BAB 3
ANALISIS DAN PERANCANGAN SISTEM

3.1 Analisis Sistem

Pada bagian analisis sistem ini, akan dilakukan berbagai macam analisis yang berhubungan dengan *game* yang akan dibangun, mulai dari analisis terhadap *game* sejenis analisis berbagai kebutuhan baik fungsional maupun non-fungsional.

3.1.1 Analisis Masalah

Dalam hal ini diperlukan sebuah analisis yang dilakukan terhadap aplikasi permainan lain yang sejenis untuk menentukan dimana letak kekurangan aplikasi permainan tersebut, sehingga dapat dikembangkan dengan membuat aplikasi permainan baru agar semakin menarik dibandingkan dengan aplikasi permainan yang dianalisis. Berdasarkan analisis yang dilakukan terhadap *game* sejenis yang berjudul *Epic Adventure Times Revolt in Ogre Town* bahwa di dalam game tersebut masih terdapat kelemahan, diantaranya yaitu perilaku terhadap musuh yang masih bersifat pasif dalam bergerak maupun menyerang, tidak memiliki level sehingga *game* terkesan monoton, serta game hanya memiliki misi menghindari dan membunuh musuh saja. Pada analisis game yang lain yang berjudul *"Colonial Wars"* yang menceritakan tentang penjajahan pulau terdapat beberapa kelemahan juga, yaitu kurangnya perlawanan musuh dan rintangan-rintangan dalam penyerangan, sehingga membuat pemain bosan memainkannya.

3.1.2 Analisis *Game* Sejenis

Dalam hal ini diperlukan sebuah analisis yang dilakukan terhadap aplikasi permainan lain yang sejenis untuk menentukan dimana letak kekurangan aplikasi permainan tersebut, sehingga mendapatkan konsep permainan baru dan

perbandingan dari *game* yang telah ada. Aplikasi permainan sejenis yang akan dianalisis adalah *Epic Adventure Times Revolt in Ogre Town* dan *Colonial Wars*.

A. *Epic Adventure Times Revolt in Ogre Town*

a. Pengenalan

Game Epic Adventure Times Revolt in Ogre Town adalah *game* yang *bergenre adventure*. *Game* ini adalah *game* 2D (dua dimensi) dimana pemain akan menyerbu kastil untuk memulihkan kekacauan dan kebebasan bagi Kota Ogre.

b. *Screen Capture* Respon Pengguna

Berdasarkan hasil penilaian dari sejumlah pengguna terhadap game tersebut dapat diketahui beberapa kelemahan dalam game ini. untuk hasil komentar para pengguna dapat dilihat pada gambar 3.1 :



Gambar 3.1 *Screen Capture* Responden *Game* Epic [2]

c. *Gameplay*

Epic Adventure Times Revolt in Ogre Town menawarkan sebuah permainan dimana dalam *game* tersebut tidak ada level tetapi terdapat 5 pintu yang bebas untuk dilalui, 3 pintu merupakan area permainan,1 pintu bagaimana cara bermain dan 1 pintu untuk melihat point tertinggi, di setiap pintu pemain akan berperan berbeda-beda, dipintu pertama pemain akan berperan sebagai *Swordsman*, dipintu kedua berperan sebagai *Riflmen* dan dipintu ketiga berperan sebagai *Magician* dimana pemain diharuskan mengalahkan atau melewati musuh lalu menemukan pintu dan masuk ke pintu untuk keluar dari permainan di pintu tersebut dan memilih pintu lagi untuk permainan selanjutnya.

d. Elemen *Game*

Elemen permainan yang terdapat dalam permainan game *Epic Adventure Times Revolt in Ogre Town* diantaranya adalah sebagai berikut:

1. Arena Permainan

   Pada permainan *game Epic Adventure Times Revolt in Ogre Town* arena permainannya adalah di sebuah kastil yang dipenuhi banyak rintangan. Terdapat rintangan berbentuk musuh-musuh yang menghadang dan terdapat ranjau. Diarena permainan inilah pemain harus meloloskan diri dari kejaran musuh dan dari ranjau-ranjau yang terdapat dalam permainan. Berikut area permainan tiap pintu nya :



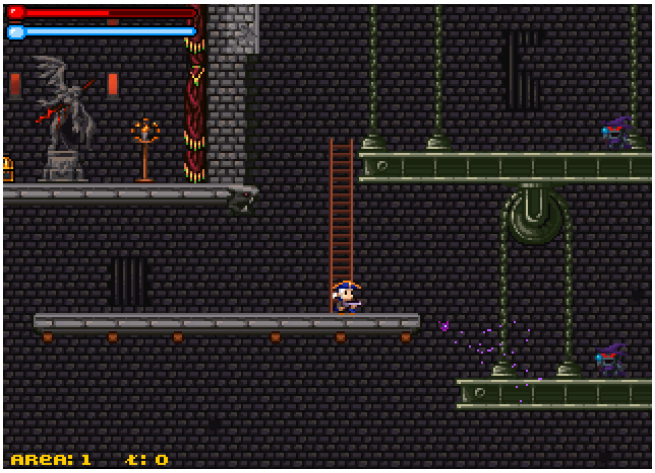
Gambar 3.2 Tampilan cara bermain di pintu pertama [2]

Di pintu pertama pemain akan diperlihatkan petunjuk permainan, dan elemen-elemen yang ada pada *game.*



Gambar 3.3 Tampilan area permainan pada pintu 2 [2]

Di pintu kedua, pemain mulai melaksanakan misi yaitu melewati dan membunuh musuh yang ada dalam permainan, pemain berperan sebagai *Swordsman* dan memiliki pedang sebagai senjatanya.



Gambar 3.4 Tampilan area permainan pada pintu 3 [2]

Di pintu ketiga, pemain berperan sebagai *Riflemen* dan memiliki pistol sebagai senjatanya. Misi level ini sama seperti level sebelumnya yaitu melewati dan mengalahkan musuh.



Gambar 3.5 Tampilan area permainan pada pintu 4 [2]

Di pintu keempat, pemain melaksanakan misi yaitu melewati dan mengalahkan musuh seperti di level sebelumnya, di level ini pemain berperan sebagai *Magician* dan memiliki bola *magic* sebagai senjatanya.



Gambar 3.6 Tampilan *Score* tertinggi pada pintu 5 [2]

Pada pintu kelima, pemain bisa melihat *score* tertinggi dari yang memainkan permainan ini.

2. Karakter pemain

Karakter pemain merupakan entitas utama yang akan dikendalikan oleh pengguna dalam arena permainan ini. Pemain harus menemukan pintu untuk melanjutkan ke pintu selanjutnya, disana juga pemain dapat mengambil item-item untuk menambah *health point* (batas kehidupan).Selain itu pemain juga harus menghindari mush dan ranjau-ranjau yang terdapat dalam permainan. Berikut nama dan keterangan pemain di tiap pintunya :

Tabel 3.1 Karakter Pemain dalam *game Epic Adventure*

No	Nama	Gambar	Keterangan
1	*Swordsman*		Ini adalah karakter utama di pintu kedua dikendalikan oleh pemain melalui inputan dari *keyboard* dan memiliki pedang sebagai senjatanya.
2	*Riflemen*		Ini adalah karakter utama di pintu ketiga dikendalikan oleh pemain

			melalui inputan dari *keyboard* dan memiliki pistol sebagai senjatanya.
3	*Magician*		Ini adalah karakter utama di pintu keempat dikendalikan oleh pemain melalui inputan dari *keyboard* dan memiliki bola *magic* sebagai senjatanya.

3. Karakter musuh

Karakter musuh dalam permainan *Epic Adventure Times Revolt in Ogre Town* adalah entitas yang bergerak pasif. Apabila musuh menyentuh atau menembak pemain maka darah pemain akan berkurang. Musuh terdiri dari beberapa jenis diantaranya angkatan besar paku, sejenis *bubble*, kepiting, tembok yang mengeluarkan api, jebakan, *bomb* , dan pasukan api dimana kemampuan musuh ini berbeda-beda.

Tabel 3.2 Karakter Musuh dalam *game Epic Adventure*

No	Nama	Gambar	Keterangan
1	Angkatan Besar Paku		Ini adalah karakter musuh yang bergerak pasif dan menembakan bola *magic*, apabila bersentuhan dan terkena tembakannya maka batas kehidupan karakter utama akan berkurang
2	Pasukan Api		Ini adalah karakter musuh yang berbadan penuh dengan api dan mengeluarkan api sebagai senjata utamanya, apabila bersentuhan dan terkena tembakannya maka batas kehidupan karakter utama akan

			berkurang.
3	Kepiting		Ini adalah karakter musuh yang bergerak loncat-loncat untuk menghalangi gerak karakter utama,apabila bersentuhan maka batas kehidupan karakter utama akan berkurang.
4	*Bubble*		Ini adalah karakter musuh yang mengeluarkan cairan dari atas, apabila mengenai karakter utama maka batas kehidupannya akan berkurang.
5	Jebakan		Bagian dari area permainan yang tidak boleh dilewati pemain, apabila krakter utama bersentuhan dengan jebakan tersebut batas kehidupannya akan langsung habis dan permainan selesai
6	*Bomb*		Jebakan ini berfungsi apabila pemain berjalan di atasnya, secara otomatis *bomb* tersebut akan jatuh dan mengurangi batas kehidupan karakter utama apabila terkena *bomb* tersebut.

4. *Life point*

Life Point adalah entitas yang berisi *health point* (batas kehidupan) yang dimiliki oleh tiap karakter pemain dan karakter musuh. *Life point* pada tiap karakter berbeda-beda. *Life point* juga menjadi entitas terpenting dalam permainan, karena menentukan tiap karakter dapat bertahan dalam arena permainan.

Tabel 3.3 ***Life Point Game Epic Adventure***

No	Nama	Gambar	Keterangan
1	Minuman		Ini adalah item yang digunakan untuk menambah darah pemain.
2	*Gold*		Ini adalah item yang digunakan untuk menambah darah pemain hingga penuh dan menambah point permainan.

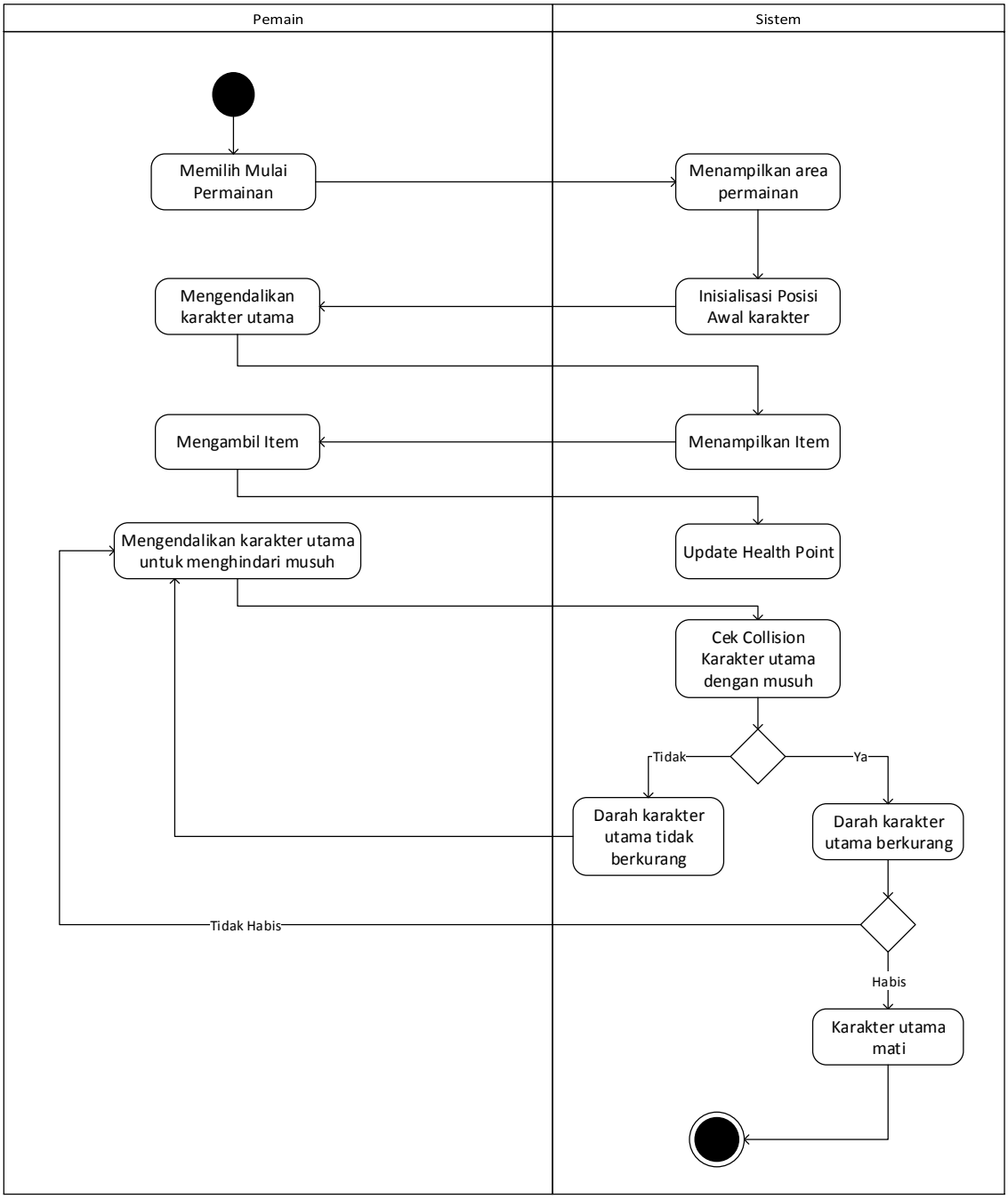
e. Skenario *Game*

Skenario *game* yang terdapat *Epic Adventure Times Revolt in Ogre Town* diantaranya adalah sebagai berikut:

Alur permainan pada game *Epic Adventure Times Revolt in Ogre Town* :

a. Pemain berusaha untuk menghindari musuh disebuah kastil dan mencari pintu keluar dari kastil tersebut.
b. Pemain mengambil item-item untuk menambah *health point* (batas kehidupan).
c. Pemain harus menghindar dari musuh yang berada di arena permainan.
d. Apabila pemain mengenai musuh, maka darah akan berkurang dan apabila habis maka permainan akan berakhir.

Adapun alur sistem dari permainan *Epic Adventure Times Revolt in Ogre Town* digambarkan melalui *activity diagram* , dapat dilihat pada gambar 3.7



Gambar 3.7 Alur Permainan *Epic Adventure Times Revolt in Ogre Town*

Adapun misi yang harus dicapai pemain yaitu pemain harus keluar dari kastil dengan mengalahkan atau menghindari lawan di dalam area

permainan, Pemain menyusuri semua jalan yang ada dalam permainan dan menemukan pintu sebagai jalan keluar dari kastil tersebut.

B. Colonial Wars

a. Pengenalan

Colonial Wars menawarkan sebuah permainan dimana dalam *game* tersebut pemain harus menguasai pulau lainnya untuk menuju ke level selanjutnya, pada awalnya pemain hanya memiliki satu pulau yang sudah terdapat pesawat, kapal laut dan perahu untuk menyerang pulau yang lain, setelah pulau yang lain berhasil direbut maka pemain bisa memakai pasukan di pulau tersebut untuk menyerang pulau lainnya.

b. *Screen Capture* Respon Pengguna

Berdasarkan hasil penilaian dari sejumlah pengguna terhadap game tersebut dapat diketahui beberapa kelemahan dalam game ini. untuk hasil komentar para pengguna dapat dilihat pada gambar 3.8 :



Gambar 3.8 ***Screen Capture Game Colonial Wars [2]***

c. *Gameplay*

Game Colonial Wars menawarkan sebuah permainan dimana dalam *game* tersebut terdapat 25 level dimana level berikutnya bisa dimainkan apabila level sebelumnya telah dimenangkan. Pemain harus tepat mengambil keputusan untuk menguasai pulau yang lainnya sebelum di dahului oleh NPC, siapa yang berhasil menguasai pulau dia lah yang keluar sebagai pemenang.

d. Elemen *Game*

Elemen permainan yang terdapat dalam permainan game *colonial wars* diantaranya adalah sebagai berikut:

1. Arena Permainan

Pada permainan *game Colonial Wars* arena permainanya adalah di laut yang terdiri dari beberapa pulau. Terdapat rintangan berupa pusaran air di tengah laut yang menghalangi serangan. Dikarena permainan inilah pemain mempertahankan pulau dan menyerang pulau yang lainnya. Berikut area permainannya :



Gambar 3.9 Tampilan area permainan saat memilih level [2]

Di level pertama pemain akan diperlihatkan petunjuk permainan, dan elemen-elemen yang ada pada *game.*



Gambar 3.10 Tampilan area permainan level 1 [2]

Level berikutnya pemain mulai melaksanakan misi yaitu menyerang pulau berikutnya, rintangan yang pada permainan ini adanya pusaran air yang mengahambat serangan pada pulau musuh.



Gambar 3.11 Tampilan area permainan level 2 [2]

2. Karakter Pemain dan Musuh

Karakter pemain dan musuh memiliki karakter yang sama, dalam permainan ini siapa yang paliing cepat menaklukan pulau lawan, dialah yang menjadi pemenang, Namun terdapat rintangan berupa ranjau-ranjau yang harus dihindari baik oleh pemain maupun musuh. Berikut nama dan keterangan karakter :

Tabel 3.4 *Karakter Colonial Wars*

No	Nama	Gambar	Keterangan
1	Kapal Induk		Ini adalah karakter terkuat yang memiliki kecepatan dan daya penghancur paling tinggi
2	Pesawat		Ini adalah karakter sedang, tidak terlalu cepat dan tidak terlalu memberi daya hancur yang tinggi
3	Kapal Laut		Karakter ini berjalan lambat dan tidak menimbulkan kerusakan yang berarti bagi pemain maupun musuh
4	Pusaran Air		Pusaran air yang membuat pemain maupun musuh terperosok kedalamnya

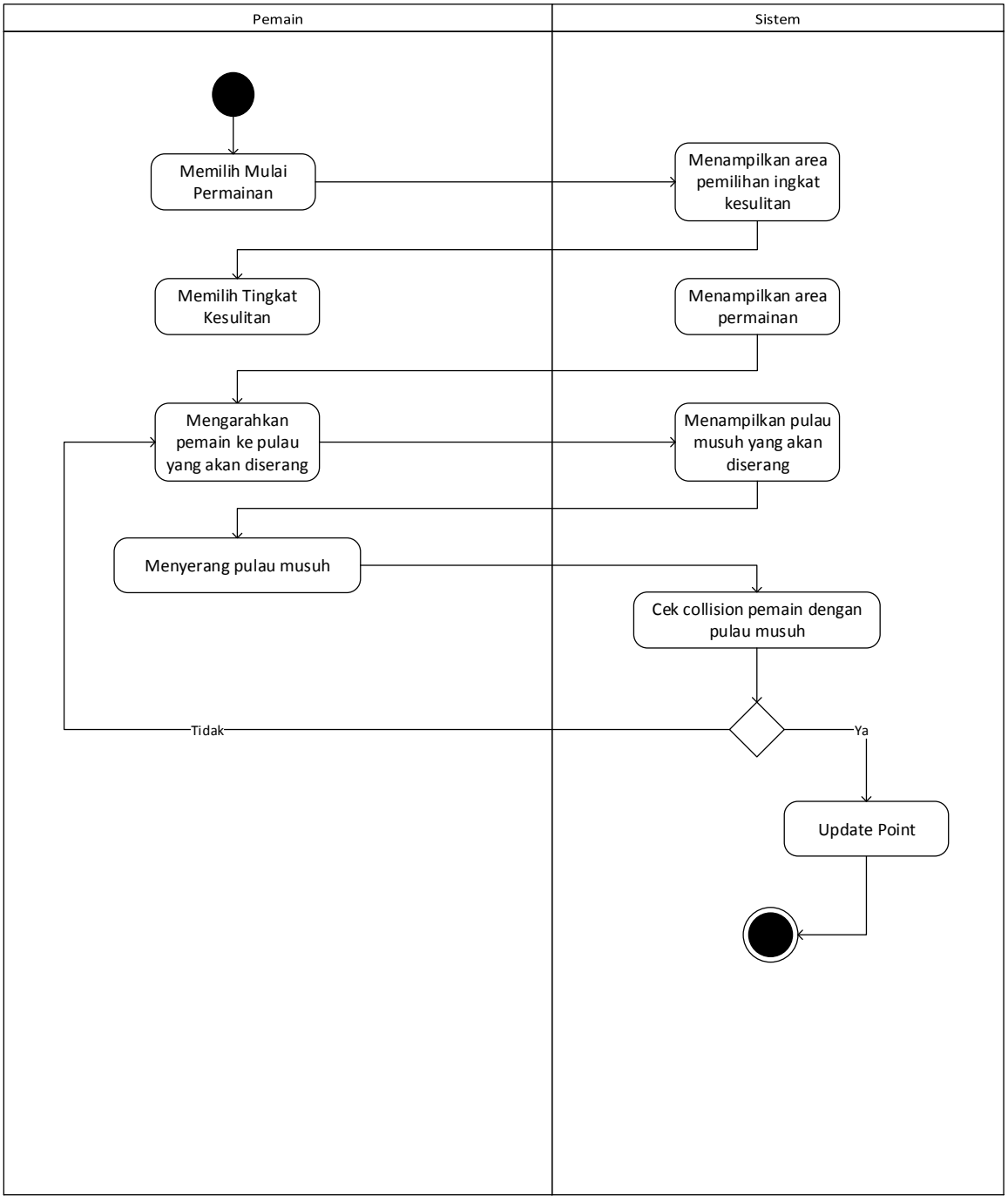
e. Skenario *Game*

Skenario *game* yang terdapat *Colonial Wars* diantaranya adalah sebagai berikut:

Alur permainan pada game *colonial wars* :

a. Pemain memilih mulai permainan
b. Pemain memilih tingkat kesulitan
c. Pemain mengontrol pergerakan pemain dengan cara di drag yang mengarah pada pulau yang akan di serang
d. Pemain menyerang pulau musuh, apabila berhasil maka point akan bertambah

Adapun alur sistem dari permainan *Colonial Wars* digambarkan melalui *activity diagram* , dapat dilihat pada gambar 3.12



Gambar 3.12 Alur Permainan *Colonial Wars*

Adapun misi yang harus dicapai pemain yaitu pemain harus menguasai pulau lainnya dengan cara menyerang pulau tersebut dan pemain harus menghindari pusaran air yang penjadi penghalang kapal penyerang milik kita untuk menyerang pulau lainnya.

3.1.3 Analisis *Game* yang Akan Dikembangkan

3.1.3.1 Pengenalan

The Island Colonized merupakan game berjenis *Action-Adventure* yang akan dibangun. Aplikasi yang akan dibangun mengadopsi cerita penjajah eropa yang menjajah ke daerah Asia dan Amerika latin untuk mencari rempah-rempah, kekayaan, kejayaan dan menyebarkan agama yang dikenal dengan sebutan Gospel, Gold, Glory (3G). *Game* ini bertujuan membebaskan pulau yang sudah diduduki para penjajah, *The Island Colonized* merupakan game dengan sistem single player dan dibangun dengan grafis 2D (dua dimensi).

3.1.3.2 Storyline

The Island Colonized menceritakan tentang seorang pejuang dari pulau Jawa Dwipa, merupakan seorang pemuda yang gagah berani dan pintar yang berada di sebuah pulau terjajah, namun dia merasakan diskriminasi yang berlebihan dari para penjajah sehingga dia beniat keluar dari pulau terjajah tersebut untuk mencari bantuan di pulau lain agar bisa menyelamatkan pulaunya yang terjajah oleh pasukan perompak. Dia pun mempersiapkan diri untuk menjelajah ke pulau yang lain untuk mencari bantuan, agar pulau yang dia huni bisa kembali terbebas dari pada penjajah. Setelah berhasil menemukan bala bantuan sesuai jumlah yang di tentukan, bala bantuan ini diperoleh dari pulau yang belum terjajah, akan tapi banyak para perompak berkumpul di pulau ini dan dia dituntut untuk merayu/membujuk mereka agar bisa menjadi bagian dari pejuang untuk membebaskan pulau. Setelah dia mendapatkan bala bantuan, dia

akan kembali ke pulaunya untuk merebut kembali tempat tinggal mereka yang dijajah dengan melewati beberapa pulau, namun tidak semudah itu karena para perompak sudah menunggu diperbatasan untuk mempertahankan daerah jajahan nya dan sudah dipersenjatai dengan lengkap. Mau tidak mau dia dan pejuang harus bertarung dengan perompak dengan alat seadanya, Setelah bisa melawan para perompak dan berhasil melewati perbatasan, kemudian dia melanjutkan perjalanan untuk ke pulau berikutnya. Dia pun berangkat ke pulau selanjutnya, untuk mendapatkan bantuan para pejuang,dimana di pulai ini berkumpulnya para penjajah dan Raja dari penjajah yaitu perompak. setelah raja perompak dikalahkan Dia pun mendapatkan kebebasan, sehingga dia dikenal sebagai pahlawan.

3.1.3.3 *Level*

Dalam *game* ini, terdapat 3 *level* yang masing-masing *level* dibedakan berdasarkan alur cerita, tingkat kesulitan rintangan yang harus dihadapi, jumlah dan jenis musuh yang bertambah. Di setiap peningkatan level, pemain akan dihadapkan pada musuh dengan tingkat kesulitan yang akan bertambah.

Di *level* pertama, pemain harus merekrut pejuang dan menyelamatkan pulau Borneo , selain merekrut pejuang pemain juga harus menyelamatkan batu bara yang dicuri oleh kawanan perompak, pemain juga mempunyai misi untuk mengusir perompak dari pulau borneo ini , terdapat 10 perompak yang harus dikalahkan untuk melanjutkan ke level selanjutnya,

Di *level* kedua, pemain harus menyelamatkan pulau Nueva Guinee dan sudah merekrut pejuang karena kawanan perompak sudah menduduki pulau, disini kawanan perompak sudah mencapai 20 orang, misi dari level ini yaitu menghadang perompak membawa emas dan mengusir mereka dari pulau tersebut.

Di *level* ketiga, Pemain kembali ke pulaunya dengan membawa bala bantuan, di level ini pemain mempunyai misi mengalahkan seluruh perompak yang berjumlah 30 dan Bos perompak untuk mengambil kembali harta kekayaan mereka dan memperoleh kejayaan/kebebasan.

3.1.3.4 *Gameplay*

Tugas utama pemain dalam memainkan permainan ini yaitu harus melalui 3 level, dimana tiap level memiliki tingkatan kesulitan yang berbeda, semakin tinggi level maka tingkat kesulitan akan semakin sulit, di setiap level pemain akan melewati berbagai rintangan, disini pemain diharuskan merekrut pasukan untuk melawan NPC, pasukan tersebut harus dipersenjatai agar dapat melawan pasukan NPC yang sedang menjajah pulau. *Game* ini mempunyai keunikan dimana tidak setiap orang mau untuk diajak melawan penjajah, pemain harus memberi jaminan keamanan bagi mereka yang tidak bersedia direkrut. Selain itu NPC pasukan perompak mereka akan menyerang agresif ketika mereka mempunyai senjata dan energy yang masih banyak, ketika energy mereka menurun, intensitas serangan mereka juga akan menurun. *Game* juga memiliki pesan supaya kita lebih menghargai sumber daya alam yang kita miliki sebelum direbut oleh orang lain.

3.1.3.5 *Scoring*

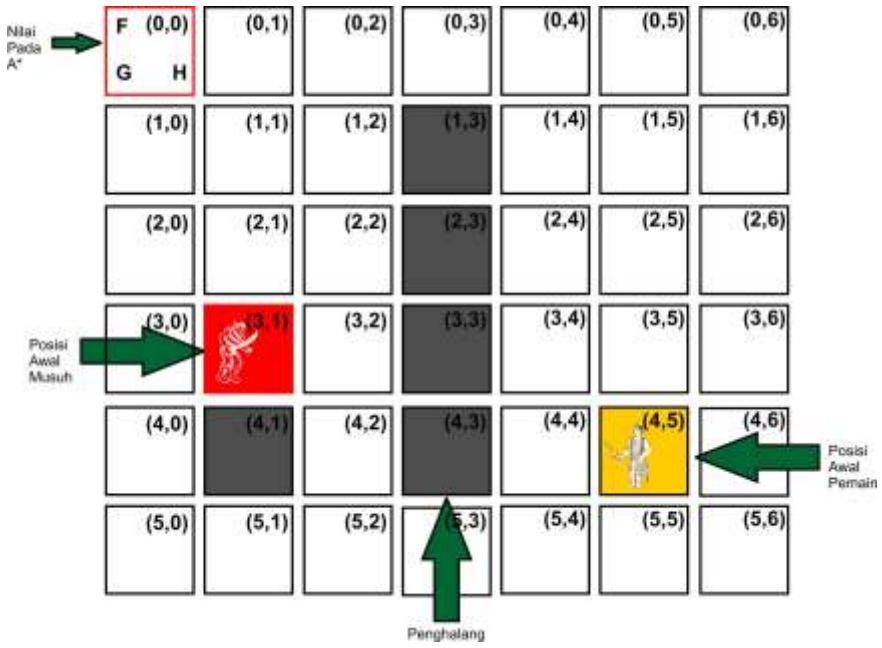
Pada *game The Island Colonized* terdapat poin terhadap setiap musuh yang dikalahkan, poin yang diberikan terhadap setiap musuh adalah 20 poin. Tujuan diberikan score ini untuk menambahkan nyawa pemain, apabila score nya sudah mencapai 500 maka nyawa pemain akan bertambah 1. Terdapat poin juga untuk item yang ditemukan disepanjang perjalanan, seperti mengambil batubara dan emas.

3.1.4 Analisis Algoritma

Pada permainan *The Island Colonized* ini, diterapkan metode dan algoritma yang akan berguna untuk membuat permainan ini menjadi lebih menarik dan menantang.

3.1.4.1 Algoritma A* (A Star)

Algoritma A* di dalam *game* akan diterapkan pada NPC sebagai penentuan jalur terpendek atau *pathfinding* menuju *target.* Cara kerja algoritma A* di dalam *game The Island Colonized* adalah sebagai berikut.



Gambar 3.13 Cara Kerja Algoritma A*

Pada gambar 3.13 terdapat beberapa kotak yang memiliki koordinat dari (0,0) sampai (5,6). 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 :

1. *Node* warna merah : Node awal dari posisi musuh yang berada pada node di koordinat (3,1).
2. *Node* warna abu : Node penghalang yang berada pada *node* di koordinat (4,1), (1,3), (2,3), (3,3), dan (4,3).
3. *Node* warna merah : *Node* dari posisi pemain atau pemain yang berada pada *node* di koordinat (4,5).

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

1. 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*.
2. Nilai G, nilai atau biaya yang dikeluarkan dari posisi awal musuh ke *node* yang ada di sekitarnya.
3. Nilai H, nilai *heuristic* yang digunakan untuk menghitung perkiraan biaya yang dikeluarkan dari posisi awal musuh menuju *node* tujuan / pemain.

Perhitungan nilai F didapat dari persamaan F= G + H. Sedangkan untuk nilai G digunakan nilai 1 untuk gerakan *vertical* atau *horizontal*, Sedangkan untuk gerakan *diagonal* maka akan dikalikan dengan √2 yang berarti 1 x√2 = 1,4.

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 , yang kemudian nilainya dikalikan dengan 10. Atau dirumuskan dengan:

H = 1*(abs(*current*X-*target*X) + abs(*current*Y-*target*Y))

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

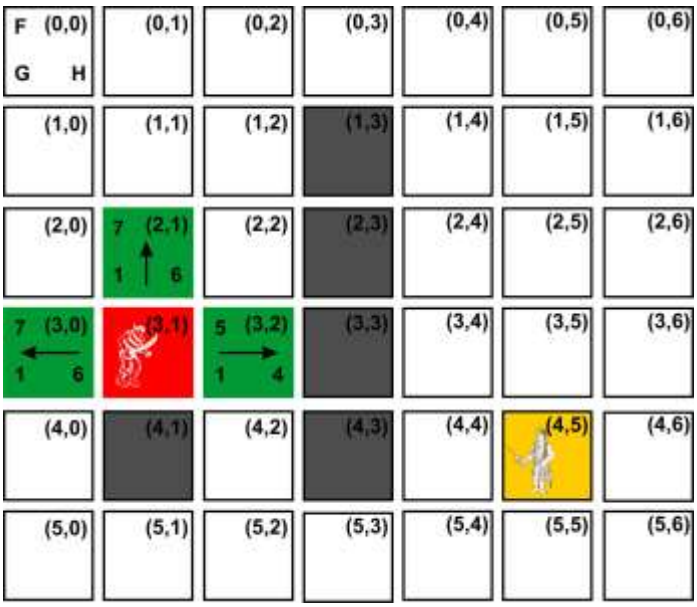
1. *Open list* adalah *list* untuk memeriksa *node-node*.
2. *Closed list* adalah *node-node* yang diabaikan atau *node-node* yang tidak perlu diperiksa.
3. *Parrent* adalah digunakan untuk menelusuri jalan.
4. *Current Node* adalah simpul yang sedang dijalankan dalam algoritma pencarian jalan terpendek.
5. *Pointer* adalah penunjuk jalan terhadap *Node* / posisi musuh *(back* pointer) ketika *node* pemain (target) telah ditemukan.

Langkah berikutnya ketika telah mendapatkan node awal adalah menentukan rute terpendek dengan cara memeriksa setiap *node* yang berdekatan

dengan *node* posisi musuh sampai *node* posisi pemain. Di bawah ini merupakan penjelasan langkah-langkahnya.

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

Berikut ini perhitungan pada langkah pertama yang berada pada node dengan koordinat (3,1) terdapat pada gambar 3.14.



Gambar 3.14 Langkah Pertama Perhitungan Algoritma A*

Pada langkah pertama, *node* (3,1) dimasukan ke dalam open list dan selanjutnya memeriksa *neighbor node* atau *node* yang berdekatan dengan *node* (3,1) yaitu, *node* (2,1), (3,0), (3,2). Setelah diperiksa *node-node* tersebut dimasukan ke dalam *open list*. Hapus *node* (3.1) dari *open list* dan masukan *node*

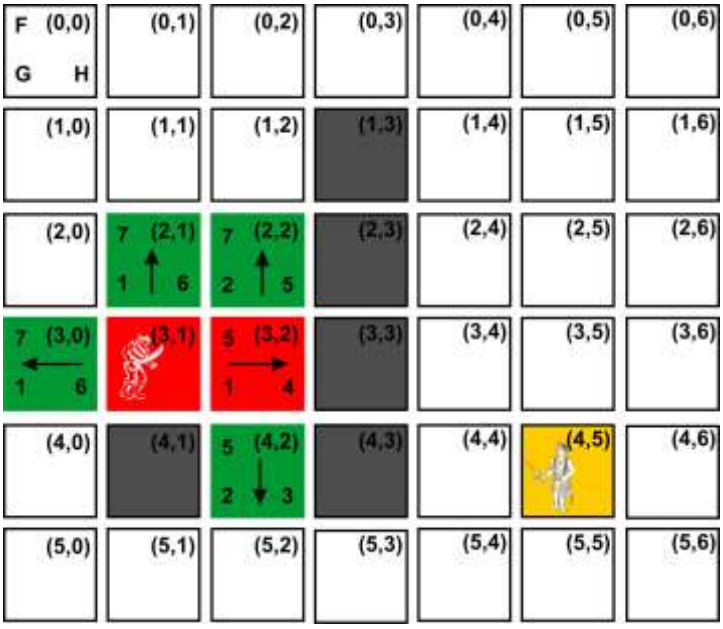
tersebut kedalam *closed list.* Lalu dari *node-node* yang berada di dalam *open list* pilih *node* dengan nilai F yang paling rendah, yaitu *node* (3,2) dengan nilai F = 5.

Langkah pertama menghasilkan *node-node* di dalam *open list* dan *closed list* dijelaskan dengan tabel 3.5.

Tabel 3.5 Perhitungan langkah pertama

Open List	(2,1),(3,0), (3,2)
Closed list	(3,1)
Parent	(3,1)
Current node	(3,2)

Berikut ini perhitungan pada langkah kedua yang berada pada node dengan koordinat (3,2) terdapat pada gambar 3.15.



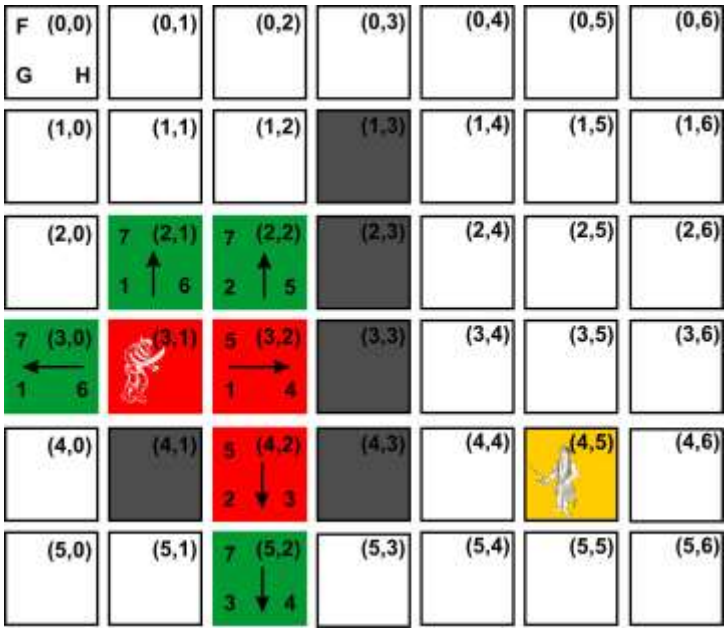
Gambar 3.15 Langkah Kedua Perhitungan Algoritma A*

Berikut ini penjelasan melalui tabel untuk pencarian jalan terpendek pada langkah kedua terdapat pada tabel 3.6.

Tabel 3.6 Perhitungan langkah kedua

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

Berikut ini perhitungan pada langkah ketiga yang berada pada *node* dengan koordinat (4,2) terdapat pada gambar 3.16.



Gambar 3.16 Langkah Ketiga Perhitungan Algoritma A*

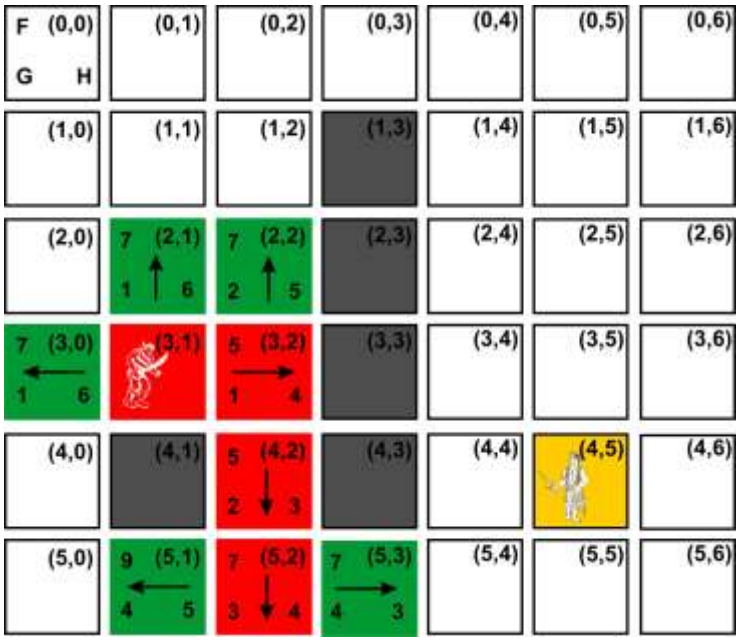
Berikut ini penjelasan melalui tabel untuk pencarian jalan terpendek pada langkah ketiga terdapat pada tabel 3.7.

Tabel 3.7 Perhitungan Langkah Ketiga

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

Current node	(5,2)

Berikut ini perhitungan pada langkah keempat yang berada pada *node* dengan koordinat (5,2) terdapat pada gambar 3.17.



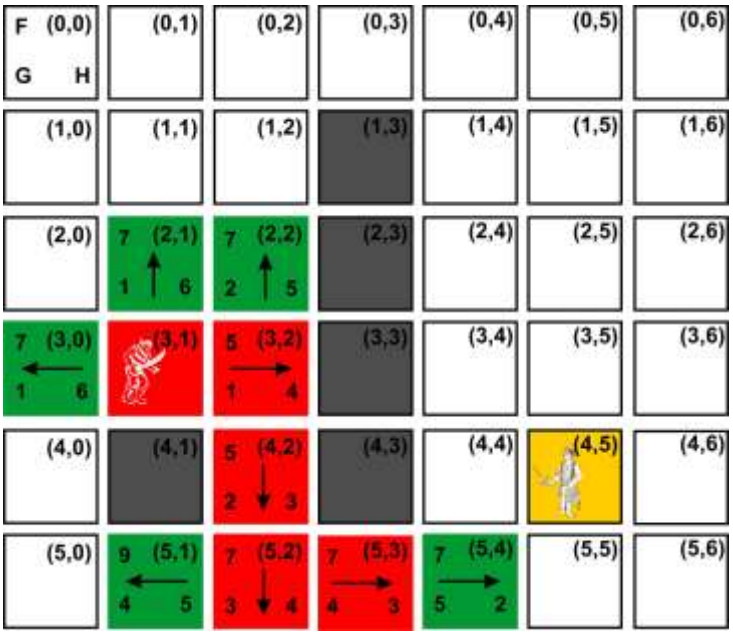
Gambar 3.17 Langkah Keempat Perhitungan Algoritma A*

Berikut ini penjelasan melalui tabel untuk pencarian jalan terpendek pada langkah keempat terdapat pada tabel 3.8.

Tabel 3.8 Perhitungan langkah keempat

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

Berikut ini perhitungan pada langkah kelima yang berada pada *node* dengan koordinat (5,3) terdapat pada gambar 3.18.



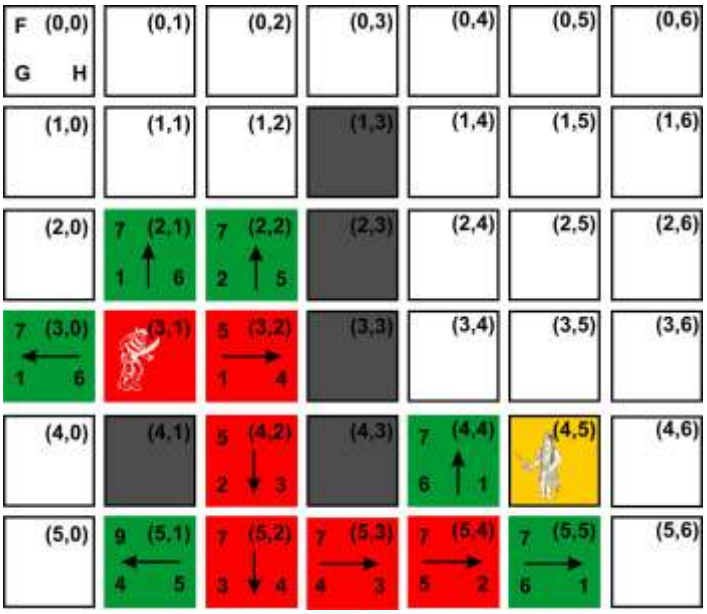
Gambar 3.18 Langkah Kelima Perhitungan Algoritma A*

Berikut ini penjelasan melalui tabel untuk pencarian jalan terpendek pada langkah kelima terdapat pada tabel 3.9.

Tabel 3.9 Perhitungan langkah kelima

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

Berikut ini perhitungan pada langkah keenam yang berada pada *node* dengan koordinat (5,4) terdapat pada gambar 3.19.



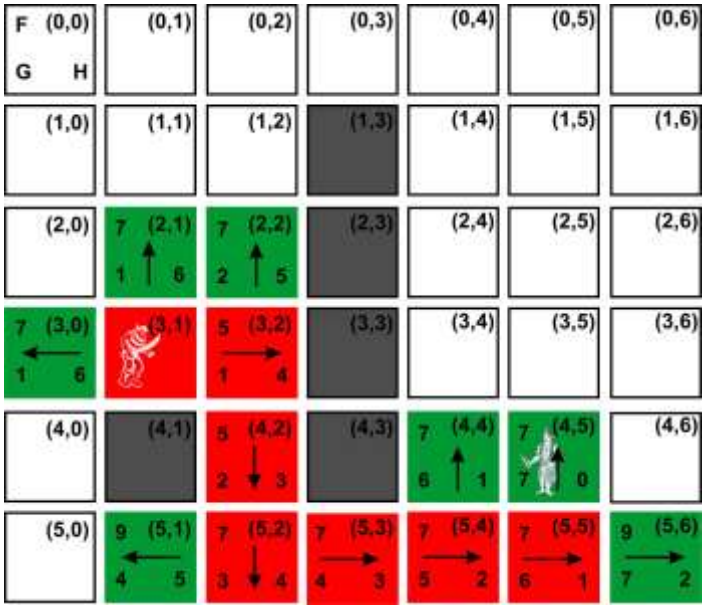
Gambar 3.19 Langkah Keenam Perhitungan Algoritma A*

Berikut ini penjelasan melalui tabel untuk pencarian jalan terpendek pada langkah keenam terdapat pada tabel 3.10.

Tabel 3.10 Perhitungan langkah ke enam

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

Berikut ini perhitungan pada langkah ketujuh yang berada pada *node* dengan koordinat (5,5) terdapat pada gambar 3.20.



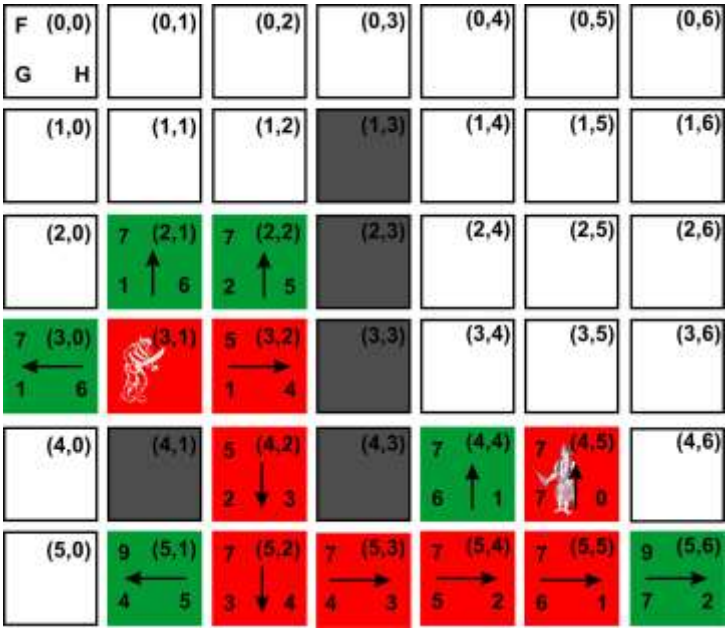
Gambar 3.20 Langkah Ketujuh Perhitungan Algoritma A*

Berikut ini penjelasan melalui tabel untuk pencarian jalan terpendek pada langkah ketujuh terdapat pada tabel 3.11.

Tabel 3.11 Perhitungan langkah ketujuh

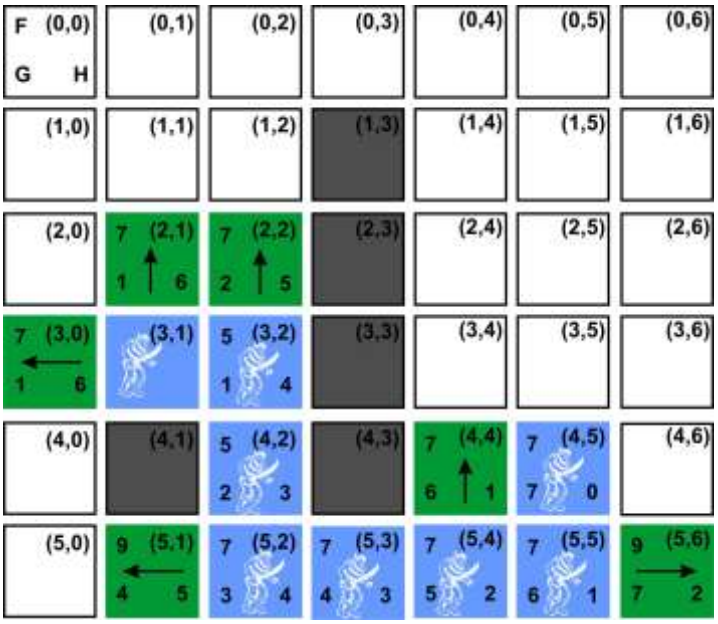
Open List	(4,5), (5,6)
Closed list	(3,1), (3,2), (4,2), (5,2), (5,3), (5,4), (5,5)
Parent	(5,5)
Current node	(4,5)

Pada gambar 3.21 posisi musuh sudah mencapai target yaitu pemain



Gambar 3.21 Target Ditemukan

Musuh akan menelusuri jalur terpendek yang telah dihitung untuk mencapai target tersebut, seperti terdapat pada gambar 3.22



Gambar 3.22 Jalur Terpendek Perhitungan Algoritma A*

Penentuan jalan terbaik dengan melakukan pemeriksaan dimulai dari *target*, dalam ilustrasi adalah kotak berwarna merah. Dengan mengikuti arah *pointer* yang berarti akan membawanya kembali ke *node* awal, maka akan didapatkan jalan terbaiknya (warna biru), Maka jika diimplementasikan pada *game,* nantinya musuh akan melewati halangan dan akhirnya akan menemukan rute terpendek ke tujuan.

Berikut adalah *pseudo code* algoritma A star terdapat pada tabel 3.12.

Tabel 3.12 *Pseudo Code* Algoritma A*

Algoritma A*	create the open list of nodes, initially containing only our starting node
1	
2	create the closed list of nodes, initially empty
3	while (we have not reached our goal) {
4	consider the best node in the open list (the node with the lowest f value)
5	if (this node is the goal) {
6	then we're done
7	}
8	else {
9	move the current node to the closed list and consider all of its neighbors
10	for (each neighbor) {
11	if (this neighbor is in the closed list and our current g value is lower) {
12	update the neighbor with the new, lower, g value
	change the neighbor's parent to our current node

13	}
14	else if (this neighbor is in the open list and our current g value is lower) {
15	update the neighbor with the new, lower, g value change the neighbor's parent to our current node
16	}
17	else this neighbor is not in either the open or closed list {
18	add the neighbor to the open list and set its g value
19	}
20	}
21	}
22	}

3.1.4.2 Fuzzy Logic

Pada *game* ini *fuzzy logic* digunakan untuk menentukan variasi prilaku yang dilakukan oleh NPC. Dengan adanya *fuzzy logic* masing-masing NPC dapat merubah perilaku berbasis perubahan variabel masukan. Metode *fuzzy* yang digunakan adalah metode Tsukamoto karena dengan metode ini output hasil inferensi dari tiap-tiap aturan diberikan secara tegas (crisp) berdasarkan α-predikat (*fire strength*), sehingga dapat mewakili nilai perilaku yang telah dirancang.

Untuk menghasilkan prilaku pada NPC ada dua variabel yang digunakan, yaitu Jarak (dekat, sedang dan jauh) dan Kekuatan (lemah dan kuat). Dengan menggunakan dua variabel diharapkan NPC mampu menentukan prilaku yang akan dilakukan.

Tahapan-tahapan dari analisis *fuzzy logic* adalah sebagai berikut :

1. Variabel Fuzzy

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

1. Variabel Jarak sebagai variabel input.
2. Variabel Kekuatan sebagai variabel input.
3. Variabel Keputusan sebagai variabel output.

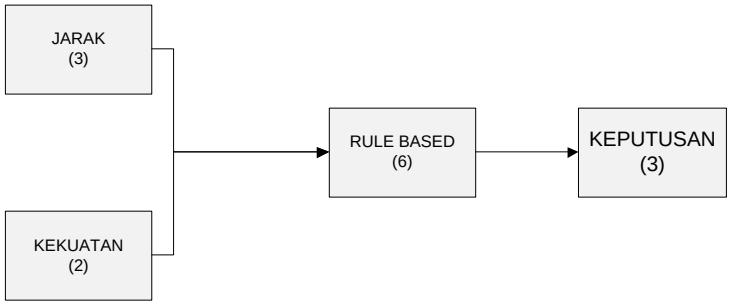
2. Nilai linguistik

Dari tiga variabel yang digunakan, maka nilai linguistiknya sebagai berikut:

1. Variabel Jarak, terbagi menjadi 3 *input* yaitu : Dekat, Sedang, Jauh.
    1. Dekat
    2. Sedang
    3. Jauh
2. Variabel Kekuatan, terbagi menjadi 2 *input* yaitu : Lemah dan Kuat.
    1. Lemah
    2. Kuat
3. Variabel Keputusan, terbagi menjadi 3 *output* yaitu :
    1. Menyerang
    2. Bertahan
    3. Melarikan Diri.

3. Fuzzyfikasi

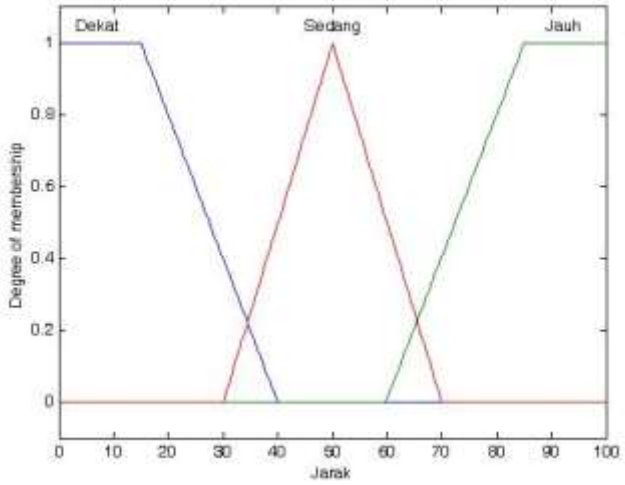
Fuzzyfikasi merupakan proses memetakan nilai *crisp* (numerik) ke dalam himpunan *fuzzy* dan menentukan derajat keanggotaannya. Secara garis besar pemetaan nilai *crisp* ke dalam himpunan *fuzzy* dijelaskan dengan gambar sebagai berikut.



Gambar 3.23 *Fuzzy Interface System*

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

1. Variabel Jarak, terbagi menjadi 3 input yaitu : Dekat, Sedang dan Jauh. *Range* nilai untuk variabel Jarak antara 0-100 akan dijelaskan sebagai berikut :
    1. Dekat = 0 - 40
    2. Sedang = 30 - 70
    3. Jauh = 60 -100



Gambar 3.24 Grafik input jarak

Pada gambar 3.24 Menunjukan sebuah grafik Jarak yang mempunyai *range* nilai dari 0 – 100, setiap nilai linguistik dari variabel jarak seperti Dekat, Sedang dan Jauh mempunyai nilai *fuzzyfikasi* yang berbeda-beda. Perhitungan nilai fuzzyfikasi didapatkan dari beberapa fungsi, fungsi yang digunakan pada Variable Jarak ada 2 yaitu fungsi Trapesium dan Fungsi Segitiga.

- Variabel linguistik Dekat yang memiliki *range* nilai antara 0 – 40 nilai fuzzyfikasinya dihitung berdasarkan fungsi trapesium, berikut persamaannya :

$$\mu Dekat[x]\begin{cases}1; x \leq 15\\ \dfrac{40-x}{25}\\ 0; z \geq 40\end{cases}; 15 \leq z \leq 40$$

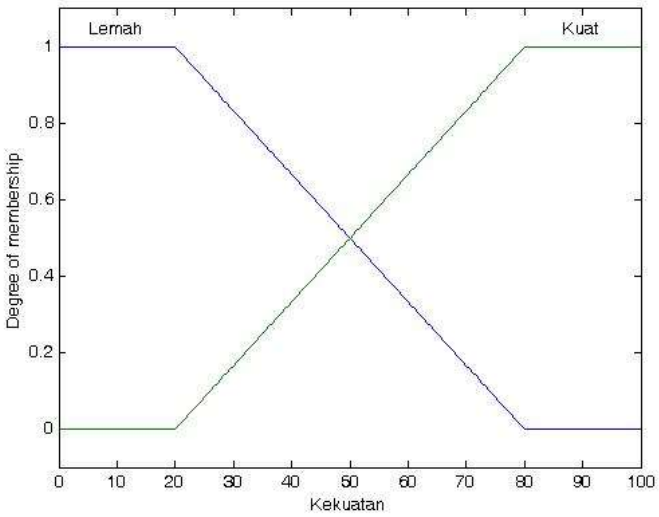
- Variabel linguistik Sedang yang memiliki *range* nilai antara 30 – 70 nilai *fuzzyfikasi*nya dihitung berdasarkan fungsi segitiga, berikut persamaannya :

$$\mu Sedang[x]\begin{cases}0; x \leq 30/x \geq 70\\ \dfrac{x-30}{20}; 30 \leq x \leq 50\\ \dfrac{50-x}{20}; 50 \leq x \leq 70\end{cases}$$

- Variabel linguistik Jauh yang memiliki *range* nilai antara 60 – 100 nilai *fuzzyfikasi*nya dihitung berdasarkan fungsi trapesium, berikut persamaannya :

$$\mu Jauh[x]\begin{cases}0; x \leq 60\\ \dfrac{x-60}{25}\\ 1; x \geq 85\end{cases}; 60 \leq x \leq 85$$

2. Variabel Kekuatan, terbagi menjadi 2 input yaitu : Lemah dan Kuat. *Range* nilai variabel Kekuatan diantara 0 – 100 akan dijelaskan sebagai berikut :
    1. Lemah = 0 - 80
    2. Kuat = 20 – 100



Gambar 3.25 Grafik Input Kekuatan

Pada gambar 3.25 Menunjukan sebuah grafik trapesium Kekuatan yang mempunyai range nilai dari 0 – 100, setiap nilai linguistik dari variabel kekuatan seperti lemah dan kuat mempunyai nilai *fuzzyfikasi* yang berbeda-beda.

- Variabel linguistik Lemah yang memiliki *range* nilai antara 0 – 80 nilai *fuzzyfikasi*nya dihitung berdasarkan fungsi trapesium, berikut persamaannya :

$$\mu Lemah[y]\begin{cases}1; y \leq 20\\ \dfrac{80-z}{60}\ ; 20 \leq z \leq 80\\ 0; z \geq 80\end{cases}$$

- Variabel linguistik Kuat yang memiliki *range* nilai antara 20 – 100 nilai *fuzzyfikasi*nya dihitung berdasarkan fungsi trapesium, berikut persamaannya :

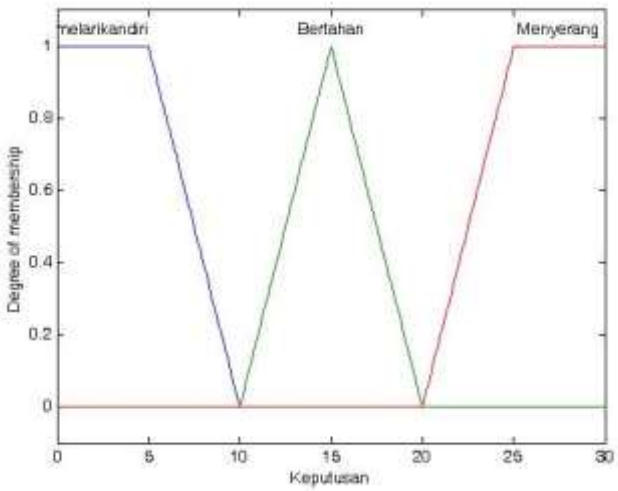
$$\mu Kuat[y]\begin{cases}0; z \leq 20\\ \dfrac{y-20}{60}\ ; 20 \leq z \leq 80\\ 1; z \geq 80\end{cases}$$

3. Variabel Keputusan, yaitu Menyerang, Bertahan dan Melarikan Diri. *Range* nilai variabel keputusan diantara 0-30, akan dijelaskan sebagai berikut :

   1. Melarikan diri = 0 -10

   2. Bertahan = 10 - 20

   3. Menyerang = 20 - 30



Gambar 3.26 Grafik Keputusan

Pada gambar 3.26 Menunjukan sebuah grafik keputusan yang mempunyai *range* nilai dari 0 – 30, setiap nilai lingustik seperti melarikan diri, bertahan dan menyerang mempunyai nilai *fuzzyfikasi* yang berbeda-beda.

- Variabel linguistik Melarikan Diri yang memiliki *range* nilai antara 0 – 10 nilai *fuzzyfikasi*nya dihitung berdasarkan fungsi trapesium, berikut persamaannya :

$$\mu MelarikanDiri[z]\begin{cases} 1; z \leq 6 \\ \dfrac{10-z}{4} \; ; 6 \leq z \leq 10 \\ 0; z \geq 10 \end{cases}$$

- Variabel linguistik Bertahan yang memiliki *range* nilai antara 10 – 20 nilai *fuzzyfikasi*nya dihitung berdasarkan fungsi segitiga, berikut persamaannya :

$$\mu Bertahan[z]\begin{cases} 0; z \leq 10/z \geq 20 \\ \dfrac{z-10}{5}; 10 \leq z \leq 15 \\ \dfrac{15-z}{5}; 15 \leq z \leq 20 \end{cases}$$

- Variabel linguistik Menyerang yang memiliki *range* nilai antara 20 – 30 nilai *fuzzyfikasi*nya dihitung berdasarkan fungsi trapesium, berikut persamaannya :

$$\mu Menyerang[z]\begin{cases} 0; z \leq 20 \\ \dfrac{z-20}{4}; 20 \leq z \leq 24 \\ 1; z \geq 24 \end{cases}$$

4. Kaidah *fuzzy* atau *fuzzy's rules*

Kaidah *fuzzy (rules)* atau aturan-aturan yang diterapkan dalam penentuan perilaku NPC berjumlah enam *rules* yaitu:

1. If (Jarak is Dekat) and (Kekuatan is Lemah) then (Perilaku NPC is Melarikan diri)
2. If (Jarak is Dekat) and (Kekuatan is Kuat) then (Perilaku NPC is Menyerang)
3. If (Jarak is Sedang) and (Kekuatan is Lemah) then (Perilaku NPC is Bertahan)
4. If (Jarak is Sedang) and (Kekuatan is Kuat) then (Perilaku NPC is Menyerang)
5. If (Jarak is Jauh) and (Kekuatan is Lemah) then (Perilaku NPC is Melarikan diri)
6. If (Jarak is Jauh) and (Kekuatan is Kuat) then (Perilaku NPC is Menyerang)

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 Tsukamoto, setiap konsekuen pada aturan yang berbentuk *IF-Then* harus direpresentasikan dengan suatu himpunan fuzzy dengan fungsi keanggotaan. Sebagai hasilnya, output hasil inferensi dari tiap-tiap aturan diberikan secara tegas (crisp) berdasarkan α-predikat (*fire strength*). Hasil akhirnya diperoleh dengan menggunakan rata-rata terbobot.

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

z*=titik pusat nilai *crisp*

μi =derajat keanggotaan masing-masing variabel

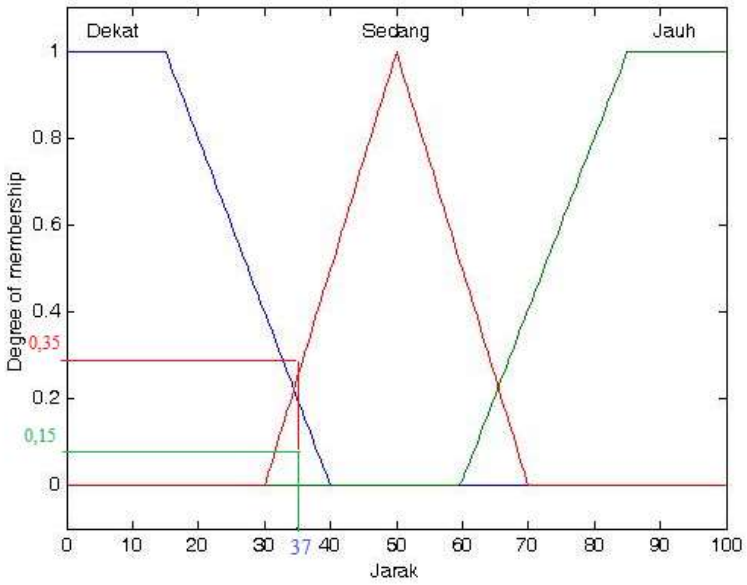
zi = nilai domain pada variabel linguistik z.

Contoh Kasus

Misal, apabila jarak memiliki nilai [37] dan Kekuatan memiliki nilai [67], maka tahapan-tahapan atau proses agregasi yang menentukan keluaran sesuai dengan *rules* yang sudah dibuat akan dijelaskan seperti berikut :

1. Fuzzifikasi

Perhitungan *Fuzzyfikasi* untuk Jarak dari setiap variabel lingustik (Dekat, Sedang dan Jauh) apabila nilai jarak [37] adalah sebagai berikut :



Gambar 3.27 Grafik input jarak

$$\mu dekat[37] = \frac{(40-37)}{(40-20)} = \frac{3}{20} = 0{,}15$$

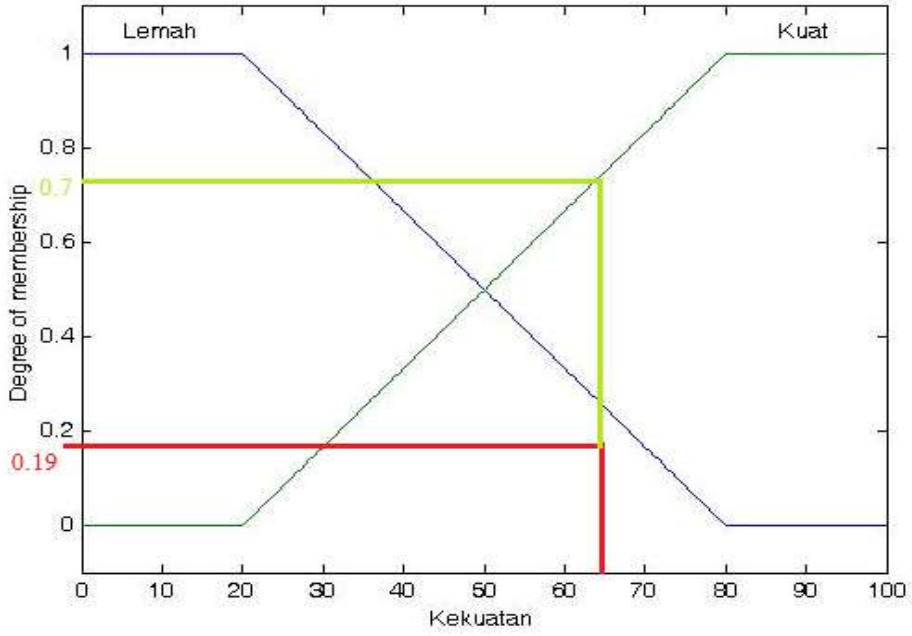
$$\mu sedang[37] = \frac{(37-30)}{(50-30)} = \frac{7}{20} = 0{,}35$$

$$\mu Jauh[37] = 0$$

Dari hasil perhitungan *fuzzifikasi* menggunakan fungsi trapesium untuk mencari nilai derajat keanggotaan, maka didapatkan nilai derajat keanggotaan dari masing-masing variabel lingustik yaitu :

1. Fungsi Keanggotaan Dekat = 0,15
2. Fungsi Keanggotaan Sedang = 0,35
3. Fungsi Keanggotaan Jauh = 0

Perhitungan *Fuzzyfikasi* untuk Kekuatan dari setiap variabel lingustik (Lemah dan Kuat) apabila nilai Kekuatan [67] adalah sebagai berikut :



Gambar 3.28 Grafik input kekuatan

$$\mu lemah[67] = \frac{(80-67)}{67} = 0{,}19$$

$$\mu kuat[67] = \frac{(67-20)}{67} = 0{,}7$$

Dari hasil perhitungan untuk mencari nilai fungsi kenaggotaan menggunakan fungsi trapesium, didapatkan nilai fungsi keanggotaan dari masing-masing variabel lingustik yaitu :

1. Fungsi Keanggotaan Lemah = 0,19
2. Fungsi Keanggotaan Kuat = 0,7

1. Implikasi

Setelah nilai derajat keanggotaan didapatkan, maka nilai-nilai derajat tersebut harus dibandingkan sesuai dengan *Rule* yang ada, tujuannya untuk mencari nilai minimum untuk setiap *rule* yang dihitung.

a. Rule 1

If (Jarak is Dekat) and (Kekuatan is Lemah) then (Perilaku NPC is Melarikan diri)

→ α-predikat1 = μDekat ∩μ Lemah

= min(μDekat [37],μLemah [67])

= **min (0.15, 0.19) = 0.15**

Defuzzy

$$\frac{10 - z}{4} = 0.2$$

$Z = 10 – 0.8 = 9.2$

b. Rule 2

If (Jarak is Dekat) and (Kekuatan is Kuat) then (Perilaku NPC is Menyerang)

→ α-predikat2 = μDekat ∩μ Kuat

= min(μDekat [37],μKuat [67])

= **min (0.15, 0.7) = 0.15**

Defuzzy

$$\frac{z - 20}{4} = 0.2$$

Z = 20 – 0,8 = 19,2

c. Rule 3

If (Jarak is Sedang) and (Kekuatan is Lemah) then (Perilaku NPC is Bertahan)

→ α-predikat3 = μDekat ∩μ Lemah

= min(μSedang [37],μLemah [67])

= **min (0.35, 0.19) = 0.19**

Defuzzy

$$\frac{z - 10}{5} = 0.25$$

Z= 10 – 1.25 = 8.75

d. Rule 4

If (Jarak is Sedang) and (Kekuatan is Kuat) then (Perilaku NPC is Menyerang)

→ α-predikat4 = μSedang ∩μ Kuat

= min(μsSedang [37],μKuat[67])

= **min (0.35, 0.7) = 0.35**

Defuzzy

$$\frac{z-20}{4} = 0.25$$

Z = 20 – 1 = 19

e. Rule 5

If (Jarak is Jauh) and (Kekuatan is Lemah) then (Perilaku NPC is Melarikan diri)

→ α-predikat5 = μJauh ∩μ Lemah

= min(μJauh [37],μLemah [67])

= **min (0, 0.19) = 0**

Defuzzy

$$\frac{10-z}{4} = 0$$

Z = 10 – 0 = 10

f. Rule 6

If (Jarak is Jauh) and (Kekuatan is Kuat) then (Perilaku NPC is Menyerang)

→ α-predikat6 = μJauh ∩μ Kuat

= min(μJauh [37],μKuat[67])

= **min (0, 0.7) = 0**

Defuzzy

$$\frac{z-20}{4}=0$$

Z = 20 – 0 = 20

$$Z=\frac{0.15x9{,}2+0.15x19.2+0.19x8.75+0.35x19+0x10+0x20}{0.15+0.15+0.19+0.35+0+0}=\frac{12.57}{0.84}$$
$$=14\ .96$$

Z = 14.96

Jadi Keputusan yang dihasilkan berdasarkan dari nilai jarak [37] dan Kekuatan [67] adalah **bertahan**. Sesuai dengan nilai z yang didapatkan yaitu **14,96**

3.1.5 Analisis Kebutuhan Non Fungsional

3.1.5.1 Analisis Kebutuhan Perangkat Keras

Analisis perangkat keras yang digunakan untuk mengetahui spesifikasi perangkat keras dalam membangun aplikasi permainan *The Island Colonized*, seperti pada tabel 3.13.

Tabel 3.13 Spesifikasi Peerangkat Keras

Nama Perangkat	Spesifikasi
Prosesor	2.13GHz
RAM	4GB
Harddisk	320GB
Display	Resolusi 1366 x 768
Keyboard	Standar
Mouse	Standar

3.1.5.2 Analisis Kebutuhan Perangkat Lunak

Analisis perangkat lunak digunakan untuk mengetahui spesifikasi perangkat lunak untuk membangun aplikasi permainan *The Island Colonized* **,** Berikut adalah spesifikasi perangkat lunak terdapat pada tabel 3.14.

Tabel 3.14 Spesifikasi Perangkat Lunak

Nama Perangkat Lunak	Spesifikasi
Sistem Operasi	Microsoft Windows 8
Tools	Construct 2

3.1.5.3 Analisis Pengguna

Ada 2 hal yang akan memudahkan pemain ketika memainkan *game* ini yaitu pengetahuan dan pengalaman, terutama pengetahuan dan pengalaman dalam memainkan *game The Island Colonized*.

1. *User Knowledge and Experience*

Berikut ini klasifikasi *knowledge and experience* dari pengguna aplikasi terdapat pada tabel 3.15.

Tabel 3.15 *knowledge and experience*

Education Level	Bisa digunakan oleh berbagai kalangan, seperti pelajar, mahasiswa hingga masyarakat awam.
Reading Level	Bisa digunakan oleh berbagai *level* pendidikan dengan reading *level* yang sedang.
Typing Skills	Tidak memerlukan *typing skills* yang tinggi.
Computer Literacy	Moderate (menengah)
Task Experience	Bisa digunakan oleh dengan pengalaman penggunaan *handphone* dan *game* yang sedang.
System Experience	Bisa digunakan oleh dengan pengalaman penggunaan *handphone* dan *game* yang sedang.
Application Experience	Bisa digunakan dalam sistem operasi Android *Gingerbread* atau versi yang lebih tinggi.

Native Language	Menggunakan satu bahasa, yakni Indonesia.
Use Of Other System	Bisa di jalankan tanpa perlu 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.*

Berikut ini klasifikasi *Users Physical Characteristic* dari pengguna aplikasi terdapat pada tabel 3.16.

Tabel 3.16 *User Physical Characteristic*

Age	10 tahun ke atas [8]
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 indikator 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.

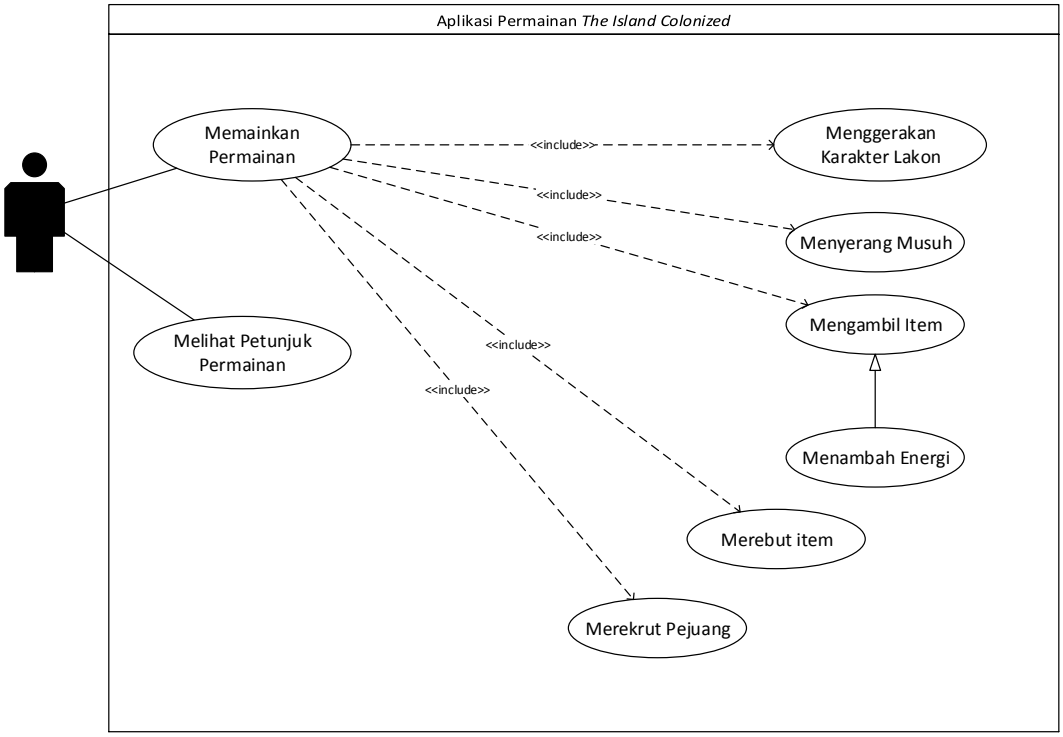
3.1.6 Analisis Kebutuhan Fungsional

Identifikasi aktor dapat dilakukan dalam analisis berorientasi objek dengan menggunakan UML yaitu menentukan aktor atau pengguna sistem. Aktor dalam konteks UML menampilkan peran pengguna atau sesuatu diluar sistem yang dikembangkan dapat berupa perangkat keras, end user, sistem yang lain dan sebagainya.

3.1.6.1 Use case Diagram

Use case adalah abstraksi dari interaksi antara sistem dan dan aktor. Use case mendeskripsikan bagaimana sistem akan terlihat di mata user.

Secara umum, interaksi antara pengguna yaitu pemain dengan aplikasi permainan *The Island Colonized* digambarkan pada diagram *Use Case*, seperti pada gambar 3.29.



Gambar 3.29 ***Use Case Diagram The Island Colonized***

3.1.6.2 Skenario *Use case*

Skenario pada diagram *use case* berisi tentang kejadian *'flow if event' use case* utama. Di dalamnya menjelaskan urutan interaksi antara aktor dengan *use case* tersebut dari awal sampai akhir. Adapun scenario *use case* yang terdapat pada aplikasi permainan *The Island Colonized* ini adalah sebagai berikut :

1. **Skenario *Use Case* Memainkan Permainan**

Skenario *use case* ini menjelaskan interaksi antara aktor dan sistem , yaitu pemain, dengan *use case* memainkan permainan yang dijelaskan pada tabel 3.17.

Tabel 3.17 Skenario *Use Case* Memainkan Permainan

Identifikasi		
Nama	Memainkan Permainan	
Tujuan	Memainkan permainan	
Deskripsi	Proses ketika pemain akan memainkan permainan dan masuk ke area permainan	
Aktor	Pemain	
Skenario Utama		
Kondisi Awal	Sistem menampilkan menu utama	
Aksi Aktor		**Reaksi Sistem**
a. Memilih menu mainkan		
		b. Menampilkan Intro 1
c. Menekan tombol lanjut		
		d. Menampilkan Intro 2
e. Menekan tombol lanjut		

	f. Menampilkan Area Permainan
	g. Sistem memutar audio
h. Pemain memainkan permainan	
	i. Sistem mengecek darah karakter lakon
	j. Jika darah tidak habis dan waktu masih tersisa
	k. Melanjutkan permainan
	l. Sistem mengecek misi
	m. Jika pemain berhasil menyelesaikan misi yang ada dalam permainan dan waktu masih tersisa
	n. Sistem menampilkan tampilan misi selesai
	o. Sistem menampilkan level berikutnya
Skenario Tambahan	
	p. Jika darah karakter lakon habis dan waktu masih tersisa maka karakter lakon mati
	q. Jika gagal menyelesaikan misi maka kembali ke awal permainan
	r. Menampilkan *Game over* dan kembali ke awal permainan
Kondisi Akhir	Sistem akan menampilkan tampilan misi selesai jika pemain berhasil menyelesaikan misi dan sistem akan kembali ke awal permainan jika pemain gagal menyelesaikan misi

2. Skenario *Use Case* Melihat Petunjuk Permainan

Skenario *use case* ini menjelaskan interaksi antara aktor, yaitu pemain, dengan seluruh *use case* Melihat Petunjuk Permainan yang dijelaskan pada tabel 3.18

Tabel 3.18 Skenario *Use Case* Lihat Petunjuk Permainan

Identifikasi		
Nama	Melihat Petunjuk Permainan	
Tujuan	Masuk ke submenu yang berisi petunjuk permainan	
Deskripsi	Proses untuk melihat informasi cara memainkan permainan	
Aktor	Pemain	
Skenario Utama		
Kondisi Awal	Pemain berada di menu utama	
Aksi Aktor		**Reaksi Sistem**
a. Memilih menu petunjuk		
		b. Menampilkan halaman petunjuk tentang info cara memainkan permainan
Kondisi Akhir	Sistem akan menampilkan informasi bagaimana memainkan aplikasi *The Island Colonized*	

3. Skenario *Use Case* Menggerakan Karakter Lakon

Skenario *use case* ini menjelaskan interaksi antara aktor, yaitu pemain, dengan *use case* Menggerakan Karakter Lakon yang dijelaskan pada tabel 3.19.

Tabel 3.19 Skenario *Use Case* Menggerakan Karakter Lakon

Identifikasi	
Nama	Menggerakan Karakter Lakon
Tujuan	Menggerakan Karakter Lakon Dalam Permainan

Deskripsi	Proses ketika pemain menggerakan karakter lakon	
Aktor	Pemain	
Skenario		
Kondisi Awal	Pemain berada pada area permaian	
Aksi Aktor		**Reaksi Sistem**
a. Pemain menggerakan karakter lakon		
b. Pemain menekan *button left*		
		c. Karakter lakon berlari ke kiri
d. Pemain menekan *button right*		
		e. Karakter lakon berlari ke kanan
f. Pemain menekan *button up*		
		g. Karakter lakon melompat ke atas
h. Pemain menekan *button down*		
		i. Karakter lakon melompat ke bawah
Kondisi Akhir	Sistem akan menampilkan karakter lakon melakukan pergerakan berlari atau melompat	

4. Skenario Use Case Menyerang Musuh

Skenario *use case* ini menjelaskan interaksi antara aktor, yaitu pemain, dengan *use case* Menyerang Musuh yang dijelaskan pada tabel 3.20.

Tabel 3.20 Skenario *Use Case* Menyerang Musuh

Identifikasi	
Nama	Menyerang Musuh
Tujuan	Karakter lakon melakukan penyerangan terhadap musuh

Deskripsi	Proses ketika karakter lakon melakukan penyerangan terhadap musuh
Aktor	Pemain
Skenario Utama	
Kondisi Awal	Pemain berada pada area permaian

Aksi Aktor	**Reaksi Sistem**
a. Pemain membuat karakter lakon mendekati musuh	
	b. Mengecek *Collision* antara karakter lakon dengan musuh
c. Jika terjadi *collision* maka pemain menekan *button attack*	
	d. Karakter lakon dalam permainan akan menyerang sesuai dengan instruksi dari pemain
	e. Sistem mengeluarkan audio
Skenario Tambahan	
f. Jika tidak terjadi *collision* maka pemain membuat karakter lakon mendekati musuh.	

Kondisi Akhir	Sistem akan menampilkan karakter lakon melakukan penyerangan terhadap musuh

5. Skenario U*se Case* Mengambil Item

Skenario *use case* ini menjelaskan interaksi antara aktor, yaitu pemain, dengan *use case* Mengambil Item yang dijelaskan pada tabel 3.21.

Tabel 3.21 Skenario *Use Case* Mengambil item

Identifikasi		
Nama	Mengambil Item	
Tujuan	Mengambil item dalam permainan	
Deskripsi	Proses ketika pemain mengambil item dalam permainan	
Aktor	Pemain	
Skenario Utama		
Kondisi Awal	Pemain berada pada area permaian	
Aksi Aktor		**Reaksi Sistem**
a. Membuat karakter lakon mendekati item		
		b. Menampilkan *icon* item
		c. Mengecek *collision* antara karakter lakon dengan item
		d. Jika terjadi *collision* maka *icon* item akan menghilang dari permainan
		e. Sistem akan menambahkan poin setelah mengambil item
Skenario Tambahan		
g. Jika tidak terjadi *collision* pemain membuat karakter lakon mendekati item		
Kondisi Akhir	Sistem menghilangkan *icon* item dan menambahkan poin apabila item berhasil di ambil tetapi jika tidak pemain membuat karakter lakon mendekati item kembali.	

6. Skenario *Use Case* Merebut Item

Skenario *use case* ini menjelaskan interaksi antara aktor, yaitu pemain, dengan *use case* Merebut Item yang dijelaskan pada tabel 3.22.

Tabel 3.22 Skenario *Use Case* Merebut item

Identifikasi	
Nama	Merebut Item
Tujuan	Merebut item dalam permainan
Deskripsi	Proses ketika karakter lakon merebut item dalam permainan
Aktor	Pemain
Skenario Utama	
Kondisi Awal	Pemain berada pada area permaian

Aksi Aktor	**Reaksi Sistem**
a. Menggerakan karakter lakon untuk mencari musuh	
	b. Menampilkan musuh
c. Karakter lakon melawan musuh	
	d. Sistem mengecek darah musuh apakah sudah habis atau belum
	e. Jika sudah habis maka sistem menampilkan item
f. Karakter lakon mengambil item yang muncul ketika musuh dikalahkan	
	g. *Icon* item menghilang dari permainan
	h. Sistem menambahkan poin setelah

	item di ambil karakter lakon
Skenario Tambahan	
i. Jika darah musuh belum habis maka pemain membuat karakter lakon melawan musuh kembali	
Kondisi Akhir	Jika darah musuh habis karakter lakon dapat menambah poin dari item tetapi jika tidak pemain membuat karakter lakon melawan musuh kembali

7. Skenario Use Case Merekrut pejuang

Skenario *use case* ini menjelaskan interaksi antara aktor, yaitu pemain, dengan *use case* Merekrut pejuang yang dijelaskan pada tabel 3.23.

Tabel 3.23 Skenario *Use Case* Mencari bantuan

Identifikasi	
Nama	Merekrut pejuang
Tujuan	Merekrut bantuan untuk mengalahkan musuh
Deskripsi	Proses ketika karakter lakon merekrut pejuang
Aktor	Pemain
Skenario Utama	
Kondisi Awal	Pemain berada pada area permaian
Aksi Aktor	**Reaksi Sistem**
a. Karakter lakon mendekati pejuang	
	b. Menampilkan pejuang
	c. Mengecek *collision* antara karakter lakon dengan pejuang
	d. Jika terjadi *collision* maka sistem akan menampilkan dialog

e. Karakter lakon berdialog dengan pejuang	
	f. Mengecek uang jaminan yang dibawa karakter lakon
	g. Jika karakter lakon membawa uang jaminan maka sistem menampilkan pejuang di *level* berikutnya
Skenario Tambahan	
h. Jika tidak terjadi *collision* antara karakter lakon dengan pejuang maka pemain membuat karakter lakon mendekati pejuang kembali	
i. Jika uang jaminan tidak dibawa oleh karakter lakon maka pemain membuat karakter lakon mencari uang jaminan	
Kondisi Akhir	Sistem menampilkan pejuang di *level* berikutnya jika karakter lakon membawa uang jaminan tetapi jika tidak membawa uang jaminan pemain membuat karakter lakon mencari uang jaminan

8. Skenario *Use Case* Menambah Energi

Skenario *use case* ini menjelaskan interaksi antara aktor, yaitu pemain, dengan *use case* Menambah Energi yang dijelaskan pada tabel 3.24.

Tabel 3.24 Skenario *Use Case* Menambah energi

Identifikasi	
Nama	Menambah Energi
Tujuan	Menambah energi karakter lakon
Deskripsi	Proses ketika karakter lakon menambah energi
Aktor	Pemain
Skenario Utama	

Kondisi Awal	Pemain berada pada area permaian
Aksi Aktor	**Reaksi Sistem**
a. Membuat karakter lakon mendekati item energi	
	b. Menampilkan *icon* energi
	c. Mengecek *collision* antara karakter lakon dengan *icon* energi
	d. Jika terjadi *collision* maka *icon* energi menghilang dari permainan
	e. Sistem menambahkan energi kepada karakter lakon
Skenario Tambahan	
f. Jika tidak terjadi *collision* antara karakter lakon dengan energi maka pemain membuat karakter lakon mendekati item energi	
Kondisi Akhir	Energi karakter lakon bertambah jika mengembil energi tetapi jika tidak pemain membuat karakter lakon mendekati item energi

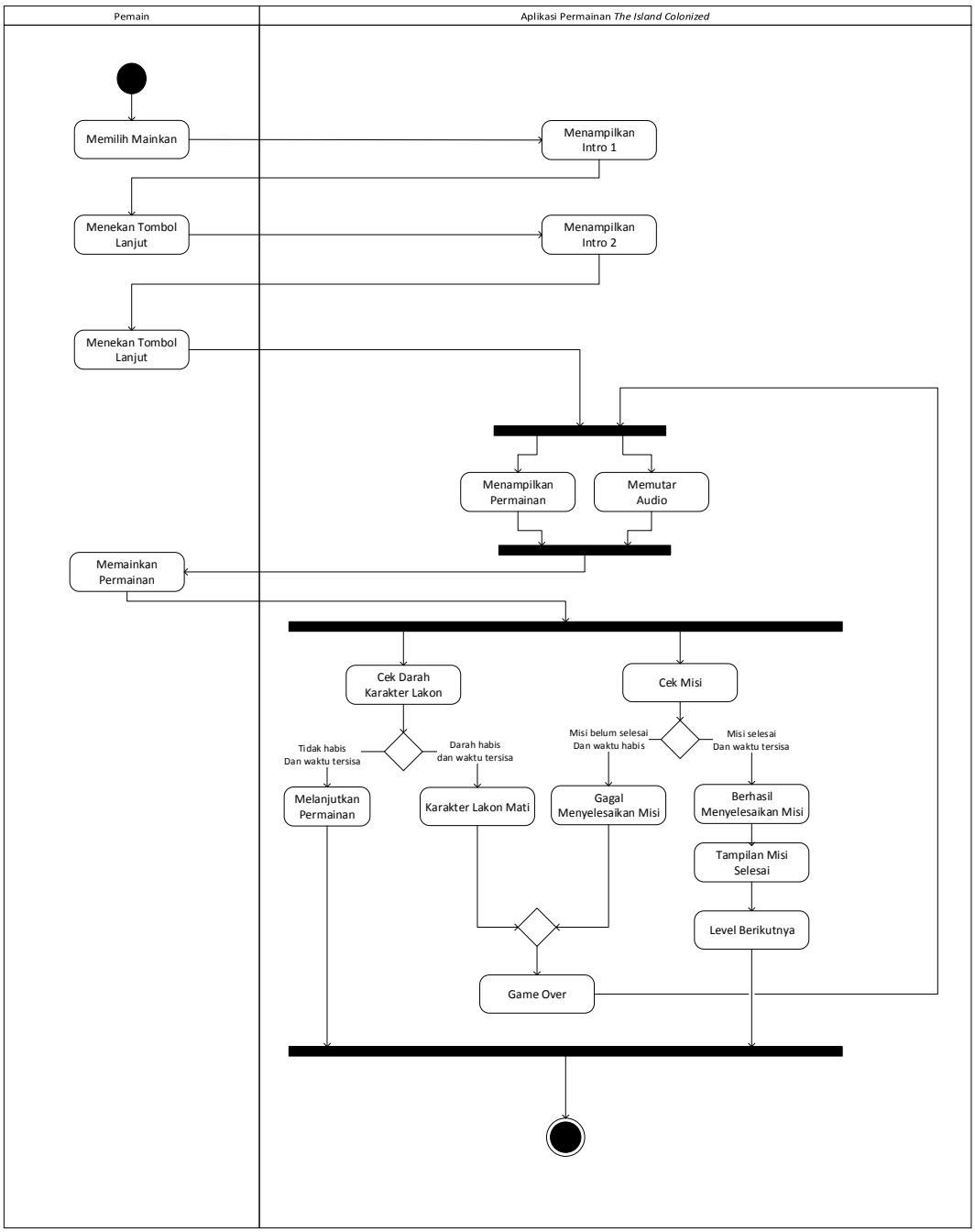
3.1.6.3 Activity Diagram

Activity diagram menggambarkan berbagai alir aktivitas dalam sistem yang sedang dirancang, bagaimana masing-masing alir berawal, *decision* yang mungkin terjadi, dan bagaimana mereka berakhir. *Activity diagram* juga dapat menggambarkan proses paralel yang mungkin terjadi pada beberapa eksekusi.

1. ***Activity Diagram* Memainkan Permainan**

Proses ini terjadi saat pemain memilih menu Mainkan pada halaman utama permainan *The Island Colonized.* Sistem akan menampilkan *intro,* yang kemudian

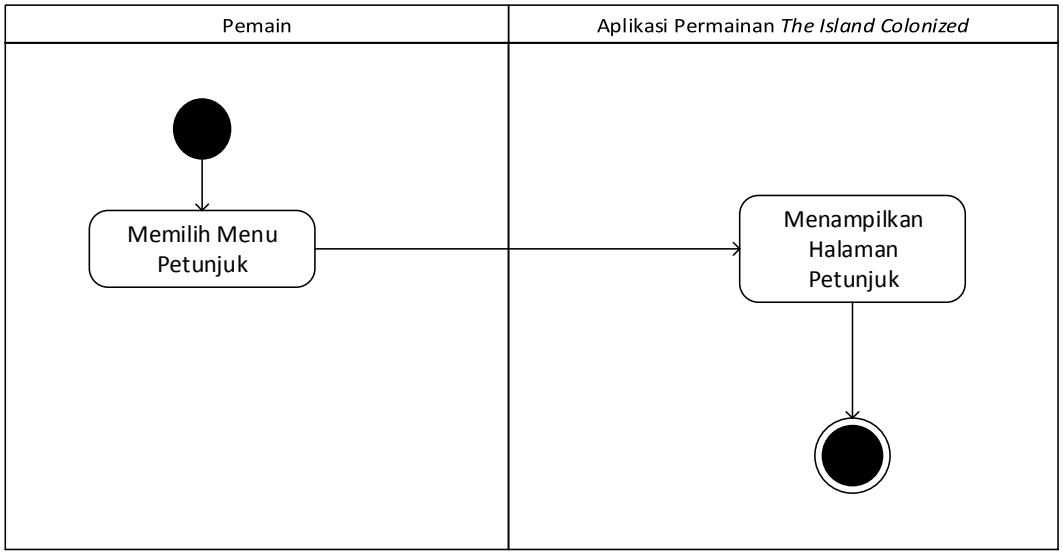
masuk ke area permainan, Activity diagram mulai permainan terdapat pada gambar 3.30.



Gambar 3.30 *Activity Diagram* Memainkan Permainan

2. *Activity Diagram* Melihat Petunjuk Permainan

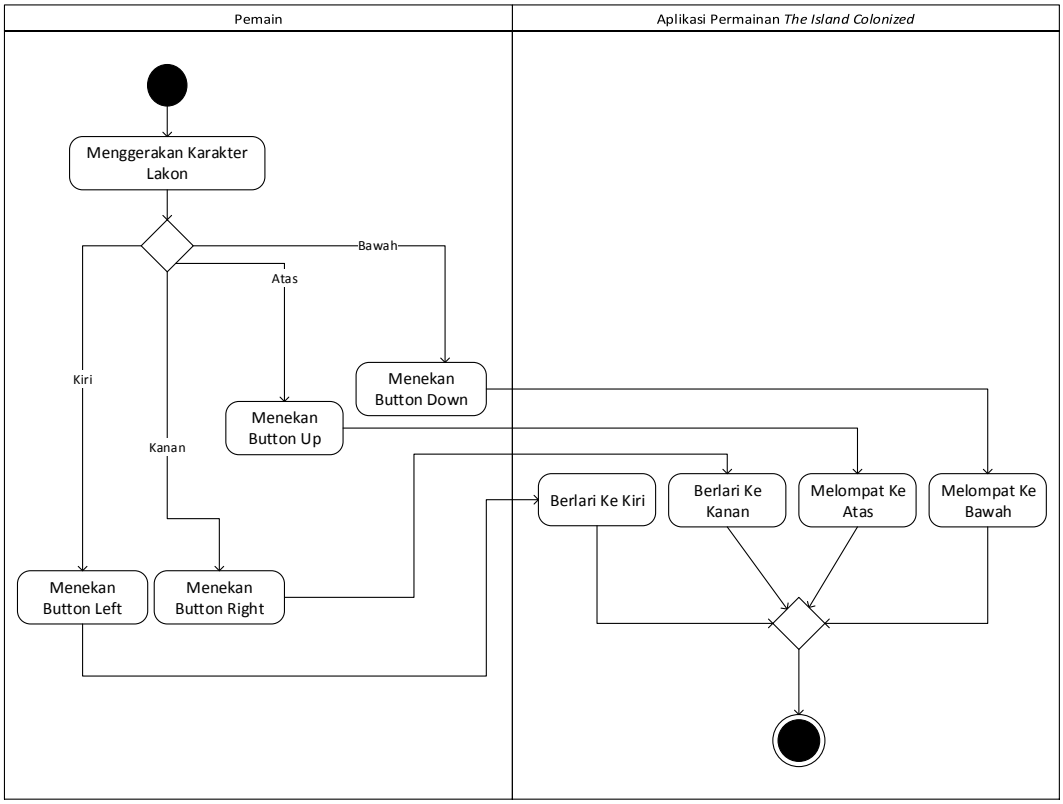
Proses ini terjadi saat pemain memilih menu lihat petunjuk pada halaman utama. Sistem akan menampilkan layar lihat petunjuk untuk memberikan informasi kepada pemain tentang petunjuk memainkan *game*. Diagram *activity* Melihat Petunjuk Permainan terdapat pada gambar 3.31.



Gambar 3.31 *Activity Diagram* Melihat Petunjuk Permainan

3. *Activity Diagram* menggerakan Karakter Lakon

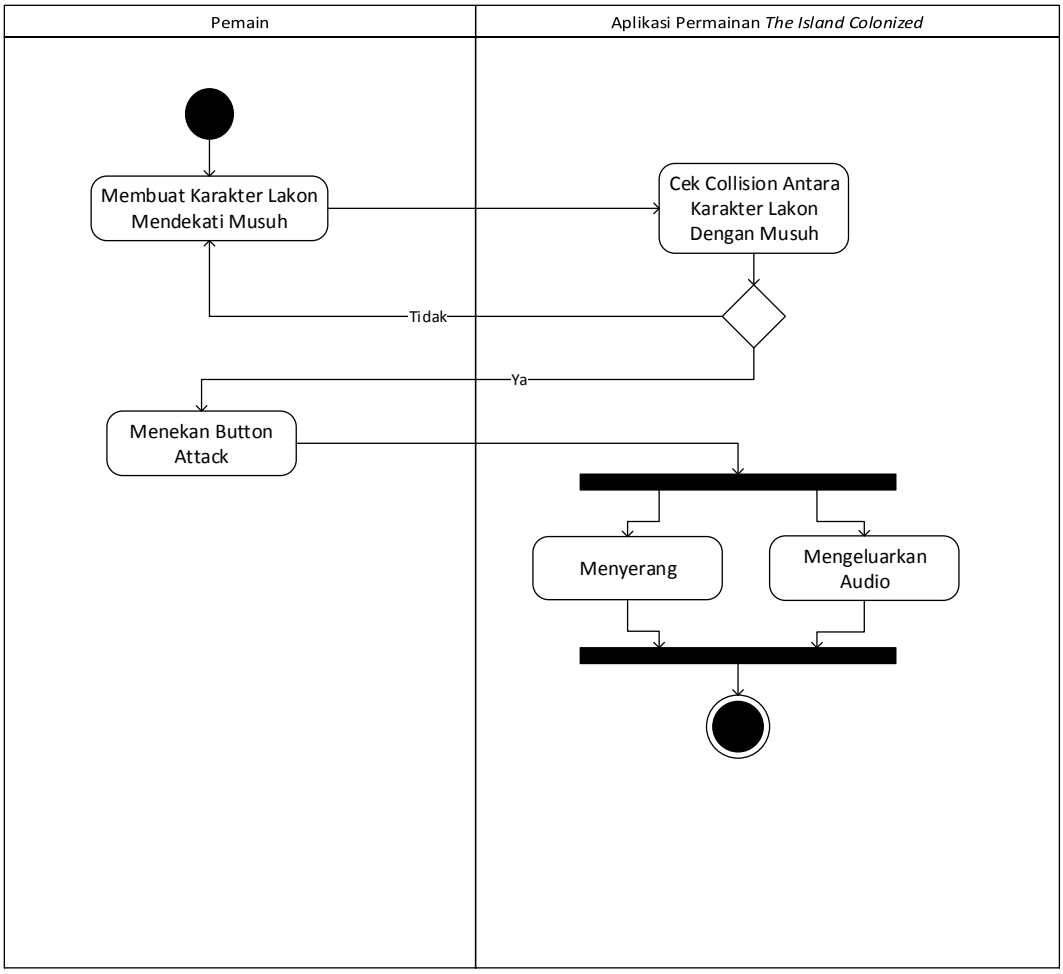
Proses ini terjadi saat Pemain menggerakan karakter lakon dalam permainan, seperti berlari dan melompat. Diagram *activity* Menggerakan Karakter Lakon terdapat pada gambar 3.32.



Gambar 3.32 *Activity Diagram* Menggerakan Karakter Lakon

4. *Activity Diagram* Menyerang Musuh

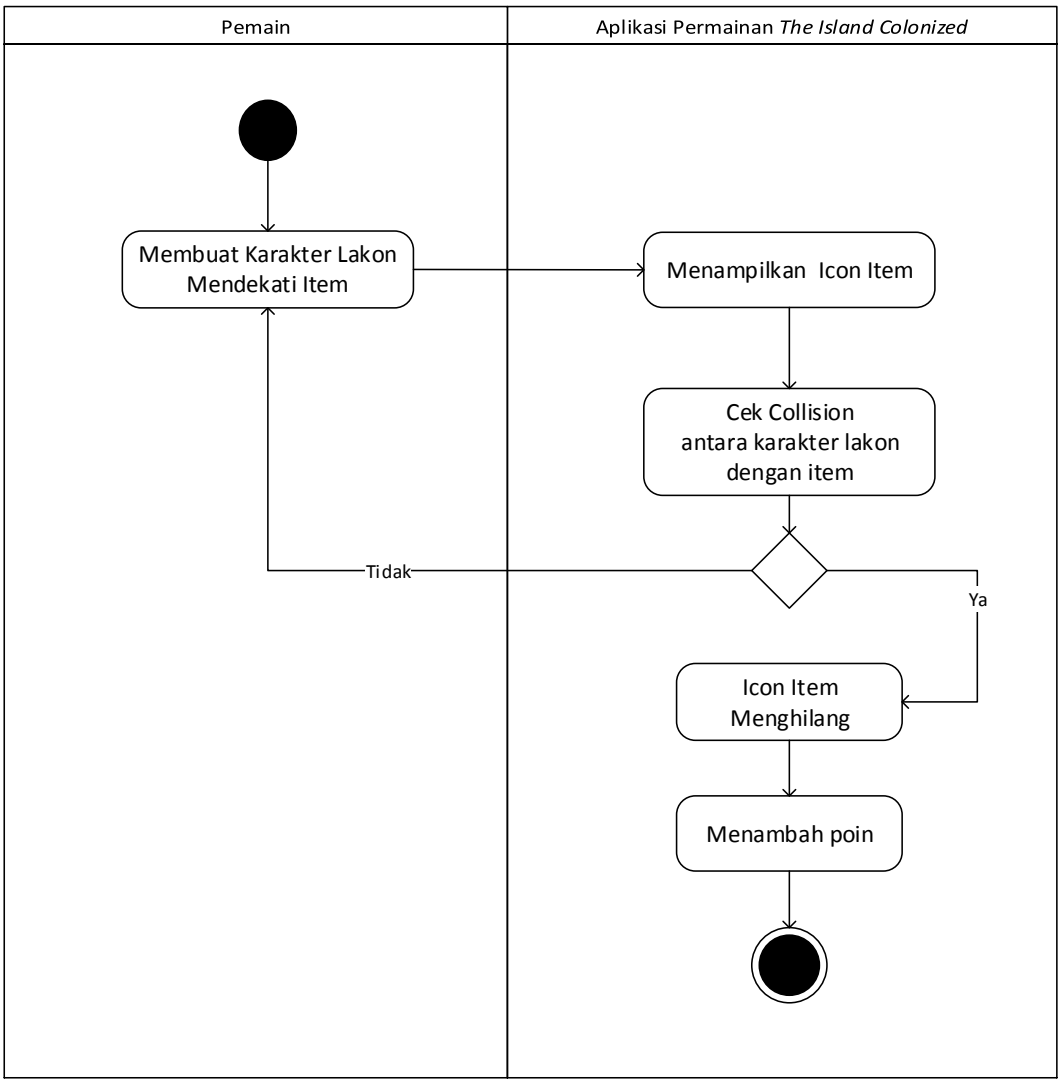
Proses ini terjadi saat permainan di jalankan. Sistem akan menampilkan musuh, dimana pemain harus mengalahkan musuh-musuh tersebut. Diagram *activity* menyerang musuh terdapat pada gambar 3.33.



Gambar 3.33 *Activity Diagram* Menyerang Musuh

5. *Activity Diagram* Mengambil Item

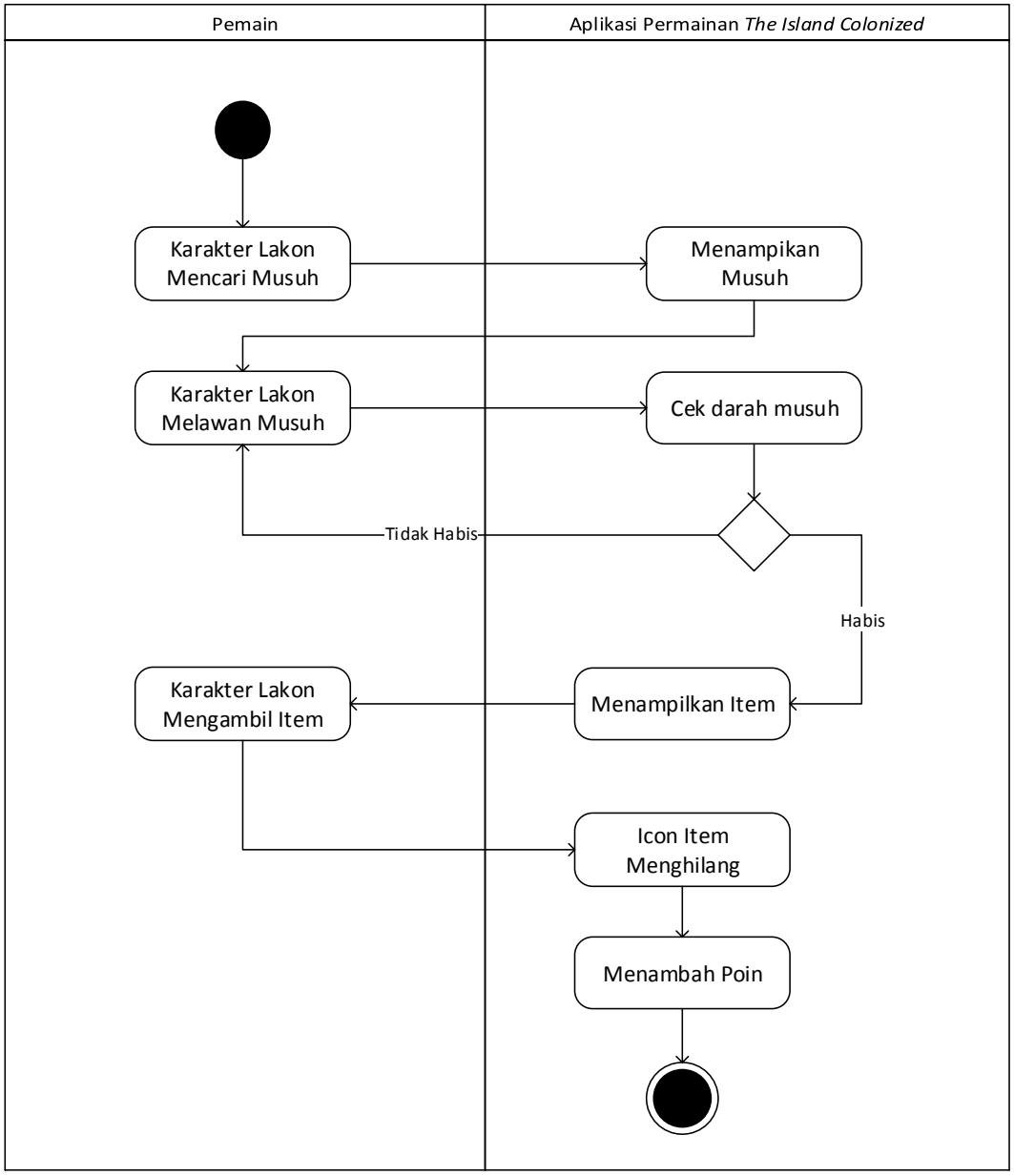
Proses ini terjadi saat permainan di jalankan. Karakter lakon akan digerakan oleh pemain untuk mengambil item-item yang terdapat di area permainan. Diagram *activity* Mengambil Item terdapat pada gambar 3.34.



Gambar 3.34 Activity Diagram Mengambil Item

6. *Activity Diagram* Merebut Item

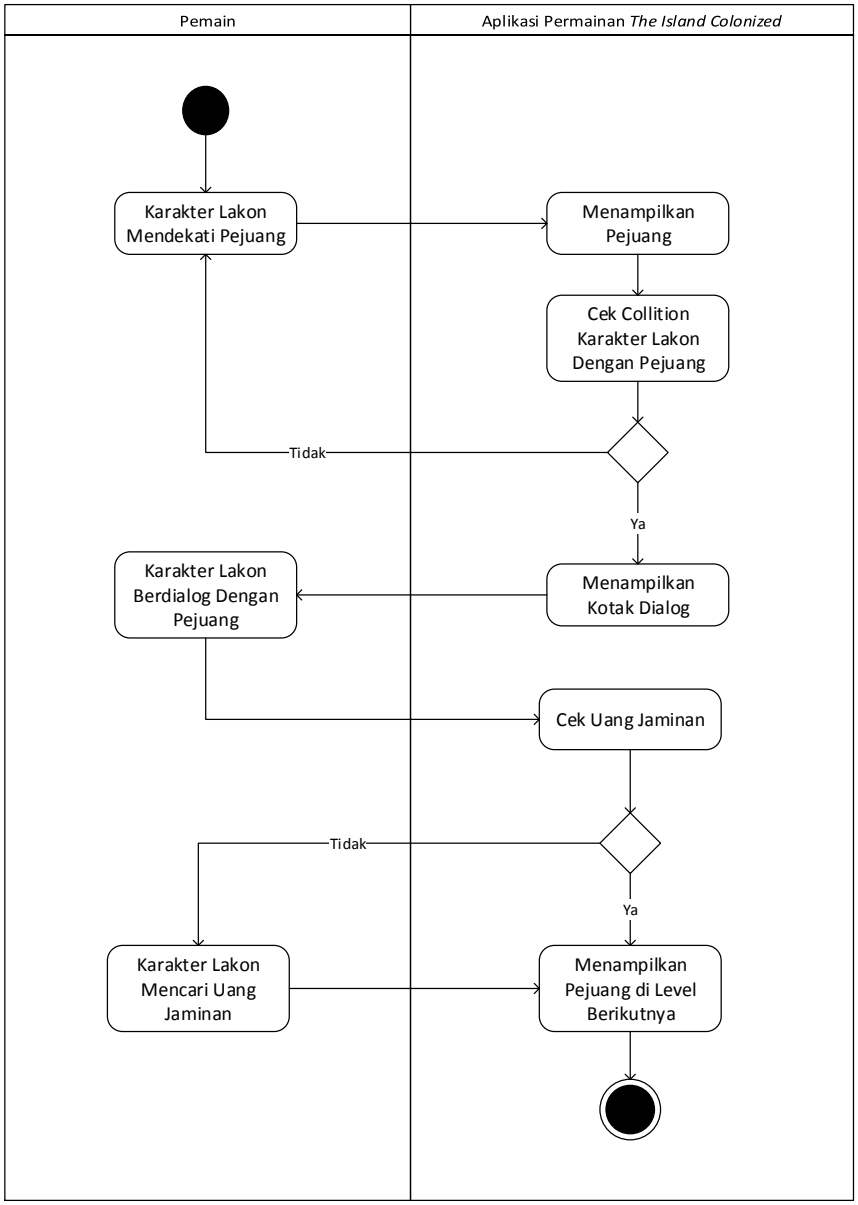
Proses ini terjadi saat karakter lakon bertarung melawan musuh, karakter lakon harus mengalahkan musuh terlebih dahulu untuk mendapatkan item tersebut. *Activity diagram* Merebut Item dapat dilihat pada gambar 3.35.



Gambar 3.35 *Activity Diagram* Merebut Item

7. *Activity Diagram* Merekrut Pejuang

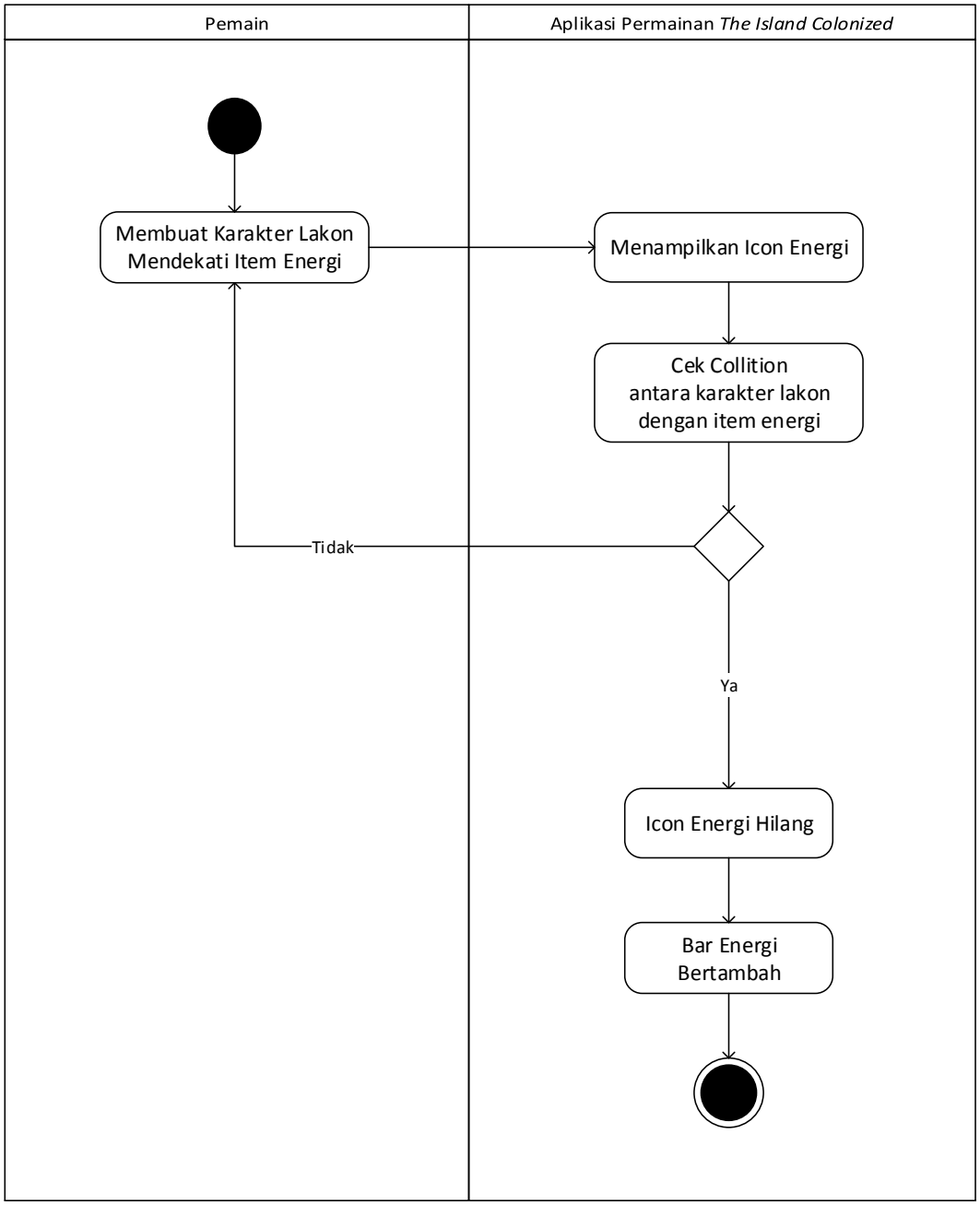
Proses ini terjadi saat karakter lakon sudah berada di area permainan, karakter mencari pejuang untuk membantu mengalahkan perompak, dan akan terjadi dialog dimana dialog tersebut menjadi jawaban pejuang apakah dinrinya mau direkrut apa tidak. *Activity diagram* Merekrut Pejuang dapat dilihat pada gambar 3.36.



Gambar 3.36 *Activity Diagram* Merekrut Pejuang

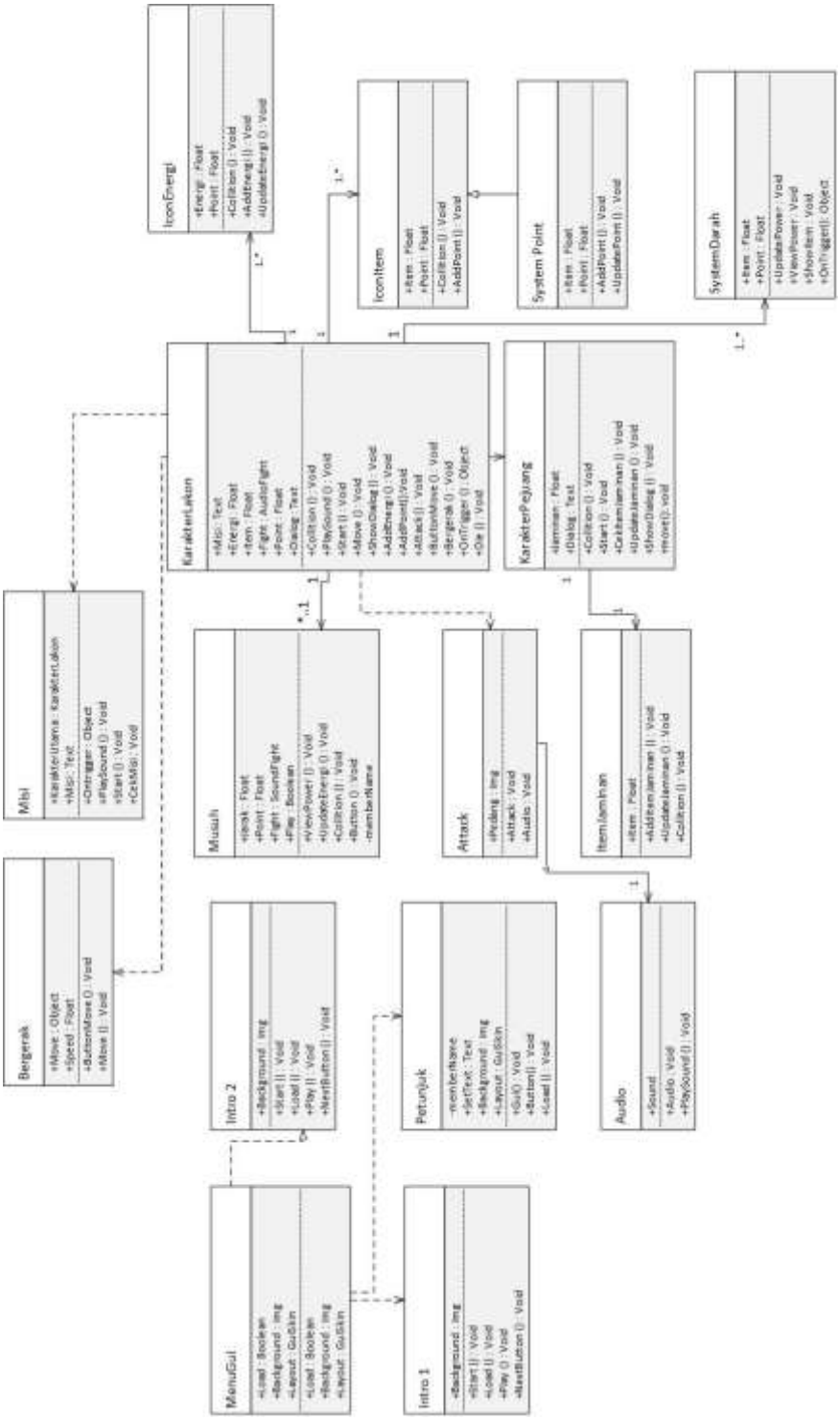
8. *Activity Diagram* Menambah Energi

Proses ini terjadi saat karakter lakon mengambil energi untuk menambah energi miliknya. *Activity diagram* Menambah Energi dapat dilihat pada gambar 3.37.



Gambar 3.37 *Activity Diagram* Menambah Energi

3.1.6.4 Class Diagram

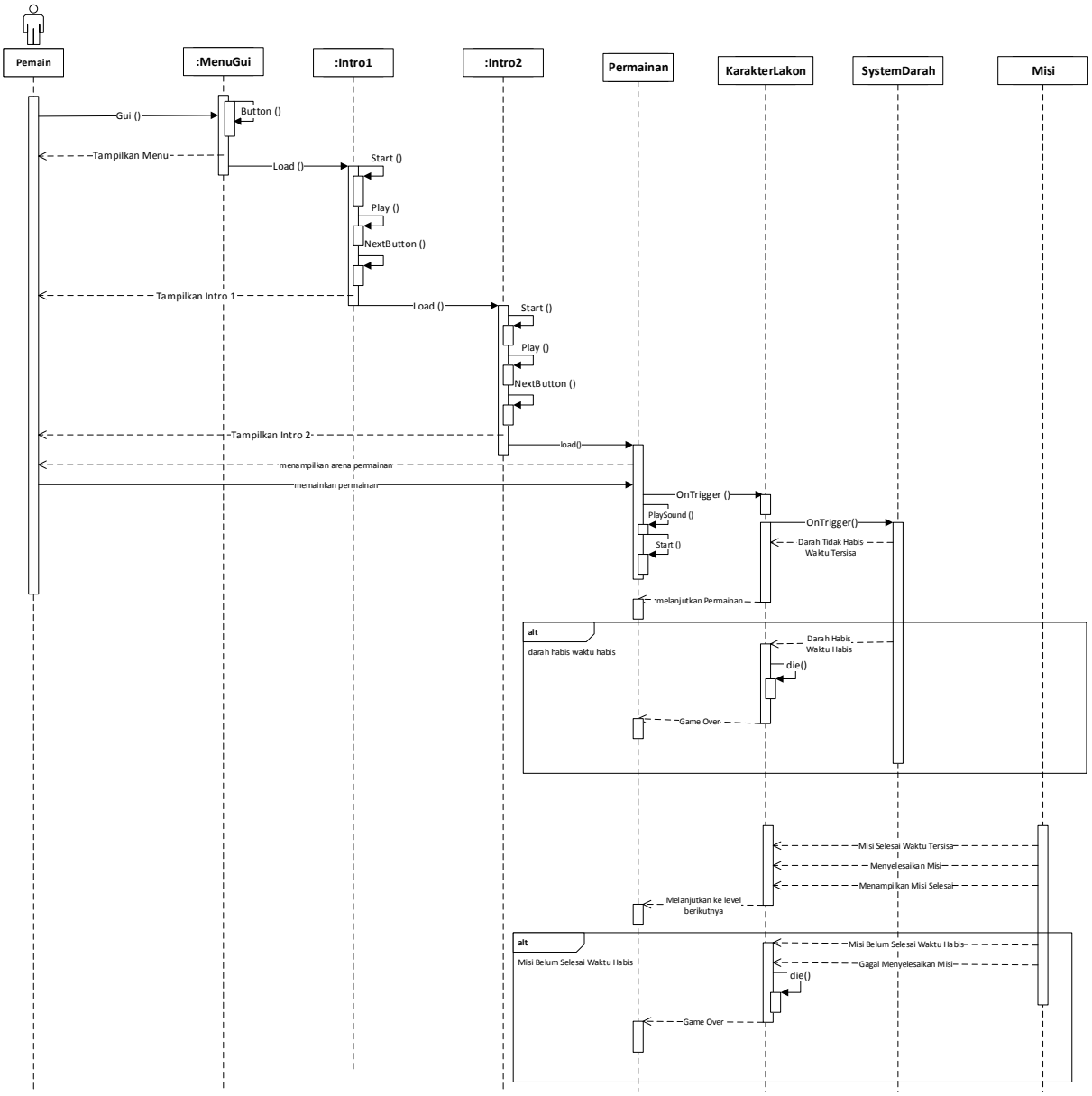


Gambar 3.38 Class Diagram

3.1.6.5 Sequence Diagram

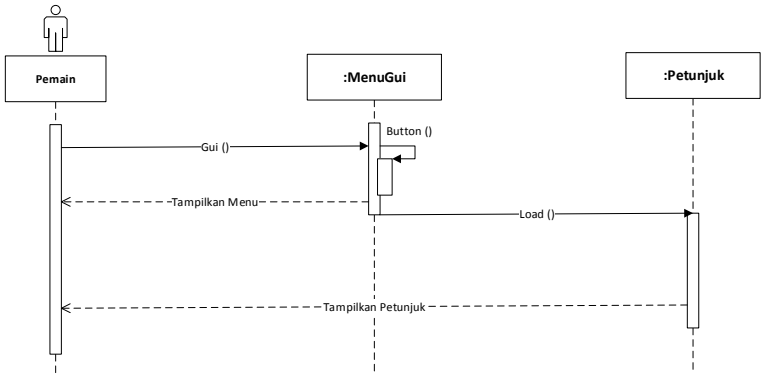
Sequence diagram menggambarkan kelakuan objek pada *use case* dengan mendeskripsikan waktu hidup dan *message* yang dikirimkan dan diterima antarobjek.

1. *Sequence Diagram* Memainkan Permainan



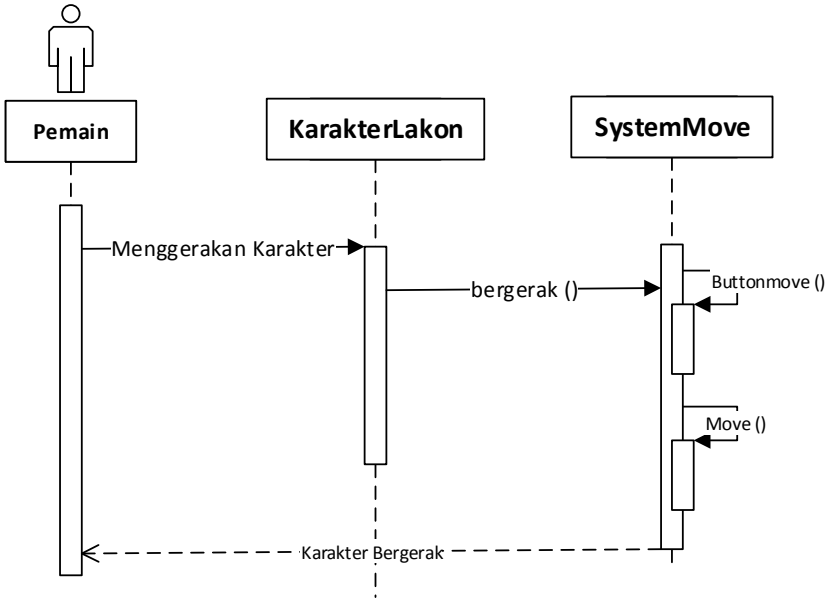
Gambar 3.39 *Sequence Diagram* Memainkan Permainan

2. *Sequence Diagram* Melihat Petunjuk



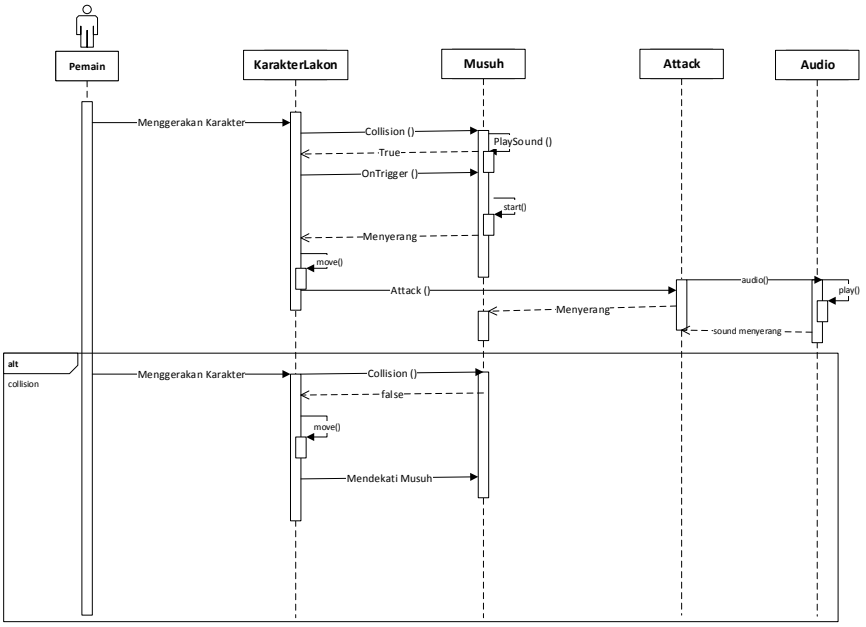
Gambar 3.40 Sequence Diagram Melihat Petunjuk

3. *Sequence Diagram* Menggerakan Karakter Lakon



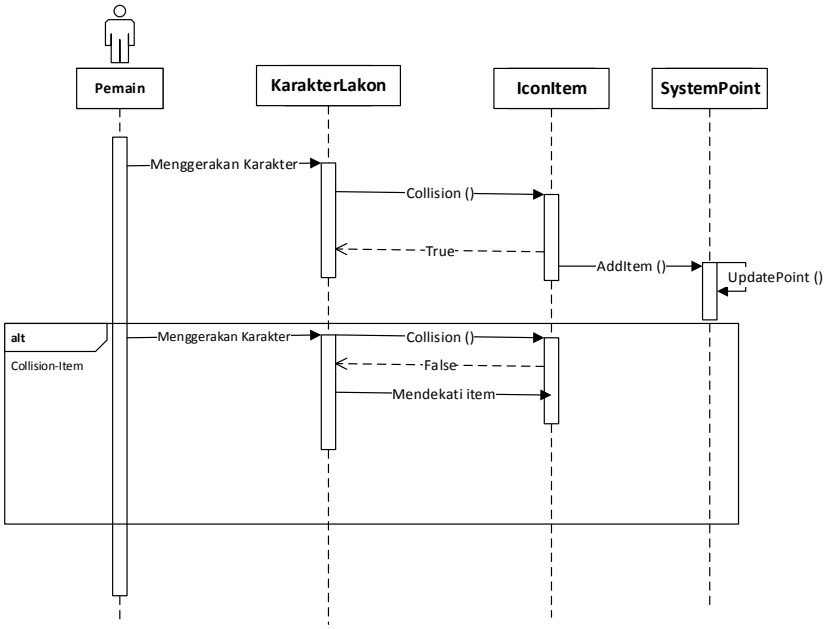
Gambar 3.41 *Sequence Diagram* Menggerakan Karakter Lakon

4. *Sequence Diagram* Menyerang Musuh



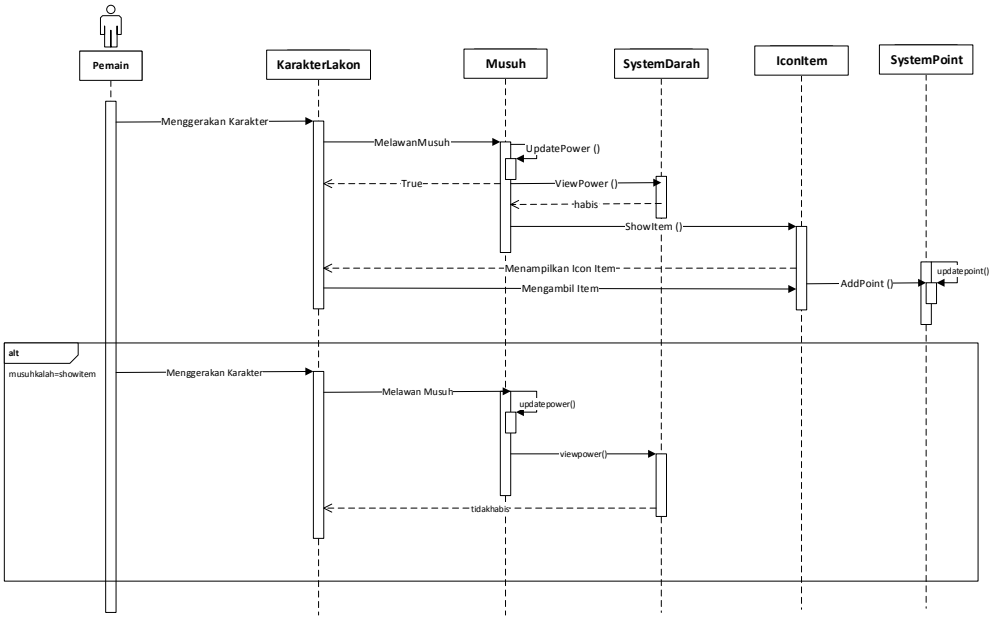
Gambar 3.42 *Sequence Diagram* Menyerang Musuh

5. *Sequence Diagram* Mengambil Item



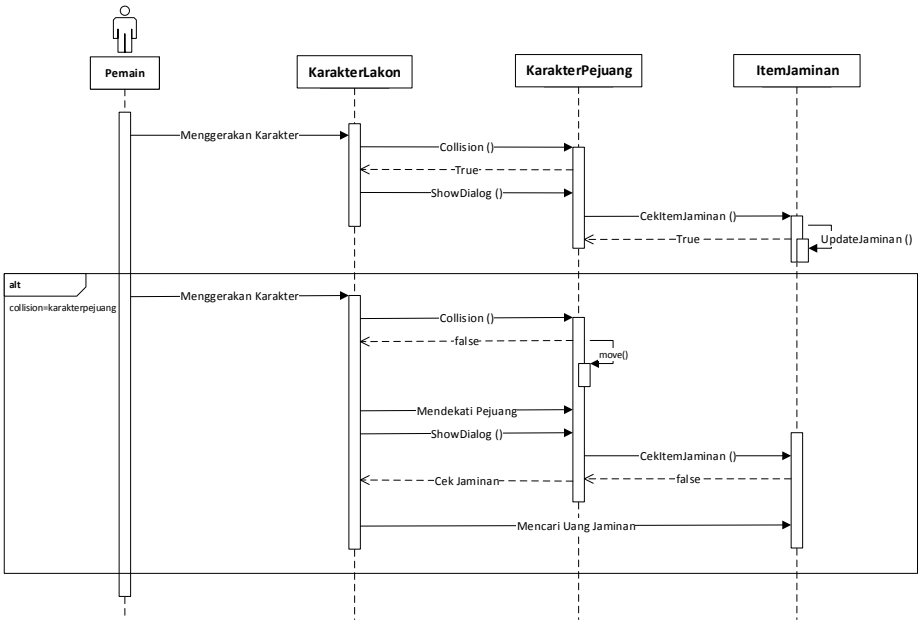
Gambar 3.43 *Diaram* Mengambil Item

6. *Sequence Diagram* Merebut Item



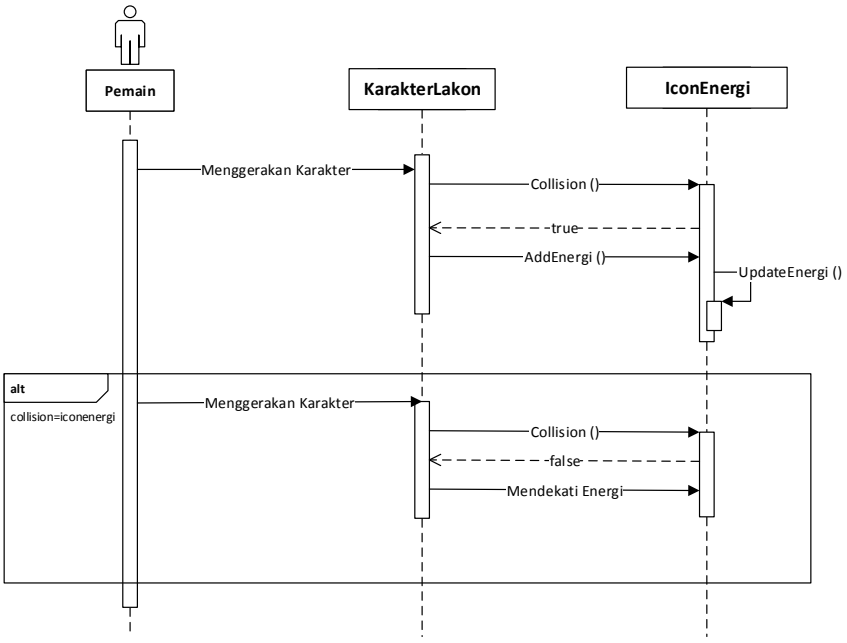
Gambar 3.44 *Sequence Diagram* Merebut Item

7. *Sequence Diagram* Merekrut Pejuang



Gambar 3.45 *Sequence Diagram* Merekrut Pejuang

8. *Sequence Diagram* Menambah Energi



Gambar 3.46 *Sequence Diagram* Menambah Energi

3.2 Perancangan Sistem

3.2.1 Perancangan Komponen Permainan

3.2.1.1 Entitas *Game*

Entitas yang terdapat di dalam *game* terbagi menjadi dua, yaitu karakter Pemain dan NPC yang berupa musuh, selain itu terdapat juga item yang dapat diambil ditengah permainan.

Entitas karakter dan item yang terdapat di dalam *game The Island Colonized* ditunjukkan pada tabel 3.25.

Tabel 3.25 Tabel Entitas *game*

NO	NAMA	GAMBAR	PENJELASAN
1	Ujang		Merupakan karakter utama yang memiliki kemampuan melompat, menghindari musuh, dan menusuk menggunakan pedang untuk

			mengalahkan musuh.
2	Anyang		Anyang merupakan pejuang dari pulau Borneo yang akan membantu pemain mengalahkan musuh dan merebut kembali pulau yang di jajah.
3	Kilangin		Anyang merupakan pejuang dari pulau Nueva Guinee yang akan membantu pemain mengalahkan musuh dan merebut kembali pulau yang di jajah.
4	Semlow		Semlow merupakan musuh pada level 1, Semlow memiliki kemampuan untuk mengejar Pemain jika berada pada radius jarak penglihatannya, melompat dan bertarung dengan Pemain
5	Mavuno		Mavuno merupakan musuh pada level 2, Mavuno memiliki kemampuan untuk menembak dari jarak jauh dengan jumlah yang banyak.
6	Bonnet		Bonnet merupakan musuh pada level 3, Bonnet adalah Bos dari Semlow dan Mavuno, memiliki kemampuan untuk menyerang pemain secara agresif, lebih kuat dari 2 musuh di level sebelumnya, dan harus dikalahkan untuk menyelesaikan permainnan tersebut
7	Hati		Sebuah item yang berfungsi untuk menambah *health point* pemain secara penuh

8	*Capsule*		Sebuah item yang berfungsi untuk menambah setengah *health point* pemain
9	*Coal*		*Coal* merupakan sebuah item yang didapatkan dari musuh yang sudah dikalahkan di pulau Borneo
10	*Gold*		*Gold* merupakan sebuah item yang didapatkan dari musuh yang sudah dikalahkan di pulau Nueva Guinee

3.2.1.2 Storyboard

Storyboard merupakan Sketsa gambar yang disusun sesuai naskah yang ada, dengan *storyboard* kita menyampaikan ide cerita kepada orang lain dengan mudah.

Berikut ini merupakan *storyboard* dari aplikasi *Game The Island Colonized:*

1. *Storyboard* Memulai Permainan

Storyboard pada gambar 3.47 Menceritakan pemain / *player* ketika memulai permainan. Terdapat beberapa pulau, pulau tersebut akan diselamatkan pemain dan merupakan tempat pemain merekrut pejuang.



Gambar 3.47 *Storyboard* Memulai Permainan

2. *Storyboard* Level 1

Pada *level* 1, pemain harus merekrut pejuang dan menyelamatkan pulau Borneo , selain merekrut pejuang pemain juga harus menyelamatkan batu bara yang dicuri oleh kawanan perompak, pemain juga mempunyai misi untuk mengusir perompak dari pulau borneo ini untuk melanjutkan ke level selanjutnya.

Berikut adalah gambar *storyboa*rd level 1 permainan *The Island Colonized* terdapat pada gambar 3.48.



Gambar 3.48 *Storyboard level 1* Borneo

3. *Storyboard* Level 2

Di *level* kedua, pemain harus menyelamatkan pulau Nueva Guinee dan sudah merekrut pejuang karena kawanan perompak sudah menduduki pulau, disini kawanan perompak sudah banyak, misi dari level ini yaitu menghadang perompak membawa emas dan mengusir mereka dari pulau tersebut.

Berikut adalah gambar *storyboa*rd level 2 permainan *The Island Colonized* terdapat pada gambar 3.49.



Gambar 3.49 *level 2* Nueva Guinee

4. *Storyboard* Level 3

Di *level* ketiga, Pemain kembali ke pulaunya dengan membawa bala bantuan, di level ini pemain mempunyai misi mengalahkan seluruh perompak dan Bos perompak untuk mengambil kembali harta kekayaan mereka dan memperoleh kejayaan/kebebasan. Apabila pemain dapat mengalahkan boss maka permainan berakhir dan pemain sebagai pemenang

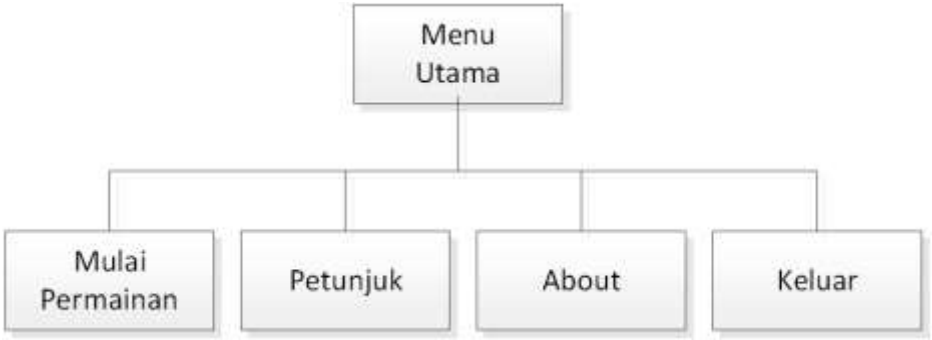
Berikut adalah gambar *storyboa*rd level 3 permainan *The Island Colonized* terdapat pada gambar 3.50.



Gambar 3.50 *Storyboard level 3* Jawa Dwipa

3.2.2 Perancangan Struktur Menu

Untuk mempermudah dalam pembuatan sistem, diperlukan perancangan struktur menu program yang akan dibangun. Perancangan struktur menu program ini membantu dalam merancang bagian-bagian dari sistem yang sebenarnya dan untuk mengetahui bagian mana yang nantinya akan diakses terlebih dahulu setelah program selesai dibuat. Perancangan arsitektur menu pada aplikasi permainan *The Island Colonized* dapat dilihat pada gambar 3.51.



Gambar 3.51 Struktur Menu Permainan *The Island Colonized*

A. Mulai Permainan : Menu untuk memulai *game The Island Colonized.*

B. Petunjuk : Menu untuk menampilkan informasi tentang cara memainkan *game The Island Colonized.*

C. *About* : Menu untuk menamplkan informasi about *game The Island Colonized.*

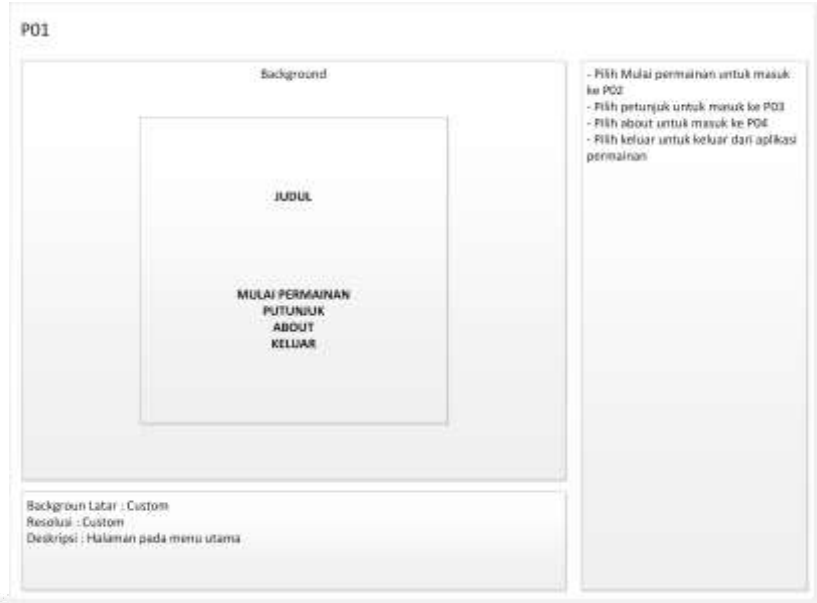
D. Keluar : Menu untuk keluar dari *game The Island Colonized.*

3.2.3 Perancangan Antarmuka

Perancangan antarmuka yang telah dibuat meliputi beberapa perancangan diantaranya perancangan antarmuka menu utama , perancangan antarmuka mulai permainan, perancangan antarmuka petunjuk, dan perancangan antarmuka about.

1. Perancangan Antarmuka Menu Utama

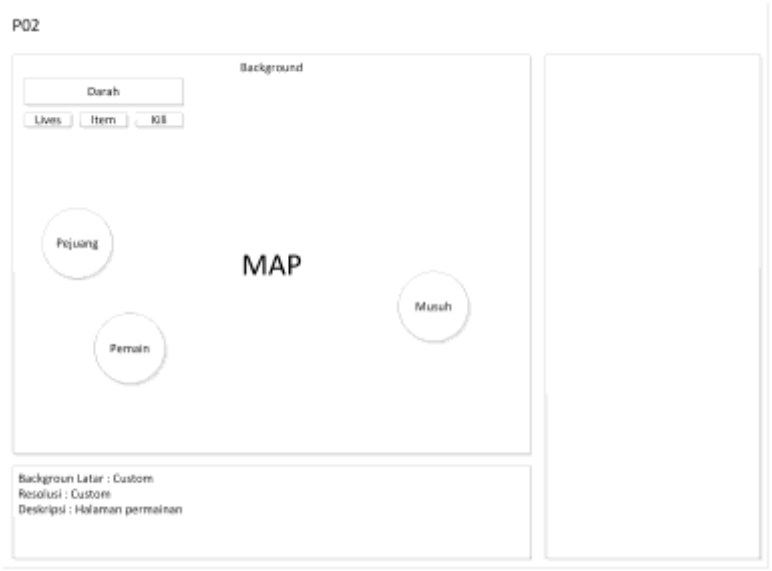
Perancangan antarmuka menu utama pada aplikasi permainan *The Island Colonized* dapat dilihat pada gambar 3.52.



Gambar 3.52 Perancangan Antarmuka Menu Utama

2. Perancangan Antarmuka Menu Mulai Permainan

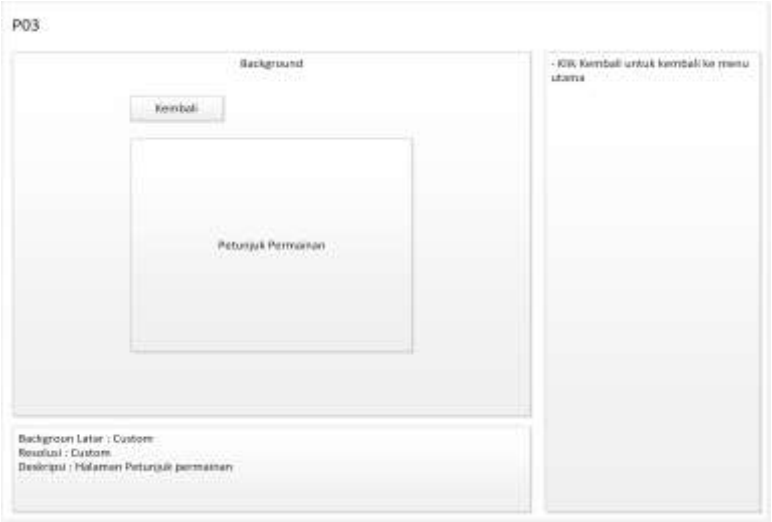
Perancangan antarmuka menu mulai permainan pada aplikasi permainan *The Island Colonized* dapat dilihat pada gambar 3.53.



Gambar 3.53 Perancangan Antarmuka Menu Mulai Permainan

3. Perancangan Antarmuka Menu petunjuk

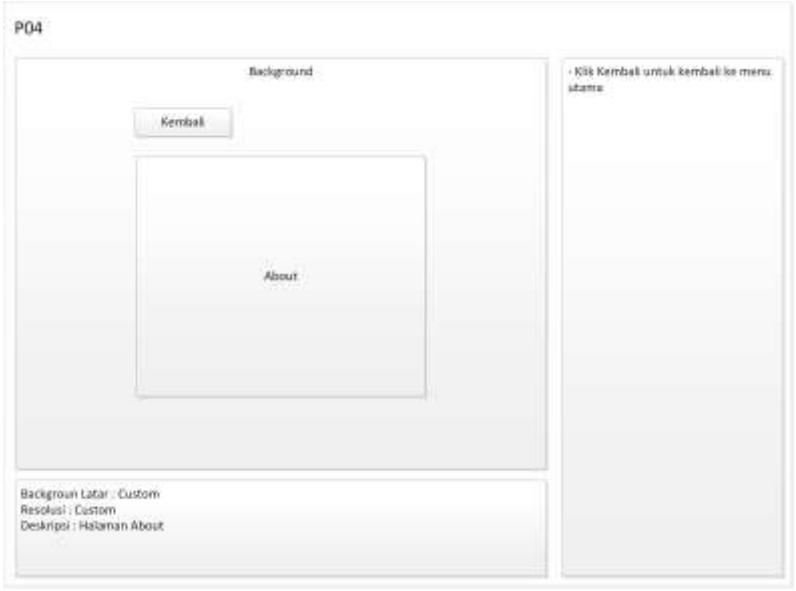
Perancangan antarmuka menu Petunjuk pada aplikasi permainan *The Island Colonized* dapat dilihat pada gambar 3.54.



Gambar 3.54 Perancangan Antarmuka Menu Petunjuk

4. Perancangan Antarmuka Menu *About*

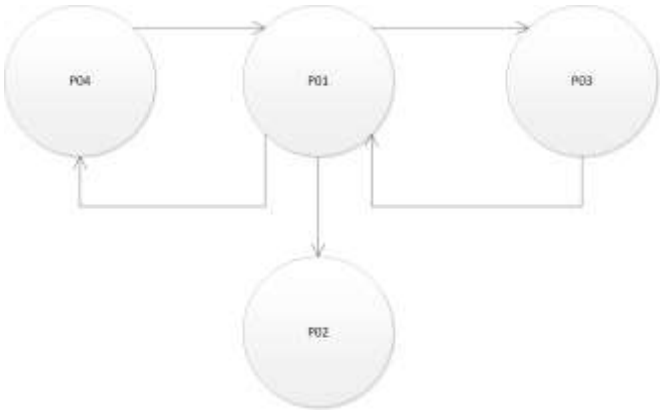
Perancangan antarmuka menu About pada aplikasi permainan *The Island Colonized* dapat dilihat pada gambar 3.55.



Gambar 3.55 Perancangan Antarmuka Menu About

3.2.3 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.



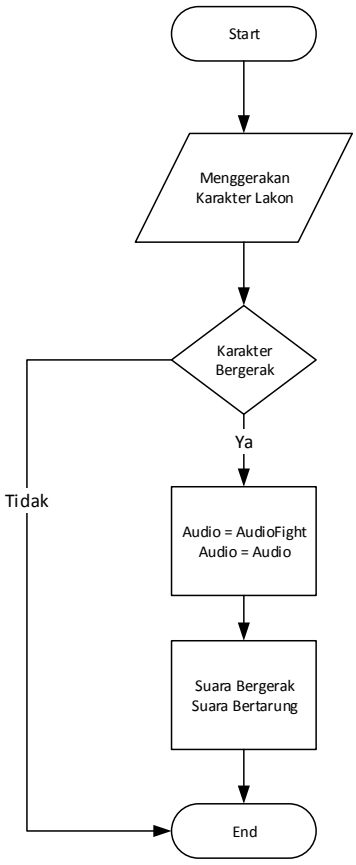
Gambar 3.56 Jaringan Semantik

3.2.4 Perancangan Method

Perancangan *method* merupakan perancangan yang berfungsi untuk mendeskripsikan *method-method* yang berada di dalam aplikasi. Perancangan ini dibuat setelah perancangan struktur menu, antarmuka dan jaringan semantik. Perancangan method berfungsi untuk mendeskripsikan method-method yang berada di dalam aplikasi permainan *The colonized Island*. Adapun method-method yang terdapat dalam aplikasi permainan *The Colonized Island* akan dibangun adalah sebagai berikut :

a. *Method PlaySound ()*

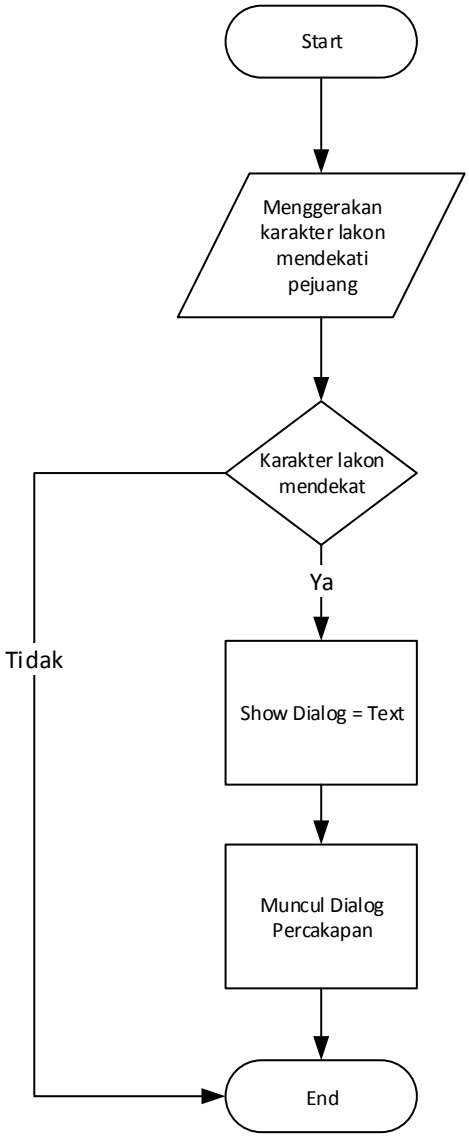
Method ini digunakanuntuk memberi efek suara, dapat dilihat pada gambar 3.57.



Gambar 3.57 *Method PlaySound*

b. *Method ShowDialog ()*

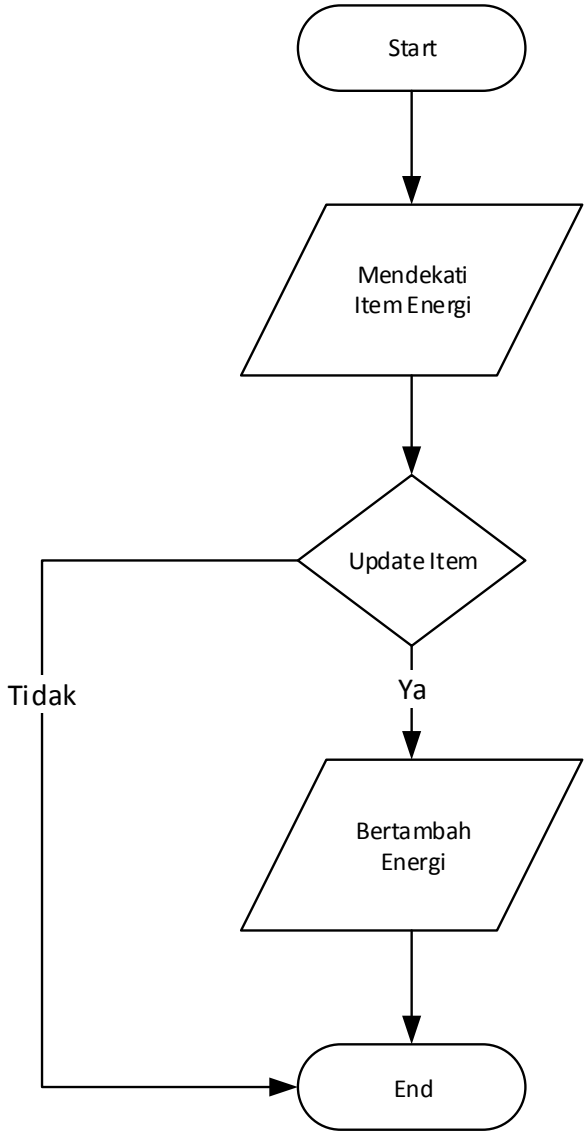
Method ini digunakan saat karakter lakon berdialog dengan karakter pejuang, dapat dilihat pada gambar 3.58.



Gambar 3.58 *Method ShowDialog*

c. Method UpdateEnergi ()

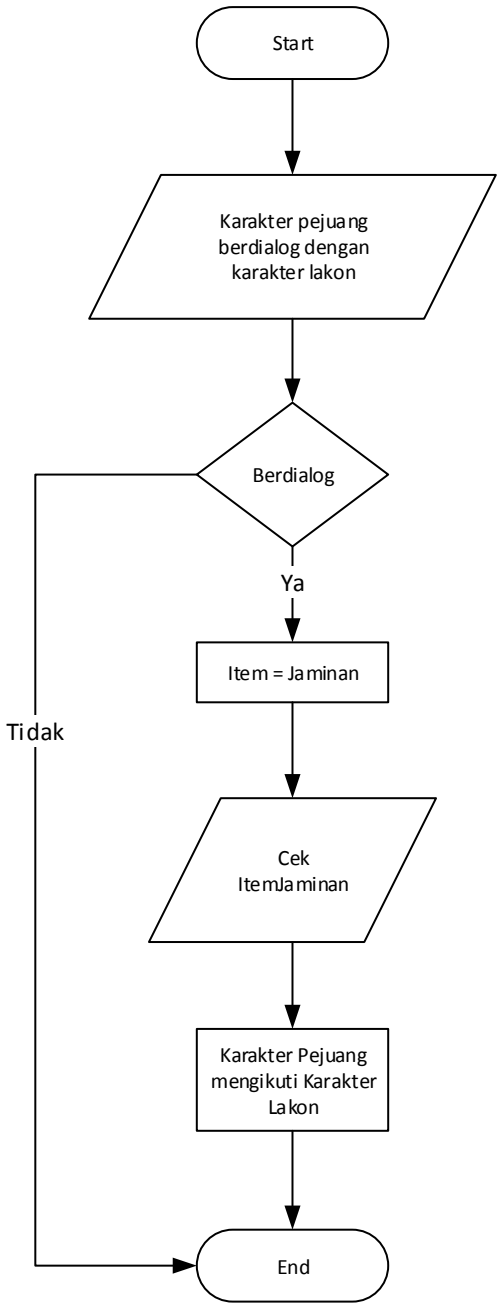
Method ini digunakan untuk menambah energi karakter lakon. Dapat dilihat pada gambar 3.59.



Gambar 3.59 *Method UpdateEnergi*

d. Method UpdateJaminan ()

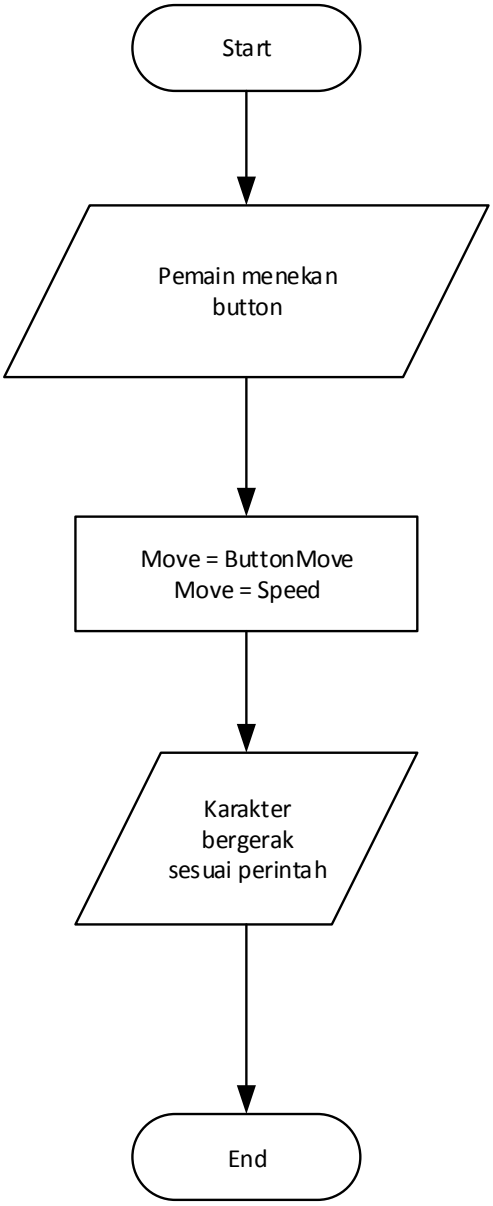
Method ini digunakan untuk mengecek jaminan ketika karakter lakon berdialog dengan karakter pejuang ketika akan merekrut karakter pejuang. Dapat dilihat pada gambar 3.60.



Gambar 3.60 *Method UpdateJaminan*

e. Method Bergerak

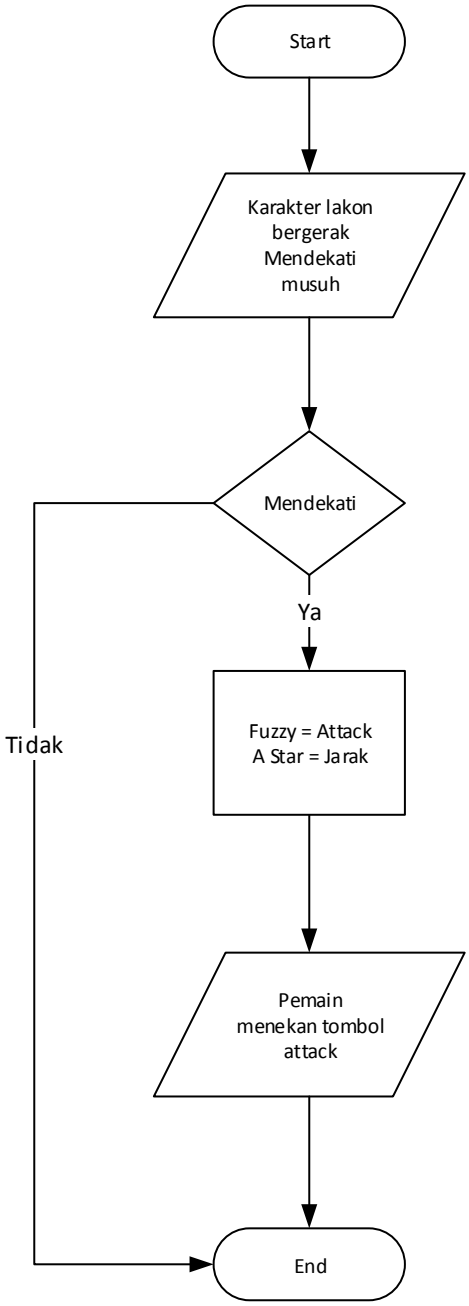
Method ini akan melakukan *update* gerakan terhadap karakter lakon yang diawali dari inputan pemain, dapat dilihat pada gambar 3.61.



Gambar 3.61 *Method Bergerak*

f. Method Attack

Method ini digunakan untuk menyerang musuh oleh karakter lakon. Dapat dilihat pada gambar 3.62



Gambar 3.62 *Method Attack*