktivitas papan permainan yang dapat dilihat pada gambar 3.18 dibawah ini.



Gambar 3.18 Activity Diagram Papan Permainan

3.5.2.3.2. Activity Diagram Algoritma Knuth-Morris-Pratt

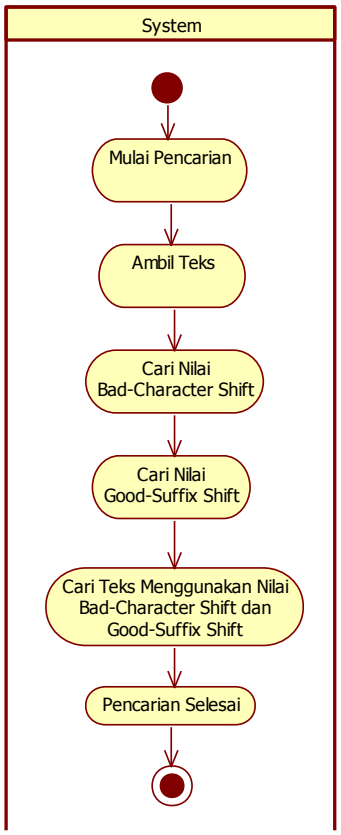
Berikut ini adalah diagram yang menunjukan alur aksi pada aktivitas algoritma *Knuth-Morris-Pratt* yang dapat dilihat pada gambar 3.19 dibawah ini.



Gambar 3.19 Activity Diagram Algoritma Knuth-Morris-Pratt

3.5.2.3.3. Activity Diagram Algoritma Boyer-Moore

Berikut ini adalah diagram yang menunjukan alur aksi pada aktivitas algoritma *Boyer-Moore* yang dapat dilihat pada gambar 3.20 dibawah ini.



Gambar 3.20 Activity Diagram Algoritma Boyer-Moore

3.5.2.3.4. Activity Diagram Matrix

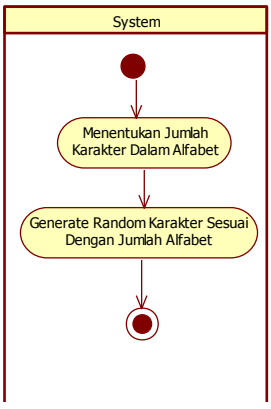
Berikut ini adalah diagram yang menunjukan alur aksi pada aktivitas matrix yang dapat dilihat pada gambar 3.21 dibawah ini.



Gambar 3.21 Activity Diagram Matrix

3.5.2.3.5. Activity Diagram Random Karakter

Berikut ini adalah diagram yang menunjukan alur aksi pada aktivitas random karakter yang dapat dilihat pada gambar 3.22 dibawah ini.

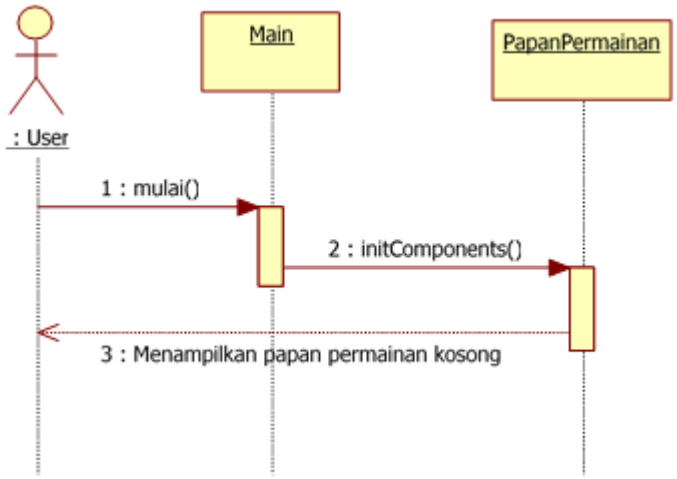


Gambar 3.22 Activity Diagram Random Karakter

3.5.2.4. Sequence Diagram

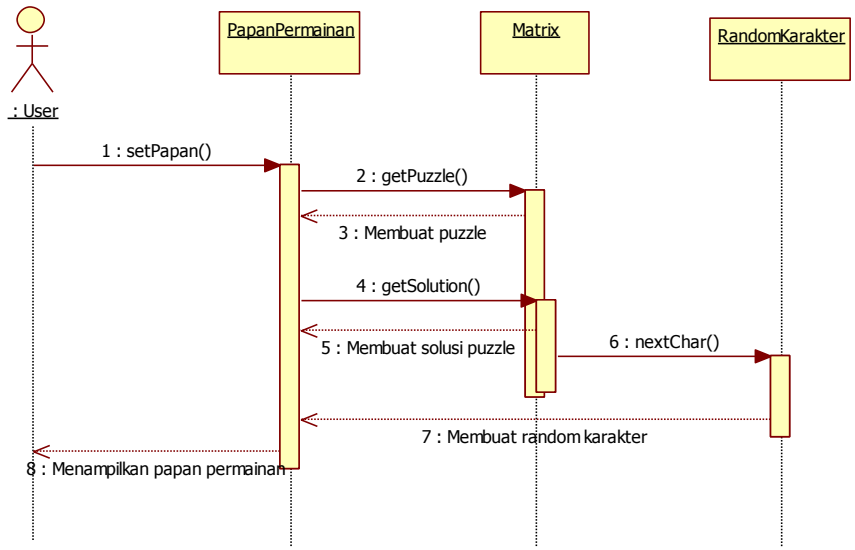
Sequence Diagram menggambarkan interaksi antar masing-masing objek pada setiap Use Case dalam urutan waktu. Interkasi ini berupa pengiriman serangkaian data antar objek-objek yang saling berinteraksi. Sequence Diagram pada game *word search puzzle* terdiri dari *Sequence Diagram* menampilkan papan permainan, *Sequence Diagram* menampilkan pola kata yang dicari, *Sequence Diagram* mencari pola kata dengan algoritma *Knuth-Morris-Pratt*, *Sequence Diagram* mencari pola kata dengan algoritma *Boyer-Moore*, *Sequence Diagram* menampilkan posisi dan waktu pencarian pola.

3.5.2.4.1. *Sequence Diagram* Menampilkan Papan Permainan



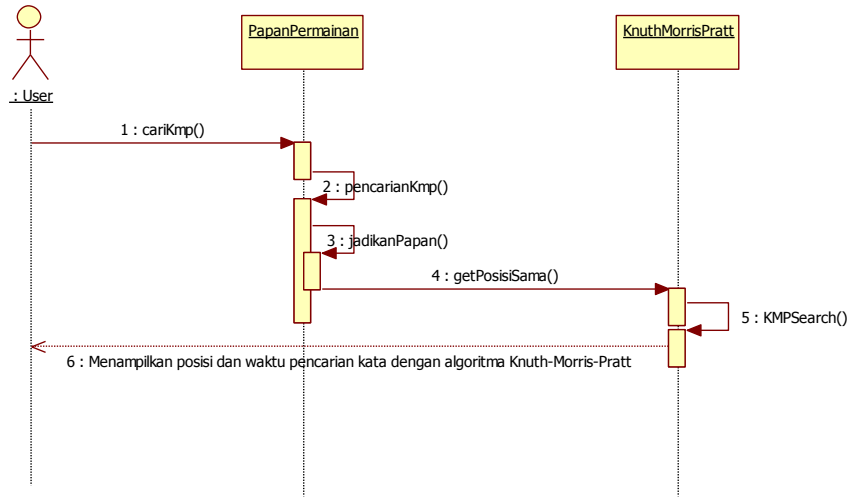
Gambar 3.23 Sequence Diagram Menampilkan Papan Permainan

3.5.2.4.2. *Sequence Diagram* Menampilkan Pola Kata yang Dicari



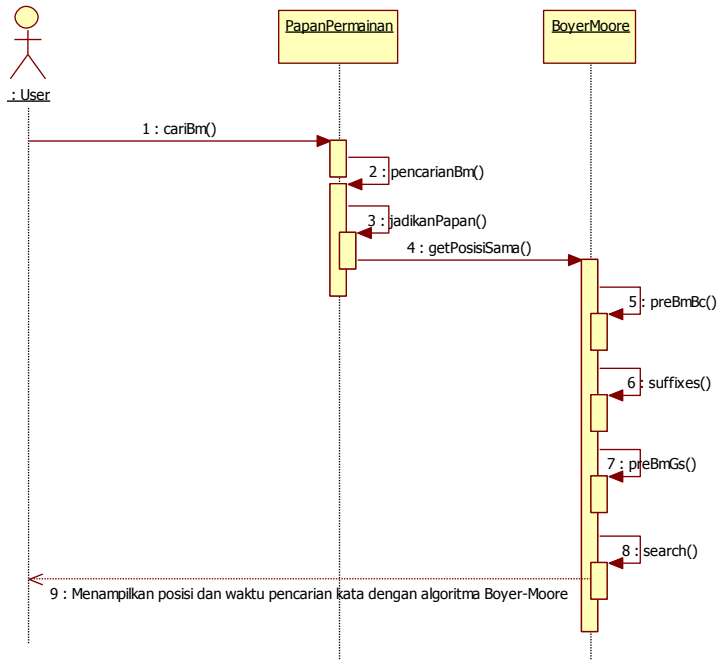
Gambar 3.24 Sequence Diagram Menampilkan Pola Kata Yang Dicari

3.5.2.4.3. *Sequence Diagram* Mencari Pola Kata dengan Algoritma *Knuth-Morris-Pratt*



Gambar 3.25 Sequence Diagram Cari Pola Kata Dengan Algoritma Knuth-Morris-Pratt

3.5.2.4.4. *Sequence Diagram* Mencari Pola kata dengan Algoritma *Boyer-Moore*



Gambar 3.26 Sequence Diagram Cari Pola Kata Dengan Algoritma Boyer-Moore

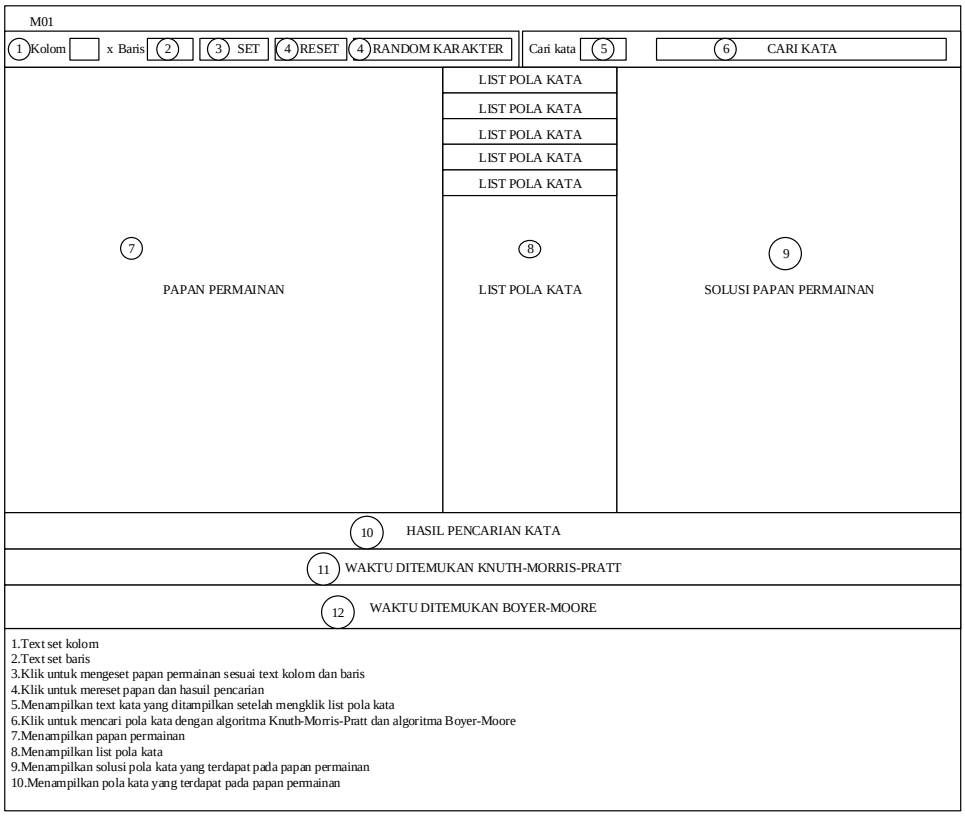
3.5. Analisis Perancangan Pengujian

Perancangan simulasi merupakan penggambaran, perencanaan, dan pembuatan sketsa atau pengaturan dari beberapa elemen yang terpisah kedalam suatu kesatuan yang utuh. Tahapan ini meliputi mengkonfigurasi komponen-komponen perangkat lunak dan perangkat keras dari suatu sistem. Adapun perancangan sistem dari perangkat lunak yang akan dibuat dijelaskan sebagai berikut.

3.6.1. Perancangan Arsitektur Perangkat Lunak Pengujian

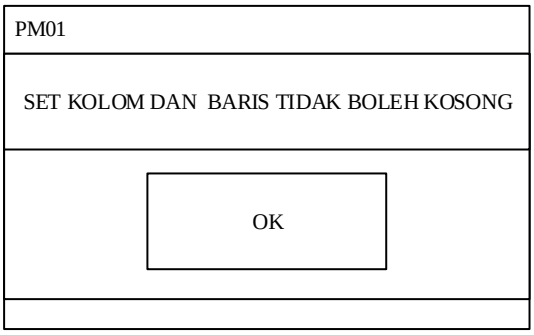
Perancangan arsitektur merupakan perancangan yang dibuat sebelum perangkat lunak pengujian dibuat. Perancangan arsitektur yang digunakan dalam perangkat lunak ini adalah program *desktop* yang dibuat menggunakan java.

3.6.1.1. Perancangan Antarmuka Papan Permainan



Gambar 3.27 Form Papan Permainan M01

3.6.1.2. Perancangan Antarmuka Pesan

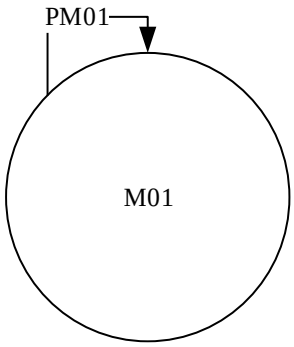


Gambar 3.28 Pesan PM01

3.6.1.3. Jaringan Semantik

Gambaran pengetahuan grafis yang menunjukan hubungan antar berbagai objek, terdiri dari lingkaran-lingkaran yang dihubungkan dengan

anak panah yang menunjukan objek dan informasi tentang objek-objek disebut jaringan semantik. Jaringan semantik dari permainan *word search puzzle*. Bisa dilihat pada gambar 3.24 dibawah ini.



Gambar 3.29 Jaringan Semantik

Keterangan :

M01 : Antarmuka papan permainan

PM01 : Antarmuka pesan permainan