BAB 4
IMPLEMENTASI DAN PENGUJIAN

4.1 Implementasi

Tahapan implementasi merupakan tahapan dimana aplikasi yang sudah dibuat dan dirancang lalu di uji kelayakannya. Bahasa pemograman dalam sistem ini menggunakan *Game Maker Language*. Selanjutnya dioperasikan sebagaimana mestinya sesuai fungsi dan kelayakannya untuk digunakan.

4.1.1 Perangkat Pendukung yang Digunakan

Dalam proses pembuatan aplikasi ini, tentunya membutuhkan perangkat keras (*hardware*) dan perangkat lunak (*software*). Berikut merupakan penjelasan dari perangkat keras dan perangkat lunak yang digunakan.

4.1.1.1 Perangkat Keras (*Hardware*)

Perangkat keras yang digunakan untuk membangun aplikasi ini memiliki spesifikasi sebagai berikut:

1. Prosesor dengan kecepatan *2.30GHz*
2. *Memory*: 4096MB
3. *Harddisk*: 600GB
4. *Video Card*: 2GB
5. *Keyboard* dan *Mouse*
6. Monitor
7. *Speaker*

4.1.1.2 Perangkat Lunak (*Software*)

Perangkat lunak yang digunakan untuk menjalankan aplikasi *game* War of Baratayuda ini adalah sistem operasi Windows 7

4.1.2 Implementasi Aplikasi

File yang ada pada *game* budaya "War of Baratayuda" ini, dibungkus menjadi satu dalam bentuk *Game* War of Baratayuda.rar, berikut langkah-langkah untuk menjalannkan aplikasi *game* ini:

1. *Extract* WarofBaratayuda.rar
2. Akan ada folder *Game* War of Baratayuda
3. *Double* Klik pada file WarOfBaratayuda.exe yang ada di folder *Game* War of Baratayuda
4. Akan tampil *window game* War of Baratayuda, dan itu menandakan bahwa *game* sukses dijalankan.

4.1.3 Implementasi Antar Muka

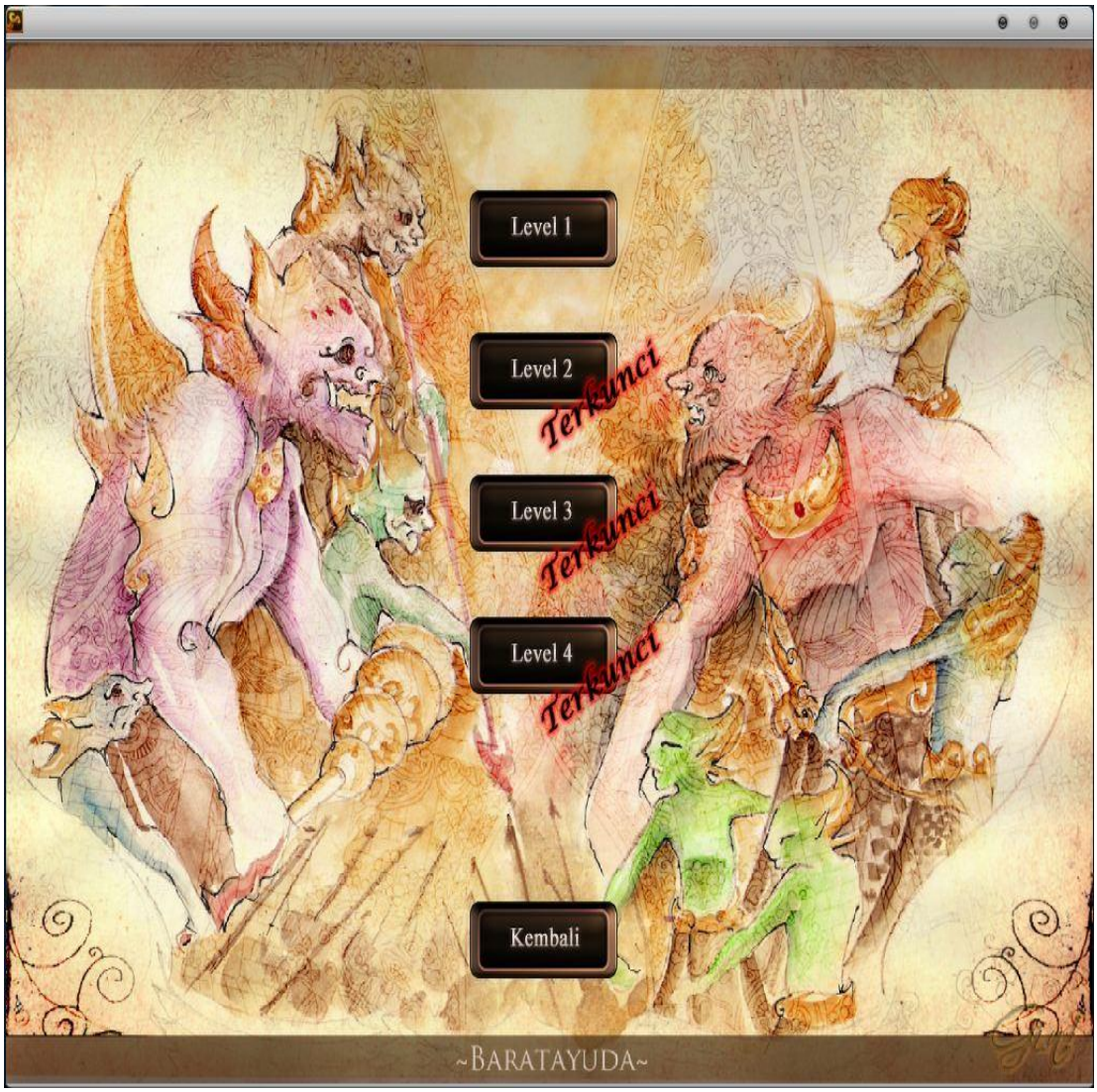
1. Tampilan Menu Utama



Gambar 4.1 Tampilan Menu Utama

Layar ini berfungsi menampilkan Tampilan Menu Utama atau tampilan awal *game* dimana didalamnya tersedia lima menu, yaitu menu Mulai, menu Pengaturan, menu Instruksi, menu Pembuat dan menu Keluar.

2. Tampilan Pilih Level



Gambar 4.2 Tampilan Pilih Level

Layar ini berfungsi menampilkan pemilihan level yang diinginkan oleh pemain (*user*), yang nantinya akan masuk kedalam permainan. Pada awal permainan Level 2, Level 3 dan Level 4 masih terkunci untuk dapat membukanya *user* harus bisa menyelesaikan misi di setiap level sebelumnya.

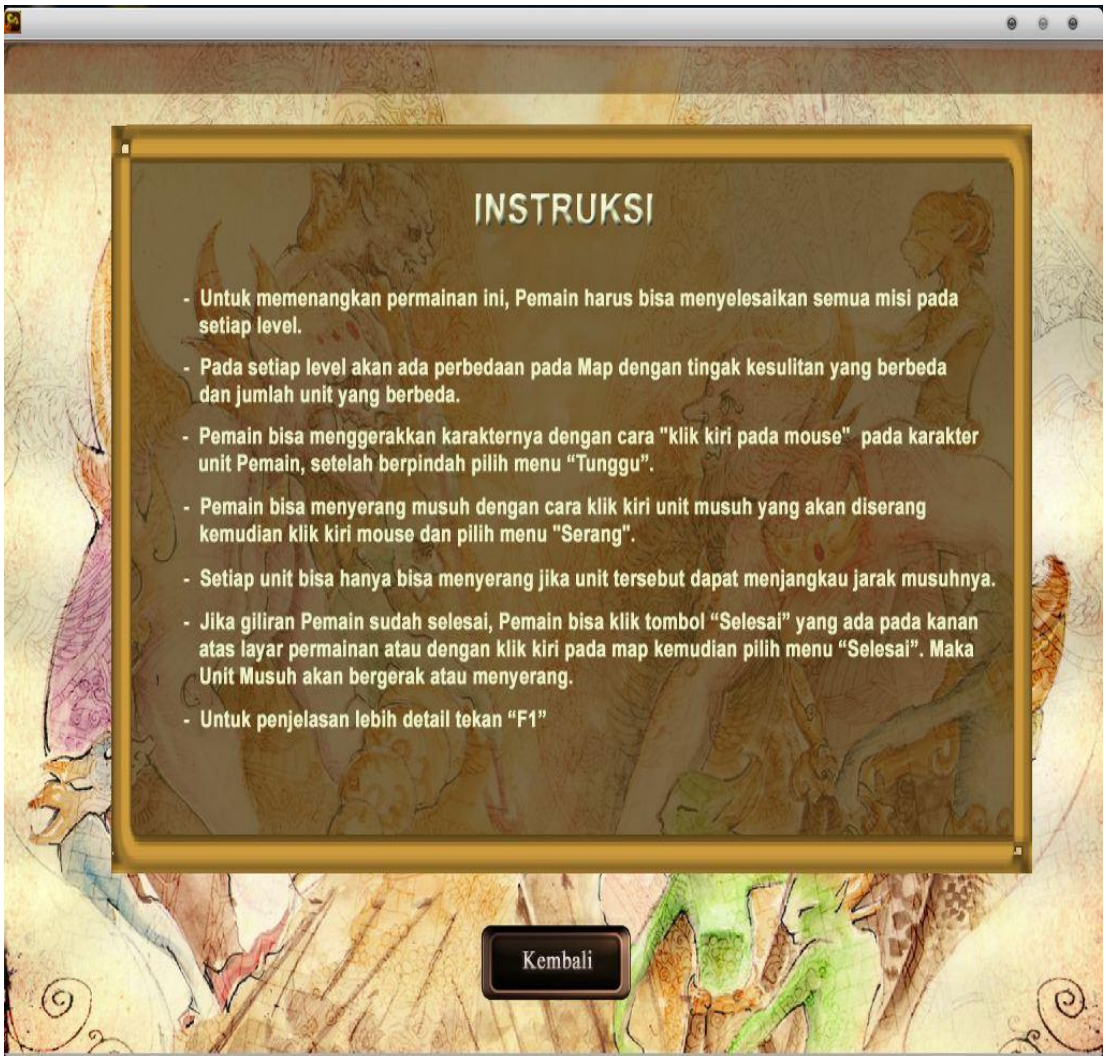
3. Tampilan Jalan Cerita Awal



Gambar 4.3 Tampilan Jalan Cerita

Layar ini berfungsi untuk menampilkan cerita pendek tentang sebab terjadinya perang baratayuda atau *storyline game*.

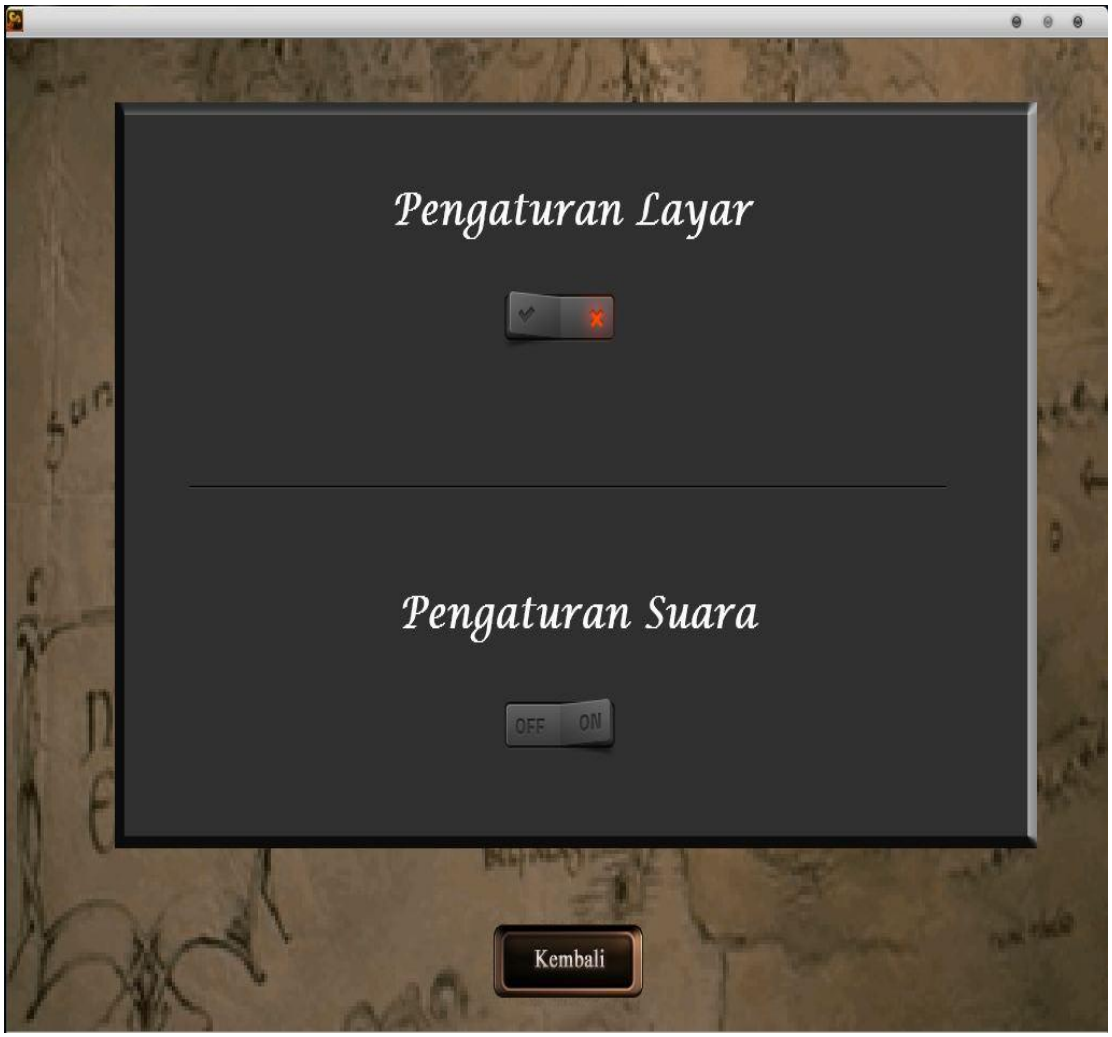
4. Tampilan Instruksi



Gambar 4.4 Tampilan Instruksi

Layar ini berfungsi menampilkan informasi mengenai petunjuk-petunjuk cara bermain dalam *game* War of Baratayuda.

5. Tampilan Pengaturan



Gambar 4.5 Tampilan Pengaturan

Layar ini berfungsi untuk mengatur layar, dan suara pada *game* War of Baratayuda.

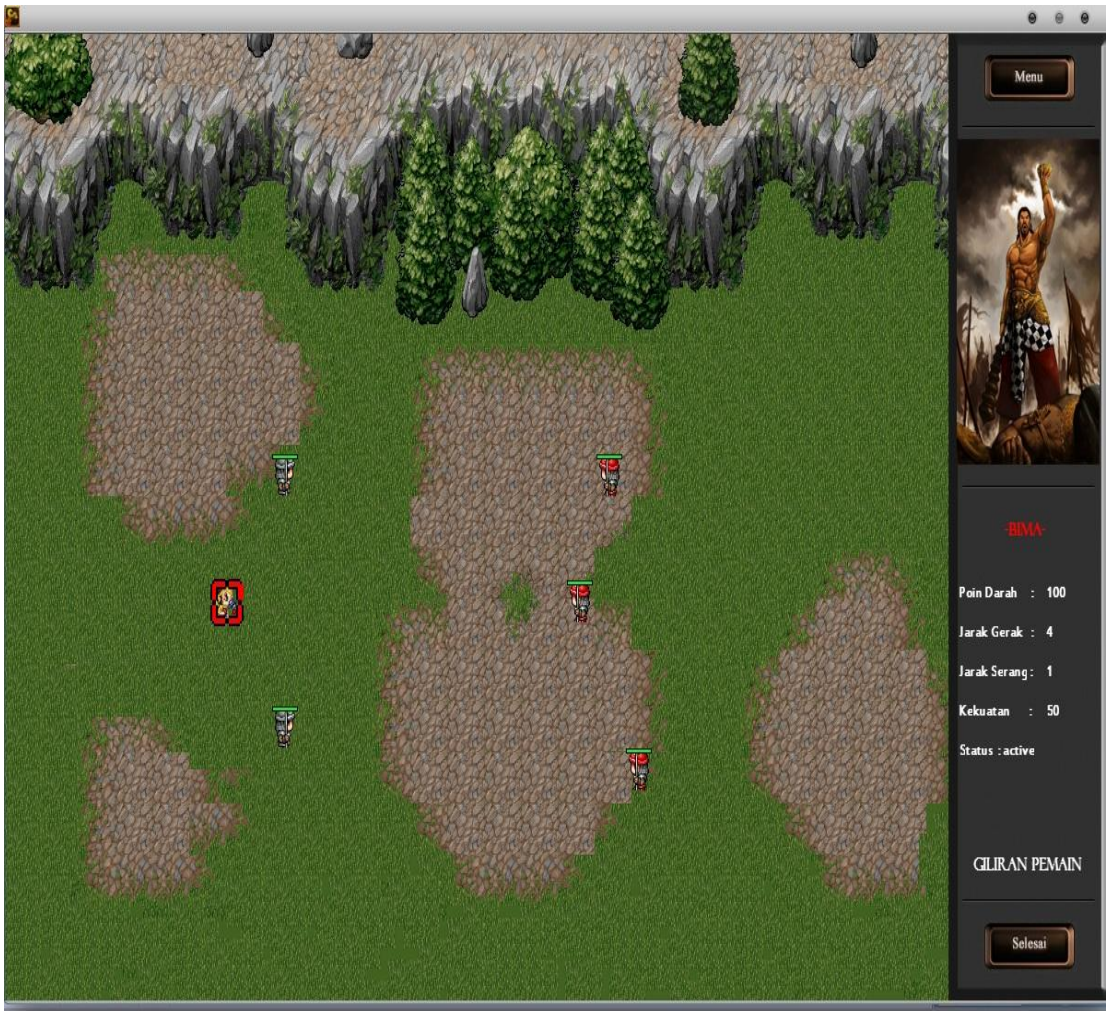
6. Tampilan Pembuat



Gambar 4.6 Tampilan Pembuat

Layar ini berfungsi menampilkan informasi mengenai ucapan dari pembuat, beserta sumber gambar, suara, dan lain-lain yang digunakan dalam *game* War of Baratayuda.

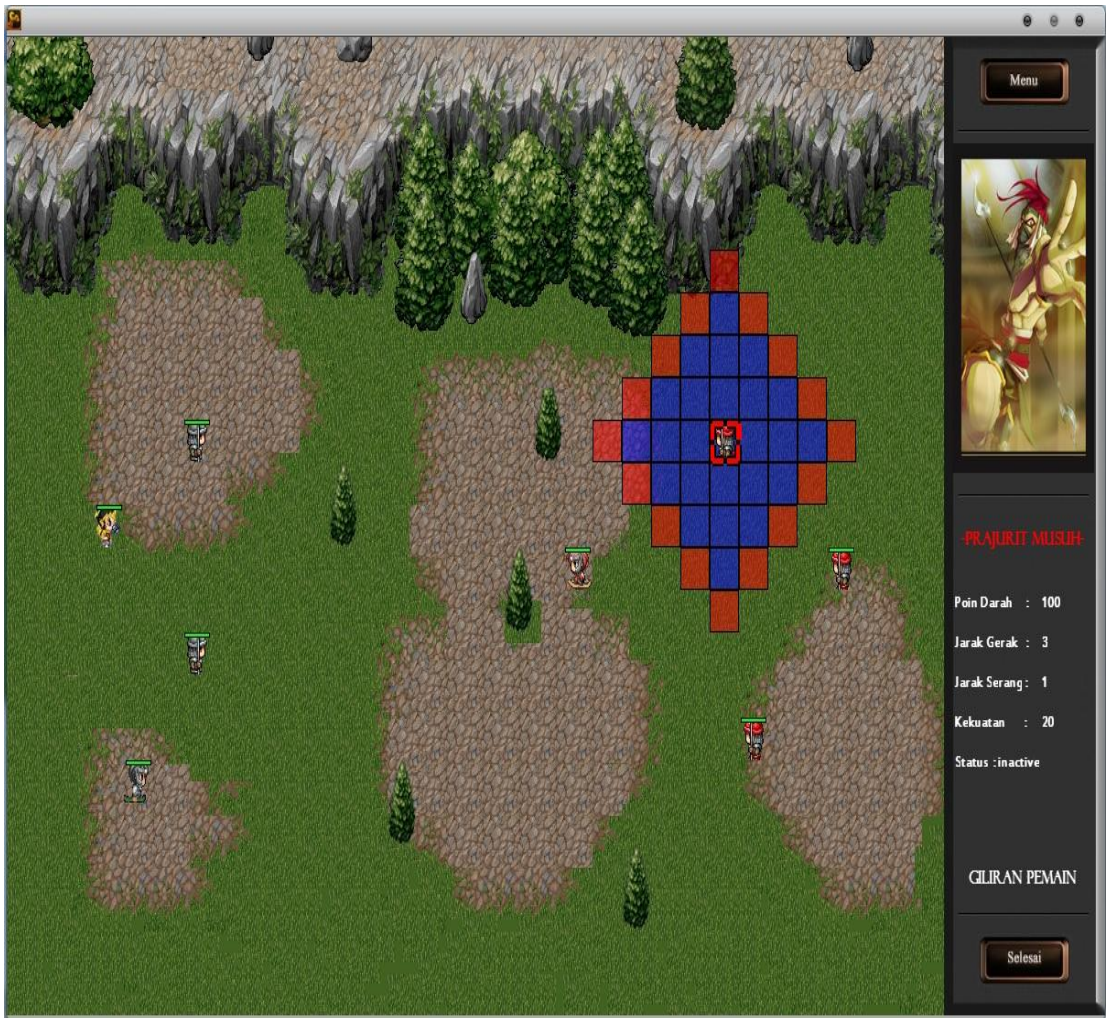
7. Tampilan Permainan Level 1



Gambar 4.7 Tampilan Permainan Level 1

Layar ini merupakan layar tampilan untuk permainan pada Level 1 dimana untuk *user* disediakan 2 unit *soldier* dan 1 unit *hero* Bima dan untuk enemy disediakan 3 unit *soldier*.

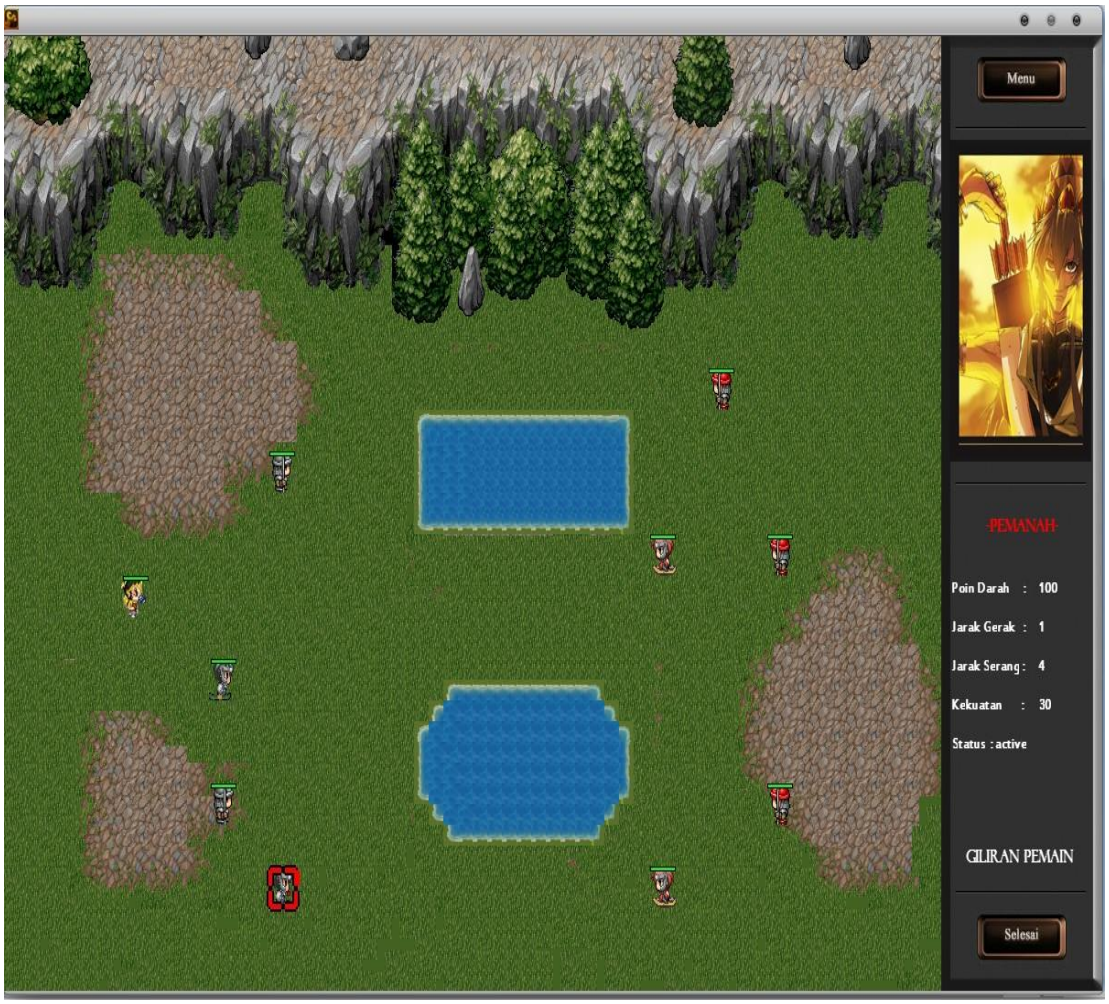
8. **T**ampilan Permainan Level 2



Gambar 4.8 Tampilan Permainan Level 2

Layar ini merupakan layar tampilan untuk permainan pada Level 2 dimana untuk *user* disediakan 2 unit soldier, 1 unit a*rcher* dan 1 unit hero Bima dan untuk *enemy* disediakan 3 unit *soldier* dan 1 unit *archer*.

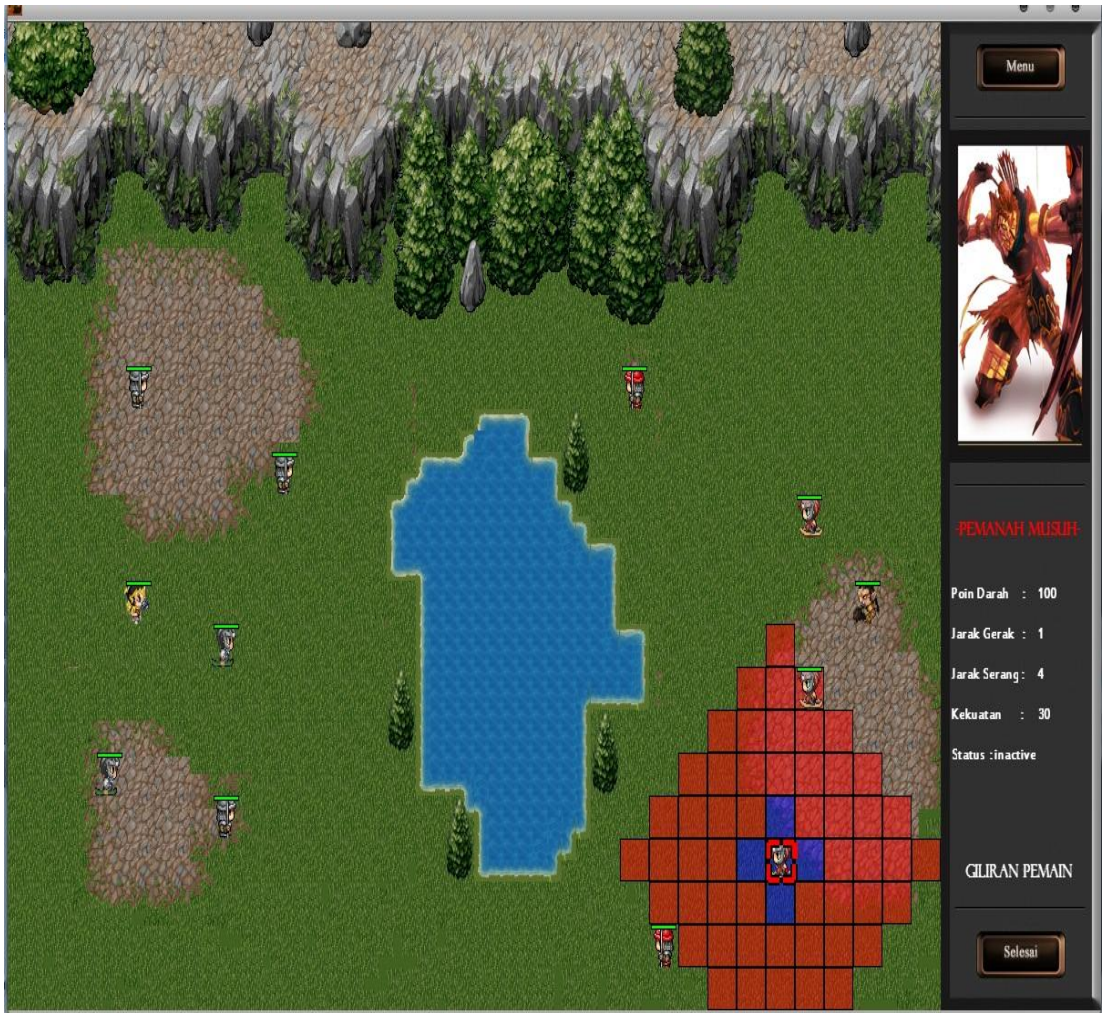
9. Tampilan Permainan Level 3



Gambar 4.9 Tampilan Permainan Level 3

Layar ini merupakan layar tampilan untuk permainan pada Level 3 dimana untuk *user* disediakan 2 unit *soldier*, 2 unit a*rcher* dan 1 unit *hero* Bima dan untuk *enemy* disediakan 3 unit *soldier* dan 2 unit *archer*.

10. Tampilan Permainan Level 4



Gambar 4.10 Tampilan Permainan Level 4

Layar ini merupakan layar tampilan untuk permainan pada Level 4 atau level terakhir pada *game* War of Baratayuda. Pada layar ini untuk *user* disediakan 3 unit *soldier*, 2 unit a*rcher* dan 1 unit *hero* Bima dan untuk *enemy* disediakan 2 unit *soldier* dan 3 unit *archer* dan 1 unit *hero* Duryudana.

11. Tampilan Jalan Cerita Akhir



Gambar 4.11 Tampilan Jalan Cerita Akhir

Layar ini merupakan layar tampilan yang menandakan bahwa pemain telah sukses menyelesaikan semua level pada permainan. Layar ini berisikan pesan edukatif mengenai cerita perang Baratayuda.

4.2 Pengujian Sistem

Pengujian sistem dilakukan untuk menemukan kekurangan atau kesalahan pada perangkat lunak yang diuji. Pengujian bermaksud untuk mengetahui perangkat lunak yang dibuat sudah memenuhi kriteria yang sesuai dengan tujuan perancangan perangkat lunak tersebut.

Pengujian yang dilakukan yaitu pengujian *alpha* dan pengujia *beta*. Pengujian *alpha* terdiri dari pengujian *white-box* dan *black-box*. Pengujian *white-box* digunakan untuk menguji sistem. Pengujian *black-box* digunakan untuk mengetahui kesalahan proses secara fungsional. Pengujian beta digunakan untuk mengetahui tanggapan user terhadap aplikasi *game*, dengan melakukan kuesioner.

4.2.1 Pengujian *Alpha*

Pengujian fungsional yang digunakan untuk menguji sistem yang baru adalah metode pengujian *alpha*. Pengujian *alpha* berfokus pada persyaratan fungsional perangkat lunak.

1. Rencana Pengujian

Rencana pengujian yang akan dilakukan pada aplikasi ini selengkapnya terlihat pada Tabel 4.1

Tabel 4.1 Rencana Pengujian Aplikasi

No.	Komponen yang Diuji	Butir Uji	Jenis Uji
1	Menu	Memilih Tombol Mulai	*Black Box*
		Memilih Tombol Instruksi	*Black Box*
		Memilih Tombol Pembuat	*Black Box*
		Memilih Tombol Pengaturan	*Black Box*
		Memilih Tombol Keluar	*Black Box*
		Memilih Tombol Level 1	*Black Box*
		Memilih Tombol Level 2	*Black Box*
		Memilih Tombol Level 3	*Black Box*

		Memilih Tombol Level 4	*Black Box*
		Memilih Tombol Lanjutkan	*Black Box*
		Memilih Tombol Main	*Black Box*
		Memilih Tombol Lewati	*Black Box*
		Memilih Tombol Menu	*Black Box*
		Memilih Tombol Selesai	*Black Box*
		Memilih Menu Selesai	*Black Box*
		Memilih Menu Batal	*Black Box*
		Memilih Menu Kembali	*Black Box*
		Memilih Menu Serang	*Black Box*
2	Pengaturan Suara	Menekan Tombol On	*Black Box*
		Menekan Tombol Off	*Black Box*
3	Pengaturan Layar	Menekan Tombol √	*Black Box*
		Menekan Tombol x	*Black Box*
4	Unit *User*	Unit Bergerak	*Black Box*
		Unit Menyerang	*Black Box*
5	Unit *Enemy*	Unit Bergerak (A* *pathfinding)*	*White Box*
		Unit Menyerang	*Black Box*
6	*Battle System*	Pengurangan HP pada Unit yang Diserang	*Black Box*
7	*Save System*	*Exit Game* Melakukan *Save*	*Black Box*
8	*Load System*	*Start Game* Melakukan *Load*	*Black Box*
9	*Lock and Unlock* Level	*Lock* Level 2, 3 dan 4	*Black Box*
		Unlock Level 2	*Black Box*
		Unlock Level 3	*Black Box*
		Unlock Level 4	*Black Box*
10	*Message Box Exit*	Menekan Tombol Ya	*Black Box*
		Menekan Tombol Tidak	*Black Box*

11	*Message Box Quit*	Menekan Tombol Ya	*Black Box*
		Menekan Tombol Tidak	*Black Box*

2. Kasus dan Hasil Pengujian

2.1 Pengujian *White Box*

Pengujian *white box* merupakan pengujian yang didasarkan pada pengecekan terhadap detail perancangan, menggunakan struktur *control* dari desain aplikasi secara prosedural untuk membagi pengujian ke dalam beberapa kasus pengujian untuk mengetahui apakah setiap node pada *independent paths* dikerjakan. Serta untuk menguji alur logika, perulangan, dan validasi parameter masukannya.

a. Algoritma A* *Pathfinding*

Berikut ini adalah algoritma A* untuk pergerakan *Enemy* yang penerapannya menggunakan bahasa pemrograman GML (*Game Maker Language*) :

```
1. with (objNode)
2. {
3. aStarParent = noone;
4. aStarF = 0;
5. aStarG = 0;
6. aStarH = 0;
7. }
8. var open, closed;
9. var start, finish, current, neighbor;
10. var tempF, tempG, i;
11. var checkOnlyMovement, checkPath;
12. start = argument0;
13. finish = argument1;
14. checkOnlyMovement = argument2;
15. checkPath = argument3;
16. open = ds_priority_create();
17. closed = ds_list_create();
18. ds_priority_add(open, start, start.aStarF);
```

```
19. current = start;
20. if (start == finish)
21. {
22. var path;
23. path = ds_stack_create();
24. ds_stack_push(path, start);
25. return path;
26. }
27. while (ds_priority_size(open) > 0 && current != finish)
28. {
29. current = ds_priority_delete_min(open);
30. ds_list_add(closed, current);
31. for (i = 0; i < ds_list_size(current.neighbors); i += 1)
32. {
33. neighbor = ds_list_find_value(current.neighbors, i);
34. if (checkOnlyMovement && !neighbor.isMovementNode)
35. continue;
36. if (checkPath && ds_priority_find_priority(cursor.path, neighbor) != 0)
37. continue;
38. if (ds_list_find_index(closed, neighbor) < 0 && neighbor.passable)
39. {
40. if (ds_priority_find_priority(open, neighbor) == 0)
41. {
42. neighbor.aStarParent = current;
43. neighbor.aStarG = current.aStarG + neighbor.cost;
44. neighbor.aStarH = heuristic(current, neighbor);
45. neighbor.aStarF = neighbor.aStarG + neighbor.aStarH;
46. ds_priority_add(open, neighbor, neighbor.aStarF);
47. }
48. else
49. {
50. tempG = current.aStarG + neighbor.cost;
51. tempF = neighbor.aStarG + neighbor.aStarH;
52. if (tempF < neighbor.aStarF)
53. {
54. neighbor.aStarParent = current;
55. neighbor.aStarG = tempG;
```

```
56. neighbor.aStarF = tempF;
57. ds_priority_change_priority(open, neighbor, neighbor.aStarF);
58. }
59. }
60. }
61. }
62. }
63. ds_priority_destroy(open);
64. var path;
65. path = ds_stack_create();
66. current = finish;
67. while (current != start && current != noone)
68. {
69. ds_stack_push(path, current);
70. current = current.aStarParent;
71. }
```

b. *Flow Graph* A* *Pathfinding*

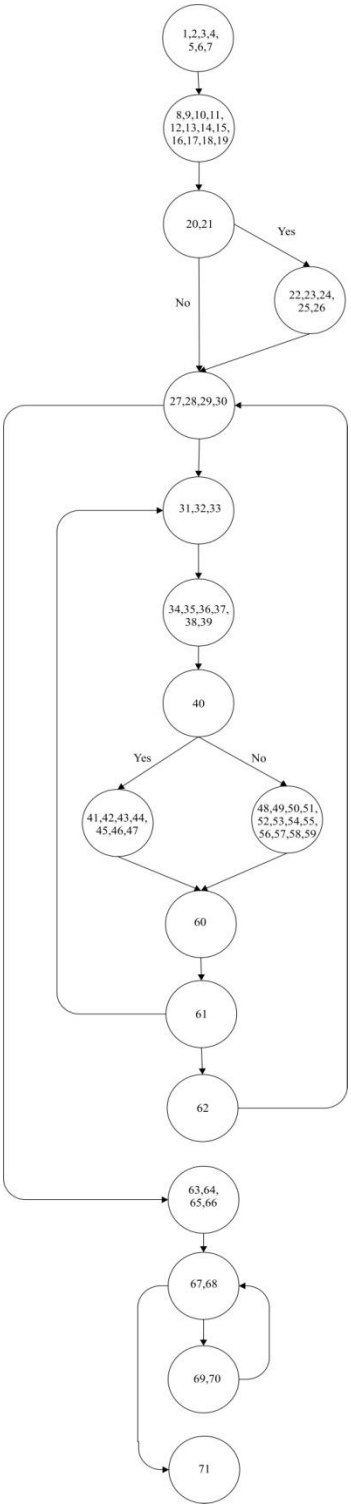
Berdasarkan *pseudocode* di atas maka *flow graph* algoritma A* adalah sebagai berikut:

Keterangan:



= Menggambarkan kondisi

= Menggambarkan aksi



Gambar 4.12 ***Flow Graph*** **A*** ***Pathfinding***

c. *Cyclomatic Complexity*

E = 20, N = 16

V(G) = E – N + 2

V(G) = 20 – 16 + 2

V(G) = 6

Keterangan :

E : Jumlah busur pada *flow graph*

N : Jumlah simpul (*node*) pada *flow graph*

V(G) : *Cyclomatic Complexity*

d. *Independent Path*

Path 1 : 1-2-3-4-5-6-7-8-9-10-11-15-16-17-18-19-20-21-22-23-24-25-26-27-28-29-30-63-64-65-66-67-68-71

Path 2 : 1-2-3-4-5-6-7-8-9-10-11-15-16-17-18-19-20-21-27-28-29-30-63-64-65-66-67-68-71

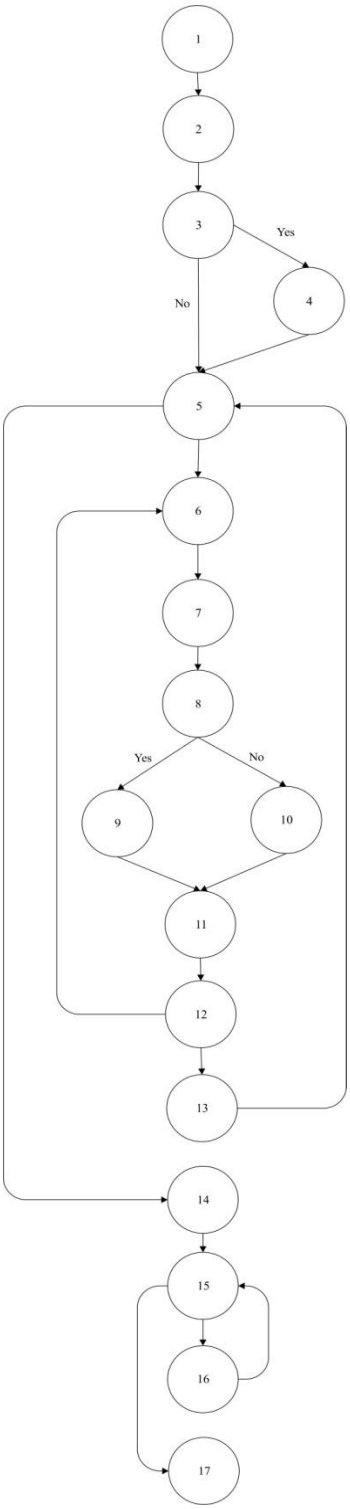
Path 3 : 1-2-3-4-5-6-7-8-9-10-12-13-14-15-16-17-18-19-20-21-27-28-29-30-31-32-33-34-35-36-37-39-40-41-42-43-44-45-46-47-60-61-62-27-28-29-30-63-64-65-66-67-68-69-70-67-68-71

Path 4 : 1-2-3-4-5-6-7-8-9-10-12-13-14-15-16-17-18-19-20-21-27-28-29-30-31-32-33-34-35-36-37-39-40-48-49-50-51-52-53-54-55-56-57-58-59-60-61-62-27-28-29-30-63-64-65-66-67-68-69-70-67-68-71

Path 5 : 1-2-3-4-5-6-7-8-9-10-12-13-14-15-16-17-18-19-20-21-27-28-29-30-31-32-33-34-35-36-37-39-40-41-42-43-44-45-46-47-60-61-31-32-33-34-35-36-37-39-40-48-49-50-51-52-53-54-55-56-57-58-59-60-61-62-27-28-29-30-63-64-65-66-67-68-71

Path 6 : 1-2-3-4-5-6-7-8-9-10-12-13-14-15-16-17-18-19-20-21-27-28-29-30-31-32-33-34-35-36-37-39-40-41-42-43-44-45-46-47-60-61-31-32-33-34-35-36-37-39-40-48-49-50-51-52-53-54-55-56-57-58-59-60-61-62-27-28-29-30-63-64-65-66-67-68-69-70-67-68-71

e. *Graph Matrices Testing*



Gambar 4.13 *Flow Graph* Setelah Penomoran Ulang

Tabel 4.2 *Graph Matrices Testing*

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	N(E)-1
1		1																0
2			1															0
3				1	1													1
4					1													0
5						1								1				1
6							1											0
7								1										0
8									1	1								1
9											`1							0
10											1							0
11												1						0
12						1							1					1
13					1													0
14															1			0
15																1	1	1
16															1			0
17																		0
Jumlah																		5

V(G) = Jumlah *graph matrices* + 1

V(G) = 5+1

V(G) = 6

f. Hasil Pengujian

Tabel 4.3 Hasil Pengujian *White Box*

No. Path	Butir Uji	Node (n)	Pengamatan	Kesimpulan
1	if (start == finish), *node* tujuan merupakan *node* awal	1-2-3-4-5-6-7-8-9-10-11-15-16-17-18-19-20-21-22-23-24-25-26-27-28-29-30-63-	Unit enemy tidak berpindah tempat	Path dikerjakan

		64-65-66-67-68-71		
2	while (ds_priority_size(open) > 0 && current != finish) , memindahkan node dari *open* ke *closed*	1-2-3-4-5-6-7-8-9-10-11-15-16-17-18-19-20-21-27-28-29-30-63-64-65-66-67-68-71	Node dengan nilai F terkecil dipindahkan ke *closed*	Path dikerjakan
3	for (i = 0; i < ds_list_size(current.neighbors); i += 1), mencari nilai F terkecil dari setiap *node* yang berada di *open*	1-2-3-4-5-6-7-8-9-10-12-13-14-15-16-17-18-19-20-21-27-28-29-30-31-32-33-34-35-36-37-39-40-41-42-43-44-45-46-47-60-61-31-32-33-34-35-36-37-39-40-48-49-50-51-52-53-54-55-56-57-58-59-60-61-62-27-28-29-30-63-64-65-66-67-68-71	Node dengan nilai F terkecil ditemukan	Path dikerjakan
4	if (ds_priority_find_priority(open, neighbor) == 0), kalkulasi nilai F dan masukan ke *open*	1-2-3-4-5-6-7-8-9-10-12-13-14-15-16-17-18-19-20-21-27-28-29-30-31-32-33-34-35-36-37-39-40-41-42-43-44-45-46-47-60-61-62-27-28-29-30-63-64-65-66-67-68-69-70-67-68-71	Memasukan nilai F ke dalam open	Path dikerjakan
5	if (ds_priority_find_priority(open, neighbor) != 0), kalkulasi ulang nilai F dan bandingkan nilai F	1-2-3-4-5-6-7-8-9-10-12-13-14-15-16-17-18-19-20-21-27-28-29-30-31-32-33-34-35-36-37-39-40-48-	Mendapatkan nilai F yang baru	Path dikerjakan

	tersebut dengan nilai F yang sudah berada di *open*	49-50-51-52-53-54-55-56-57-58-59-60-61-62-27-28-29-30-63-64-65-66-67-68-69-70-67-68-71		
6	while (current != start && current != noone) { ds_stack_push(path, current); current = current.aStarParent; }, rute ditemukan	1-2-3-4-5-6-7-8-9-10-12-13-14-15-16-17-18-19-20-21-27-28-29-30-31-32-33-34-35-36-37-39-40-41-42-43-44-45-46-47-60-61-31-32-33-34-35-36-37-39-40-48-49-50-51-52-53-54-55-56-57-58-59-60-61-62-27-28-29-30-63-64-65-66-67-68-69-70-67-68-71	Rute ditemukan	Path dikerjakan

Pada pengujian pertama atau *path* nomor 1 pada tabel 4.3 di atas, merupakan path dimana unit musuh (NPC) tidak bergerak atau berpindah tempat karena node tujuan merukan *node* awal unit musuh. Hasil yang didapat adalah unit musuh tidak berpindah tempat maka *path* dikerjakan.

Pada pengujian pada path nomor 2, merupakan *path* untuk menguji pengulangan nilai F terkecil yang dimasukkan ke *closed list*. hasil yang didapat adalah *path* dikerjakan, *node* dengan nilai F terkecil dipindahkan ke *closed list.* Pengulangan pada algoritma tidak terjadi pengulangan tak terhingga.

Pada pengujian *path* nomor 3, merupakan pengujian untuk mengecek pengulangan pencarian nilai F terkecil pada setiap *node* yang berada pada open list. Pengamatan yang didapat adalah node dengan nilai F terkecil ditemukan dan tidak terjadi pengulangan tak terhingga maka *path* dikerjakan.

Pada pengujian path nomor 4, merupakan path untuk menguji apakah *node* suksesor sudah berada pada *open list* atau belum, jika belum masukkan *node* tersebut ke dalam *open list* dan simpan nilai F setiap *node* suksesor. Pengamatan yang didapat nilai F pada setiap *node* suksesor dimasukkan ke *open list* maka *path* dikerjakan.

Pengujian pada nomor 5, merupakan *path* untuk menguji apakah *node* suksesor sudah berada pada *open list* atau belum, jika sudah lakukan pengecekan apakah nilai F pada *node* tersebut lebih baik atau kurang dari nilai F dari *current node*. Jika ya, ganti nilai F sebelumnya dengan yang baru. Pengamatan yang didapat adalah mendapatkan nilai F yang baru maka *path* dikerjakan.

Pengujian terakhir yaitu pengujian nomor 6, merupakan *path* untuk menguji pengulangan untuk mendapatkna rute yang dihasilkan dari algoritma A*. Pengamatan yang didapat adalah rute ditemukkan atau unit musuh bergerak maka *path* dikerjakan.

g. Kesimpulan

Dari hasil pengujian algoritma A* yang telah dilakukan, maka dapat disimpulkan bahwa seluruh *node* pada setiap *independent path* telah dikerjakan dan tidak ada pengulangan tak terhingga.

2.2 Pengujian *Black Box*

Berikut ini adalah kasus untuk menguji perangkat lunak yang sudah dibangun menggunakan metode *Black Box* berdasarkan Tabel 4.1 rencana pengujian yang telah dibuat sebelumnya.

Tabel 4.4 Kasus dan Hasil Uji

No.	Kasus / Diuji	Skenario Uji	Hasil yang Diharapkan	Hasil Pengujian
1	Menu	Memilih Tombol Mulai	Menampilkan menu pemilihan level	[√] Berhasil [ ] Tidak Berhasil
		Memilih Tombol	Menampilkan menu instruksi	[√] Berhasil

		Instruksi	dalam bermain	[ ] Tidak Berhasil
		Memilih Tombol Pembuat	Menampilkan menu pembuat *game*	[√] Berhasil [ ] Tidak Berhasil
		Memilih Tombol Pengaturan	Menampilkan menu pengaturan *game*	[√] Berhasil [ ] Tidak Berhasil
		Memilih Tombol Keluar	Menampilkan pesan *exit game*	[√] Berhasil [ ] Tidak Berhasil
		Memilih Tombol Level 1	Menampilkan halaman jalan cerita *game*	[√] Berhasil [ ] Tidak Berhasil
		Memilih Tombol Level 2	Menampilkan halaman permainan level 2	[√] Berhasil [ ] Tidak Berhasil
		Memilih Tombol Level 3	Menampilkan halaman permainan level 3	[√] Berhasil [√] Tidak Berhasil
		Memilih Tombol Level 4	Menampilkan halaman permainan level 4	[√] Berhasil [ ] Tidak Berhasil
		Memilih Tombol Lanjutkan	Melanjutkan jalan cerita *game*	[√] Berhasil [ ] Tidak Berhasil
		Memilih Tombol Main	Menampilkan halaman permainan level 1	[√] Berhasil [ ] Tidak Berhasil
		Memilih Tombol Lewati	Menampilkan halaman permainan level 1	[√] Berhasil [ ] Tidak Berhasil
		Memilih Tombol Menu	Menampilkan pesan *quit* dari permainan	[√] Berhasil [ ] Tidak Berhasil
		Memilih Tombol Selesai	Mengganti giliran kepada *enemy*	[√] Berhasil [ ] Tidak Berhasil
		Memilih Menu Selesai	Memilih langkah yang telah dipilih oleh *user*	[√] Berhasil [ ] Tidak Berhasil
		Memilih Menu Batal	Membatalkan pemilihan unit untuk melakukan aksinya	[√] Berhasil [ ] Tidak Berhasil

		Memilih Menu Kembali	Membatalkan langkah yang diambil untuk bergerak	[√] Berhasil [ ] Tidak Berhasil
		Memilih Menu Serang	Memampilkan halaman *battle* dan menampilkan besarnya *damage* unit yang menyerang	[√] Berhasil [ ] Tidak Berhasil
2	Pengaturan Suara	Menekan Tombol On	Memainkan semua musik pada setiap halaman pada *game*	[√] Berhasil [ ] Tidak Berhasil
		Menekan Tombol Off	Mematikan semua musik pada setiap halaman pada *game*	[√] Berhasil [ ] Tidak Berhasil
3	Pengaturan Layar	Menekan Tombol √	Membuat layar permainan menjadi *fullscreen*	[√] Berhasil [ ] Tidak Berhasil
		Menekan Tombol x	Membuat layar menjadi *window*	[√] Berhasil [ ] Tidak Berhasil
4	Unit *User*	Unit Menyerang	Ketika unit musuh ada dalam jarak serang unit *user* dan *user* memilih menu Serang, maka akan menampilkan halaman *battle* dan menampilkan *damage* unit yang menyerang	[√] Berhasil [ ] Tidak Berhasil
5	Unit *Enemy*	Unit Bergerak	Unit *enemy* akan bergerak mengikuti unit *user* yang terdekat	[√] Berhasil [ ] Tidak Berhasil
		Unit Menyerang	Unit *enemy* akan menyerang unit *user* yang terdekat	[√] Berhasil [ ] Tidak Berhasil
6	*Battle System*	Pengurangan HP pada Unit yang Diserang	Ketika salah satu unit *enemy* atau *user* menyerang maka unit yang diserang akan	[√] Berhasil [ ] Tidak Berhasil

			mengalami pengurangan HP sebanyak *damage* unit yang menyerang	
7	*Save System*	*Exit Game* Melakukan *Save*	Ketika user melakukan *exit game*, maka sistem akan otomatis melakukan *save* pada level yang telah berhasil diselesaikan	[√] Berhasil [ ] Tidak Berhasil
8	*Load System*	*Start Game* Melakukan *Load*	Ketika user memainkan aplikasi *game* ini, maka sistem akan otomatis melakukan load pada level yang telah berhasil diselesaikan	[√] Berhasil [ ] Tidak Berhasil
9	*Lock and Unlock* Level	*Lock* Level 2, 3 dan 4	*User* baru pertama kali memainkan aplikasi *game* ini, maka tombol Level 2, 3 dan 4 dalam status terkunci atau tidak bisa dimainkan	[√] Berhasil [ ] Tidak Berhasil
		Unlock Level 2	*User* telah berhasil menyelesaikan misi pada level 1, maka akan membuka Level 2	[√] Berhasil [ ] Tidak Berhasil
		Unlock Level 3	*User* telah berhasil menyelesaikan misi pada level 2, maka akan membuka Level 3	[√] Berhasil [ ] Tidak Berhasil
		Unlock Level 4	*User* telah berhasil menyelesaikan misi pada level 3, maka akan membuka Level 4	[√] Berhasil [ ] Tidak Berhasil

10	*Message Box Exit*	Menekan Tombol Ya	Keluar dari aplikasi *game*	[√] Berhasil [ ] Tidak Berhasil
		Menekan Tombol Tidak	Kembali ke Menu Utama (*Main Menu*)	[√] Berhasil [ ] Tidak Berhasil
11	*Message Box Quit*	Menekan Tombol Ya	Menampilkan halaman Menu Utama (*Main Menu*)	[√] Berhasil [ ] Tidak Berhasil
		Menekan Tombol Tidak	Kembali ke halaman permainan pada level yang sedang dimainkan	[√] Berhasil [ ] Tidak Berhasil

4.2.2 Pengujian Beta

Pengujian beta merupakan pengujian yang dilakukan secara objektif dimana diuji secara langsung ke lapangan, dengan menggunakan kuesioner mengenai tanggapan user terhadap *game* yang telah dibangun.

4.2.2.1 Evaluasi dari Pengguna

Untuk mengetahui sejauh mana sistem yang dibangun dapat menjadi alternatif penyelesaian permasalahan yang telah dijelaskan di latar belakang, telah disebarkan kuesioner kepada 30 responden masyarakat umum berdasarkan pada sasaran user. Teknik pengujian yang digunakan dalam pengujian beta ini adalah teknik *skala likert*.

a. Penafsiran Bobot Pengujian

Pengujian menggunakan 5 kategori jawaban dengan bobot yang berbeda pada setiap jawabannya seperti pada tabel berikut.

Tabel 4.5 Bobot Jawaban

Pernyataan	Sangat Tidak Setuju	Tidak Setuju	Cukup	Setuju	Sangat Setuju
Bobot	1	2	3	4	5

b. Kuesioner terhadap Masyarakat Umum

Data hasil kuisioner yang telah diberikan dapat dicari presentasinya dari masing-masing jawaban dengan menggunakan rumus :

$$P = \frac{S}{skorideal} x100\%$$

Keterangan :

P = Nilai persentase yang dicari

S = Jumlah frekuensi dikalikan dengan nilai yang di tetapkan jawaban

Skor ideal = Nilai tertinggi dikalikan dengan jumlah sampel

Berikut ini adalah hasil persentase masing-masing jawaban yang sudah dihitung nilainya dengan menggunakan rumus diatas.

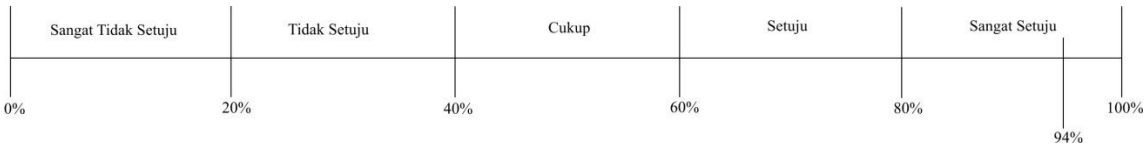
Hasil Kuesioner:

1. Apakah anda setuju tampilan *game* ini menarik?

Tabel 4.6 Perhitungan Persentase Pertanyaan Kesatu

Pertanyaan	Jawaban	Skor	Frekuensi	S
1.	Sangat Tidak Setuju	1	0	0
	Tidak Setuju	2	0	0
	Cukup	3	2	6
	Setuju	4	5	20
	Sangat Setuju	5	23	115
Jumlah				141

$$P = \frac{141}{150} x100 = 94\%$$



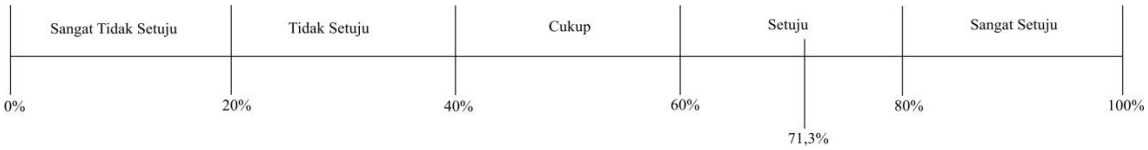
Gambar 4.14 Diagram Hasil Perhitungan Kuesioner Pertanyaan Kesatu

2. Apakah dengan adanya *game* ini, anda lebih tertarik untuk mengenal budaya Indonesia?

Tabel 4.7 Perhitungan Persentase Pertanyaan Kedua

Pertanyaan	Jawaban	Skor	Frekuensi	S
2.	Sangat Tidak Setuju	1	0	0
	Tidak Setuju	2	4	8
	Cukup	3	10	30
	Setuju	4	11	44
	Sangat Setuju	5	5	25
Jumlah				107

$$P = \frac{107}{150} x100 = 71{,}3\%$$



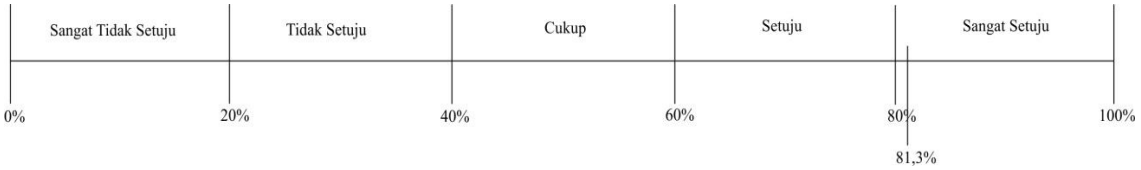
Gambar 4.15 Diagram Hasil Perhitungan Kuesioner Pertanyaan Kedua

3. Apakah dengan adanya *game* ini, anda jadi mengenal/lebih mengenal cerita perang baratayuda?

Tabel 4.8 Perhitungan Persentase Pertanyaan Ketiga

Pertanyaan	Jawaban	Skor	Frekuensi	S
3.	Sangat Tidak Setuju	1	0	0
	Tidak Setuju	2	1	2
	Cukup	3	6	18
3	Setuju	4	13	52
	Sangat Setuju	5	10	50
Jumlah				122

$$P = \frac{122}{150} x100 = 81{,}3\%$$



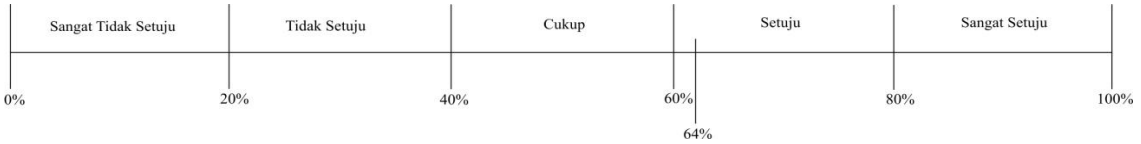
Gambar 4.16 Diagram Hasil Perhitungan Kuesioner Pertanyaan Ketiga

4. Apakah anda menangkap pesan edukatif yang terkandung dalam cerita *game*?

Tabel 4.9 Perhitungan Persentase Pertanyaan Keempat

Pertanyaan	Jawaban	Skor	Frekuensi	S
4.	Sangat Tidak Setuju	1	0	0
	Tidak Setuju	2	4	8
	Cukup	3	18	54
	Setuju	4	6	24
	Sangat Setuju	5	2	10
Jumlah				96

$$P = \frac{96}{150} x100 = 64\%$$



Gambar 4.17 Diagram Hasil Perhitungan Kuesioner Pertanyaan Keempat

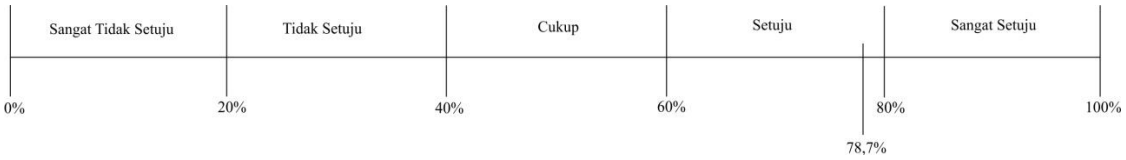
5. Apakah petunjuk yang terdapat pada *game* War of Baratayuda ini membantu anda dalam memainkan *game*?

Tabel 4.10 Perhitungan Persentase Pertanyaan Kelima

Pertanyaan	Jawaban	Skor	Frekuensi	S
5.	Sangat Tidak Setuju	1	0	0
	Tidak Setuju	2	0	0
	Cukup	3	11	33
	Setuju	4	10	40
	Sangat Setuju	5	9	45

Jumlah		118

$$P = \frac{118}{150} x100 = 78{,}7\%$$



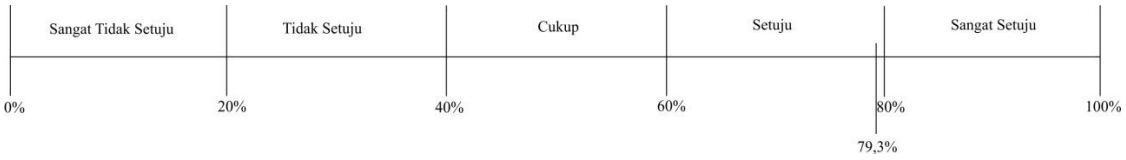
Gambar 4.18 Diagram Hasil Perhitungan Kuesioner Pertanyaan Kelima

6. Apakah anda kesulitan dalam menyelesaikan setiap misi yang ada pada *game* ini?

Tabel 4.11 Perhitungan Persentase Pertanyaan Keenam

Pertanyaan	Jawaban	Skor	Frekuensi	S
6.	Sangat Tidak Setuju	1	0	0
	Tidak Setuju	2	1	2
	Cukup	3	6	18
	Setuju	4	16	64
	Sangat Setuju	5	7	35
Jumlah				119

$$P = \frac{119}{150} x100 = 79{,}3\%$$



Gambar 4.19 Diagram Hasil Perhitungan Kuesioner Pertanyaan Keenam

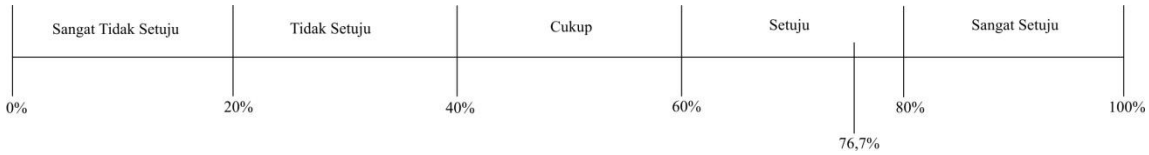
7. Apakah anda tertarik untuk memainkan *game* ini kembali?

Tabel 4.12 Perhitungan Persentase Pertanyaan Ketujuh

Pertanyaan	Jawaban	Skor	Frekuensi	S
7.	Sangat Tidak Setuju	1	0	0

	Tidak Setuju	2	0	0
	Cukup	3	12	36
	Setuju	4	11	44
	Sangat Setuju	5	7	35
Jumlah				115

$$P = \frac{115}{150} x100 = 76{,}7\%$$



Gambar 4.20 Diagram Hasil Perhitungan Kuesioner Pertanyaan Ketujuh

Berdasarkan jawaban dari hasil kuesioner yang diberikan kepada pada responden, maka dapat diambil kesimpulan yaitu:

1. Responden sangat setuju bahwa tampilan *game* sangat menarik.
2. Responden setuju bahwa mereka tertarik untuk lebih mengenal budaya Indonesia.
3. Responden sangat setuju bahwa dengan memainkan *game* ini pengetahuan mereka terhadap budaya di Indonesia terutama perang baratayuda menjadi lebih bertambah.
4. Responden setuju bahwa dalam *game* ini mengandung pesan yang edukatif.
5. Responden setuju bahwa petunjuk dalam *game* membantu untuk pengoperasian program.
6. Responden setuju bahwa mereka kesulitan untuk menyelesaikan setiap level *game*.
7. Responden setuju bahwa mereka tertarik untuk memainkan *game* ini kembali.