BAB 3
ANALISIS DAN PERANCANGAN SISTEM

3.1 Analisis Sistem

Analisis sistem merupakan proses penguraian konsep ke dalam bagian bagian yang lebih sederhana, sehingga struktur logisnya menjadi lebih jelas. Pada analisis sistem ini akan mencakup analisis masalah, analisis kebutuhan non-fungsional dan analisis kebutuhan fungsional. Metode digunakan untuk menguji, membandingkan, menilai, serta memahami sistem pemikiran yang lebih kompleks dan memecahkan permasalahan dan berinteraksi untuk mencapai tujuan yang dicapai.

3.1.1 Analisis Masalah

Perjuangan sejarah Medan Area adalah salah satu dari sekian banyaknya upaya bangsa Indonesia dalam mempertahankan kemerdekaan. Peperangan ini bergejolak pada tanggal 1 Desember, antara pasukan Komando Rasismen Laskar Rakyat Medan Area dengan pasukan NICA dan sekutu Belanda, yang dipicu oleh kemarahan para pemuda atas tingkah laku semena – mena pasukan sekutu Belanda.

Menurut bapak Ichwan Azhari sebagai kepala pusat studi Sejarah dan Ilmu sosial Universitas Negeri Medan, berpendapat bahwa sejarah perjuangan Medan Area sudah terlupakan mulai dari peristiwa, jejak perang, dan tokoh yang terlibat didalamnya juga semakin hilang dari ingatan kolektif masyarakat, sehingga menimbulkan banyak versi mengenai jumlah korban yang terlibat di dalamnya, dan salah satu versi mengatakan bahwa ratusan pejuang Indonesia gugur dan puluhan pasukan Belanda dan Sekutunya tewas, namun dibandingkan dengan perang perjuangan lainnya yang lebih sering dibahas di dalam buku pelajaran seperti 10 November, Bandung lautan api membuat sejarah peristiwa Medan Area tidak begitu di ingat dan dikenal[2].

Media interaktif merupakan sarana pendidikan yang semakin berkembang di era modern sekarang, selah satunya adalah media game dimana banyak jenis-

jenis game yang bersifat edukasi, selain konten yang di berikan di dalam game berupa materi-materi pembelajaran, dukungan grafis juga sangat berpengaruh. Sejarah Medan Area sendiri merupakan salah satu pelajaran di sub bab buku pelajaran SMA kelas XII yang cocok di jadikan sebuah game yang bergenre *first person shooter* (FPS) karena di latar belakangi pertempuran mempertankan pasca kemerdekaan.

Game FPS sendiri merupakan salah satu *genre game* yang menggunakan sudut orang pertama di dalam permainannya, di mana permainan ini lebih cocok dengan desain grafis 3D sehingga dapat memberikan nuansa yang lebih nyata dalam *game*. Namun agar *game* terlihat lebih sempurna maka di dalam *Non Playable Character* (NPC) di terapkan *Artificial Intelligence* yang berupa algoritma A**(A Star)* untuk pencarian rute terpendek antara musuh ke *player* dan *Fuzzy Logic* sebagai pengambilan keputusan NPC.

Berdasarkan latar belakang tersebut, maka dibuatlah sebuah *game first person shooter* yang dilatar belakangi dari cerita perjuangan rakyat Medan yaitu Pembangunan *Game FPS* Sejarah Medan Area dengan harapan *game* menjadi media pembelajaran yang mampu memperkenalkan salah satu perjuangan bangsa Indonesia untuk mempertahankan kemerdekaan.

3.1.2 Analisis *Game* Sejenis

Analisis *game* sejenis merupakan analisis yang akan membahas mengenai *game* yang menjadi acuan dalam pembangunan sebuah *game*. Dalam *game* sejenis akan dibahas mengenai berbagai hal yang ada di dalam *game* sejenis diantaranya meliputi *storyline*, *gameplay*, serta komponen-komponen apa saja yang ada dalam *game* tersebut sehingga dapat menjadi acuan bagi kebutuhan yang akan dibangun. Berikut sedikit ulasan mengenai *game* yang akan dibahas yaitu *game* P10NER (Pertempuran 10 November).



Gambar 3.1 Tampilan Menu Utama Game P10NER

3.1.2.1 Pengenalan Game P10NER

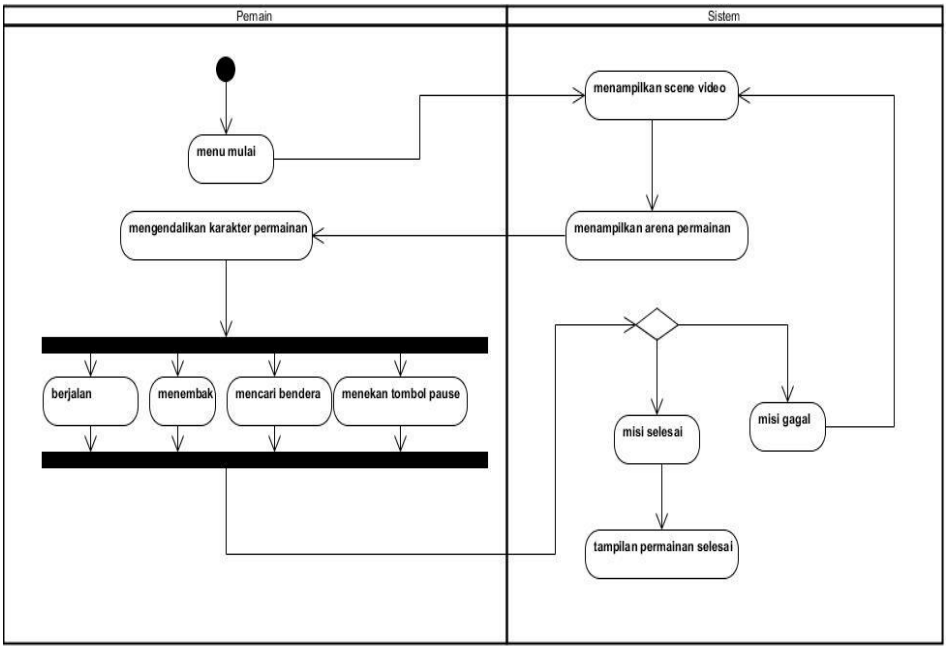
Game P10NER, Pertempuran 10 November merupakan *game* edukasi yang menceritakan salah satu pertempuran dalam mempertahankan kemerdekaan, yang terjadi di Surabaya pada tanggal 10 November 1945.

1. ***Storyline***

   Dalam *game* ini *player* berperan sebagai Wawan yaitu pejuang yang berusaha untuk mengalahkan serangan tentara Belanda yang dipersenjatai peralatan yang canggih, dan *player* diharuskan melewati 4 level seperti merebut Hotel Yamato dari tangan sekutu, perobekan bendera merah putih biru, pertempuran di jembatan merah dan yang terakhir misi pertempuran 10 November 1945 yang menewaskan Brigjen AWS Mallaby.

2. ***Gameplay***

   Pada *game* P10NER 10 November 1945, pemain harus mencari dan mengambil beberapa kunci yang disediakan oleh sistem disetiap *level* yang dimainkan, untuk digunakan pemain ketika akan membuka pintu yang membutuhkan kunci untuk membukanya. Adapun alur dari game P10NER ini sendiri dapat di gambarkan seperti activity diagram di bawah ini



Gambar 3.2 Alur permainan *game* P10NER

3. Skenario *Game*

Skenario dalam permainan *game* P10NER dapat dilihat dari alur permaiannya sebagai berikut :

a. Pemain berada di dalam Hotel Yamato yang dipenuhi oleh musuh.
b. Pemain harus mencari kunci apabila ingin membuka sebuah pintu atau jalan utama ketika berada di dalam permainan.
c. Pemain harus menghindari dan membunuh musuh ketika berada di arena permainan.
d. Jika pemain terkena tembakan oleh musuh, maka darah / *health point* akan berkurang dan apabila habis maka permainan diulangi ke awal permainan.

4. Komponen pada game P10NER

Dari beberapa referensi situs internet yang sudah di telusuri dan telah melakukan pengujian langsung terhadap game yang bersangkutan maka di dapatkan komponen – komponen yang di gunakan dalam game P10NER tersebut yaitu dapat di lihat pada tabel 3.1 dibawah ini :

Tabel 3.1 Komponen game P10NER

No	Komponen	Keterangan
1	Tokoh Utama	Game ini memiliki satu tokoh utama yaitu wawan.
2	Senjata	Mempunyai Pistol, dan senapan
4	Grafis	Game ini memiliki grafis 3D
5	Segi Edukasi	Terdapat di scene video.

3.1.3 Analisis *Game* Yang Akan Di Bangun

Game Sejarah Medan Area yang akan dibangun yaitu *game* yang mempunyai grafis 3 dimensi yang ber-*genre first person shooter (FPS). Game* ini mempunyai alur cerita yang sama seperti sejarah perjuangan Medan Area, dimana dalam penamaan tokoh dan konsep tempat pertempuran semua di buat hampir mirip.

Game dibangun dengan konsep sudut pandang orang pertama atau *first person shooter* dengan tujuan agar kesan pertempuran yang dihadirkan lewat *game* lebih terlihat nyata.

Beberapa fitur yang ditawarkan di dalam *game* ini :

1. *Single player*
2. *Game* diangkat dari cerita salah satu sejarah pertempuran di Indonesia, yaitu sejarah Medan Area
3. *Game* dibuat dengan grafis 3D.
4. *Game* dibuat dengan *genre first person shooter* dengan konsep sudut pandang orang pertama.
5. Musuh di *stage* 1 menggunakan Algoritma A*, dan di *stage* 2 menggunakan *fuzzy logic*.

3.1.3.1 S*toryline*

Game ini akan menceritakan kronologis sejarah perjuangan Medan Area di kota Medan, insiden berawal dari tingkah semena-mena pasukan NICA yang menginap di hotel jalan Bali terhadap seorang pemuda, yang akhirnya menimbulkan kemarahan rakyat Medan, dimana Achmad Tahir (Naga Bonar)

sebagai karakter pemain bertugas untuk menyerang pasukan NICA dan Sekutu yang berada di hotel jalan Bali dan gudang penyimpanan senjata pasukan sekutu yang berada di dekat hotel di jalan Bali.

Setelah insiden di depan hotel jalan Bali terjadi, ternyata pasukan Sekutu menyerang kota Medan dan membuat tulisan – tulisan *"FIXED BOUNDERIUS MEDAN AREA"* di berbagai sudut kota Medan, yang pada akhirnya membuat Acmad Tahir (Naga Bonar) harus mencari lokasi dan menghancurkan tulisan – tulisan tersebut.

3.1.3.2 *Gameplay*

Di dalam *game* ini terdapat 2 *stage* yang saling berhubungan, dimana *stage* 1 harus selesai sebelum ke *stage* 2, adapun isi pembagian *stage* dapat di lihat seperti di bawah ini :

1. *Stage* 1
   - Pemain berjalan mencari jalan Bali.
   - Pemain harus mengambil petunjuk untuk menemukan jalan Bali.
   - Pemain menembak musuh sebanyak 30 orang pasukan Belanda di *stage* 1 yang berada di depan Hotel De Boer, Hotel Astori, dan Grand Hotel yang terdapat di jalan Bali.
   - Pemain mencari *icon* obat untuk menambah darah.
   - *Icon* darah bisa diambil jika darah pemain berkurang.
   - Pemain dapat menggunakan 3 jenis senjata yang digunakan secara bergantian, yaitu Pistol, Senapan, dan Granat, dengan menggunakan *scroll* pada *mouse*.
   - Senjata granat dapat diambil dengan mendekat ke *icon* senjata dan menekan tombol "E" pada *keyboard*.
   - Senjata Pistol hanya dapat mengeluarkan peluru sebanyak 12 kali sebelum di *reload* dengan menekan tombol "R" pada *keybord*.
   - Senjata Senapan mempunyai 50 peluru, hanya bisa di *reload* dengan menekan tombol "R" pada *keybord*.

- Senjata Granat mempunyai 10 granat yang dapat di lempar, dan jika granat habis bisa di ambil dengan dengan mendekat ke icon granat dan menekan tombol "R" pada *keyboard*.

2. *Stage* 2
   - Pemain mencari *icon* petunjuk yang berisi nama tempat untuk mengetahui lokasi tempat gardu yang bertulisan *"FIXED BOUNDERIUS MEDAN AREA"* yang terdapat di sudut-sudut kota Medan.
   - Pemain berjalan menuju tulisan – tulisan "*FIXED BOUNDERIUS MEDAN AREA*" yang terdapat di Medan Timur yaitu Kampung Sukarame, dan Desa Sungai Kerah, Medan Barat yaitu Padang Bulan, Petisah, Jalan Pringgan dan di Medan Selatan yaitu Kota matsum.
   - Pemain harus menghancurkan tulisan *"FIXED BOUNDERIUS MEDAN AREA"* sebanyak 6 gardu untuk menyelesaikan misi.
   - Pemain dapat menggunakan 3 jenis senjata yang digunakan secara bergantian, yaitu Pistol, Senapan, dan Granat dengan cara menggunakan *scroll* pada *mouse*.
   - Senjata Pistol hanya dapat mengeluarkan peluru sebanyak 12 kali sebelum di *reload* dengan menekan tombol "R" pada *keybord*.
   - Senjata Senapan mempunyai 50 peluru, hanya bisa di *reload* dengan menekan tombol "R" pada *keybord*.
   - Senjata Granat mempunyai 10 granat yang dapat di lempar, dan jika granat habis bisa di ambil dengan mengambil icon granat.

3.2 Analisis Perancangan Multimedia

Di dalam game perancangan multimedia merupakan salah satu bagian penting yang harus di perhatikan, dimana dengan adanya multimedia, game akan terlihat lebih menarik dan hidup. Adapun unsur yang perlu di perhatikan dalam pembangunan game FPS 3D Sejarah Medan Area adalah :

1. Animasi

   Dalama pembuatan animasi FPS 3D Sejarah Medan Area ini menggunakan *3D Max* dan *GooglesketchUP* untuk menghasilan bentuk 3D.

2. Audio/Sound

   Dalam pembuatan *audio* atau *sound game* 3D Sejarah Medan Area ini menggunakan *tool* ogg *convert* untuk mengubah format avi, mp3, wap ke format ogg, ogg sendiri merupakan format yang bisa di baca dalam unity pro.

3. Teks

   Dalalm pembuatan *teks* dalam *game* tidak menggunakan *software* pendukung hanya, cukup menggunakan *unity*.

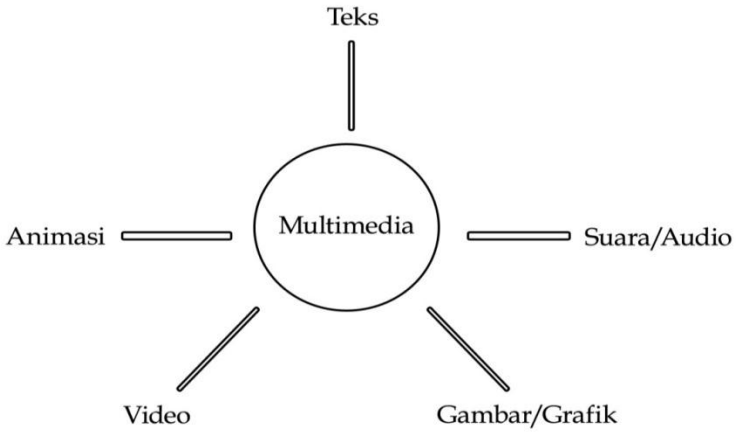
4. Gambar/grafis

   Dalam pembuatan gambar, disini menggunakan *photoshop* untuk menghasilkan bentuk gambar 2D.

5. Video

   Dalam pembuatan video, *tool* yang di gunakan adalah *movie maker* untuk membuat video, yang kemudian di *convert* menjadi format .ogg dengan menggunakan OGG *Convertor*.

Multimedia sendiri merupakan kombinasi dari unsur – unsur yang di gabungkan satu sama lain, dapat di lihat pada gambar 3.4 seperti di bawah ini



Gambar 3.3Unsur – unsur dalam multimedia

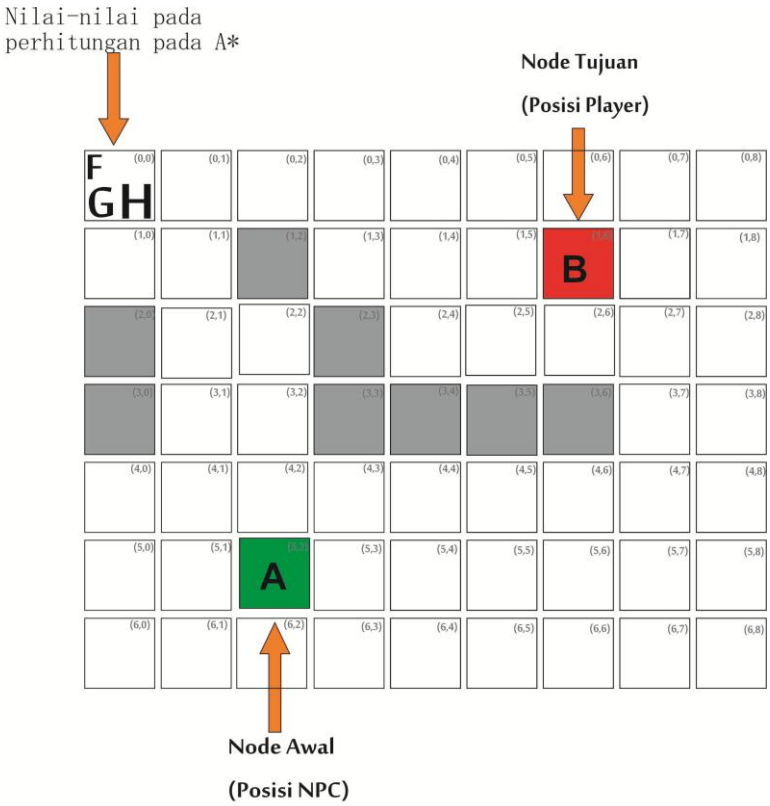
Penggabungan unsur – unsur itu sendiri nantinya akan membentuk satu kesatuan yang dapat menghasilkan visual yang menarik di dalam *game* FPS 3D sejarah Medan Area.

3.3 ANALISIS AI

Artificial Intelligence atau yang bahasa Indonesia merupakan kecerdasan buatan Kecerdasan diciptakan dan dimasukkan ke dalam suatu mesin (komputer) agar dapat melakukan pekerjaan seperti yang dapat dilakukan manusia. Adapun AI yang digunakan dalam pembuatan *game* FPS Sejarah Medan Area adalah algoritma A* dan *Fuzzy logic*, yang akan di jelaskan dibawah ini.

3.3.1 Analisis Algoritma A*(A Star)

Algoritma A* di dalam *game* akan diterapkan pada NPC (*Non Playable Character*) sebagai penentuan jalur terpendek atau *pathfinding* menuju *target.* Cara kerja algoritma A* di dalam *game first person shooter* 3D sejarah Medan Area lebih jelasnya sebagai berikut.



Gambar 3.4 Penjelasan A*

Pada gambar 3.5 terdapat beberapa kotak yang memiliki koordinat dari (0,0) sampai (6,8). Kotak atau *node* yang masing-masing telah memiliki koordinat merupakan *node* yang digunakan untuk melakukan pencarian jalur terpendek / *pathfinding*. Terdapat beberapa *node* yang memiliki warna berbeda, berikut penjelasan dari masing -masing *node* tersebut :

a. *Node* warna hijau : *Node* awal dari posisi NPC (*Non Playbel Character*) yang berada pada *node* di koordinat (5,2)
b. *Node* warna abu : *Node* penghalang (*collider*) yang berada pada *node* di koordinat (1,2), (2,0), (2,3), (3,0), (3,3), (3,4), (3,5) dan (3,6).
c. *Node* warna merah : *Node* dari posisi pemain atau *player* yang berada pada *node* di koordinat (1,6).

Untuk nilai-nilai yang digunakan untuk perhitungan A *star*, terdapat 3 nilai yaitu :

a. Nilai F, nilai yang diperoleh dari penjumlahan nilai G, jumlah nilai tiap simpul dalam jalur terpendek dari *starting point* ke *current node*, dan H, jumlah nilai perkiraan dari sebuah simpul ke target *point*.

b. Nilai G, nilai atau biaya yang dikeluarkan dari *node* awal / *node* A ke *node* yang ada di sekitarnya, dimana nilai G digunakan nilai 10 untuk gerakan *vertical* atau *horizontal*, angka 10 digunakan hanya untuk memudahkan perhitungan. Sedangkan untuk gerakan *diagonal* maka akan dikalikan dengan √2 yang berarti 10x√2 = 14,14. Untuk memudahkan perhitungan angka 14,14 dibulatkan menjadi 14.
c. Nilai H, nilai *heuristic* yang digunakan untuk menghitung perkiraan biaya yang dikeluarkan dari *node* awal / *node* A menuju *node* tujuan / *node* B.

Perhitungan nilai F didapat dari persamaan F= G+H. Untuk perhitungan nilai H digunakan fungsi *heuristic*, metode yang digunakan di dalam contoh ini adalah metode *Manhattan* dimana perhitungan jumlah *node* hanya yang bergerak secara *vertical* dan *horizontal* menuju tujuannya serta mengabaikan penghalang atau *collider,* yang kemudian nilainya dikalikan dengan 10. Atau dirumuskan dengan:

H = 10*(abs(*current*X-*target*X) + abs(*current*Y-*target*Y))

Pada pencarian jalur terpendek oleh *Astar* ada beberapa istilah yang digunakan seperti *open list*, *closed list*, *parrent*, *current node* dan *pointer*.

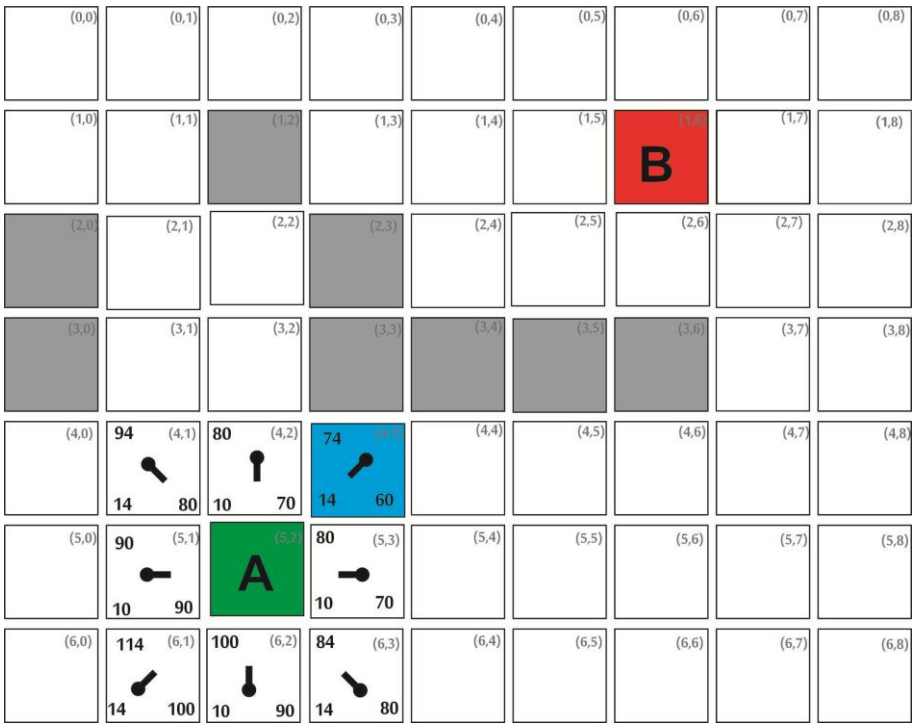
a. *Open list* adalah *list* untuk memeriksa *node-node*.
b. *Closed list* adalah *node-node* yang diabaikan atau *node-node* yang tidak perlu diperiksa.
c. *Parrent* adalah digunakan untuk menelusuri jalan.
d. *Current Node* adalah simpul yang sedang dijalankan dalam algoritma pencarian jalan terpendek.
e. *Pointer* adalah penunjuk jalan terhadap *Node* A *(back pointer*) ketika *node* B (target) telah ditemukan.

Langkah berikutnya ketika telah mendapatkan *node* awal adalah menentukan rute terpendek dengan cara memeriksa setiap *node* yang berdekatan dengan *node* A sampai *node* B. Di bawah ini merupakan penjelasan langkah-langkahnya.

1. Langkah pertama adalah memasukan *node* A kedalam *open list*.

2. Langkah kedua adalah memeriksa *node-node* yang berdekatan dengan *node* A, abaikan *node* dengan *collision* atau penghalang lalu tambahkan semua *node* tersebut kedalam *open list*. *Node* A sendiri nantinya akan menjadi "*parent*" dan digunakan untuk menelusuri jalan.
3. Langkah ketiga adalah menghapus *node* A dari *open list*, dan dimasukan ke dalam *closed list*.

Berikut ini perhitungan pada langkah pertama yang berada pada *node* dengan koordinat (4,3).



Gambar 3.5 Penjelasan A* pada langkah pertama

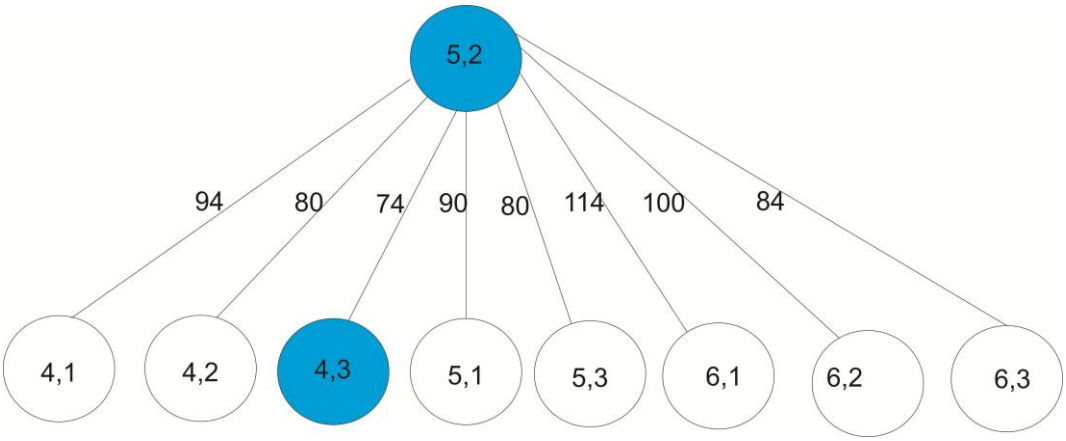
Pada langkah pertama, *node* (5,2) dimasukan ke dalam *open list* dan selanjutnya memeriksa *neighbor node* atau *node* yang berdekatan dengan *node* (5,2) yaitu, *node* (4,1), (4,2), (4,3), (5,1), (5,3), (6,1), (6,2), (6,3). Setelah diperiksa *node-node* tersebut dimasukan ke dalam *open list*. Hapus *node* (5,2) dari *open list* dan masukan *node* tersebut kedalam *closed list.* Lalu dari *node-node* yang berada di dalam *open list* pilih *node* dengan nilai F yang paling rendah, yaitu *node* (4,3) dengan nilai F = 74.

Langkah pertama menghasilkan *node-node* di dalam *open list* dan *closed list* dijelaskan dengan tabel sebagai berikut.

Tabel 3.2 Tabel Perhitungan langkah pertama

Open List	(4,1), (4,2), (4,3), (5,1), (5,3), (6,1), (6,2), (6,3)
Closed list	(5,2)
Parent	(5,2)
Current node	(4,3)

Gambar 3.6 merupakan penjelasan langkah pertama pencarian jalur terpendek/ *pathfinding* menggunakan *tree*. *Node* (4,3) merupakan *current node* yang dihasilkan pada perhitungan langkah pertama.



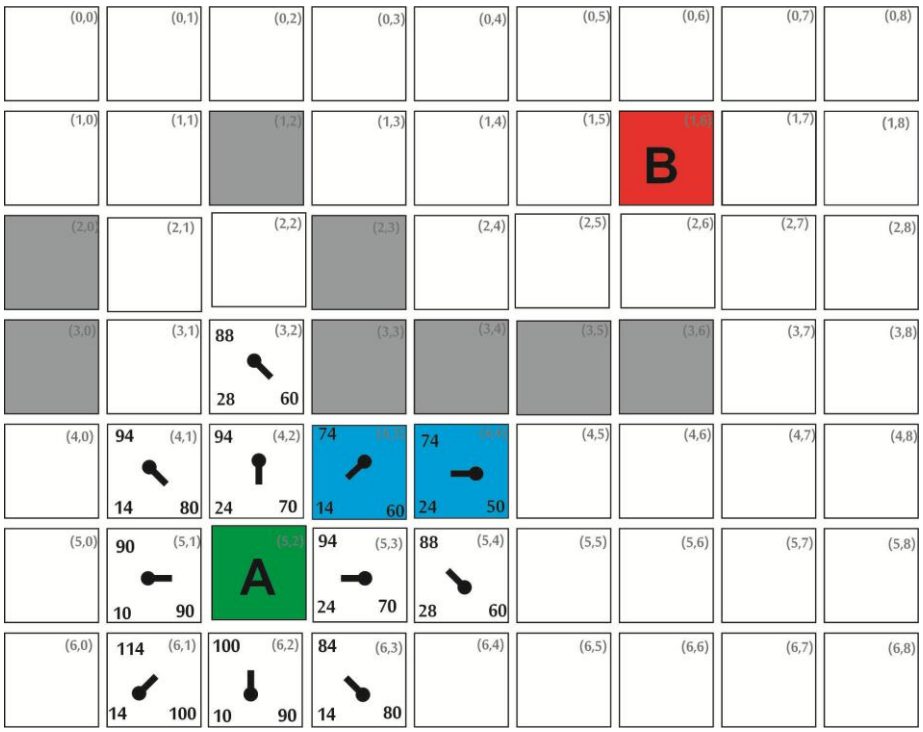
Gambar 3.6 Penjelasan A* dengan *tree* pada langkah pertama

Proses yang dijelaskan di atas, dilakukan berulang kali atau *looping*, dimulai dari *open list* dan menemukan *node* dengan nilai F paling rendah. Langkahnya adalah sebagai berikut :

1. Hapus *current node* tersebut dari *open list* dan masukkan ke dalam *closed list.*
2. Periksa semua *node* yang berdekatan dengan *current node*, abaikan *node* yang *unwalkable* atau dalam contoh ini adalah *node* penghalang. Jika tidak ada pada *open list,* tambahkan ke dalam *open list.* Dan tambahkan "*parent*" untuk *node* tersebut.

3. Jika *node* sudah ada pada *open list,* periksa apakah ini jalan yang lebih baik dari *node* awal, dengan kata lain periksa apakah nilai G-nya lebih rendah apabila menggunakan *node* ini, jika lebih rendah maka hitung ulang nilai F dan G serta merubah arah *pointer,* jika tidak maka jangan lakukan apa-apa.

Berikut ini perhitungan pada langkah kedua yang diambil adalah *node* dengan koordinat (4,3).



Gambar 3.7 Penjelasan A* pada langkah kedua

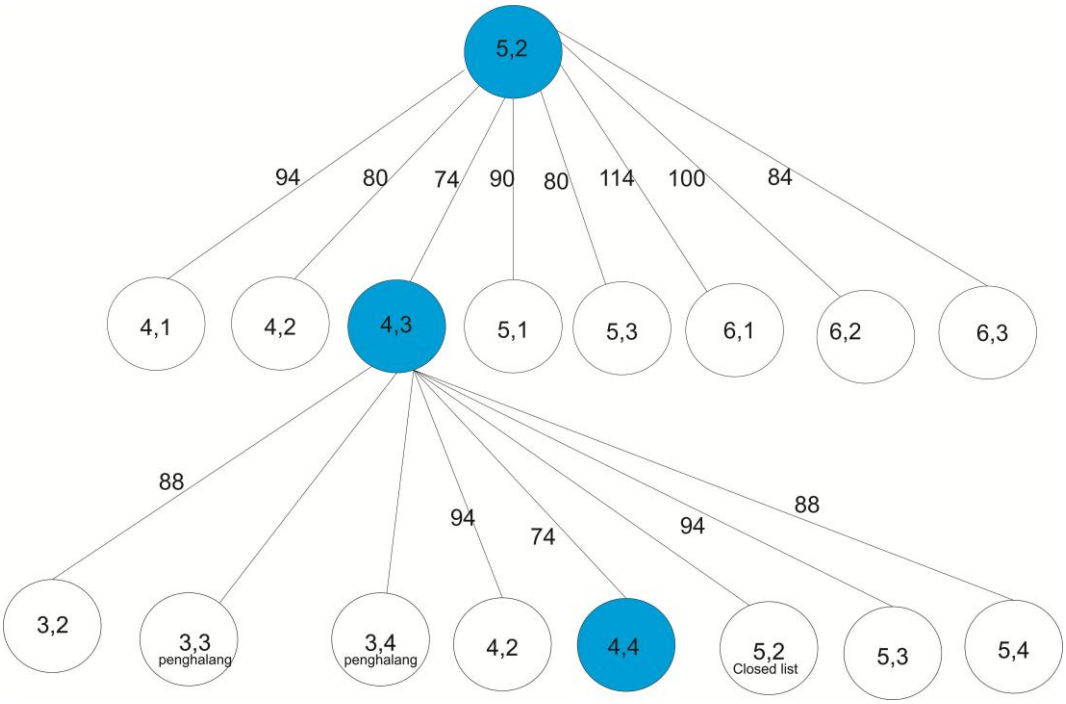
Pada gamber 3.7 terlihat ada 4 *node* yang masuk ke dalam *open list* yaitu *node* (4,5), (5,3), (5,4) dan (5,5). Sedangkan untuk *node* (4,3) merupakan *node* c*losed list* dan *node* (3,3), (3,4), (3,5) merupakan penghalang atau *collider*. Dari keempat *node* yang masuk ke dalam *open list* terdapat 1 *node* yang mempunyai nilai F yang terkecil yaitu *node* (4,4).

Berikut ini penjelasan melalui tabel untuk pencarian jalan terpendek pada langkah kedua.

Tabel 3.3 Tabel Perhitungan langkah kedua

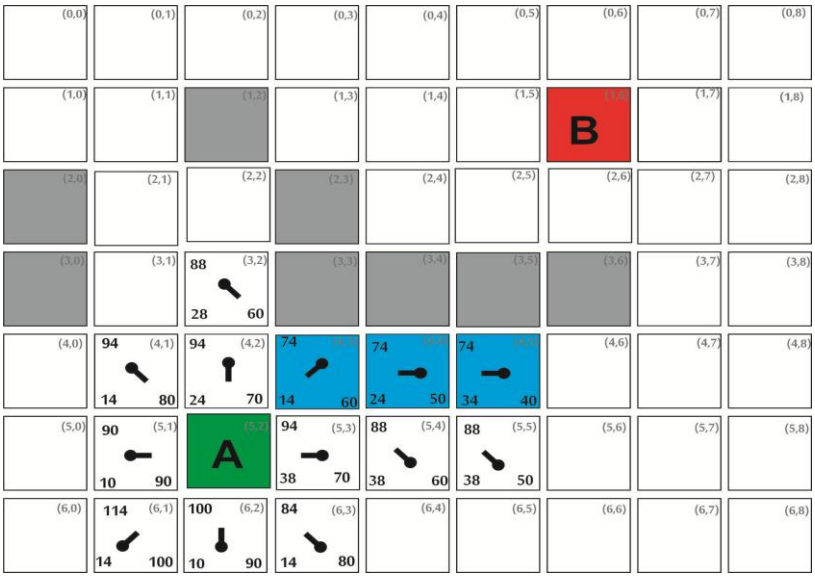
Open List	(4,4), (4,5), (5,3), (5,4), (5,5)
Closed list	(5,2), (4,3)
Parent	(4,3)
Current node	(4,4)

Gambar 3.8 merupakan penjelasan pencarian jalur terpendek / *pathfinding* menggunakan *tree*. *Node* (4,4) merupakan *current node* yang dihasilkan pada perhitungan langkah kedua.



Gambar 3.8 Penjelasan A* dengan *tree* pada langkah kedua

Gambar 3.9 menunjukan perhitungan pada langkah ketiga yang diambil adalah *node* dengan koordinat (4,5).



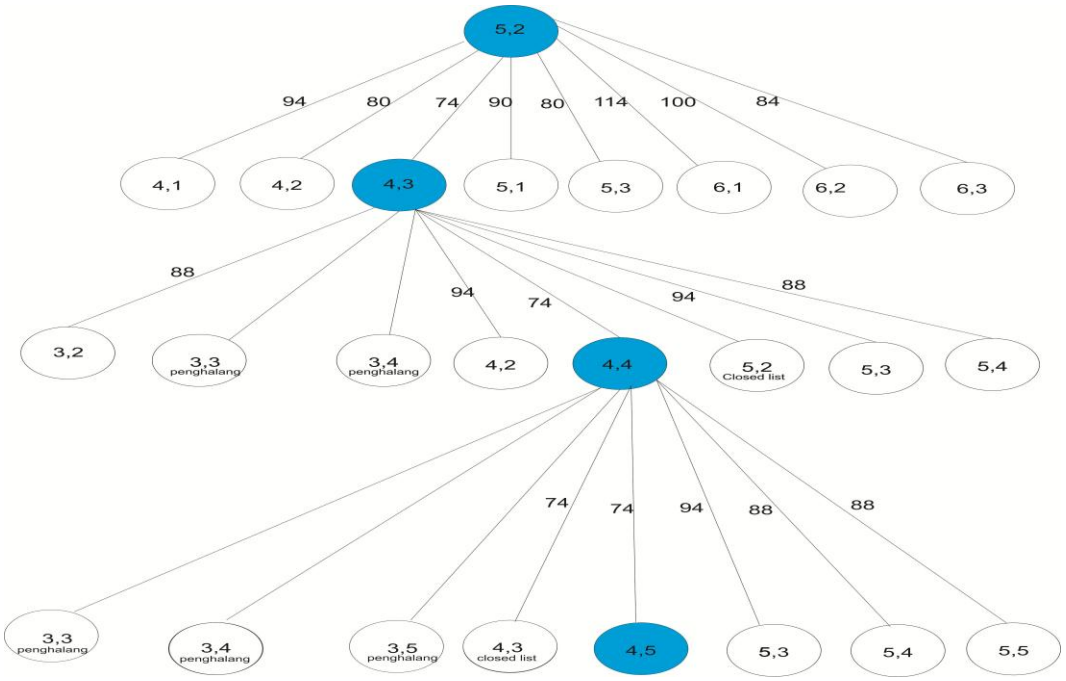
Gambar 3.9 Penjelasan A* pada langkah ketiga

Berikut ini penjelasan melalui tabel 3.4 untuk pencarian jalan terpendek pada langkah ketiga.

Tabel 3.4 Tabel Perhitungan langkah ketiga

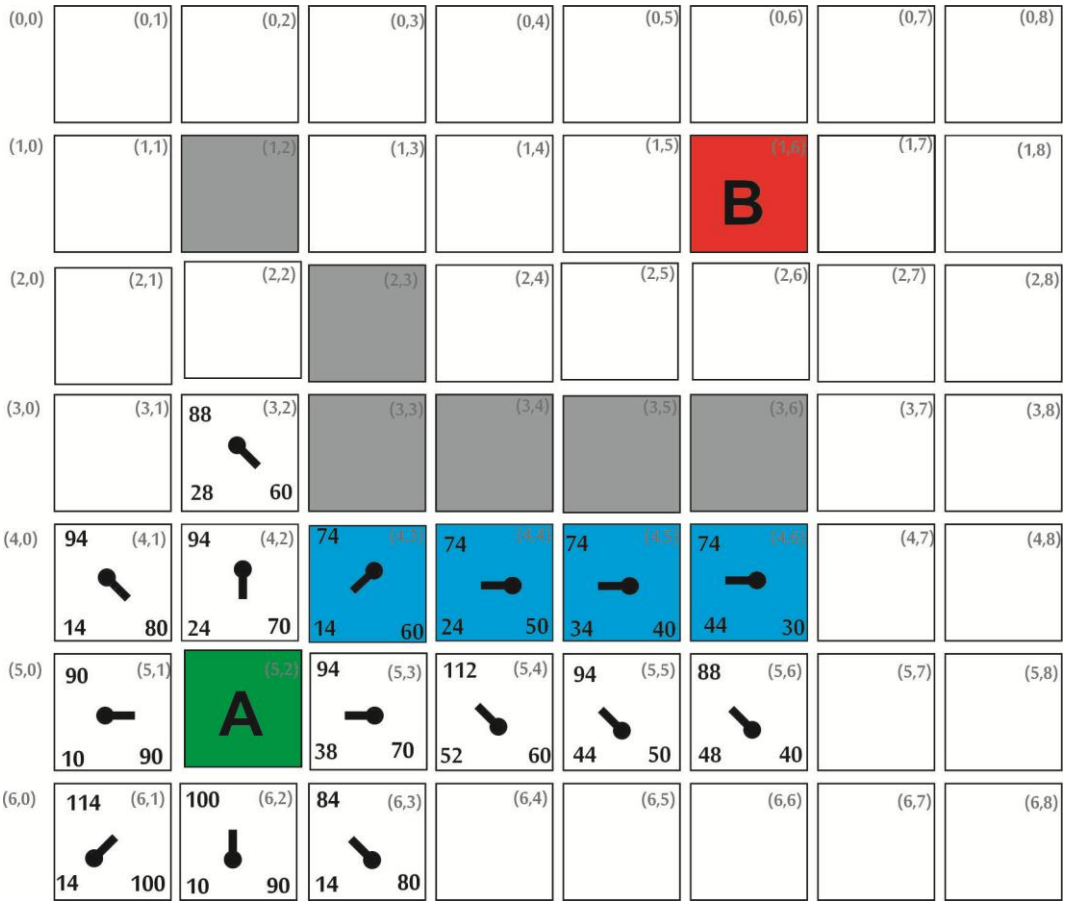
Open List	(4,5), (5,3), (5,4), (5,5)
Closed list	(5,2), (4,3), (4,4)
Parent	(4,4)
Current node	(4,5)

Gambat 3.9 merupakan penjelasan pencarian jalur terpendek / *pathfinding* menggunakan *tree*. *Node* (4,5) merupakan *current node* yang dihasilkan pada perhitungan langkah ketiga.



Gambar 3.10 Penjelasan A* dengan *tree* pada langkah ketiga

Gambar 3.11 menunjukan perhitungan pada langkah keempat yang berada pada *node* dengan koordinat (4,6).



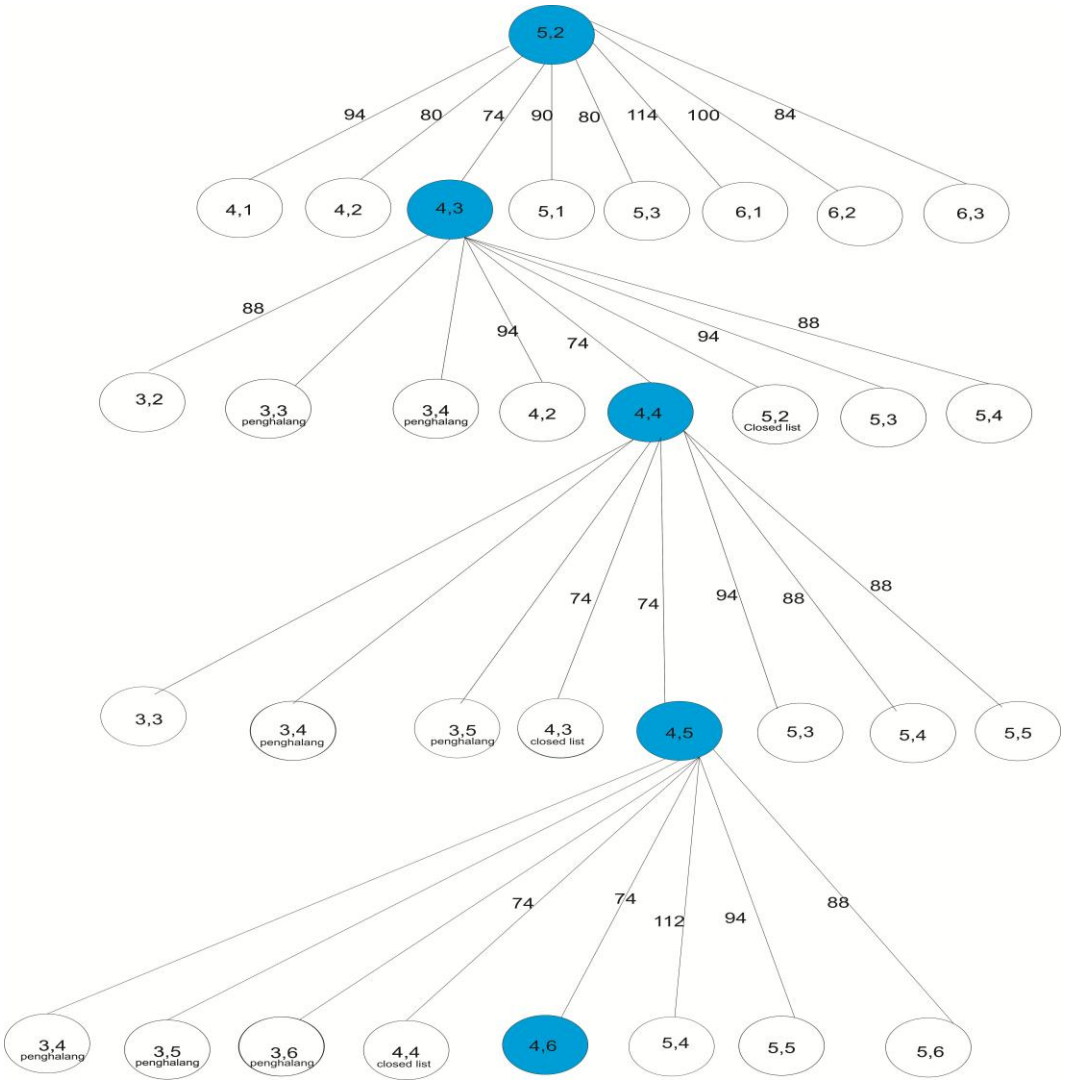
Gambar 3.11 Penjelasan A* pada langkah keempat

Berikut ini penjelasan melalui tabel untuk pencarian jalan terpendek pada langkah keempat.

Tabel 3.5 Tabel Perhitungan langkah keempat

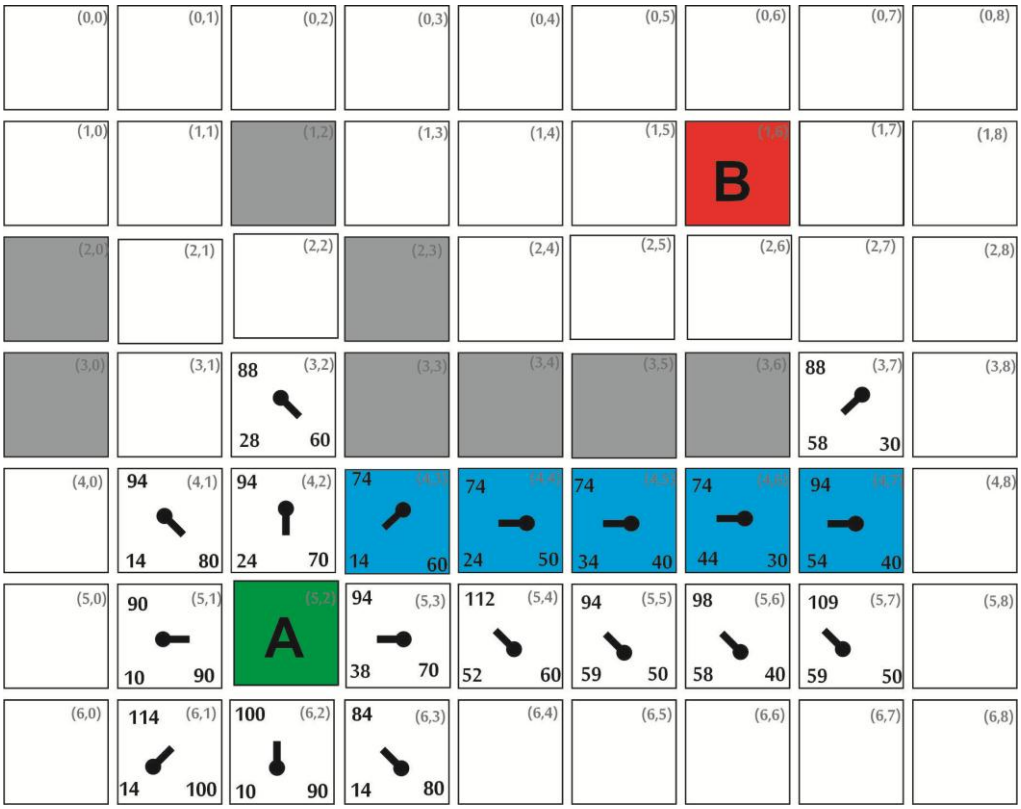
Open List	(4,6), (5,4), (5,5), (5,6)
Closed list	(5,2), (4,3), (4,4), (4,5)
Parent	(4,5)
Current node	(4,6)

Gambat 3.12 merupakan penjelasan pencarian jalur terpendek atau *pathfinding* menggunakan *tree*. *Node* (4,6) merupakan *current node* yang dihasilkan pada perhitungan langkah keempat.



Gambar 3.12 Penjelasan A* dengan *tree* pada langkah keempat

Gambar 3.13 menunjukan perhitungan pada langkah kelima yang berada pada *node* dengan mengambil koordinat (4,7) karena nilai F nya lebih kecil dibandingkan *node* (5,5), (5,6), (5,7), sedangkan terhadap *node* (3,7) karakter tidak bisa bergerak secara diagonal menembus dinding atau colison sehingga *node* (4,7) yang di ambil.



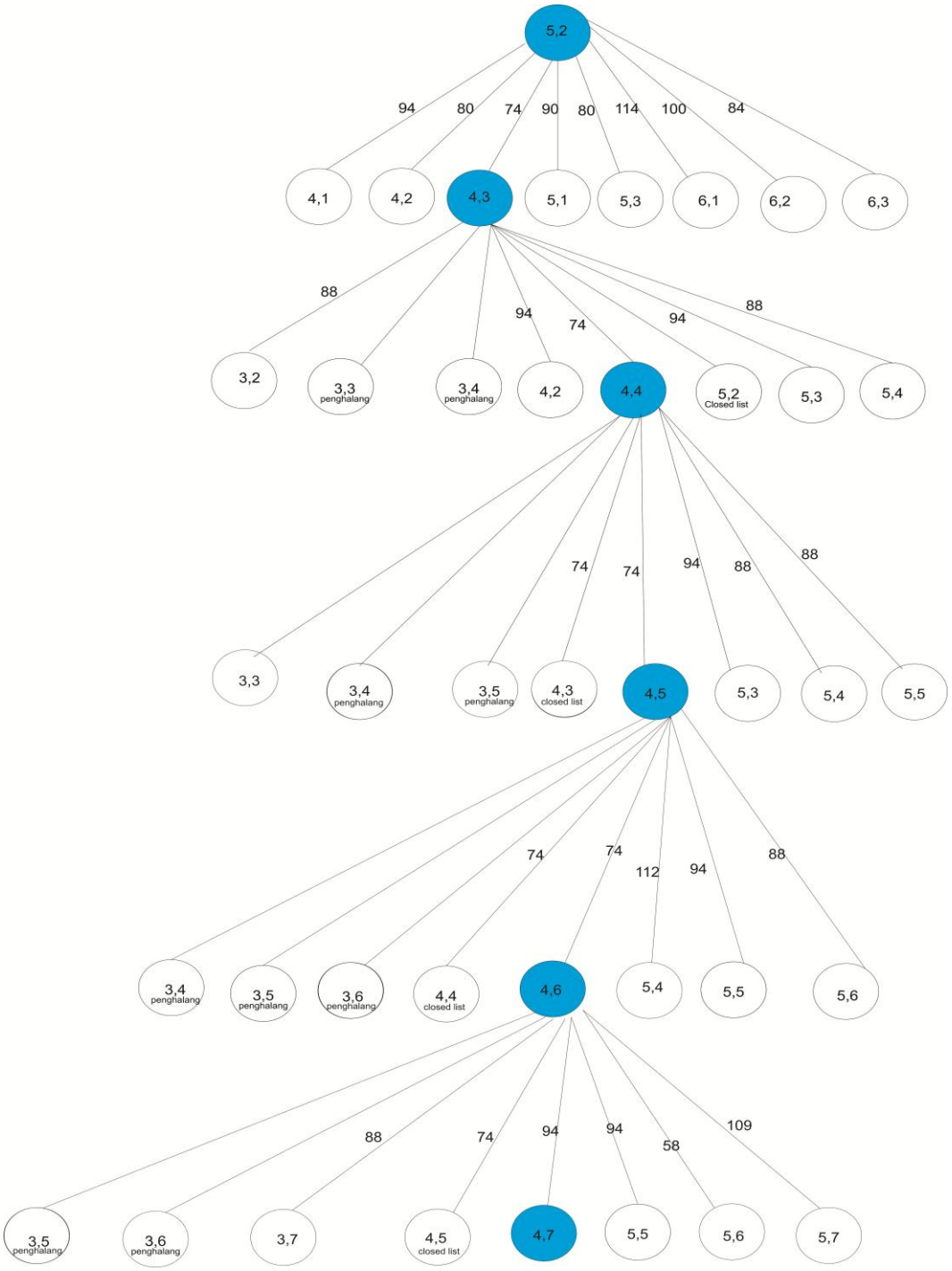
Gambar 3.13 Penjelasan A* pada langkah kelima

Berikut ini penjelasan melalui tabel untuk pencarian jalan terpendek pada langkah kelima.

Tabel 3.6 Tabel Perhitungan langkah kelima

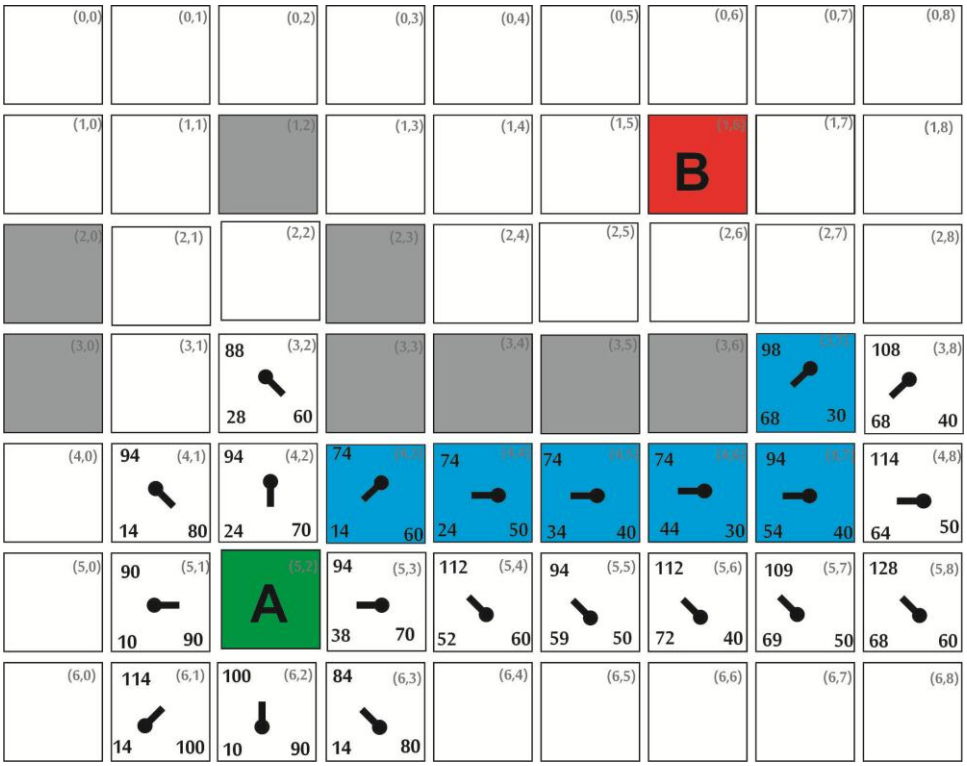
Open List	(4,7), (5,5), (5,6), (5,7)
Closed list	(5,2), (4,3), (4,4), (4,5), (4,6)
Parent	(4,6)
Current node	(4,7)

Gambar 3.14 merupakan penjelasan pencarian jalur terpendek / *pathfinding* menggunakan *tree*. *Node* (4,7) merupakan *current node* yang dihasilkan pada perhitungan langkah kelima.



Gambar 3.14 Penjelasan A* dengan *tree* pada langkah kelima

Gambar 3.15 menunjukan perhitungan pada langkah kelima yang berada pada *node* dengan koordinat (3,7).



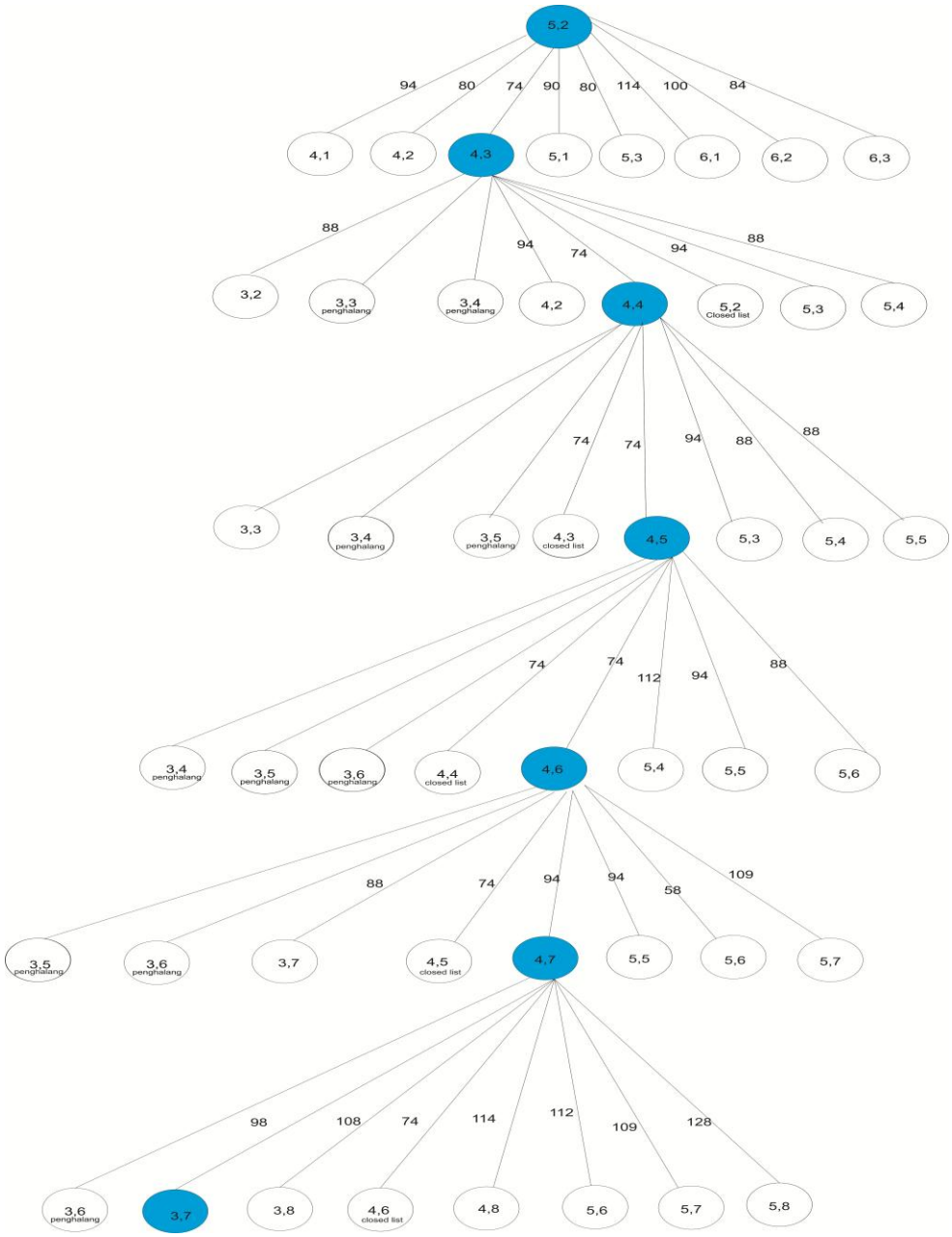
Gambar 3.15 Penjelasan A* pada langkah keenam

Berikut ini penjelasan melalui tabel untuk pencarian jalan terpendek pada langkah keenam.

Tabel 3.7 Tabel Perhitungan langkah keenam

Open List	(3,7), (3,8), (4,8), (5,6), (5,7), (5,8)
Closed list	(5,2), (4,3), (4,4), (4,5), (4,6), (4,7)
Parent	(4,7)
Current node	(3,7)

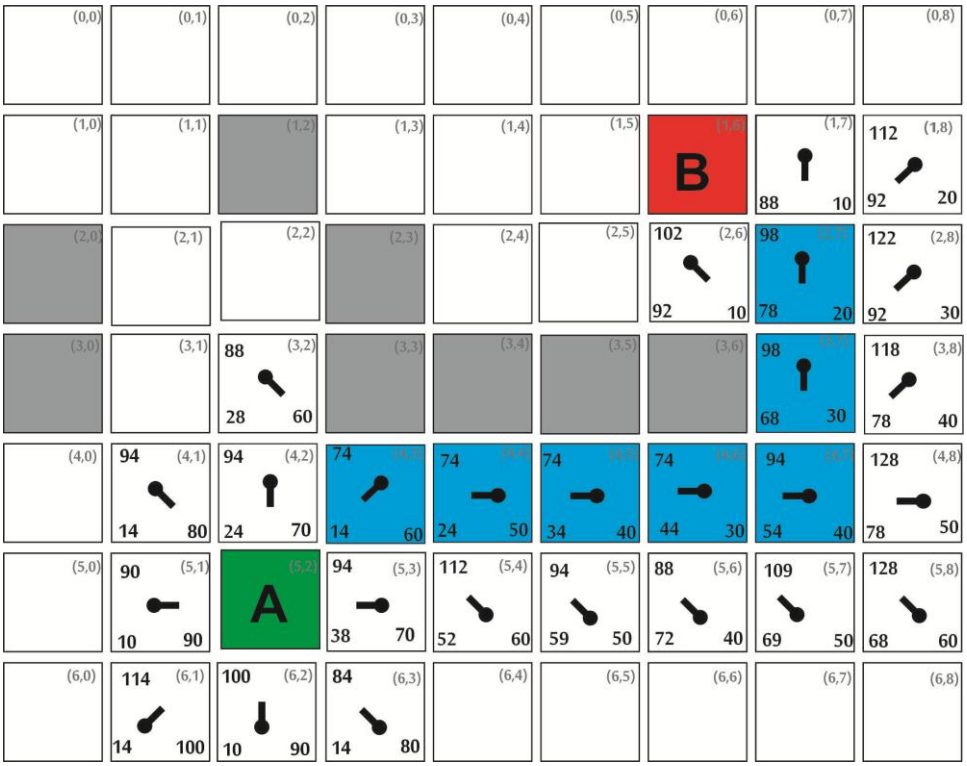
Gambar 3.16 merupakan penjelasan pencarian jalur terpendek / *pathfinding* menggunakan *tree*. *Node* (3,7) merupakan *current node* yang dihasilkan pada perhitungan langkah ke enam.



Gambar 3.16 Penjelasan A* dengan *tree* pada langkah keenam

Penentuan jalan terbaik dengan melakukan pemeriksaan dimulai dari *target* atau *node* B, dalam ilustrasi adalah kotak berwarna biru. Dengan mengikuti arah *pointer* yang berarti akan membawanya kembali ke *node* awal, maka akan didapatkan jalan terbaiknya, Maka jika diimplementasikan pada *game,* nantinya musuh / NPC akan melewati halangan dan akhirnya akan menemukan rute terpendek ke tujuan, lebih jelasnya seperti gambar berikut.

Berikut ini perhitungan pada langkah terakhir yang berada pada *node* dengan koordinat (2,7).

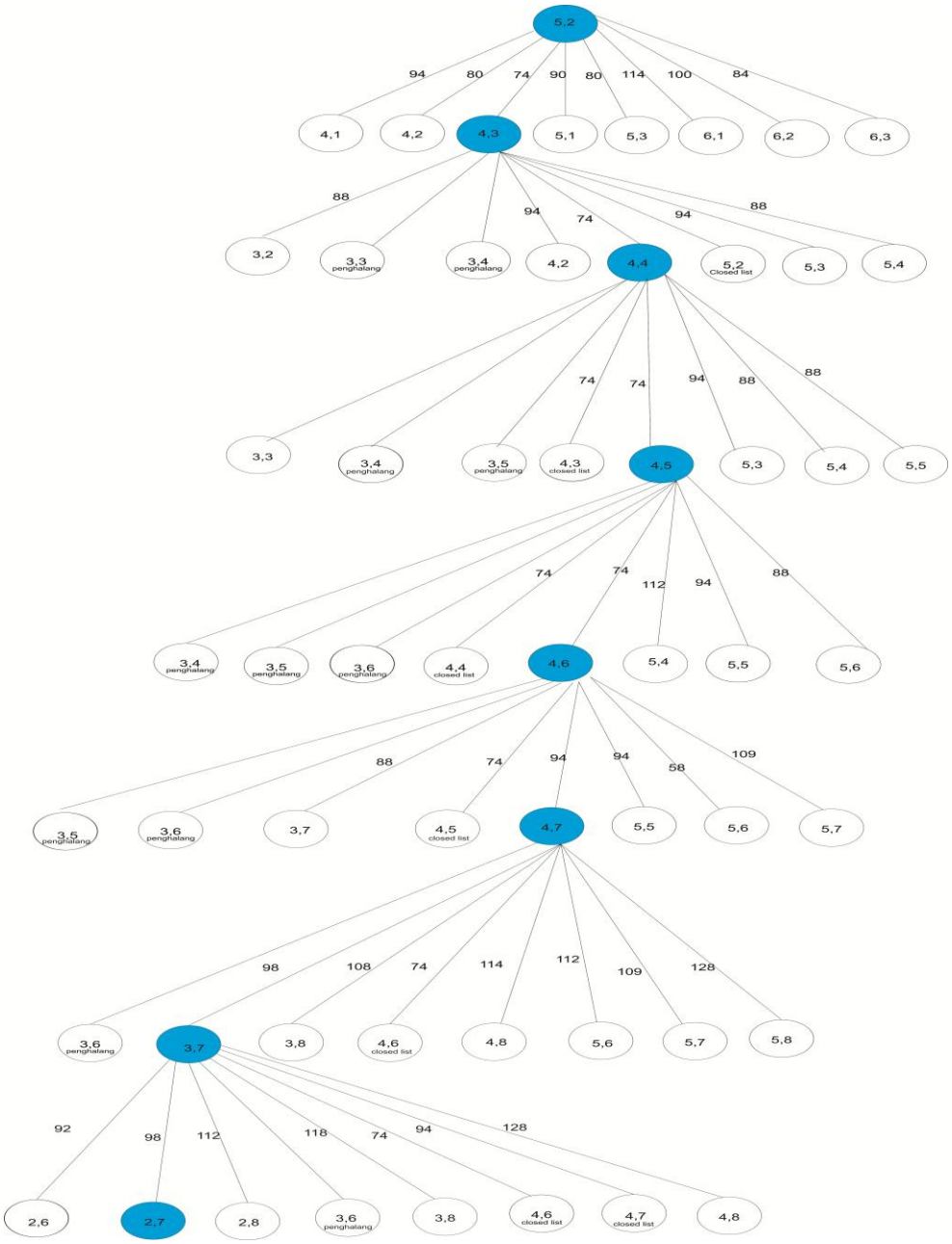


Gambar 3.17 Penjelasan A* pada langkah terakhir

Berikut ini penjelasan melalui tabel untuk pencarian jalan terpendek pada langkah terakhir.

Tabel 3.8 Tabel Perhitungan langkah terakhir

Open List	(2,6), (2,7), (2,8), (3,8), (4,8)
Closed list	(5,2), (4,3), (4,4), (4,5), (4,6), (4,7), (3,7)
Parent	(3,7)
Current node	(2,7)



Gambar 3.18 Penjelasan A* dengan *tree* pada langkah terakhir

3.3.2 Analisis *Fuzzy Logic*

Pada *game* ini *fuzzy logic* digunakan untuk menentukan variasi prilaku yang dilakukan oleh musuh / NPC (*Non Playable Character*). Dengan adanya *fuzzy logic* masing-masing musuh / NPC (*Non Playable Character*) dapat merubah perilaku berbasis perubahan variabel masukan. Metode *fuzy* yang

digunakan adalah metode sugeno karena metode ini menghasilkan keluaran yang berupa konstan tegas, sehingga dapat mewakili nilai perilaku yang telah dirancang.

Untuk menghasilkan prilaku pada musuh ada tiga variabel yang digunakan, yaitu *Distance* (*Near*, dan *Far*), *Health Point* (*Injured* dan *Fit*) dan *Decision Maker* (*Escape, Defend, dan Attack*) Dengan menggunakan dua variabel diharapkan NPC mampu menentukan prilaku yang akan dilakukan.

Tahapan-tahapan dari analisis *fuzzy logic* yaitu :

A. Variabel Fuzzy

Ada 3 variabel yang digunakan dalam *fungsi fuzzy*, yaitu sebagai berikut:

1. *Distance* antara *player* dengan musuh atau NPC (*Non Playable Character*).
2. *HP (Health Point),* merupakan point darah (kesehatan) dari musuh / NPC (*Non Playable Character*).
3. *Decision Maker,* keputusan yang akan diambil oleh musuh / NPC (*Non Playable Character*).

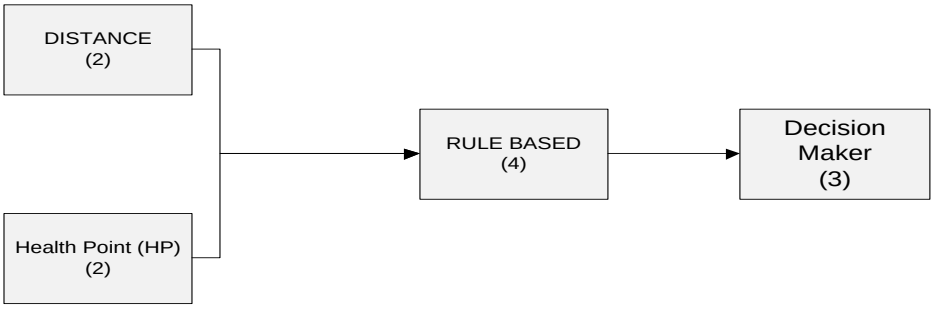
B. Nilai linguistik / terma

Dari tiga variabel yang digunakan, memiliki nilai linguistik sebagai berikut :

1. *Distance* : *Near, Far*
2. *HP (Health Point*) : *Injured, Fit*
3. *Decision Maker* : *Escape, Defend, Attack*

C. Fuzzyfikasi

Fuzzyfikasi merupakan proses memetakan nilai *crisp* (numerik) ke dalam himpunan *fuzzy* dan menentukan derajat keanggotaannya. Secara garis besar pemetaan nilai *crisp* ke dalam himpunan *fuzzy* dijelaskan dengan gambar 3.19



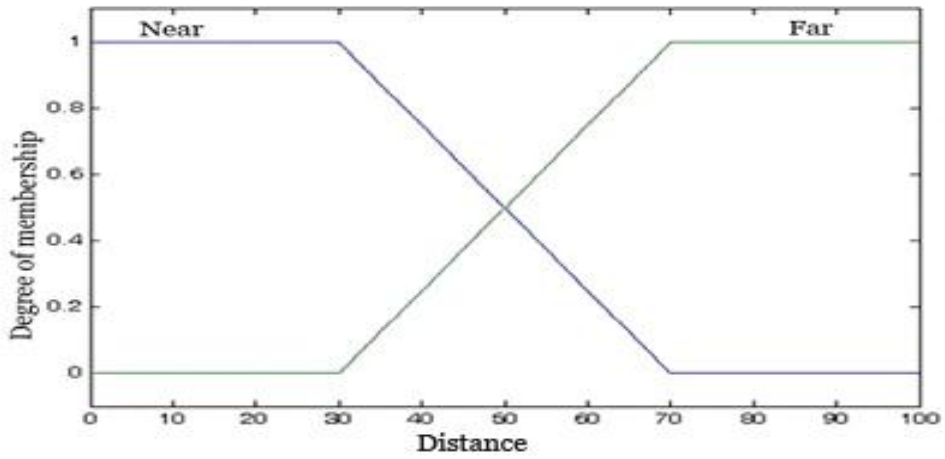
Gambar 3.19 *Fuzzy Interface System*

Berdasarkan *fuzzy interface system* di atas maka pemetaan *input-output* adalah sebagai berikut:

1. *Variabel Distance*, terbagi menjadi beberapa *input* yaitu : *near* dan *far*. *Range* nilai untuk variabel *distance* diantara 0-100 dengan penjelasan setiap *terma* sebagai berikut

a) *Terma Near* = 0 – 70

b) *Terma Far* = 30 – 100



Gambar 3.20 Grafik variabel *Distance*

Perhitugan *Fuzzyfikasi* variabel *Distance*:

$$\mu DisNear[x]\begin{cases}1; x \leq 30\\ \dfrac{70-x}{40}\\ 0; x \geq 70\end{cases}; 30 \leq x \leq 70$$

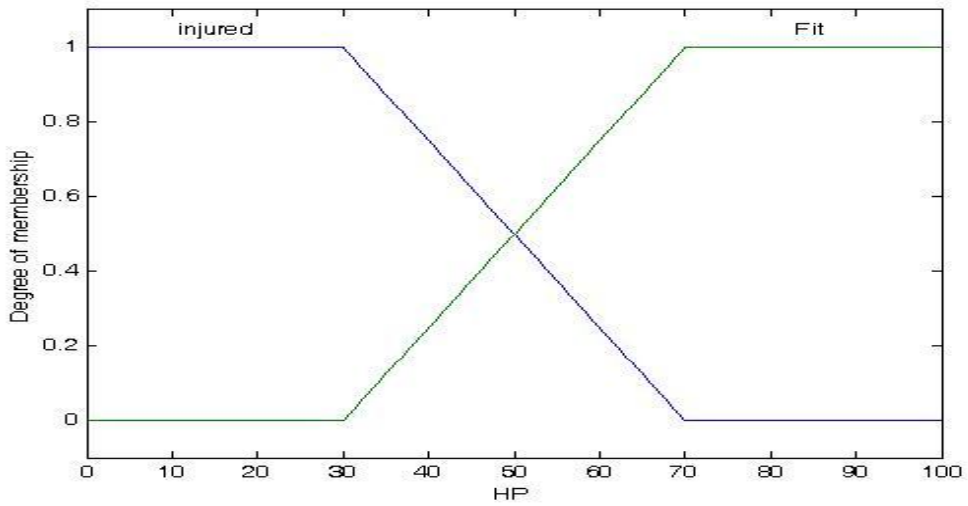
$$\mu DisFar[x]\begin{cases}0; x \leq 30\\ \dfrac{x-30}{40}\\ 1; x \geq 70\end{cases}; 30 \leq x \leq 70$$

Keterangan :

μ = derajat keanggotaan variabel

2. Variabel *Health Point* (HP), terbagi menjadi beberapa *input* yaitu : *injured* dan *fit*. *Range* nilai variabel HP diantara 0 – 10 dengan penjelasan *terma* sebagai berikut:

a) *Terma Injured* = 0 – 70
b) *Terma Fit* = 30 – 100



Gambar 3.21 Grafik variabel HP

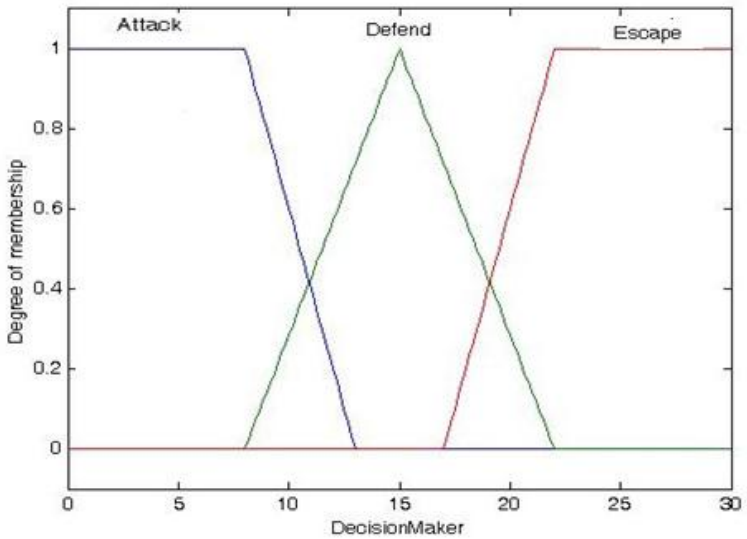
Perhitungan *Fuzzyfikasi* variabel HP:

$$\mu HPInjured[y] = \begin{cases} 1; y \le 30 \\ \dfrac{70-y}{40}; 30 \le y \le 70 \\ 0; y \ge 70 \end{cases}$$

$$\mu HPFit[y] = \begin{cases} 0; y \le 30 \\ \dfrac{y-30}{40}; 30 \le y \le 70 \\ 1; y \ge 70 \end{cases}$$

3. Variabel *Decision maker*, yaitu menyerang, bertahan dan melarikan diri. *Range* nilai variabel *Desicion Maker* diantara 0-30 dengan penjelasan *terma* sebagai berikut:

a) *Terma Attack* = 0 – 13
b) *Terma Defend* = 8 – 22
c) *Terma Escape* = 17- 30



Gambar 3.22 Grafik *output Decisionmaker*

Perhitungan *fuzzyfikasi Output Decisionmaker*

$$\mu DAttack[z]\begin{cases}1; z \le 8\\ \dfrac{13-z}{5}\\ 0; z \ge 13\end{cases}; 8 \le z \le 13$$

$$\mu DDefend[z]=\begin{cases}0; z \le 8 / z \ge 22\\ \dfrac{z-8}{7}; 8 \le z \le 15\\ \dfrac{15-z}{7}; 15 \le z \le 22\end{cases}$$

$$\mu DEscape[z]\begin{cases}0; z \leq 17\\ \dfrac{z-13}{5}\\ 1; z \geq 22\end{cases}; 17 \leq z \leq 22$$

4. Kaidah fuzzy atau *Rule Based*

Kaidah *fuzzy (rules)* atau aturan-aturan yang diterapkan dalam penentuan perilaku *Decision Maker* ada enam *rules* yaitu:

***Rule-Based* tindakan karakter musuh**

	DISTANCE		**HP**		**Decision Maker**
IF	Near	AND	Injured	THEN	**Escape**
IF	Near	AND	Fit	THEN	**Attack**
IF	Far	AND	Injured	THEN	**Defend**
IF	Far	AND	Fit	THEN	**Attack**

5. Agregasi dan Defuzzyfikasi

Agregasi untuk menentukan keluaran semua *rules* yang dikombinasikan menjadi sebuah *fuzzy set* tunggal. *Defuzzyfikasi* merupakan proses pemetaan besaran dari himpunan *fuzzy set* yang dihasilkan ke dalam bentuk nilai *crisp.* *Defuzzifikasi* pada metode Sugeno adalah dengan menghitung *center of single-ton* atau titik pusat nilai *crisp* dengan metode rata-rata (average)

$$z^* = \frac{\sum \mu i.zi}{\sum \mu i}$$

z*=titik pusat nilai *crisp*

μi=derajat keanggotaan masing-masing variabel

zi = nilai domain pada variabel linguistik z

Contoh Penyelesaian Kasus

Misalkan variabel distance diberikan nilai *crisp* sebesar 60, dan variabel hp diberikan nilai sebesar 40, maka proses agregasi yang menentukan keluaran sesuai dengan *rules* yang sudah dibuat akan dijelaskan seperti berikut.

[Distance= 60 HP=40]

Fuzzyfikasi:

$$\mu DisNear[60] = \frac{70-60}{40} = 0{,}25$$

$$\mu DisFar[60] = \frac{60-30}{40} = 0{,}75$$

$$\mu HPInjured[40] = \frac{70-40}{40} = 0{,}75$$

$$\mu HPFit[40] = \frac{40-30}{40} = 0{,}25$$

Operasi Logika Fuzzy dan Implikasi

a) Rule 1 : IF Distance is *Near* and HP is *Injured* Then *Escape*

→ min(0.25 , 0.75) = 0.25

b) Rule 2 : IF Distance is *Near* and HP is *Fit* Then *Attack*

→ min(0.25 , 0,25) = 0.25

c) Rule 3 : IF Distance is *Far* and HP is *Injured* Then *Defend*

→ min(0.75 , 0.75) = 0.75

d) Rule 4 : IF Distance is *Far* and HP is *Fit* Then *Attack*

→ min(0.75 , 0.25) = 0.25

Defuzzy

a) Rule 1: $\frac{13-z}{5} = 0.25$

$z = 13 - 1.25 = 11.75$

b) Rule 2: $\frac{z-17}{5} = 0.25$

$z = 1.25 + 17 = 18.25$

c) Rule 3: $\frac{z-8}{7} = 0.75$

$z = 5.25 + 8 = 13.25$

d) Rule 4: $\frac{z-17}{5} = 0.25$

$z = 1.25 + 17 = 18.25$

$$z^* = \frac{\sum \mu i.zi}{\sum \mu i}$$

➔ $z^* = \frac{0.25 \times 11.75 + 0.25 \times 18.75 + 0.75 \times 13.25 + 0.25 \times 18.25}{0.25 + 0.25 + 0.75 + 0.25}$

➔ $z^* = \frac{2.9 + 4.6 + 9.9 + 4.6}{0.25 + 0.25 + 0.75 + 0.25}$

➔ $z^* = \frac{22}{1.5} = 14,66$

Jadi keputusan *output* yang diambil pada *Decisionmaker* bernilai 14,66 (*Defend*) sesuai dengan grafik *decisionmaker* dan sesuai dengan *Rules* 3.

Dengan Distance = 60 dan HP = 40 maka decisionmaker yang diambil adalah 14,66 (Defend)

3.4 Analisis Kebutuhan Non-Fungsional

Analisis non-fungsional merupakan analisis yang dibutuhkan untuk menentukan spesifikasi kebutuhan sistem. Spesifikasi ini juga meliputi elemen atau komponen-komponen apa saja yang dibutuhkan untuk sistem yang akan dibangun sampai dengan sistem tersebut diimplementasikan. Analisis kebutuhan ini juga menentukan spesifikasi masukan yang diperlukan sistem, keluaran yang akan dihasilkan sistem dan proses yang dibutuhkan untuk mengolah masukan sehingga menghasilkan suatu keluaran yang diinginkan. Pada analisis kebutuhan

sistem non fungsional ini dijelaskan analisis kebutuhan perangkat lunak, analisis kebutuhan perangkat keras, dan analisis pengguna.

3.4.1 Analisis Kebutuhan Perangkat Lunak

Perangkat lunak atau *software* merupakan hal yang terpenting dalam mendukung kinerja sebuah sistem. Perangkat lunak digunakan dalam sebuah sistem merupakan perintah-perintah yang diberikan kepada perangkat keras agar dapat saling berinteraksi diantara keduanya. Perangkat lunak yang dapat dibutuhkan untuk membangun aplikasi *game First Person Shooter* 3D Sejarah Medan Area ini adalah sebagai berikut:

1. Aplikasi *game* yang dihasilkan berbasis *desktop*.
2. Pemodelan dan perancangan sistem dalam penulisan laporan ini menggunakan pemodelan berbasis objek dengan *tools* UML (*Unified Modeling Language).*
3. Dikembangkan dengan menggunakan *tools game engine Unity 3D,* dengan bahasa pemrograman berupa C#, *Javascript,* dan *Boo.*
4. *Rendering* gambar 3D menggunakan *3ds Max,* dan gambar 2D menggunakan *Adobe Photoshop CS5*.

Dari hasil pengamatan perangkat lunak, terdapat beberapa hal penting yang harus diperhatikan agar sistem yang akan dirancang dapat berjalan dengan baik dan sesuai dengan yang diharapkan. Pengguna diharapkan dapat memenuhi kebutuhan perangkat lunak yang telah direkomendasikan agar aplikasi dapat berjalan dengan baik.

3.4.2 Analisis Kebutuhan Perangkat Keras

Agar aplikasi dapat berjalan dengan baik, maka dibutuhkan perangkat keras yang sesuai dengan kebutuhan aplikasi. Spesifikasi minimum perangkat keras yang dibutuhkan agar dapat menjalankan *game* FPS 3D Sejarah Medan Area ini dengan optimal adalah sebagai berikut:

1. Prosesor minimal kecepatan 2.0 GHz.

2. RAM minimal 2 GB.
3. Harddisk dengan ruang kosong 10 GB.
4. VGA Card shared 1 GB.
5. Monitor dengan resolusi 1024 x 768 pixels.
6. *Speaker*.
7. *Keyboard* dan *mouse*

3.4.3 Analisis *User*

Pada analisis *user* (pemakai) ini akan mencakup analisis beberapa parameter terhadap calon *user* dari aplikasi.

1. ***User Knowledge and Experience* dari target user yang akan menggunakan game.**

Game FPS sejarah medan area ini bisa digunakan oleh kalangan apapun, tetapi pengetahuan dan pengalaman akan memudahkan *user* dalam penggunaanya. Terutama pengetahuan dan pengalaman dalam memainkan *game action* atau *first person shooter*. Berikut ini klasifikasi *knowledge and experience* dari pengguna aplikasi:

Tabel 3.9 Analisis klasifikasi *knowledge and experience*

Educational level	**Reading Level**	**Typing Skills**
Game ini bisa digunakan oleh semua kalangan, seperti pelajar, mahasiswa hingga masyarakat awam	Game ini bisa digunakan oleh berbagai level pendidikan dengan reading level yang sedang	Game ini tidak memerlukan typing skills yang tinggi
Computer Literacy	**Task Experience**	**System Experience**
Game ini bisa digunakan oleh pengguna yang	Game ini bisa digunakan oleh pengguna dengan kemampuan pengetahuan	Game ini bisa digunakan oleh pengguna dengan

memiliki kemampuan komputer yang moderate (menengah)	komputer dan *game* yang sedang.	pengalaman komputer dan *game* yang sedang
Application Experience	**Native Language**	**Use Of Other System**
Game ini bisa digunakan dalam sistem operasi Windows 7	Game ini menggunakan satu bahasa, yakni Bahasa Indonesia	Game bisa di jalankan tanpa install aplikasi lain

2. *Users Physical Characteristic*

Keadaan fisik seseorang mungkin akan berpengaruh pada penggunaan aplikasi *game* ini. Ada hal-hal yang harus diperhatikan juga terhadap *user* dari karakteristik fisiknya untuk dapat menggunakan aplikasi ini yaitu, *Color Blind*, *Handednes*, dan *Gender*.

Tabel 3.10 Analisis *Users Physical Characteristic*

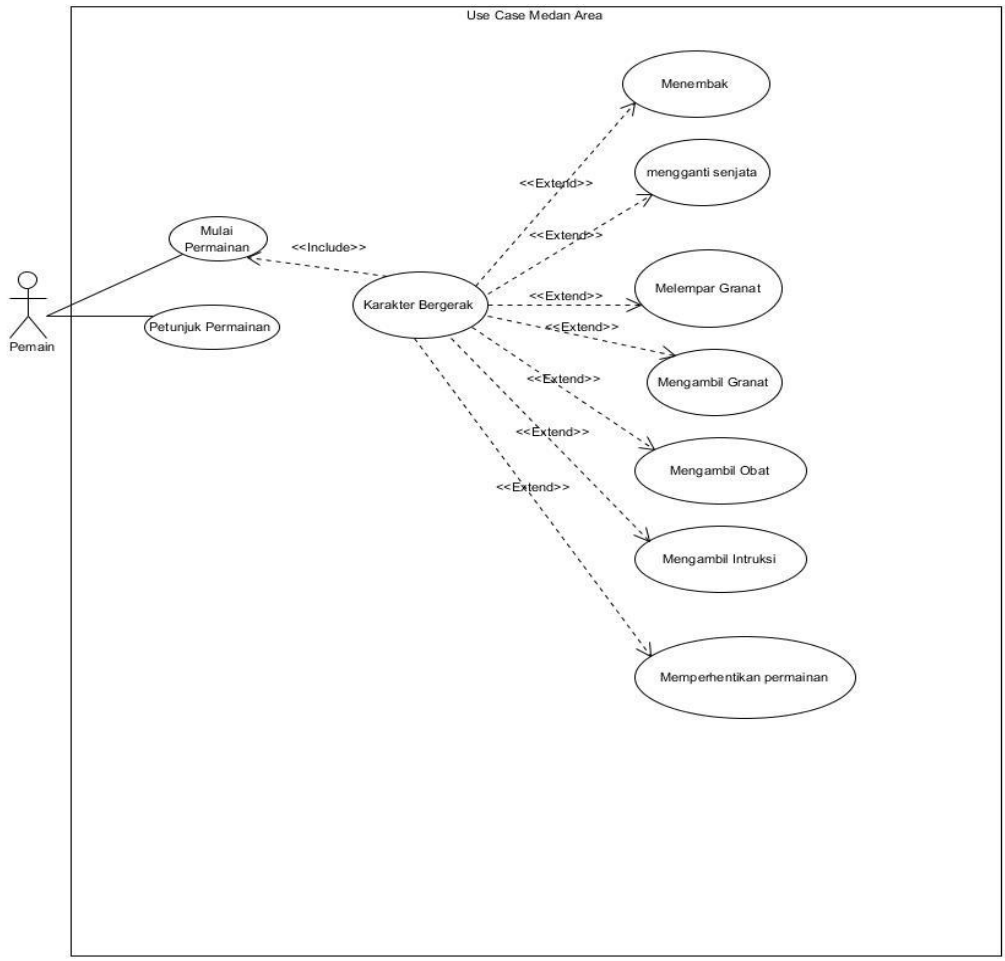
Age	Untuk anak SMA kelas XII
Gender	Pria dan Wanita
Handedness	Kanan dan Kiri
Color Blind	User yang tidak bisa membedakan warna yang satu dengan yang lainnya (buta warna) masih mampu menggunakan aplikasi ini, karena tidak ada indicator warna-warna khusus yang membedakan antara fungsional yang satu dengan fungsional yang lainnya. Akan tetapi penggunaannya tidak akan optimal karena dalam *game* ini terdapat banyak sekali perbedaan warna yang menunjang interaksi dan ketertarikan dalam permainan.

3.5 Analisis Kebutuhan Fungsional

Tahap pertama yang dilakukan dalam melakukan analisis berorientasi objek menggunakan UML adalah menentukan aktor atau pengguna sistem. Kata aktor dalam konteks UML, menampilkan peran (*roles*) pengguna atau sesuatu diluar sistem yang dikembangkan yang dapat berupa perangkat keras, *end user*, sistem yang lain, dan sebagainya.

3.5.1 Use Case Diagram

Use case diagram digunakan untuk memodelkan dan menyatakan unit fungsi atau layanan yang disediakan oleh sistem ke pemakai. Berikut ini adalah *use case* diagram dari *game* :



Gambar 3.23 Use Case Diagram *Game* Sejarah Medan Area

3.5.1.1 Defenisi *Use Case*

Definisi *use case* berfungsi untuk menjelaskan proses yang terdapat pada setiap *use case*. Berikut di bawah ini tabel dari definisi *use case* yaitu :

Tabel 3.11 Definisi *Use Case*

No	*Use Case*	Deskripsi
1	Mulai permainan	Proses untuk memulai permainan.
2	Petunjuk permainan	Proses untuk menampilkan petunjuk pengguanaan pada *game*.
3	Karakter bergerak	Proses untuk menggerakan karakter.
4	Menembak	Proses untuk menghancurkan pasukan NICA, gudang senjata dan tulisan "*FIXED BOUNDERIUS MEDAN AREA"*.
5	Mengganti senjata	Proses untuk mengganti senjata pada *player*
6	Melempar granat	Proses untuk melempar senjata granat terhadap musuh.
7	Mengambil granat	Proses untuk menambah senjata granat.
8	Melihat obat	Proses untuk menambah darah karakter jika berkurang.
9	Mengambil intruksi	Proses untuk mengetahui petunjuk pemain.
10	Memperhentikan permainan	Proses untuk menghentikan sejenak permainan dengan menekan tombol ESC.

3.5.1.2 Skenario Use Case

Skenario *use case* merupakan bagian pada *use case* yang menunjukkan proses apa saja yang terjadi pada setiap bagian di dalam *use case*, dimana pengguna memberikan perintah pada setiap bagian dan respon apa yang diberikan oleh sistem kepada pengguna setelah pengguna memberikan perintah pada setiap bagian-bagian *use case*. Berikut ini beberapa skenario *use case* berdasarkan *use case* yang ada yaitu :

Tabel 3.12 Skenario Use case Mulai Permainan

Identifikasi		
Nomor	1	
Nama	Mulai permainan	
Tujuan	Untuk memulai permainan	
Deskripsi	Proses awal dalam memulai permainan	
Aktor	Pemain	
Skenario Utama		
Kondisi awal	Pemain berada di menu utama	
Aksi Aktor		Reaksi Sistem
Memilih menu mulai permainan dengan mengerakkan mouse		Langsung menampilkan scene permainan.

Tabel 3.13 Skenario *Use Case* Petunjuk Permainan

Identifikasi		
Nomor	2	
Nama	Petunjuk Permainan	
Tujuan	Untuk mengetahui cara memainkan game	
Deskripsi	Proses untuk menampilkan petunjuk cara permainan	
Aktor	Pemain	
Skenario Utama		
Kondisi awal	Pemain berada di menu utama	
Aksi Aktor		Reaksi Sistem
Memilih menu petunjuk permainan		Menampilkan petunjuk cara memainkan permainan.

Identifikasi		
Nomor	3	
Nama	Karakter bergerak	
Tujuan	Untuk menggerakkan karakter pemain	
Deskripsi	Proses untuk menggerakan karakter pemain	
Aktor	Pemain	
Skenario Utama		
Kondisi awal	Pemain berada di dalam permainan	
Aksi Aktor		Reaksi Sistem
Menekan tombol W,A,S,D pada keyboard		Karakter bergerak maju, mundur, kiri, dan kanan.

Tabel 3.14 Skenario *Use Case* Menembak

Identifikasi	
Nomor	4
Nama	Menembak
Tujuan	Untuk menyerang musuh atau membunuh musuh
Deskripsi	Proses untuk menyelesaikan misi
Aktor	Pemain
Skenario Utama	
Kondisi awal	Pemain menggunakan senjata
Aksi Aktor	Reaksi Sistem
Mengklik tombol kiri pada mouse	Karakter mengeluarkan peluru dari senjata yang di gunakan.
Peluruh di dalam bar berkurang, jika habis maka reload otomatis	

	Sistem akan mengeluarkan peluru dalam senjata yang sedang di gunakan.

Table 3.15 Skenario *Use Case* mengganti senjata

Identifikasi		
Nomor	5	
Nama	Menganti senjata	
Tujuan	Untuk mengganti senjata yang di gunakan oleh pemain	
Deskripsi	Proses untuk mengganti senjata	
Aktor	Pemain	
Skenario Utama		
Kondisi awal	Pemain menggunakan salah satu senjata	
Aksi Aktor		Reaksi Sistem
Menggerakkan scroll pada mouse		Senjata karakter berganti.

Table 3.16 Skenario *Use Case* Melempar Granat

Identifikasi		
Nomor	6	
Nama	Melempar Granat	
Tujuan	Membunuh musuh atau gardu dengan melempar granat	
Deskripsi	Proses untuk melempar granat terhadap musuh	
Aktor	Pemain	
Skenario Utama		
Kondisi awal	Pemain menggunakan salah satu senjata	
Aksi Aktor		Reaksi Sistem
Menggerakkan scroll pada mouse		Senjata menggunakan granat

Table 3.17 Skenario *Use Case* Mengambil Granat

Identifikasi		
Nomor	7	
Nama	Mengambil granat	
Tujuan	Untuk menambah jumlah granat	
Deskripsi	Proses untuk menambahkan senjata granat pemain	
Aktor	Pemain	
Skenario Utama		
Kondisi awal	Pemain mendekati ikon granat	
Aksi Aktor		Reaksi Sistem
Menekan tombol E pada keyboard		Ikon granat hilang dan granat bertambah.

Tabel 3.18 Skenario *Use Case* mengambil obat

Identifikasi		
Nomor	8	
Nama	Mengambil obat	
Tujuan	Menambah darah pemain	
Deskripsi	Proses ketika pemain akan mengambil ikon obat	
Aktor	Pemain	
Skenario Utama		
Kondisi awal	Sistem menampilkan ikon obat	
Aksi Aktor		Reaksi Sistem
		Sistem menampilkan ikon obat

Pemain mengambil ikon obat (darah karakter berkurang)	
	Ikon obat hilang
Darah karakter pemain bertambah	
Pemain mengambil ikon obat (darah karakter dalam kondisi penu)	
	Ikon obat tidak menghilang
Darah karakter tidak bertambah / tetap	
Kondisi Akhir	Sistem akan menambah darah karakter pemain jika darah pemain dalam keadaan berkurang

Tabel 3.19 Skenario *Use Case* Mengambil intruksi

Identifikasi	
Nomor	9
Nama	Mengambil intruksi
Tujuan	Untuk mengetahui petunjuk permainan.
Deskripsi	Proses untuk mengetahui intruksi permainan
Aktor	Pemain
Skenario Utama	
Kondisi awal	Pemain berada di dekat ikon intruksi
Aksi Aktor	Reaksi Sistem
Karakter di gerakkan menyentuh ikon intruksi	
	Icon mengeluarkan suara dan tulisan
Ikon intruksi menghilang	
	Informasi tempat di ketahui.

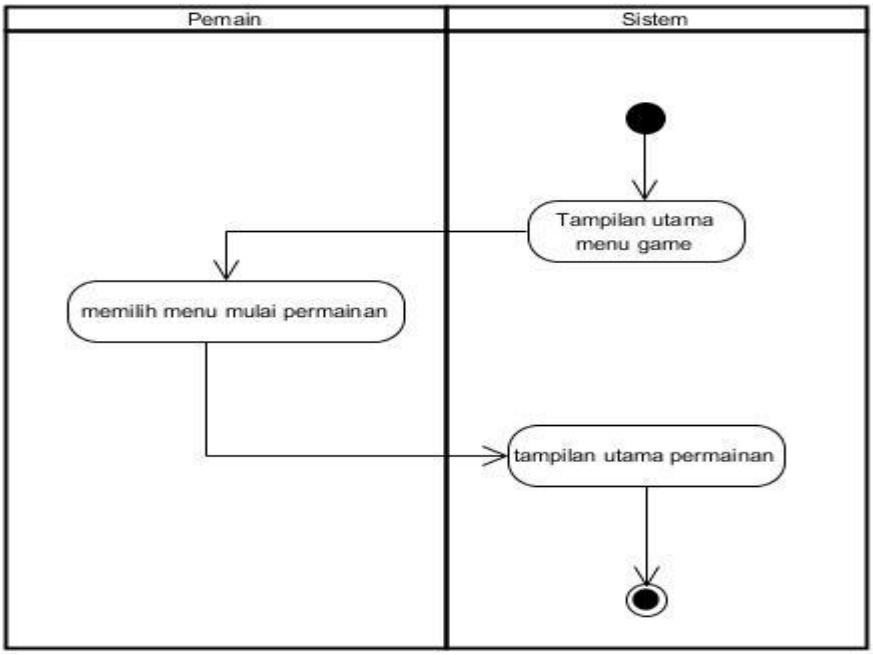
Tabel 3.20 Skenario *Use Case* Memperhentikan permainan

Identifikasi	
Nomor	10
Nama	Memperhentikan permainan
Tujuan	Untuk menghentikan sejenak permainan
Deskripsi	Proses untuk berhenti sejenak dalam permainan
Aktor	Pemain
Skenario Utama	
Kondisi awal	Pemain menekan tombol ESC pada keyboard
Aksi Aktor	Reaksi Sistem
Menekan tombol ESC pada keyboard	
	Menghentikan permainan sejenak
	Muncul 3 pilihan menu
Pemain memilih menu Lanjut	
	Permainan dapat di mainkan kembali di kondisi seperti sebelum di hentikan
Pemain memilih menu Mulai baru	
	Permainan kembali ke kondisi permainan baru.
Pemain memilih keluar	
	Keluar dari permainan dan kembali ke munu utama.

3.5.2 *Activity Diagram*

Activity diagram menggambarkan berbagai aliran aktivitas dalam sistem yang sedang dirancang. Berikut ini beberapa *activity diagram* yang terdapat pada *game* ini yaitu :

a. *Activity Diagram* Mulai Permainan

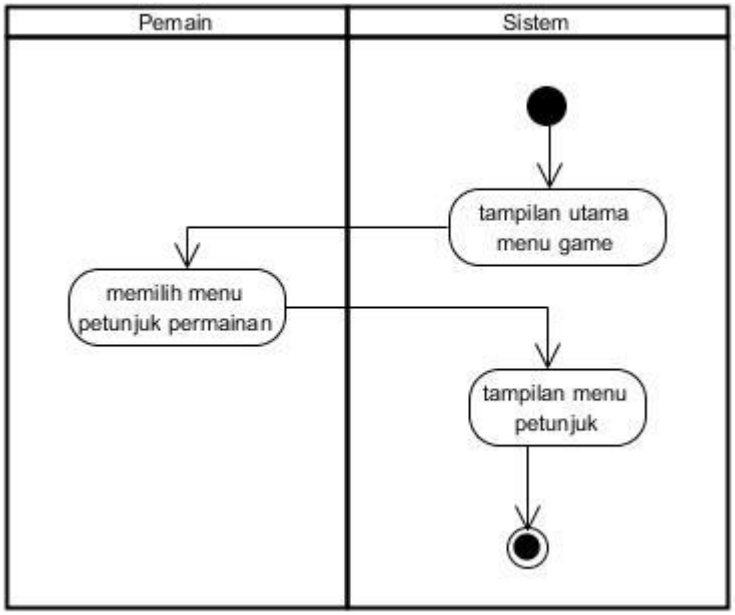


Gambar 3.24 *Activity Diagram* Mulai Permainan

Tabel 3.21 Penjelasan *Activity Diagram* Mulai Permainan

Deskripsi	Proses Mulai permainan
Nomor Flow	*1.* Tampilan Menu utama dalam *game* 2. Pemain memilih menu mulai permain 3. Tampilan menu utama permainan

b. *Activity Diagram* Petunjuk Permainan

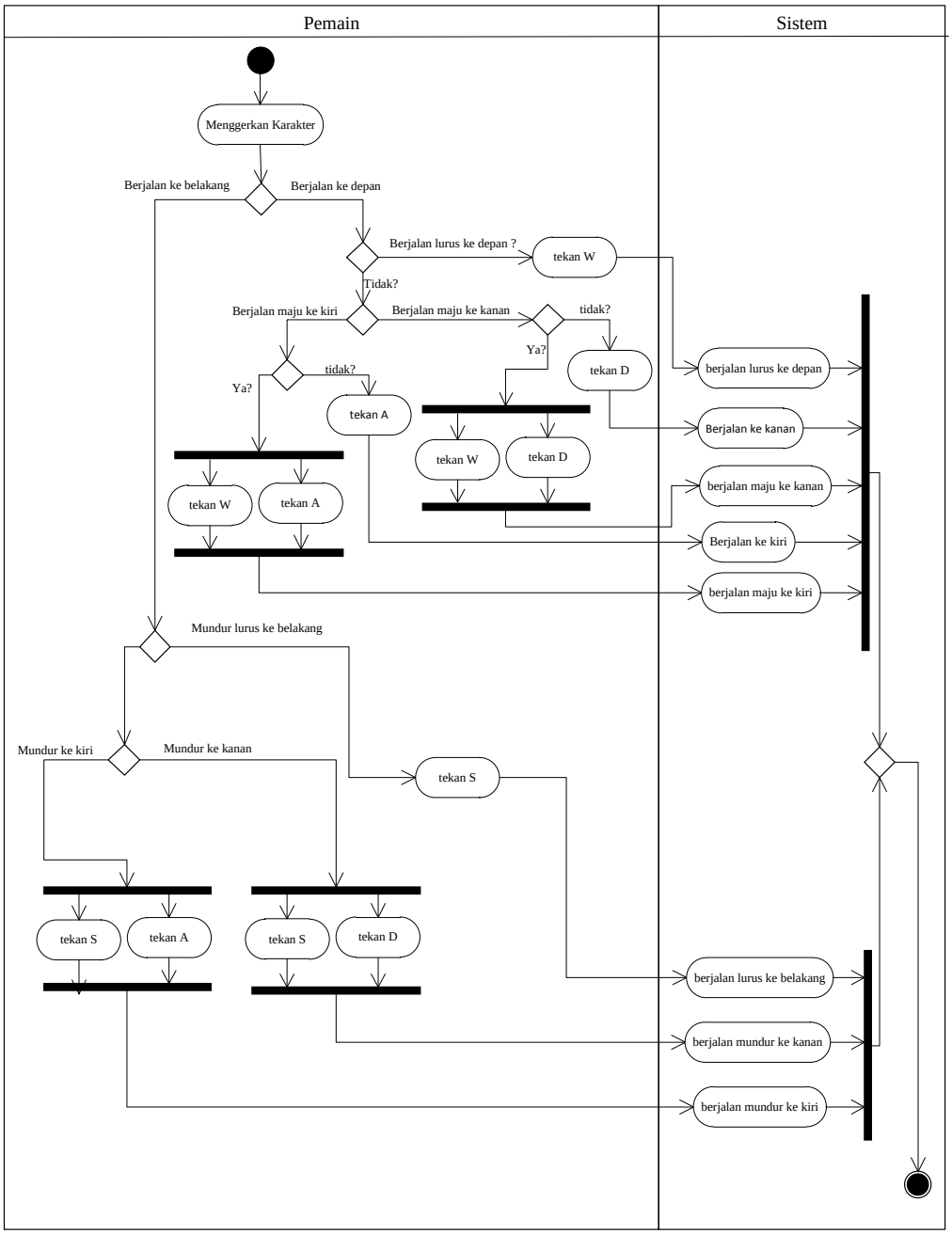


Gambar 3.25 *Activity Diagram* Petunjuk Permainan

Tabel 3.22 Penjelasan *Activity Diagram* Petunjuk Permainan

Deskripsi	Proses Petunjuk permainan
Nomor Flow	1. Tampilan Menu utama dalam game 2. Pemain memilih menu petunjuk permain 3. Tampilan menu petunjuk cara permainan

c. *Activity Diagram* Pergerakan Karakter



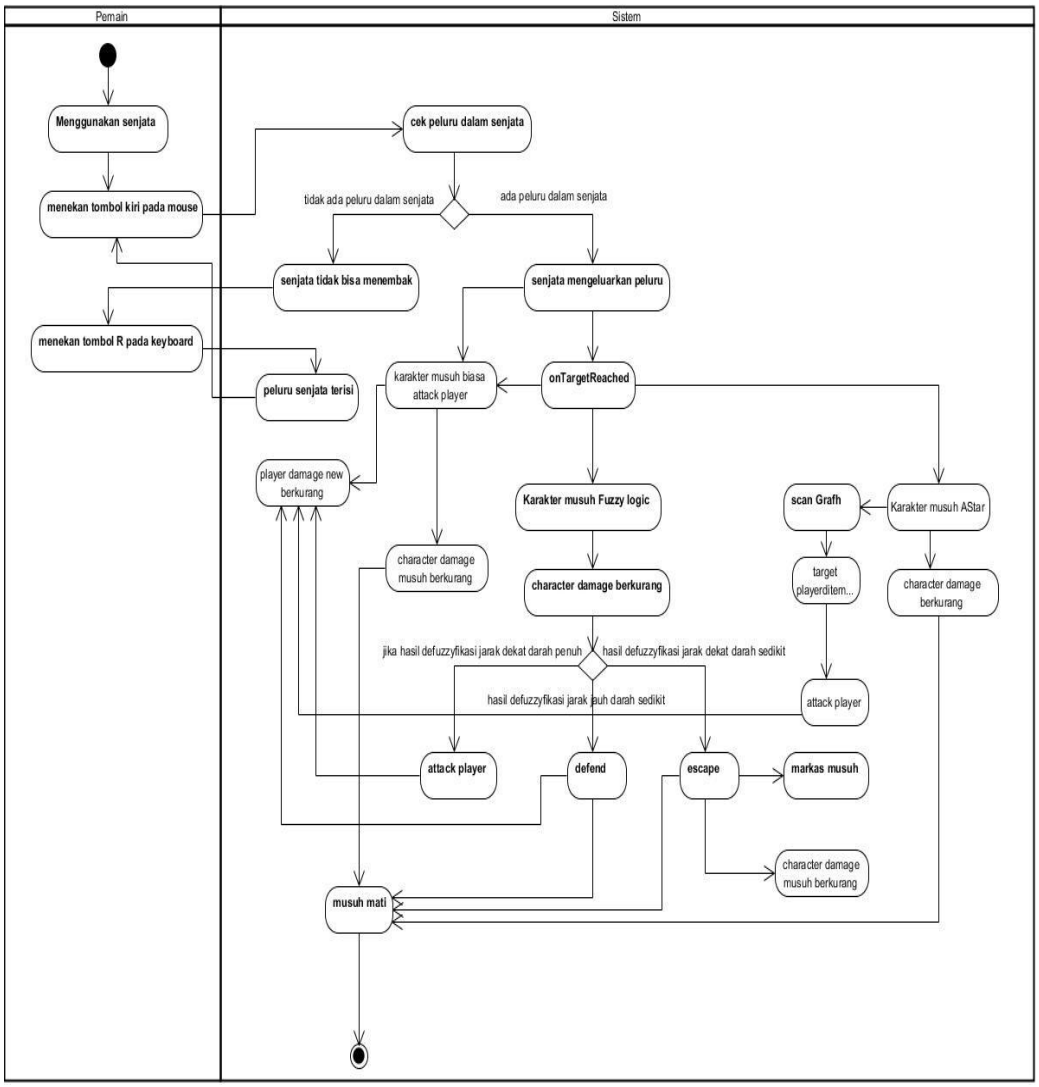
Gambar 3.26 *Activity Diagram* Pergerakan Karakter

Tabel 3.23 Penjelasan Proses *activity diagram* Pergerakan Karakter

Deskripsi	Proses Berjalan
Normal Flow	1. Pemain menggerakan karakter

	2. Pemain menekan tombol W 3. Karakter berjalan ke depan 4. Pemain menekan tombol W dengan A secara bersamaan 5. Karakter berjalan maju ke sebelah kiri 6. Pemain menekan tombol A 7. Karakter berjalan ke kiri 8. Pemain menekan tombol W dengan D secara bersamaan 9. Karakter berjalan maju ke kanan 10. Pemain menekan tombol D 11. Karakter berjalan ke kanan 12. Pemain menekan tombol S 13. Karakter berjalan lurus mundur 14. Pemain menekan tombol S dengan D secara bersamaan 15. Karakter berjalan mundur ke kanan 16. Pemain menekan tombol S dengan A secara bersamaan 17. Karakter berjalan mundur ke kiri D untuk berjalan ke arah kanan.

d. *Activity* proses Menembak



Gambar 3.27 *Activity diagram* Proses Menembak

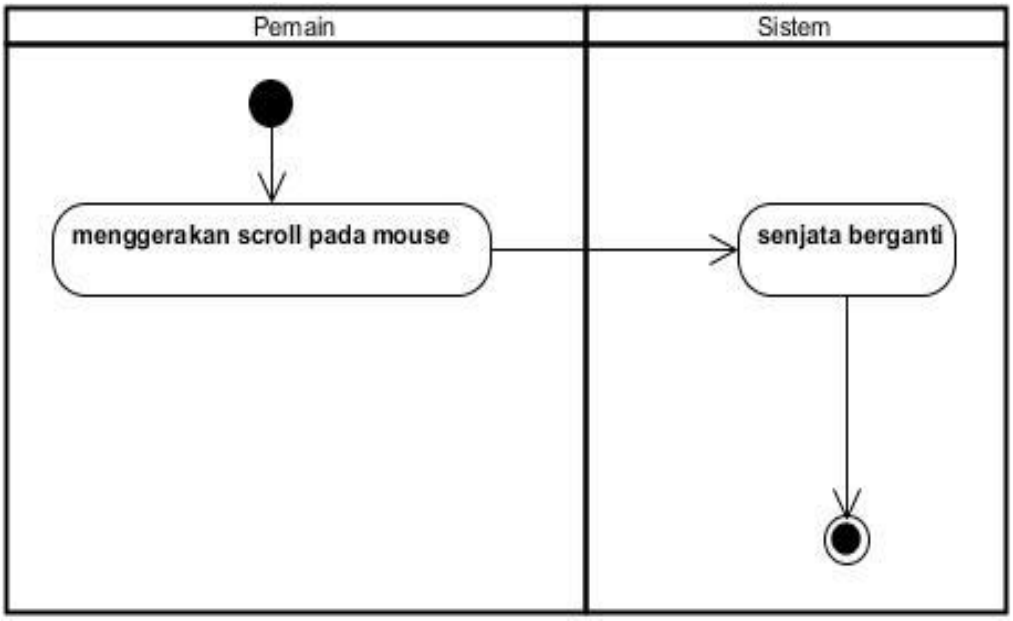
Tabel 3.24 Penjelasan Proses *activity* diagram Menembak

Deskripsi	Karakter mengklik kiri pada *mouse* untuk mengeluarkan peluru pada senjata.
Normal Flow	1. Pemain mengklik kiri pada *mouse* 2. Peluru senjata di cek 3. Peluru senjata ada

	4. Senjata mengeluarkan peluru 5. Peluru senjata tidak ada 6. Pemain menekan tombol R pada keyboard 7. Peluru senjata terisi 8. Peluru senjata mengenai onTargetReached 9. Target musuh terdiri dari 3 yaitu musuh biasa, fuzzy logic, musuh AStar. 10. Karakter musuh biasa tertembak 11. Damage berkurang 12. Karakter musuh mati 13. Karakter musuh biasa attack player 14. Player damage pemain berkurang 15. Karakter musuh fuzzy logic tertembak 16. Character damage berkurang 17. Musuh fuzzy logic attack jika kondisi jarak dekat dan darah penuh 18. Player damage new berkurang 19. Musuh fuzzy logic defend jika kondisi jarak jauh dan darah sedikit 20. Player damage new berkurang 21. Musuh melarikan diri kemarkas 22. Musuh fuzzy logic escape jika kondisi jarak dekat dan dan darah sedikit 23. Musuh melarikan diri 24. Musuh mati jika character damage habis. 25. Karakter musuh Astar tertembak 26. Character damage berkurang 27. Karakter musuh mati 28. Karakter musuh Astar mencari jalar terpendek. 29. Jalur terpendek ditemukan

	30. Karakter musuh menyerang player.

e. Activity Diagram Proses Mengganti Senjata

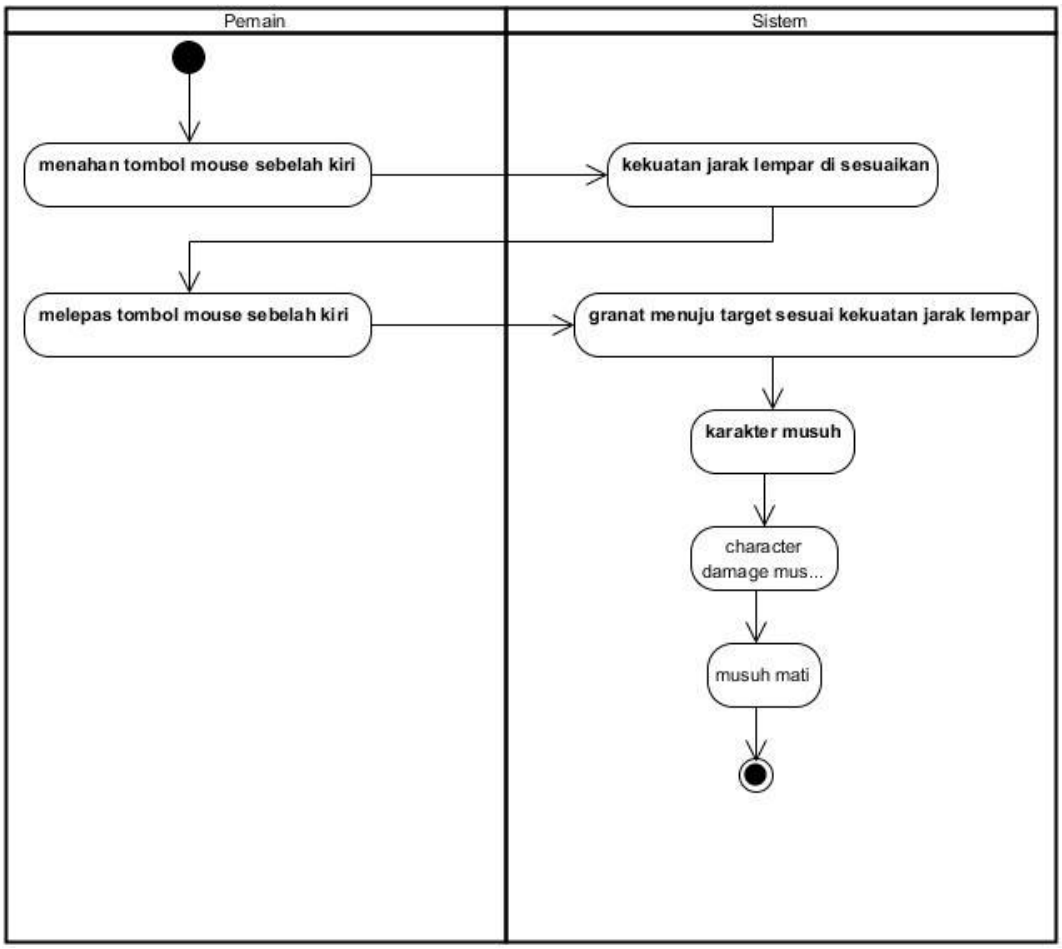


Gambar 3.28 Activity Diagram Proses Mengganti Senjata

Tabel 3.25 Penjelasan Proses *activity* Mengganti Senjata

Deskripsi	Karakter menggerakkan *scroll* pada *mouse*
Normal Flow	1. Pemain menggerakan *scroll* pada *mouse* 2. Senjata digunakan berganti sesuai urutannya.

f. Activity Diagram Proses Melempar Granat



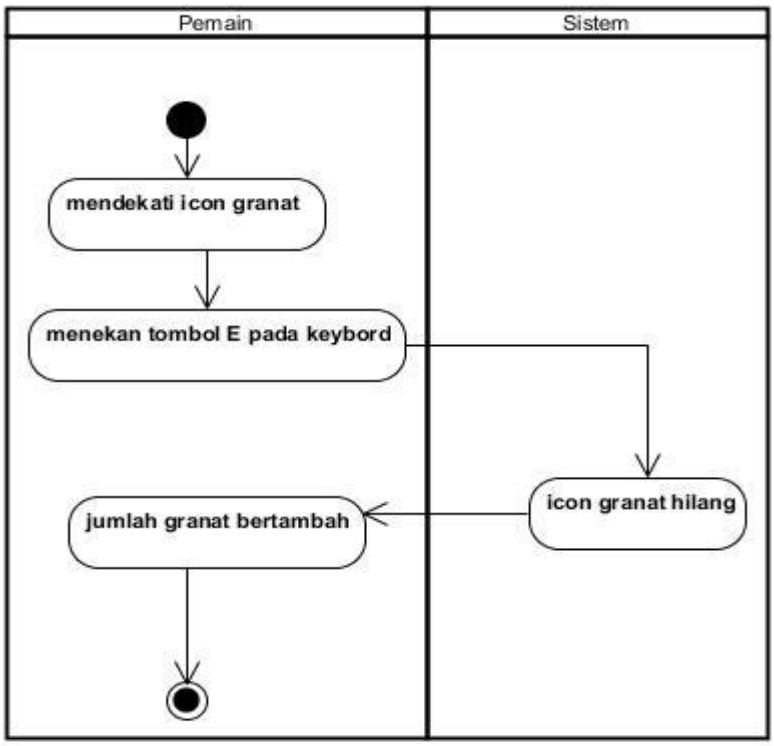
Gambar 3.29 Activity Diagram Proses Melempar Granat

Tabel 3.26 Penjelasan Proses *activity* Melempar Granat

Deskripsi	Karakter menggerakkan *scroll* pada *mouse*
Normal Flow	1. Pemain menahan tombol mouse sebelah kiri
	2. Setelaha kekuatan jarak lempar sesuai

	3. Pemain melepas tombol mouse yang di tahan tersebut. 4. Granat menuju target sasaran musuh 5. Granat meledak 6. Character damage musuh habis 7. Musuh mati

g. Activity Diagram Proses Mengambil Granat



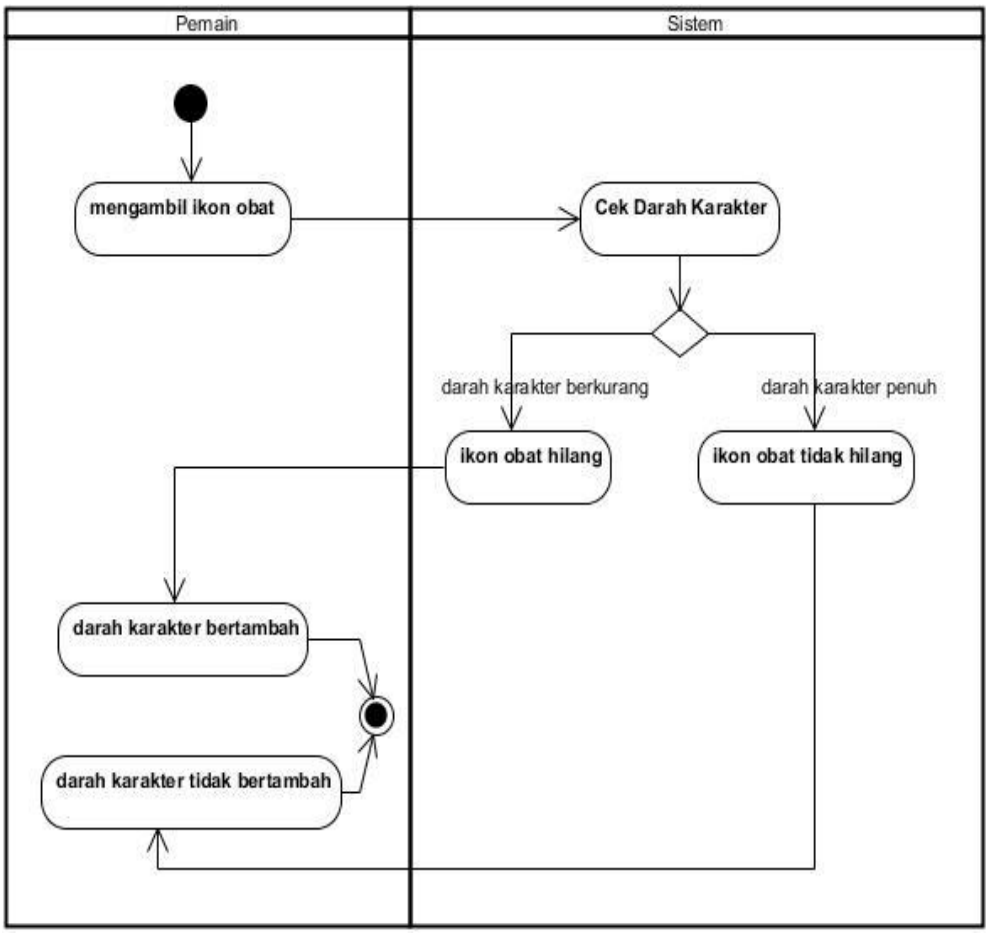
Gambar 3.30 Activity Diagram Proses mengambil Granat

Tabel 3.27 Penjelasan Proses *activity* Mengambil Granat

Deskripsi	Karakter menggerakkan *scroll* pada *mouse*
Normal Flow	1. Pemain mendekati ikon granat 2. Pemain menekan tombol E pada keybord

	3. Ikon granat hilang dari permainan 4. Jumlah granat pemain langsung bertambah 10.

h. *Activity Diagram* Proses Mengambil Obat



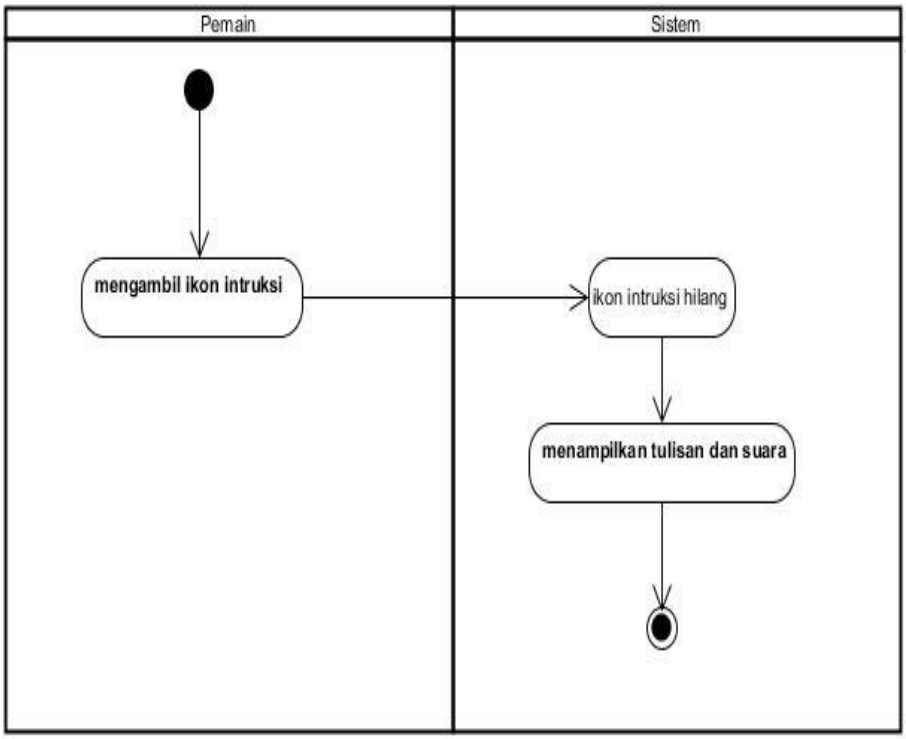
Gambar 3.31 *Activity Diagram* Proses mengambil Obat

Tabel 3.28 Penjelasan Proses *activity* Mengambil Obat

Deskripsi	Karakter mengambil Obat
Normal Flow	1. Pemain mengambil ikon obat. 2. Kondisi darah karakter di cek.

	3. Kondisi darah berkurang. 4. Ikon obat hilang. 5. Darah karakter bertambah. 6. Kondisi darah penuh. 7. Ikon obat tidak hilang. 8. Darah karakter tidak berubah.

i. *Activity Diagram* Proses Melihat Intruksi



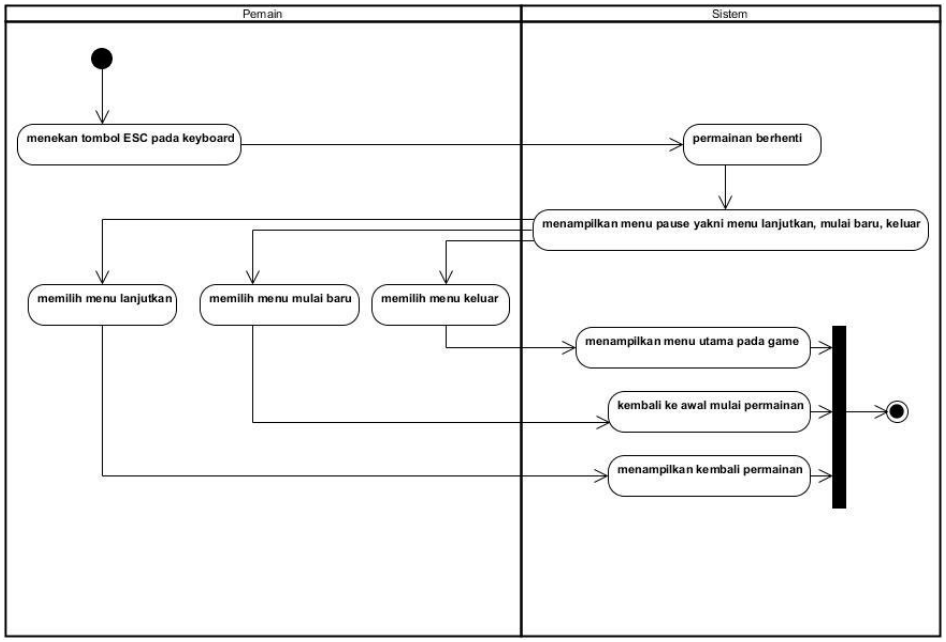
Gambar 3.32 *Activity diagram* Proses Melihat Intruksi

Tabel 3.29 Penjelasan Proses *activity* Mengambil Intruksi

Deskripsi	Karakter mengambil ikon intruksi
Normal Flow	1. Pemain mengambil ikon intruksi 2. Ikon intruksi hilang

	3. Sistem mengeluarkan suara dan tulisan

J. Activity Diagram Proses Memperhentikan Permainan



Gambar 3.33 Activity Diagram Proses Memperhentikan Permainan

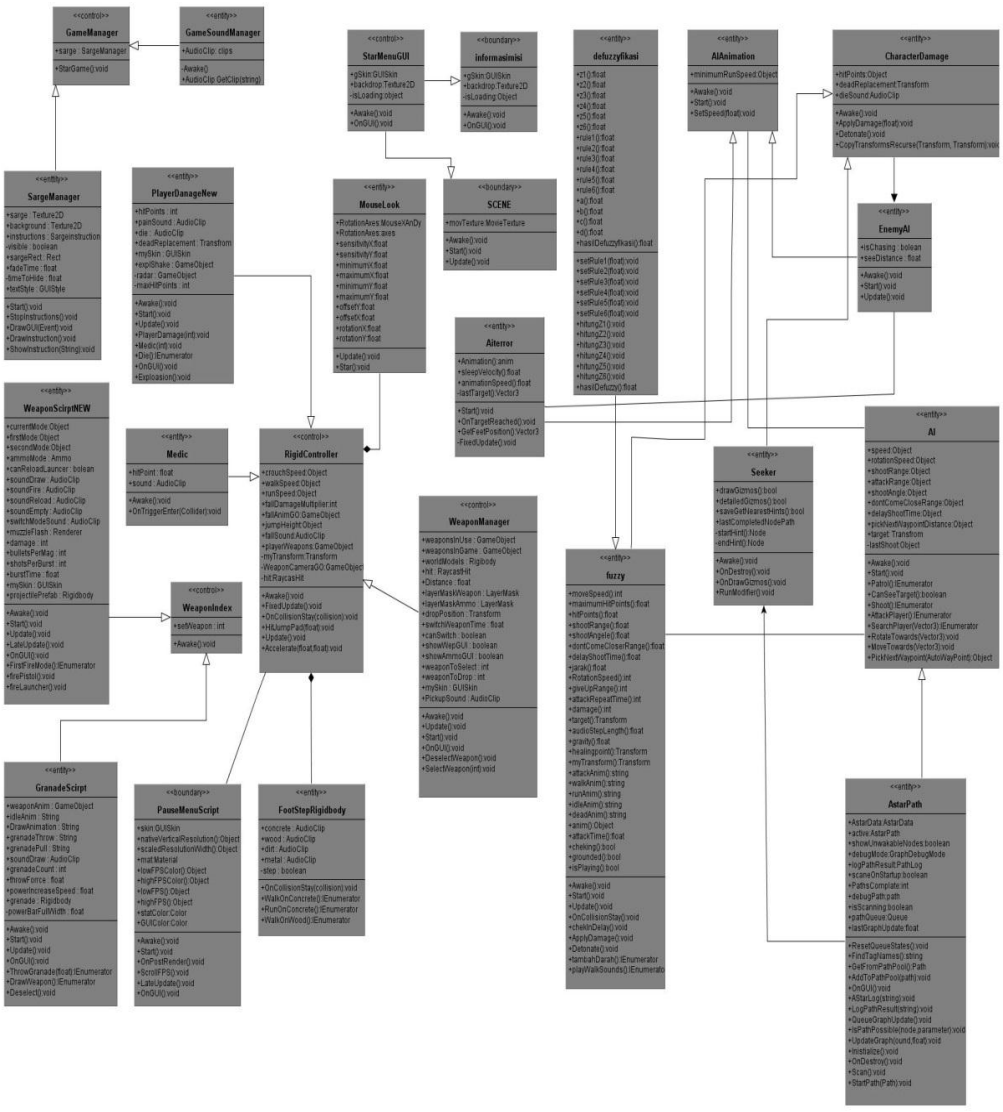
Tabel 3.30 Penjelasan Proses *activity* Memperhentikan Permainan

Deskripsi	Karakter mengambil ikon intruksi
Normal Flow	1. Pemain menekan tombol ESC pada keyboard 2. Permainan berhenti 3. Saat permainan berhenti ada 3 menu pilihan 4. Pilihan pertama menu lanjutkan 5. Permainan bias di lanjutkan kembali 6. Pelihan menu ke dua 7. Permainan di ulangi dari tahap awal

	8. Pilihan menu ketiga menu keluar 9. Permainan keluar dari area permainan dan kembali kemenu utama.

3.5.3 *Class Diagram*

Class diagram menggambarkan struktur sistem dari segi pendefinisian kelas-kelas yang dibuat untuk membangun sistem. Berikut adalah *class* diagram pada *game* FPS medan area

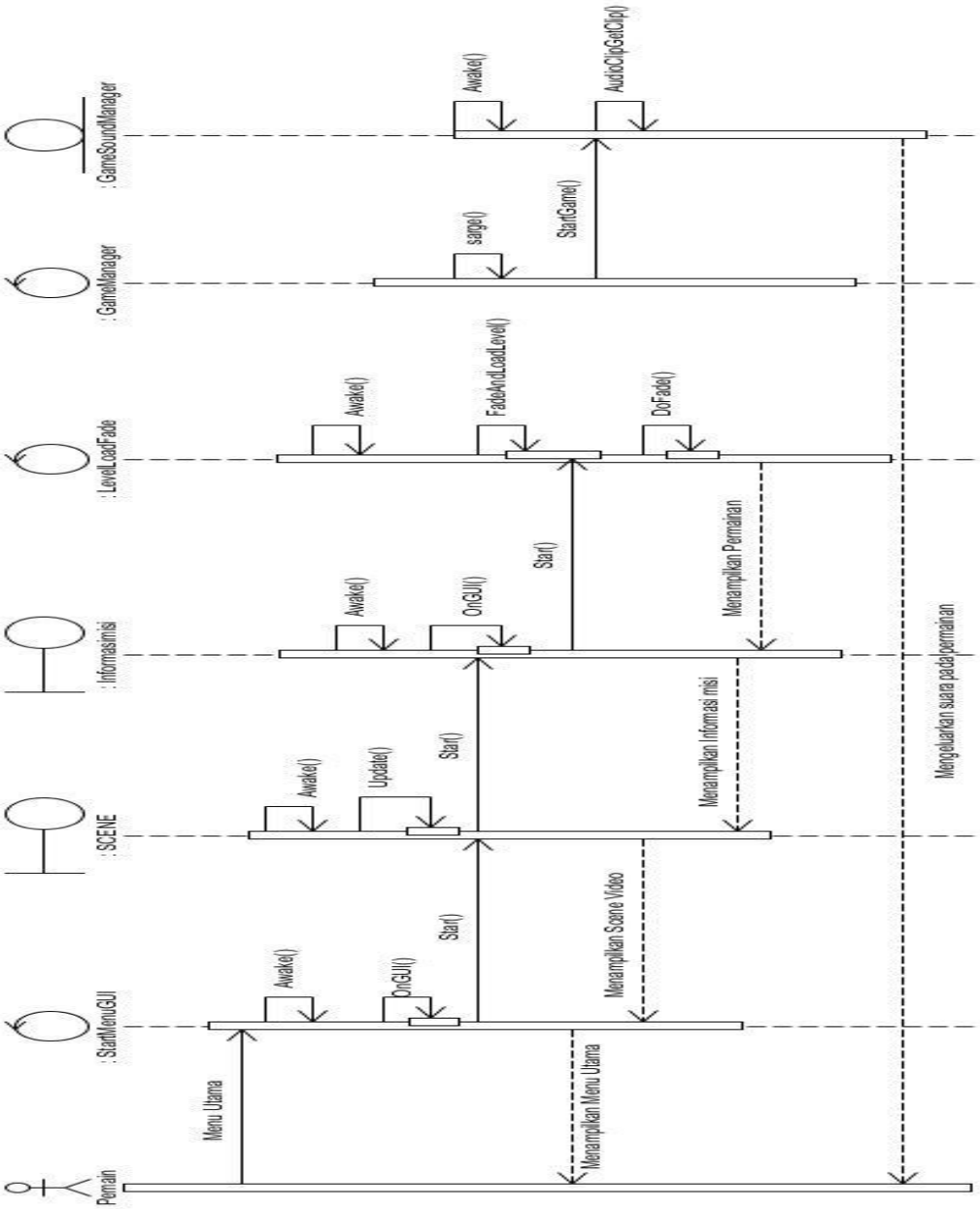


Gambar 3.34 *Class diagram* FPS 3D Sejarah Medan Area

3.5.4 *Sequential Diagram*

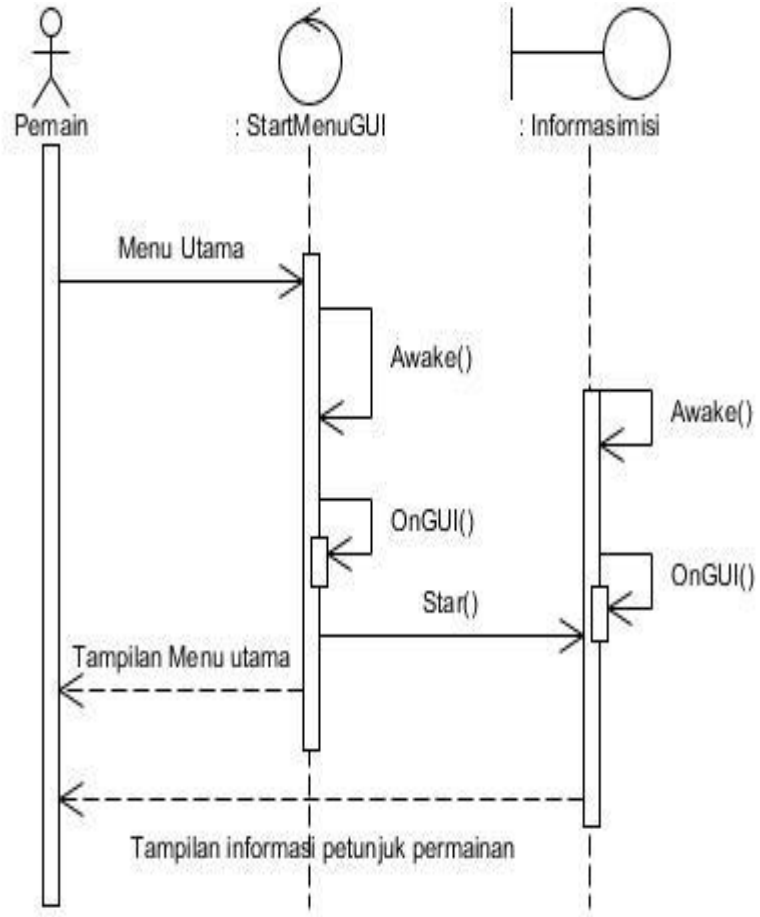
Sequencial diagram menggambarkan interaksi antar objek di dalam dan disekitar sistem berupa *message* yang digambarkan terhadap waktu. Berikut ini beberapa *sequencial* diagram yang terdapat pada *game* tersebut yaitu :

A. *Sequence Diagram* Mulai Permainan



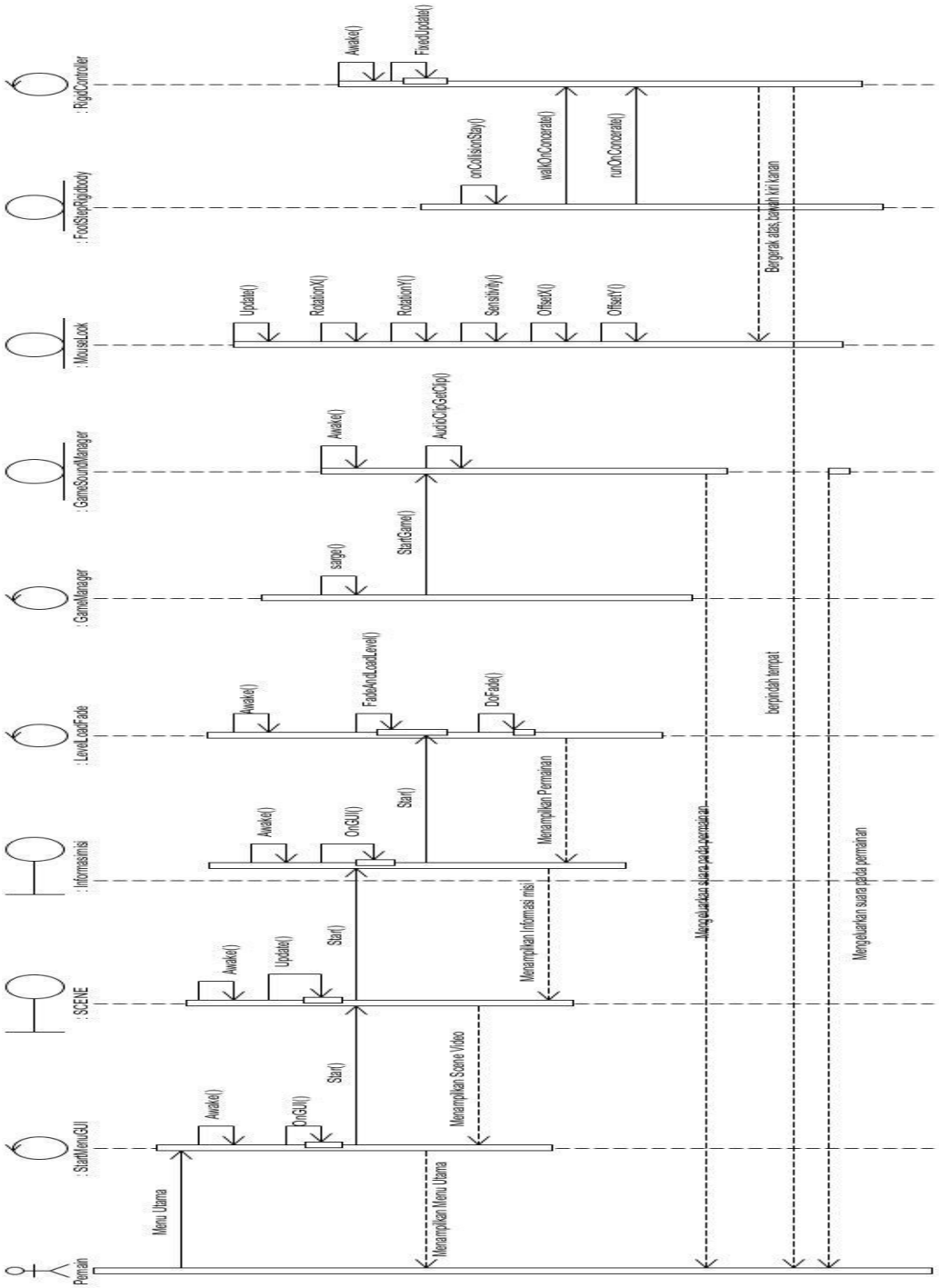
Gambar 3.35 *Sequence Diagram* Mulai Permainan

B. Sequence Diagram Menu Petunjuk



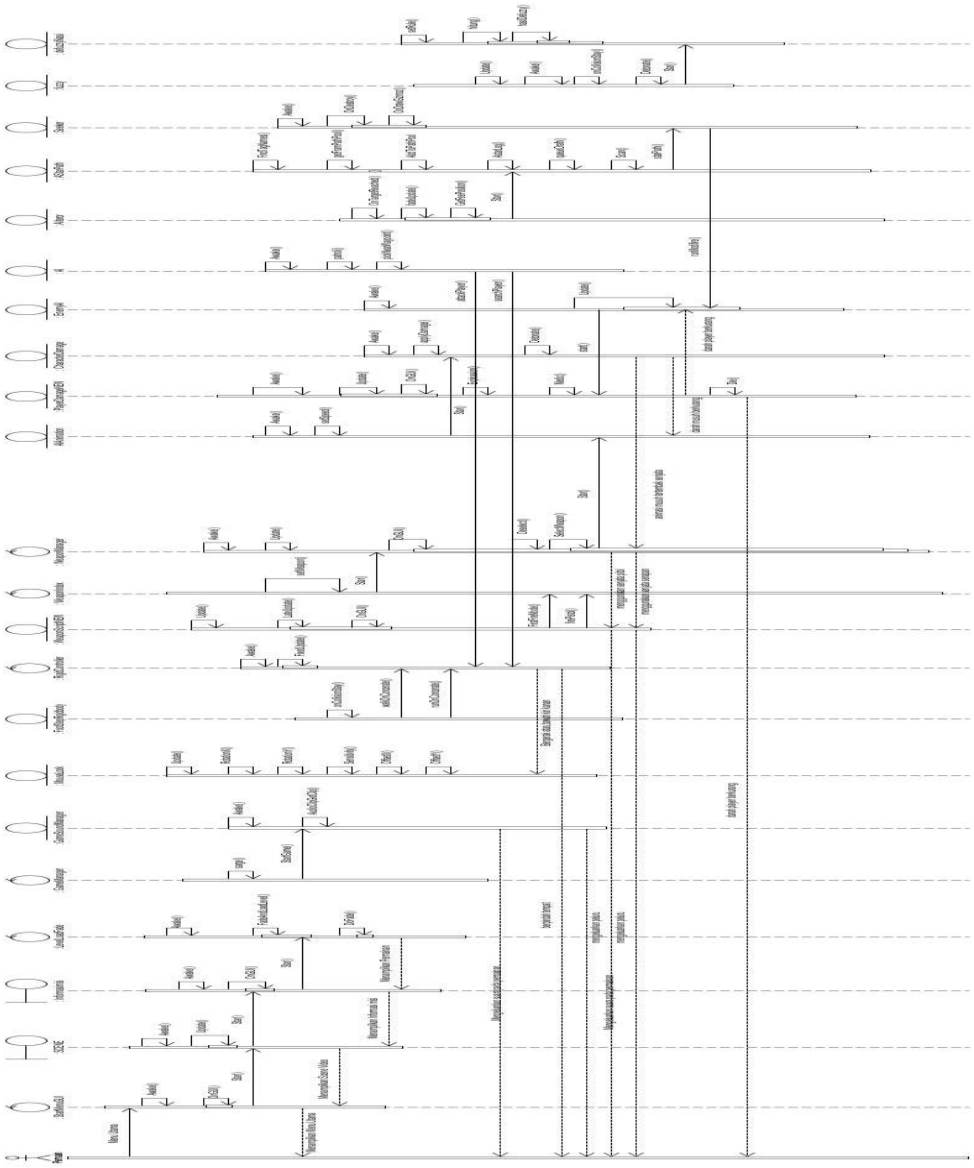
Gambar 3.36 *Sequence Diagram* Menu Petunjuk

C. *Sequence Diagram* Pergerakan Karakter



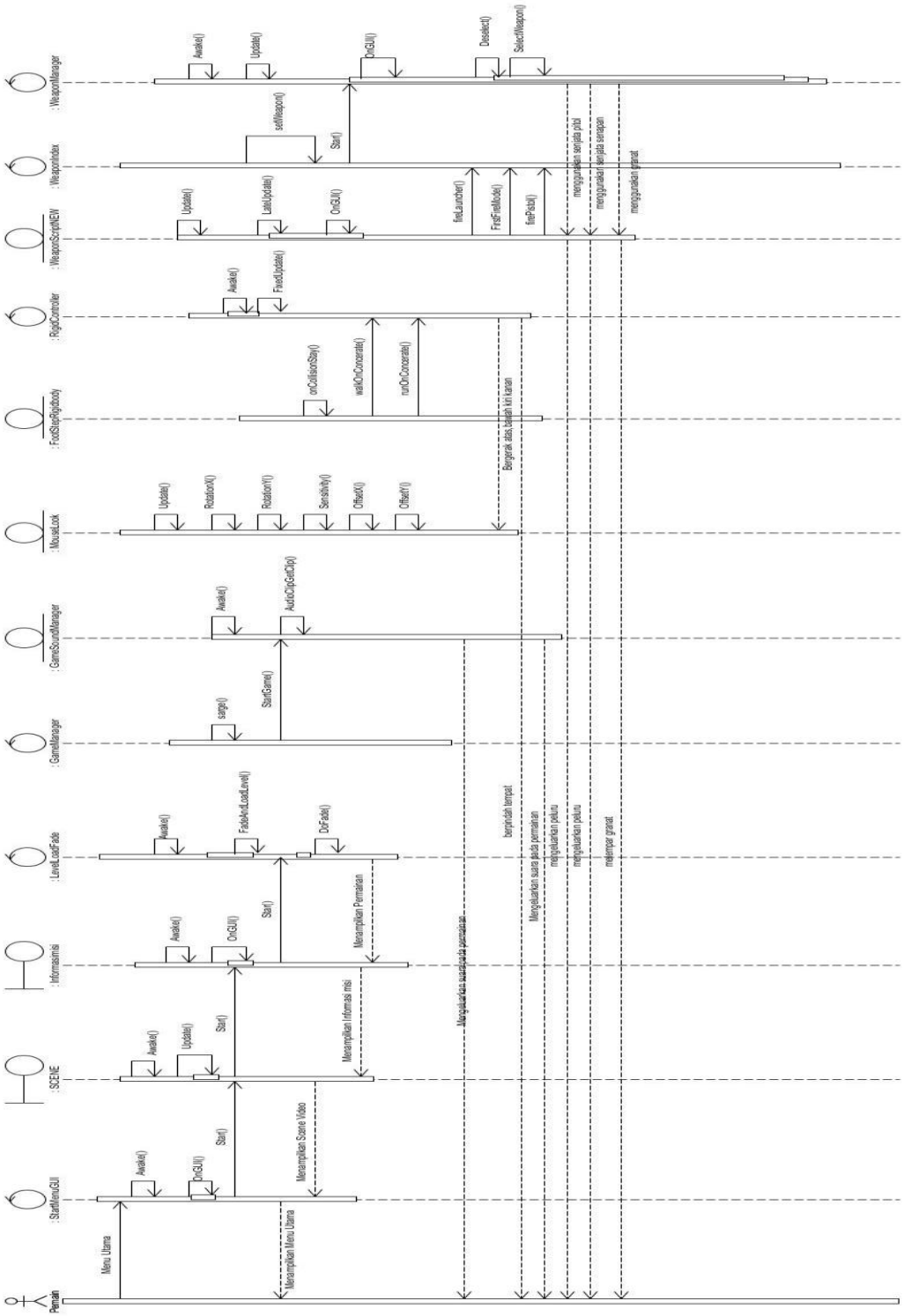
Gambar 3.37 *Sequence Diagram* Pergerakan Karakter

D. *Sequence Diagram* Menembak



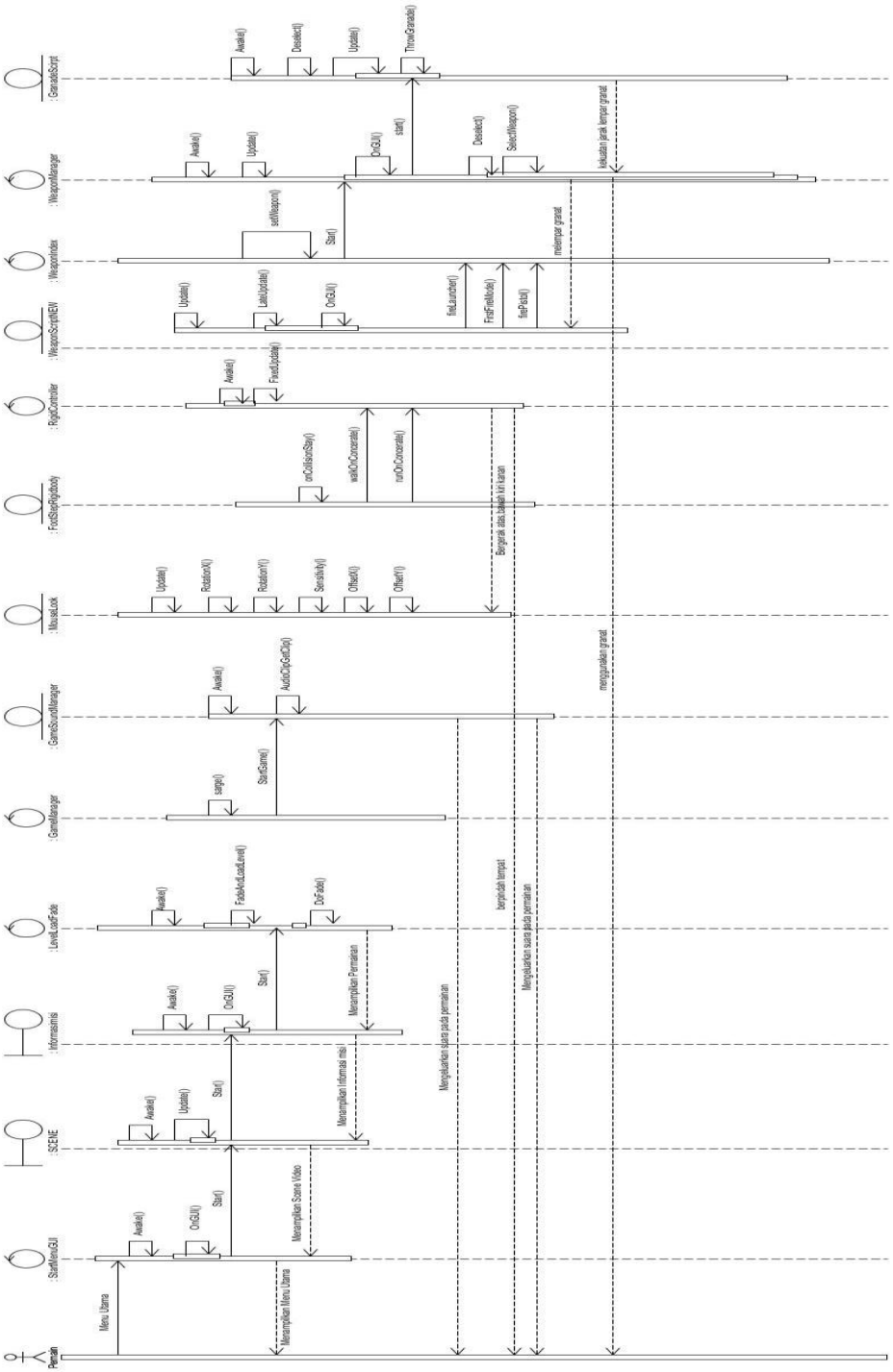
Gambar 3.38 *Sequence Diagram* Menembak

E. Sequence Diagram Menganti Senjata



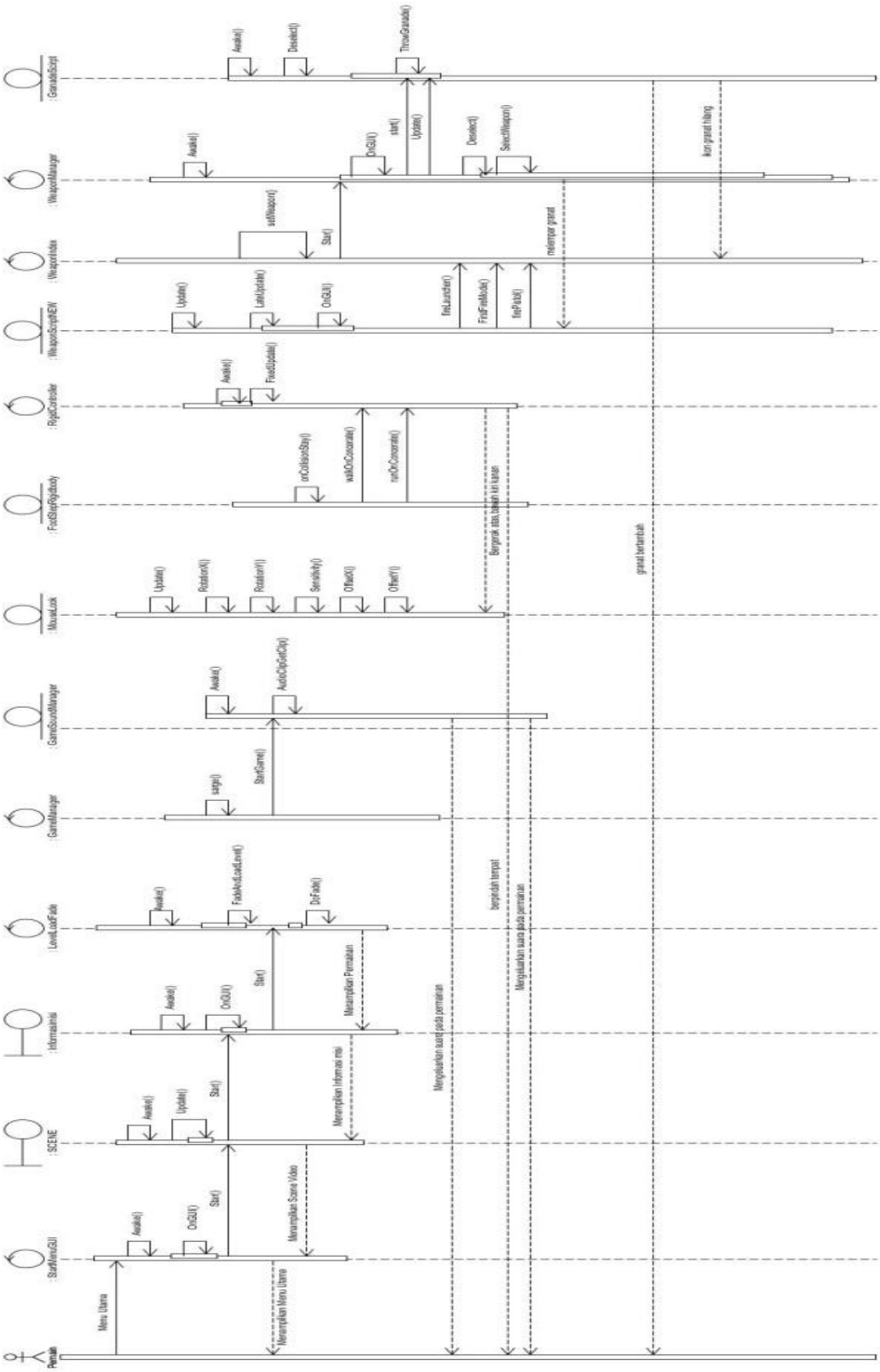
Gambar 3.39 Sequence Diagram Mengganti Senjata

F. Sequence Diagram Melempar Granat



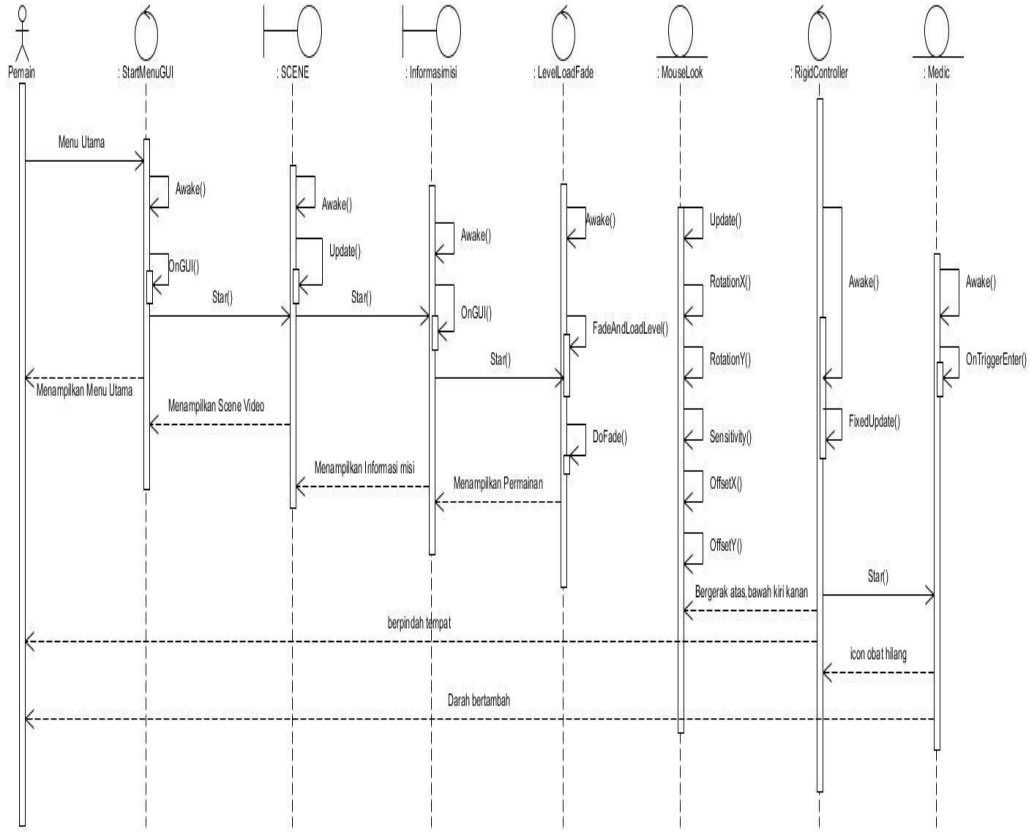
Gambar 3.40 Sequence Melempar Granat

G. Sequence Diagram Mengambil Granat



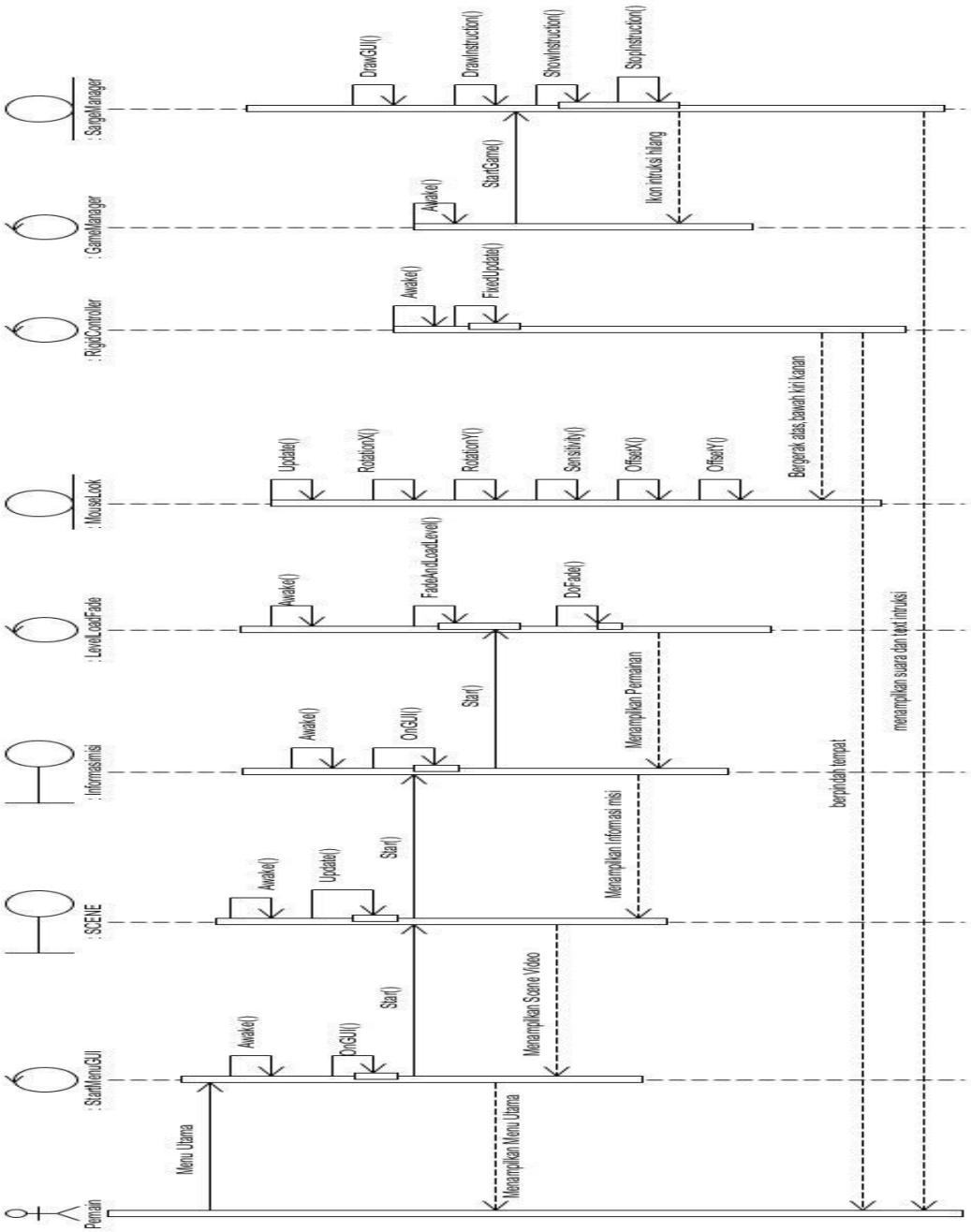
Gambar 3.41 Sequence Diagram Mengambil Granat

H. *Sequence Diagram* Mengambil Obat



Gambar 3.42 *Sequence Diagram* Mengambil Obat

I. *Sequence Diagram* Melihat Intruksi



Gambar 3.43 *Sequence Diagram* Melihat Intruksi

3.6 Perancangan Sistem

Perancangan adalah suatu bagian dari metodologi pengembangan suatu perangkat lunak yang dilakukan setelah tahapan untuk memberikan gambaran secara terperinci. Dalam tahap ini digambarkan rancangan sistem yang akan dibangun sebelum dilakukan pengkodean ke dalam suatu bahasa pemrograman.

3.6.1 Perancangan Komponen Permainan

Perancangan komponen permainan meliputi karakter dan *storyboard* yang didalamnya menjelaskan karakter beserta senjata apa saya yang dimiliki pemain maupun musuh (NPC) dan *storyboard* sebagai gambar perancangan pembuatan *game* Medan Area.

3.6.1.1 Karakter

Karakter utama dalam *game* ini bernama Naga Bonar seorang tentara yang dirancang untuk melawan para tentara Belanda. Naga Bonar dilengkapi dengan persenjataan berupa satu senjata berupa senapan dan satu senjata berupa Bambu runcing. Karena *game* ini merupakan *game* yang mempunyai *genre first person shooter* yang mempunyai sudut pandang orang pertama, maka karakter Naga Bonar seperti tubuh, kepala, kaki dan tangan tidak diperlihatkan tetapi hanya memperlihatkan bagian laras senjatanya saja. Properti yang dimiliki karakter utama dapat dilihat pada Tabel 3.19.

Tabel 3.31 Properties Karakter Utama

Nama	NagaBonar
Darah	500
Kemampuan	Berjalan, menembak
Senjata	Pistol, Senapan dan Granat

Senjata yang digunakan oleh Nago Bonar ada 3 macam, pertama adalah senjata senapan yang memiliki 50 peluru di dalam senjata dapat dilihat pada gambar 3.44, kedua adalah pistol yang mempunyai 12 peluru di dalam senjata,

seperti pada gambar 3.45 dan granat sebanyak 10 yang dapat dilihat pada Gambar 3.46 dan *properties* senjata pada karakter dapat dilihat pada tabel 3.42.



Gambar 3.44 Senapan



Gambar 3.45 Pistol



Gambar 3.46 Granat

Tabel 3.32 Karakteristik Senjata

Properti	Senapan	Pistol	Granat
Damage peluru senjata	35	10	50
Jarak peluru senjata	100	100	35
Reload peluru	3detik	0.5detik	-
Peluru	12/reload	50/reload	10

A. Karakter Musuh

Musuh yang terdapat pada *game* ini adalah tentara Belanda, Tentara Belanda ini memiliki satu senjata berupa laras panjang yang dapat dilihat pada Gambar 3.47



Gambar 3.47 Karakter utama musuh

B. Gardu Tulisan Fixed Boundarius Medan Area

Gardu yang bertuliskan *"Fixed Boundarius Medan Area"* ini merupakan gardu yang harus di hancurkan di dalam stage 2, karena gardu ini merupakan batas-batas yang di buat oleh pasukan Belanda dan sekutu. Dapat dilihat pada gambar dibawah ini :



Gambar 3.48 Gardu *Fixed Boundarius Medan Area*

3.6.1.2 Storyboard

Storyboard adalah sketsa gambar yang disusun berurutan sesuai dengan naskah, dengan menggunakan *storyboard* dapat menyampaikan ide cerita kepada orang lain dengan lebih mudah karena dapat dengan *storyboard* dapat mengimajinasikan khayalan yang sesuai dengan gambar-gambar yang ada sehingga dapat menghasilkan persepsi yang sama pada ide cerita yang dibuat.

Berikut beberapa *storyboard* dari *game first person shooter* 3D sejarah medan area

1. Tampilan *Storyboard* di *Stage* 1 bagian 1



Gambar 3.49 *Storyboard stage* 1.1 Awal mulai *game*

2. Tampilan *Storyboard* di *Stage* 1 bagian 2



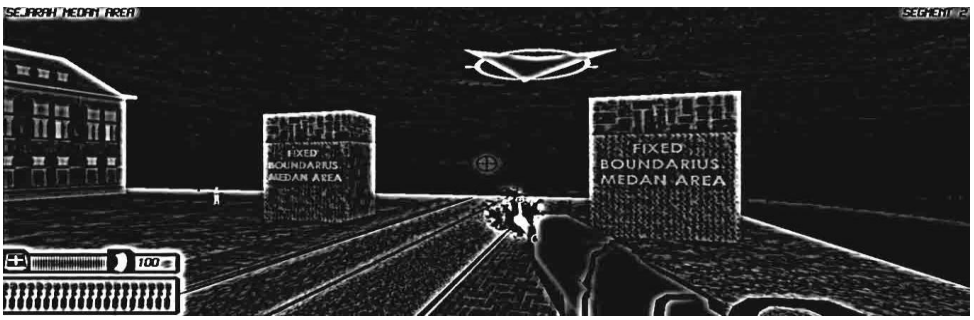
Gambar 3.50 *Storyboard stage* 1.2 Misi inseden di Hotel Depan Jalan Bali

3. Tampilan *Storyboard* di *Stage* 2 bagian 1



Gambar 3.51 *Storyboard stage* 2.1 Awal Mulai Stage 2

4. Tampilan *Storyboard* di *Stage* 2 bagian 2



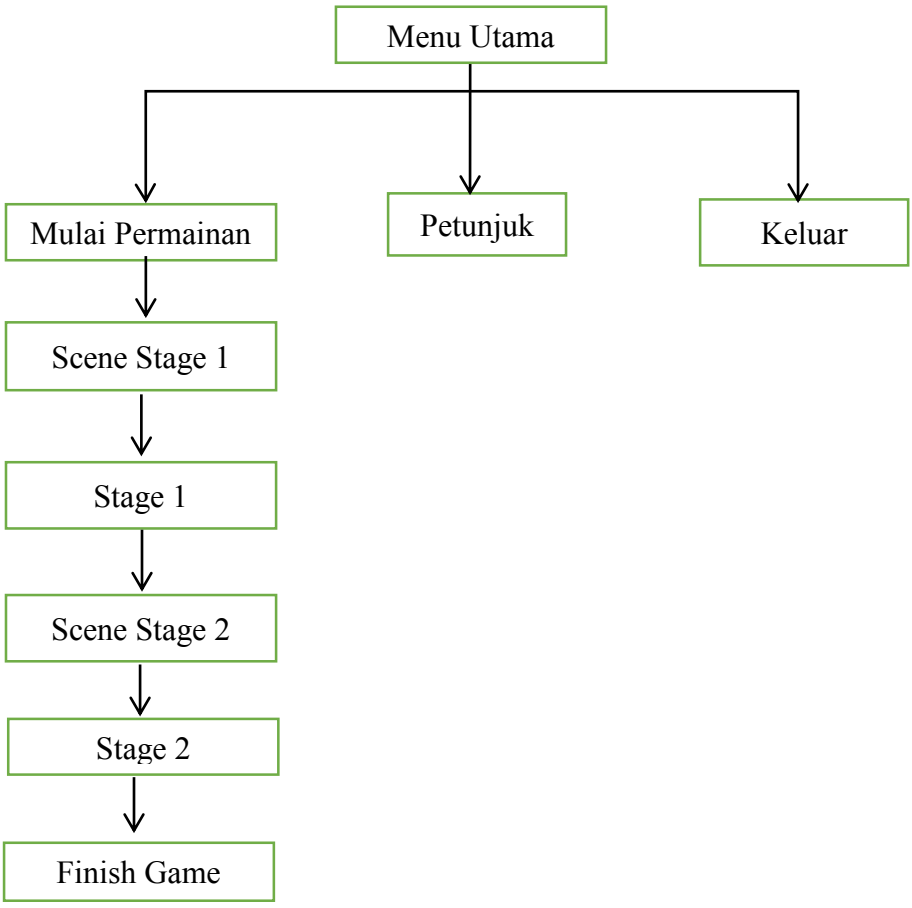
Gambar 3.52 *Storyboard stage* 2.2 Misi *Fixed Boundereus Medan Area*

3.6.2 Perancangan Arsitektur

Setelah melakukan perancangan data pada sistem yang dibangun, maka dilakukanlah perancangan arsitektur. Perancangan arsitektur ini meliputi beberapa perancangan diantaranya struktur menu, perancangan antarmuka.

3.6.2.1 Perancangan Struktur Menu

Perancangan struktur menu merupakan gambaran menu untuk menyediakan fungsi-fungsi yang akan digunakan dalam *game FPS* Sejarah Medan.



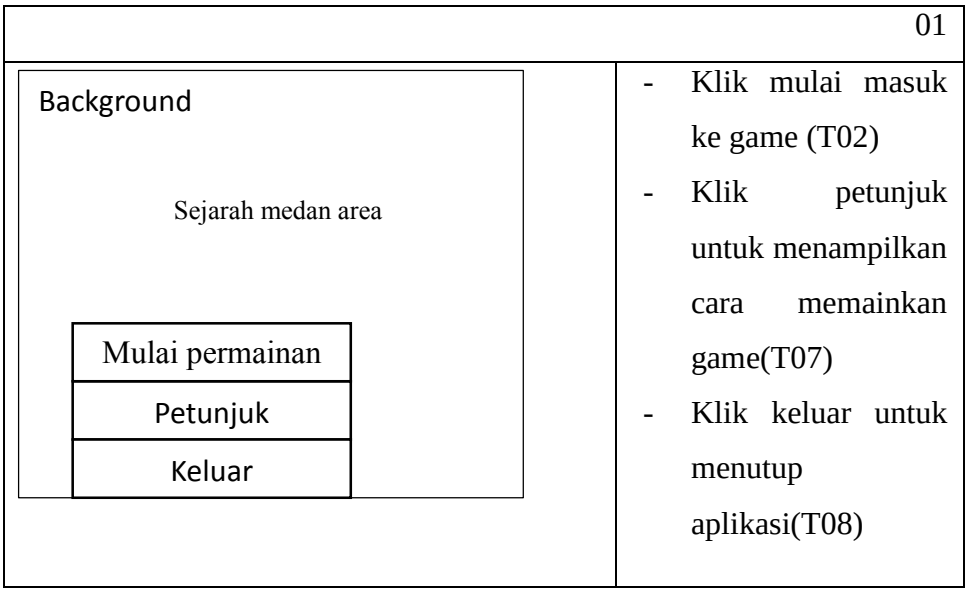
Gambar 3.53 Struktur menu utama

3.6.3 Perancangan Antarmuka

Perancangan antarmuka dibutuhkan untuk mawakili keadaan sebenarnya dari aplikasi yang akan dibangun. Berikut ini beberapa perancangan anatarmuka dari aplikasi yang akan dibangun yaitu :

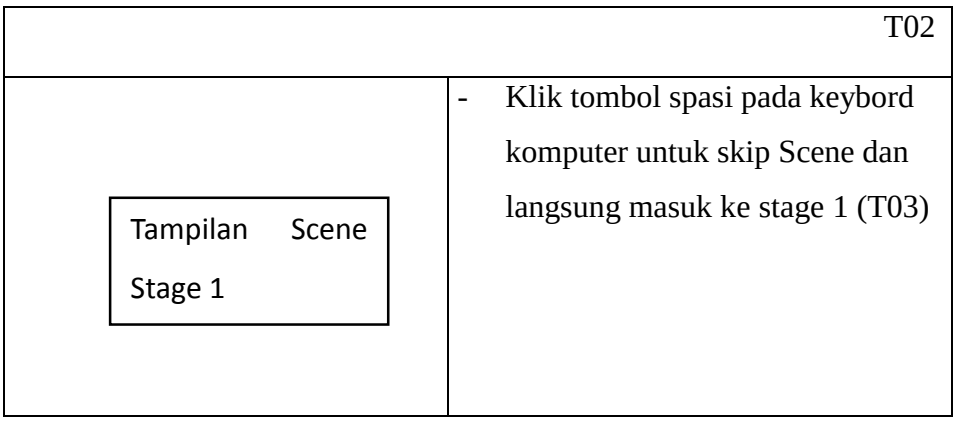
1. Menu Utama

Berikut dapat dilihat perancangan antarmuka menu utama dari pembangunan aplikasi *game* sejarah medan area.



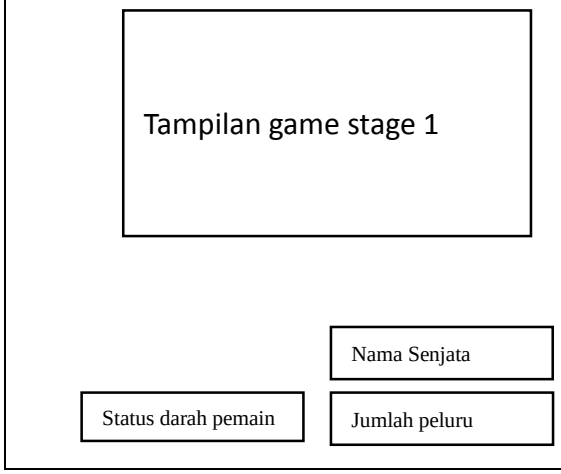
Gambar 3.54 Tampilan Menu Utama

2. Scene *stage* 1 Permainan



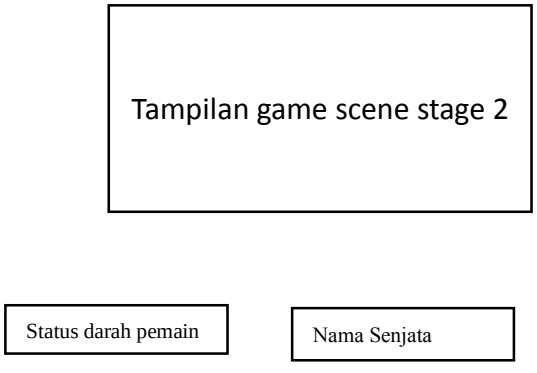
Gambar 3.55 Scene *Stage* 1 Permainan

3. *Stage* 1

	T03
	- Klik Tombol ESC untuk Pause game (T06) - Klik Tombol 1 atau 2 pada keybord untuk mengganti senjata atau scroll pada mouse.

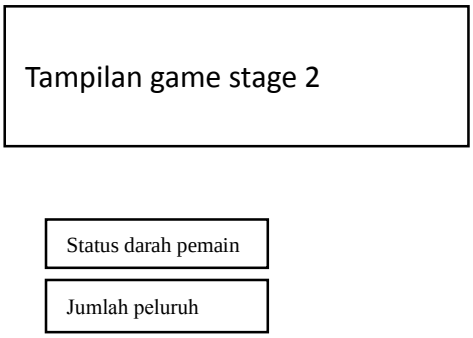
Gambar 3.56 *Stage* 1 Permainan

4. Scene *Stage* 2

	T04
	- Klik tombol spasi pada keybord komputer untuk skip Scene dan langsung masuk ke stage 1 (T05)

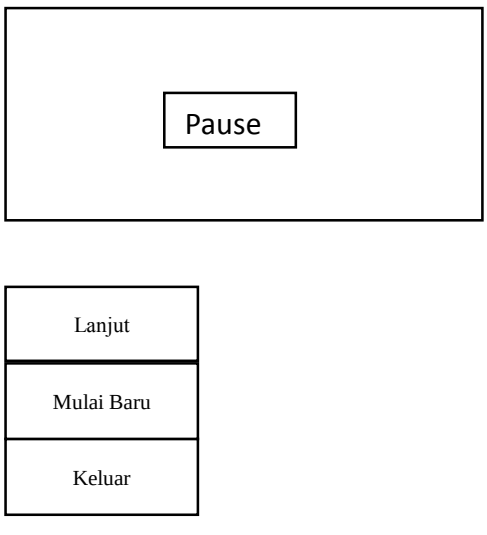
Gambar 3.57 *Scene stage* 2 Permainan

5. Stage 2

T05	
	- Klik Tombol ESC untuk Pause game (T06) - Klik Tombol 1 atau 2 pada Keybord untuk mengganti senjata

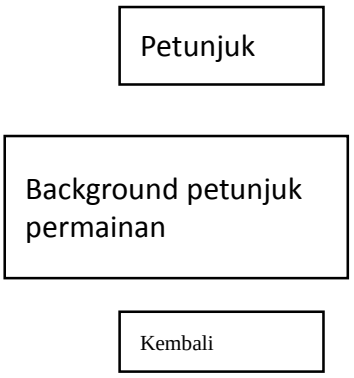
Gambar 3.58 *Stage* 2 Permainan

6. Menu *Pause*

T06	
	- Klik Lanjutkan masuk ke T02 atau T04 (tergantung di pause di stage yang dimainkan) untuk melanjutkan permainan - Mulai Baru masuk ke T03 - Klik keluar game masuk ke T01

Gambar 3.59 Menu *Pause*

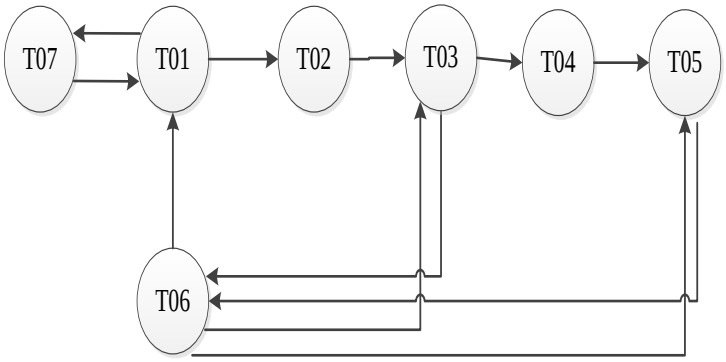
7. Menu Petunjuk

	T07
	- Klik petunjuk yang terdapat di halaman menu utama(T01) - Klik kembali untuk kembali ke menu utama(T01)

Gambar 3.60 Menu Petunjuk

3.6.3.1 Jaringan Semantik

Struktur aplikasi ini dibuat secara moduler, yaitu program dipecah menjadi modul-modul kecil yang mudah dibuat, mudah dites, dan mudah dimodifikasi. Dalam pembuatan *game* ini menggunakan alat bantu bagan struktural yang disebut juga jaringan semantik.



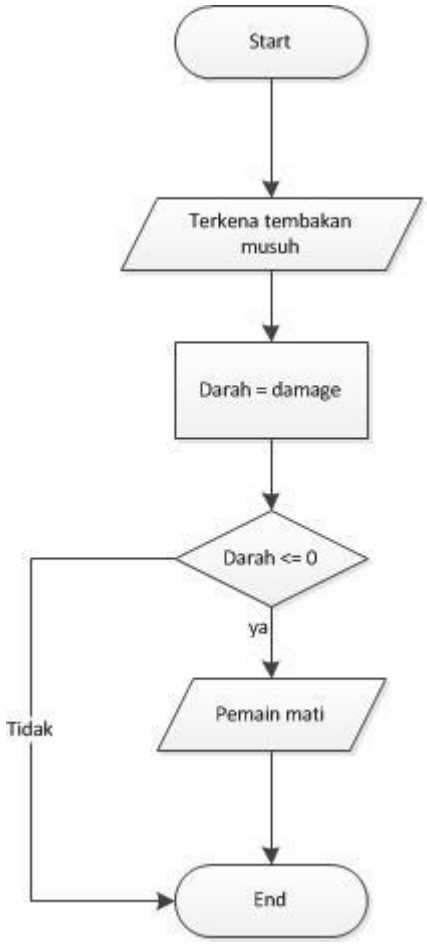
Gambar 3.61 Jaringan Semantik

3.6.4 Perancangan Method

Perancangan *method* merupakan perancangan yang berfungsi untuk mendeskripsikan *method-method* yang berada di dalam aplikasi. *Method* dapat dipanggil dengan menyertakan variabel, baik hanya satu variabel, banyak variabel atau bahkan tidak ada sama sekali. Adapun *method-method* yang terdapat dalam *game* FPS 3D sejarah Medan Area yang akan dibangun dijelaskan melalui *flowchart* sebagai berikut.

3.6.4.1 *Method ApplyDamage*

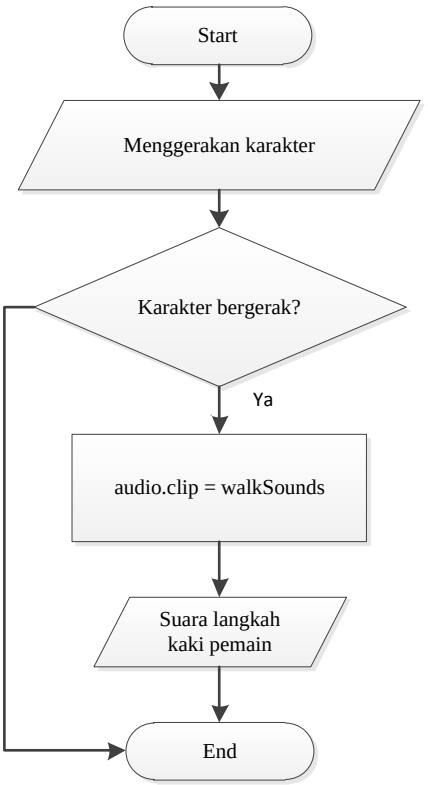
Method ini akan melakukan pengecekan terhadap setiap tembakan / *damage* dari musuh yang mengenai pemain yang dapat dilihat pada gambar 3.51.



Gambar 3.62 ***Flowchart Method ApplyDamage***

3.6.4.2 *Method PlayStepSounds*

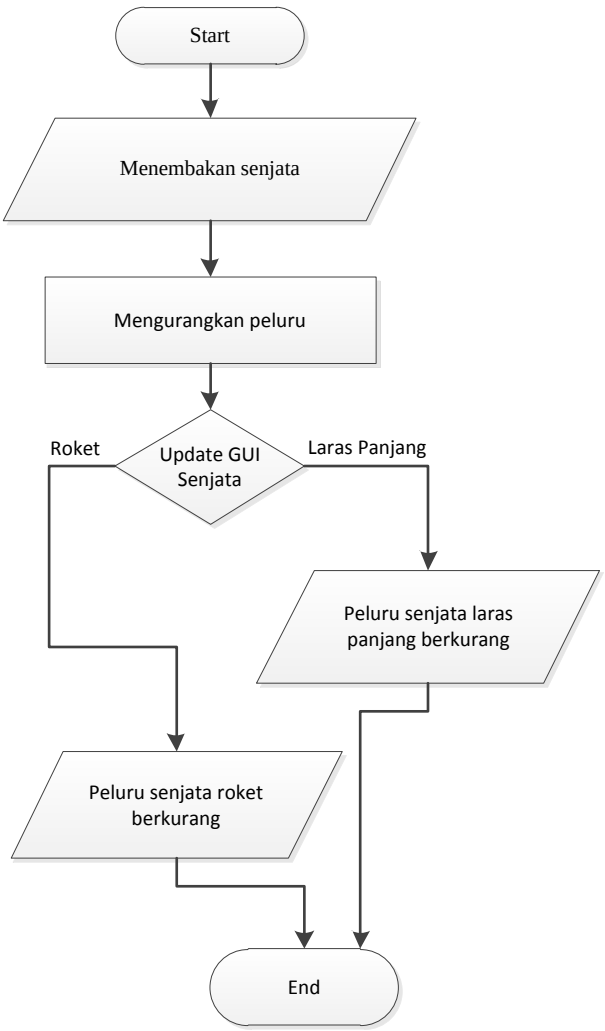
Method ini akan memberikan efek suara langkah kaki terhadap karakter ketika pemain menggerakan karakter yang dapat dilihat pada gambar 3.58.



Gambar 3.63 ***Flowchart Method PlayStepSounds***

3.6.4.3 Method UpdateGUI

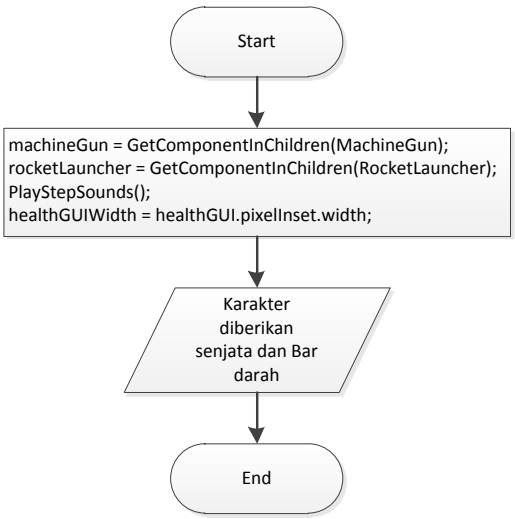
Method ini akan melakukan perubahan terhadap GUI berupa pengurangan peluru ketika pemain menembak, baik menembak menggunakan senjata laras panjang ataupun senjata basoka yang dapat dilihat pada gambar 3.59



Gambar 3.64 ***Flowchart Method UpdateGUI***

3.6.4.4 Method Awake

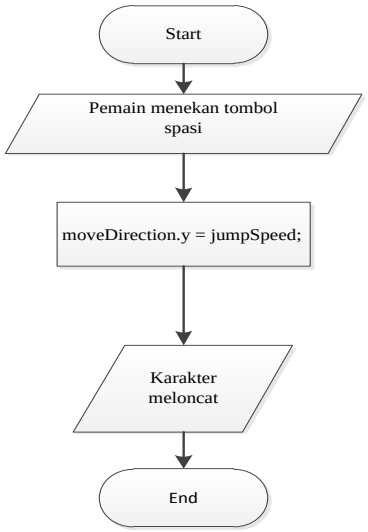
Method ini akan memberikan senjata dan GUI darah / health terhadap karakter pada saat permaian dimulai yang dapat dilihat pada gambar 3.60.



Gambar 3.65 ***Flowchart Method Awake***

3.6.4.5 Method FixedUpdate

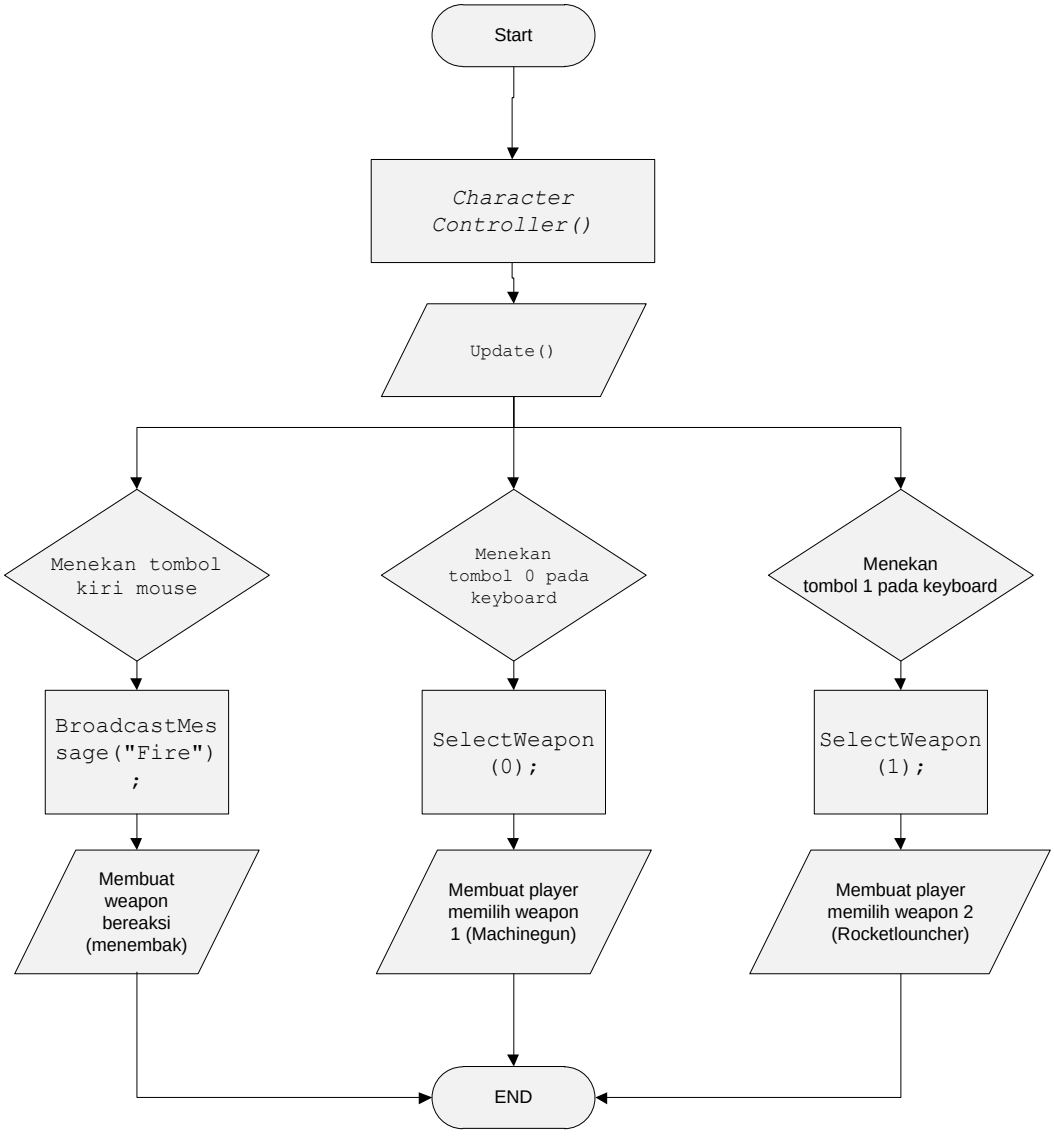
Method ini akan melakukan update berupa gerakan meloncat terhadap karakter yang diawali dari inputan yang dilakukan pemain yang dapat dilihat pada gambar 3.61



Gambar 3.61 ***Flowchart Method Fixed Update***

3.6.4.6 *Method Player Weapon*

Method Player Weapon digunakan untuk menampung semua atribut-atribut senjata dari yang ada pada *FPS Player.* Untuk lebih jelasnya dapat dilihat pada gambar *flowchart* dibawah ini:



Gambar 3.62 *Flowchart Method PlayerWeapon*