BAB 4

IMPLEMENTASI DAN PENGUJIAN

4.1 Implementasi Program

Implementasi program adalah implementasi jalannya sistem yang telah dibuat sehingga diharapkan dengan adanya implementasi ini dapat dipahami jalannya suatu sistem.

Aplikasi ini terdiri dari beberapa menu. Diantaraya menu *Eiji : The Lost Canvas Configuration* yang terdiri atas :

1. *Graphics Config.*
2. *Input Config.*

Sedangkan didalam *game* terdapat menu *play game, history, credit,* dan *Guide.*

4.2 Perangkat Pendukung yang Digunakan

Dalam proses pembuatan aplikasi ini, ternyata membutuhkan perangkat keras (*Hardware*) dan perangkat lunak (*Software*). Berikut penjelasan dari perangkat keras dan perangkat lunak yang digunakan.

4.2.1 Perangkat Keras (*Hardware*)

Perangkat keras yang digunakan selama membangun *game* ini memiliki spesifikasi sebagai berikut :

1. AMD Phenom II X3 2.8 GHz.
2. RAM 5 GB.
3. VGA Nvidia Gforce 1GB
4. Hardisk 1.5 TB
5. Monitor
6. Keyboard dan mouse.

4.2.2 Perangkat Lunak (*Software*)

Perangkat lunak yang digunakan selama membangun *game* ini memiliki spesifikasi sebagai berikut :

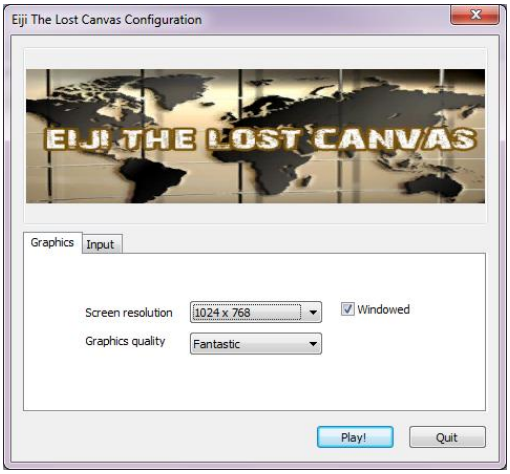
1. Sistem operasi : Windows 7.
2. *Game* engine : Unity.
3. IDE : Mono Development dan Visual Studio 2010.
4. Model 3D : Autodeks 3DS max 2012 dan Maya 2012.
5. Texture 2D : Adobe Photoshop CS4 .

4.3 Implementasi Aplikasi

4.3.1 Sisi *User*

Pada sisi *user*, penulis menyajikan tampilan *game* yang telah dibuat.

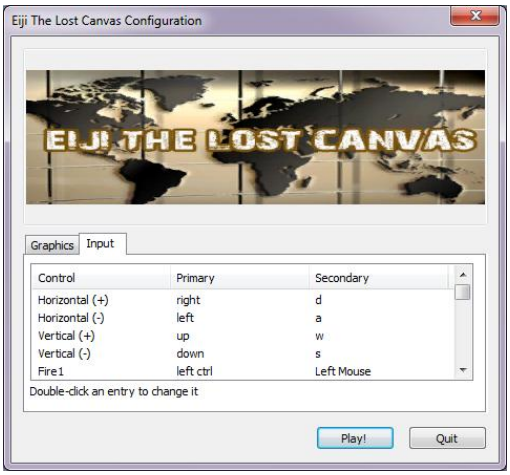
1. Tampilan Konfigurasi Grafis



Gambar 4.1 Tampilan Konfigurasi Grafis**.**

Gambar diatas menampilkan tampilan konfigurasi grafis ketika aplikasi dijalankan. Pada gambar diatas *user* bisa melakukan konfigurasi grafis yang akan dipakai. Mulai dari resolusi layar, kualitas grafis dan mode layar yang digunakan. *User* bisa langsung memainkan permainan setelah klik tombol *Play!* dan *Quit* untuk menutup aplikasi.

2. Tampilan Konfigurasi *Input*



Gambar 4.2 Tampilan Konfigurasi *Input*.

Gambar diatas menampilkan tampilan konfigurasi *input*. Pada gambar diatas *user* bisa melakukan konfigurasi *input* tombol yang akan dipakai. *User* bisa langsung memainkan permainan setelah klik tombol *Play!* dan *Quit* untuk menutup aplikasi.

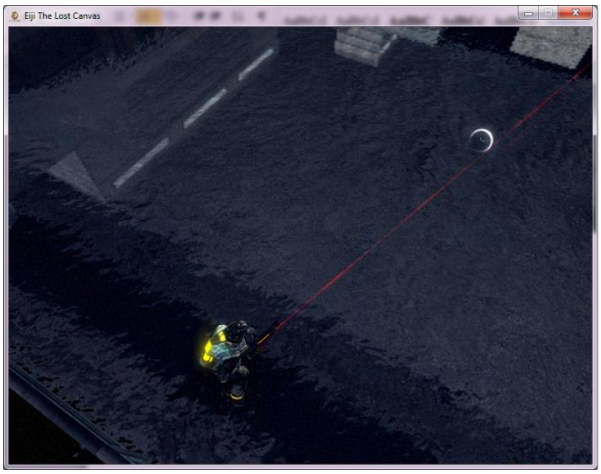
3. Tampilan Utama



Gambar 4.3 Tampilan Utama.

Gambar diatas adalah tampilan utama dalam *game Eiji The : Lost Canvas*. *User* disuguhkan beberapa menu yang bisa dipilih. Diantaranya tombol *Play game* yang berguna untuk memulai permainan, tombol *Credit* berguna untuk menampilkan biografi pembuat *game*, tombol *Guide* berguna untuk menampilkan petunjuk penggunaan *game*, tombol *history* berguna untuk menampilkan sekilas cerita *game*, dan tombol Quit *game* berguna untuk keluar dari *game*.

4. Tampilan *Stage* 1



Gambar 4.4 Tampilan *Stage* 1.

Gambar diatas adalah tampilan *Stage* 1 dari *game* ini. Ketika *user* klik tombol Play *game* pada Tampilan utama, *user* akan memulai permainan dari *Stage* 1. Dimana *user* harus menyelesaikan permainan *Stage* 1 dengan menghancurkan musuh yang ada dan akan berpindah ke *Stage* 2. Namun ketika player mati dalam perjalanan *Stage* 1, maka *player* akan kembali ke *checkpoint* yang telah dilewati.

5. Tampilan *Stage* 2



Gambar 4.5 Tampilan *Stage* 2**.**

Gambar diatas adalah tampilan *Stage* 2 dari *game* ini. Ketika *user* telah menyelesaikan *Stage* 1, maka *user* akan berpindah tempat ke *Stage* 2. *User* harus menyelesaikan permainan *Stage* 2 dengan menghancurkan musuh yang ada dan akan berpindah ke *Stage* 3. Namun ketika palyer mati dalam perjalanan *Stage* 2, maka *player* akan kembali ke *checkpoint* yang telah dilewati.

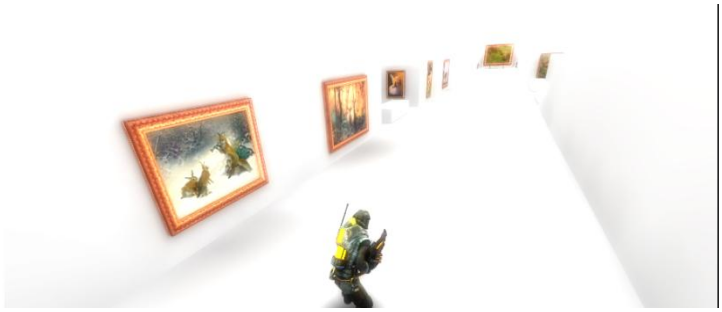
6. Tampilan *Stage* 3



Gambar 4.6 Tampilan *Stage* 3**.**

Gambar diatas adalah tampilan *Stage* 3 dari *game* ini. Ketika *user* telah menyelesaikan *Stage* 2, maka *user* akan berpindah tempat ke *Stage* 3. *User* harus menyelesaikan permainan *Stage* 3 dengan menghancurkan musuh yang ada dan akan berpindah ke *Final Stage* . Namun ketika palyer mati dalam perjalanan *Stage* 3, maka *player* akan kembali ke *checkpoint* yang telah dilewati.

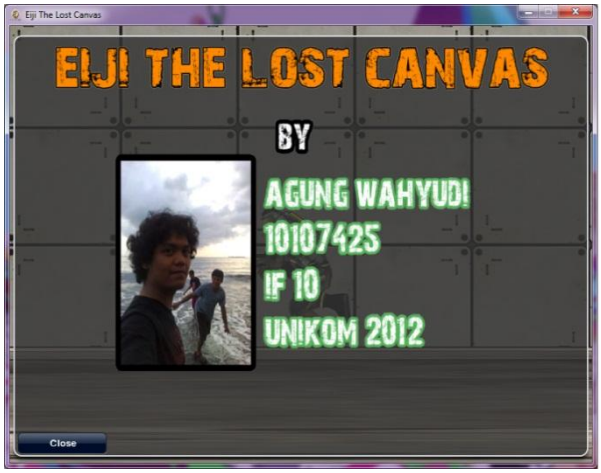
7. Tampilan *Final Stage*



Gambar 4.7 Final *Stage*.

Gambar diatas adalah tampilan *Final Stage* dari *game* ini. Ketika *user* telah menyelesaikan *Stage* 3, maka *user* akan berpindah tempat ke *Final Stage*.

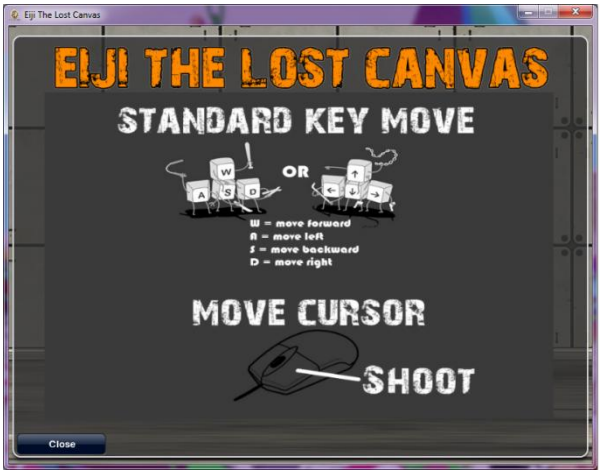
8. Tampilan *Credit*



Gambar 4.8 Tampilan *Credit*.

Tampilan diatas adalah tampilan yang menampilkan informasi pembuat *game*.

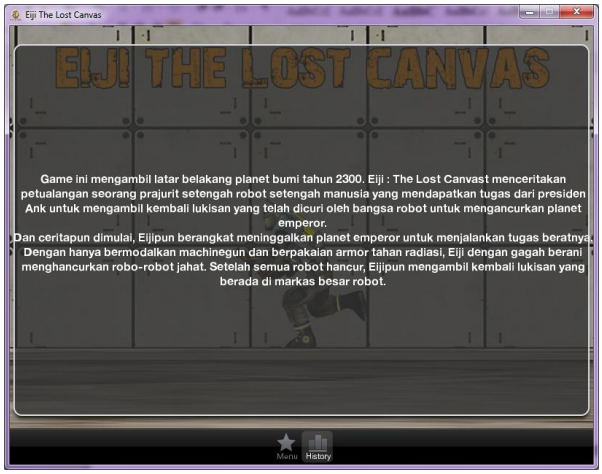
9. Tampilan *Guide*



Gambar 4.9 Tampilan *Guide*.

Tampilan diatas adalah tampilan yang menampilkan informasi petunjuk *game*.

10. Tampilan *History*

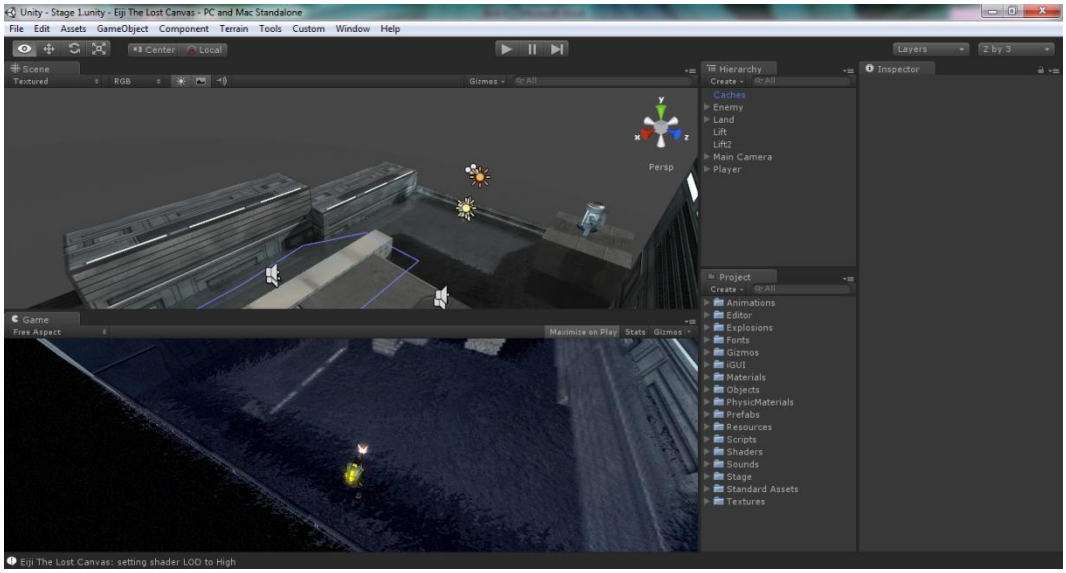


Gambar 4.10 Tampilan *History***.**

Tampilan diatas adalah tampilan yang menampilkan informasi *history game*.

4.3.2 Sisi Pembuat

Pada tampilan ini, penulis menampilkan unity *game engine* yang digunakan dalam pembuatan *game Eiji : The Lost Canvas*.



Gambar 4.11 Tampilan Pengembangan Game**.**

4.4 Pengujian Sistem

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

4.5 Pengujian Alpha (Fungsional)

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

4.5.1 Pengujian Dengan Menggunakan Black Box

Test case ini bertujuan untuk menunjukkan fungsi PL tentang cara beroperasinya, apakah pemasukan data keluaran telah berjalan sebagaimana yang diharapkan dan apakah informasi yang disimpan secara eksternal selalu dijaga kemutakhirannya.

a. Rencana pengujian

Rencana pengujian yang akan dilakukan apada aplikasi selengkapnya terlihat pada tabel dibawah ini.

Tabel 4.1 Rencana Pengujian Aplikasi**.**

Kelas Uji	Detail Pengujian	Jenis Uji
Graphics *Config*	Menampilkan informasi konfigurasi grafis	Black Box
Input Config	Menampilkan tampilan konfigurasi *input*	Black Box
Play Game	Menampilkan *Stage* 1 hingga *Final Stage*	Black Box
Credit	Menampilkan Tampilan *Credit*	Black Box
Guide	Menampilkan Tampilan *Guide*	Black Box
History	Menampilkan Tampilan *History*	Black Box
Player	Pengujian fungsi tombol dan mouse yang	Black Box

	digunakan untuk menggerakan *player*, regenerasi *Health* dan kembali ke *checkpoint* ketika mati	
Buzzer	Pengujian pengejaran *player*, regenerasi *Health*, menembak *player*	Black Box
Spider	Pengujian pengejaran *player*, penghancuran diri, regenerasi *Health*	Black Box
Mecha	Pengujian pengejaran *player*, regenerasi *Health*, menembak *player*	Black Box

b. Kasus dan hasil pengujian

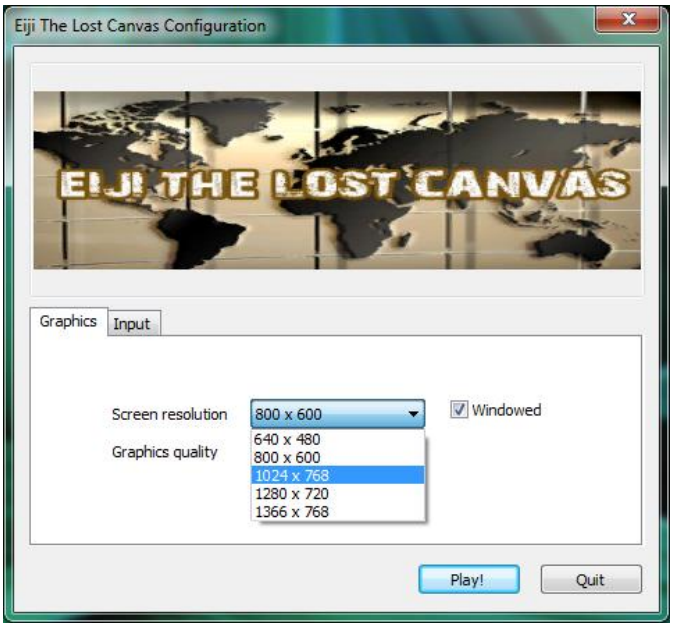
1. Pengujian *Graphics Config*

Pengujian ini dilakukan untuk membuktikan konfigurasi yang dipilih oleh *user* berfungsi secara normal pada *Eiji : The Lost Canvas*.



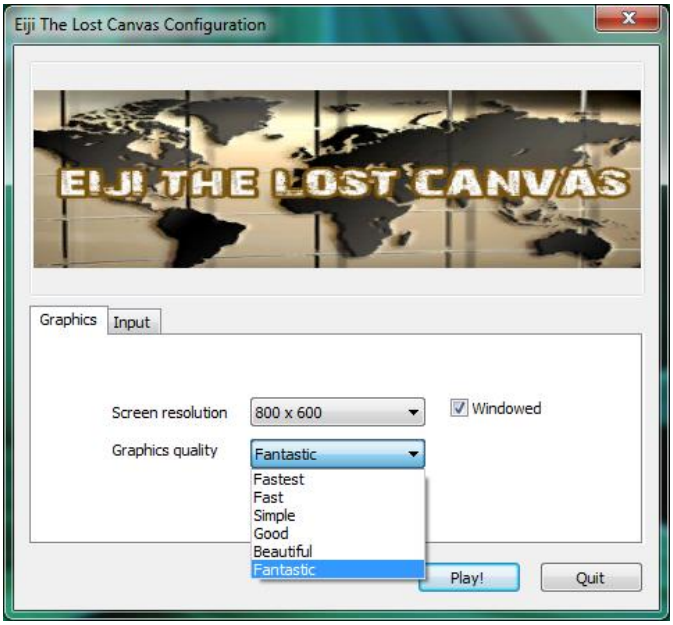
Gambar 4.12 Klik *dropdown menu***.**

Ketika *user* klik *dropdown srceen resolution*, maka *user* bisa memilih resolusi yang akan digunakan.



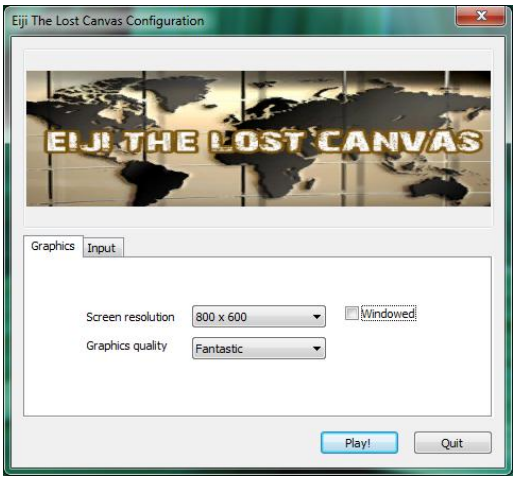
Gambar 4.13 *User* memilih resolusi.

Dan ketika *user* klik *dropdown menu Graphics Quality*, maka *user* bisa memilih kualitas grafis yang akan digunakan.



Gambar 4.14 Memilih kualitas grafis**.**

Dan ketika *user* klik checkbox, *user* bisa memilih mode tampilan yang akan digunakan, baik *windowed* yang berarti tidak *full screen* dan *uncheck* pada *checkbox* berarti tampilan akan *full screen*.



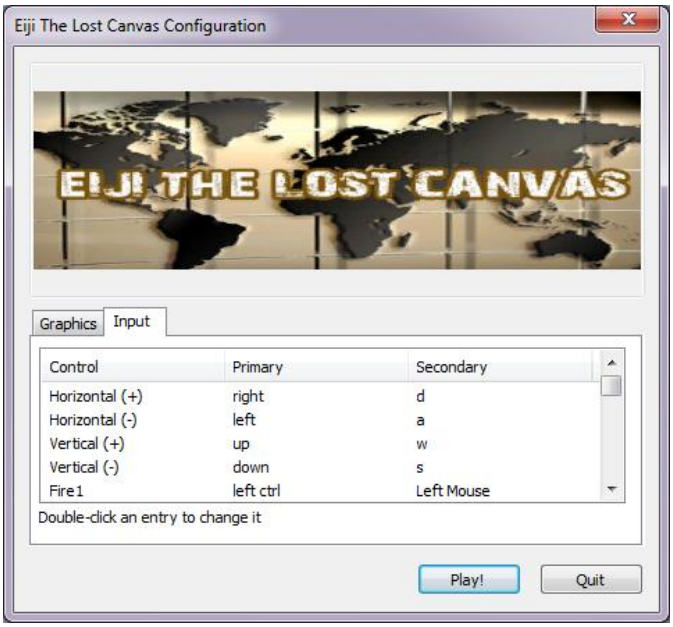
Gambar 4.15 Memilih mode tampilan**.**

Tabel 4.2 Pengujian *Graphics Config***.**

Kasus dan hasil uji (data normal)			
Datan masukan	**Yang diharapkan**	**Pengamatan**	**Kesimpulan**
Graphics *Config*	Menampilkan informasi konfigurasi grafis	informasi konfigurasi grafis ditampilkan	[√] diterima [ ] ditolak

2. Pengujian *Input Config*

Pengujian ini dilakukan untuk membuktikan konfigurasi *input* yang dipilih oleh *user* berfungsi secara normal pada *Eiji : The Lost Canvas*.



Gambar 4.16 konfigurasi input**.**

Tabel 4.3 Pengujian *Input Config***.**

Kasus dan hasil uji (data normal)			
Datan masukan	**Yang diharapkan**	**Pengamatan**	**Kesimpulan**
input Config	Menampilkan informasi konfigurasi *input*	informasi konfigurasi *input* ditampilkan	[√] diterima [ ] ditolak

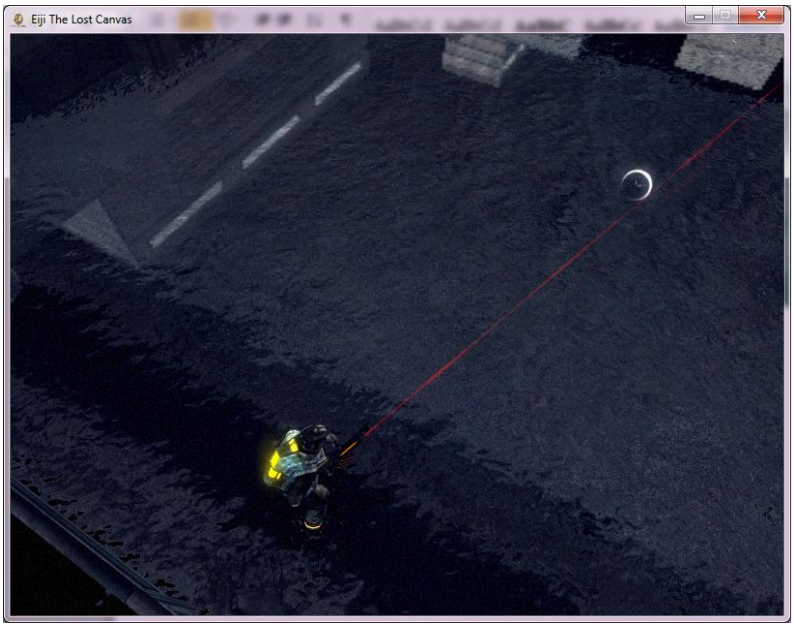
3. Pengujian Play *game*

Pengujian ini dilakukan untuk menampilkan *Stage* 1 hingga *Final Stage*.



Gambar 4.17 *Main menu*.

Ketika *user* klik button play *game*, maka *user* akan mulai bermain permainan *Eiji : The Lost Canvas* dari *Stage* 1 hingga *Final Stage*.



Gambar 4.18 *Stage* 1**.**

Tabel 4.4 Pengujian *Playgame***.**

Kasus dan hasil uji (data normal)			
Datan masukan	**Yang diharapkan**	**Pengamatan**	**Kesimpulan**
Play game	*User* bisa memainkan permainan *Eiji : The Lost Canvas* dari *Stage* 1 hingga *Final Stage*	informasi permainan yang ditampilkan	[√] diterima [ ] ditolak

4. Pengujian *Credit*

Pengujian ini dilakukan untuk menampilkan *Credit scene* pada Main menu ketika *user* klik *Credit* button.

Tabel 4.5 Pengujian *Credit*.

Kasus dan hasil uji (data normal)			
Datan masukan	**Yang diharapkan**	**Pengamatan**	**Kesimpulan**
Credit	*User* melihat tampilan *Credit*	informasi *Credit* yang ditampilkan	[√] diterima [ ] ditolak

5. Pengujian *Guide*

Pengujian ini dilakukan untuk menampilkan *Guide scene* pada Main menu ketika *user* klik *Guide button*.

Tabel 4.6 Pengujian *Guide*.

Kasus dan hasil uji (data normal)			
Datan masukan	**Yang diharapkan**	**Pengamatan**	**Kesimpulan**

Guide	*User* melihat tampilan *Guide*	informasi *Guide* yang ditampilkan	[√] diterima [ ] ditolak

6. Pengujian *History*

Pengujian ini dilakukan untuk menampilkan *History* pada Main menu ketika *user* klik *History button*.

Tabel 4.7 Pengujian *History***.**

Kasus dan hasil uji (data normal)			
Datan masukan	**Yang diharapkan**	**Pengamatan**	**Kesimpulan**
History	*User* melihat tampilan *History*	informasi *History* yang ditampilkan	[√] diterima [ ] ditolak

7. Pengujian *Player*

Pengujian ini bertujuan untuk membuktikan fungsi-fungsi tombol dan mouse yang digunakan untuk menggerakan *player*, regenerasi *Health* dan kembali ke *checkpoint* ketika mati.

Tabel 4.8 Pengujian *Player***.**

Kasus dan hasil uji (data normal)			
Datan masukan	**Yang diharapkan**	**Pengamatan**	**Kesimpulan**
Tombol W	*Player* bergerak maju	*Player* bergerak	[√] diterima [ ] ditolak
Tombol A	*Player* bergerak ke kiri	*Player* bergerak	[√] diterima [ ] ditolak
Tombol S	*Player* bergerak mundur	*Player* bergerak	[√] diterima [ ] ditolak
Tombol D	*Player* bergerak ke kanan	*Player* bergerak	[√] diterima [ ] ditolak
Mouse	*Player* berbelok sesuai kursor yang diarahkan	*Player* bergerak	[√] diterima [ ] ditolak
LeftClick	*Player* menembak sesuai kursor yang diarahkan	*Player* menembak	[√] diterima [ ] ditolak
Health	*Health player* berkurang ketika	*Health player*	[√] diterima

	tertembak		[ ] ditolak
Regeneration Health	*Health player* bertambah hingga penuh kembali	*Health player*	[√] diterima [ ] ditolak
Checkpoint	*Player* hidup kembali setelah mati	*Player Dead*	[√] diterima [ ] ditolak

8. Pengujian Buzzer

Pada pengujian ini, Buzzer sebagai NPC bisa mengejar *player* dan menembak hingga *player* mati dengan jarak yang telah ditentukan atau Buzzer sendiri yang mati diserang oleh *player*.

Tabel 4.9 Pengujian Buzzer**.**

Kasus dan hasil uji (data normal)			
Datan masukan	**Yang diharapkan**	**Pengamatan**	**Kesimpulan**
Player	Buzzer mengejar *player*	*Player* begerak	[√] diterima [ ] ditolak
Player	Buzzer menembak *player*	*Player* bergerak	[√] diterima

			[ ] ditolak
Player	Buzzer mati ditembak *player*	*Player* menembak	[√] diterima [ ] ditolak
Buzzer	Regenerasi *Health* apabila tertembak *player*	*Health* Buzzer	[√] diterima [ ] ditolak

9. Pengujian Spider

Pada pengujian ini, Spider sebagai NPC bisa mengejar *player* dan meledakan ketika dekat *player* dengan jarak yang telah ditentukan atau Spider sendiri yang mati diserang oleh *player*.

Tabel 4.10 Pengujian Spider**.**

Kasus dan hasil uji (data normal)			
Datan masukan	**Yang diharapkan**	**Pengamatan**	**Kesimpulan**
Player	Spider mengejar *player*	*Player* begerak	[√] diterima [ ] ditolak
Player	Spider meledakan diri ketika dekat plaer	*Player* bergerak	[√] diterima

			[ ] ditolak
Player	Spider mati ditembak *player*	*Player* menembak	[√] diterima [ ] ditolak
Spider	Regenerasi *Health* apabila tertembak *player*	*Health* Spider	[√] diterima [ ] ditolak
Spider	Spider kembali ke pos dia diam ketika jauh dari jangakauan pengejaran	Spider bergerak	[√] diterima [ ] ditolak

10. Pengujian Mecha

Pada pengujian ini, Mecha sebagai NPC bisa mengejar *player* ketika Mecha tertembak oleh *player* dan mendekati jarak jangakauan pengejaran, berpatroli, regenerasi *Health* dan musnah ditembak oleh *player*.

Tabel 4.11 Pengujian Mecha**.**

Kasus dan hasil uji (data normal)			
Datan masukan	**Yang diharapkan**	**Pengamatan**	**Kesimpulan**

Player	Mecha mengejar *player* ketika tertembak atau masuk jangkauan pengejaran	*Player* begerak	[√] diterima [ ] ditolak
Mecha	Mecha bergerak patroli	Mecha bergerak	[√] diterima [ ] ditolak
Player	Mecha mati ditembak *player*	*Player* menembak	[√] diterima [ ] ditolak
Mecha	Regenerasi *Health* apabila tertembak *player*	*Health* Mecha	[√] diterima [ ] ditolak
Mecha	Mecha kembali berpatroli setelah mengejar *player* setelah tidak terjangkau atau *player* terhalang objek penghalang dan ketika *player* mati	Mecha bergerak	[√] diterima [ ] ditolak

4.5.2 Pengujian Dengan Menggunakan White Box

Meramalkan cara kerja perangkat lunak secara rinci, karenanya logikal path (jalur logika) perangkat lunak akan ditest dengan menyediakan test case yang akan mengerjakan kumpulan kondisi dan atau

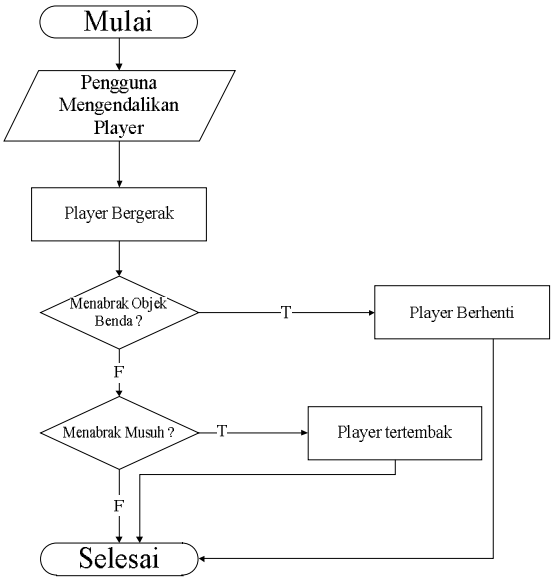
pengulangan secara spesifik. Secara sekilas dapat diambil kesimpulan white box testing merupakan petunjuk untuk mendapatkan program yang benar secara 100%.

A. Collision Detection

1. Player

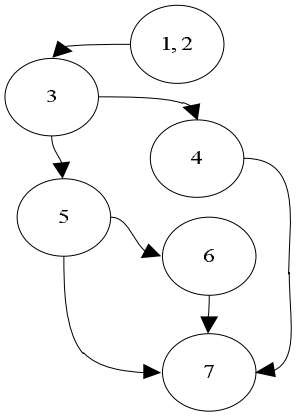
Menguji apakah *collision* pada entitas player telah terdeteksi dan hasilnya sesuai dengan yang diharapkan.

a. *Flowchart*



Gambar 4.19 *Flow chart collision* player.

b. *Flow Graph*



Gambar 4.20 *Flow graph collision* player.

c. *Cyclomatic Complexity* V(G)

E = 7, N = 6

V(G) = E – N + 2 = 7 – 6 + 2 = 2 + 2 = 3

d. *Independent Paths*

Path 1 = 1,2 – 3 – 4 – 7

Path 2 = 1,2 – 3 – 5 – 6 – 7

Path 3 = 1,2 – 3 – 5 – 7

e. Hasil Uji

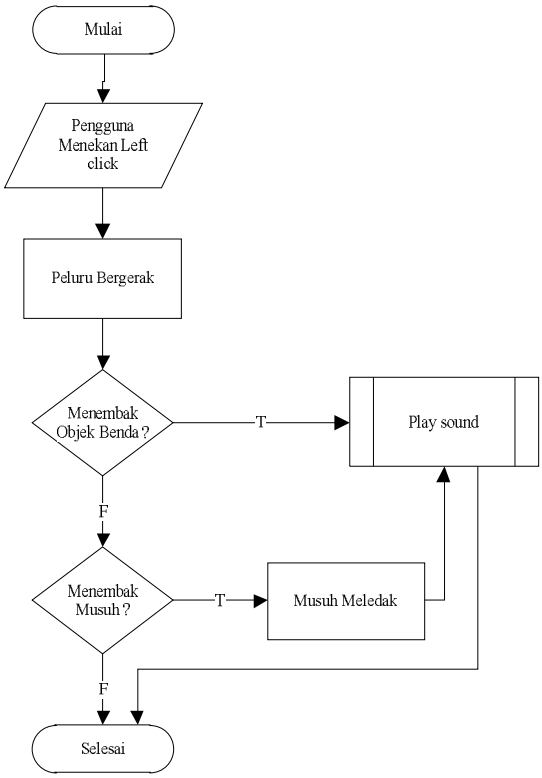
Tabel 4.12 Hasil uji *collision* player.

Path No.	***Node* (n), Input Condition**	***Excpected Result***	***Test Result***	***Conclusion***
1	1,2, (void) 3, (true) 4, (void) 7, (void)	Pergerakan player terhenti	Pergerakan player terhenti	*Path* Dikerjakan
2	1,2,(void) 3, (false) 5, (true) 6,(void) 7, (void)	Player ditembak musuh	Player ditembak musuh	*Path* Dikerjakan
3	1,2, (void) 3, (false) 5, (false) 7, (void)	Collision tidak terdeteksi	Collision tidak terdeteksi	*Path* Dikerjakan

2. Peluru

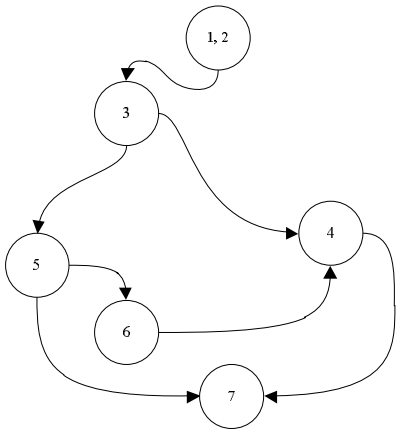
Menguji apakah *collision* pada entitas peluru telah terdeteksi dan hasilnya sesuai dengan yang diharapkan.

a. *Flow chart*



Gambar 4.21 *Flow chart collision* peluru.

b. *Flow graph*



Gambar 4.22 *Flow graph collision* peluru.

c. *Cyclomatic Complexity* V(G)

E = 7, N = 6

V(G) = E – N + 2 = 7 – 6 + 2 = 1 + 2 = 3

d. *Independent Paths*

Path 1 = 1,2 – 3 – 4 – 7

Path 2 = 1,2 – 3 – 5 – 6 – 4 – 7

Path 3 = 1,2 – 3 – 5 – 7

e. Hasil Uji

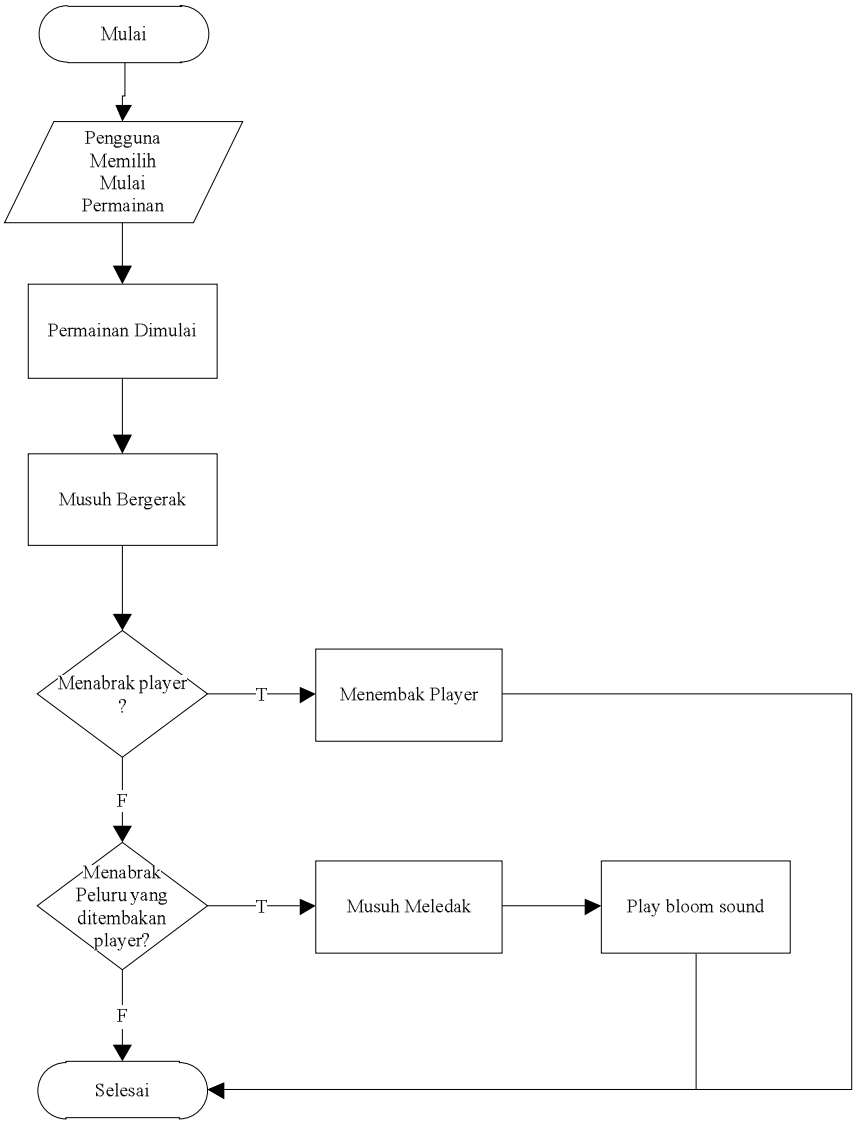
Tabel 4.13 Hasil uji *collision* peluru.

Path No.	***Node* (n), Input Condition**	***Excpected Result***	***Test Result***	***Conclusion***
1	1,2 (void) 3, (true) 4, (void) 7, (void)	Player menembak objek benda, terdengar suara	Player menembak objek benda, terdengar suara	*Path* Dikerjakan
2	1,2 (void) 3, (false) 6, (true) 4, (void) 7, (void)	Player menembak musuh, terdengar suara	Player menembak musuh, terdengar suara	*Path* Dikerjakan
3	1,2 (void) 3, (false) 5, (false) 7, (void)	Tidak terdeteksi *collision*	Tidak terdeteksi *collision*	*Path* Dikerjakan

3. Musuh

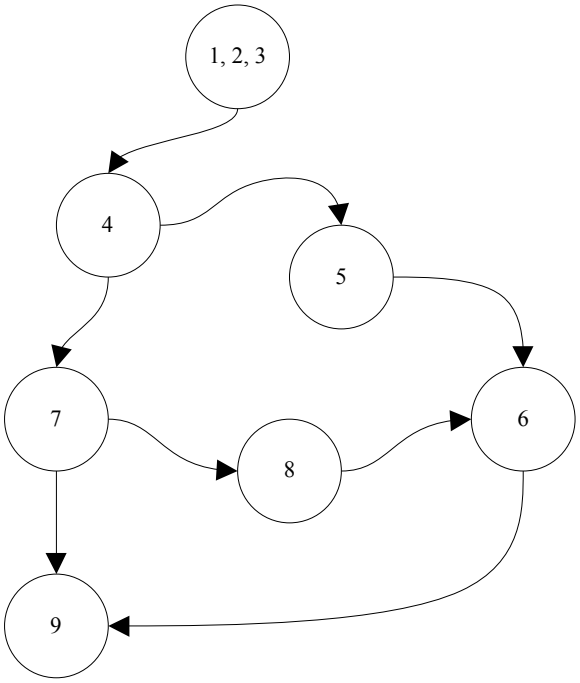
   Menguji apakah *collision* pada entitas Musuh telah terdeteksi dan hasilnya sesuai dengan yang diharapkan

a. *Flow chart*



Gambar 4.23 *Flow chart collision* musuh.

b. *Flow graph*



Gambar 4.24 *Flow graph collision* musuh.

c. *Cyclomatic Complexity* V(G)

E = 8, N = 7

V(G) = E – N + 2 = 8 – 7 + 2 = 1 + 2 = 3

d. *Independent Paths*

Path 1 = 1,2,3 – 4 – 5 – 6 – 9

Path 2 = 1,2,3 – 4 – 7 – 8 – 6 – 9

Path 3 = 1,2,3 – 4 – 7 – 9

e. Hasil uji

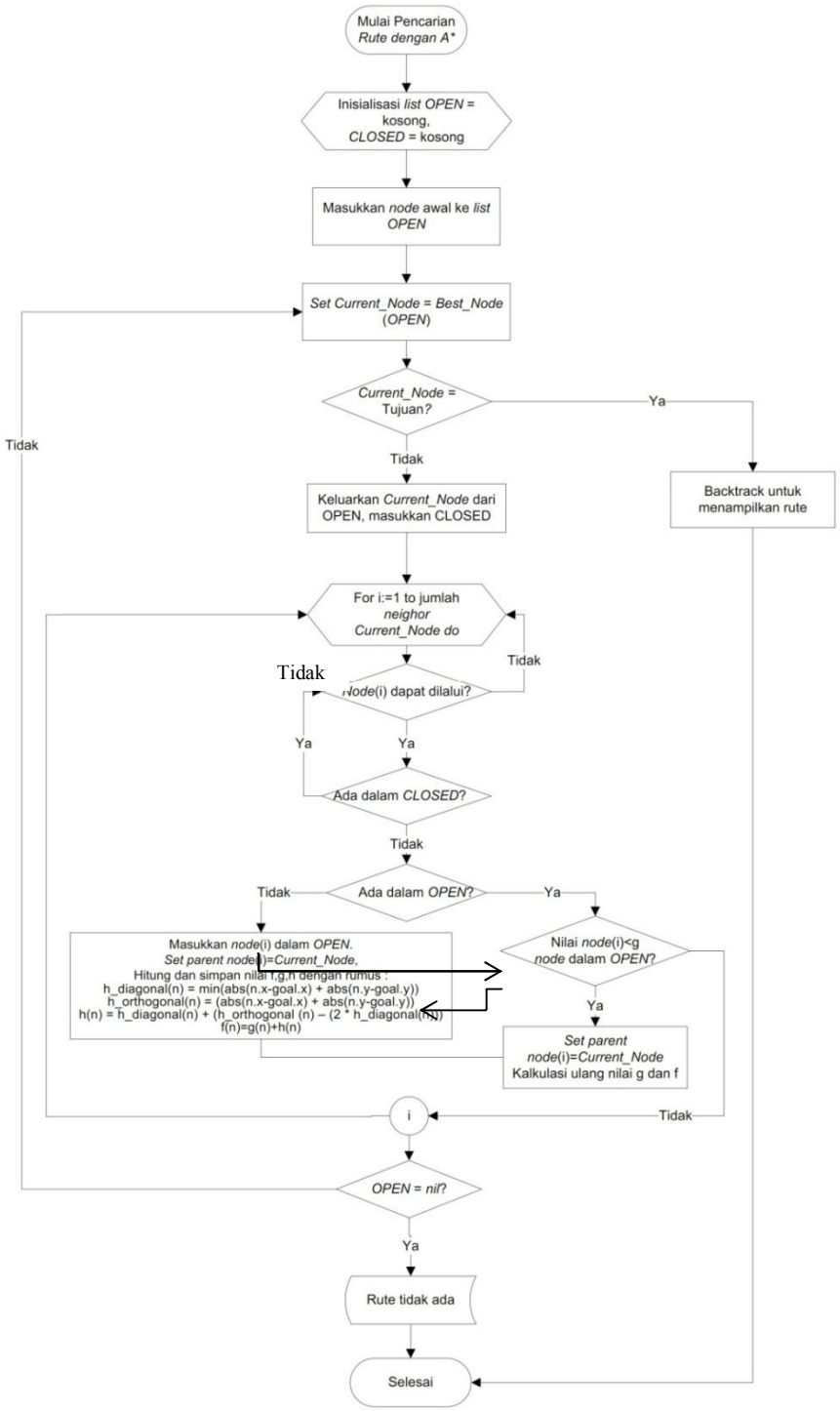
Tabel 4.14 Hasil uji *collision* musuh.

Path No.	***Node* (n), Input Condition**	***Excpected Result***	***Test Result***	***Conclusion***
1	1, 2, 3 (void) 4, (true) 5, (void) 6, (void) 9, (void)	Health player berkurang	Health player berkurang	*Path* Dikerjakan
2	1, 2, 3 (void) 4, (false) 7, (true) 8, (void) 6, (void) 9, (void)	Musuh Meledak, terdengar suara ledakan	Musuh Meledak, terdengar suara ledakan	*Path* Dikerjakan
3	1, 2, 3 (void) 4, (false) 7, (false) 9, (void)	*Collision* tidak terdeteksi	*Collision* tidak terdeteksi	*Path* Dikerjakan

B. Algoritma A*

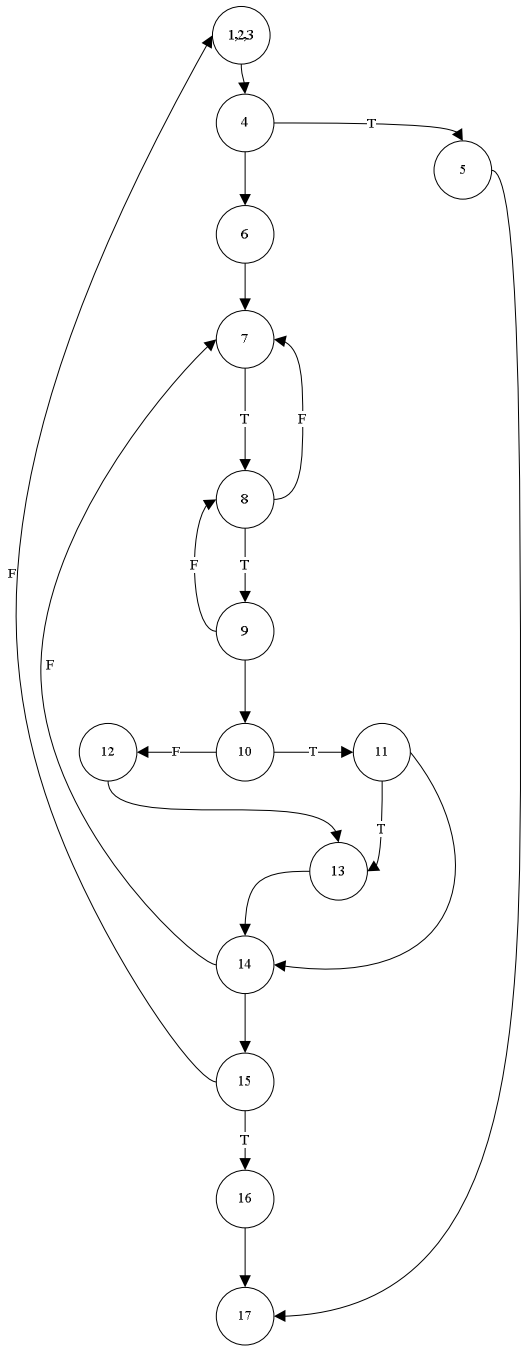
Pengujian Algoritma A* yang digunakan untuk mencari rute terpendek dalam hal pengejaran player.

a. *Flow chart*


Mulai Pencarian Rute dengan A*
Inisialisasi list OPEN = kosong, CLOSED = kosong
Masukkan node awal ke list OPEN
Set Current_Node = Best_Node (OPEN)
Current_Node = Tujuan?
Ya
Tidak
Backtrack untuk menampilkan rute
Keluarkan Current_Node dari OPEN, masukkan CLOSED
For i:=1 to jumlah neighor Current_Node do
Node(i) dapat dilalui?
Tidak
Ya
Ada dalam CLOSED?
Ada dalam OPEN?
Masukkan node(i) dalam OPEN.
Set parent node(i)=Current_Node,
Hitung dan simpan nilai f,g,h dengan rumus :
h_diagonal(n) = min(abs(n.x-goal.x) + abs(n.y-goal.y))
h_orthogonal(n) = (abs(n.x-goal.x) + abs(n.y-goal.y))
h(n) = h_diagonal(n) + (h_orthogonal (n) – (2 * h_diagonal(n))
f(n)=g(n)+h(n)
Nilai node(i)<g node dalam OPEN?
Set parent node(i)=Current_Node
Kalkulasi ulang nilai g dan f
i
OPEN = nil?
Rute tidak ada
Selesai

Gambar 4.25 *Flow chart* algoritma A*.

b. Flow Graph



Gambar 4.26 *Flow graph* algoritma A*.

c. *Cyclomatic Complexity* V(G)

E = 21, N = 15

V(G) = E – N + 2 = 21 – 15 + 2 = 8

d. *Independent Paths*

Path 1 = 1,2,3 – 4 – 5 – 17

Path 2 = 1,2,3 – 4 – 6 – 7 – 8 – 9 – 10 – 11 – 14 – 15 – 16 – 17

Path 3 = 1,2,3 – 4 – 6 – 7 – 8 – 9 – 10 – 11 – 13 – 14 – 15 – 16 –17

Path 4 = 1,2,3 – 4 – 6 – 7 – 8 – 9 – 10 – 11 –14 – 7

Path 5 = 1,2,3 – 4 – 6 – 7 – 8 – 9 – 10 – 11 – 14 – 15 – 1,2,3

Path 6 = 1,2,3 – 4 – 6 – 7 – 8 – 9 – 10 – 12 – 13 – 14 – 15 –16 –17

Path 7 = 1,2,3 – 4 – 6 – 7 – 8 – 9 – 10 – 12 – 13 – 14 – 6

Path 8 = 1,2,3 – 4 – 6 – 7 – 8 – 9 – 10 – 12 – 13 – 14 – 15 – 1,2,3

e. *Looping Paths*

Path L1 = 1,2,3 – 4 – 6 – 7 – 8

Path L2 = 1,2,3 – 4 – 6 – 7 – 8 – 1,2,3 – 4 – 6 – 7 – 8 – 9

f. Hasil uji

Tabel 4.15 Hasil uji algoritma A*.

Independent Paths				
Path No.	***Node* (n), Input Condition**	***Excpected Result***	***Test Result***	***Conclusion***
1	1,2,3 , (void) 4 , (true) 5 , (void) 17, (void)	Backtrack untuk menampilkan rute	Backtrack untuk menampilkan rute	*Path* Dikerjakan
2	1,2,3 , (void) 4 , (false) 6 , (void) 7, (void) 8, (true) 9, (true) 10, (true) 11, (false) 14, (true) 15, (true) 16, (void) 17, (void)	Perlihatkan node dari OPEN	Perlihatkan node dari OPEN	*Path* Dikerjakan

3	1,2,3 , (void) 4 , (false) 6 , (void) 7, (void) 8, (true) 9, (true) 10, (true) 11, (true) 13, (void) 14, (true) 15, (true) 16, (void) 17, (void)	Kalkulasikan ulang dilai g dan f	Kalkulasikan ulang dilai g dan f	*Path* Dikerjakan
4	1,2,3 , (void) 4 , (false) 6 , (void) 7, (void) 8, (true) 9, (true) 10, (true) 11, (false) 14, (false) 7, (conditional)	Menghitung jumlah current node	Menghitung jumlah current node	*Path* Dikerjakan
5	1,2,3 , (void) 4 , (false) 6 , (void)	Kalkulasi ulang nilai f dan g kemudian masukan ke index	Kalkulasi ulang nilai f dan g kemudian masukan ke index	*Path* Dikerjakan

	7, (void) 8, (true) 9, (true) 10, (true) 11, (false) 14, (true) 15, (false) 1,2,3, (conditional)			
6	1,2,3 , (void) 4 , (false) 6 , (void) 7, (void) 8, (true) 9, (true) 10, (false) 12, (void) 13, (void) 14, (true) 15, (true) 16, (void) 17, (void)	Menghitung nilai Heuristik	Menghitung nilai Heuristik	*Path* Dikerjakan
7	1,2,3 , (void) 4 , (false) 6 , (void) 7, (void)	Memasukan nilai Heuristik ke index	Memasukan nilai Heuristik ke index	*Path* Dikerjakan

	8, (true) 9, (true) 10, (false) 12, (void) 13, (void) 14, (false) 7, (conditional)			
8	1,2,3 , (void) 4 , (false) 6 , (void) 7, (void) 8, (true) 9, (true) 10, (false) 12, (void) 13, (void) 14, (true) 15, (false) 1,2,3, (conditional)	Rute tidak ada	Rute tidak ada	*Path* Dikerjakan
Looping Paths				
Path No.	***Node* (n), Input Condition**	***Excpected Result***	***Test Result***	***Conclusion***
L1	1,2,3, (void) 4, (false) 6, (void)	Node adalah tujuan	Node adalah tujuan	*Path* Dikerjakan

	7, (void) 8, (conditional)			
L2	1,2,3, (void) 4, (false) 6, (void) 7, (void) 8, (true) 9, (conditional)	Current Node = Best Node	Current Node = Best Node	*Path* Dikerjakan

4.6 Pengujian *Beta*

Pada pengujian *beta* ini, pengujian akan dilakukan secara objektif dengan cara dicoba langsung oleh beberapa pengguna. Sample yang diambil adalah 10 orang yang nantinya pengguna akan dipersilahkan untuk mencoba aplikasi dan mengisi kuesioner yang telah disediakan.

Kuesioner ini terdiri atas 10 pertanyaan yang mencakup luang lingkup *game Eiji : The Lost Canvas*. Berdasarkan data hasil kuesioner tersebut, dapat dicari Prosentase (%) masing-masing jawaban dengan menggunakan rumus :

Y = P/Q * 100%

Keterangan :

P = Banyaknya jawaban responden tiap soal

Q = Jumlah responden

Y = Nilai Prosentase (%)

Hasil pengujian *beta* sebagai berikut :

1. Seberapa lama anda bermain *game* dalam sehari ?

No.	Keterangan	Responden	Prosentase (%)
1	1 jam dalam sehari	2	20%
2	2 jam dalam sehari	3	30%
3	Lebih dari 2 jam dalam sehari	5	50%

2. Apakah yang membuat anda tertarik dengan sebuah *game* ?

No.	Keterangan	Responden	Prosentase (%)
1	Alur cerita	1	10%
2	Karakter	1	10%
3	Kualitas grafis	4	40%
4	*Gameplay* yang menarik	4	40%

3. Tema *game* apa yang anda sukai ?

No.	Keterangan	Responden	Prosentase (%)
1	Fantasi	4	40%
2	Sejarah	1	10%
3	Kerajaan	2	20%
4	Masa depan	3	30%

4. Fitur apakah yang anda suka dalam sebuah *game* ?

No.	Keterangan	Responden	Prosentase (%)
1	Komunikasi antar pemain	1	10%
2	Selalu ada penambahan konten	6	60%
3	*Control* yang bisa dirubah	2	20%
4	*Portabe*	1	10%

5. Kendala apa yang sering anda temui dalam bermain *game* ?

No.	Keterangan	Responden	Prosentase (%)

1	*Control* yang sulit	3	30%
2	Musuh yang terlalu tangguh untuk dikalahkan	5	50%
3	Teka-teki yang sulit dipecahkan	2	20%

6. Apakah *Gameplay* dari *game* Eiji : The Lost Canvas ini menarik bagi anda?

No.	Keterangan	Responden	Prosentase (%)
1.	Sangat menarik	4	40%
2.	Menarik	5	50%
3	Cukup menarik	1	10%
4	Kurang menarik	0	0%

7. Apakah *game* ini mudah untuk dimainkan?

No.	Keterangan	Responden	Prosentase (%)
1	Ya	10	100%
2	Tidak	0	0%

8. Apakah visualisasi dalam *game* ini menarik bagi anda ?

No.	Keterangan	Responden	Prosentase (%)
1	Ya	10	100%
2	Tidak	0	0%

9. Apakah fitur-fitur dalam *game* ini sudah mendukung kebutuhan dalam sebuah permainan ?

No.	Keterangan	Responden	Prosentase (%)
1	Sangat mendukung	3	30%
2	Mendukung	3	30%
3	Cukup mendukung	3	30%
4	Kurang mendukung	1	10%

10. Bagaimanakan pendapat anda tentang *game* ini ?

No.	Keterangan	Responden	Prosentase (%)
1	Bagus	7	70%
2	Cukup	2	20%
3	Membosankan	1	10%

4.7 Kesimpulan Pengujian *Beta*

Hasil kesimpulan dari pengujian *beta* membuktikan bahwa seorang manusia membutuhkan waktu lebih dari 2 jam dalam sehari untuk bermain game dengan dengan game yang mempunyai tampilan grafis yang baik dan *gameplay* yang menarik dan tema game yang diharapkan bernuansa fantasi.

Setiap pemain *game* mengharapkan untuk setiap game yang mereka mainkan bisa melakukan *update* konten. Setiap pemain selalu mengalami kesusahan dalam menghadapi musuh yang tangguh, Namun dalam *game Eiji : The Lost Canvas* memiliki *gameplay* yang menarik, mudah untuk dimainkan, tampilan visualisai yang baik. Dan kekurangan dalam *game* ini adalah di bagian fitur-fitur yang ada. Tidak seperti *game thrid person shooter* yang lain, seperti *health bar*, tampilan *GUI* yang menarik, dan tipe musuh yang bervariasi.