BAB 4

IMPLEMENTASI DAN PENGUJIAN

Implementasi dan pengujian dilakukan pada *game* yang dibangun, yaitu *game* Kapten Indonesia. Tahap ini dilakukan setelah perancangan selesai dibuat yang kemudian akan diimplementasikan ke dalam bahasa pemrograman yang digunakan. Setelah tahap implementasi, selanjutnya dilakukan tahap pengujian terhadap *game* yang dibangun.

4.1 Implementasi

Tahap implementasi adalah tahap dimana *game* yang dirancang pada tahap sebelumnya untuk kemudian direalisasikan menjadi sebuah aplikasi *game* yang siap untuk digunakan. Tahapan ini mencakup implementasi perangkat keras, perangkat lunak, *game*, serta implementasi antarmuka.

4.1.1 Implementasi Perangkat Keras

Penggunaan perangkat keras untuk menjalankan *game* Kapten Indonesia tidak diharuskan perangkat yang berspesifikasi tinggi karena game tersebut dapat dijalankan dengan *web browser*. Adapun spesifikasi minimum yang dapat digunakan untuk menjalankan *game* ini adalah sebagai berikut :

1. Prosesor dengan kecepatan 1,8 GHz.
2. Memori RAM 512 MB.
3. Harddisk internal 128 GB.
4. VGA Card 128 MB.
5. Monitor.
6. Keyboard.
7. Mouse.
8. Modem/ Broadband/ Wi-fi dengan kecepatan 500 kbps.

4.1.2 Implementasi Perangkat Lunak

Spesifikasi perangkat lunak yang dibutuhkan pengguna untuk dapat menjalankan game ini adalah diantaranya sistem operasi *Microsoft Windows 7 / Windows Vista / Windows XP* serta *web browser* seperti *Google Chrome* atau lainnya, minimal versi yang sudah mendukung HTML 5.

4.1.3 Implementasi *Game*

Game ini menggunakan *Quintus* sebagai *game engine* serta aplikasi *Brackets* dengan bahasa pemrograman *JavaScript.*

4.1.4 Implementai *Class*

Implementasi *class* merupakan implementasi dari perancangan *class* diagram yang terbentuk dari *use case diagram* ke dalam bentuk *file* fisik dengan ekstensi ".as".

Tabel 4.1 Implementasi *Class*

No.	Nama *class*	Nama *class* fisik
1.	Quintus	Quintus.as
2.	loadTMX	loadTMX.as
3.	Asearch	Asearch.as
4.	HighScore	HighScore.as
5.	Bantuan	Bantuan.as
6.	Player	Player.as
7.	Stage1	Stage1.as
8.	Stage2	Stage2.as
9.	Stage3	Stage3.as
10.	Stage4	Stage4.as
11.	Collectable	Collectable.as
12.	Throwerbox	Throwerbox.as
13.	Box	Box.as
14.	Door	Door.as
15.	Enemy	Enemy.as
16.	Ladder	Ladder.as
17.	Tower	Tower.as

4.1.5 Implementasi Antarmuka

Implementasi antarmuka bertujuan untuk memberikan gambaran nyata terhadap *game* yang dibangun secara rinci. Berikut ini akan dipaparkan tampilan antarmuka beserta penjelasannya.

4.2.5.4 Antarmuka Menu Utama

Tampilan dari gambar 4.1 dibawah merupakan tampilan awal ketika pemain menjalankan permainan. Pada tampilan tersebut terdapat menu pilihan diantaranya mulai permainan, pilih stage, bantuan, serta nilai tertinggi.

Gambar 4.1 Tampilan Antarmuka Menu Utama

4.2.5.5 Antarmuka Pilih Stage

Tampilan dari gambar 4.2 berikut ini merupakan antarmuka ketika pemain memilih menu pilih *stage*. Adapun menu pilihan yang tersedia diantaranya *stage* 1, *stage* 2, *stage* 3 dan *stage* 4.

Gambar 4.2 Tampilan Antarmuka Pilih Stage

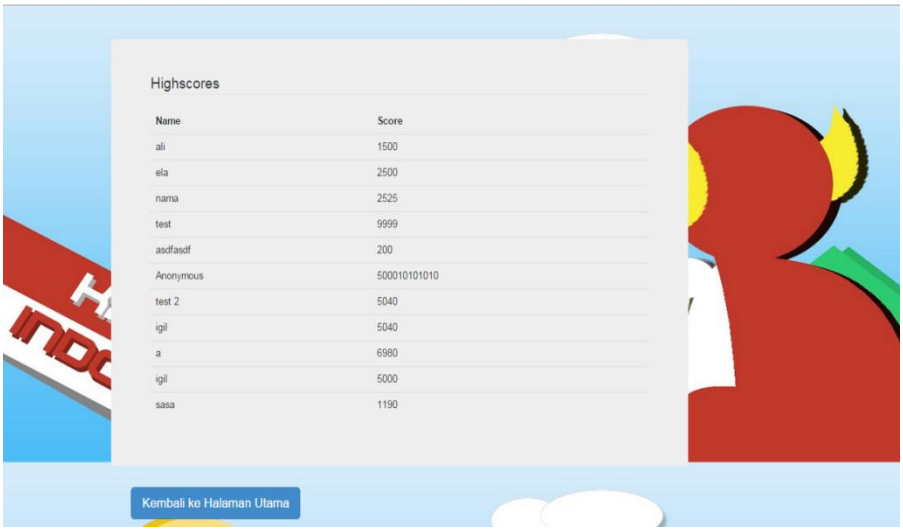
4.2.5.6 Antarmuka Bantuan

Tampilan dari gambar 4.3 berikut ini merupakan antarmuka ketika pemain memilih menu bantuan. Menu bantuan berisi informasi berupa cara bermain dan item yang terdapat di dalam *game*.

Gambar 4.3 Tampilan Antarmuka Bantuan

4.2.5.7 Antarmuka Nilai Tertinggi

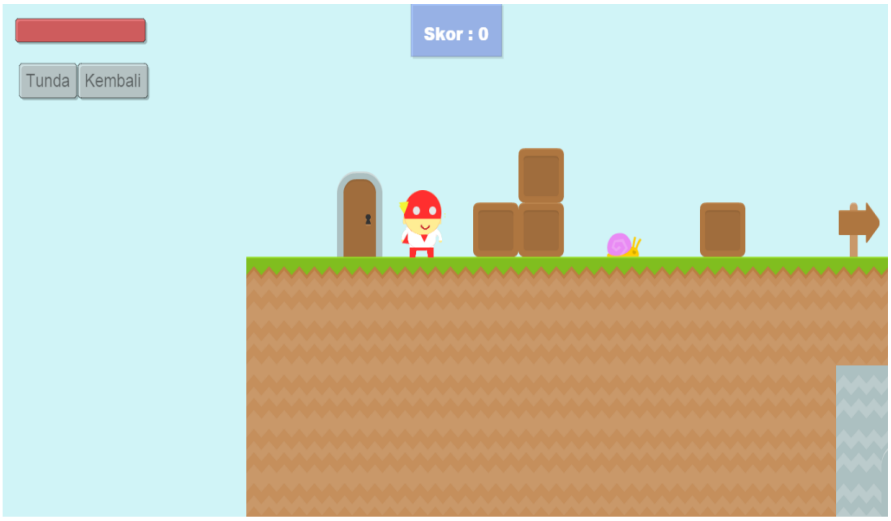
Tampilan dari gambar 4.4 berikut ini merupakan antarmuka ketika pemain memilih menu nilai tertinggi. Pada menu ini pemain dapat melihat nama dan jumlah skor yang diperoleh.



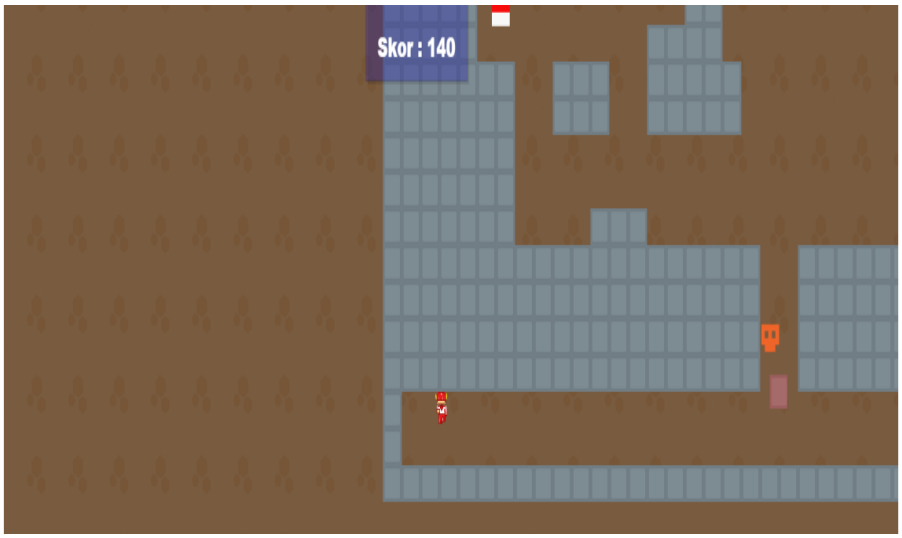
Gambar 4.4 Tampilan Antarmuka Nilai Tertinggi

4.2.5.8 Antarmuka Permainan

Pada gambar 4.5 sampai 4.8 berikut ini adalah tampilan permainan pada *stage* 1 hingga *stage* 4. Pada antarmuka ini, pemain mula – mula melihat pesan cerita dan kemudian memilih mulai permainan.



Gambar 4.5 Tampilan Antarmuka *Stage* 1



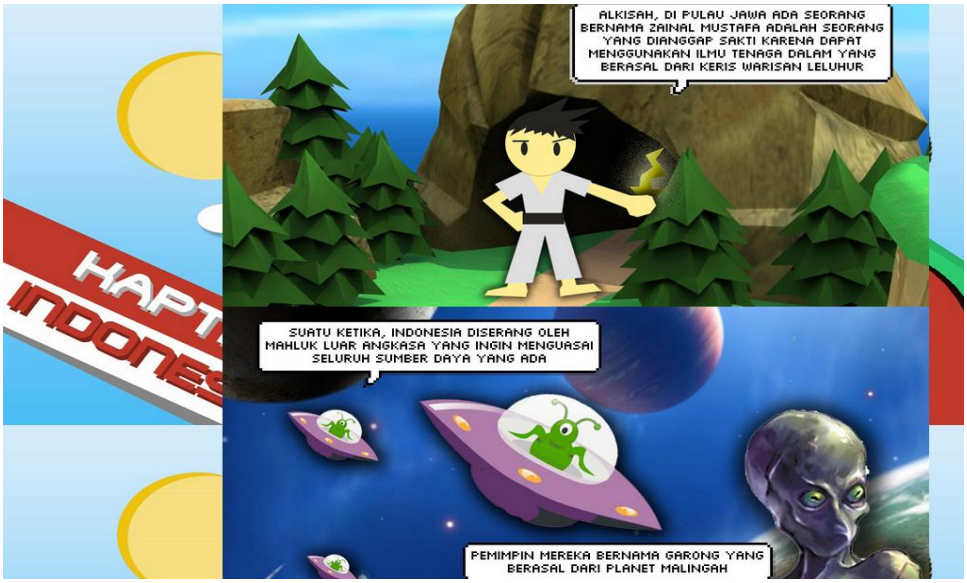
Gambar 4.6 Tampilan Antarmuka *Stage* 2

Gambar 4.7 Tampilan Antarmuka *Stage* 3

Gambar 4.8 Tampilan Antarmuka *Stage* 4

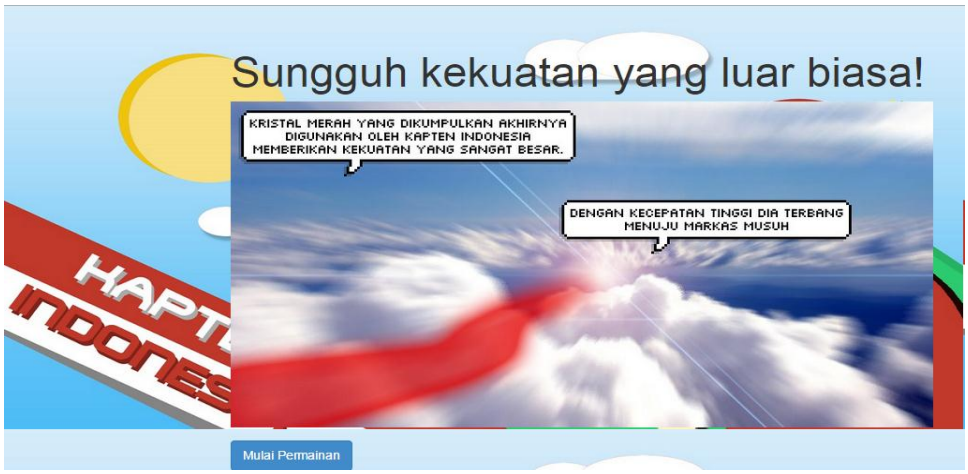
4.2.5.9 Antarmuka Pesan Cerita

Pada gambar 4.9 sampai 4.13 berikut ini adalah tampilan pesan cerita pada *stage* 1 hingga akhir permainan. Tampilan pesan ini muncul di antara *stage* atau setiap perpindahan *stage*.



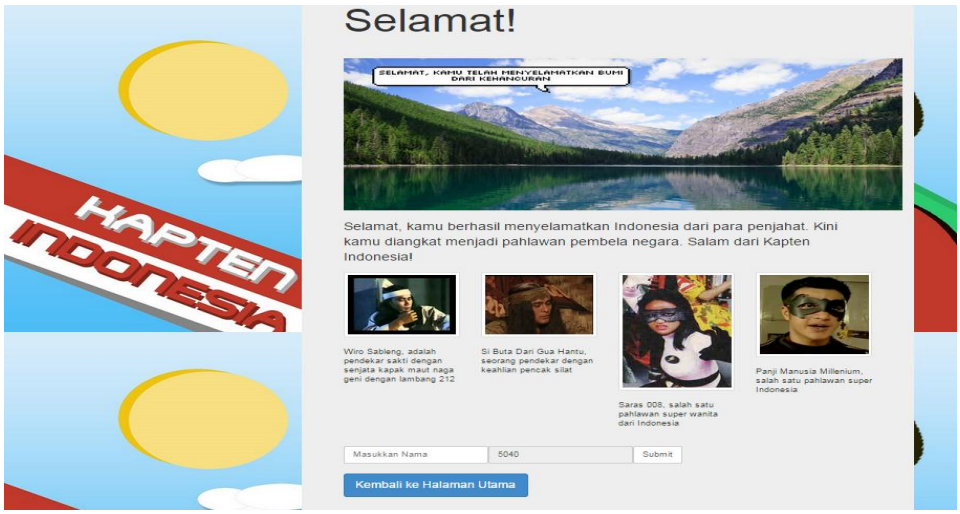
Gambar 4.9 Tampilan Antarmuka Pesan Cerita *Stage* 1

Gambar 4.10 Tampilan Antarmuka Pesan Cerita *Stage* 2



Gambar 4.11 Tampilan Antarmuka Pesan Cerita *Stage* 3

Gambar 4.12 Tampilan Antarmuka Pesan Cerita *Stage* 4



Gambar 4.13 Tampilan Antarmuka Pesan Cerita Akhir *Stage*

4.2 Pengujian

Pengujian terhadap Game Kapten Indonesia dilakukan dengan dua macam teknik, yaitu pengujian *black-box* dan pengujian *white-box*. Adapun pengujian *black box* difokuskan untuk menemukan kesalahan program, sedangkan pengujian *white box* difokuskan untuk struktur internal (*source code*) program untuk mengetahui apakah masih terjadi error. Hasil pengujian dilakukan dengan dua tahap, yaitu tahap pengujian alpha dan tahap pengujian beta.

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.

4.2.2 Pengujian *Black Box*

Pengujian dilakukan untuk memastikan *Game* Kapten Indonesia telah berjalan dengan benar dan sesuai dengan kebutuhan yang diinginkan. Pengujian *blackbox* menitik beratkan kepada persyaratan fungsional perangkat lunak. Adapun rencana pengujiannya dapat diamati melalui tabel 4.2 berikut ini.

Tabel 4.2 Tabel Rencana Pengujian Aplikasi

No	Kelas Uji	Detail Pengujian	Jenis Uji
1.	Memainkan permainan	Memulai permainan dari menu utama	*Black Box*
2.	Memilih stage	Menampilkan menu pilihan stage	*Black Box*
3.	Menampilkan bantuan	Menampilkan halaman bantuan	*Black Box*
4.	Menampilkan nilai tertinggi	Menampilkan halaman nilai tertinggi	*Black Box*
5.	Memainkan stage 1	Memulai permainan pada stage 1	*Black Box*
6.	Memainkan stage 2	Memulai permainan pada stage 2	*Black Box*
7.	Memainkan stage 3	Memulai permainan pada stage 3	*Black Box*
8.	Memainkan stage 4	Memulai permainan pada stage 4	*Black Box*
9.	Menggerakkan karakter	Pergerakan karakter melalui interaksi tombol keyboard	*Black Box*

10.	Mengambil koin	Proses penambahan skor dari koin	*Black Box*
11.	Mengambil kristal	Proses penambahan skor dari kristal	*Black Box*
12.	Memasuki pintu	Uji masuk dan keluar pintu	*Black Box*
13.	Menaiki tangga	Uji naik dan turun tangga	*Black Box*
14.	Menyentuh musuh	Uji respon sistem jika musuh mengenai pemain	*Black Box*
15.	Menyentuh kotak kayu	Uji respon sistem jika kotak kayu mengenai pemain	*Black Box*
16.	Menyentuh tuas	Uji respon sistem jika pemain menyentuh tuas	*Black Box*
17.	Mengambil bendera	Uji respon sistem jika pemain mengambil bendera	*Black Box*
18.	Memasukkan nama	Menginputkan nama untuk nilai tertinggi	*Black Box*
19.	Melihat pesan cerita	Menampilkan pesan cerita	*Black Box*
20.	Menunda permainan	Uji respon terhadap pilihan tunda	*Black Box*

4.2.5.4 Kasus Dan Hasil Pengujian (*Black Box*)

Berikut ini merupakan kasus untuk menguji *game* yang sudah dibangun dengan menggunakan metode *black box* berdasarkan rencana pengujian yang terdapat pada Tabel 4.2 yang telah dijelaskan. Hasil pengujian yang akan dilakukan pada *game* ini selengkapnya dapat dilihat pada Tabel 4.3 hingga tabel 4.16 berikut ini.

1. Memainkan Permainan

Tabel 4.3 Hasil Pengujian Memainkan Permainan

Test ID	Keterangan	Hasil yang diahrapkan	Hasil pengujian
1	Kondisi awal : Pemain berada di menu utama. Kemudian pemain memilih menu mulai permainan	Sistem memindahkan pemain ke pesan cerita stage 1	[√] Berhasil [ ] Tidak berhasil

2. Memilih Stage

Tabel 4.4 Hasil Pengujian Memilih Stage

Test ID	Keterangan	Hasil yang diahrapkan	Hasil pengujian
2	Kondisi awal : Pemain berada di menu utama. Kemudian pemain memilih menu pilih stage	Sistem menampilkan halaman menu pilihan stage 1 sampai 4	[√] Berhasil [ ] Tidak berhasil
3	Pemain memilih kembali ke menu utama	Sistem menampilkan menu utama	[√] Berhasil [ ] Tidak berhasil

3. Menampilkan Bantuan

Tabel 4.5 Hasil Pengujian Menampilkan Bantuan

Test ID	Keterangan	Hasil yang diahrapkan	Hasil pengujian
4	Kondisi awal : Pemain berada di menu utama. Kemudian pemain memilih menu bantuan	Sistem menampilkan halaman bantuan cara bermain	[√] Berhasil [ ] Tidak berhasil
5	Pemain memilih kembali ke menu utama	Sistem menampilkan menu utama	[√] Berhasil [ ] Tidak berhasil

4. Menampilkan Nilai Tertinggi

Tabel 4.6 Hasil Pengujian Menampilkan Nilai Tertinggi

Test ID	Keterangan	Hasil yang diahrapkan	Hasil pengujian
6	Kondisi awal : Pemain berada di menu utama. Kemudian pemain memilih menu nlai tertinggi	Sistem menampilkan halaman nilai tertinggi	[√] Berhasil [ ] Tidak berhasil
7	Pemain memilih kembali ke menu utama	Sistem menampilkan menu utama	[√] Berhasil [ ] Tidak berhasil

5. Memainkan Stage 1

Tabel 4.7 Hasil Pengujian Memainkan Stage 1

Test ID	Keterangan	Hasil yang diahrapkan	Hasil pengujian
8	Kondisi awal : Pemain berada di menu pilihan stage. Kemudian pemain memilih stage 1	Sistem menampilkan permainan stage 1	[√] Berhasil [ ] Tidak berhasil

6. Memainkan Stage 2

Tabel 4.8 Hasil Pengujian Memainkan Stage 2

Test ID	Keterangan	Hasil yang diahrapkan	Hasil pengujian
9	Kondisi awal : Pemain berada di menu pilihan stage. Kemudian pemain memilih stage 2	Sistem menampilkan permainan stage 2	[√] Berhasil [ ] Tidak berhasil

7. Memainkan Stage 3

Tabel 4.9 Hasil Pengujian Memainkan Stage3

Test ID	Keterangan	Hasil yang diahrapkan	Hasil pengujian
9	Kondisi awal : Pemain berada di menu pilihan stage. Kemudian pemain memilih stage 3	Sistem menampilkan permainan stage 3	[√] Berhasil [ ] Tidak berhasil

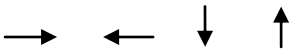
8. Memainkan Stage 4

Tabel 4.10 Hasil Pengujian Memainkan Stage 4

Test ID	Keterangan	Hasil yang diahrapkan	Hasil pengujian
10	Kondisi awal : Pemain berada di menu pilihan stage. Kemudian pemain memilih stage 4	Sistem menampilkan permainan stage 4	[√] Berhasil [ ] Tidak berhasil

9. Menggerakkan Karakter

Tabel 4.11 Hasil Pengujian Menggerakkan Karakter

Test ID	Keterangan	Hasil yang diahrapkan	Hasil pengujian
11	Kondisi awal : Pemain berada didalam permainan. Kemudian pemain melakukan interaksi dengan menggunakan tombol arah pada keyboard Tombol arah : 	Sistem merespon dengan menggerakkan karakter. Pergerakan : Melompat, merunduk, bergerak keatas, bergerak kebawah, bergerak kedepan, bergerak kebelakang	[√] Berhasil [ ] Tidak berhasil

10. Mengambil Koin

Tabel 4.12 Hasil Pengujian Mengambil Koin

Test ID	Keterangan	Hasil yang diahrapkan	Hasil pengujian
12	Kondisi awal : Pemain berada didalam permainan. Kemudian pemain mengambil koin	Sistem menghilangkan koin dan menambah skor	[√] Berhasil [ ] Tidak berhasil

11. Mengambil Kristal

Tabel 4.13 Hasil Pengujian Mengambil Kristal

Test ID	Keterangan	Hasil yang diahrapkan	Hasil pengujian
13	Kondisi awal : Pemain berada didalam permainan. Kemudian pemain mengambil kristal	Sistem menghilangkan kristal dan menambah skor	[√] Berhasil [ ] Tidak berhasil

12. Memasuki Pintu

Tabel 4.14 Hasil Pengujian Memasuki Pintu

Test ID	Keterangan	Hasil yang diahrapkan	Hasil pengujian
14	Kondisi awal : Pemain berada didalam permainan. Kemudian pemain menekan tombol arah pada keyboard	Sistem merespon dengan memindahkan karakter ke dalam atau ke luar ruangan	[√] Berhasil [ ] Tidak berhasil

13. Menaiki Tangga

Tabel 4.15 Hasil Pengujian Menaiki Tangga

Test ID	Keterangan	Hasil yang diahrapkan	Hasil pengujian
15	Kondisi awal : Pemain berada didalam permainan. Kemudian pemain menekan tombol arah pada keyboard	Sistem merespon dengan menggerakkan karakter naik atau turun tangga	[√] Berhasil [ ] Tidak berhasil

14. Menyentuh Musuh

Tabel 4.16 Hasil Pengujian Menyentuh Musuh

Test ID	Keterangan	Hasil yang diahrapkan	Hasil pengujian
16	Kondisi awal : Pemain berada di dalam permainan stage 1 atau 4. Kondisi : Pemain mengenai musuh	Sistem merespon dengan mengurangi darah karakter	[√] Berhasil [ ] Tidak berhasil
17	Kondisi awal : Pemain berada di dalam permainan stage 2 Kondisi : Pemain mengenai musuh	Sistem menampilkan pesan gagal	[√] Berhasil [ ] Tidak berhasil

15. Menyentuh Kotak Kayu

Tabel 4.17 Hasil Pengujian Menyentuh Kotak Kayu

Test ID	Keterangan	Hasil yang diahrapkan	Hasil pengujian
18	Kondisi awal : Pemain berada di dalam permainan stage 3 Kondisi : Pemain mengenai kotak kayu	Sistem menampilkan pesan gagal	[√] Berhasil [ ] Tidak berhasil

16. Menyentuh Tuas

Tabel 4.18 Hasil Pengujian Menyentuh Tuas

Test ID	Keterangan	Hasil yang diahrapkan	Hasil pengujian
19	Kondisi awal : Pemain berada di dalam permainan stage 2 Kondisi : Menyentuh tuas	Sistem menghilangkan penghalang	[√] Berhasil [ ] Tidak berhasil

17. Mengambil Bendera

Tabel 4.19 Hasil Pengujian Mengambil Bendera

Test ID	Keterangan	Hasil yang diahrapkan	Hasil pengujian
20	Kondisi awal : Pemain berada di dalam permainan. Kemudian pemain menyentuh bendera	Sistem menampilkan pesan berhasil	[√] Berhasil [ ] Tidak berhasil

18. Memasukkan Nama

Tabel 4.20 Hasil Pengujian Memasukkan Nama

Test ID	Keterangan	Hasil yang diahrapkan	Hasil pengujian
21	Kondisi awal : Pemain berada di menu pilihan stage. Kemudian	Sistem menampilkan permainan stage 4	[√] Berhasil [ ] Tidak berhasil

	pemain memilih stage 4		

19. Melihat Pesan Cerita

Tabel 4.21 Hasil Pengujian Melihat Pesan Cerita

Test ID	Keterangan	Hasil yang diahrapkan	Hasil pengujian
22	Kondisi awal : Pemain berada di pesan cerita akhir stage. Kemudian pemain menginput nama dan memilih tombol simpan	Sistem menyimpan skor	[√] Berhasil [ ] Tidak berhasil
23	Kondisi : Pemain memilih tombol kembali ke menu utama	Sistem menampilkan menu utama	[√] Berhasil [ ] Tidak berhasil

20. Menunda Permainan

Tabel 4.22 Hasil Pengujian Menunda Permainan

Test ID	Keterangan	Hasil yang diahrapkan	Hasil pengujian
24	Kondisi awal : Pemain berada di dalam permainan. Kemudian pemain memilih tombol tunda	Sistem merespon dengan menunda permainan	[√] Berhasil [ ] Tidak berhasil
25	Kondisi : Pemain memilih tombol kembali	Sistem menampilkan menu utama	[√] Berhasil [ ] Tidak berhasil

4.2.3 Pengujian *White Box*

Berikut ini adalah kasus menguji *game* yang telah dibangun menggunakan metode *white-box*. Pengujian *white-box* menggunakan *flow graph* yang digunakan untuk menggambarkan alur dari algoritma A *star* yang diterapkan pada karakter musuh dalam *Game* Kapten Indonesia dan *graph matrix* yang digunakan untuk menggenerasi *flow graph*. Adapun pengujian *white-box* adalah sebagai berikut.

1. Pengujian Algoritma A *star*

Pengujian algoritma A *star* adalah proses pengujian terhadap jalannya algoritma yang diterapkan pada karakter musuh dengan mencari jalur tercepat dalam pergerakan menuju karakter utama atau karakter pemain sebagai titik tujuan. Alurnya dapat diamati pada tabel 4.23 berikut.

Tabel 4.23 *List* Program *A Star*

No	Class ASearch
0	Q.Class.extend("ASearch",{
	init: function() {},
1	euclideanDistance: function(titikAwal,titikTujuan) {
	var distance = Math.sqrt(Math.pow(titikAwal.x -
	titikAwal.y,2)+Math.pow(titikTujuan.x- titikTujuan.y,2));
	return Math.floor(distance)*10;
	},
2	normalDistance: function(titikAwal,titikTujuan){
	var direction ='';
	var blockPoint = 0;
	var outOfBound = false;
3	if(mapArray[titikTujuan.y][titikTujuan.x] == 1){
4	blockPoint = 1000;
	}
5	var horVer = 10+blockPoint;
6	var diagonal = 14+blockPoint;
	var point;
7	if(titikAwal.x == titikTujuan.x && titikAwal.y >
	titikTujuan.y){
8	direction ='up';
	point = horVer;
9	}else if(titikAwal.x > titikTujuan.x && titikAwal.y >

	titikTujuan.y){
10	direction ='upleft';
	point = diagonal;
11	}else if(titikAwal.x > titikTujuan.x && titikAwal.y ==
	titikTujuan.y){
12	direction ='left';
	point = horVer;
13	}else if(titikAwal.x > titikTujuan.x && titikAwal.y <
	titikTujuan.y){
14	direction ='downleft';
	point = diagonal;
15	}else if(titikAwal.x == titikTujuan.x && titikAwal.y <
	titikTujuan.y){
16	direction ='down';
	point = horVer;
17	}else if(titikAwal.x < titikTujuan.x && titikAwal.y <
	titikTujuan.y){
18	direction ='downright';
	point = diagonal;
19	}else if(titikAwal.x < titikTujuan.x && titikAwal.y ==
	titikTujuan.y){
20	direction ='right';
	point = horVer;
21	}else if(titikAwal.x < titikTujuan.x && titikAwal.y >
	titikTujuan.y){
22	direction ='upright';
	point = diagonal;
	}
23	console.log(point);
24	return

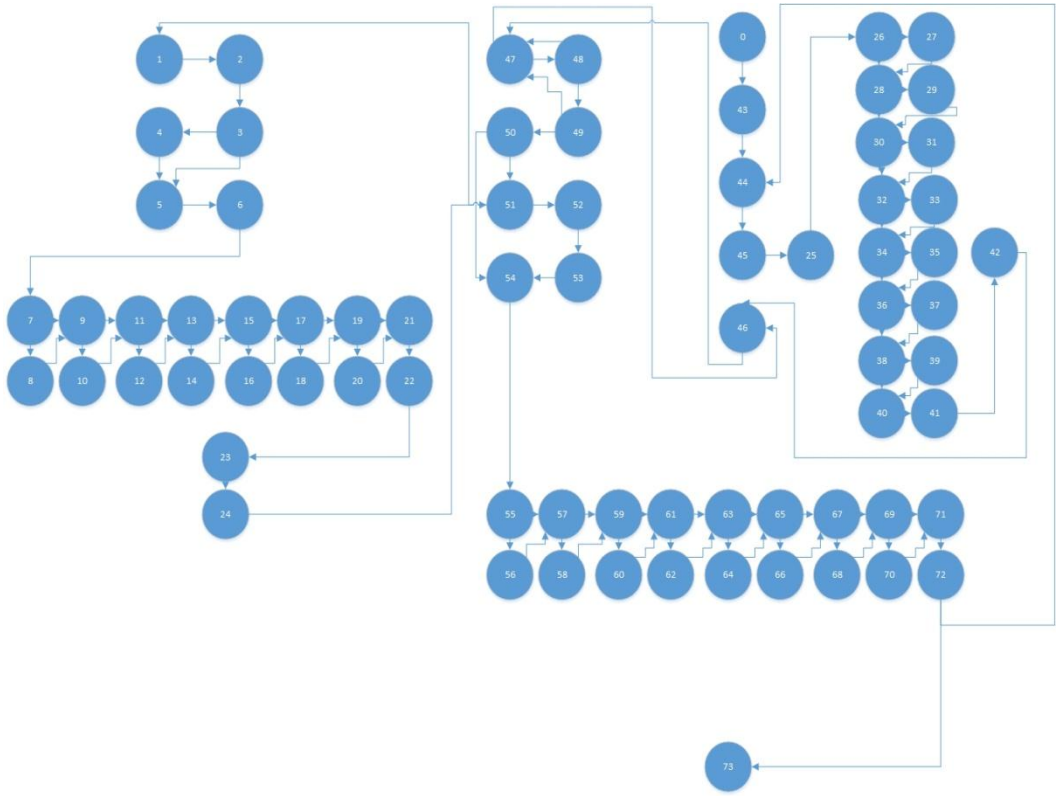
	{movePoint:point,moveDirection:direction};
	},
25	getNeighbour: function(titik){
	var neighbour = [];
26	if(titik.y-1 >=0){
27	neighbour.push({x:titik.x, y:titik.y-1});
	}
28	if(titik.x + 1 < mapArray[0].length && titik.y-1>=0){
29	neighbour.push({x:titik.x+1, y:titik.y-1});
	}
30	if(titik.x + 1 < mapArray[0].length){
31	neighbour.push({x:titik.x+1, y:titik.y});
	}
32	if(titik.x + 1 < mapArray[0].length && titik.y+1 <
	mapArray.length){
33	neighbour.push({x:titik.x+1, y:titik.y+1});
	}
34	if(titik.y + 1 <mapArray.length){
35	neighbour.push({x:titik.x, y:titik.y+1});
	}
36	if(titik.x-1 >=0 && titik.y +1 <mapArray.length){
37	neighbour.push({x:titik.x-1, y:titik.y+1});
	}
38	if(titik.x-1 >= 0){
39	neighbour.push({x:titik.x-1, y:titik.y});
	}
40	if(titik.x-1 >=0 && titik.y-1 >=0){
41	neighbour.push({x:titik.x-1, y:titik.y-1});
	}
42	return neighbour;
	}
	,
43	findPath: function(titikAwal,titikTujuan){

```
        var currentNode = titikAwal;
        var tmpPath = [];
        var tmpBestPath = [];
        var minScore = 99999;
        var tmpBestNode = null;
        var found =false;
        var skippedIndex = 0;
44      while(!found){
45          var neighbour = this.getNeighbour(currentNode);
46          for(var i=0;i<neighbour.length;i++){
                var skip = false;
47          for(var j=0;j<tmpPath.length;j++){
48                  if(neighbour[i].x == tmpPath[j].x && neighbour[i].y ==
    tmpPath[j].y){
49                      skip = true;
                        skippedIndex = j;
                    }
                }
50              if(!skip){
51              var score =
    this.normalDistance(currentNode,neighbour[i]).movePoint +
    this.euclideanDistance(currentNode,titikTujuan);
52                  if(score < minScore){
53                      minScore = score;
                tmpBestNode = neighbour[i];
                    }
                }
            }
54          minScore = 99999;
            tmpPath.push(currentNode);
            tmpBestPath.push(tmpBestNode);
            currentNode = tmpBestNode;
55          if(currentNode.x+1 == titikTujuan.x
```

```
                && currentNode.y+1 == titikTujuan.y){
56               found = true;
                 tmpBestPath.push(titikTujuan);
             }
57           if(currentNode.x == titikTujuan.x
                && currentNode.y+1 == titikTujuan.y){
58               found = true;
                 tmpBestPath.push(titikTujuan);
             }
59           if(currentNode.x-1 == titikTujuan.x
                && currentNode.y+1 == titikTujuan.y){
60               found = true;
                 tmpBestPath.push(titikTujuan);
             }
61           if(currentNode.x-1 == titikTujuan.x
                && currentNode.y == titikTujuan.y){
62               found = true;
             tmpBestPath.push(titikTujuan);
             }
63           if(currentNode.x-1 == titikTujuan.x
                && currentNode.y-1 == titikTujuan.y){
64               found = true;
                 tmpBestPath.push(titikTujuan);
             }
65           if(currentNode.x == titikTujuan.x
                && currentNode.y-1 == titikTujuan.y){
66               found = true;
                 tmpBestPath.push(titikTujuan);
             }
67           if(currentNode.x+1 == titikTujuan.x
                && currentNode.y-1 == titikTujuan.y){
68               found = true;
                 tmpBestPath.push(titikTujuan);
             }
69           if(currentNode.x+1 == titikTujuan.x
                && currentNode.y == titikTujuan.y){
```

70	found = true;
	tmpBestPath.push(titikTujuan);
	}
71	if(currentNode.x == tmpPath[j].x && currentNode.y ==
	tmpPath[j].y){
72	found = true;
	tmpBestPath.push(tmpBestNode);
	}
	}
73	return tmpBestPath;
	}
	});

2. *Flowgraph*



Gambar 4.14 *Flowgraph*

Keterangan :

= Menggambarkan Kondisi (N)

= Menggambarkan Aksi (E)

3. *Cyclomatic Complexity* (VG)

Berdasarkan pada gambar 4.14 diatas, dapat dihitung nilai *cyclomatic complexity* (VG) dengan perhitungan sebagai berikut :

E = 102

N = 73

V(G) = E – N + 2

V(G) = 102 – 73 + 2

V(G) = (29) + 2 = 31

4. ***Independent Path***

Path 1 = 0-43-44-45-25-26-27-28-29-30-31-32-33-34-35-36-37-38-39-40-41-42-46-47-48-49-50-51-52-53-54-1-2-3-4-5-6-7-8-9-10-11-12-13-14-15-16-17-18-19-20-21-22-23-24-51-52-53-54-55-56-57-58-59-60-61-62-63-64-65-66-67-68-69-70-71-72-73

Path 2 = 0-43-44-45-25-26-27-28-29-30-31-32-33-34-35-36-37-38-39-40-41-42-46-47-48-49-50-51-1-2-3-4-5-6-7-8-9-10-11-12-13-14-15-16-17-18-19-20-21-22-23-24-51-52-53-54-55-56-57-58-59-60-61-62-63-64-65-66-67-68-69-70-71-72-73

Path 3 = 0-43-44-45-25-26-28-30-32-34-36-38-40-41-42-46-47-48-49-50-51-1-2-3-4-5-6-7-8-9-10-11-12-13-14-15-16-17-18-19-20-21-22-23-24-51-52-53-54-55-56-57-58-59-60-61-62-63-64-65-66-67-68-69-70-71-72-73

Path 4 = 0-43-44-45-25-26-28-30-32-34-36-38-40-41-42-46-47-48-49-50-51-1-2-3-4-5-6-7-9-11-13-15-17-19-21-22-23-24-51-52-53-54-55-56-57-58-59-60-61-62-63-64-65-66-67-68-69-70-71-72-73

Path 5 = 0-43-44-45-25-26-28-30-32-34-36-38-40-41-42-46-47-48-49-50-51-1-2-3-4-5-6-7-9-11-13-15-17-19-21-22-23-24-51-52-53-54-55-57-59-61-63-65-67-69-71-72-73

Path 6 = 0-43-44-45-25-26-28-30-32-34-36-38-40-41-42-46-47-48-49-50-51-52-53-54-55-56-57-58-59-60-61-62-63-64-65-66-67-68-69-70-71-72-73

Path 7 = 0-43-44-45-25-26-28-30-32-34-36-38-40-41-42-46-47-48-49-50-51-52-53-54-55-57-59-61-63-65-67-69-71-72-73

5. Graph Matriks

	0	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	40	41	42	43	44	45	46	47	48	49	50	51	52	53	54	55	56	57	58	59	60	61
0																																											1																										
1			1																																																																		
2				1																																																																		
3					1	1																																																															
4						1																																																															
5							1																																																														
6								1																																																													
7									1	1																																																											
8										1																																																											
9											1	1																																																									
10												1																																																									
11													1	1																																																									
12														1																																																							
13															1	1																																																					
14																1																																																					
15																	1	1																																																				
16																		1																																																		
17																			1	1																																																		
18																				1																																																
19																					1	1																																															
20																						1																																														
21																							1																																													
22																								1																																												
23																									1																																											
24																																																																						
25																	1																																																			
26																		1	1																																																		
27																			1																																																	
28																				1	1																																																
29																						1																																														
30																																																																					
31																																																																					
32																																																																					
33																																																																					
34																																																																					
35																																																																					
36																																																																					
37																																																																					
38																																																																					
39																																																																					
40																																																																					
41																																																																					
42																																																																					
43																																																																					
44																																																																					
45																																																																					
46																																																																					
47																																																																					
48																																																																					
49																																																																					
50																																																																					
51																																																																					
52																																																																					
53																																																																					
54																																																																					
55																																																																					
56																																																																					

V(G) = Jumlah Graph Matriks + 1

V(G) = 30 + 1

V(G) = 31

6. Kesimpulan

Berdasarkan pengujian pada penerapan algoritma A* diatas, hasilnya dapat diamati pada tabel 4.24 berikut ini.

Tabel 4.24 Hasil Uji *White Box* Penerapan Algoritma A Star

Path No	Node (n)	Hasil yang diharapkan	Hasil sesuai uji kasus	Keterangan
1	0-43-44-45-25-26-27-28-29-30-31-32-33-34-35-36-37-38-39-40-41-42-46-47-48-49-50-51-52-53-54-1-2-3-4-5-6-7-8-9-10-11-12-13-14-15-16-17-18-19-20-21-22-23-24-51-52-53-54-55-56-57-58-59-60-61-62-63-64-65-66-67-68-69-70-71-72-73	Membuka semua suksesor pada setiap neighbour node sekitar untukdijadikan open list	Membuka semua suksesor pada setiap neighbour node sekitar untukdijadikan open list	[√] Terlewati [ ] Tidak terlewati
2	0-43-44-45-25-26-27-28-29-30-31-32-33-34-35-36-37-38-39-40-41-42-46-47-48-49-50-51-1-2-3-4-5-6-7-8-9-10-11-12-13-14-15-16-17-18-19-20-21-22-23-24-51-52-53-54-55-56-57-58-59-60-61-62-63-64-65-66-67-68-69-70-71-72-73	Menentukan open list yang baru dan jalur sebelumnya masuk closed list	Menentukan open list yang baru dan jalur sebelumnya masuk closed list	[√] Terlewati [ ] Tidak terlewati
3	0-43-44-45-25-26-	Memeriksa	Memeriksa	[√] Terlewati

	28-30-32-34-36-38-40-41-42-46-47-48-49-50-51-1-2-3-4-5-6-7-8-9-10-11-12-13-14-15-16-17-18-19-20-21-22-23-24-51-52-53-54-55-56-57-58-59-60-61-62-63-64-65-66-67-68-69-70-71-72-73	neighbour node pada open list yang baru	neighbour node pada open list yang baru	[ ] Tidak terlewati
4	0-43-44-45-25-26-28-30-32-34-36-38-40-41-42-46-47-48-49-50-51-1-2-3-4-5-6-7-9-11-13-15-17-19-21-22-23-24-51-52-53-54-55-56-57-58-59-60-61-62-63-64-65-66-67-68-69-70-71-72-73	Menentukan open list yang baru dan jalur sebelumnya masuk closed list	Menentukan open list yang baru dan jalur sebelumnya masuk closed list	[√] Terlewati [ ] Tidak terlewati
5	0-43-44-45-25-26-28-30-32-34-36-38-40-41-42-46-47-48-49-50-51-1-2-3-4-5-6-7-9-11-13-15-17-19-21-22-23-24-51-52-53-54-55-57-59-61-63-65-67-69-71-72-73	Memeriksa neighbour node pada open list yang baru	Memeriksa neighbour node pada open list yang baru	[√] Terlewati [ ] Tidak terlewati
6	0-43-44-45-25-26-28-30-32-34-36-38-40-41-42-46-47-48-49-50-51-52-53-54-55-56-57-58-59-60-61-62-63-64-65-66-67-68-69-70-71-72-73	Menentukan open list yang baru dan jalur sebelumnya masuk closed list	Menentukan open list yang baru dan jalur sebelumnya masuk closed list	[√] Terlewati [ ] Tidak terlewati

7	0-43-44-45-25-26-28-30-32-34-36-38-40-41-42-46-47-48-49-50-51-52-53-54-55-57-59-61-63-65-67-69-71-72-73	Mendapatkan jalur terpendek dan menghentikan pencarian	Mendapatkan jalur terpendek dan menghentikan pencarian	[√] Terlewati [ ] Tidak terlewati

Berdasarkan pada tabel 4.26 diatas, dapat disimpulkan bahwa pengujian pada tahap pencarian algoritma A* berjalan dengan baik serta hasil dari perhitungan *cyclomatic complexity* dan graph matriks memiliki hasil yang sama yaitu 31. Oleh karena itu proses penerapan algoritma A* pada game yang dibangun berjalan baik seperti yang diharapkan.

4.2.4 Kesimpulan Pengujian Alpha

Berdasarkan hasil pengujian yang dilakukan, disimpulkan bahwa *game* yang dibangun dapat berfungsi sesuai dengan harapan. Jadi, semua fungsional yang ada pada *game* ini dapat menghasilkan keluaran yang diinginkan.

4.2.5 Pengujian Beta

Pada tahap pengujian ini, dilakukan pengujian secara objektif dengan menguji *game* yang dibangun kepada pengguna secara langsung. Metode yang dilakukan dalam tahap ini yaitu metode kuantitatif berdasarkan data dari pengguna. Pengumpulan data dengan kuesioner dilakukan untuk mengetahui secara garis besar daya tarik pengguna terhadap *game* yang dibangun, dalam hal ini penggunanya adalah siswa sekolah dasar dengan mengambil sampel sebanyak 30 data responden secara acak dengan rentang usia 6 hingga 11 tahun dengan pertanyaan sebanyak 10 butir. Berikut ini adalah butir – butir pertanyaan yang diberikan kepada responden.

1. Apakah game ini menyenangkan untuk adik? (P1)
2. Apakah game ini mudah untuk dimainkan? (P2)
3. Apakah petunjuk cara bermain yang ada memudahkan adik untuk bermain game ini? (P3)
4. Apakah adik tertarik untuk memainkan game ini lagi? (P4)

5. Apakah tampilan animasi game yang adik mainkan menarik? (P5)
6. Apakah lebih mudah memainkan game ini melalui internet / web? (P6)
7. Apakah adik memahami alur cerita permainan dari game ini? (P7)
8. Apakah setelah memainkan game ini adik jadi mengetahui tokoh superhero Indonesia? (P8)
9. Apakah adik tertarik pada tokoh – tokoh superhero Indonesia setelah memainkan game ini? (P9)
10. Apakah karakter utama yang ditampilkan dapat menjadi salah satu wakil superhero Indonesia? (P10)

Berdasarkan pada butir pertanyaan yang diberikan diatas, maka disusun jawaban yang diberikan responden pada tabel 4.25 berikut ini.

Tabel 4.25 Hasil Perhitungan Kuesioner

Responden	Kelas	Umur	1	2	3	4	5	6	7	8	9	10
1	4a	9	1	1	1	1	1	1	0	1	1	1
2	4a	9	1	1	1	1	1	1	1	1	1	1
3	4a	9	1	1	1	1	1	1	1	1	1	1
4	4a	11	1	1	1	1	1	1	1	1	1	1
5	4a	9	1	1	1	1	1	1	1	1	1	1
6	4a	9	1	1	0	1	1	1	1	1	1	1
7	4a	9	1	1	1	1	1	1	1	1	1	1
8	4a	9	1	0	1	1	1	1	0	1	1	1
9	4a	10	1	1	1	1	1	0	1	1	1	1
10	4a	9	1	1	1	1	1	0	1	1	1	1
11	1b	6	1	1	1	1	1	1	1	1	1	1
12	1b	6	1	1	1	1	1	1	1	1	1	1
13	1b	6	1	0	0	0	0	0	1	1	1	1
14	1b	6	1	1	1	1	1	1	1	1	1	1
15	1b	6	1	0	0	0	1	1	1	1	1	1
16	1b	6	1	0	0	1	1	1	1	1	1	1
17	1b	6	1	1	0	1	1	1	1	1	1	1
18	1b	6	1	0	0	1	1	1	1	1	1	1
19	1b	6	1	0	1	1	1	1	1	1	1	1
20	1b	6	1	1	1	1	1	1	1	1	1	1
21	3	9	1	1	1	1	1	0	1	1	1	1
22	3	8	1	1	1	1	1	1	1	1	1	1
23	3	9	1	1	1	1	0	0	1	1	1	1

24	3	9	1	1	1	1	1	1	1	1	1	1
25	3	9	1	1	0	1	1	1	1	1	1	1
26	3	8	1	1	0	1	0	1	1	1	1	1
27	3	8	1	1	1	1	1	1	1	1	1	1
28	3	9	1	0	1	1	1	1	1	1	1	1
29	3	9	1	1	1	1	1	1	1	1	1	1
30	3	8	1	1	1	1	1	0	1	1	1	1
Total Ya			**30**	**23**	**22**	**28**	**27**	**24**	**28**	**30**	**30**	**30**
Total Tidak			**0**	**7**	**8**	**2**	**3**	**6**	**2**	**0**	**0**	**0**
Persentase	**Jawaban a**		**100 %**	**77 %**	**73 %**	**93 %**	**90 %**	**80 %**	**93 %**	**100 %**	**100 %**	**100 %**
Persentase	**Jawaban b**		**0%**	**23 %**	**27 %**	**7%**	**10 %**	**20 %**	**7%**	**0%**	**0%**	**0%**

Setelah mendapatkan hasil dari jawaban responden yang dikumpulkan, hal yang dilakukan selanjutnya adalah menghitung rata – rata total jawaban Ya dan total jawaban Tidak pada tabel 4.26 dibawah ini.

Tabel 4.26 Hasil Perhitungan Rata – Rata Jawaban

Pertanyaan	(%) Jawaban Ya	(%) Jawaban Tidak
P1	30	0
P2	23	7
P3	22	8
P4	28	2
P5	27	3
P6	24	6
P7	28	2
P8	30	0
P9	30	0
P10	30	0
TOTAL	**272**	**28**
RATA-RATA	**27,2**	**2,8**

Nilai untuk jawaban Ya : 1

Nilai Untuk jawaban Tidak : 0

Dikonversikan dalam persentase :

Jawaban Ya : 1x100% = 100%

Jawaban Tidak : 0x100% = 0% (sehingga tidak perlu dihitung)

Perhitungan jawaban Ya dari kuesioner :

Jawaban Ya rata – rata : 27,2 / 30 x 100% = 90,7%

4.2.5.5 Kesimpulan Pengujian Beta

Berdasarkan pada hasil kuesioner dan melalui tahap penghitungan dan pengukuran menggunakan skala *Guttman*, dapat diambil kesimpulan bahwa *game* yang dibangun memenuhi tujuan yaitu membangun *game* yang dapat memperkenalkan tokoh *superhero* yang pernah ada di Indonesia kepada anak – anak serta penggunaan basis *web* memudahkan pemain dalam mengakses dan memainkan *game*.