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

Analisis masalah menjelaskan proses identifikasi serta evaluasi mengenai *game* sejenis dan *game* yang akan dibangun.

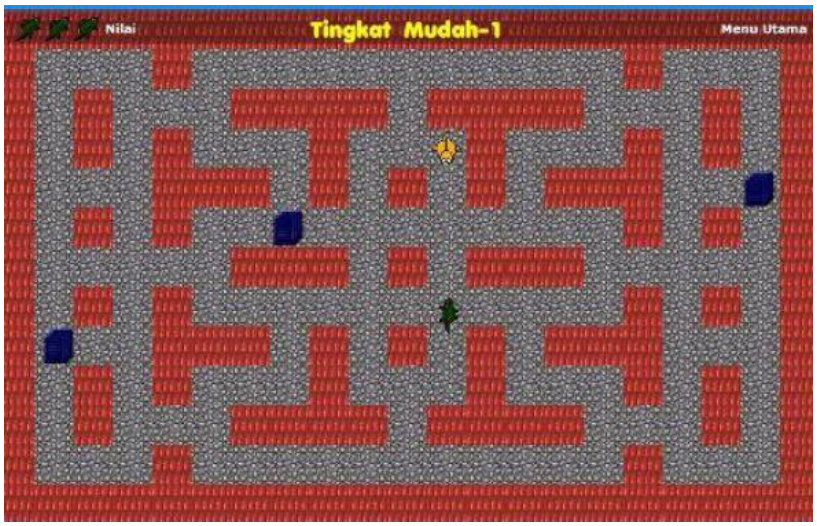
Saat ini *game* edukasi menjadi alat bantu yang sangat efektif dalam melakukan pembelajaran, banyak sekali beberapa *game* edukasi yang sudah disebarluaskan, seperti *game English Words* (edukasi bahasa), *Slice it!* (edukasi asah otak), kasir Karbitan Bejo (edukasi bisnis), *Crayon Physics Deluxe* (edukasi fisika), Petualangan Berburu Bintang Hijaiyah (edukasi islam), Merapi Boy Perjalanan Bocah Merapi Menghindari Wedhus Gembel (edukasi kepedulian sosial), *Trauma Center* (edukasi kesehatan), *Math Games For Kids* (edukasi matematika), Perjalanan Kartini (edukasi sejarah), Palawa (edukasi sejarah), P10Ner (edukasi sejarah) dan sebagainya.

Game yang akan dibahas dan menjadi acuan disini ialah *game Smart Labyrinth*, *Smart Shoot*, dan *Fun Math Tric*, Edukasi sejenis yang mengadaptasi matematika sebagai edukasi serupa dengan edukasi *game Fun Math For Smart* yang akan dibangun. Oleh karena itu akan dikembangkan tipe *game* sejenis dengan menambahkan beberapa *genre* di dalamnya.

3.1.2 Analisis *Game* Sejenis

Pada tahap ini dilakukan analisis terhadap *game* sejenis yang telah ada. Tujuan dari analisis ini yaitu untuk membandingkan dan mendapatkan konsep permainan dari *game* yang sudah ada untuk kemudian dikembangkan. *Game* yang dibandingkan dan akan menjadi acuan adalah *Smart Labyrinth*, *Smart Shoot,* dan *Fun Math Trikc Lite.*

1. Smart Labyrinth



Gambar 3.1 Jendela Tampilan *Game Smart Labyrinth*

Gambar 3.1 [7], menampilkan jendela tampilan *game smart labyrinth. Smart labyrinth* adalah permainan video yang berjenis permainan *Maze* atau labirin.

Dalam permainan ini pemain akan mengontrol karakter utama bernama coco, pemain harus mengambil barang yang disebarkan oleh musuh atau NPC. Dalam permainan ini pemain harus menghindari musuh atau NPC bernama dino.

Kelebihan dari permainan ini menmbahkan unsur hiburan kedalam sebuah edukasi. Kekurangan dalam game ini hanya memiliki satu *genre.*

2. Smart Shoot

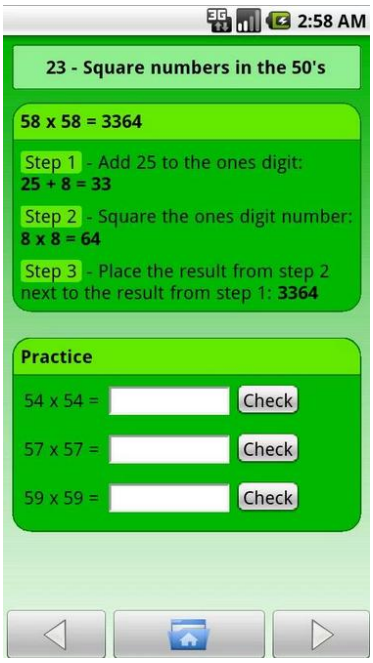


Gambar 3.2 Jendela Tampilan *Game Smart Shot*

Gambar 3.2 [13], menampilkan jendela tampilan *game smart Shoot. Smart Shoot* adalah permainan video yang berjenis permainan *Quiz.* Dalam permainan ini, pemain akan mencari materi matematika dan menjawab pertanyaan dengan cara mengarahkan target ke arah jawaban yang benar.

Kelebihan dalam permainan ini menampilkan semua materi sesuai dengan kebutuhan pengguna, dan kekurangan dalam permainan ini tidak ada unsur hiburan dan hanya memiliki satu *genre.*

3. Fun Math Trick Lite



Gambar 3.3 Jendela Tampilan *Game Fun Math Trick Lite*

Gambar 3.1 [12], menampilkan jendela tampilan *game fun math trick lite. Fun math trick lite* adalah aplikasi android matematika. Dalam aplikasi ini pemain akan belajar trik berhitung matematika, tantangan dalam permainannya adalah pemain harus menjawab pertanyaan dengan waktu yang telah ditentukan oleh sistem.

Kelebihan dalam permainan ini adalah memberikan trik-trik mengerjakan matematika dengan mudah, kekurangannya adalah tidak ada unsur hiburan dalam permainan.

3.1.3 Analisis Pembangunan *Game* Edukasi *Fun Math For Smart*

Game edukasi *fun math for smart* yang akan dibangun yaitu *game* yang mempunyai grafis 2D. *game* ini mempunyai beragam *gameplay* atau *multi genre* seperti *side scrolling, maze, auto-runner,* dan *quiz.*

Game yang dibangun menerapkan konsep *multi genre (Side Scrolling, Maze, Auto-Runner, Quiz)* sebagai lingkungan permainannya dengan tujuan agar mengenalkan nuansa petualangan yang diharapkan dapat melatih keterampilan, melatih motorik halus, melatih cara pemecahan masalah, melatih koordinasi mata dan tangan serta melatih kesabaran dan konsentrasi dalam menghadapi rintangan.

3.1.3.1 Story Line

Cerita *game* ini berawal dari seorang anak yang bernama Ken sedang pergi sekolah seperti biasa. Setelah sampai depan sekolah, gerbang utama sekolah tersebut terkunci rapat dan kunci tersbut diambil oleh penjahat yang datang dari dimensi lain. Kemudian Ken menantang penjahat tersebut untuk bertanding dengan menjawab soal matematika. Tiba-tiba salah satu penjahat yang menyimpan kunci sekolah lari ke portal dimensi lain dan kemudian Ken mengejarnya sebelum portal tersebut tertutup. Petualangan Ken dimulai di tempat tersebut untuk mendapatkan kunci yang diambil oleh penjahat tersebut.

3.1.3.2 Analisis Tingkat Kesulitan

Tabel 3.1 Deskripsi Tingkat Kesulitan

Stage 1	Misi dalam *stage* ini adalah mengalahkan musuh dengan cara menjawab pertanyaan dan mencari item yang tersembunyi di suatu tempat. Apabila tidak bisa menjawab darah karakter utama akan berkurang dan jika bisa menjawab maka musuh kalah dan menghilang.
Stage 2	Misi dalam *stage* ini adalah pemain atau karakter utama harus mengalahkan musuh dan mencari 3 kotak kemudian menjawab pertanyaan yang ada dalam kotak. Apabila musuh dapat di kalahkan dan menjawab pertanyaan maka pintu terbuka untuk menuju ke *stage* berikutnya. Musuh akan

	terus mengejar pemain apabila pemain berada dalam jangkauan musuh dan akan mengurangi life point jika pemain terKena musuh, dan jika pemain mati maka akan mengulang di *stage* tersebut
Stage 3	Misi dalam *stage* ini adalah karakter utama berlari dan harus menghindari rintangan. Untuk bisa menghindari rintangan karakter utama harus menjawab pertanyaan. Jika tidak bisa menjawab maka karakter utama akan terkena ranjau atau rintangan dan darah akan berkurang.
Stage 4	*Stage* ini adalah *stage* terakhir, di misi ini pemain harus mengalahkan bos dengan cara bertanding cerdas cermat matematika untuk bisa memenangkan game ini dan merebut kembali kunci utama sekolah.

3.1.3.3 *Gameplay*

Dalam permainan ini terdapat 4 *genre* atau 4 stage yang dapat dimainkan yaitu :

1. *Stage* 1 ber-*genre Side Scrolling.*
2. *Stage* 2 ber-*genre Maze.*
3. *Stage* 3 ber-*genre Auto-running.*
4. *Stage* 4 ber-*genre Quiz.*

Pada *stage* pertama, Ken berpetualang di dimensi lain dengan tampilan jenis permainan *side scrolling.* Disini Ken akan menghadapi monster prajurit dan Ken akan di beri pertanyaan oleh monster tersebut. Jika Ken bisa menjawab pertanyaan tersebut maka monster tersbut hilang, dan jika tidak bisa menjawab darah berkurang.

Pada *stage* berikutnya, Ken akan berhadapan dengan tampilan *maze*. disini Ken harus mencari 3 kotak yang dijaga oleh monster prajurit. Setelah menjawab semua pertanyaan dalam kotak tersebut muncul pintu keluar dari labirin.

Pada stage ke-3, Ken berlari mengejar monster yang sedang membawa kunci. Dalam *stage* ini akan ada banyak rintangan. Untuk menghadapi rintangan Ken harus menjawab pertanyaan, jika bisa menjawab pertanyaan maka akan terbebas dari rintangan jika tidak bisa menjawab Ken akan terKena rintangan tersebut dan darah akan berkurang atau *game over*.

Pada stage terakhir ini, Ken akan beradu otak dengan raja monster. Dalam stage ini Ken dan monster raja diberi 10 pertanyaan matematika. Jika dalam waktu yang ditentukan tidak bisa menjawab maka akan dijawab oleh monster raja dan darah akan berkurang, jika Ken yang menjawab maka darah moster akan berkurang.

3.1.3.4 *Scoring*

Pada *game fun math for smart* terdapat poin terhadap setiap menjawab pertanyaan, poin yang diberikan terhadap jawaban yang benar adalah 100 poin dan mengambil *item* adalah 50 poin, poin akan di akumulasikan diakhir permainan setelah menyelesaikan semua misi.

3.2 Analisis Algoritma A* (A-Star)

Algoritma yang akan diterapkan dalam *game* adalah algoritma pencarian A* (*A-star*). Algoritma A* di dalam *game* akan diterapkan pada NPC (*Non Playable Character*) sebagai penentuan jalur terpendek atau *pathfinding* menuju *target* (*player*). Musuh/NPC, *player*, penghalang, dan jalur yang bisa dilalui diasumsikan menjadi kotak-kotak atau disebut *node*.

Ilustrasinya adalah sebagai berikut :

Masukan posisi awal musuh/NPC pada titik (1,2) dan posisi pemain/*player* pada titik (5,2), terlihat pada gambar 3.4.

(1,1)	(2,1)	(3,1)	(4,1)	(5,1)
(1,2)	(2,2)	(3,2)	(4,2)	(5,2)
(1,3)	(2,3)	(3,3)	(4,3)	(5,3)

Gambar 3.4 Tampilan posisi awal musuh dan posisi player

Keterangan :

= Musuh/NPC

= Player

= Penghalang

Posisi awal musuh/ NPC adalah posisi terpilih. Posisi awal musuh akan selalu menjadi terpilih karena hanya mengecek satu posisi. Dapat dilihat pada gambar 3.5.

(1,1)	(2,1)
(1,2) Musuh/NPC	(2,2)
(1,3)	(2,3)

Gambar 3.5 Tampilan posisi awal musuh

Masukan posisi awal musuh menjadi posisi terbaik dan bangkitkan posisi yang berdekatan dengan posisi terpilih ((0,2),(1,1),(1,3),(2,2)). Terlihat pada gambar 3.6.

(0,1)	(1,1)	(2,1)
(0,2)	(1,2) Musuh/NPC	(2,2)
(0,3)	(1,3)	(2,3)

Gambar 3.6 Kondisi awal perhitungan koordinat ((0,2),(1,1),(1,3),(2,2))

Hitung nilai h, g, dan f pada posisi yang dibangkitkan. Lalu cari nilai f terendah pada posisi yang dibangkitkan dan masukan kedalam posisi terpilih.

Fungsi f sebagai estimasi fungsi evaluasi terhadap node n, dapat dituliskan :

$$f(n) = g(n) + h(n)$$

dimana,

f(n) : fungsi evaluasi (jumlah g(n) dengan h(n))

g(n) : nilai yang dibutuhkan untuk menempuh jarak dari point awal menuju kotak selanjutnya

h(n) : estimasi nilai yang dibutuhkan untuk menempuh jarak dari kotak selanjutnya untuk menempuh point tujuan.

Pergerakan diagonal pada map diperbolehkan, maka digunakan fungsi heuristic Non-Manhattan Distance. Maka fungsi heuristic yang digunakan adalah sebagai berikut :

h_diagonal(n) = min(abs(n.x-goal.x) + abs(n.y-goal.y))

h_orthogonal(n) = (abs(n.x-goal.x) + abs(n.y-goal.y))

h(n) = h_diagonal(n) + (h_orthogonal (n) – (2 * h_diagonal(n)))

Dengan perhitungan biaya seperti diatas, algoritma A* adalah complete dan optimal. Untuk lebih jelasnya berikut merupakan gambaran dari penggunaan algoritma A* dalam pencarian rute terpendek dari point A menuju point B :

1. Koordinat n(0,2)

Nilai n.x = 0

Nilai n.y = 2

Simpul tujuan(5,2) sehingga goal.x = 5, goal.y = 2

g(n) = g (0,2) = 2

h_orthogonal(n) = (abs(n.x-goal.x) + abs(n.y-goal.y))

h_orthogonal(0,2) = (abs(0 - 5) + abs(2 - 2))

= (abs(-5) + abs(0))

= 5

h_diagonal(n) = min(abs(n.x-goal.x) + abs(n.y-goal.y))

h_diagonal(0,2) = min(abs(0 - 5)+abs(2 - 2))

= min(abs(-5)+abs(0))

= min 5

h(n) = h_diagonal(n) + (h_orthogonal (n) – (2 * h_diagonal(n)))

h(0,2) = -5 + (5-(2*(-5)))

= -5+15

= 10

f (0,2) = g (0,2) + h (0,2)

= 2 + 10

= 12

2. Koordinat n(1,1)

Nilai n.x = 1

Nilai n.y = 1

Simpul tujuan(5,2) sehingga goal.x = 5, goal.y = 2

g (1,1) = 1

h_orthogonal(n) = (abs(n.x-goal.x) + abs(n.y-goal.y))

h_orthogonal(1,1) = (abs(1 - 5) + abs(1 - 2))

= (abs(-4) + abs(-1))

= 5

h_diagonal(n) = min(abs(n.x-goal.x) + abs(n.y-goal.y))

h_diagonal(1,1) = min(abs(1 - 5)+abs(1 - 2))

= min(abs(-4)+abs(-1))

= min 5

h(n) = h_diagonal(n) + (h_orthogonal (n) – (2 * h_diagonal(n)))

h(1,1) = -5 + (5-(2*(-5)))

= -5 + 20

= 15

f (1,1) = g (1,1) + h (1,1)

= 1 +15

= 16

3. Koordinat n(1,3)

Nilai n.x = 1

Nilai n.y = 3

Simpul tujuan(5,2) sehingga goal.x = 5, goal.y = 2

g (1,3) = 3

h_orthogonal(n) = (abs(n.x-goal.x) + abs(n.y-goal.y))

h_orthogonal(1,3) = (abs(1 - 5) + abs(3 - 2))

= (abs(-4) + abs(1))

= 3

h_diagonal(n) = min(abs(n.x-goal.x) + abs(n.y-goal.y))

h_diagonal(1,3) = min(abs(1 - 5)+abs(3 - 2))

= min(abs(-4)+abs(1))

= min 3

h(n) = h_diagonal(n) + (h_orthogonal (n) – (2 * h_diagonal(n)))

h(1,3) = -3 + (3-(2*(-3)))

= -3 + 9

= 3

f (1,3) = g (1,3) + h (1,3)

= 3 + 3

= 6

4. Koordinat n(2,2)

Nilai n.x = 2

Nilai n.y = 2

Simpul tujuan(5,2) sehingga goal.x = 5, goal.y = 2

g(n) = g (2,2) = 2

h_orthogonal(n) = (abs(n.x-goal.x) + abs(n.y-goal.y))

h_orthogonal(2,2) = (abs(2 - 5) + abs(2 - 2))

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

= 3

h_diagonal(n) = min(abs(n.x-goal.x) + abs(n.y-goal.y))

h_diagonal(2,2) = min(abs(2 - 5)+abs(2 - 2))

= min(abs(-3)+abs(0))

= min 3

h(n) = h_diagonal(n) + (h_orthogonal (n) – (2 * h_diagonal(n)))

h(2,2) = -3 + (3-(2*(-3)))

= -3 + 9

= 3

f (2,2) = g (0,2) + h (0,2)

= 2 + 3

= 5

Berdasarkan perhitungan langkah pertama maka hasil perhitungan memiliki nilai f tekecil yaitu 5. Terlihat pada gambar 3.7

(0,1)	(1,1) 16	(2,1)
(0,2) 12	(1,2) Musuh/NPC	(2,2) 5
(0,3)	(1,3) 6	(2,3)

Gambar 3.6 Hasil perhitungan koordinat ((0,2),(1,1),(1,3),(2,2))

Ulangi langkah diatas hingga mendapatkan posisi pemain/*player* (5,2). Hasil dari pencarian jalur menggunakan algoritma A* pada *game.* Jarak terpendek yang didapatkan adalah posisi (5,2) sebagai posisi *player*/pemain. Terlihat pada gambar 3.7.

(0,1)	(1,1) 16	(2,1)	(3,1)	(4,1)	(5,1)	(6,1)
(0,2) 12	(1,2) Musuh/NPC	(2,2) 5	(3,2) Penghalang	(4,2)	(5,2) Player 5	(6,2)
(0,3)	(1,3) 6	(2,3)	(3,3)	(4,3) 8	(5,3)	(6,3) 9
(0,4)	(1,4)	(2,4)	(3,4)	(4,4)	(5,4)	(6,4)

Gambar 3.6 Hasil Pencarian Jalur Menggunakan Algoritma A*

3.3 Analisis Kebutuhan Non Fungsional

Analisis non-fungsional merupakan analisis yang dibutuhkan untuk menentukan spesifikasi kebutuhan sistem. Spesifikasi ini juga meliputi elemen atau komponen-komponen apa saja yang dibutuhkan untuk sistem yang akan dibangun sampai dengan sistem tersebut diimplementasikan. Analisis kebutuhan ini juga menentukan

spesifikasi masukan yang diperlukan sistem, keluaran yang akan dihasilkan sistem dan proses yang dibutuhkan untuk mengolah masukan sehingga menghasilkan suatu keluaran yang diinginkan.

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

3.3.1 Analisis 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. Agar aplikasi dapat berjalan dengan baik, maka dibutuhkan perangkat keras yang sesuai dengan kebutuhan aplikasi. Spesifi yang digunakan dalam pembuatan *game* ini adalah sebagai berikut:

Tabel 3.2 Spesifikasi Perangkat Keras Developer

Nama Perangkat	Spesifikasi
Prosesor	Intel Core 2 Duo 2 GHz
RAM	1GB
Harddisk	320GB
Display	Resolusi 1280 x 800
Keyboard	Standar
Mouse	Standar

Adapun spesifikasi standar perangkat keras yang dibutuhkan untuk dapat menjalankan aplikasi yang akan dibangun adalah:

Tabel 3.3 Spesifikasi Perangkat Keras Pengguna

Nama Perangkat	Spesifikasi
Prosesor	Intel Pentium 4 1GHz
RAM	512MB
Harddisk	40GB
Display	Resolusi 800 x 600
Keyboard	Standar
Mouse	Standar

3.3.2 Analisis Perangkat Lunak

Analisis perangkat lunak digunakan untuk mengetahui spesifikasi perangkat keras dalam menjalankan aplikasi permainan *Fun Math For Smart*..

Tabel 3.4 Spesifikasi Perangkat Lunak Developer

Nama Perangkat Lunak	Spesifikasi
Sistem Operasi	Microsoft Windows XP, Linux
Tools	Construct 2
Browser	Mozilla Firefox, Google Chrome

Tabel 3.5 Spesifikasi Perangkat Lunak Pengguna

Nama Perangkat Lunak	Spesifikasi
Sistem Operasi	Microsoft Windows XP, Linux
Browser	Internet Explorer 9+, Mozilla Firefox, Google Chrome, Opera atau support html5

3.3.3 Analisis Pengguna

Spesifikasi pengguna pada permainan *Fun Math For Smart*, seperti pada tabel 3.7.

Tabel 3.6 Analisis Pengguna

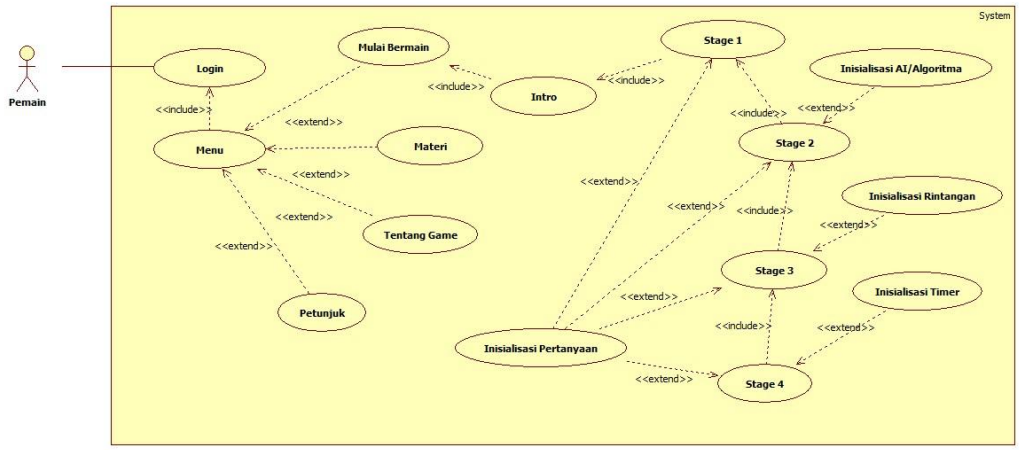
Ketentuan	Spesifikasi
Pengguna	Pria/Wanita
Usia	10 tahun keatas
Keahlian	Dapat menggunakan komputer

3.4 Analisis Kebutuhan Fungsional

Analisis kebutuhan fungsional menggambarkan proses kegiatan yang akan diterapkan dalam sebuah sistem dan menjelaskan kebutuhan yang diperlukan sistem agar sistem dapat berjalan dengan baik sesuai kebutuhan.

3.4.1 Diagram *Use Case*

Secara umum, interaksi antara pengguna yaitu pemain dengan aplikasi permainan *Fun Math For Smart* digambarkan pada diagram *Use Case*, seperti pada gambar 3.7.



Gambar 3.7 Diagram *Use Case Fun Math For Smart*

3.4.2 Skenario *Use Case*

Skenario pada diagram *use case* berisi tentang alur kejadian '*flow of 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 ini adalah sebagai berikut:

1. Skenario *Use Case Stage* 1

Skenario *use case* ini menjelaskan interaksi antara aktor, yaitu pemain, dengan *use case* mulai permainan *stage* 1 yang dijelaskan pada tabel 3.8.

Nama *Use Case* : *Stage* 1

Skenario :

Tabel 3.7 Skenario *Use Case Stage* 1

Aksi Aktor	Reaksi Sistem
Skenario Normal	
	a. Tampilkan layar permainan *stage* 1
	b. Inisialisasi informasi awal pemain
	c. Inisialisasi posisi Hero, NPC, Audio, Item, Score dan Pertanyaan
d. Mengontrol hero dan menjawab pertanyaan	
e. Menyelesaikan misi permainan	
	f. *Update* status permainan
	g. Menampilkan informasi untuk berpindah ke *stage* selanjutnya
Skenario Alternatif	
	a. Tampilkan layar permainan *stage* 1
	b. Inisialisasi informasi awal pemain
	c. Inisialisasi posisi Hero, NPC, Audio, Item, Score dan Pertanyaan
d. Mengontrol hero dan menjawab	

Aksi Aktor	Reaksi Sistem
pertanyaan	
e. Misi permainan tidak terselesaikan	
	f. Kembali ke tampilan awal permainan *stage* 1
	g. Inisialisasi informasi awal pemain
	h. Inisialisasi posisi Hero, NPC, Audio, Item, Score dan Pertanyaan
i. Mengontrol hero dan menjawab pertanyaan	
j. Menyelesaikan misi permainan	
	k. *Update* status permainan
	l. Menampilkan informasi untuk berpindah ke *stage* selanjutnya

2. Skenario *Use Case Stage* 2

Skenario *use case* ini menjelaskan interaksi antara aktor, yaitu pemain, dengan *use case* mulai permainan *stage* 2 yang dijelaskan pada tabel 3.9.

Nama *Use Case* : *Stage* 2

Skenario :

Tabel 3.8 Skenario *Use Case Stage* 2

Aksi Aktor	Reaksi Sistem
Skenario Normal	
	a. Tampilkan layar permainan *stage* 2
	b. Inisialisasi informasi awal pemain
	c. Inisialisasi posisi Hero, NPC, Audio, AI, Item, Score dan Pertanyaan
d. Mengontrol hero dan menjawab pertanyaan	

Aksi Aktor	Reaksi Sistem
e. Menyelesaikan misi permainan	
	f. *Update* status permainan
	g. Menampilkan informasi untuk berpindah ke *stage* selanjutnya
Skenario Alternatif	
	a. Tampilkan layar permainan *stage* 2
	b. Inisialisasi informasi awal pemain
	c. Inisialisasi posisi Hero, NPC, Audio, AI, Item, Score dan Pertanyaan
d. Mengontrol hero dan menjawab pertanyaan	
e. Misi permainan tidak terselesaikan	
	f. Kembali ke tampilan awal permainan *stage* 2
	g. Inisialisasi informasi awal pemain
	h. Inisialisasi posisi Hero, NPC, Audio, AI, Item, Score dan Pertanyaan
i. Mengontrol hero dan menjawab pertanyaan	
j. Menyelesaikan misi permainan	
	k. *Update* status permainan
	l. Menampilkan informasi untuk berpindah ke *stage* selanjutnya

3. Skenario *Use Case Stage* 3

Skenario *use case* ini menjelaskan interaksi antara aktor, yaitu pemain, dengan *use case* mulai permainan *stage* 3 yang dijelaskan pada tabel 3.10.

Nama *Use Case* : *Stage* 3

Skenario :

Tabel 3.9 Skenario *Use Case Stage* 3

Aksi Aktor	Reaksi Sistem
Skenario Normal	
	a. Tampilkan layar permainan *stage* 3
	b. Inisialisasi informasi awal pemain
	c. Inisialisasi posisi Hero, Rintangan, Audio, Item, Score dan Pertanyaan
d. Mengontrol hero dan menjawab pertanyaan	
e. Menyelesaikan misi permainan	
	f. *Update* status permainan
	g. Menampilkan informasi untuk berpindah ke *stage* selanjutnya
Skenario Alternatif	
	a. Tampilkan layar permainan *stage* 3
	b. Inisialisasi informasi awal pemain
	c. Inisialisasi posisi Hero, Rintangan, Audio, Item, Score dan Pertanyaan
d. Mengontrol hero dan menjawab pertanyaan	
e. Misi permainan tidak terselesaikan	
	f. Kembali ke tampilan awal permainan *stage* 3
	g. Inisialisasi informasi awal pemain
	h. Inisialisasi posisi Hero, Rintangan, Audio, Item, Score dan Pertanyaan
i. Mengontrol hero dan menjawab	

Aksi Aktor	Reaksi Sistem
pertanyaan	
j. Menyelesaikan misi permainan	
	k. *Update* status permainan
	l. Menampilkan informasi untuk berpindah ke *stage* selanjutnya

4. Skenario *Use Case Stage* 4

Skenario *use case* ini menjelaskan interaksi antara aktor, yaitu pemain, dengan *use case stage* 4 yang dijelaskan pada tabel 3.11.

Nama *Use Case* : *Stage* 4

Skenario :

Tabel 3.10 Skenario *Use Case Stage* 4

Aksi Aktor	Reaksi Sistem
Skenario Normal	
	a. Tampilkan layar permainan *stage* 4
	b. Inisialisasi informasi awal pemain
	c. Inisialisasi posisi Hero, NPC, Audio, Timer, Score dan Pertanyaan
d. Mengontrol hero dan menjawab pertanyaan	
e. Menyelesaikan misi permainan	
	f. *Update* status permainan
	g. Menampilkan informasi untuk berpindah ke *stage* selanjutnya
Skenario Alternatif	
	a. Tampilkan layar permainan *stage* 4
	b. Inisialisasi informasi awal pemain
	c. Inisialisasi posisi Hero, NPC,

Aksi Aktor	Reaksi Sistem
	Audio, Timer, Score dan Pertanyaan
d. Mengontrol hero dan menjawab pertanyaan	
e. Misi permainan tidak terselesaikan	
	f. Kembali ke tampilan awal permainan *stage* 4
	g. Inisialisasi informasi awal pemain
	h. Inisialisasi posisi Hero, NPC, Audio, Timer, Score dan Pertanyaan
i. Mengontrol hero dan menjawab pertanyaan	
j. Menyelesaikan misi permainan	
	k. *Update* status permainan
	l. Menampilkan informasi untuk berpindah ke *stage* selanjutnya

5. Skenario *Use Case* Tampil Materi

Skenario *use case* ini menjelaskan interaksi antara aktor, yaitu pemain, dengan seluruh *use case* tampil materi yang dijelaskan pada tabel 3.11.

Nama *Use Case* : Tampil Materi

Skenario :

Tabel 3.11 Skenario *Use Case* Tampil Materi

Aksi Aktor	Reaksi Sistem
Skenario Normal	
a. Klik Materi	
	b. Tampil layar materi
Skenario Alternatif	

Aksi Aktor	Reaksi Sistem
a. Klik Materi	
	b. Layar materi tidak tampil
	c. Kembali ke menu utama
d. Klik Materi	
	e. Tampil layar materi

6. Skenario *Use Case* Tampil Petunjuk

Skenario *use case* ini menjelaskan interaksi antara aktor, yaitu pemain, dengan seluruh *use case* tampil petunjuk yang dijelaskan pada tabel 3.12.

Nama *Use Case* : Tampil Petunjuk

Skenario :

Tabel 3.12 Skenario *Use Case* Tampil Petunjuk

Aksi Aktor	Reaksi Sistem
Skenario Normal	
a. Klik Petunjuk	
	b. Tampil layar petunjuk permainan
Skenario Alternatif	
a. Klik Petunjuk	
	b. Layar petunjuk permainan tidak tampil
	c. Kembali ke menu utama
d. Klik Petunjuk	
	e. Tampil layar petunjuk permainan

7. Skenario *Use Case* Tampil Tentang *Game*

Skenario *use case* ini menjelaskan interaksi antara aktor, yaitu pemain, dengan seluruh *use case* tampil tentang *game* yang dijelaskan pada tabel 3.13.

Nama *Use Case* : Tampil Tentang *Game*

Skenario :

Tabel 3.13 Skenario *Use Case* Tampil Tentang *Game*

Aksi Aktor	Reaksi Sistem
Skenario Normal	
a. Klik Tentang *Game*	
	b. Tampil layar tentang *game*
Skenario Alternatif	
a. Klik Tentang *Game*	
	b. Layar petunjuk tentang *game* tidak tampil
	c. Kembali ke menu utama
d. Klik Tentang *Game*	
	e. Tampil layar tentang *game*

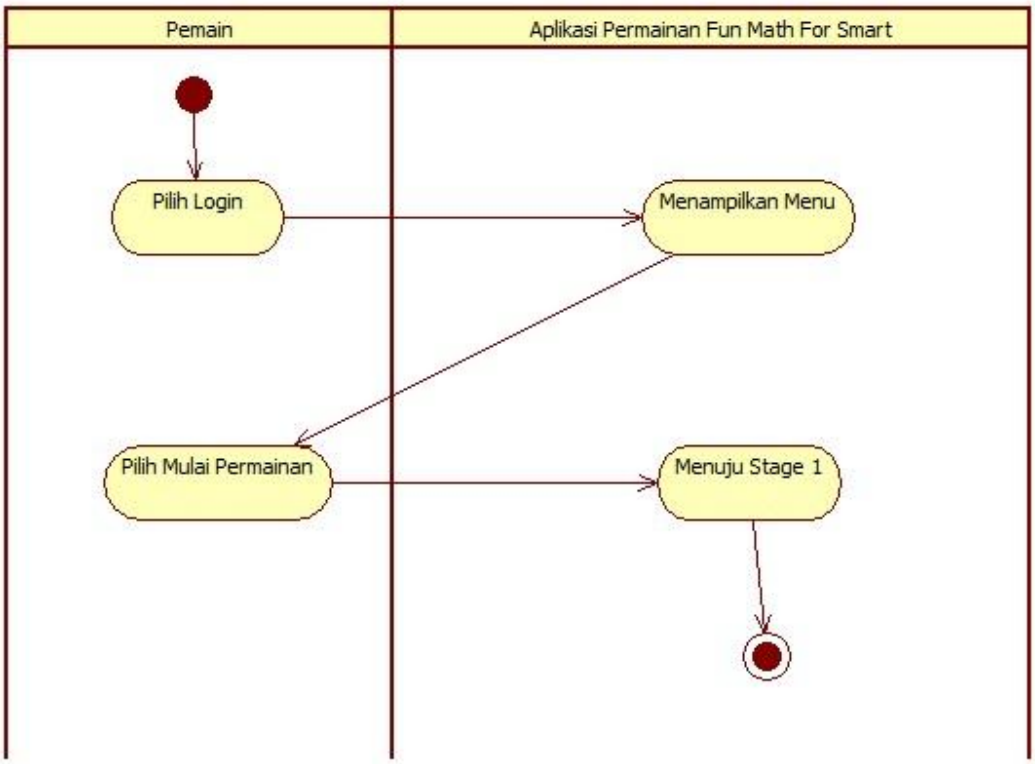
3.4.3 Diagram *Activity*

Diagram ini berisi penjelasan secara lengkap mengenai urutan aktivitas yang terjadi antara pemain dengan aplikasi permainan *Fun Math For Smart*.

A. Diagram *Activity Menu*

Proses ini terjadi setelah pemain login menuju ke menu aplikasi permainan *Fun Math For Smart.* Sistem ini akan menampilkan menu kemudian memilih mulai permainan.

Diagram *activity* mulai permainan *stage* 1 terdapat pada gambar 3.5.

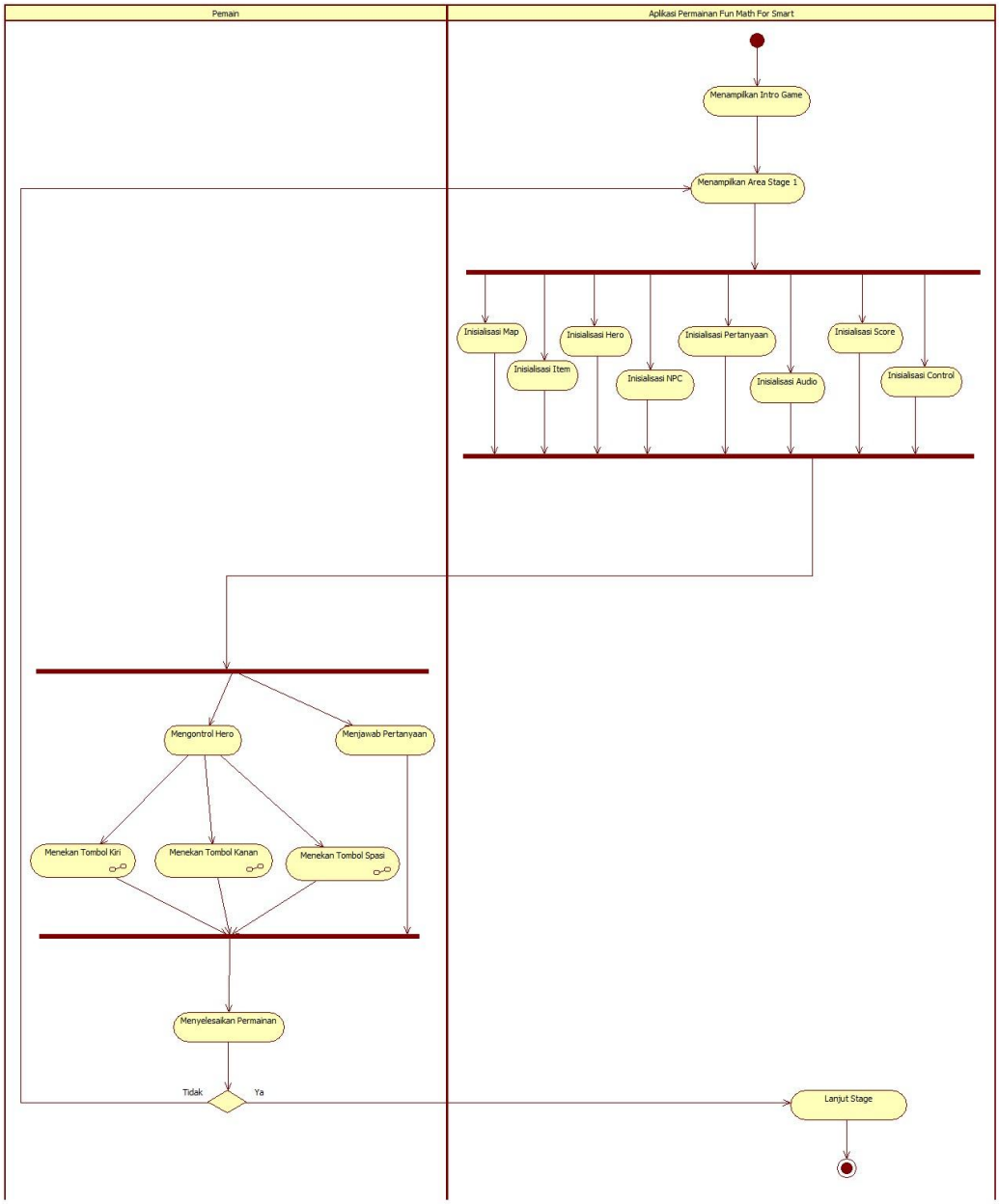


Gambar 3.8 Diagram *Activity* Menu *Fun Math For Smart*

B. Diagram *Activity Stage* 1

Proses ini terjadi saat pemain memilih menu mulai permainan pada menu utama aplikasi permainan *Fun Math For Smart.* Sistem akan menampilkan arena permainan dimana pemain akan memulai dan menyelesaikan misi permainan pada *stage* 1, apabila misi tidak terselesaikan maka pemain kembali ke titik awal untuk memulai permainan *stage* 1 kembali.

Diagram *activity* mulai permainan *stage* 1 terdapat pada gambar 3.6.

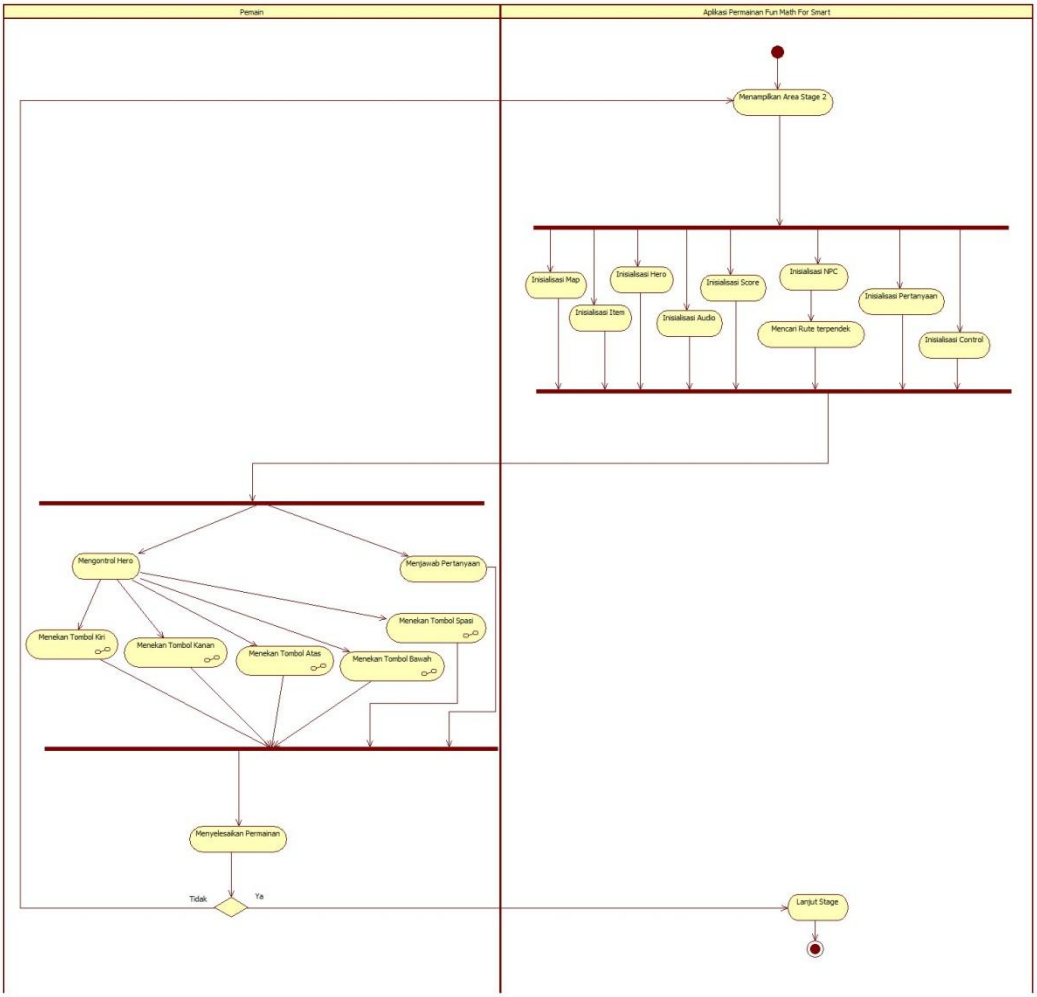


Gambar 3.9 Diagram *Activity Stage* 1 *Fun Math For Smart*

C. Diagram *Activity Stage* 2

Proses ini terjadi saat pemain memilih menu mulai permainan pada menu utama aplikasi permainan *Fun Math For Smart.* Sistem akan menampilkan arena permainan dimana pemain akan memulai dan menyelesaikan misi permainan pada *stage* 2, apabila misi tidak terselesaikan maka pemain kembali ke titik awal untuk memulai permainan *stage* 2 kembali.

Diagram *activity* mulai permainan *stage* 2 terdapat pada gambar 3.7.

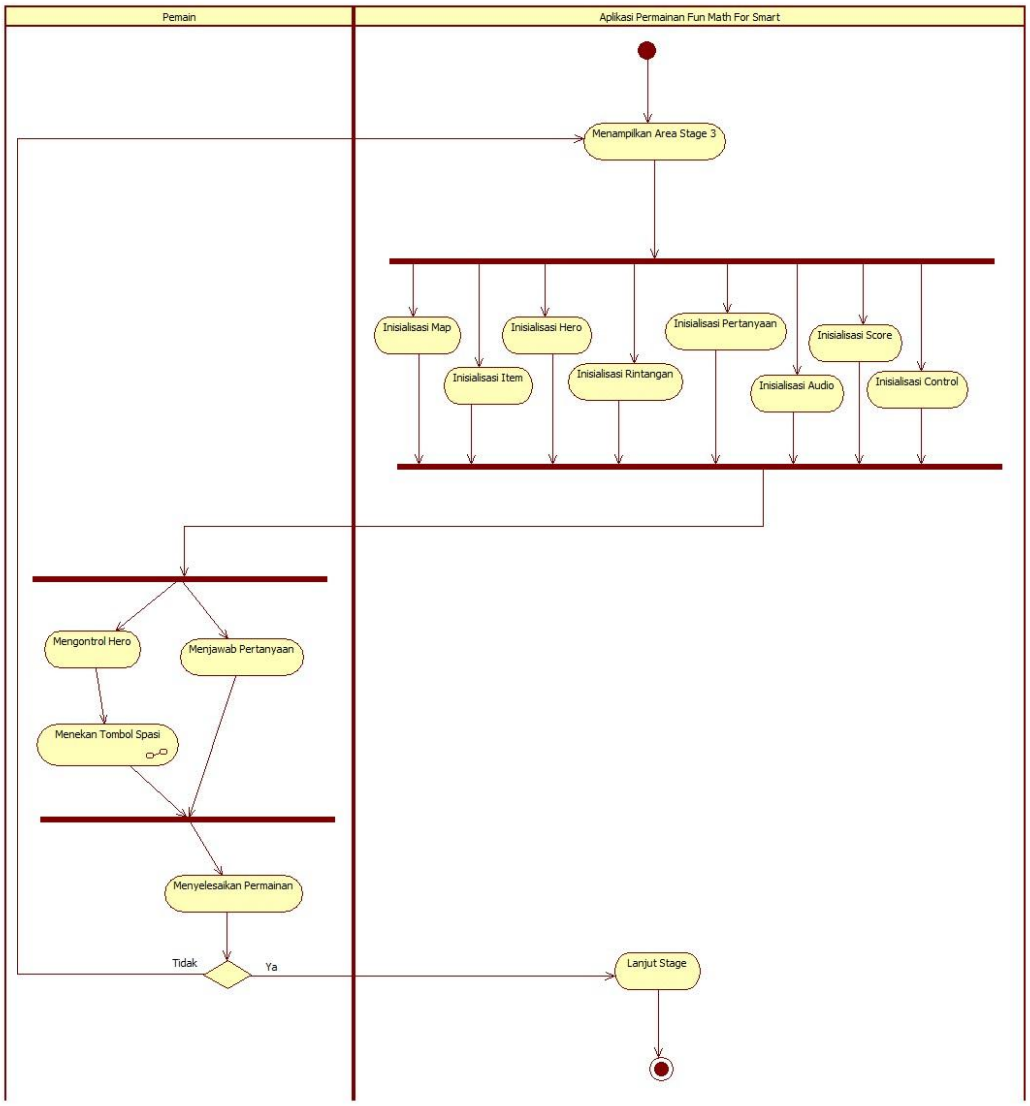


Gambar 3.10 Diagram *Activity Stage* 2 *Fun Math For Smart*

D. Diagram *Activity Stage* 3

Proses ini terjadi saat pemain memilih menu mulai permainan pada menu utama aplikasi permainan *Fun Math For Smart.* Sistem akan menampilkan arena permainan dimana pemain akan memulai dan menyelesaikan misi permainan pada *stage* 3, apabila misi tidak terselesaikan maka pemain kembali ke titik awal untuk memulai permainan *stage* 3 kembali.

Diagram *activity* mulai permainan *stage* 3 terdapat pada gambar 3.8.

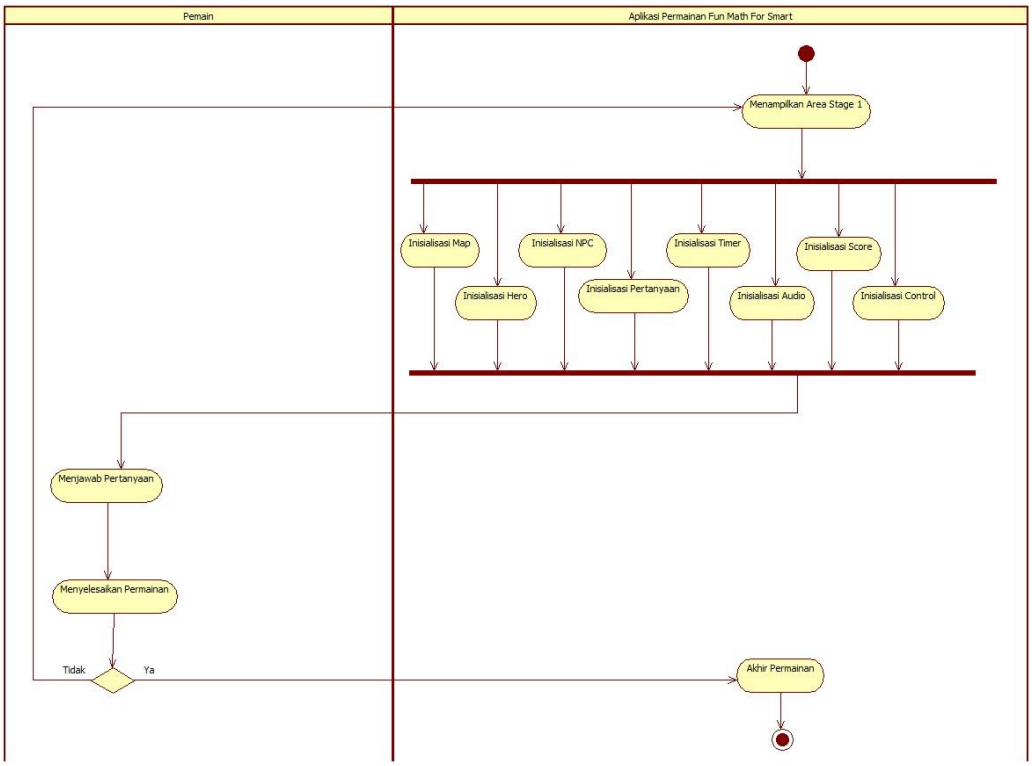


Gambar 3.11 Diagram *Activity Stage* 3 *Fun Math For Smart*

E. Diagram *Activity Stage* 4

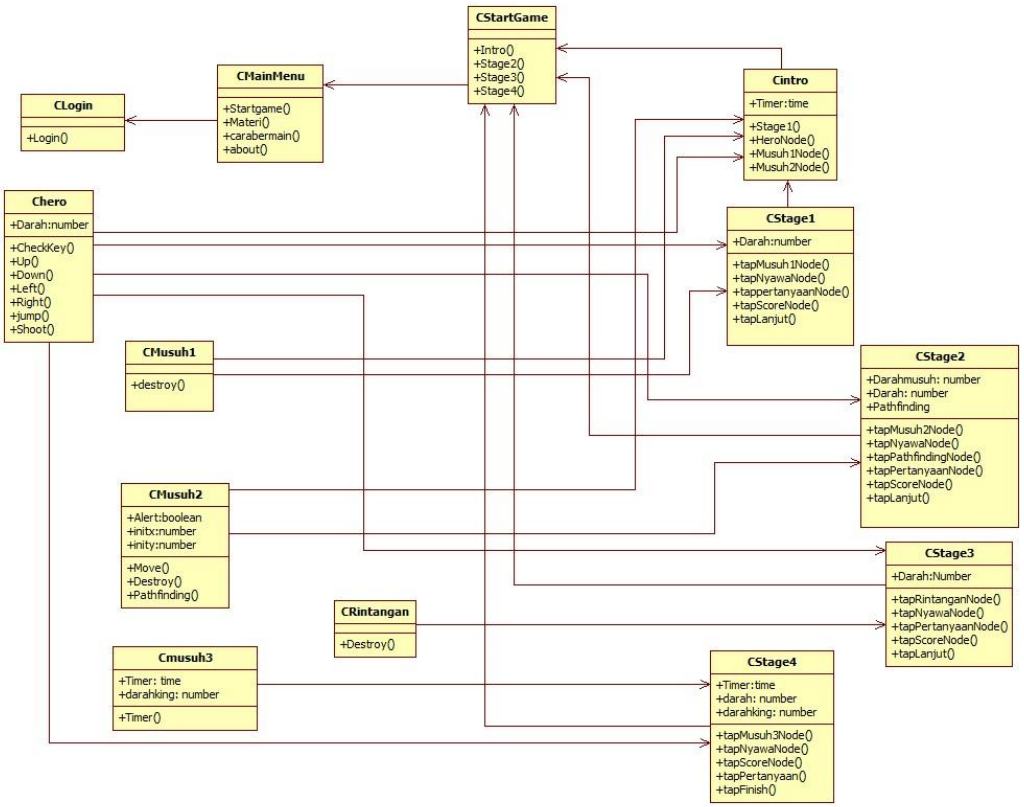
Proses ini terjadi saat pemain memilih menu mulai permainan pada menu utama aplikasi permainan *Fun Math For Smart.* Sistem akan menampilkan arena permainan dimana pemain akan memulai dan menyelesaikan misi permainan pada *stage* 4, apabila misi tidak terselesaikan maka pemain kembali ke titik awal untuk memulai permainan *stage* 4 kembali.

Diagram *activity* mulai permainan *stage* 4 terdapat pada gambar 3.9.



Gambar 3.12 Diagram *Activity Stage* 4 *Fun Math For Smart*

3.4.4 Class Diagram

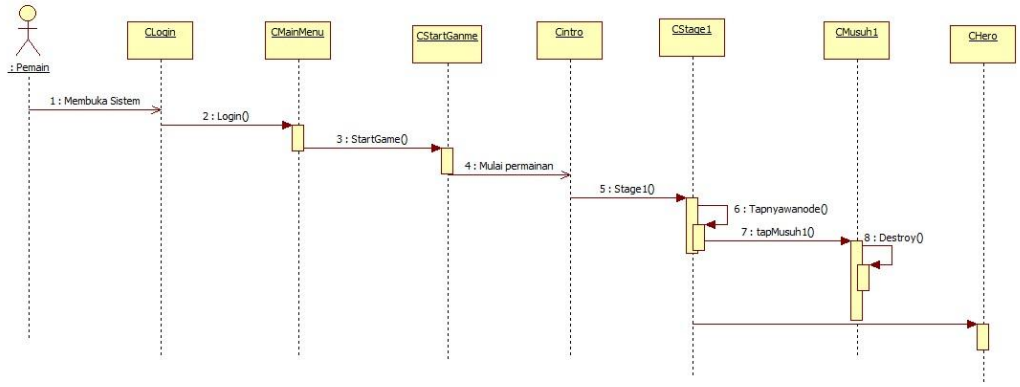


Gambar 3.13 Class Diagram *Stage* 4 *Fun Math For Smart*

3.4.5 Diagram *Sequence*

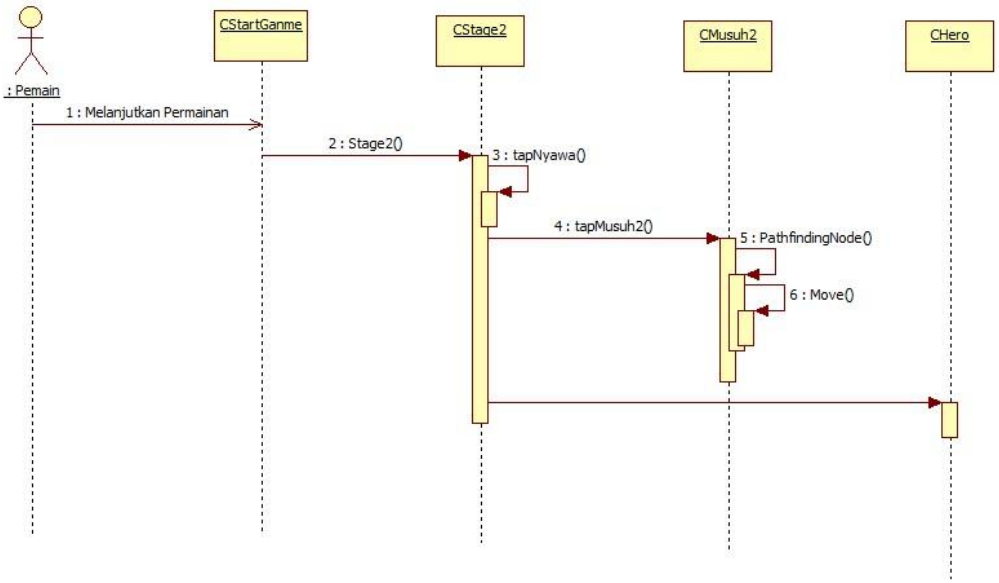
Berikut ini merupakan diagram *sequence* yang terdapat pada aplikasi permainan *Fun Math For Smart* berdasarkan *use case* yang ada.

A. Diagram *Sequence Stage 1*



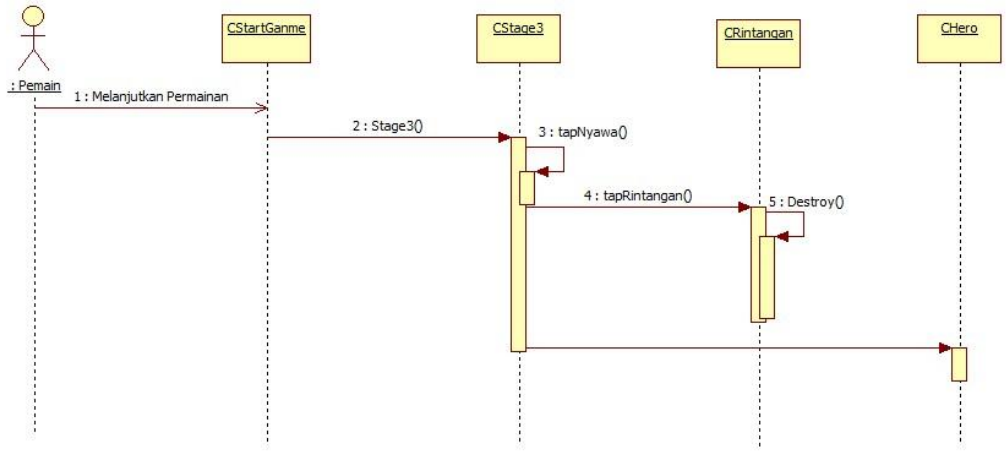
Gambar 3.14 Diagram *Sequence Stage 1*

B. Diagram *Sequence Stage 2*



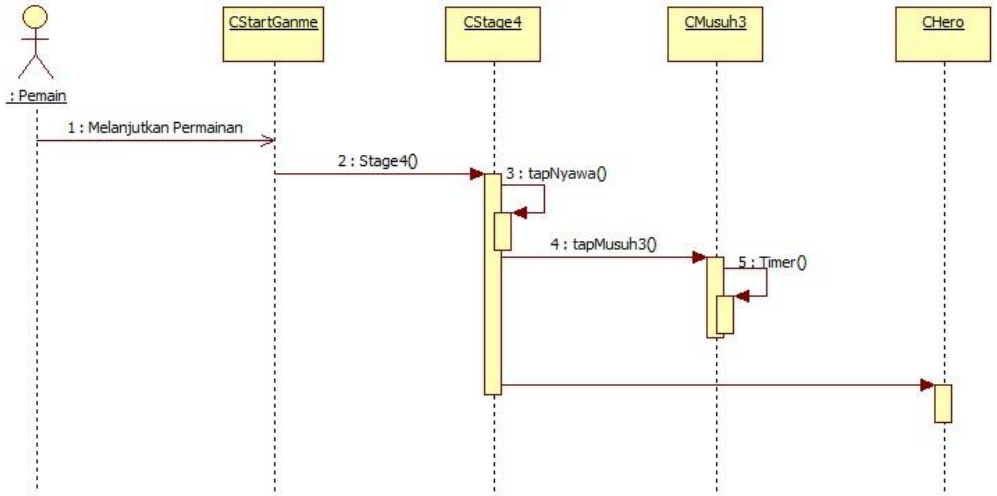
Gambar 3.15 Diagram *Sequence Stage 2*

C. Diagram *Sequence Stage 3*



Gambar 3.16 Diagram *Sequence Stage 3*

D. Diagram *Sequence Stage 4*



Gambar 3.17 Diagram *Sequence Stage 4*

3.5 Perancangan Sistem

Perancangan adalah suatu bagian dari metodologi pengembangan suatu perangkat lunak yang dilakukan setelah tahapan untuk memberikan gambaran secara terperinci.

Berdasarkan uraian diatas perancangan sistem merupakan tahapan dari siklus pengembangan sistem yang didefinisikan dari kebutuhan-kebutuhan fungsional dan persiapan untuk rancang bangun implementasi yang menggambarkan bagaimana suatu sistem dibentuk, yang dapat berupa penggambaran, perancangan, dan pembuatan sketsa atau pengaturan dari beberapa elemen yang terpisah kedalam satu kesatuan yang utuh dan berfungsi, juga menyangkut konfigurasi dari komponen-komponen perangkat keras dan perangkat lunak dari suatu sistem.

3.5.1 Perancangan Komponen Permainan

3.5.1.1 Entitas Game

Entitas yang terdapat di dalam *game* terbagi menjadi dua, yaitu karakter *hero* dan NPC yang berupa musuh, selain itu terdapat juga item yang digunakan untuk membantu *hero* dalam pertahanan.

Entitas karakter dan item yang terdapat di dalam game ditunjukkan pada tabel 3.14.

Tabel 3.14 Entitas karakter dan item

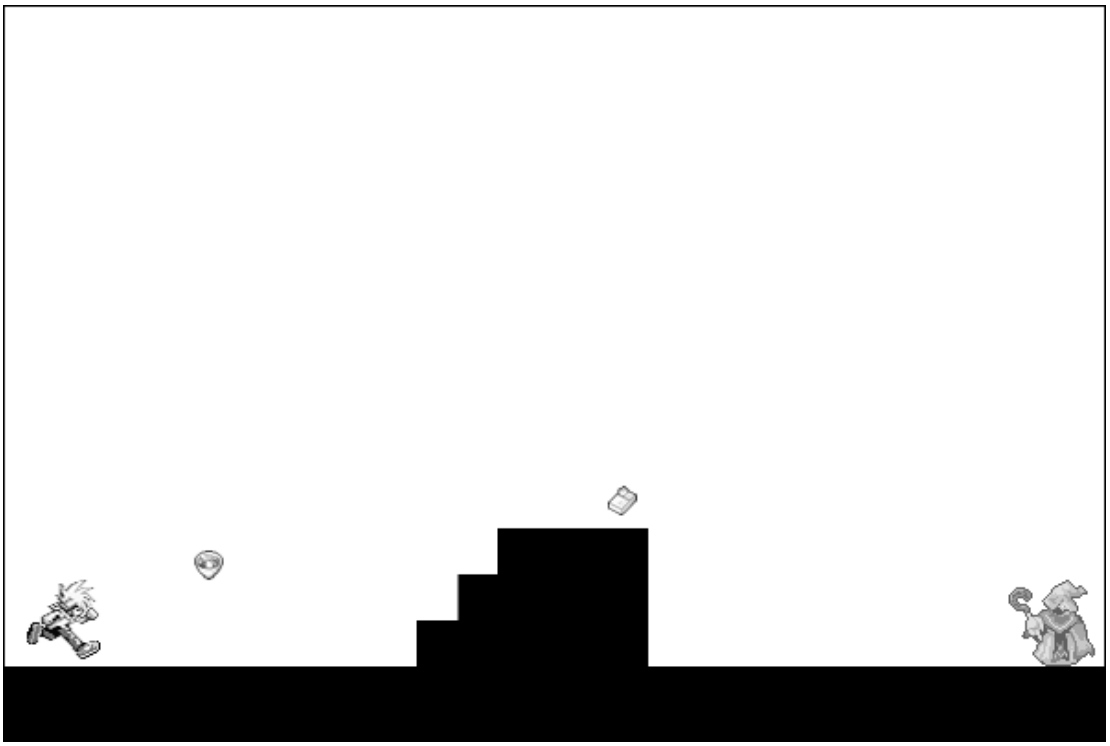
NO	NAMA	GAMBAR	PENJELASAN
1	Ken		Ken adalah seorang siswa dan karakter utama pada *game* ini. Ken memiliki kemampuan bergerak ke kiri, ke kanan dan melompat pada *stage* 1, bergerak ke kiri, ke kanan, ke atas, ke bawah dan menembak pada *stage* 2, dan

NO	NAMA	GAMBAR	PENJELASAN
			harus menjawab Pertanyaan.
2	Majo		Majo adalah karakter musuh pada pada *stage* 1. Pada *stage* 1 saat Ken beradu dengan Majo akan muncul pertanyaan dan Ken harus menjawab pertanyaan. HP=1
3	Mon		Mon adalah karakter musuh Pada *Stage* 2. Pada *Stage* ini Mon akan mengejar Ken. HP=2
4	King		King adalah musuh terakhir atau raja terakhir dalam permainan ini kemampuan musuh ini adalah menjawab pertanyaan. HP=5
5	Japfood		Sebuah item yang berfungsi untuk menambah poin sebesar 50.
6	Health		Sebuah item yang berfungsi untuk menambah health poin Pemain.
7	Kotak Pertanyaan		Sebuah item yang berfungsi untuk menampilkan pertanyaan.

3.5.1.2 Storyboard

Storyboard merupakan sketsa gambar yang disusun berurutan sesuai naskah yang ada, dan dengan storyboard kita dapat menyampaikan ide cerita kita kepada orang lain dengan mudah.

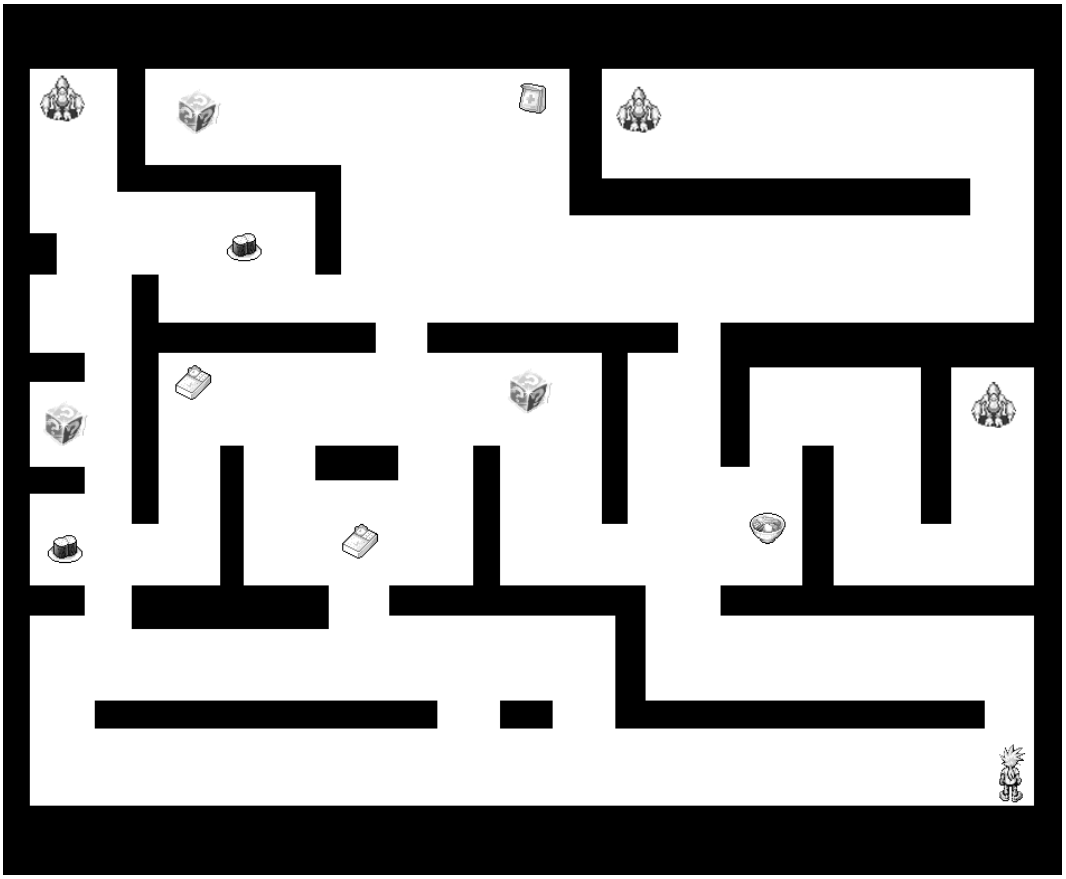
1. *Storyboard stage* 1



Gambar 3.18 *Storyboard Stage* 1

Pada *Stage 1*, pemain berlari mengambil item dan jika bertabrakan dengan musuh muncul pertanyaan. Pemain harus menjawab pertanyaan dengan cara menembak jawaban yang benar. Setelah berhasil menjawab pertanyaan bisa lanjut ke *stage* berikutnya jika gagal kembali ke poin awal.

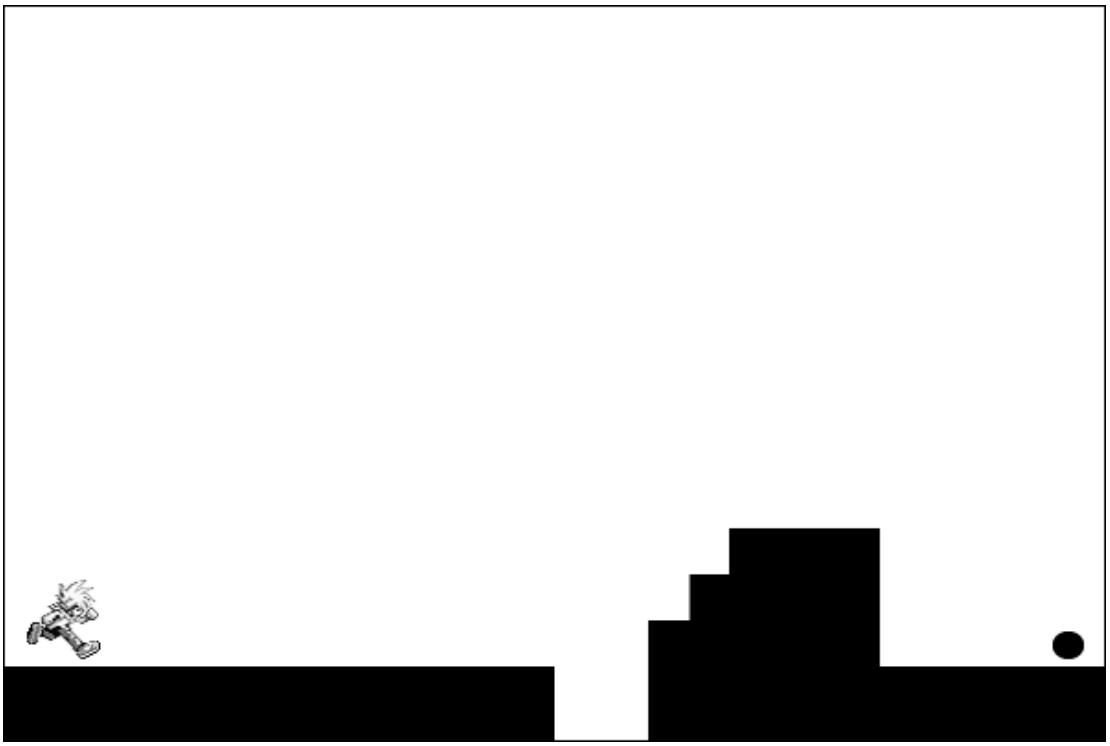
2. *Storyboard stage* 2



Gambar 3.19 *Storyboard Stage* 2

Pada *Stage 2,* Pemain harus mencari kotak rahasia dan pemain akan di hadang oleh musuh. Musuh pada *stage* ini sudah diterapkan algoritma A* *pathfinding* untuk menuju ke tempat hero akan mendekat.

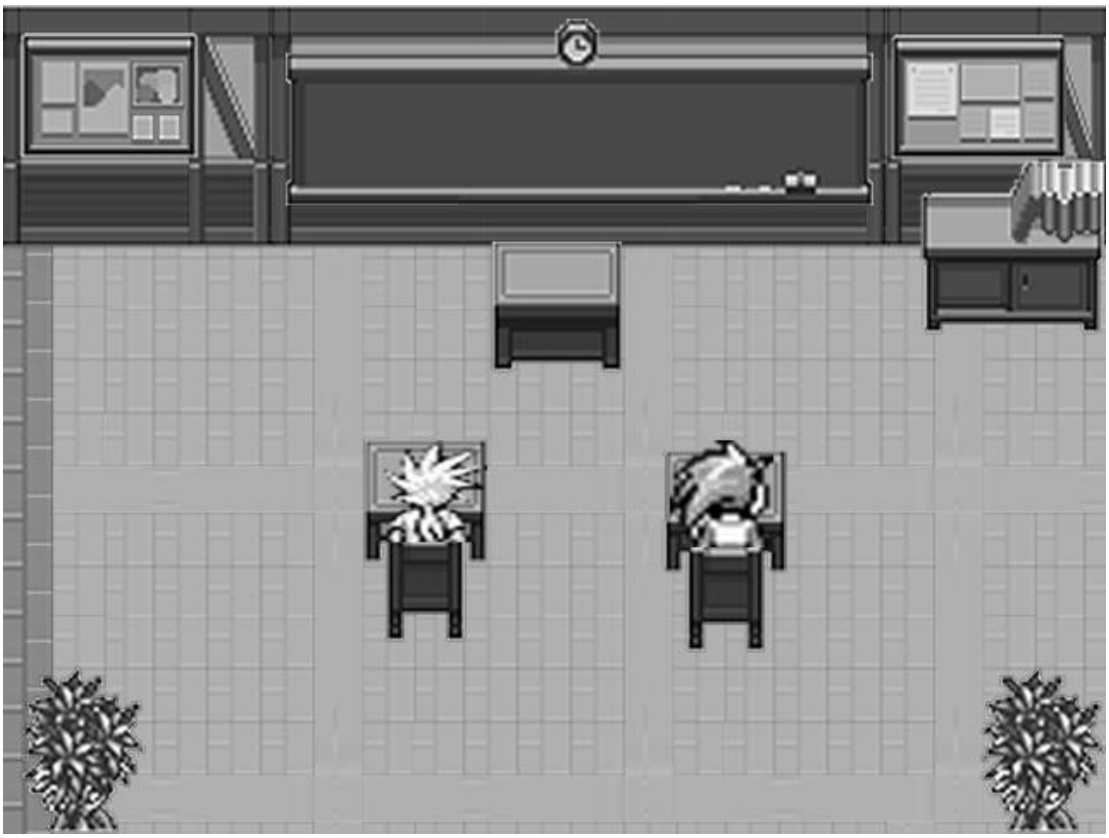
3. *Storyboard stage* 3



Gambar 3.20 *Storyboard Stage* 3

Pada *Stage 3*, Pemain akan berlari dan menghadapi rintangan. Jika menemui rintangan pemain akan mendapatkan pertanyaan, jika pertanyaan tersebut benar maka pemain akan lolos dari rintangan tersebut jika salah langsung *Game Over.*

4. *Storyboard stage* 4

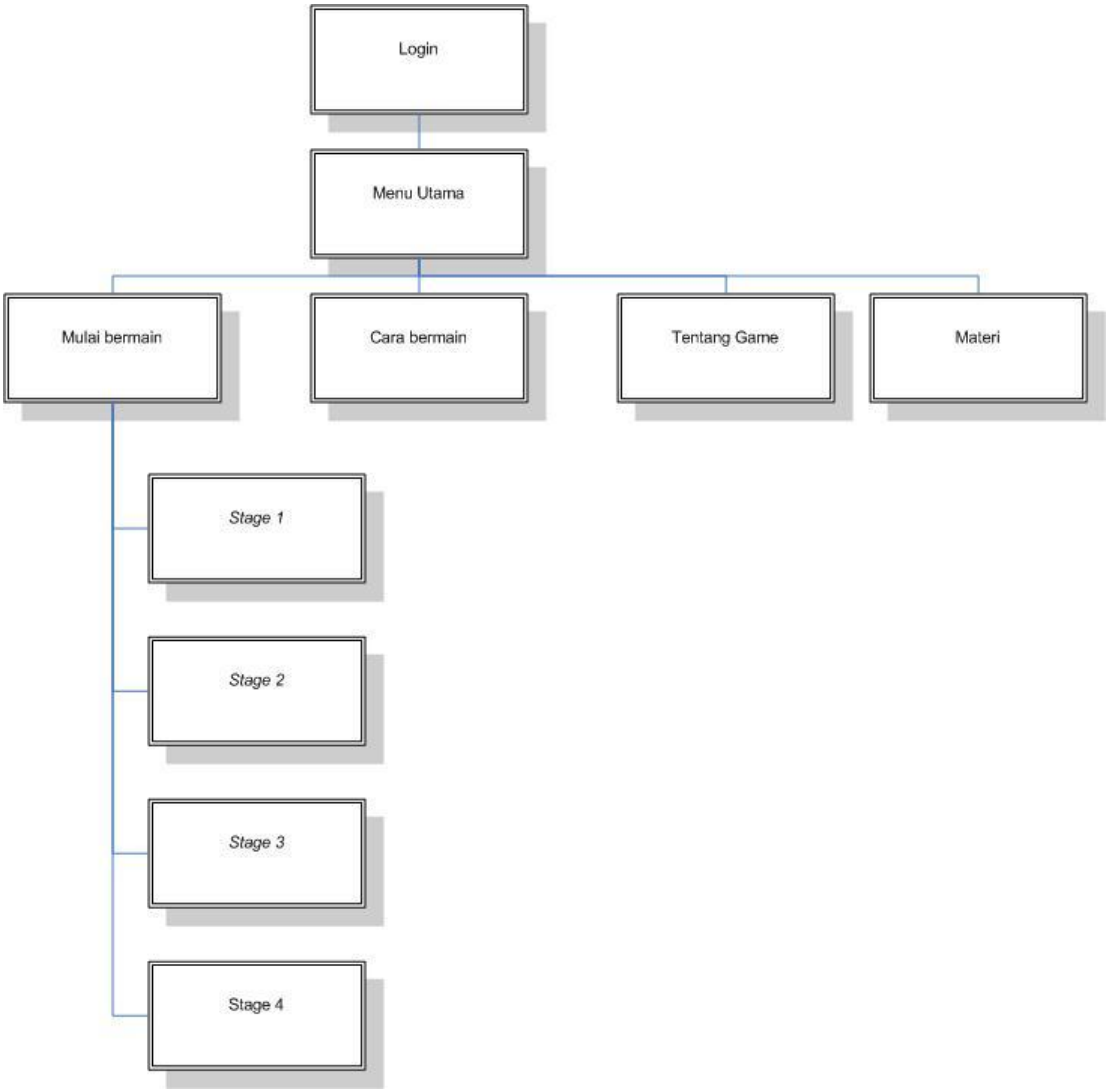


Gambar 3.21 *Storyboard Stage* 4

Pada *Stage* ini, pemain harus menjawab 10 pertanyaan dan beradu cepat dengan musuh. Jika pemain tidak bisa menjawab maka darah pemain akan berkurang jika menjawab dengan benar maka darah musuh akan berkurang.

3.5.2 Perancangan Struktur Menu

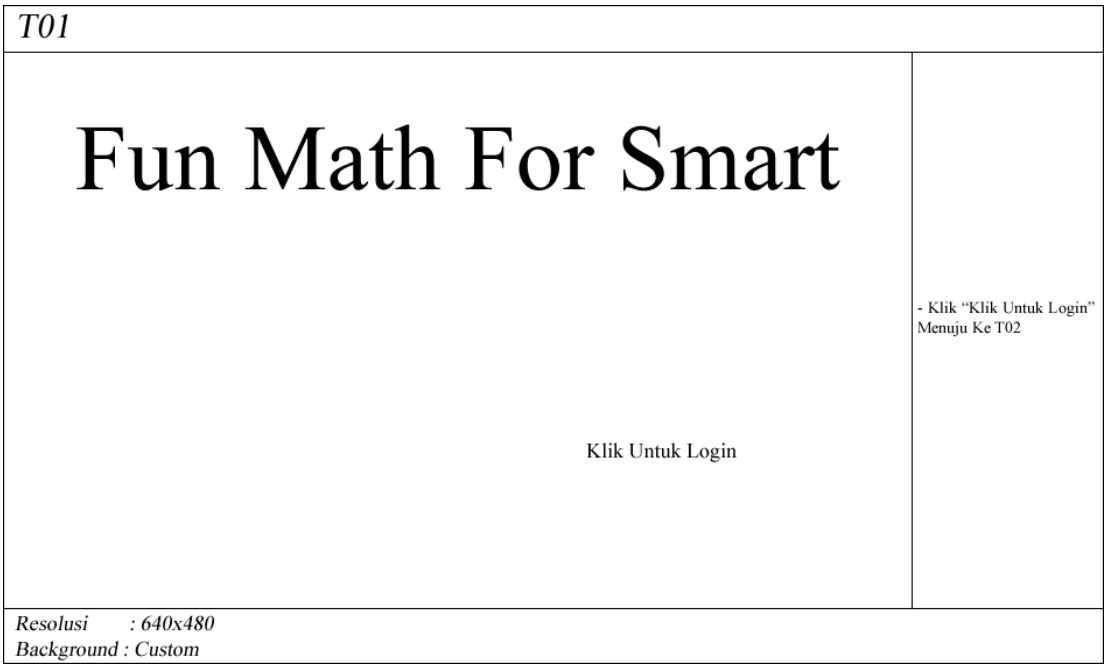
Untuk mempermudah dalam pembuatan sistem, dibutuhkan perancangan struktur menu dari program uang akan di bangun. Perancangan struktur menu pada program ini membantu dalam merancang bagian-bagian dari sistem yang sebenarnya dan untuk mengetahui bahwa bagian mana yang akan di akses terlebih dahulu setelah program selesai dibuat.



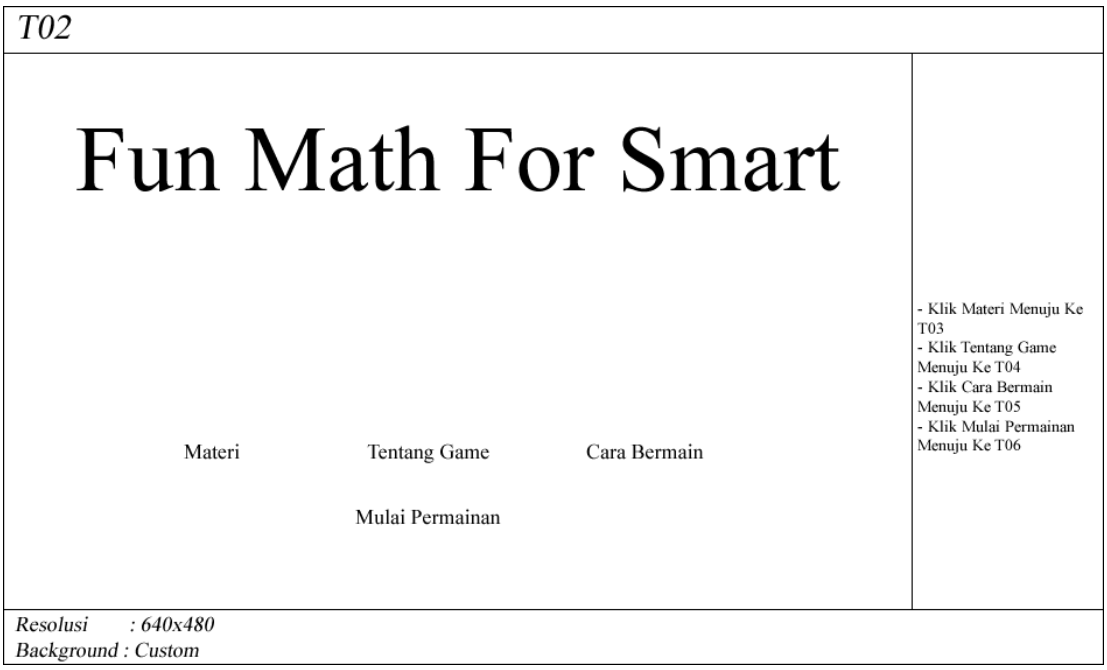
Gambar 3.22 Struktur Menu

3.5.3 Perancangan Antarmuka

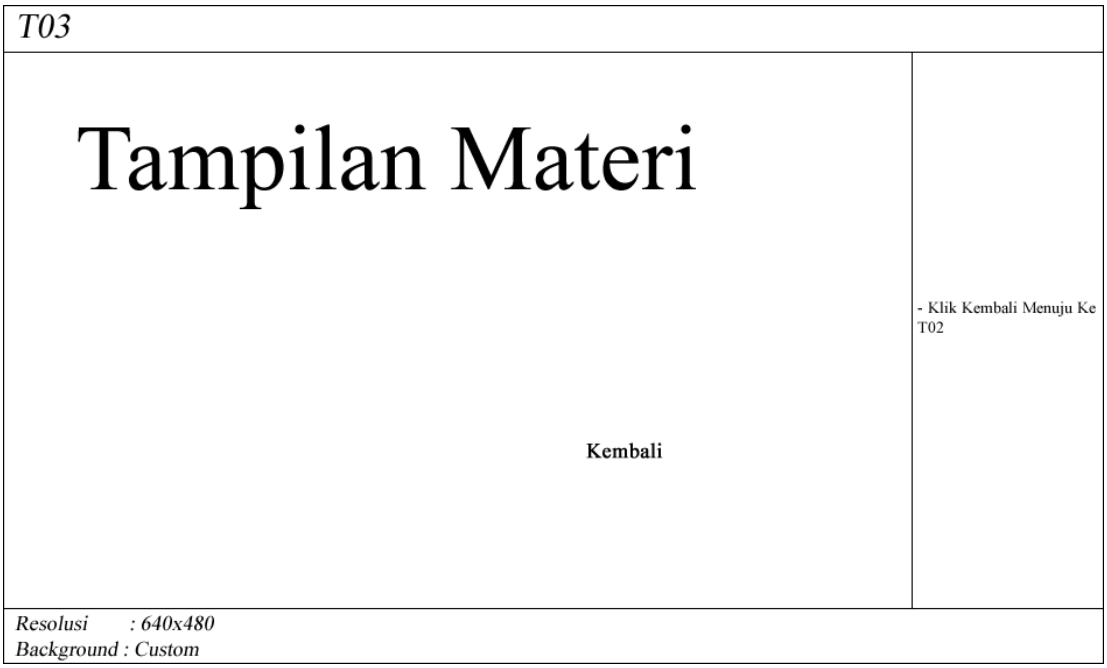
Perancangan antarmuka dilakukan untuk mempermudah dalam mengimplementasikan sistem yang akan dibangun.



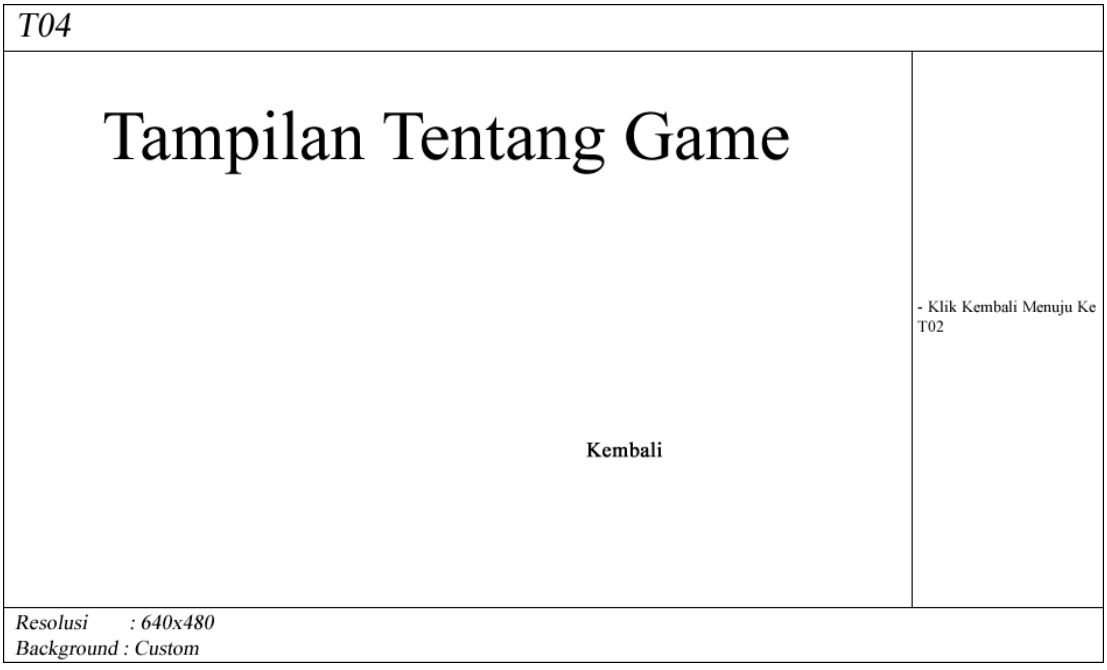
Gambar 3.23 Perancangan Antarmuka Login



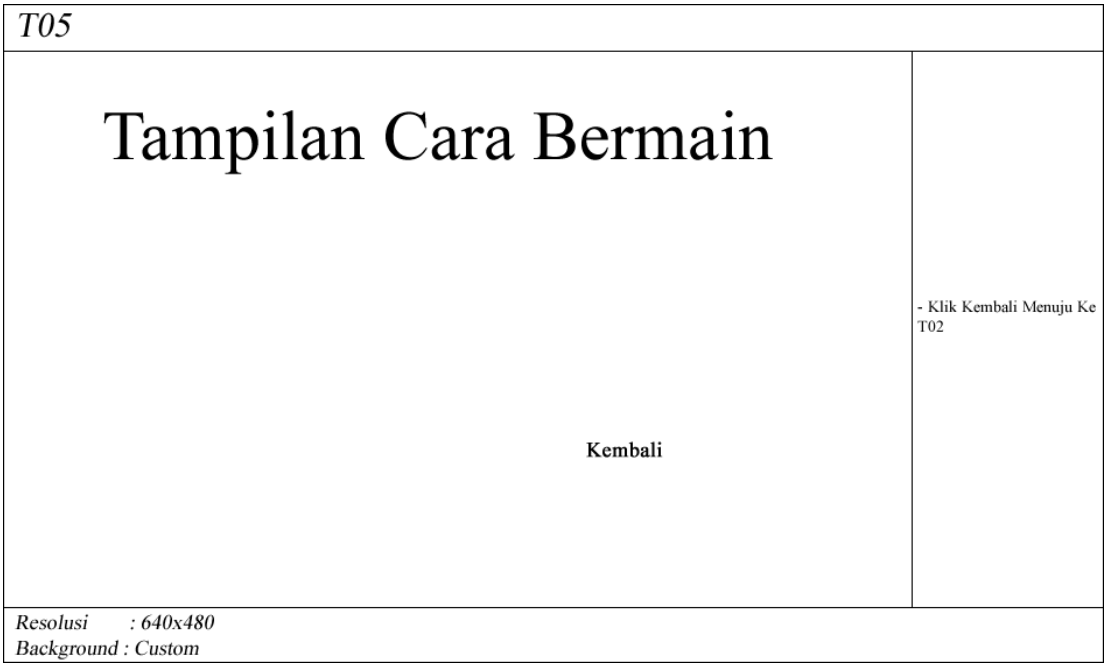
Gambar 3.24 Perancangan Antarmuka Menu



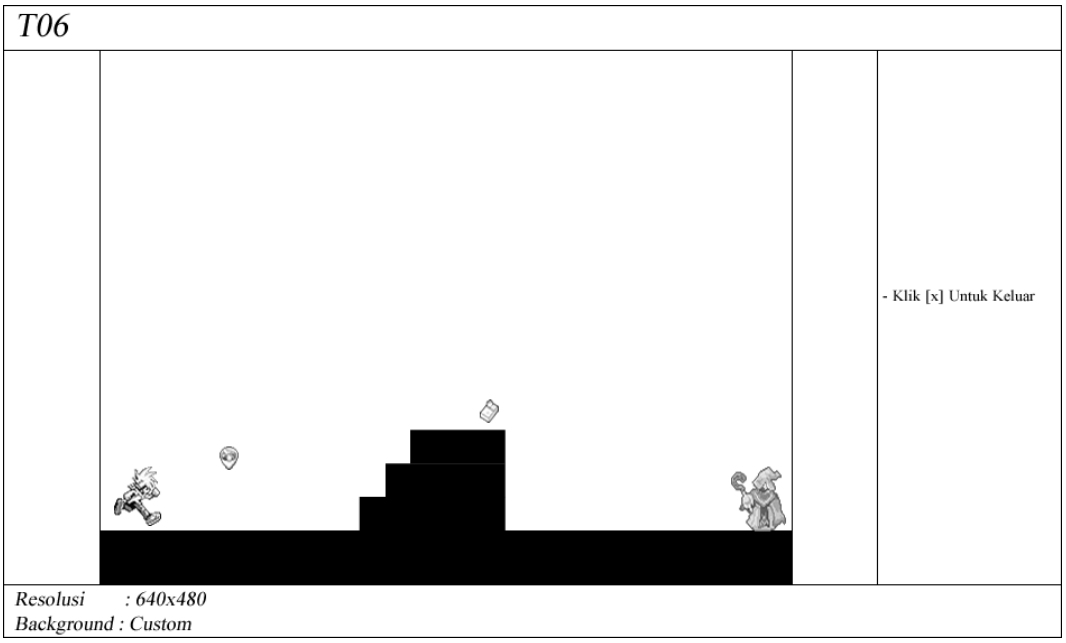
Gambar 3.25 Perancangan Antarmuka Materi



Gambar 3.26 Perancangan Antarmuka Tentang Game

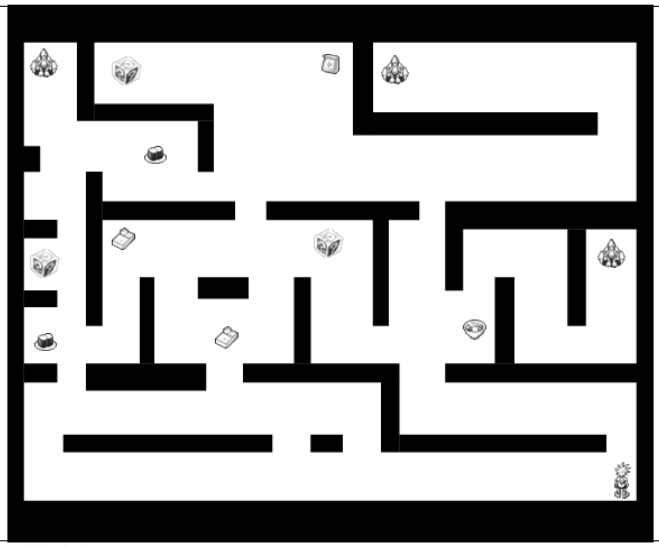


Gambar 3.27 Perancangan Antarmuka Cara Bermain



Gambar 3.28 Perancangan Antarmuka Permainan Stage 1

T07

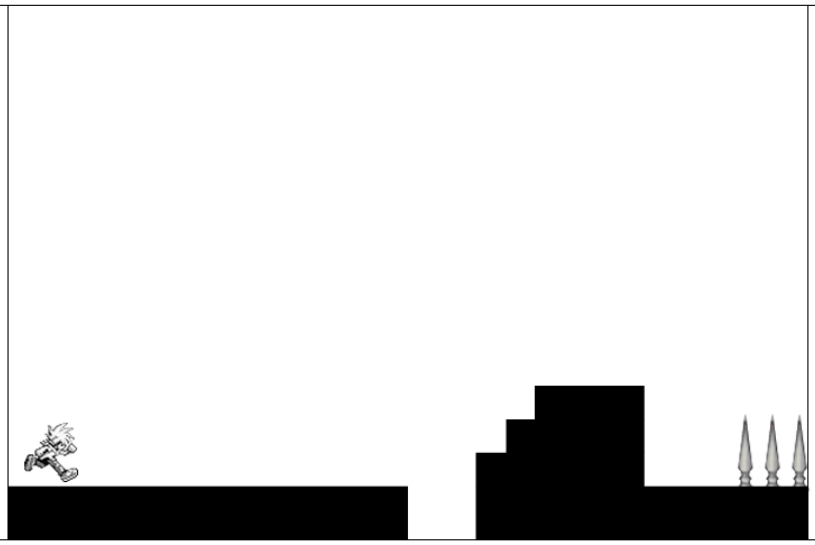


- Klik [x] Untuk Keluar

Resolusi : 640x480
Background : Custom

Gambar 3.29 Perancangan Antarmuka Permainan Stage 2

T08

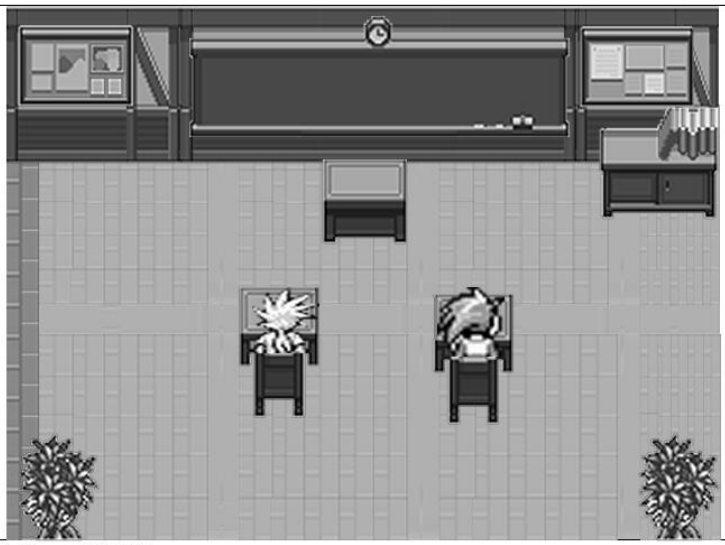


- Klik [x] Untuk Keluar

Resolusi : 640x480
Background : Custom

Gambar 3.30 Perancangan Antarmuka Permainan Stage 3

T09



- Klik [x] Untuk Keluar

Resolusi : 640x480
Background : Custom

Gambar 3.31 Perancangan Antarmuka Permainan Stage 4

T10



Game Over

Lanjut ? Main Lagi

10

Score Terakhir

500

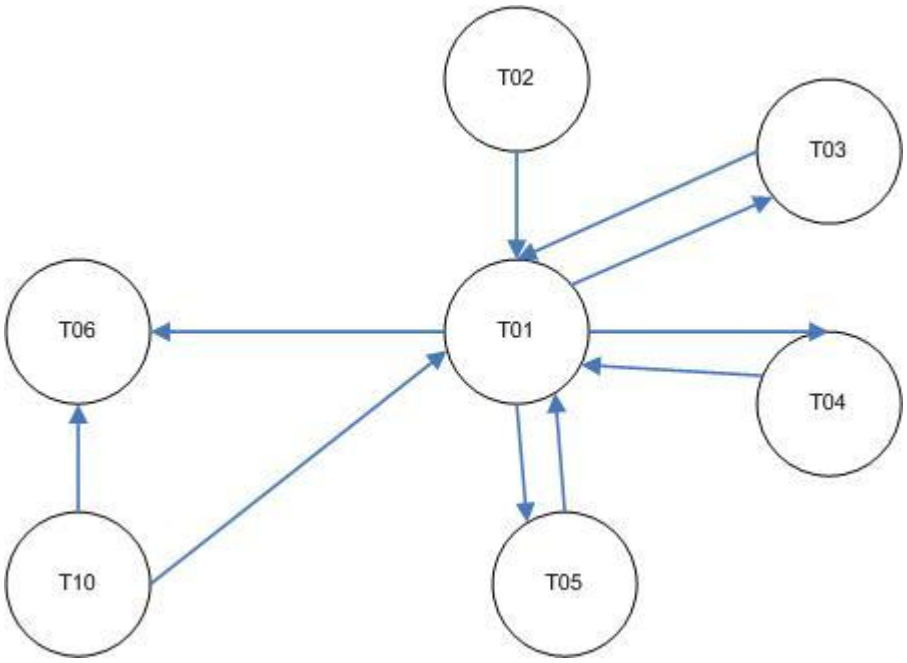
- Klik Menu Utama Menuju Ke T02
- Klik kembali Menuju Ke T06

Resolusi : 640x480
Background : Custom

Gambar 3.32 Perancangan Antarmuka Game Over

3.5.4 Jaringan Semantik

Jaringan semantik merupakan jaringan data yang menunjukan hubungan antar berbagai objek.



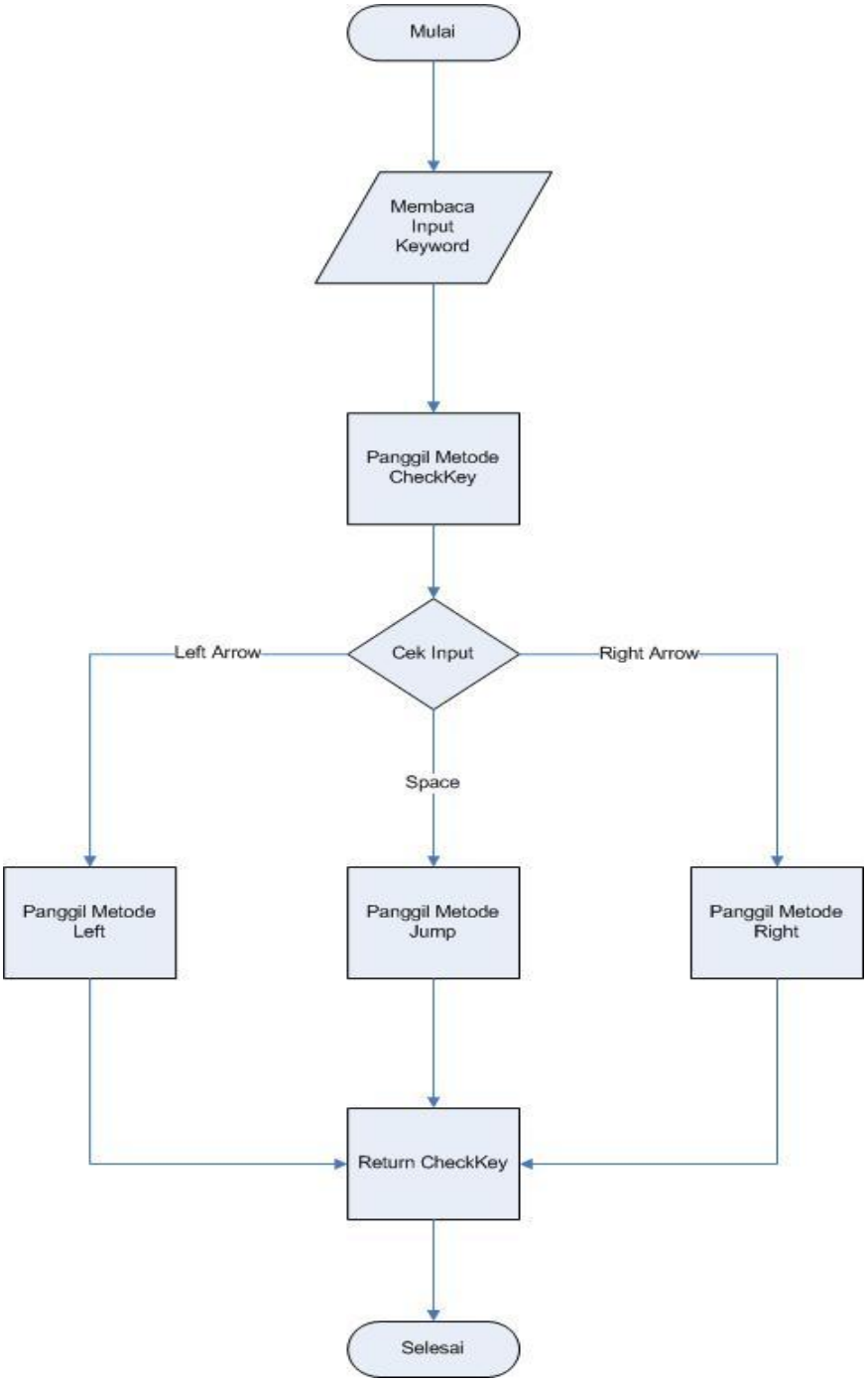
Gambar 3.33 Jaringan Semantik

3.5.5 Perancangan Method

Perancangan komponen method merupakan perancangan yang dibuat setelah perancangan struktur menu, antarmuka dan jaringan semantik. Perancangan ini berfungsi untuk mendeskripsikan method-method yang berada di dalam aplikasi permainan. Adapun method-method yang terdapat dalam aplikasi permainan *Fun Math For Smart* yang akan dibangun adalah sebagai berikut :

1. **Perancangan Method *Check Key Stage 1***

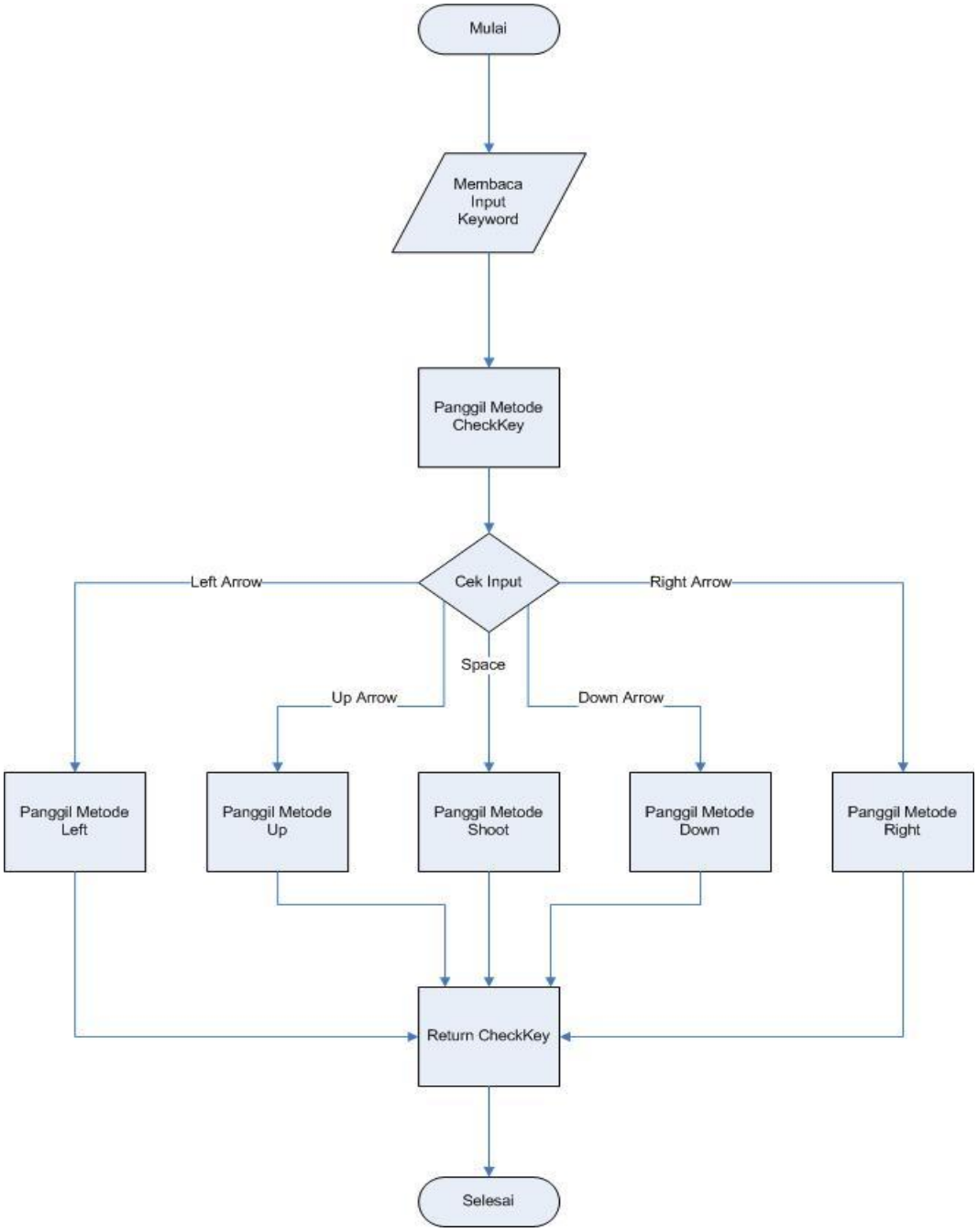
Metode ini berfungsi untuk penanganan pergerakan pada *player* pada *stage 1*.



Gambar 3.34 Perancangan Method *Check Key Stage 1*

2. Perancangan Method *Check Key Stage 2*

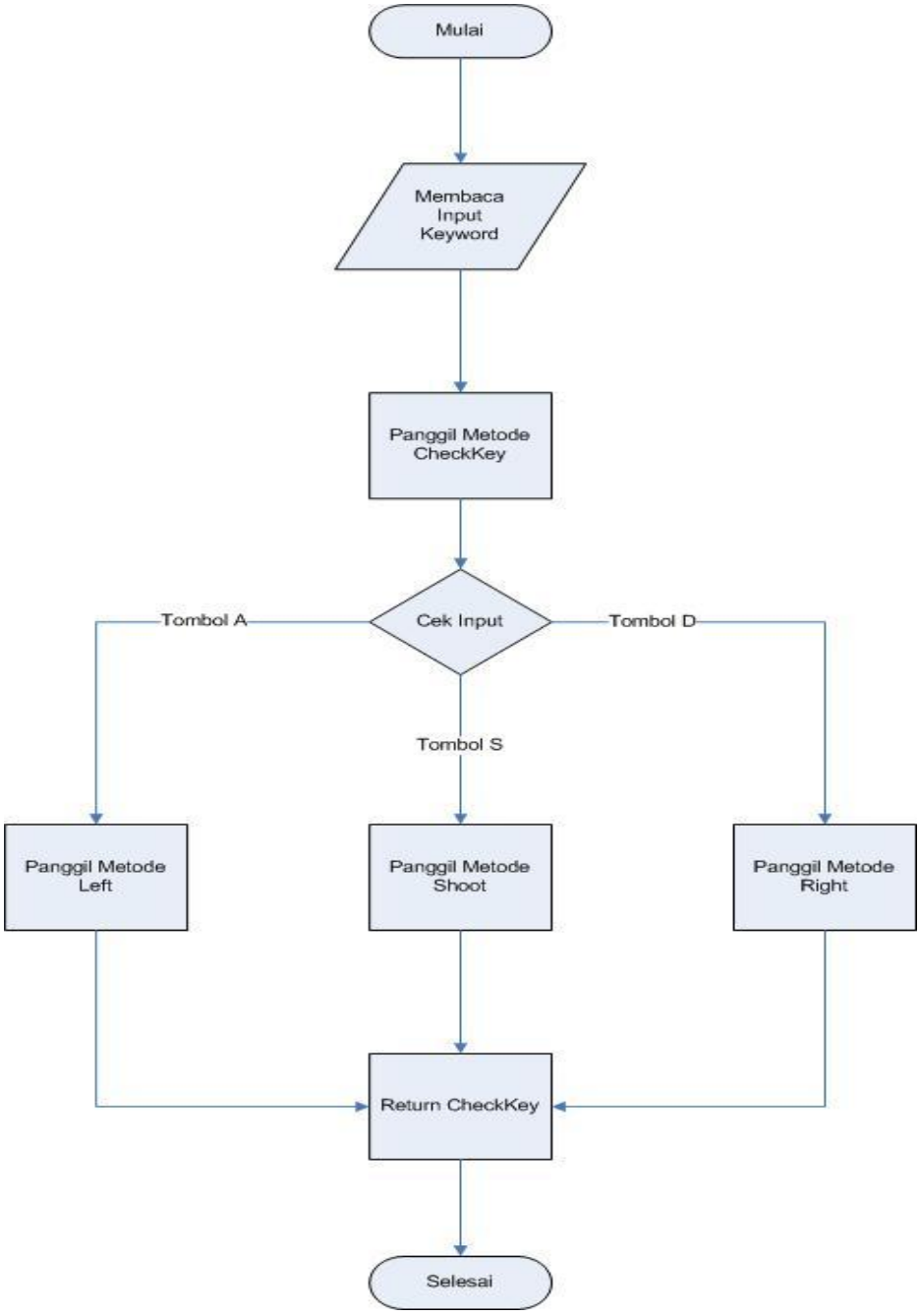
Metode ini berfungsi untuk penanganan pergerakan pada *player* pada *stage 2*.



Gambar 3.35 Perancangan Method *Check Key Stage 2*

3. Perancangan Method *Check Key* Jawab Pertanyaan

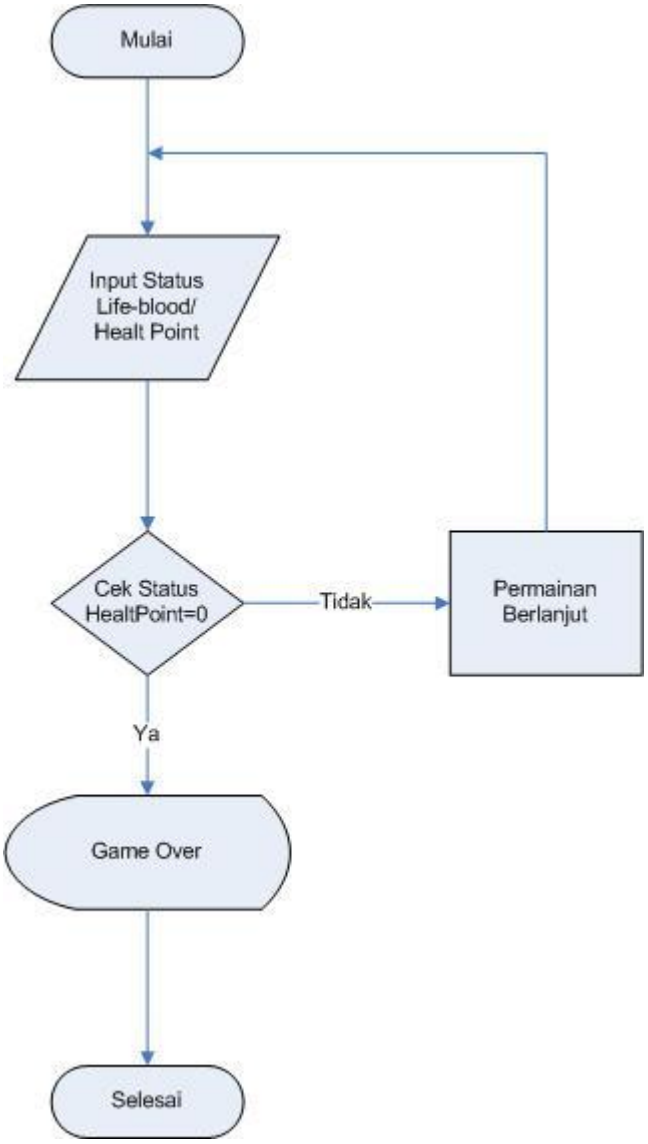
Metode ini berfungsi untuk penanganan pergerakan pada *player* saat menjawab pertanyaan.



Gambar 3.36 Perancangan Method *Check Key* Menjawab Pertanyaan

4. Perancangan Method Nyawa / *Healt Point*

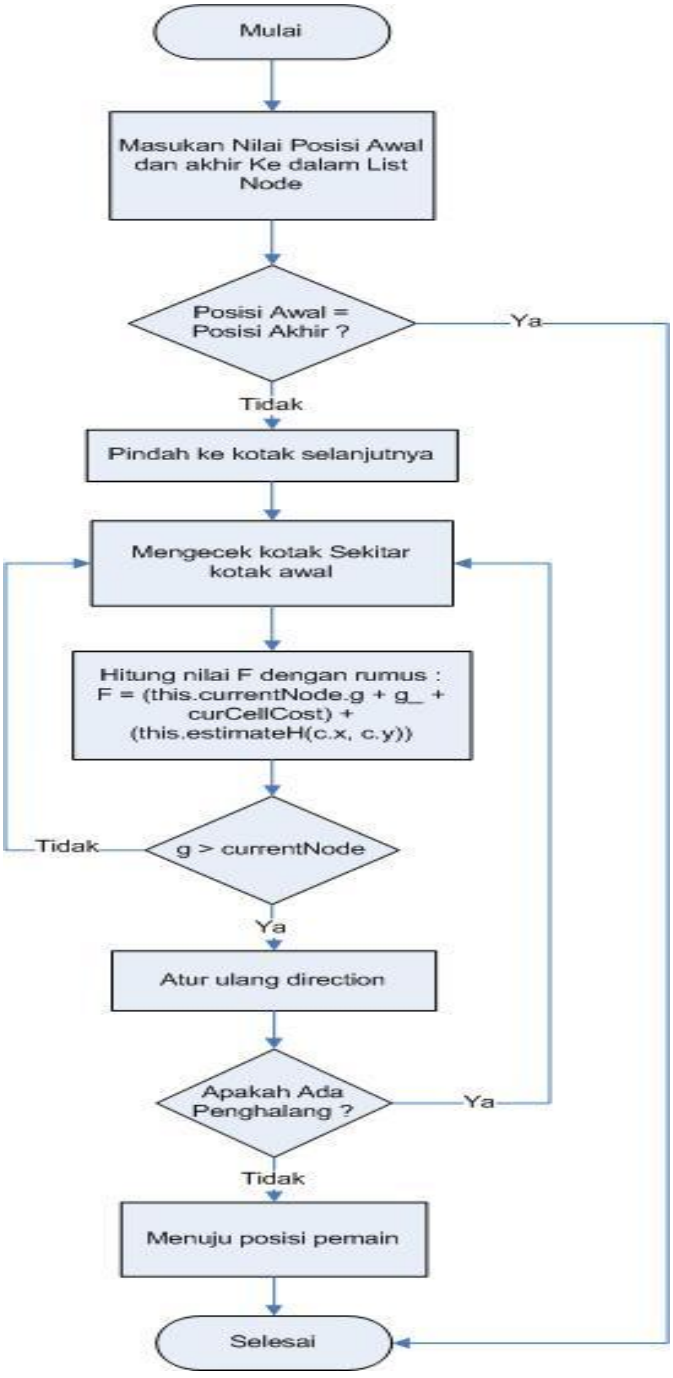
Metode ini berfungsi untuk mengetahui *healt point Hero*



Gambar 3.37 Perancangan Method Nyawa / *Healt Point*

5. Perancangan Method Pathfind

Metode ini berfungsi untuk Musuh 2 menentukan rute terpendek.



Gambar 3.38 Perancangan Method Pathfind