BAB 3
ANALISIS DAN PERANCANGAN

3.1 Analisis Sistem

Analisis sistem merupakan penguraian dari suatu sistem informasi yang utuh ke dalam bagian-bagian komponennya, yang dimaksudkan untuk mengevaluasi segala permasalahan dan hambatan yang ada, serta kebutuhan baru yang nantinya dapat menjadi acuan untuk dilakukan perbaikan

3.1.1 Analisis Masalah

Masalah yang melatarbelakangi dibuatnya *game* cerita rakyat berjudul kisah budak pahatu lalis adalah :

1. Telah hilangnya cerita rakyat berjudul kisah budak pahatu lalis
2. Cerita rakyat berjudul kisah budak pahatu lalis memiliki banyak pesan moral yang amat baik untuk dibaca oleh semua orang khususnya anak-anak
3. Kurangnya perhatian masyarkat dalam menjaga dan melestarikan budaya bangsa

3.1.2 Analisis *Game* Sejenis

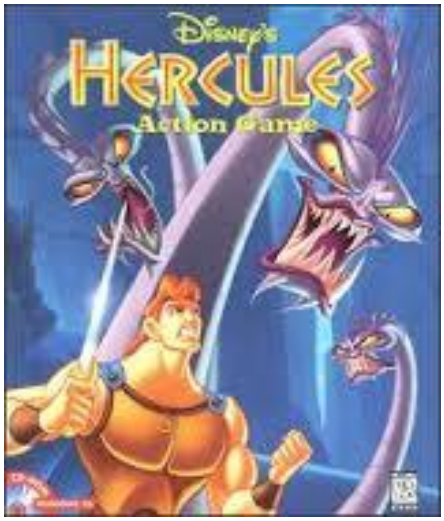
Pada analisis *game* sejenis, akan dilakukan observasi terhadap *game* yang memiliki kesamaan *gameplay* dan *genre* dengan *game* yang akan dibangun, tujuannya agar mendapat perbandingan dari *game* sejenis mengenai kelebihan dan kekurangannya *Game* yang jadi pembanding adalah *game* berjudul Hercules *Adventure*.

1. Kemiripan *game* Hercules *Adventure* dengan *game* Budak Pahatu Lalis

   Kesamaan yang paling kuat yang membuat *game* Hercules *Adventure* menjadi pembanding dengan *game* kisah budak pahatu lalis adalah kesamaan tokoh dan alur cerita yang sama-sama diangkat dari sebuah cerita atau dongeng dimana Hercules merupakan tokoh yang terdapat pada dongeng mitologi yunani, sama dengan kisah budak pahatu lalis yang merupakan dongeng atau cerita rakyat dari Jawa Barat

2. Pengenalan *game* Hercules *Adventure*

   Game yang menjadi pembanding adalah *game* dari perusahaan *video game* besar SONY PLAYSTATION yaitu *game* Hercules *Adventure, game* Hercules *Adventure* merupakan *game* 3D namun memiliki objek berupa gambar kartun.



Gambar 3.1 *Game* Hercules *Adventure*

3. *Gameplay game* Hercules *Adventure*

   Gameplay game Hercules *Adventure* yaitu pemain hanya menggunakan satu karakter yaitu tokoh Hercules dari awal sampai akhir, pergerakan yang dapat dilakukan yaitu tokoh dapat bergerak secara bebas ke 4 arah (→,←,↑,↓), ditengah petualangan tokoh menemukan musuh dan langsung *fighting,* dalam

perjalanan tokoh dapat menemukan *health* atau fasilitas untuk menambah nyawa, senjata lebih hebat, dan lain-lain.

4. *Scenario* dan konten *game* Hercules *Adventure*
    a. Pada *stage* pertama pemain yang memainkan tokoh Hercules berada pada tempat dimana pemain belum mempunyai senjata apa-apa hanya bermodalkan tangan kosong untuk melawan musuh. Disini pemain dituntut untuk mencari senjata
    b. Pada *stage* kedua pemain mulai mencari jalan terhadap misi-misi yang ada pada *stage* final
    c. Musuh yang dihadapi *stage* demi *stage* semakin berat, untuk itu pemain dituntut untuk banyak mencari senjata, *health* (fasilitas nyawa)
    d. Disamping *stage* inti terdapat juga *stage* bonus untuk mengumpulkan koin
    e. Diakhir *stage* pemain mendapatkan tontonan seperti gerakan-gerakan animasi baru yang lucu yang dikeluarkan melalui karakter Hercules

5. Kekurangan *Game* Hercules *Adventure*

    Kekurangan yang terdapat pada *game* ini yaitu inti cerita yang diusung kurang jelas, awalnya *public* mengira ini merupakan *game* yang mengangkat cerita tentang Hercules yaitu tokoh orang yang sangat kuat dalam *mitologi* Yunani karena didalam cerita banyak tokoh-tokoh yang ada juga pada cerita Hercules seperti Medusa, Hera, Zeus, dan Puteri yang harus diselamatkan (yang merupakan teman dekat Hercules dalam cerita) namun cerita pada *game* sangatlah berbeda dengan cerita pada cerita sebenarnya maka disinilah kekurangan *game* ini yang kurang menjelaskan inti cerita padahal didalamnya banyak tokoh-tokoh yang juga ada pada dongeng namun *storyline* pada *game* kurang menceritakan sesuai isi cerita aslinya

3.1.3 Analisis *Game* Budak Pahatu Lalis

1. *Storyline*

Cerita yang diangkat ada didalam *game* diangkat sepenuhnya dari cerita rakyat berjudul "kisah budak pahatu lalis". Adapun *game* dibagi menjadi 3 *stage*

a. *Stage* 1 pemain dituntut untuk mencari bahan makanan untuk tokoh sang adik didalam cerita yang merupakan pesan moral seorang kakak yang (Sesuai pesan moral yang disampaikan dalam cerita)
b. *Stage* 2 pemain dituntut melawan musuh-musuh yang datang mengganggu tokoh sang adik didalam cerita yang merupakan pesan moral seorang kakak yang selalu melindungi adiknya dari semua yang mengganggunya (Sesuai pesan moral yang disampaikan dalam cerita)
c. *Stage* 3 pemain dituntut untuk mencari ular yang telah memakan adiknya kemudian melawan ular tersebut , jika menang maka pemain harus mencari pisau yang terbuat dari bambu untuk mengeluarkan sang adik dari perut ular

2. *Gameplay*

Gameplay adalah aturan yang ada didalam *game* yang menjadi batasan pemain dalam menyelesaikan semua misi yang ada didalam *game* [1]

Gameplay yang dibuat pada *game* budak pahatu lalis adalah sebagai berikut :

a. Pemain akan memainkan tokoh sang kakak didalam cerita sebagai karakter yang dimainkan didalam *game*
b. Aplikasi *game* yang akan dibangun ber-*genre adventure* TPS (*Third Person Shooter*) Posisi kamera akan menyorot dari belakang tokoh *controller* yaitu tokoh sang kakak yang dimainkan oleh pemain
c. Pemain dapat bergerak bebas maju ke 4 arah mata angin sambil menjalankan misi yang ada

d. Sebelum mulai bermain pemain akan masuk pada bagian *intro* atau pengenalan tentang cerita "Pada jaman dahulu hidup seorang anak yatim piatu bla..bla..blaa……"

e. *Stage* pertama pemain diberi tantangan mencari sumber makanan untuk sang adik, dengan nyawa sang adik yang semakin berkurang jika terlalu lama

f. *Stage* kedua pemain harus melawan semua musuh yang datang untuk mengganggu sang adik, disini pemain dituntut kecepatan agar nyawa sang adik tidak habis karena musuh yang datang akan membuat nyawa sang adik berkurang dan bila keburu habis maka permainan berakhir

g. *Stage* terakhir pemain akan berjalan-jalan bersama sang adik yang tanpa disadari ada ular besar yang datang menghampiri sang adik lalu dengan cepat memakannya kemudian ular itu pergi menjauh, disini pemain dituntut untuk mencari ular karena nyawa sang adik akan terus berkurang setelah dimakan, dan setelah bertemu pemain harus segera melawannya sambil ular pun melawan disini pemain pun akan berkurang nyawanya

3. Leveling

Leveling (Perbedaan tingkat kesulitan) merupakan tantangan yang ada didalam *game* yang harus dicapai oleh pemain untuk menyelesaikan misinya. Adapun tingkat kesulitan yang ada pada *game* kisah budak pahatu lalis dari *stage* 1 sampai *stage* 3 adalah:

a. Tantangan pada *stage* pertama adalah mencari bahan makanan, disini pemain harus berlomba dengan nyawa dari sang adik yang terus berkurang dari waktu ke waktu dan akan selesai ketika target makanan sesuai jumlah yang ditentukan telah terpenuhi

b. Target pada *stage* kedua adalah melawan semua karakter yang datang untuk mengurangi nyawa sang adik, disini pemain dituntut kecepatan, karena

monster yang datang banyak, dan akan selesai ketika pemain berhasil melawan semua musuh sesuai jumlah yang ditentukan

c. Target pada *stage* ketiga yaitu mencari ular yang telah memakan sang adik kemudian melawanna sebelum nyawa sang adik habis apabila pemain tidak cepat maka nyawa sang adik habis dan permainan berakhir dan ular pun melawan maka pemain juga harus memperhatikan nyawanya jika nyawa pemain pun habis maka permainan pun akan berakhir.

4. *Scoring*

Scoring (Pengelolaan skor, hukuman, dan sebagainya) yang terdapat pada *game* budak pahatu lalis adalah

Ketika sang pemain telah selesai menyelesaikan *stage* per *stage* maka pemain dapat melanjutkan ke level berikutnya sekaligus mendapatkan pesan moral yang tersurat berupa pujian kepada sang tokoh yang dapat dibaca, tergantung di level berapa dia menyelesaikannya dan setelah pemain menyelesaikan semua stage maka cerita keseluruhan tentang cerita rakyat kisah budak pahatu lalis akan muncul dilayar, jika pemain tidak memenuhi tuntutan maka pemain akan keluar dari permainan dan permainan berakhir dan akan mendapat kata-kata berupa “Anak yang kurang bertanggung jawab”, “Anak yang pengecut” tergantung pada *stage* berapa dia tidak dapat menyelesaikannya

3.1.4 Analisis Algoritma

Algoritma yang dipakai didalam *game* adalah algoritma pencarian terbimbing atau *heuristic search* dan algortima yang sesuai dengan kasus pada *game* adalah algoritma A*

Algoritma A* di dalam *game* akan diterapkan pada *Enemy* atau musuh sebagai penentuan jalur terpendek atau *pathfinding* menuju target yaitu karakter utama.

Cara kerja algoritma A* di dalam *game* budak pahatu lalis dijelaskan pada gambar 3.2 sebagai berikut :

(0.3)	(1.3)	(2.3)	(3.3)	(4.3)	(5.3)	(6.3)
(0.2) **I**	(1.2) **A**	(2.2) **K**	(3.2)	(4.2)	(5.2)	(6.2)
(0.1) **J**	(1.1)	(2.1) **L**	(3.1)	(4.1)	(5.1)	(6.1)
F (0.0) G H	(1.0)	(2.0)	(3.0)	(4.0) **B**	(5.0)	(6.0)

Gambar 3.2 Ilustrasi Algortima A*

Penjelasan :

1. *Node* A : Merupakan *node* awal tempat dimana posisi musuh berada yang berada pada koordinat (1.2) dan ditandai warna merah
2. *Node* B : Merupakan *node* tujuan tempat dimana posisi karakter utama berada yang berada pada koordinat (4.0) dan ditandai warna hijau
3. *Node* I,J,K,L : Merupakan *node* penghalang atau *obstacle* yang tidak dapat dilewati oleh karakter musuh. Node ini berada pada koordinat (0.1), (0.2), (2.1), (2.2) dan ditandai warna biru

Untuk perhitungan pada algoritma A* dilakukan dengan perhitungan rumus F = G+H, dimana ketiga *variable* tersebut adalah :

1. *Variable* F : Merupakan *variable* nilai yang hasilnya menjadi penentu *node* untuk dipilih, *node* dengan nilai F terkecil merupakan *node* yang terpilih sebagai *current node*

2. *Variable* G : Merupakan nilai atau biaya yang dikeluarkan dari *node* awal menuju *node* sekitarnya nilai *vertical* dan *horizontal* bernilai 10 sementara nilai diagonal bernilai $10\sqrt{2}$ atau digenapkan menjadi 14

3. *Variable* H : Merupakan nilai *heuristic* yang digunakan untuk menghitung perkiraan biaya yang dikeluarkan dari *node* awal atau *node* A menuju *node* tujuan atau *node* B. 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 dirumuskan dengan: H = (abs(*current*X-*target*X) + abs(*current*Y-*target*Y)). Pada analisis yang dilakukan penulis nilai *Heuristic* yang didapat akan dikalikan 10 untuk membuat perhitungan lebih *real*

Pada pencarian jalur terpendek oleh A *star* ada beberapa istilah yang digunakan seperti *open list, closed list, current node*. Berikut penjelasannya :

1. *Open List* : Merupakan *node neighbor* atau *node* yang dapat dikunjungi dari posisi *closed list* untuk mencapai tujuan

2. *Current Node* : Merupakan *node* yang terpilih berdasarkan nilai F terkecil diantara *node* lain yang terdapat pada *open list, current node ditandai* dengan warna abu-abu pada gambar

3. *Closed List* : Merupakan kumpulan *node* yang telah dipilih untuk mencapai tujuan, *closed list* ditandai dengan warna merah pada gambar dan *node* yang

telah berada pada *close list* tidak dapat dikunjungi kembali pada iterasi berikutnya

Selanjutnya adalah melakukan perhitungan untuk menemukan jalan dengan cara menghitung *node-node* yang berdekatan dari *node* A sampai *node* B. Langkah-langkahnya adalah sebagai berikut:

1. Tetapkan *node* awal tempat *node* akan bergerak menuju *node* tujuan disini *node* A merupakan *node* awal menuju *node* B
2. Periksa apakah *current node* sama dengan *node* B, atau *current node* yang sudah didapatkan merupakan *node* tujuan. Jika iya maka jalan sudah ditemukan. (Proses selesai) jika tidak maka periksa semua *node* yang berdekatan dengan *node* A dan masukan kedalam *open list* kemudian lakukan perhitungan F.
3. Lakukan perhitungan F terhadap semua *node* disekeliling *node* A yang akan dijadikan *current node*, *node* dengan nilai F terkecil menjadi pilihan
4. Tetapkan *node* dengan nilai F terkecil sebagai *current node* sampai menuju target yang dicapai

Untuk lebih jelasnya proses perhitungan dilakukan per-iterasi dan dijelaskan sebagai berikut.

Iterasi Pertama :

84 (0.3) 14 ↘ 70	70 (1.3) 10 ↓ 60	64 (2.3) 14 ↙ 50	(3.3)	(4.3)	(5.3)	(6.3)
(0.2) **I**	50 (1.2) **0 A 50**	(2.2) **K**	(3.2)	(4.2)	(5.2)	(6.2)
(0.1) **J**	50 (1.1) 10 ↑ 40	(2.1) **L**	(3.1)	(4.1)	(5.1)	(6.1)
(0.0)	(1.0)	(2.0)	(3.0)	(4.0) **B**	(5.0)	F (6.0) G H

Gambar 3.3 Ilustrasi Iterasi Pertama

1. Tetapkan *node* A (1.2) sebagai *node* awal menuju *node* B (4.0) yang merupakan *node* tujuan
2. Periksa semua *node* yang berdekatan dengan *node* (1.2) tempat A berada disini terdapat *node* (0.3), (1.1), (1.3), (2.3) dan *node* (0.1), (0.2), (2.1), (2.2) tidak dapat dijadikan *node open list* karena *node* tersebut merupakan penghalang sehingga tidak dapat dilewati
3. Lakukan perhitungan F pada setiap *node* yang berdekatan dengan *node* A
   a. *Node* (0.3) : F = G+H

      G = 14 (*Diagonnal*)

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

      = (abs(0-4) + abs(3-0)) * 10

      = (abs(-4) + abs(3)) * 10

      = 70

      F = 14 + 70

      F *Node* (0.3) = 84

b. *Node* (1.1) : F = G+H

G = 10 (*Vertical*)

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

= (abs(1-4) + abs(1-0)) * 10

= (abs(-3) + abs(1)) * 10

= 40

F = 10 + 40

F *Node* (1.1) = 50

c. *Node* (1.3) : F = G+H

G = 10 (*Horizontal*)

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

= (abs(1-4) + abs(3-0)) * 10

= (abs(-3) + abs(3)) * 10

= 60

F = 10 + 60

F *Node* (1.3) = 70

d. *Node* (2.3) : F = G+H

G = 14 (*Diagonnal*)

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

= (abs(2-4) + abs(3-0)) * 10

= (abs(-2) + abs(3)) *10

= 50

F = 14 + 50

F *Node* (2.3) = 64

4. Node (1.1) merupakan *current node* berdasarkan nilai F terkecil yaitu : 50

Tabel 3.1 Perhitungan Iterasi Pertama

Open List	*Current Node*	*Closed List*
(0.3)	(1.1)	(1.2)
(1.1)		
(1.3)		
(2.3)		

Iterasi Kedua :

(0.3)	(1.3)	(2.3)	(3.3)	(4.3)	(5.3)	(6.3)
(0.2) **I**	50 (1.2) **0** **50**	(2.2) **K**	(3.2)	(4.2)	(5.2)	(6.2)
(0.1) **J**	50 (1.1) 10 **A** 40	(2.1) **L**	(3.1)	(4.1)	(5.1)	(6.1)
64 (0.0) 24 ↗ 40	50 (1.0) 20 ↑ 30	44 (2.0) 24 ↖ 20	(3.0)	(4.0) **B**	(5.0)	F (6.0) G H

Gambar 3.4 Ilustrasi Iterasi Kedua

1. Tetapkan *node* (1.1) sebagai *node* awal menuju *node* B (4.0) yang merupakan *node* tujuan, dan sekarang *node* A adalah *node* (1.1)
2. Periksa semua *node* yang berdekatan dengan *node* (1.1) tempat A berada disini terdapat *node* (0.0), (1.0), (2.0) dan *node* (0.1), (0.2), (2.1), (2.2) tidak dapat dijadikan *node open list* karena *node* tersebut merupakan penghalang sehingga tidak dapat dilewati
3. Lakukan perhitungan F pada setiap *node* yang berdekatan dengan *node* A
   a. *Node* (0.0) : F = G+H

      G = 24 (*Diagonnal*) dihitung berdasarkan nilai sebelumnya 10 + 14

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

= (abs(0-4)+abs(0-0)) * 10

= (abs(-4)+ 0) * 10

= 40

F = 24 + 40

F *Node* (3.0) = 64

b. *Node* (1.0) : F = G+H

G = 20 (*Vertical*) dihitung berdasarkan nilai sebelumnya 10 + 10

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

= (abs(1-4) + abs(0-0)) * 10

= (abs(-3) + 0) * 10

= 30

F = 20 + 30

F *Node* (3.1) = 50

c. *Node* (2.0) : F = G+H

G = 24 (*Diagonnal*) dihitung berdasarkan nilai sebelumnya 10 + 14

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

= (abs(2-4) + abs(0-0) * 10

= (abs(-2) + 0) * 10

= 20

F = 24 + 20

F *Node* (2.0) = 44

4. Node (2.0) merupakan *current node* berdasarkan nilai F terkecil yaitu : 44

Tabel 3.2 Perhitungan Iterasi Kedua

Open List	*Current Node*	*Closed List*
(0.0)	(2.0)	(1.1)
(1.0)		(1.2)
(2.0)		

Iterasi Ketiga :

(0.3)	(1.3)	(2.3)	(3.3)	(4.3)	(5.3)	(6.3)
(0.2) **I**	50 (1.2) **0** **50**	(2.2) **K**	(3.2)	(4.2)	(5.2)	(6.2)
(0.1) **J**	50 (1.1) 10 40	(2.1) **L**	58 (3.1) 38 ↙ 20	(4.1)	(5.1)	(6.1)
(0.0)	64 (1.0) 34 → 30	44 (2.0) 24 **A** 20	44 (3.0) 34 ← 10	(4.0) **B**	(5.0)	F (6.0) G H

Gambar 3.5 Ilustrasi Iterasi Ketiga

1. Tetapkan *node* (2.0) sebagai *node* awal menuju *node* B (4.0) yang merupakan *node* tujuan, dan sekarang *node* A adalah *node* (2.0)
2. Periksa semua *node* yang berdekatan dengan *node* (2.0) tempat A berada disini terdapat *node* (1.0), (3.0), (3.1) dan *node* (2.1) tidak dapat dijadikan *node open list* karena *node* tersebut merupakan penghalang sehingga tidak dapat dilewati
3. Lakukan perhitungan F pada setiap *node* yang berdekatan dengan *node* A
   a. *Node* (1.0) : F = G+H

      G = 34 (*Vertical*) dihitung berdasarkan nilai sebelumnya 10 + 24

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

=(abs(1-4) + abs(0-0)) * 10

= (abs(-3) + 0) * 10

= 30

F = 34 + 30

F *Node* (1.0) = 64

b. *Node* (3.0) : F = G+H

G = 34 (*Vertical*) dihitung berdasarkan nilai sebelumnya 10 + 24

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

= (abs(3-4) + (abs(0-0) * 10

= (abs(-1) + 0) * 10

= 10

F = 34 + 10

F *Node* (3.0) = 44

c. *Node* (3.1) : F = G+H

G = 38 (*Diagonnal*) dihitung berdasarkan nilai sebelumnya 14 + 24

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

= (abs(3-4) + abs(1-0)) * 10

= (abs(-1) + abs(1)) * 10

= 20

F = 38 + 20

F *Node* (3.1) = 58

4. Node (3.0) merupakan *current node* berdasarkan nilai F terkecil yaitu : 44

Tabel 3.3 Perhitungan Iterasi Ketiga

Open List	*Current Node*	*Closed List*
(1.0)	(3.0)	(1.1)
(3.0)		(1.2)
(3.1)		(2.0)

Iterasi Keempat :

(0.3)	(1.3)	(2.3)	(3.3)	(4.3)	(5.3)	(6.3)
(0.2) **I**	50 (1.2) **0** **50**	(2.2) **K**	(3.2)	(4.2)	(5.2)	(6.2)
(0.1) **J**	50 (1.1) 10 40	(2.1) **L**	64 (3.1) 44 ↓ 20	58 (4.1) 48 ↙ 10	(5.1)	(6.1)
(0.0)	(1.0)	44 (2.0) 24 20	44 (3.0) 34 **A** 10	44 (4.0) 44 **B** **0**	(5.0)	F (6.0) G H

Gambar 3.6 Ilustrasi Iterasi Keempat

1. Tetapkan *node* (3.3) sebagai *node* awal menuju *node* B (3.4) yang merupakan *node* tujuan, dan sekarang *node* A adalah *node* (3.3)
2. Periksa semua *node* yang berdekatan dengan *node* (3.3) tempat A berada disini terdapat *node* (2.3), (2.4),(3.4). *Node* (3.2) tidak dapat dilalui karena telah masuk kedalam *closed list*. Karena tujuan telah ditemukan maka node berikutnya adalah *node* (3.4) dan proses selesai

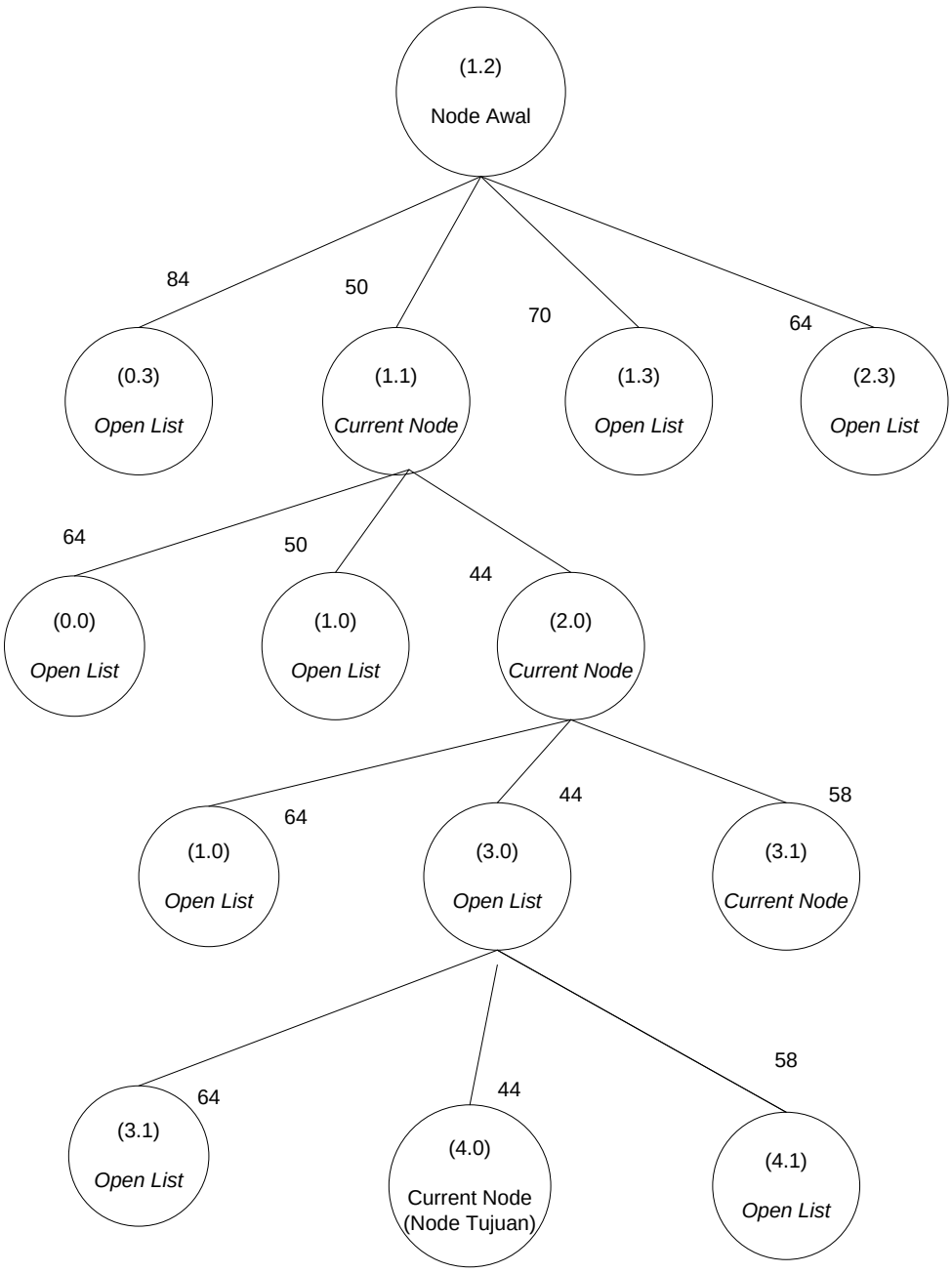
Tabel 3.4 Perhitungan Iterasi Keempat

Open List	*Current Node*	*Closed List*
(3.1)	(3.4) *Node* Tujuan	(1.1)
(4.1)		(1.2)
(4.0)		(2.0)
		(3.0)

Hasil Akhir :

(0.3)	(1.3)	(2.3)	(3.3)	(4.3)	(5.3)	(6.3)
(0.2) **I**	50 (1.2) **0** **50**	(2.2) **K**	(3.2)	(4.2)	(5.2)	(6.2)
(0.1) **J**	50 (1.1) 10 40	(2.1) **L**	(3.1)	(4.1)	(5.1)	(6.1)
(0.0)	(1.0)	44 (2.0) 24 20	44 (3.0) 34 **A** 10	44 (4.0) 44 **B** **0**	(5.0)	F (6.0) G H

Gambar 3.7 Ilustrasi Akhir Dari Penerapan Algoritma



Gambar 3.8 *Tree* Ilustrasi Perhitungan A*

3.2 Analisis Kebutuhan Non-Fungsional

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

3.2.1 Analisis Kebutuhan Perangkat Keras

Perangkat keras atau *hardware* merupakan salah satu hal yang penting karena tanpa *hardware* yang memenuhi syarat, program yang akan dibuat tidak akan dapat berjalan. Berikut spesifikasi standar perangkat keras yang digunakan penulis dalam membangun aplikasi *game*, serta spesifikasi standard untuk dapat memainkan *game* kisah budak pahatu lalis ini.

Spesifikasi perangkat keras dalam membangun *game* tersebut adalah :

1. Processor Intel Core i5 M 460 2.6 GHz
2. RAM 2 GB
3. VGA 1GB
4. Harddisk 500 GB
5. Monitor
6. Mouse, Keyboard, and Speaker

Spesifikasi standard minimal perangkat keras untuk menjalankan aplikasi adalah :

1. Processor Dual Core 1.8 GHz
2. RAM 512 GB
3. VGA 512 MB
4. Harddisk 300 MB (*Free*)

5. Monitor
6. Keyboard and Mouse
7. Speaker (*Optional*)

3.2.3 Analisis Kebutuhan Perangkat Lunak

Perangkat lunak atau *software* merupakan hal terpenting dalam mendukung kinerja sebuah sistem. Perangkat lunak digunakan dalam sebuah sistem untuk memberikan perangkat keras sebuah perintah agar perangkat keras melakukan apa yang di inginkan oleh user. Berikut merupakan analisis perangkat lunak yang digunakan untuk membangun aplikasi *game* dan perangkat lunak minimal yang dibutuhkan dalam menjalankan aplikasi *game*

Spesifikasi perangkat lunak dalam membangun *game* tersebut adalah :

1. Sistem Operasi Windows 7 Home Premium 64-bit
2. Unity Pro Versi 4.1.5
3. 3DS Max 2011
4. iClone 3DXchange v5.5 Pipeline Monitor
5. Graphic Card
6. DirectX 9

Spesifikasi standar minimal perangkat lunak untuk menjalankan aplikasi adalah :

1. Sistem Operasi Windows XP SP 2
2. Graphic Card
3. DirectX 9

3.2.3 Analisis Pengguna

Aplikasi *game* kisah budak pahatu lalis dikhususkan untuk anak-anak yang berusia 8-12 tahun, namun tidak dilarang apabila kalangan usia lain memainkan *game* ini karena *design*, *gameplay,* dan tampilan-tampilan lain didalam *game* dibuat tidak

sepenuhnya bertemakan anak-anak, hanya pesan-pesan yang ditampilkan pada layar lebih cocok apabila dibaca oleh anak-anak kalangan sekolah dasar.

Adapun criteria pengguna yang dapat menggunakan aplikasi *game* ini :

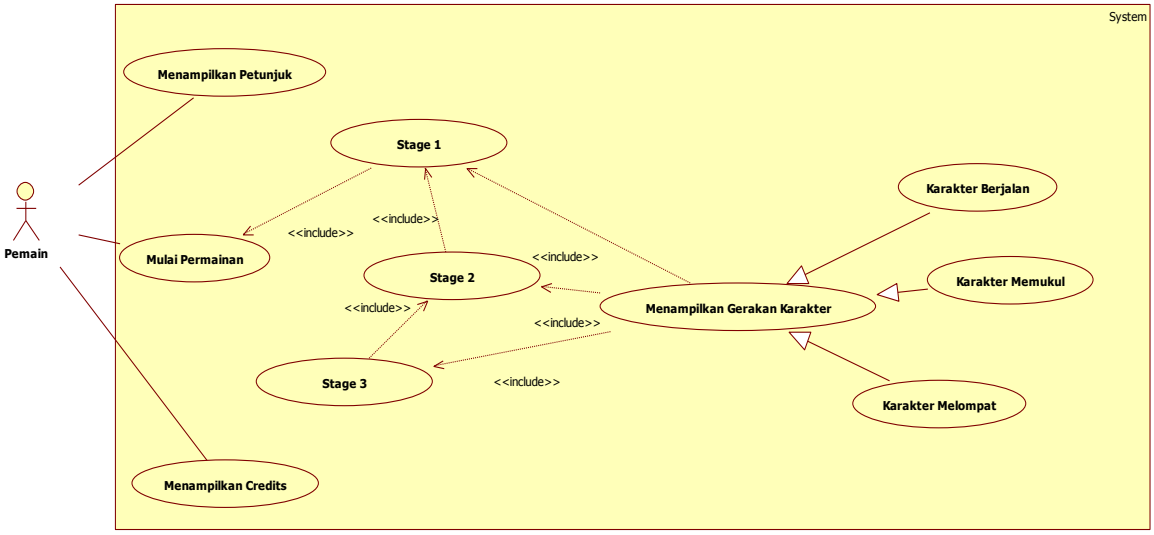
1. Mengerti menggunakan komputer secara umum
2. Mengerti pengaplikasian sistem operasi windows
3. Mengerti penggunaan mouse dan keyboard

3.3 Analisis Kebutuhan Fungsional

Analisis kebutuhan fungsional merupakan analisis terhadap perancangan *game* yang merupakan kerangka untuk selanjutnya dibangun menjadi *game* analisis 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. Disini penulis melakukan analisis dengan pendekatan berorientasi objek menggunakan pemodelan UML dengan membangun *Use Case Diagram*, *Activity Diagram*, *Class Diagram*, *Sequence Diagram*.

3.3.1 *Use Case Diagram*

Use case diagram merupakan pemodelan untuk mendeskripsikan interaksi antara satu atau lebih aktor dengan sistem yang akan dibuat. Berikut ini adalah perancangan proses-proses yang terdapat pada *game* kisah budak pahatu lalis. Terdapat pada gambar 3.14 :



Gambar 3.9 *Use Case Diagram*

3.3.2 Definisi *Actor*

Definisi *Actor* berfungsi untuk menjelaskan *actor* yang terdapat pada *use case diagram*. Definisi *actor* diterangkan pada tabel 3.7

Tabel 3.5 Definisi *Actor*

No.	*Actor*	Deskripsi
1	Pengguna	Pemain (orang yang memainkan *game*)

3.3.3 Definisi *Use Case*

Definisi *use case* berfungsi untuk menjelaskan fungsi *use case* yang terdapat pada *use case diagram*. Definisi *use case* diterangkan pada tabel 3.8

Tabel 3.6 *Definisi Use Case*

No	Use Case	Deskripsi
1	Petunjuk	Fungsional untuk menampilkan menu petunjuk
2	Mulai Permainan	Fungsional untuk memulai permainan
3	*Credits*	Fungsional untuk melihat menu *credits*
4	Memainkan *stage* 1	Proses permainan *stage* 1
5	Memainkan *stage* 2	Proses permainan *stage* 2
6	Memainkan *stage* 3	Proses permainan *stage* 3
7	Menampilkan Gerakan Karakter	Proses Generalisasi yang meliputi tiga proses pada karakter
8	Karakter Berjalan	Proses Membuat Karakter Berjalan
9	Karakter Memukul	Proses Membuat Karakter Memukul
10	Karakter Melompat	Proses Membuat Karakter Melompat

3.3.4 *Scenario Use Case*

Scenario setiap bagian pada *use case* menunjukkan proses apa yang terjadi pada setiap bagian didalam *use case* tersebut, 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*

Tabel 3.7 *Scenario Use Case* Petunjuk

Identifikasi		
Nomor	1	
Nama	Petunjuk	
Tujuan	Untuk masuk menu petunjuk	
Deskripsi	Proses untuk masuk menu petunjuk dan membaca menu petunjuk	
Actor	Pemain	
Skenario Utama		
Kondisi Awal	Pemain berada di menu utama	
Aksi *Actor*		Reaksi Sistem
1. Memilih *button* Petunjuk		
		2. Memproses inputan dan masuk ke menu petunjuk
Kondisi Akhir	Sistem menampilkan menu petunjuk	

Tabel 3.8 *Scenario Use Case* Mulai Permainan

Identifikasi	
Nomor	2
Nama	Mulai Permainan
Tujuan	Untuk masuk menu permainan dan mulai bermain
Deskripsi	Proses untuk memulai *game*

Actor	Pemain
Skenario Utama	
Kondisi Awal	Pemain berada di menu utama
Aksi *Actor*	Reaksi Sistem
1.Memilih *button* Mulai Permainan	
	2. Memproses inputan dan masuk ke *intro scene*
Kondisi Akhir	Sistem menampilkan *intro scene*

Tabel 3.9 *Scenario Use Case Credits*

Identifikasi	
Nomor	3
Nama	*Credits*
Tujuan	Menampilkan menu *credits*
Deskripsi	Proses untuk masuk menu *credits* dan membaca menu *credits*
Actor	Pemain
Skenario Utama	
Kondisi Awal	Pemain berada di menu utama
Aksi *Actor*	Reaksi Sistem
1. Memilih *button credits*	
	2. Memproses inputan dan masuk ke menu *credits*
Kondisi Akhir	Sistem menampilkan menu *credits*

Tabel 3.10 *Scenario Use Case* Memainkan *Stage* 1

Identifikasi	
Nomor	4

Nama	*Stage* 1	
Tujuan	Memainkan *game stage* 1	
Deskripsi	Proses memainkan *game stage* 1	
Actor	Pemain	
Skenario Utama		
Kondisi Awal	Pemain berada di *stage* 1	
Aksi *Actor*		Reaksi Sistem
1.Menekan tombol *next* pada *intro scene* sampai selesai		
		2. Membawa pemain ke *stage* 1
3. Menyelesaikan semua tantangan *stage* 1		
		4. Memeriksa hasil permainan dan membawa pemain ke *stage* 2 jika pemain bisa menyelesaikan tantangan
Kondisi Akhir	Sistem membawa pemain ke *stage* 2	

Tabel 3.11 *Scenario Use Case* Memainkan *Stage* 2

Identifikasi		
Nomor	5	
Nama	*Stage* 2	
Tujuan	Memainkan *game stage* 2	
Deskripsi	Proses memainkan *game stage* 2	
Actor	Pemain	
Skenario Utama		
Kondisi Awal	Pemain berada di *stage* 2	
Aksi *Actor*		Reaksi Sistem
1. Memenuhi semua tantangan dari *stage* 1		

	2. Membawa pemain ke *stage* 2
3. Menyelesaikan semua tantangan *stage* 2	
	4. Memeriksa hasil permainan dan membawa pemain ke *stage* 3 jika pemain bisa menyelesaikan tantangan
Kondisi Akhir	Sistem membawa pemain ke *stage* 3

Tabel 3.12 *Scenario Use Case* Memainkan *Stage* 3

Identifikasi	
Nomor	6
Nama	*Stage* 3
Tujuan	Memainkan *game stage* 3
Deskripsi	Proses memainkan *game stage* 3
Actor	Pemain
Skenario Utama	
Kondisi Awal	Pemain berada di *intro scene*

Aksi *Actor*	Reaksi Sistem
1. Memenuhi semua tantangan dari *stage* 2	
	2. Membawa pemain ke *stage* 3
3. Menyelesaikan semua tantangan *stage* 3	
	4. Menampilkan cerita keseluruhan jika pemain bisa menyelesaikan tantangan
Kondisi Akhir	Menampilkan cerita keseluruan

Tabel 3.13 *Scenario Use Case* Menampilkan Gerakan Karakter

Identifikasi	
Nomor	7

Nama	Menampilkan Gerakan Karakter	
Tujuan	Memainkan *game stage* 3	
Deskripsi	Proses generalisasi yang meliputi 3 gerakan yang dapat dilakukan oleh karakter	
Actor	Pemain	
Skenario Utama		
Kondisi Awal	Pemain berada di *stage* yang dimainkan	
Aksi *Actor*		Reaksi Sistem
1. Menekan tombol sesuai gerakan yang diinginkan		
		2. Menampilkan karakter beserta gerakan yang diinginkan pemain
Kondisi Akhir	Karakter bergerak sesuai yang diinginkan pemain	

Tabel 3.14 *Scenario Use Case* Karakter Berjalan

Identifikasi		
Nomor	8	
Nama	Karakter Berjalan	
Tujuan	Membuat karakter berjalan	
Deskripsi	Proses untuk membuat karakter berjalan didalam permainan	
Actor	Pemain	
Skenario Utama		
Kondisi Awal	Pemain berada di *stage* yang dimainkan	
Aksi *Actor*		Reaksi Sistem
1. Menekan A, W, S, D dan menggerakan kursor *mouse*		
		2. Menampilkan karakter berjalan
Kondisi Akhir	Karakter berjalan selama tombol ditekan	

Tabel 3.15 *Scenario Use Case* Karakter Memukul

Identifikasi		
Nomor	9	
Nama	Karakter Memukul	
Tujuan	Membuat karakter memukul	
Deskripsi	Proses untuk membuat karakter memukul didalam permainan	
Actor	Pemain	
Skenario Utama		
Kondisi Awal	Pemain berada di *stage* yang dimainkan	
Aksi *Actor*		Reaksi Sistem
1. Menekan tombol kiri *mouse* (*fire 1)*		
		2. Menampilkan karakter memukul
Kondisi Akhir	Karakter memukul selama tombol ditekan	

Tabel 3.16 *Scenario Use Case* Karakter Melompat

Identifikasi		
Nomor	10	
Nama	Karakter Melompat	
Tujuan	Membuat karakter melompat	
Deskripsi	Proses untuk membuat karakter melompat didalam permainan	
Actor	Pemain	
Skenario Utama		
Kondisi Awal	Pemain berada di *stage* yang dimainkan	
Aksi *Actor*		Reaksi Sistem

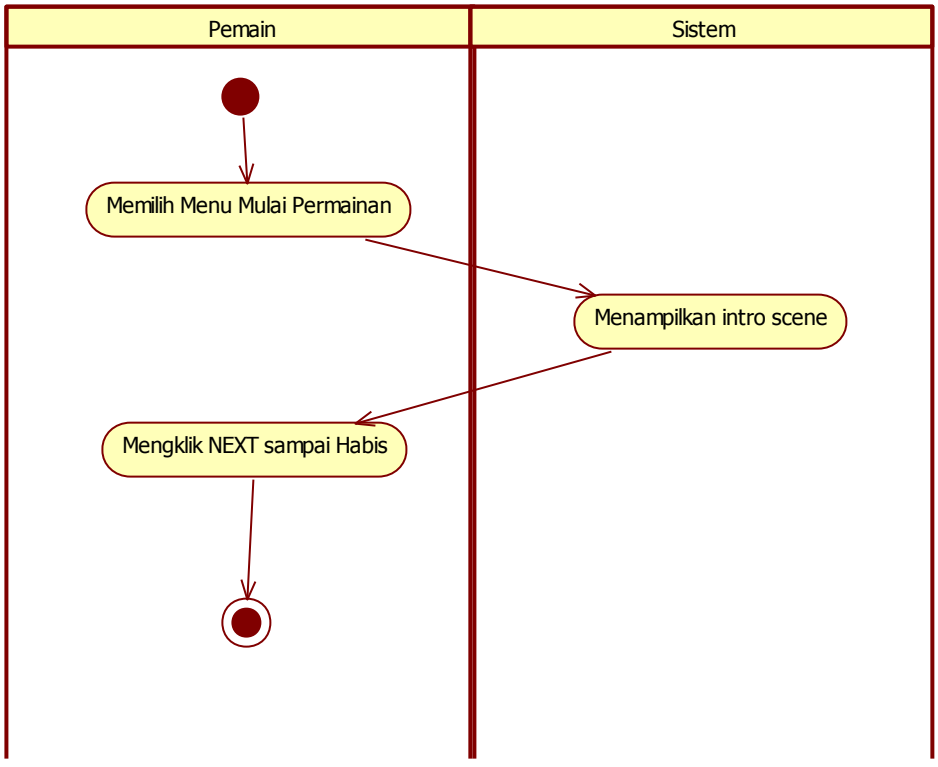
1. Menekan tombol *space* pada *keyboard*	
	2. Menampilkan karakter melompat
Kondisi Akhir	Karakter melompat selama tombol ditekan

3.3.5 *Activity Diagram*

Activity Diagram atau Diagram Aktivitas dibuat untuk menggambarkan *workflow* (aliran kerja) atau aktivitas dari sebuah sistem atau proses bisnis, atau menu yang ada pada perangkat lunak.

Diagram aktivitas yang ada pada *game* ini yaitu :

1. *Activity Diagram* Mulai Permainan

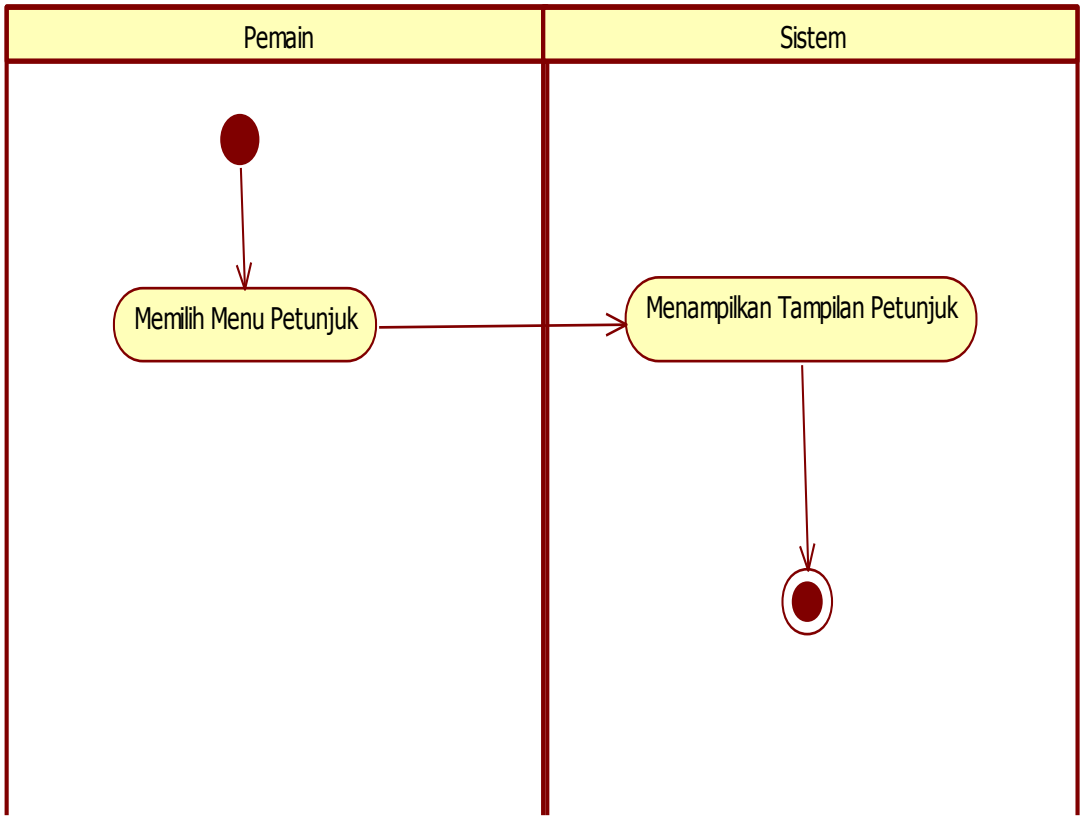


Gambar 3.10 *Activity Diagram* Mulai Permainan

Tabel 3.17 Penjelasan *Activity Diagram* Mulai Permainan

Deskripsi	Menu Mulai Permainan merupakan menu untuk masuk kedalam proses memainkan *game*
Alur Proses	1. Pemain memilih menu Mulai Permainan 2. Sistem akan menampilkan *intro scene* sebagai awal permainan 3. Pemain Mengklik *button* NEXT sampai habis

2. *Activity Diagram* Petunjuk

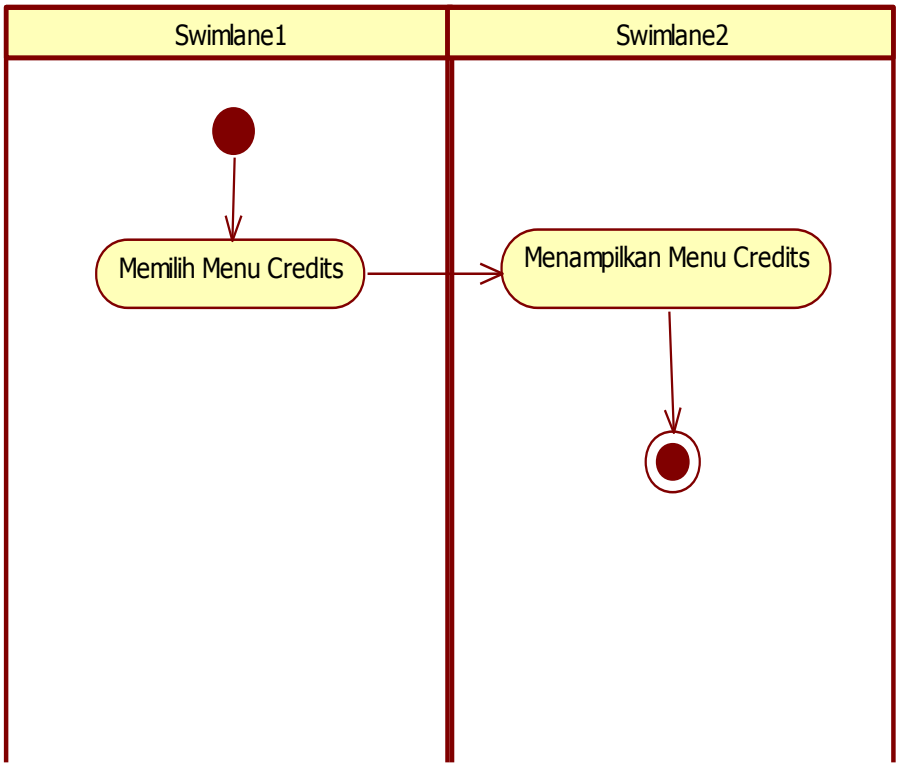


Gambar 3.11 *Activity Diagram* Petunjuk

Tabel 3.18 Penjelasan *Activity Diagram* Petunjuk

Deskripsi	Menu Petunjuk merupakan menu tampilan mengenai aturan bermain
Alur Proses	1. Pemain memilih menu Petunjuk 2. Sistem akan menampilkan tampilan menu Petunjuk

3. *Activity Diagram Credits*



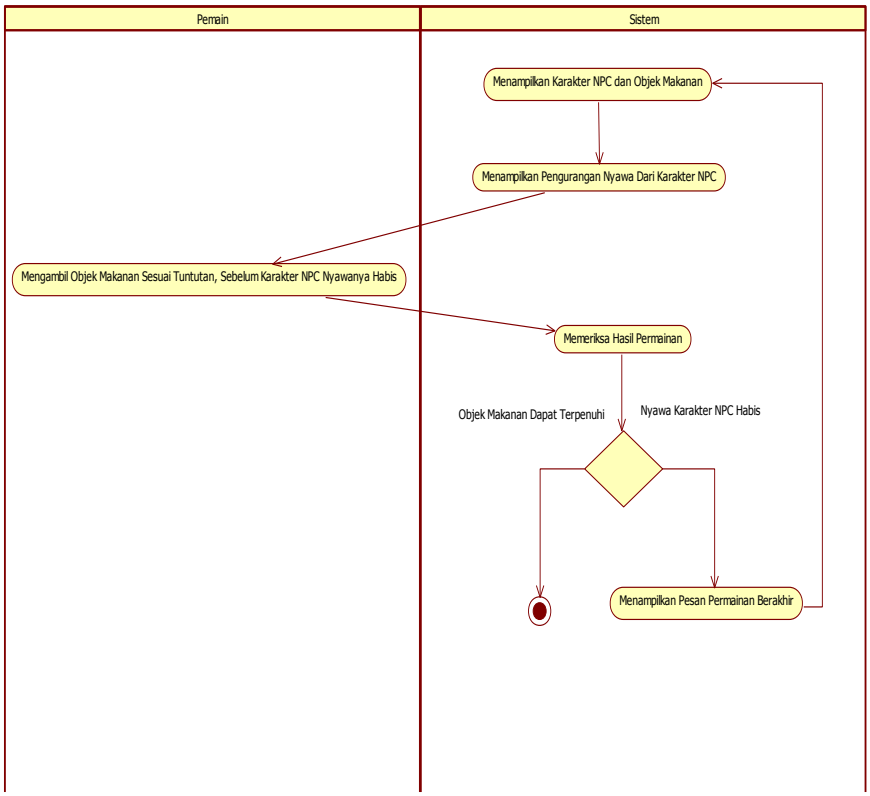
Gambar 3.12 *Activity Diagram Credits*

Tabel 3.19 Penjelasan *Activity Diagram Credits*

Deskripsi	Menu *Credits* merupakan menu tampilan mengenai pembuat *game*
Alur Proses	1. Pemain memilih menu *Credits*

	2. Sistem akan menampilkan tampilan menu *Credits*

4. *Activity Diagram Stage* 1



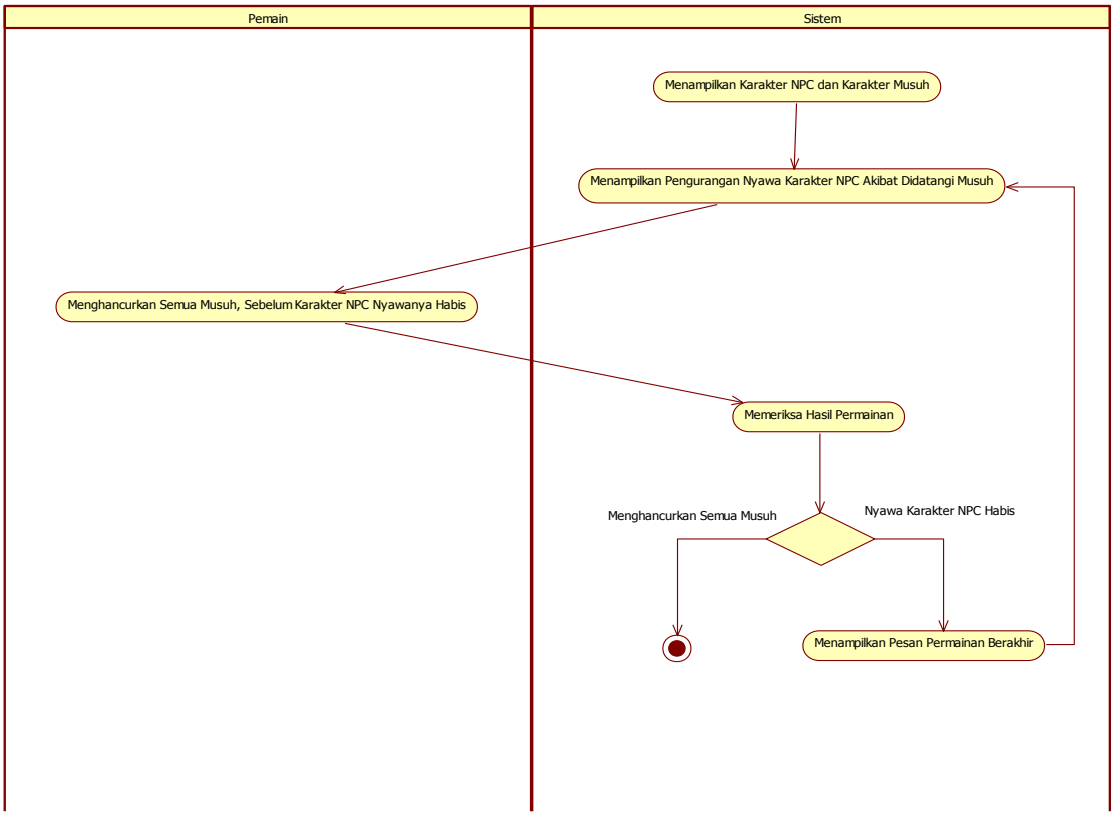
Gambar 3.13 *Activity Diagram Stage* 1

Tabel 3.20 Penjelasan *Activity Diagram* Memainkan *Stage* 1

Deskripsi	Setelah memilih menu mulai permainan maka pemain akan mulai bermain pada level 1 dengan tantangan mengumpulkan makanan sebelum nyawa sang adik didalam *game* habis
Alur Proses	1. Sistem menampilkan Karakter *NPC* dan objek makanan 2. Pemain mulai bermain dengan mencari semua bahan makanan

	3. Sistem memeriksa hasil pemain 4. Jika pemain telah mengumpulkan makanan sesuai jumlah yang diminta maka pemain dinyatakan menang jika tidak maka pemain dianggap tidak dapat menyelesaikan *stage* 1 dan kembali mengulangi dari awal

5. *Activity Diagram Stage* 2



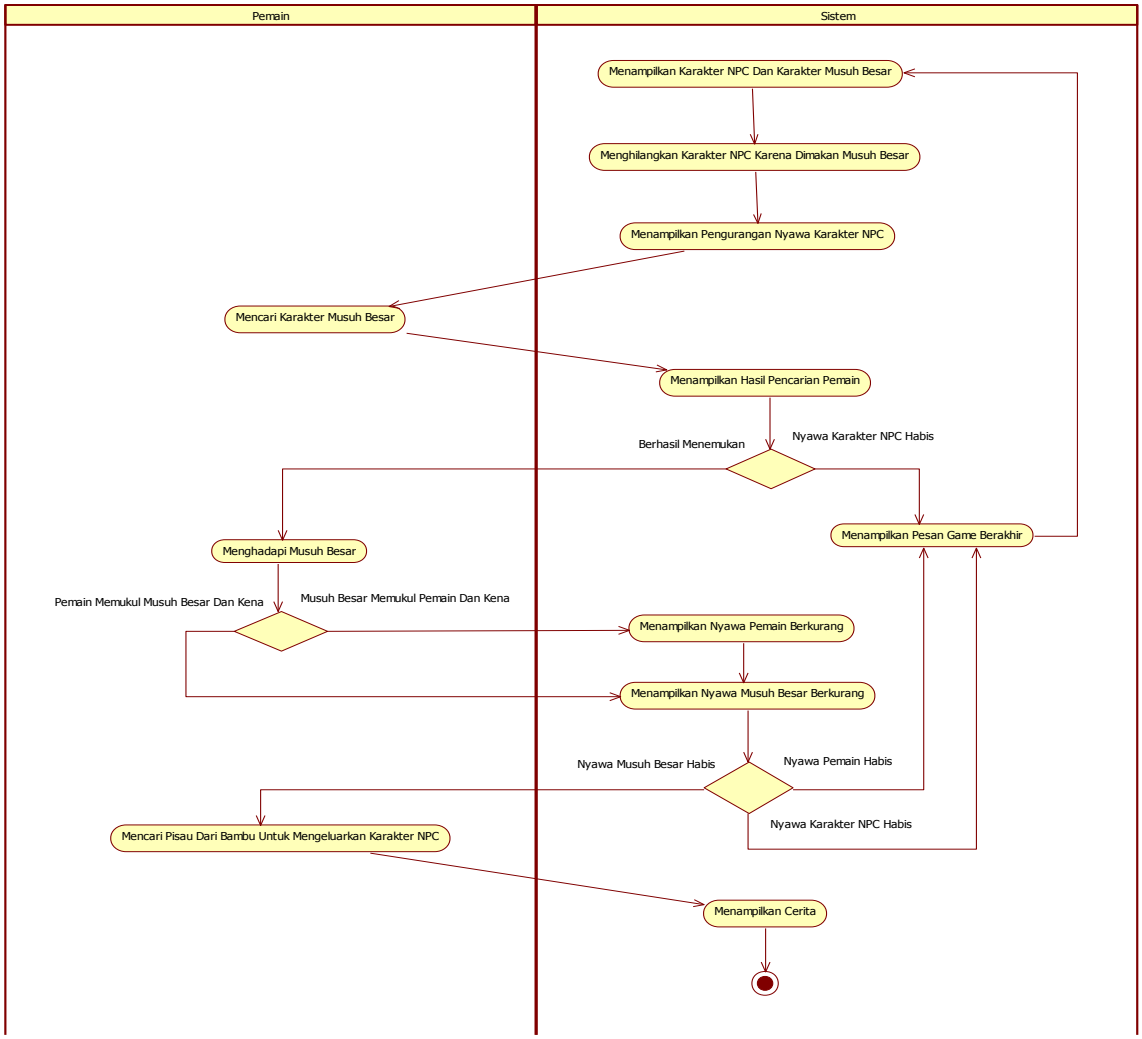
Gambar 3.14 *Activity Diagram Stage* 2

Tabel 3.21 Penjelasan *Activity Diagram* Memainkan *Stage* 2

Deskripsi	Setelah menyelesaikan *stage* 1 maka pemain akan mulai bermain pada

	stage 2 dengan tantangan melawan semua musuh yang mengurangi nyawa sang adik atau karakter *NPC*
Alur Proses	1. Sistem menampilkan Karakter *NPC* dan karakter musuh 2. Pemain mulai bermain dengan memukul semua musuh yang mengurangi nyawa sang adik 3. Sistem memeriksa hasil pemain 4. Jika pemain telah menghancurkan musuh sesuai jumlah yang diminta maka pemain dinyatakan menang jika tidak maka pemain dianggap tidak dapat menyelesaikan *stage* 2 dan kembali mengulangi dari awal

6. *Activity Diagram Stage* 3



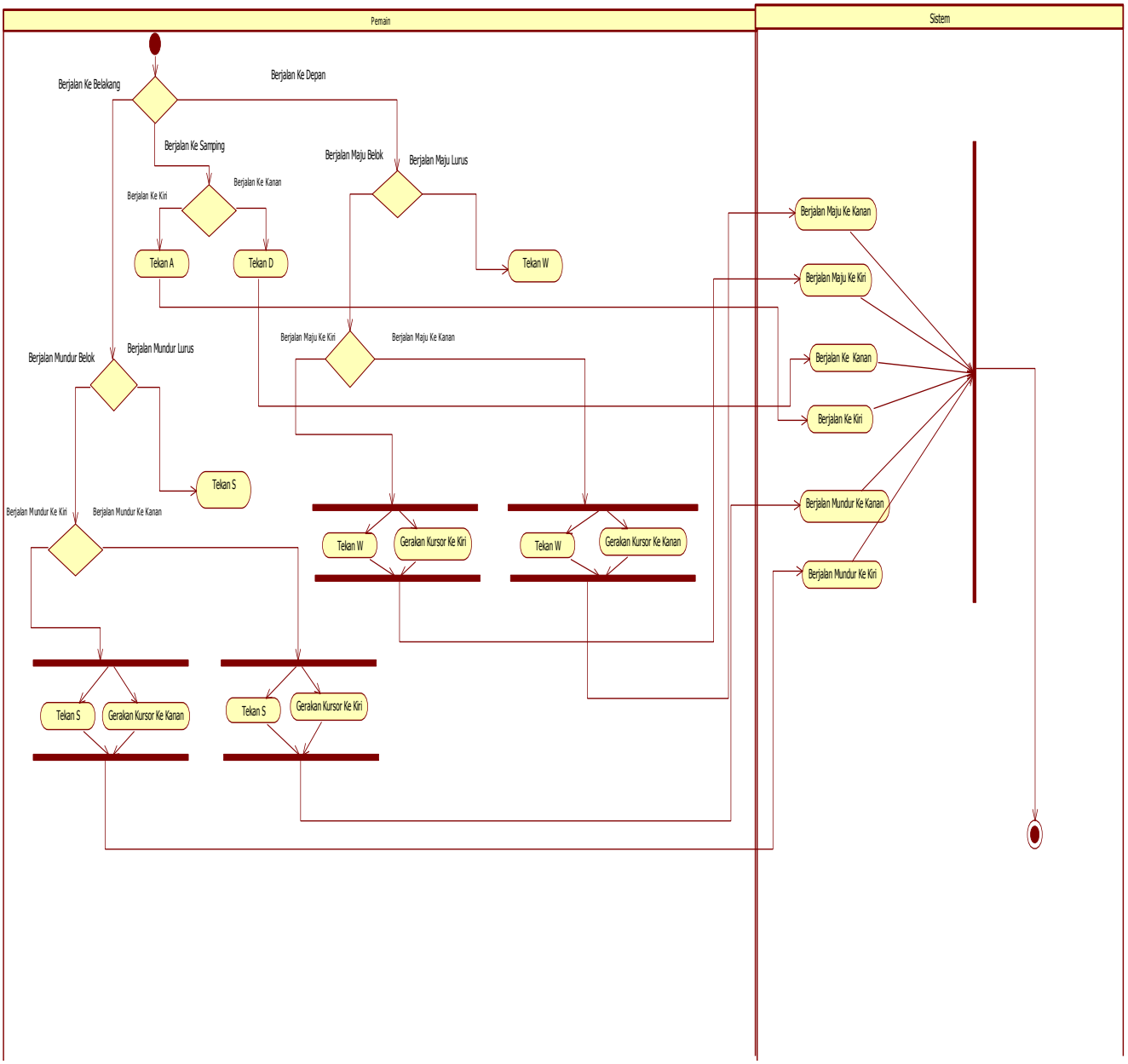
Gambar 3.15 *Activity Diagram Stage* 3

Tabel 3.22 Penjelasan *Activity Diagram* Memainkan *Stage* 3

Deskripsi	Setelah menyelesaikan *stage* 2 maka pemain akan mulai bermain pada *stage* atau *stage final* dengan tantangan melawan musuh besar berupa ular yang telah memakan karakter sang adik atau karakter *NPC*
Alur Proses	1. Sistem menampilkan Karakter *NPC* dan karakter musuh besar 2. Sistem menghilangan karakter *NPC* karena telah dimakan oleh musuh besar mengurangi nyawa sang adik

	3. Sistem menampilkan pengurangan nyawa sang adik 4. Pemain diharuskan mencari ular sebelum karakter *NPC* nyawanya habis 5. Setelah menemukan karakter ular atau musuh besar pemain harus melawannya dengan cara memukul-mukulnya sambil ular pun melawan 6. Jika serangan ular mengenai pemain maka nyawa pemain berkurang, jika pukulan pemain mengenai ular maka nyawa ular berkurang 7. Jika pemain berhasil menang (menghabiskan nyawa karakter ular) maka pemain harus mencari pisau dari bambu untuk mengeluarkan adiknya dari perut ular, jika kalah maka pemain akan kembali ke *stage* 3 kembali 8. Setelah berhasil mengeluarkan sang adik maka *game* dinyatakan selesai dan akan keluar cerita asli dari kisah budak pahatu lalis

7. *Activity Diagram* Berjalan

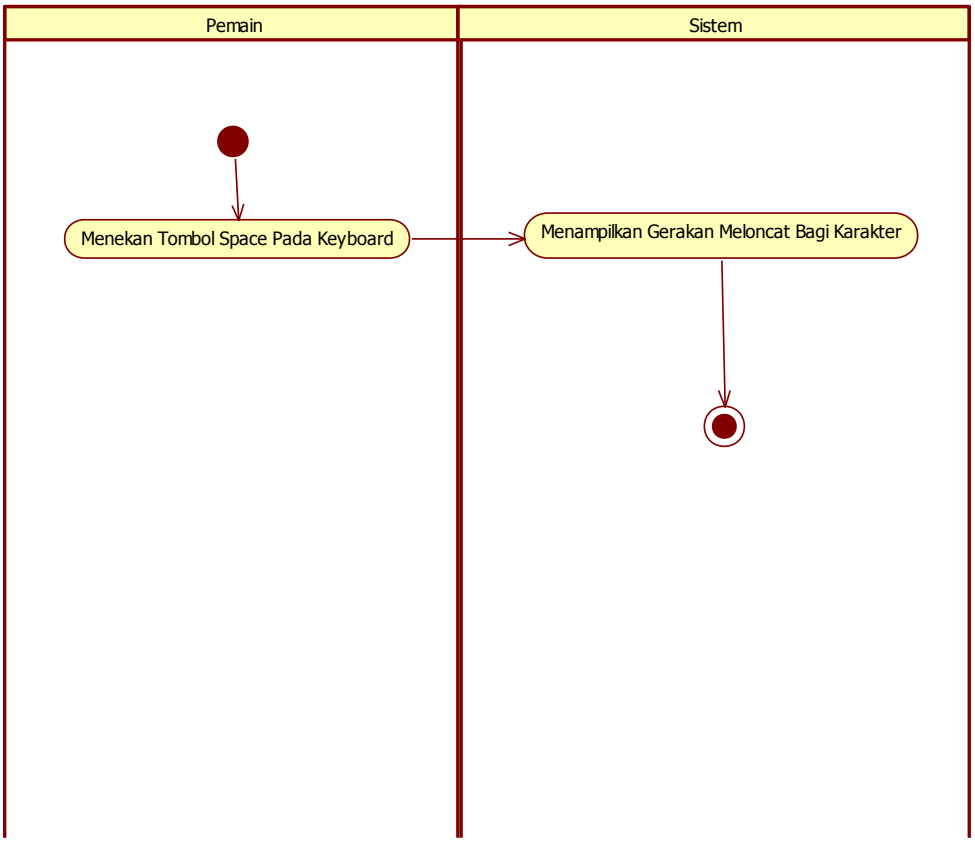


Gambar 3.16 *Activity Diagram* Berjalan

Tabel 3.23 Penjelasan *Activity Diagram* Berjalan

Deskripsi	Dalam memainkan *game* karakter yang dikendalikan oleh pemain dapat melalakukan beberapa gerakan salah satunya berjalan dan ini lah diagram aktivitas dari satu kemampuan berjalan
Alur Proses	1. Pemain memilih untuk berjalan kemana 2. Untuk berjalan lurus ke depan pemain cukup menekan tombol ‘W’ pada *keyboard* 3. Untuk berjalan ke depan namun belok kanan pemain dapat menekan tombol ‘W’ sambil menggerakan kursor ke kanan 4. Untuk berjalan ke depan namun belok kiri pemain dapat menekan tombol ‘W’ sambil menggerakan kursor ke kiri 5. Untuk berjalan ke kanan pemain cukup menekan tombol ‘D’ pada *keyboard* 6. Untuk berjalan ke kiri pemain cukup menekan tombol ‘A’ pada *keyboard* 7. Untuk berjalan ke belakang namun belok kanan pemain dapat menekan tombol ‘S’ sambil menggerakan kursor ke kiri 8. Untuk berjalan ke depan namun belok kiri pemain dapat menekan tombol ‘W’ sambil menggerakan kursor ke kanan

8. *Activity Diagram* Meloncat

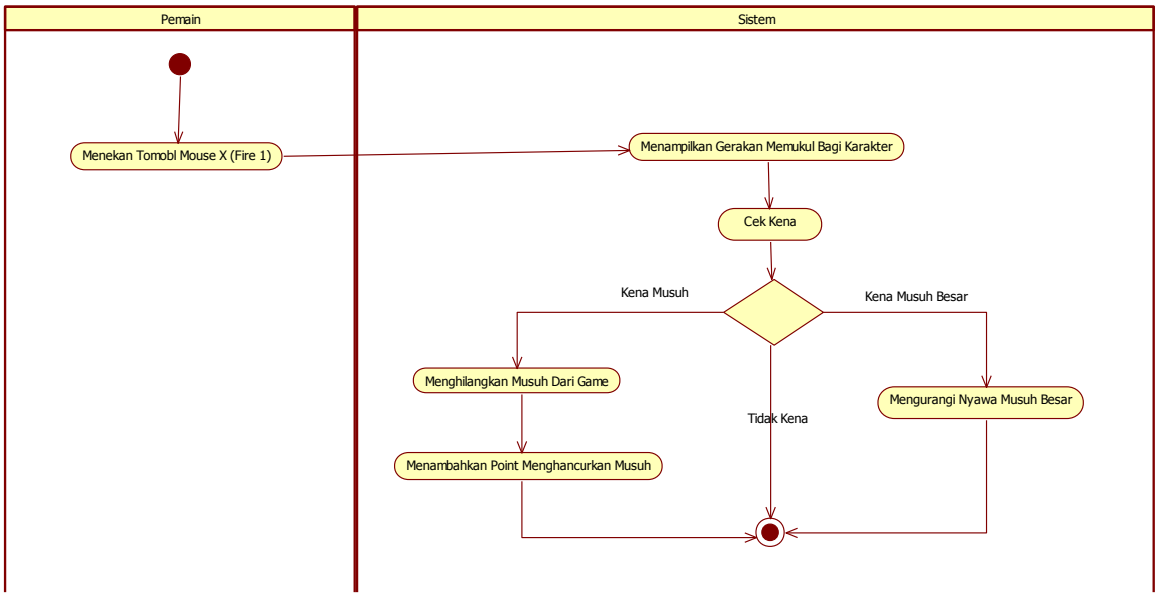


Gambar 3.17 *Activity Diagram* Meloncat

Tabel 3.24 Penjelasan *Activity Diagram* Melocat

Deskripsi	Dalam memainkan *game*, karakter yang dikendalikan oleh pemain dapat melalakukan beberapa gerakan salah satunya Meloncat dan ini lah diagram aktivitas dari satu kemampuan Meloncat
Alur Proses	1. Pemain memilih menekan tombol *space* pada *keyboard* 2. Sistem akan menampilkan gerakan meloncat bagi karakter

9. *Activity Diagram* Memukul



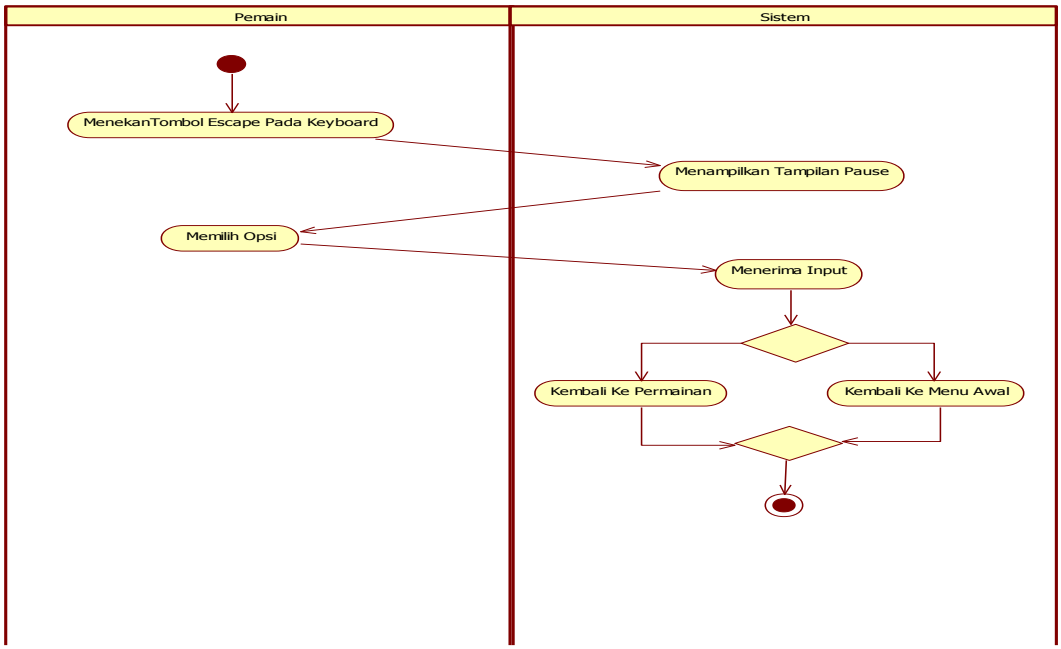
Gambar 3.18 *Activity Diagram* Memukul

Tabel 3.25 Penjelasan *Activity Diagram* Memukul

Deskripsi	Dalam memainkan *game*, karakter dapat melakukan kemampuan memukul untuk memenuhi tantangan pada *stage* 2 dan *stage* 3 dan inilah diagram aktivitas dari kemapuan memukul
Alur Proses	1. Pemain memilih menekan tombol mouse x atau tombol sebelah kiri pada *mouse* (*fire* 1) 2. Sistem akan menampilkan gerakan memukul bagi karakter 3. Kemudian sistem akan mengecek apakah pukulan karakter mengenai musuh, musuh besar atau tidak mengenai apa-apa 4. Jika mengenai musuh maka musuh akan menghilang dan 5. Sistem akan menambahkan poin menghancurkan musuh 6. Jika mengenai musuh besar yaitu ular yang memakan sang adik pada *stage* 3, maka sistem akan mengurangi nyawa musuh besar 7. Jika pukulan mengenai objek lain selain dua objek tersebut yaitu

	musuh dan musuh besar maka tidak terjadi perubahan apa-apa pada sistem

10. *Activity Diagram Pause*



Gambar 3.19 *Activity Diagram Pause*

Tabel 3.26 Penjelasan *Activity Diagram Pause*

Deskripsi	*Pause* merupakan menu apabila pemain ingin sejenak beristirahat atau sejenak lepas dari *game* tanpa keluar dari ruan *game*, selain itu dalam menu *pause* pun terdapat menu yang dapat membuat pemain kembali ke menu utama apabila dia ingin menyudahi permainan
Alur Proses	1. Pemain memilih menekan tombol *escape* pada *keyboard* 2. Sistem akan menampilkan menu *pause* 3. Ketika ditampilkan menu *pause*, *game* akan berhenti dan tidak ada aktivitas di dalam *game*

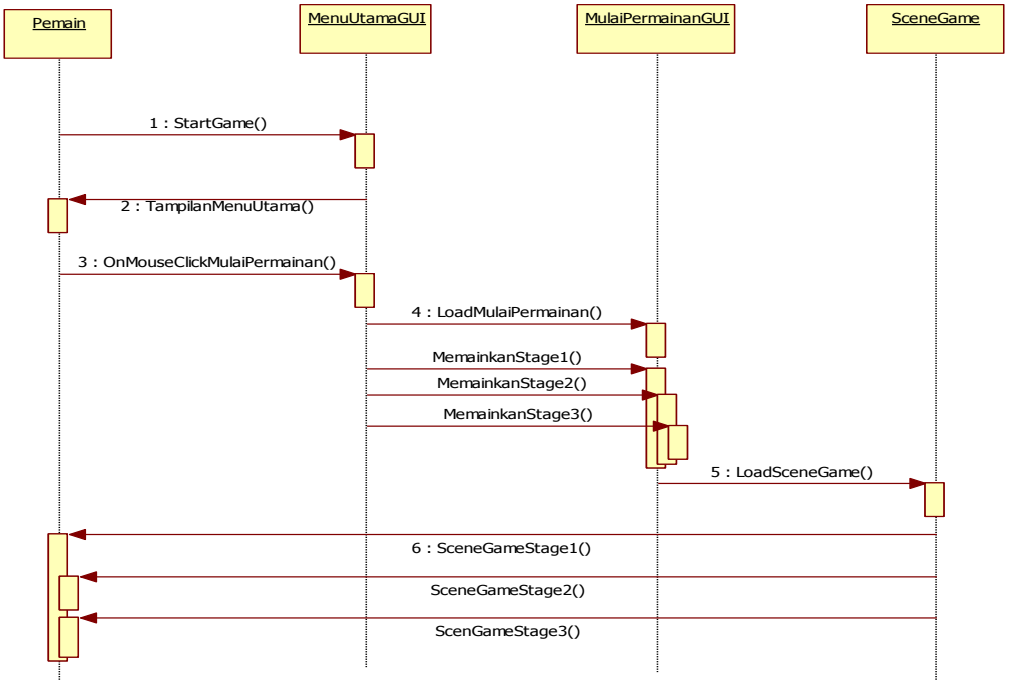
	4. Jika pemain ingin melanjutkan kembali pilih menu kembali ke *game* 5. Jika pemain ingin keluar dan kembali ke manu utama, maka pilih kembali ke menu utama

3.3.6 *Sequence 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* kisah budak pahatu lalis

1. *Sequence Diagram* Mulai Permainan

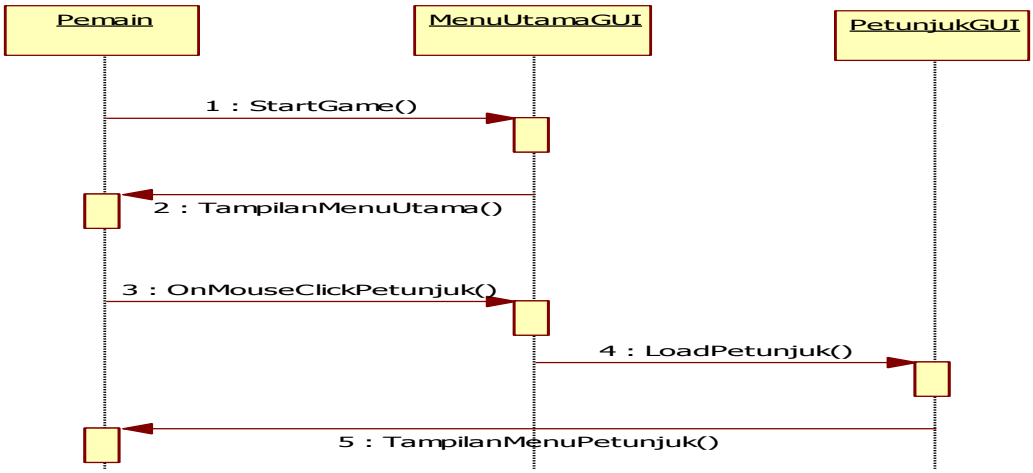
   Sequence Diagram Mulai Permainan merupakan gambaran objek untuk memulai permainan. *Sequence Diagram* Mulai Permainan terdapat pada gambar 3.25



Gambar 3.20 *Sequence Diagram* Mulai Permainan

2. *Sequence Diagram* Petunjuk

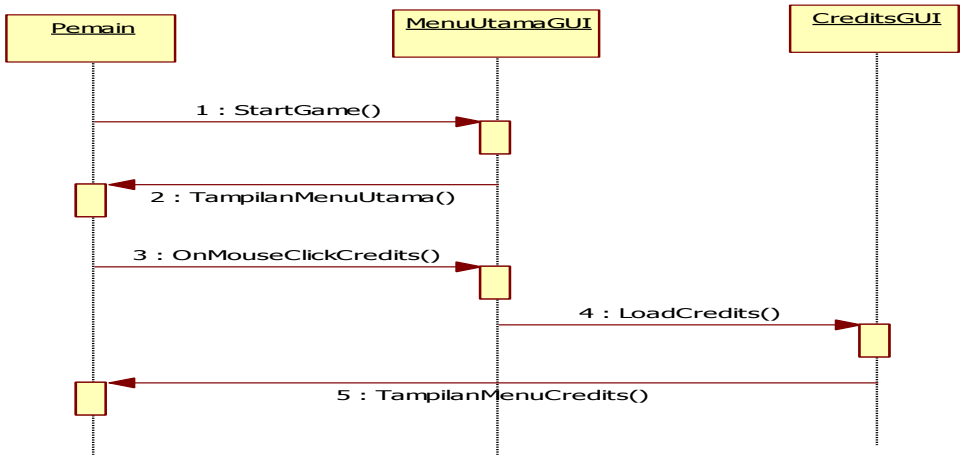
Sequence Diagram Petunjuk merupakan gambaran objek untuk menuju menu petunjuk. *Sequence Diagram* Petunjuk terdapat pada gambar 3.26



Gambar 3.21 *Sequence Diagram* Petunjuk

3. *Sequence Diagram Credits*

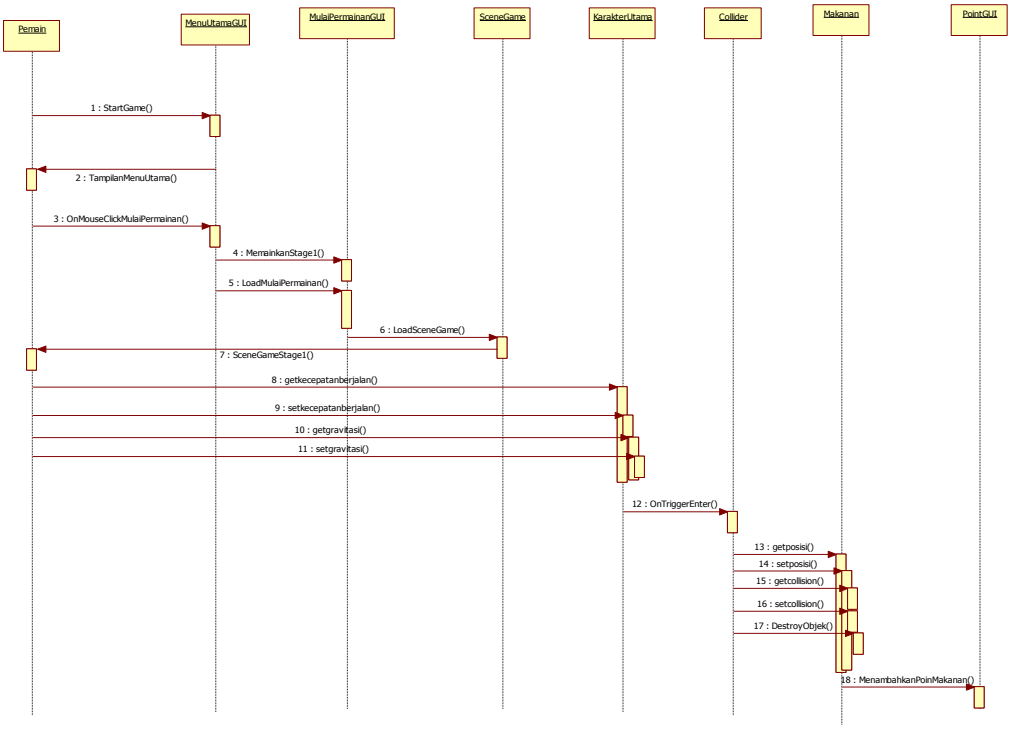
Sequence Diagram Petunjuk merupakan gambaran objek untuk menuju menu *credits*. *Sequence Diagram Credits* terdapat pada gambar 3.27



Gambar 3.22 *Sequence Diagram Credits*

4. *Sequence Diagram Stage* 1

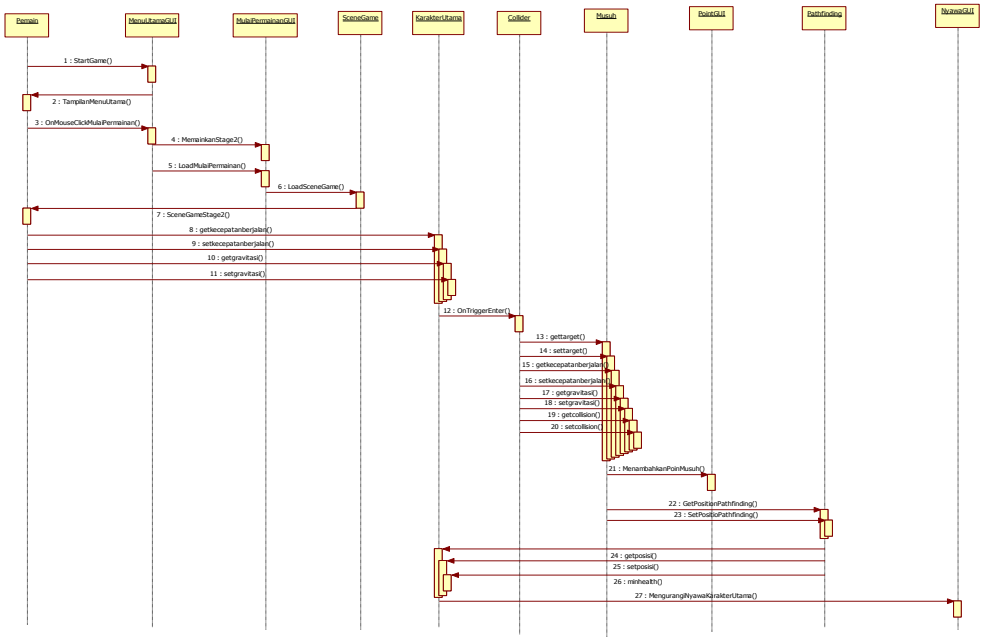
Sequence Diagram Stage 1 merupakan gambaran objek dalam permainan *stage* 1. *Sequence Diagram Stage* 1 terdapat pada gambar 3.28



Gambar 3.23 *Sequence Diagram Stage* 1

5. *Sequence Diagram Stage* 2

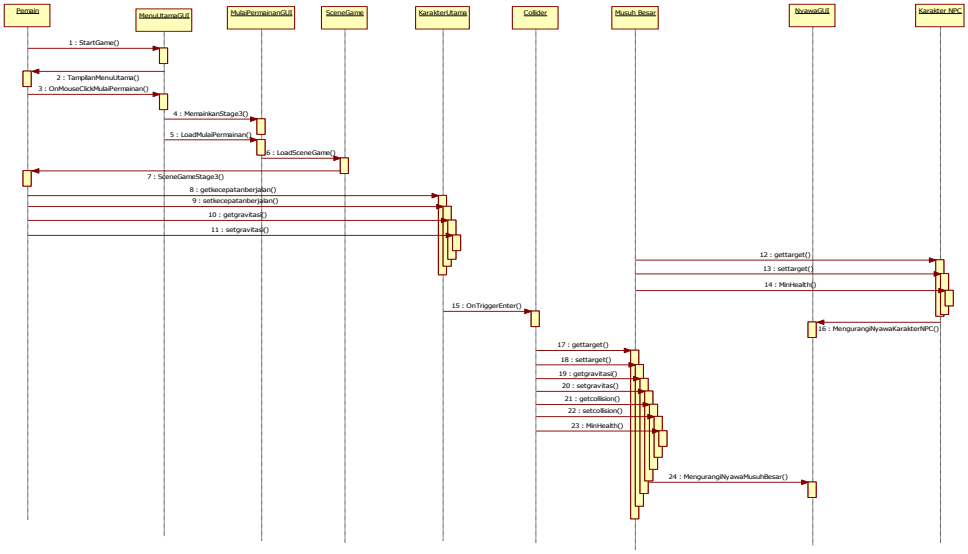
Sequence Diagram Stage 2 merupakan gambaran objek dalam permainan *stage* 2. *Sequence Diagram Stage* 2 terdapat pada gambar 3.29



Gambar 3.24 *Sequence Diagram Stage* 2

6. *Sequence Diagram Stage* 3

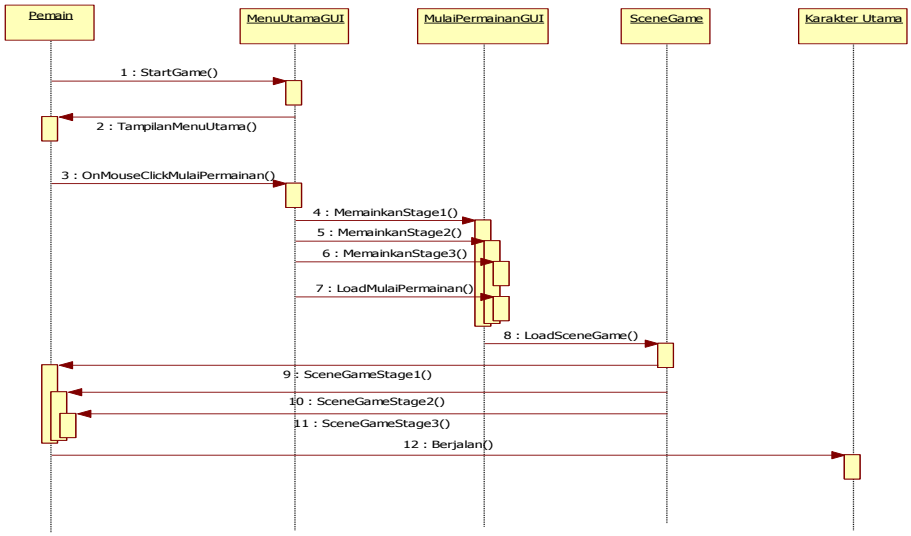
Sequence Diagram Stage 3 merupakan gambaran objek dalam permainan *stage* 3. *Sequence Diagram Stage* 3 terdapat pada gambar 3.30



Gambar 3.25 *Sequence Diagram Stage* 3

7. *Sequence Diagram* Berjalan

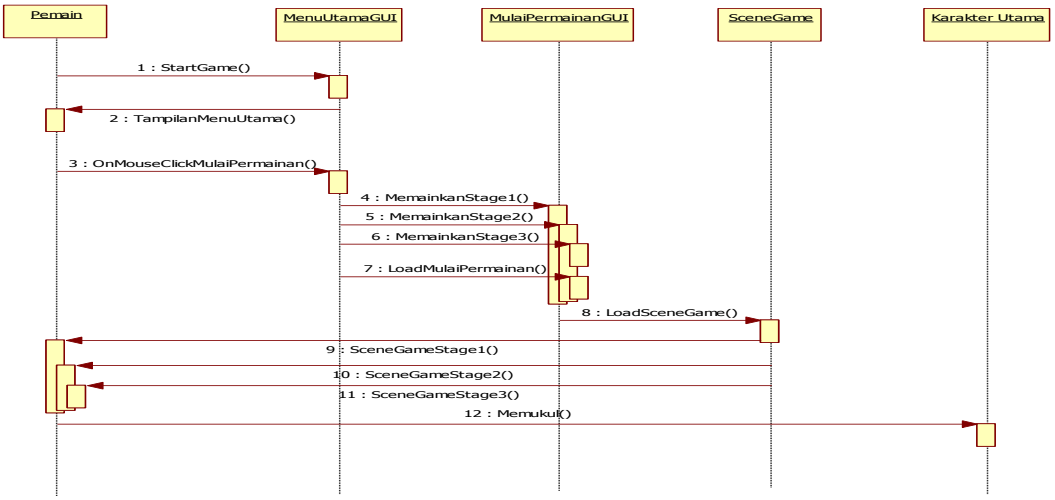
Sequence Diagram Berjalan merupakan gambaran objek untuk menampilkan gerakan karakter berjalan. *Sequence Diagram* Berjalan terdapat pada gambar 3.31



Gambar 3.26 *Sequence Diagram* Berjalan

8. *Sequence Diagram* Memukul

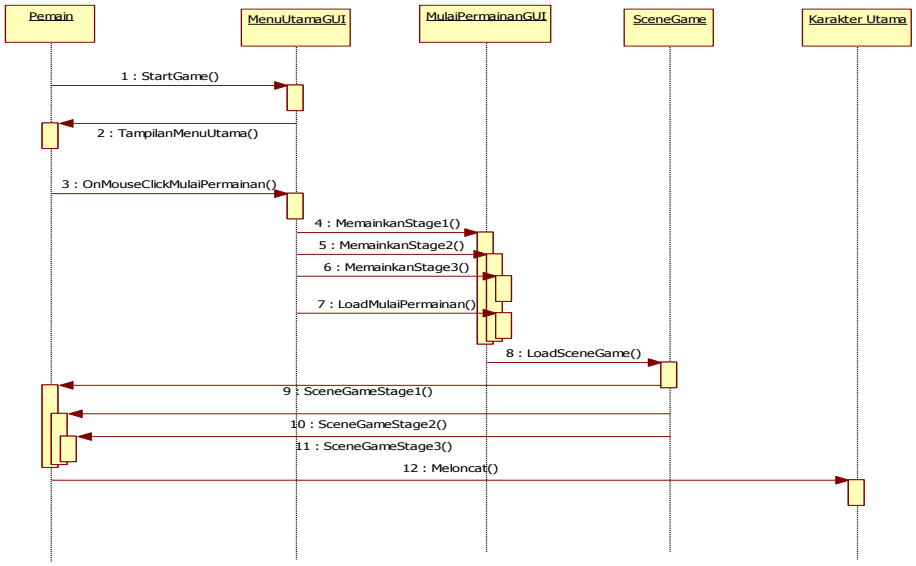
Sequence Diagram Memukul merupakan gambaran objek untuk menampilkan gerakan karakter Memukul. *Sequence Diagram* Memukul terdapat pada gambar 3.32



Gambar 3.27 *Sequence Diagram* Memukul

9. *Sequence Diagram* Meloncat

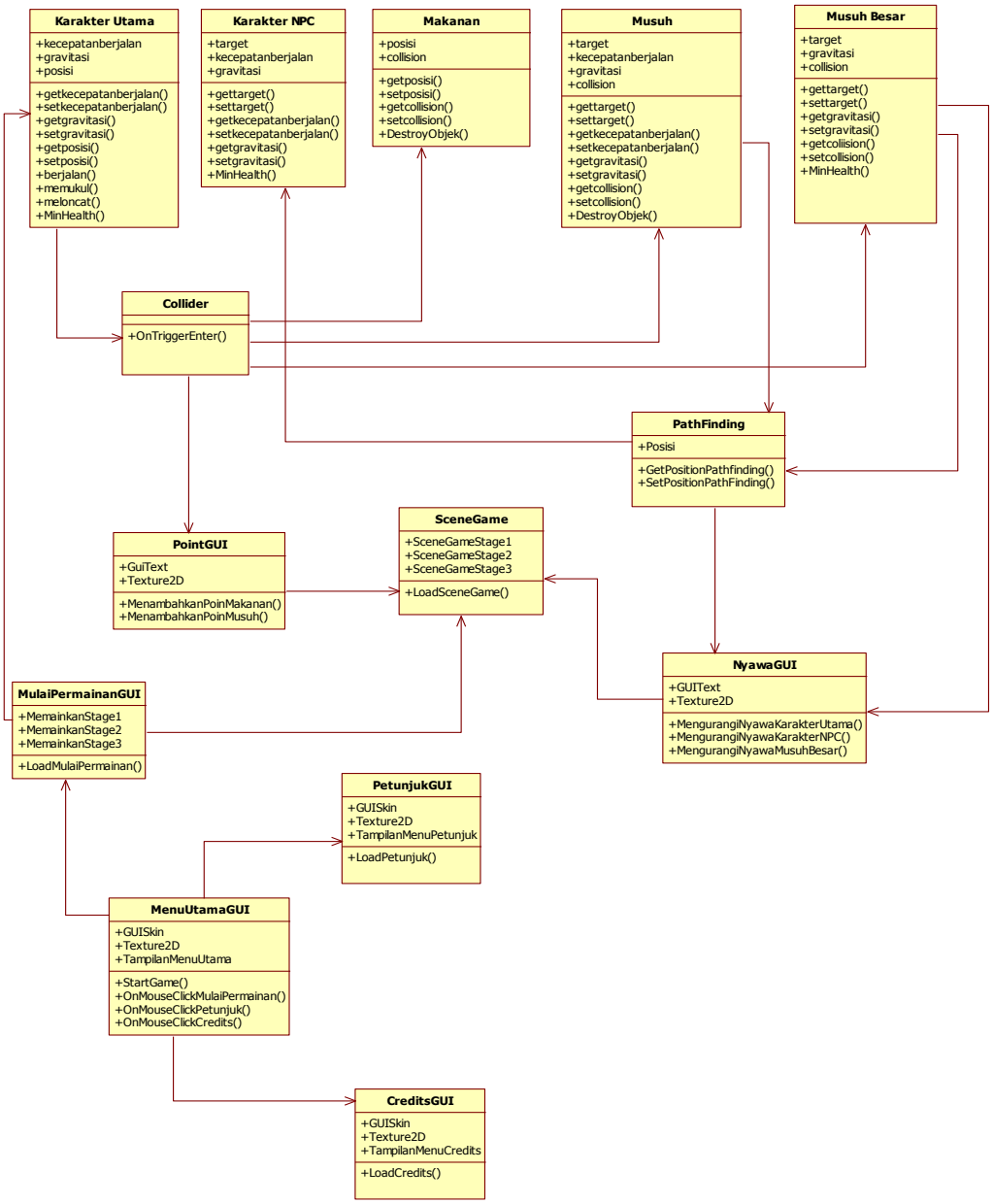
Sequence Diagram Memukul merupakan gambaran objek untuk menampilkan gerakan karakter Meloncat. *Sequence Diagram* Memukul terdapat pada gambar 3.33



Gambar 3.28 *Sequence Diagram* Meloncat

3.3.7 *Class Diagram*

Class diagram menggambarkan struktur sistem dari segi pendefinisian kelas-kelas yang dibuat untuk membangun sistem. Berikut adalah *class diagram* dari *game* kisah budak pahatu lalis yang terdapat pada gambar 3.33



Gambar 3.29 *Class Diagram*

3.4 Perancangan Sistem

Perancangan adalah bagian dari metodologi pengembangan suatu perangkat lunak yang dilakukan memberi gambaran secara terperinci. Dalam tahap ini digambarkan rancangan sistem berupa perancangan komponen permainan

3.4.1 Perancangan Komponen Permainan

Perancangan Komponen Permainan merupakan penjabaran tentang semua komponen yang dipakai didalam *gam*. Komponen atau objek yang terdapat pada game ini antara lain terdiri dari:

- Karakter Utama
- Karakter *NPC* (*Non Playable Charracter)*
- Musuh
- Musuh Besar
- Makanan
- Pohon, Rumah, dll

1. Karakter Utama

Karakter utama merupakan karakter yang dapat dikendalikan oleh pemain. Karakter utama dalam *game* merupakan karakter sang kakak dalam cerita, maka didalam *game* pun karakter utama diberi nama Sang Kakak. Sang kakak lah yang akan menentukan berhasil atau gagalnya misi yang ada pada setiap *stage* didalam *game* karena pemain hanya akan memainkan satu karakter. Gambar karakter utama terdapat pada gambar 3.34



Gambar 3.30 Karakter Utama

2. Karakter *NPC*

Karakter *NPC* (*Non Playable Charracter*) merupakan karakter yang menjadi tujuan misi pada setiap *stage*. Karakter *NPC (Non Playable Charracter)* berarti

karakter yang tidak dimainkan, karakter *NPC* disini adalah karakter sang adik didalam cerita yang merupakan tokoh yang harus dilindungi inipun yang diadaptasi kedalam *game*, dimana karakter *NPC* lah yang menjadi penentu berhasi atau gagalnya sebuah *stage* jika nyawa karakter *NPC* habis sebelum tujuan terpenuhi maka permainan berakhir dan pemain dinyatakan tidak dapat menyelesaikan permainan. Gambar karakter NPC terdapat pada gambar 3.35

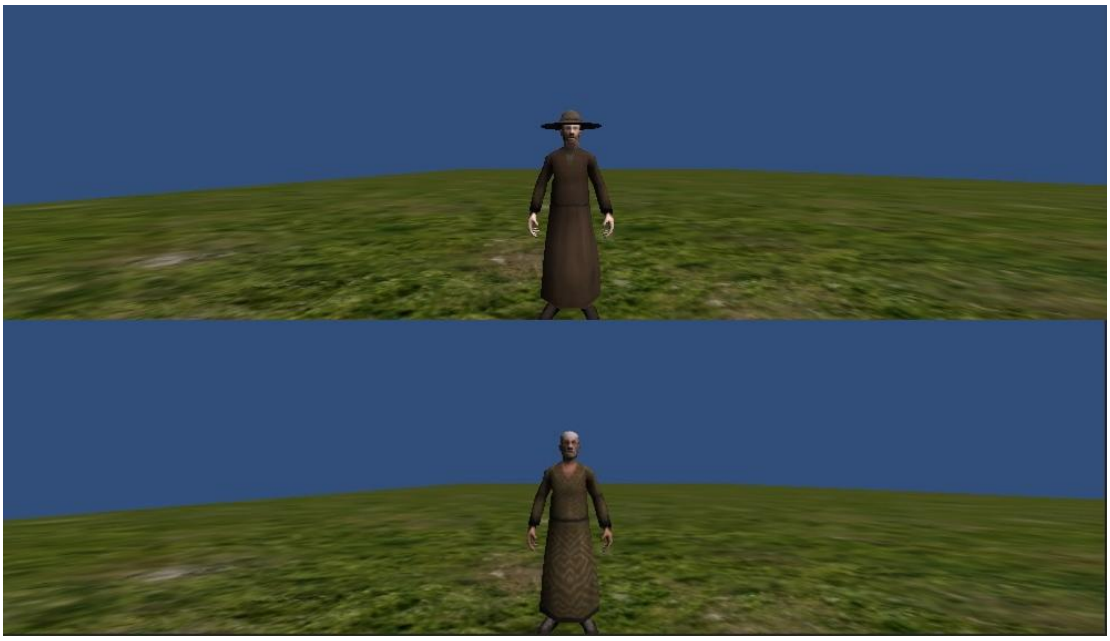


Gambar 3.31 Karakter *NPC*

3. Musuh

Musuh merupakan komponen atau objek yang dibutuhkan untuk membuat *game* menjadi lebih menantang. Seperti yang ada pada cerita dimana sang kakak selalu melindungi adiknya dari berbagai hal jahat maka dalam *game* hal jahat tersebut digambarkan oleh objek musuh. Musuh diberikan pada *stage* 2 untuk datang dan mengurangangi nyawa sang adik, apabila karakter utama tidak melawan maka lama

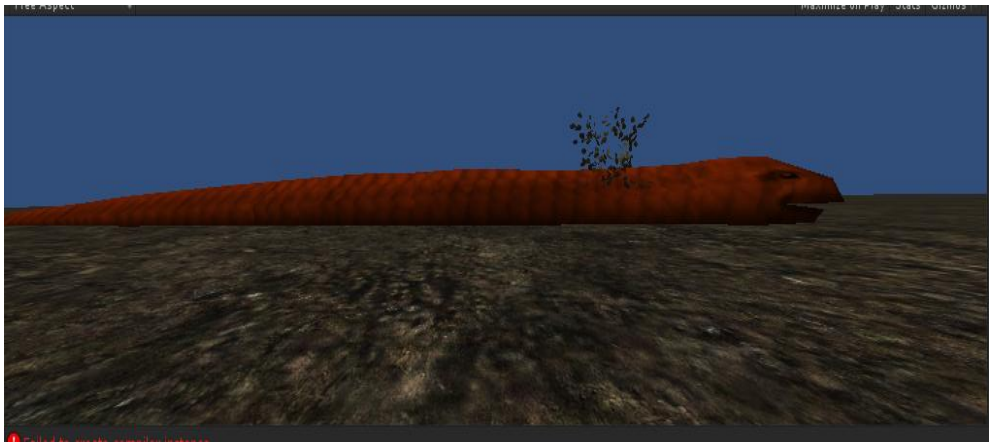
kelamaan karakter sang adik akan berkurang nyawanya dan *game* selesai. Gambar karakter musuh terdapat pada gambar 3.36



Gambar 3.32 Karakter Musuh

4. Musuh Besar

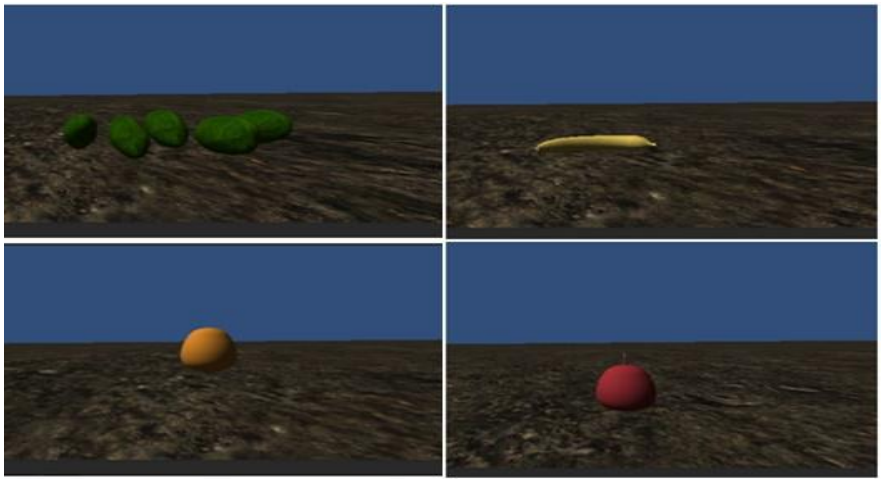
Musuh besar merupakan musuh yang terakhir yang harus dihadapi oleh karakter utama pada *game* musuh besar akan dijumpai pada *stage* ke-3 dimana musuh besar akan datang untuk memakan karakter sang adik. Seperti pada cerita dimana sang adik akan dimakan oleh ular besar dan sang kakak harus mencari ular tersebut kemudian melawannya. Musuh besar disini pun berbentuk ular yang harus dikalahkan oleh karakter utama. Gambar karakter musuh besar terdapat pada gambar 3.37



Gambar 3.33 Karakter Musuh Besar (Ular)

5. Makanan

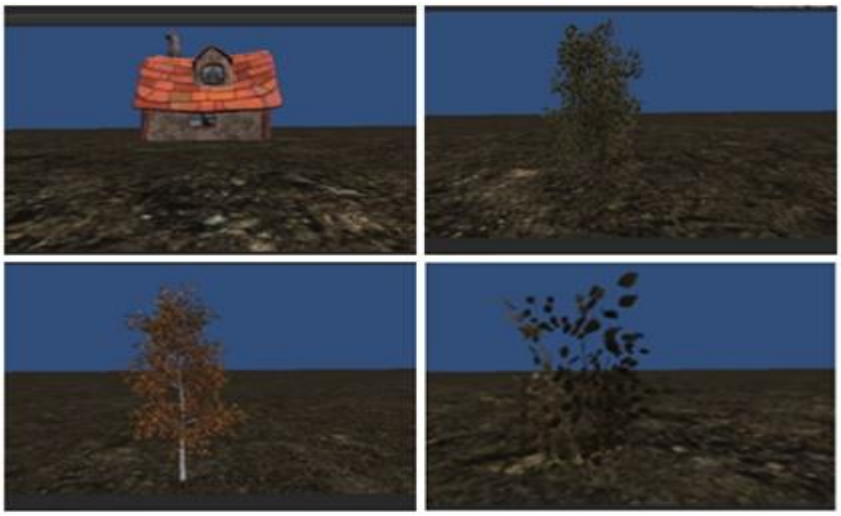
Makanan merupakan komponen atau objek yang dibutuhkan sebagai tantangan pada *stage* 1 dimana apabila karakter utama tidak dapat mencari makanan sebanyak yang ditentukan sebelum nyawa sang adik habis maka *game* selesai. Seperti pada cerita dimana sang kakak sangat bertanggungjawab terhadap asiknya dan selalu memenuhi kebutuhan adiknya, maka disini dibuat objek makanan untuk dicari oleh karakter utama. Gambar objek makanan terdapat pada gambar 3.38



Gambar 3.34 Objek Makanan

6. Objek Pelengkap

Komponen atau objek rumah, pohon, dll, dibuat untuk untuk menggambarkan latar agar nampak seperti jaman dahulu kala, pemilihan teksture terrain serta gambar yang nantinya menjadi objek pun dipilih semirip mungkin dengan keadaan jaman dahulu kala. Gambar karakter utama terdapat pada gambar 3.39

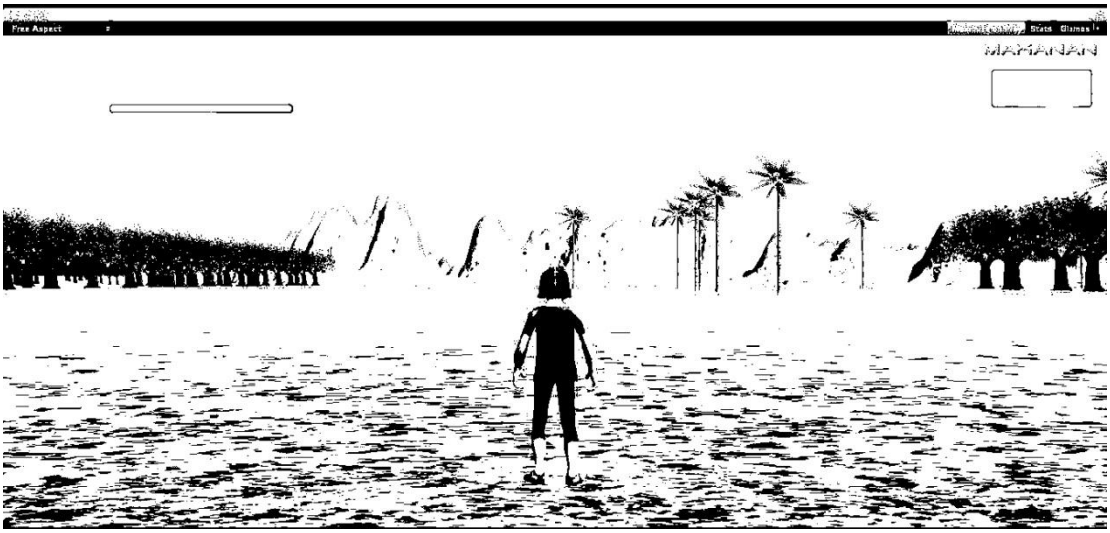


Gambar 3.35 Objek Pelengkap

3.4.2 Perancangan *Storyboard*

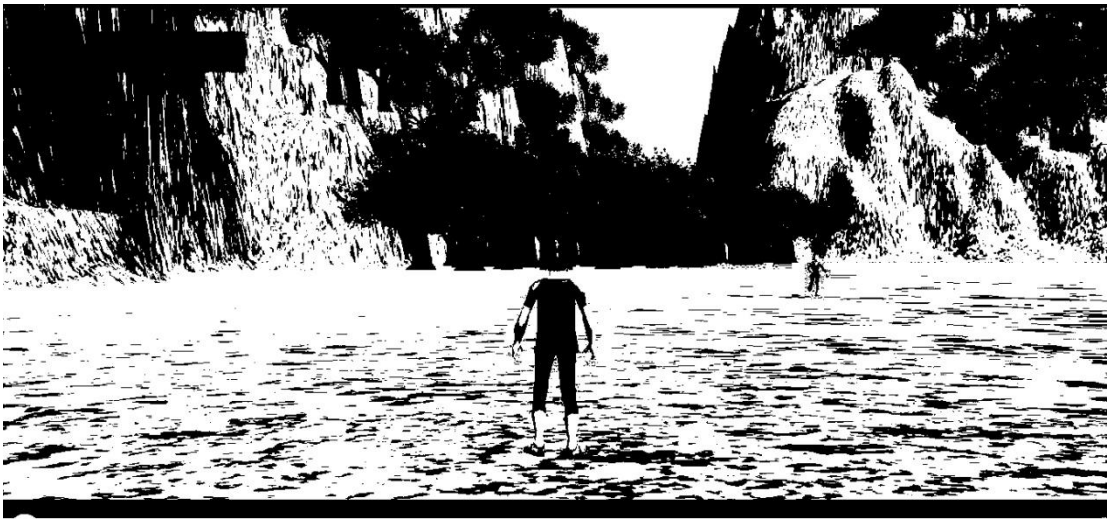
Storyboard adalah sketsa gambar yang disusun berurutan sesuai dengan naskah, dengan storyboard kita dapat menyampaikan ide cerita kita kepada orang lain dengan lebih mudah, karena kita dapat menggiring khayalan seseorang mengikuti gambar-gambar yang tersaji, sehingga menghasilkan persepsi yang sama pada ide cerita kita.

1. Perancangan *Storyboard stage* 1 terdapat pada gambar 3.40



Gambar 3.36 *Storyboard Stage* 1

2. Perancangan *Storyboard stage* 2 terdapat pada gambar 3.41



Gambar 3.37 *Storyboard Stage* 2

3. Perancangan *Storyboard stage* 3 terdapat pada gambar 3.42



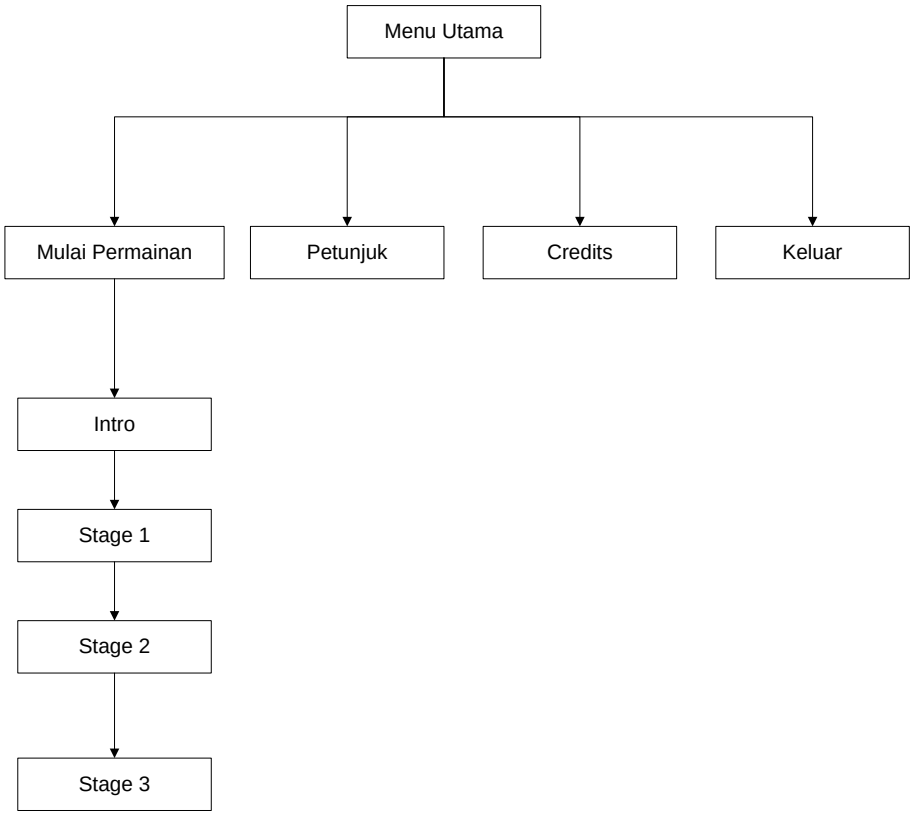
Gambar 3.48 *Storyboard Stage* 3

3.5 Perancangan Arsitetur

Perancangan arsitektur merupakan perancangan yang dilakukan untuk mengetahui kerangka dari aktifitas *game* yang ada. Perancangan Arsitektur meliputi perancangan struktur menu, perancangan interface, jaringan semantik, perancangan *method*

3.5.1 Perancangan Struktur Menu

Perancangan struktur menu dibuat untuk mengetahui fungsi-fungsi yang ada pada aplikasi. Berikut struktur menu yang ada pada *game* budak pahatu lalis terdapat pada gambar 3.43



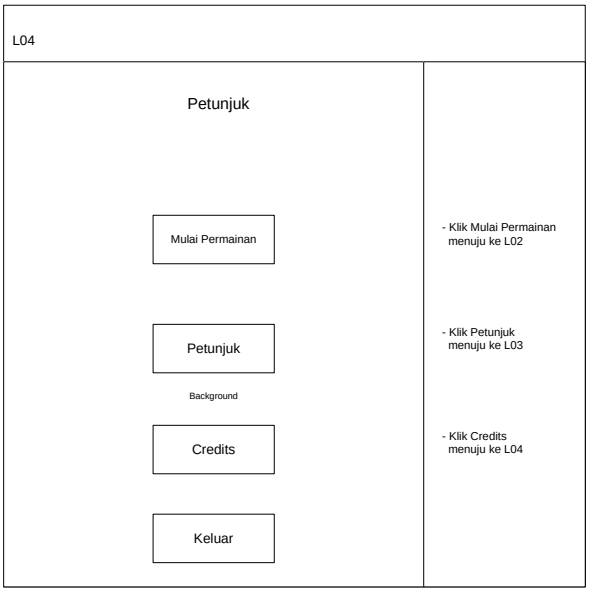
Gambar 3.39 Struktur Menu

3.5.2 Perancangan Antarmuka

Perancangan antarmuka dibuat untuk mengetahui menu-menu yang ada pada aplikasi secara lebih detail dan terperinci. Berikut antarmuka yang ada pada *game* kisah budak pahatu lalis

1. Menu Utama

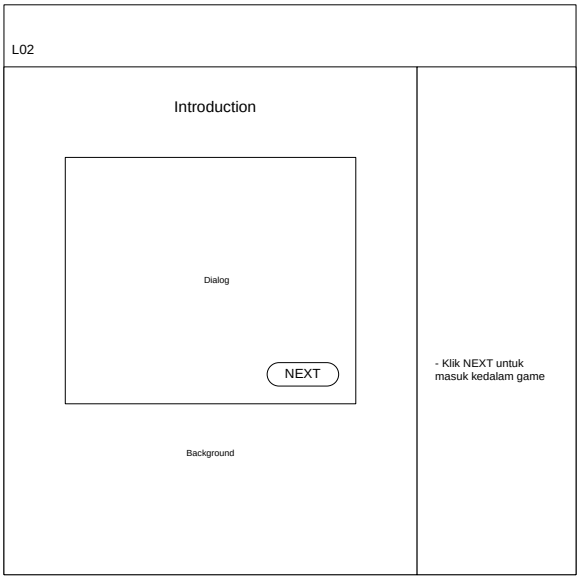
Menu utama merupakan menu pertama atau *main* menu pada *game* kisah budak pahatu lalis. Gambar menu utama terdapat pada gambar 3.44



Gambar 3.40 Antarmuka Menu Utama

2. Menu Mulai Permainan

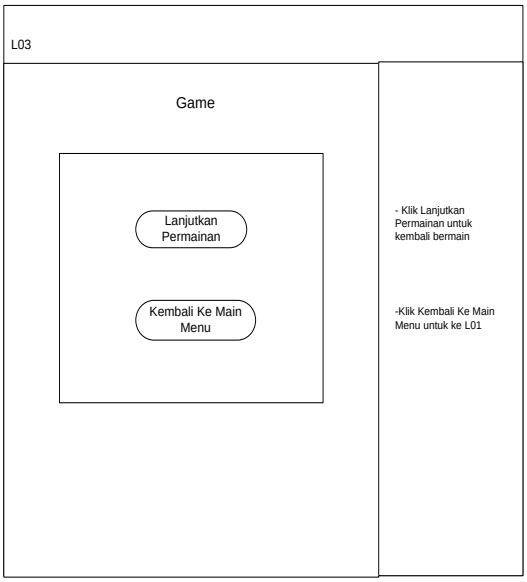
Menu mulai permainan merupakan menu masuk untuk memainkan *game*. Gambar menu mulai permainan terdapat pada gambar 3.45



Gambar 3.41 Antarmuka Menu Mulai Permainan

3. Menu *Pause*

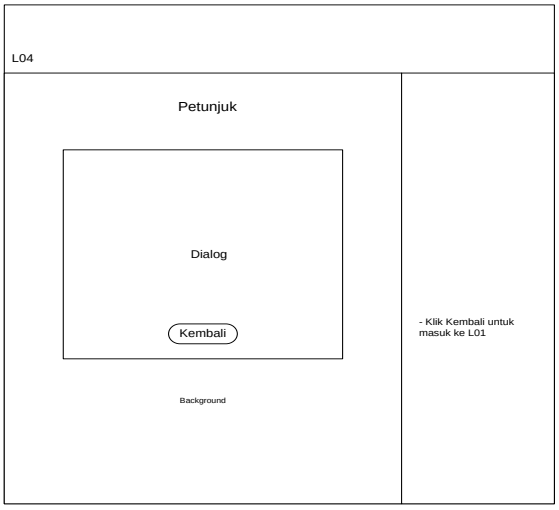
Menu *pause* merupakan menu masuk untuk keluar ketika bermain, menu pause bisa juga sebagai menu untuk melakukan jeda ketika bermain *game*. Gambar menu *pause* terdapat pada gambar 3.46



Gambar 3.42 Antarmuka Menu *Pause*

4. Menu Petunjuk

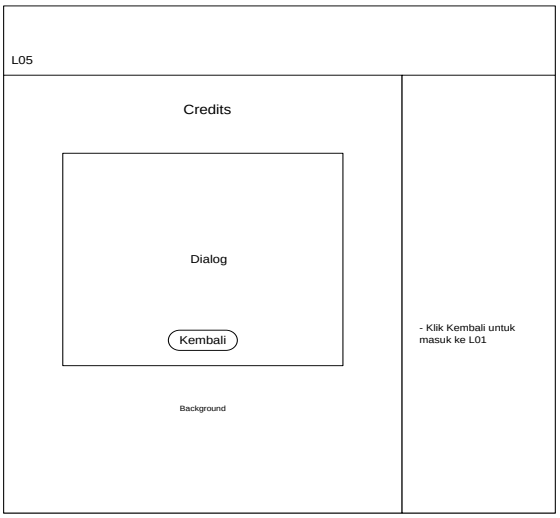
Menu petunjuk merupakan menu untuk membaca petunjuk tentang *game* budak pahatu lalis. Gambar menu petunjuk terdapat pada gambar 3.47



Gambar 3.43 Antarmuka Menu Petunjuk

5. Menu *Credits*

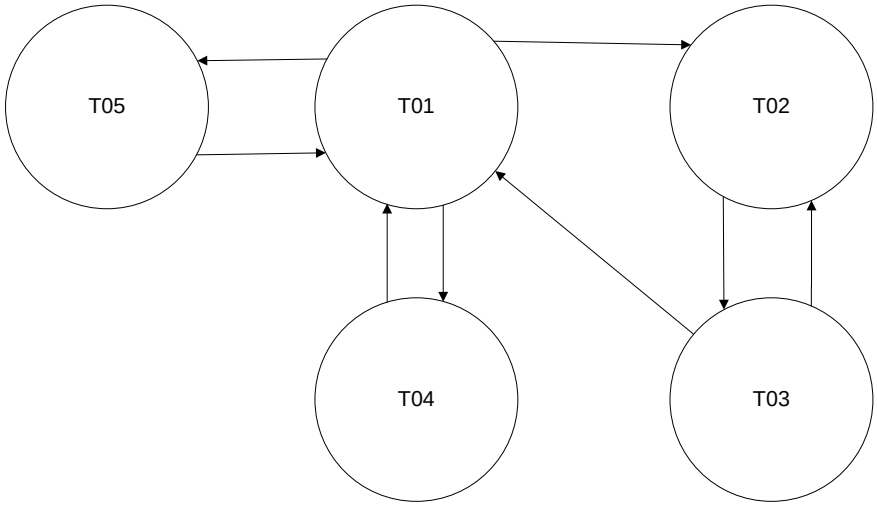
Menu *credits* merupakan menu untuk membaca pembuat *game* budak pahatu lalis. Gambar menu *credits* terdapat pada gambar 3.48



Gambar 3.44 Antarmuka Menu *Credits*

3.5.3 Jaringan Semantik

Jaringan Semantik merupakan perancangan secara moduler, disini fungsional menu pada program dipecah menjadi modul-modul kecil yang yang mudah dibuat, dites, dan dimodifikasi. Jaringan semantik pada *game* budak pahatu lalis terdapat pada gambar 3.49



Gambar 3.45 Jaringan Semantik