BAB 4
IMPLEMENTASI DAN PENGUJIAN SISTEM

4.1 Implementasi

Proses implementasi merupakan tahapan dimana aplikasi yang sudah dibuat dan dirancang, diuji kelayakannya.

4.1.1 Implementasi Perangkat Keras

Perangkat keras yang digunakan untuk melakukan implementasi dan pengujian aplikasi yaitu sebagai berikut :

1. *Smartphone* Android dengan layar *touchscreen*
2. *Processor* 1 GHz, 512 MB RAM

4.1.2 Implementasi Perangkat Lunak

Perangkat lunak yang digunakan untuk melakukan implementasi dan pengujian aplikasi *game* yaitu Sistem Operasi Android Android *2.2 (froyo)* keatas.

4.1.3 Implmentasi Antarmuka

Pada bagian ini akan ditampilkan implementasi antarmuka menu utama, menu main, menu perlengkapan, dan menu petunjuk.

4.1.4.1 Implementasi Antarmuka Menu utama

Berikut adalah Implementasi Antarmuka Menu Utama yang ditunjukkan oleh gambar 4-1.



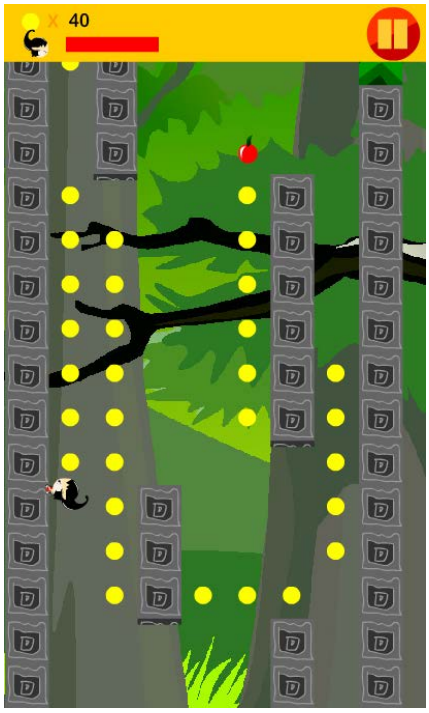
Gambar 4-1 Antarmuka Menu Utama

4.1.4.2 Implementasi Antarmuka Menu Main

Berikut adalah Implementasi Antarmuka Menu Main yang ditunjukkan oleh gambar berikutyang dimana terdapat antarmuka intro, antarmuka arena permainan dan antarmuka melawan bos.

1. Arena permainan

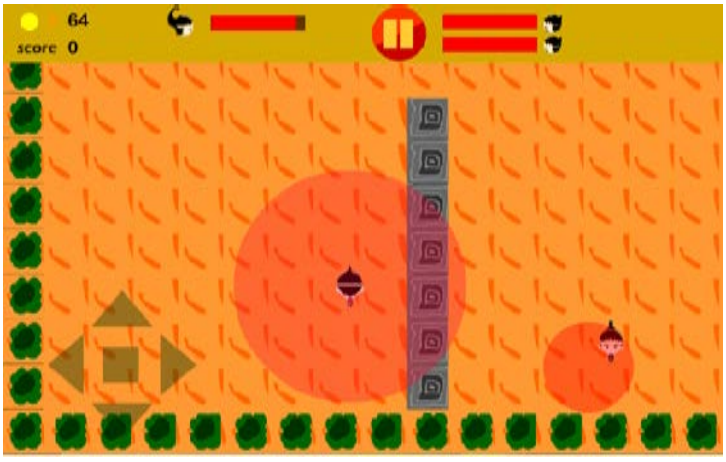
   Antarmuka arena permainan merupakan layar yang tampil pada saat intro selesai. Tampilan antarmuka arena permainan dapat dilihat pada gambar 4-2.



Gambar 4-2 Antarmuka Arena Permainan

2. Antarmuaka melawan bos

   Antarmuka melawan bos merupakan layar yag tampil pada saat arjuna telah mememukan bos besar untuk dilawan. Tampilan antarmuka melawan bos dapat dilihat pada gambar 4-3.



Gambar 4-3 Antarmuka Melawan Bos

4.1.4.3 Implementasi Antarmuka Menu Perlengkapan

Berikut adalah Implementasi Antarmuka Menu Perlengkapan yang ditunjukkan oleh gambar 4-4.



Gambar 4-4 Antarmuka Menu Perlengkapan

4.1.4.4 Impementasi Antarmuka Menu Petunjuk

Berikut adalah Implementasi Antarmuka Menu Perlengkapan yang ditunjukkan oleh gambar 4-5.



Gambar 4-5 Antarmuka Menu Petunjuk

4.1.4 Implementasi Pesan

Pada bagian implementasi pesan ini akan ditampilkan implementasi pesan pause menu, pesan perlengkapan, dan pesan exit.

4.1.4.1 Implementasi Pesan Pause Menu

Berikut adalah Implementasi Pesan *Pause Menu* yang ditunjukkan oleh gambar 4-6.



Gambar 4-6 Pesan Pause Menu

4.1.4.2 Implementasi Pesan Perlengkapan

Berikut adalah Implementasi Pesan perlengkapan yang ditunjukkan oleh gambar 4-7.



Gambar 4-7 Pesan Perlengkapan

4.1.4.3 Implementasi Pesan Keluar *Game*

Berikut adalah Implementasi Pesan perlengkapan yang ditunjukkan oleh gambar 4-8.



Gambar 4-8 Pesan Keluar *Game*

Implementasi antarmuka dan implementasi pesan lebih lengkapnya ditampilkan pada bagian Lampiran A taitu lampilran tampilan antarmuka dari *game* “Larilah Arjuna”.

4.2 Pengujian Sistem

Dalam melakukan pengujian terhadap *game* Larilah Arjuna dilakukan dengan dua macam teknik, yaitu pengujian black-box dan pengujian white-box. Pengujian black box berfokus pada menemukan kesalahan program, sedangkan pengujian white box berfokus kepada struktur internal (*source code*) program untuk mengetahui apakah masih terjadi *error*.

4.2.1 Pengujian Alpha

Pengujian dilakukan terhadap aplikasi untuk memastikan bahwa aplikasi berjalan dengan benar sesuai dengan kebutuhan dan tujuan yang diharapkan.

4.2.1.1 Skenario Pengujian

Skenario pengujian menjelaskan pengujian terhadap sistem yang ada pada *game* Larilah Arjuna. Skenario pengujian dari aplikasi permainan ini dapat dilihat pada tabel berikut.

Tabel 4-1 Skenario Pengujian

Pengujian	Butir Uji	Jenis Pengujian
Menampilkan permainan	Memilih menu main	*Blackbox*
	Menampilkan intro	*Blackbox*
	Menampilkan arena permainan	*Blackbox*
	Mengarahkan karakter	*Blackbox*
	Karakter mendapatkan item	*Blackbox*
	Karakter menabrak prajurit musuh	*Blackbox*
	Darah pemain berkurang jika terkena rintangan, prajurit dan musuh	*Blackbox*
	Karakter menembakkan panah	*Blackbox*
	Darah musuh berkurang jika terkena panah pemain	*Blackbox*
	Score bertambah jika prajurit dan musuh mati	*Blackbox*
	Musuh mengikuti karakter	*Blackbox*
	Permainan berhenti sementara	*Blackbox*
	Algoritma A*	*Whitebox*
Menampilkan menu perlengkapan	Memilih menu perlengkapan *game*	*Blackbox*
	Memilih perlengkapan	*Blackbox*
Menampilkan menu petunjuk	Memilih menu petunjuk *game*	*Blackbox*
Keluar dari *game*	Memilih menu keluar	*Blackbox*
	Menampilkan pesan keluar	*Blackbox*
	Memilih ya	*Blackbox*
	Memilih tidak	*Blackbox*

4.2.1.2 Pengujian Blackbox

Tabel 4.2 berikut adalah hasil pengujian *blackbox* yang dilakukan sesuai skenario pengujian yang telah dirancang sebelumnya :

Tabel 4-2 Hasil Pengujian *Blackbox*

Menampilkan permainan		
Masukan	Hasil yang diharapkan	Hasil pengujian
Memilih menu main	Menampilkan prolog	[√] diterima [ ] ditolak
	Menampilkan arena permainan	[√] diterima [ ] ditolak
Men-tap layar	Karakter berpindah posisi	[√] diterima [ ] ditolak
Karakter menabrak item	Karakter menapatkan item	[√] diterima [ ] ditolak
Karakter membunuh prajurit musuh	Prajurit musuh mati	[√] diterima [ ] ditolak
Karakter terkena rintangan atau prajurit musuh	Darah berkurang	[√] diterima [ ] ditolak
Menekan tombol panah	Karakter menembakkan panah	[√] diterima [ ] ditolak
Membunuh prajurit atau bos	Skor bertambah	[√] diterima [ ] ditolak

Karakter bererak	Musuh mengikuti karakter	[√] diterima [ ] ditolak
Men-tap tombol pause	Permain berhenti sementara	[√] diterima [ ] ditolak
Menampilkan menu perlengkapan		
Memilih menu perlengkapan	Menampilkan perlengkapan	[√] diterima [ ] ditolak
Memilih prlengkapan	Menampilkan pesan	[√] diterima [ ] ditolak
Menampilkan menu petujuk		
Masukan	Hasil yang diharapkan	Hasil pengujian
Memilih menu petunjuk	Menampilkan menu cara bermain *game*	[√] diterima [ ] ditolak
Eluar dari *game*		
Masukan	Hasil yang diharapkan	Hasil pengujian
Memilih menu keluar	Menampilkan pesan keluar	[√] diterima [ ] ditolak
Memilih ya	Keluar dari *game*	[√] diterima [ ] ditolak
Memilih tidak	Batal keluar dari *game*	[√] diterima [ ] ditolak

4.2.1.3 Kesimpulan Pengujian *Blackbox*

Berdasarkan hasil pengujian terhadap butir uji yang telah dilakukan memberikan kesimpulan bahwa pada proses sudah benar. Pesan kesalahan yang ada sudah tampil cukup maksimal. Secara fungsionalitas *game* sudah dapat menghasilkan *output* yang diharapkan.

4.2.1.4 Pengujian *Whitebox*

Di bawah ini merupakan kasus algoritma A* (A star) untuk menguji perangkat lunak yang sudah dibangun dengan menggunakan metode *whitebox*, berdasarkan skenario pengujian aplikasi yang sudah dijabarkan. *Source code* yang dipakai pada game larilah arjuna merupakan *source code* yang dibuat oleh Joe Hocking yang terdapat pada websitenya yaitu *www.newarteest.com*.

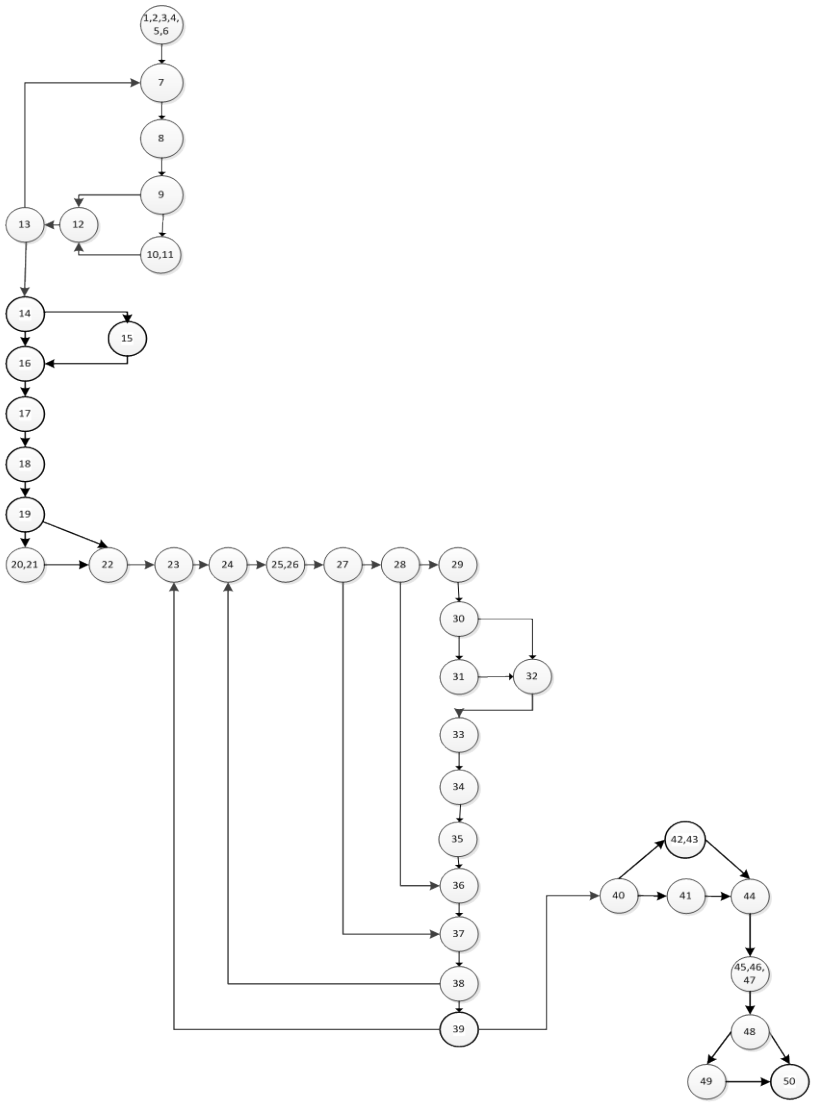
a. Algoritma A*

Tabel 4-3 Pseudocode Algoritma A*

```
    //algoritma a* pathfinding
1      Function SearchLevel()
2         curNode : PathNode
3         lowF : <integer>
4         finished : boolean
5         lowF <- 100000
6         finished <- false
    //pertama tentukan node dengan nilai F terkecil
7       for each node in openList
8                curF <- node.g + node.h
9                if lowF > curF then
10                     lowF = curF
11                     curNode = node
12               endif
13        endfor
    //jalan tidak ditemukan!
```

```
14    if curNode = null then
15        return
16    endif
      //ubah node terpilih dari open ke closed
17    hapus openList[curNode.x + " " + curNode.y]
18    closedList[curNode.x + " " + curNode.y] <- curNode
      //cek target
19    if curNode.x = finX and curNode.y = finY then
20        endNode = curNode
21        finished = true
22    endif
      //cek disetiap node yang akan dilewati
23     for  i <- -1 to i < 2 do i+1
24      for j <- -1 to j < 2 do j+1
25          col <- curNode.x + i
26          row <- curNode.y + j
        //pastikan grid bukan current node
27        if (col > 0 and col < level[0].length) and (row > 0 and
              row < level.length) and (i ≠  0 or j ≠ 0) then
            //cek g adalah open list
28             if level[row][col] = 0 and closedList[col + " " +
                row] = null and openList[col + " " + row] = null
                then
                 //menetukan g
29                  g <- 10
30                   if i ≠ 0 and j ≠ 0 then
31                       g <- 14
32                  end if
                  //menetukan h
33                 h  <-  absolut(col  -  finX))  +  absolut(row  -
                    finY)) * 10
                 //membuat node
34                  found  <-  new  PathNode(col,  row,  g,  h,
                    curNode)
35                  openList[col + " " + row] <- found
36              end if
37        end if
38      endfor
39     endfor
      //temukan jalan lain jika target tidak tercapai
40         if finished = false then
41           SearchLevel() do
42           else
43           RetracePath() do
44         end if
      // membangun berbagai poin dengan menapak mencari node
45        Function RetracePath
46        step <- new Point(node.x, node.y)
47        path.push(step)
48        if  node.g > 0  then
49              RetracePath(node.parentNode) do
50        end if
```

b. *Flow Graph*



Gambar 4-9 *Flow Graph* Algoritma A*

c. *Cyclomatic complexity* (V(G))

E (jumlah busur) = 49, N (jumlah simpul) = 39

V(G) = E – N + 2

V(G) = 49 – 39 + 2

V(G) = 12

d. Independent path

Path 1 = 1-2-3-4-5-6-7-8-9-10-11-12-13-14-15-16-17-18-19-20-21-22-23-24-25-26-27-28-29-30-31-32-33-35-36-37-38-39

Path 2 = 1-2-3-4-6-7-8-9-10-11-12-13-14-15-16-17-18-19-20-21-22-23-24-25-26-27-28-29-30-31-32-33-35-36-37-38-39

Path 3 = 1-2-3-4-5-6-7-8-10-11-12-13-14-15-16-17-18-19-20-21-22-23-24-25-26-27-28-29-30-31-32-33-35-36-37-38-39

Path 4 = 1-2-3-4-5-6-7-8-9-10-11-12-13-15-16-17-18-19-20-21-22-23-24-25-26-27-28-29-30-31-32-33-35-36-37-38-39

Path 5 = 1-2-3-4-5-6-7-8-9-10-11-12-13-14-15-16-17-18-19-29-30-31-32-33-35-36-37-38-39

Path 6 = 1-2-3-4-5-6-7-8-9-10-11-12-13-14-15-16-17-18-19-20-28-29-30-31-32-33-35-36-37-38-39

Path 7 = 1-2-3-4-5-6-7-8-9-10-11-12-13-14-15-16-17-18-19-20-21-22-24-25-26-27-28-29-30-31-32-33-35-36-37-38-39

Path 8 = 1-2-3-4-5-6-7-8-9-10-11-12-13-14-15-16-17-18-19-20-21-22-23-24-25-26-27-28-29-30-17-18-19-20-21-22-23-24-25-26-27-28-29-30-31-32-33-34-35-36-37-38-39

Path 9 = 1-2-3-4-5-6-7-8-9-10-11-12-13-14-15-16-17-18-19-20-21-22-23-24-25-26-27-28-29-30-31-32-34-35-36-37-38-39

Path 10 = 1-2-3-4-5-6-7-2-3-4-5-6-7-8-9-10-11-12-13-14-15-16-17-18-19-20-21-22-23-24-25-26-27-28-29-30-31-32-33-35-36-37-38-39

Path 11 = 1-2-3-4-5-6-7-8-9-10-11-12-13-14-15-16-17-18-19-20-21-22-23-24-25-26-27-28-29-30-17-18-19-20-21-22-23-24-25-26-27-28-29-30-31-32-33-35-36-37-38-39

Path 12 = 1-2-3-4-5-6-7-8-9-10-11-12-13-14-15-16-17-18-19-20-21-22-23-24-25-26-27-28-29-30-31-1-2-3-4-5-6-7-8-9-10-11-12-13-14-15-16-17-18-19-20-21-22-23-24-25-26-27-28-29-30-31-32-33-35-36-37-38-3932-33-35-36-37-38-39

e. *Graph Matrices*

Setelah *flow graph* diberi nomor baru, lalu dibuat tabel *graph matrice* seperti pada gambar 4-10.

N	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32	33	34	35	36	37	38	39	hasil
1		1																																							0
2			1																																						0
3				1																																				0
4					1	1																																		1
5						1																																		0
6							1																																	0
7		1						1																																1
8									1	1																														1
9										1																														0
10											1																													0
11												1																												0
12													1																											0
13														1	1																									1
14															1																									0
15																1																									0
16																	1																								0
17																		1																							0
18																			1																						0
19																				1									1												1
20																					1							1													1
21																						1																			0
22																							1	1																	1
23																								1																	0
24																									1																0
25																										1															0
26																											1														0
27																												1													0
28																													1												0
29																														1											0
30																	1														1										1
31																1																1									1
32																																	1	1							1
33																																			1						0
34																																			1						0
35																																				1					0
36																																					1				0
37																																						1	1		1
38																																							1		0
39																																									
SUM(E+1)																																									11+1=12

Gambar 4-10 *Graph Matrices*

f. Hasil uji algoritma A*

Tabel 4-4 Hasil uji Algoritma A*

No	Kasus uji	Yang diharapkan	Pengamatan	Keterangan
1	1-2-3-4-5-6-7-8-9-10-11-12-13-14-15-16-17-18-19-20-21-22-23-24-25-26-27-28-29-30-31-32-33-35-36-37-38-39	Pencarian *node* yang dicari belum ada	Pencarian dilakukan sampai mendapatkan nilai F terkecil	[√] Alur terlewati [ ] Alur tidak terlewati
2	1-2-3-4-6-7-8-9-10-11-12-13-14-15-16-17-18-19-20-21-22-23-24-25-26-27-28-29-30-31-32-33-35-36-37-38-39	Mengambil Nilai *Node* terkecil dari Node yang dicek	Pencarian dilakukan dengan mengambil nilai *node* terkecil	[√] Alur terlewati [ ] Alur tidak terlewati
3	1-2-3-4-5-6-7-8-10-11-12-13-14-15-16-17-18-19-20-21-22-23-24-25-26-27-28-29-30-31-32-33-35-36-37-38-39	Melakukan pengecekan kembali karena node terkecil tidak ditemukan	Pencarian node terkecl dilakukan kembali.	[√] Alur terlewati [ ] Alur tidak terlewati
4	1-2-3-4-5-6-7-8-9-10-11-12-13-15-16-17-18-19-20-	Node yang dicari ditemukan maka	Musuh mengkuti pemain	[√] Alur terlewati

No	Kasus uji	Yang diharapkan	Pengamatan	Keterangan
	21-22-23-24-25-26-27-28-29-30-31-32-33-35-36-37-38-39	musuh mengikuti pemain		[ ] Alur tidak terlewati
5	1-2-3-4-5-6-7-8-9-10-11-12-13-14-15-16-17-18-19-29-30-31-32-33-35-36-37-38-39	Pengecekan setiap node yang ada bukan merupakan current node	Setiap node dilakukan pengecekan.	[√] Alur terlewati [ ] Alur tidak terlewati
6	1-2-3-4-5-6-7-8-9-10-11-12-13-14-15-16-17-18-19-20-28-29-30-31-32-33-35-36-37-38-39	Proses pengecekan untuk memastikan bahwa g yang dihitung berada pada openlist dapat dilalui	Proses pengecekan nilai g pada open list diberi nilai 10	[√] Alur terlewati [ ] Alur tidak terlewati
7	1-2-3-4-5-6-7-8-9-10-11-12-13-14-15-16-17-18-19-20-21-22-24-25-26-27-28-29-30-31-32-33-35-36-37-38-39	Proses pengecekan untuk memastikan bahwa g yang dihitung berada pada openlist dapat dilalui dan menetukan nilai g	Proses pengecekan pada openlist dieri nilai 14	[√] Alur terlewati [ ] Alur tidak terlewati
8	1-2-3-4-5-6-7-8-9-10-11-12-13-14-15-16-17-18-19-20-21-22-23-24-25-26-27-28-29-30-31-32-34-35-36-37-38-39	Melakukan kembali fungsi retracepath jika finish tidak ditmukan	Fungsi retracepath dilakukan untuk menemukan node tujuan	[√] Alur terlewati [ ] Alur tidak terlewati
9	1-2-3-4-5-6-7-8-9-10-11-12-13-14-15-16-17-18-19-20-21-22-23-24-25-26-27-28-29-30-31-32-33-35-36-37-39	Jika nilai node yang akan dilewati lebih dari 1 maka melakukan fungsi retracepath	Melakukan fungsi retracepath atau mencari node kembali.	[√] Alur terlewati [ ] Alur tidak terlewati
10	1-2-3-4-5-6-7-2-3-4-5-6-7-8-9-10-11-12-13-14-15-16-17-18-19-20-21-22-23-24-25-26-27-28-29-30-31-32-33-35-36-37-38-39	Melakukan proses looping untuk mencek nilai f terkecil	Proes looping dilakukan sehingga mendapatkan nilai f terkecil.	[√] Alur terlewati [ ] Alur tidak terlewati
11	1-2-3-4-5-6-7-8-9-10-11-12-13-14-15-16-17-18-19-20-21-22-23-24-25-26-27-28-29-30-17-18-19-20-21-22-23-24-25-26-27-28-29-30-31-32-33-35-36-37-38-39	Melakukan looping untuk mencek titik i pada setiap node yang akan dilalui	Dilakukan proses looping pegecekan titik i pada setiap node yang akan dilewati	[√] Alur terlewati [ ] Alur tidak terlewati
12	1-2-3-4-5-6-7-8-9-10-11-12-13-14-15-16-17-18-19-20-21-22-23-24-25-26-27-28-29-30-31-1-2-3-4-5-6-7-8-9-10-11-12-13-14-15-16-17-18-19-20-21-22-23-24-25-26-27-28-29-30-31-32-33-35-36-37-38-3932-33-35-36-37-38-39	Melakukan looping untuk mencek titik j pada setiap node yang akan dilalui	Dilakukan proses looping pegecekan titik j pada setiap node yang akan dilewati	[√] Alur terlewati [ ] Alur tidak terlewati

4.2.1.5 Kesimpulan Pengujian Whitebox

Dari hasil pengujian algoritma A* yang dilakukan, maka didapat kesimpulan bahwa hasil cyclomatic complexity dari algoritma A* adalah 12 dimana hasil tersebut menunjukan tingkat kekompleksan prosedur yang cukup rumit dan hasil pengujian pada *independent path* yang dilakukan menunjukan bahwa seluruh *node* yang ada pada setiap *independent path* telah dikerjakan dengan tidak terjadi pengulangan tak terhingga .

4.2.2 Pengujian Beta

Pengujian beta merupakan pengujian yang dilakukan secara objektif, dimana dilakukan pengujian secara langsung terhadap pengguna dengan menggunakan kuesioner mengenai kepuasan pengguna atas aplikasi yang telah dibangun. Adapun metode penilaian pengujian yang digunakan adalah metode kuantitatif berdasarkan data dari pengguna.

4.2.2.1 Kuisioner

Kuesioner disebarkan kepada anak-anak 12 tahun keatas dan disebarkan kepada 30 orang kemudian dari hasil kuesioner tersebut akan dilakukan perhitungan agar dapat diambil kesimpulan terhadap penilaian penerapan sistem yang telah dibangun. Kuesioner terdiri dari 7 pertanyaan dengan menggunakan skala 1 sampai 5 (skala likert). Berikut adalah hasil perhitungan masing-masing jawaban yang telah dihitung.

1. Menurut anda, apakah tampilan dari *game* ini menarik?

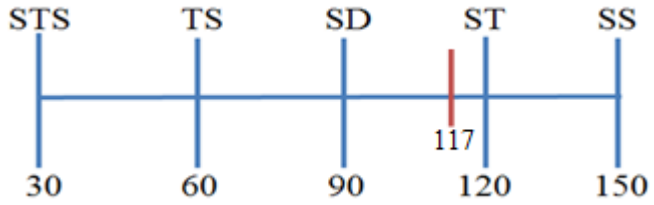
Tabel 4-5 Hasil Pengolahan Data Kuisioner Pertanyaan ke-1

JAWABAN	SKOR	FREQUENSI	JUMLAH SKOR
Sangat Setuju	5	5	25
Setuju	4	17	68
Ragu ragu	3	8	24
Tidak setuju	2	0	0
Sangat tidak setuju	1	0	0
		TOTAL	117

$$P = \frac{117}{150} \times 100\% = 78\%$$

Dapat disimpulkan bahwa penilaian terhadap menariknya tampilan *game* Larilah Arjuna adalah 78% dari yang diharapkan 100%. *Score* maksimal yang

didapat untuk penilaian kuesioner pada pertanyaan ke-1 dapat dilihat pada *rating scale* gambar 4-11 berikut ini.



Gambar 4-11 Interpretasi Skor Jawaban ke-1

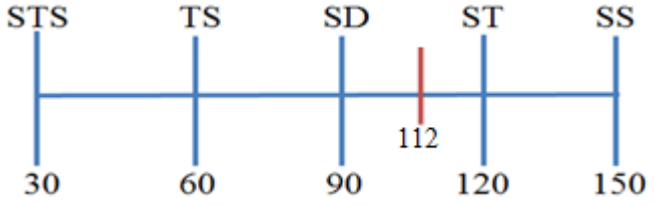
2. Apakah informasi dari petunjuk *game* ini membantu anda dalam memainkan *game*?

Tabel 4-6 Hasil Pengolahan Data Kuisioner Pertanyaan ke-2

JAWABAN	SKOR	FREQUENSI	JUMLAH SKOR
Sangat Setuju	5	6	30
Setuju	4	12	48
Ragu ragu	3	10	30
Tidak setuju	2	2	4
Sangat tidak setuju	1	0	0
		TOTAL	112

$$P = \frac{112}{150} \times 100\% = 74{,}6\%$$

Dapat disimpulkan bahwa penilaian terhadap informasi dari petunjuk *game* apakah membantu dalam memainkan *game* adalah 74,6% dari yang diharapkan 100%. *Score* maksimal yang didapatuntuk penilaian kuesioner pada pertanyaan ke-2 dapat dilihat pada *rating scale* gambar 4-12 berikut ini.



Gambar 4-12 Interpretasi Skor Jawaban ke-2

3. Apakah setelah memainkan *game* ini anda menjadi tahu tentang cerita perang Baratayuda?

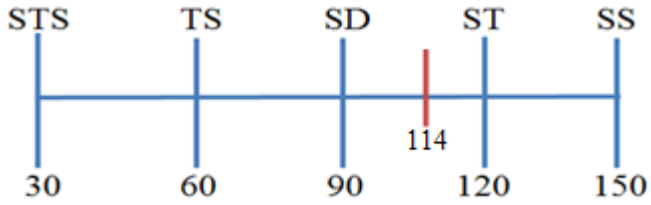
Tabel 4-7 Hasil Pengolahan Data Kuisioner Pertanyaan ke-3

JAWABAN	SKOR	FREQUENSI	JUMLAH SKOR
Sangat Setuju	5	6	30
Setuju	4	13	52

Ragu ragu	3	10	30
Tidak setuju	2	1	2
Sangat tidak setuju	1	0	0
		TOTAL	114

$$P = \frac{114}{150} \times 100\% = 76\%$$

Dapat disimpulkan bahwa penilaian terhadap apakah setelah memainkan *game* menjadi tahu tentang cerita perang Baratayuda adalah 76% dari yang diharapkan 100%. *Score* maksimal yang didapatuntuk penilaian kuesioner pada pertanyaan ke-3 dapat dilihat pada *rating scale* gambar 4-13 berikut ini.



Gambar 4-13 Interpretasi Skor Jawaban ke-3

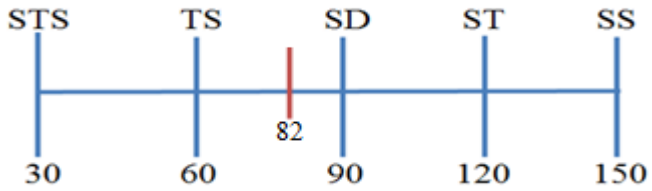
4. Apakah setelah memainkan *game* ini anda mempunyai keinginan untuk mencari tahu lebih dalam tentang kisah wayang dan Arjuna?

Tabel 4-8 Hasil Pengolahan Data Kuisioner Pertanyaan ke-4

JAWABAN	SKOR	FREQUENSI	JUMLAH SKOR
Sangat Setuju	5	2	10
Setuju	4	6	24
Ragu ragu	3	5	15
Tidak setuju	2	16	32
Sangat tidak setuju	1	1	1
		TOTAL	82

$$P = \frac{82}{150} \times 100\% = 54{,}66\%$$

Dapat disimpulkan bahwa penilaian terhadap apakah setelah memainkan *game* anda mempunyai keinginan untuk mencari tahu lebih dalam tentng kisah wayang dan Arjuna adalah 54,66% dari yang diharapkan 100%. *Score* maksimal yang didapatuntuk penilaian kuesioner pada pertanyaan ke-4 dapat dilihat pada *rating scale* gambar 4-14 berikut ini.



Gambar 4-14 Interpretasi Skor Jawaban ke-4

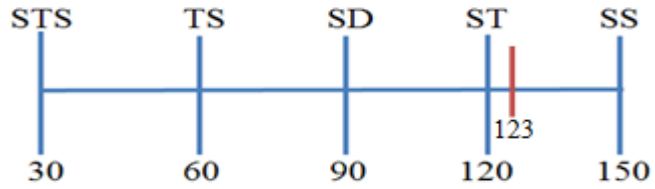
5. Apakah cara bermain *game* ini membuat anda ingin menyelesaikan *game* ini?

Tabel 4-9 Hasil Pengolahan Data Kuisioner Pertanyaan ke-5

JAWABAN	SKOR	FREQUENSI	JUMLAH SKOR
Sangat Setuju	5	7	45
Setuju	4	11	44
Ragu ragu	3	10	30
Tidak setuju	2	2	4
Sangat tidak setuju	1	0	0
		TOTAL	123

$$P = \frac{123}{150} \times 100\% = 82\%$$

Dapat disimpulkan bahwa penilaian terhadap apakah setelah memainkan *game* anda mempunyai keinginan untuk mencari tahu lebih dalam tentng kisah wayang dan Arjuna adalah 82% dari yang diharapkan 100%. *Score* maksimal yang didapatuntuk penilaian kuesioner pada pertanyaan ke-5 dapat dilihat pada *rating scale* gambar 4-15 berikut ini



Gambar 4-15 Interpretasi Skor Jawaban ke-5

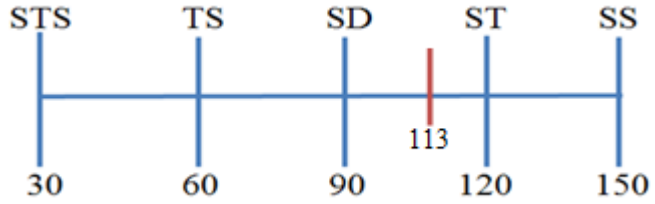
6. Dengan kemampuan musuh yang dapat mencari pemain, apakah *game* ini menantang untuk dimainkan?

Tabel 4-10 Hasil Pengolahan Data Kuisioner Pertanyaan ke-6

JAWABAN	SKOR	FREQUENSI	JUMLAH SKOR
Sangat Setuju	5	10	50
Setuju	4	8	32
Ragu ragu	3	7	21
Tidak setuju	2	5	10
Sangat tidak setuju	1	0	0
		TOTAL	113

$$P = \frac{113}{150} \times 100\% = 75{,}33\%$$

Dapat disimpulkan bahwa penilaian terhadap apakah dengan kemampuan musuh yang dapat mencari pemain *game* ini menantang untuk dimainkan adalah 75,33% dari yang diharapkan 100%. *Score* maksimal yang didapatuntuk penilaian kuesioner pada pertanyaan ke-6 dapat dilihat pada *rating scale* gambar 4-16 berikut ini.



Gambar 4-16 Interpretasi Skor Jawaban ke-6

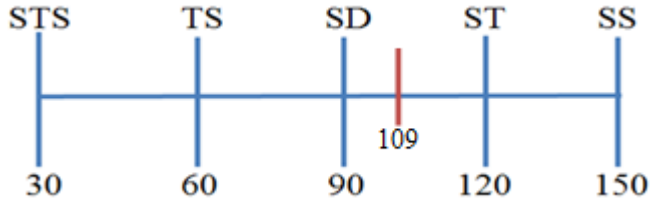
7. Apakah anda tertarik untuk memainkan *game* ini kembali?

Tabel 4-11 Hasil Pengolahan Data Kuisioner Pertanyaan ke-7

JAWABAN	SKOR	FREQUENSI	JUMLAH SKOR
Sangat Setuju	5	5	25
Setuju	4	15	60
Ragu ragu	3	5	15
Tidak setuju	2	4	8
Sangat tidak setuju	1	1	1
		TOTAL	109

$$P = \frac{109}{150} \times 100\% = 72{,}66\%$$

Dapat disimpulkan bahwa penilaian terhadap apakah tertarik untuk memainkan kembali *game* ini adalah 72,66% dari yang diharapkan 100%. *Score* maksimal yang didapatuntuk penilaian kuesioner pada pertanyaan ke-7 dapat dilihat pada *rating scale* gambar 4-17 berikut ini.



Gambar 4-17 Interpretasi Skor Jawaban ke-7

4.2.2.2 Kesimpulan Pengujian Beta

Berdasarkan hasil persentase jawaban setiap user atau responden terhadap pertanyaan yang diajukan pada pengujian beta dapat ditarik kesimpulan bahwa *game* yang dibangun memiliki tampilan yang menarik, informasi dari petunjuk dapat membantu dalam memainkan *game*, setelah memainkan *game* pemain tahu tentang cerita perang Baratayuda tetapi setelah memainkan *game* pemain kurang tertarik untuk mencari tahu tentang cerita perang Baratayuda dan Arjuna, pemain mempunyai keinginan untuk menyelsaikan *game*, dengan kemapuan musuh yang dapat mencari pemain *game* ini menantang untuk dimainkan dan pemain tertarik untuk memainkan *game* ini kembali.