BAB 3
ANALISIS DAN PERANCANGAN SISTEM

3.1 Analisis Sistem

Analisis sistem dapat didefinisikan sebagai penguraian dari suatu sistem yang utuh kedalam bagian-bagian komponennya dengan maksud untuk mengidentifikasi dan mengevaluasi kebutuhan-kebutuhan yang diharapkan sehingga dapat diusulkan perbaikan-perbaikannya.

3.1.1 Analisis Masalah

Berdasarkan analisis sistem yang dilakukan dengan mengevaluasi terhadap pengetahuan generasi muda tentang keanekaragaman budaya dan cerita rakyat Indonesia maka masalah yang dihadapi oleh generasi muda khususnya anak-anak yang usianya 10 tahun ke atas dalam melakukan pengenalan tentang kebudayaan adalah sebagai berikut :

1. Pentingnya pengetahuan tentang keanekaragaman budaya Indonesia khususnya di 5 pulau besar Indonesia, karena dengan mengenal dan mengetahui keanekaragaman budaya akan menambah wawasan dan melestarikan kebudayaan di Indonesia.
2. Perlunya memperkenalkan kembali cerita rakyat Sangkuriang kepada anak-anak Indonesia, karena cerita rakyat yang menceritakan asal-usul Gunung Tangkuban Perahu tersebut hampir terlupakan sebagai warisan budaya Indonesia **[3]**.
3. Pengenalan melalui media cetak kurang optimal untuk mengenalkan keanekaragaman budaya dan cerita rakyat Sangkuriang **[2]**, sehingga dibutuhkan sebuah aplikasi *game* agar pengenalan terlihat menarik.

3.1.2 Analisis *Game* Sejenis yang Sudah Ada

Sebelum merancang *game* yang akan dibuat, terlebih dahulu dilakukan analisis pada *game* sejenis yang sudah ada. Ini bertujuan agar *game* yang sudah dibuat nanti memiliki solusi dari permasalahan *game* yang sudah ada.

Game yang memiliki unsur budaya pada *platform mobile* dengan sistem operasi android sudah ada beberapa, diantaranya yaitu *Temple Rush : The Legend of Prambanan Temple* dan *The Adventure of Wanara*. Berikut ini deskripsi dari *game-game* tersebut.

3.1.2.1 Pengenalan Temple Rush : The Legend of Prambanan Temple

Temple Rush adalah *game* untuk android yang mengambil tema berdasarkan pada cerita rakyat Indonesia. *Game* ini bergenre casual *game* dengan permainan yang sangat sederhana, yaitu pemain harus mengendalikan beberapa karakter pemain hingga tujuan atau goal dari *game* ini tercapai. Tujuan atau goal dari *game* ini adalah membangun candi hingga selesai.

1. Storyline

Seorang kesatria yang memiliki nama Bandung Bondowoso yang hendak menikahi seorang putri yangi bernama Roro Jonggrang. Roro Jonggrang ternyata tidak menyukai Bandung Bondowoso dan hendak menolak lamaran sang kesatria Bandung Bondowoso tersebut. Roro Jonggrang tidak ingin menyakiti hati Bandung Bondowoso dan akhirnya dia menolak dengan cara yang halus, dia meminta agar dibuatkan 1000 (seribu) candi dalam kurun waktu 1 (satu) malam. Roro Jonggrang berpikir Bandung tidak akan bisa memenuhi permintaannya tersebut, tanpa di kira ternyata Bandung Bondowoso menyanggupi permintaan Roro Jonggrang tersebut. Akhirnya pada suatu malam yang telah ditentukan Bandung Bondowoso memulai membuat candi tersebut dengan bantuan Jin. Permainan pun dimulai dengan tujuan untuk menyelesaikan 1000 candi. Berikut ini tampilan dari *game* Temple Rush.



Gambar 3.1. Tampilan *Game Temple Rush*

2. *Gameplay*

Tugas utama pemain adalah mengendalikan beberapa karakter jin pembawa batu candi mengikuti jalan cerita yang ada hingga tujuan atau goal dari *game* tercapai, yaitu membangun candi dengan mencocokkan warna karakter jin pembawa batu candi dengan warna roh api sebagai petunjuk candi yang akan dibangun dengan waktu yang telah ditentukan. Beberapa karakter jin dikendalikan dalam *game world* dengan menggunakan *touchscreen*. Selain terdapat karakter jin pembawa batu candi, ada pula jin penambah waktu agar pemain dapat menambahkan waktu permainan dalam hal ini adalah waktu hingga matahari terbit yang dapat menyebabkan pemain kalah. Perolehan score berdasarkan candi yang berhasil dibangun.

Secara umum, *game Temple Rush* dapat dikategorikan sebagai permainan yang sangat sederhana dan mudah dimainkan. Namun memiliki *gameplay* yang monoton sehingga pemain cepat merasa bosan.

3. Scoring

Cara mendapatkan skor berdasarkan dari berapa banyak karakter pemain dapat mencocokkan warna antara karakter jin pembawa batu candi dan roh api.

3.1.2.2 Pengenalan Tetuko

Tetuko adalah sebuah *game* Android yang pemeran utamanya adalah Tetuko, nama dari Gatot Kaca muda. Tantangan dari *game* ini adalah menggerakkan Tetuko agar bisa selamat dari berbagai serangan dari arah-arah yang berlainan. *Game* Tetuko bergenre adventure karena pemain harus menghadapi rintangan-rintangan yang ada.

1. Storyline

Tetuko merupakan Gatot Kaca muda dari cerita wayang, ksatria yang gagah dan sangat sakti. Sehingga dia harus waspada dalam menjalani hidupnya karena adanya tantangan-tantangan atau musuh yang menyerang.



Gambar 3.2. Tampilan *Game* Tetuko

2. *Gameplay*

Tugas utama pemain adalah menggerakkan karakter pemain agar bisa selamat dari berbagai serangan dari arah-arah yang berlainan, seperti serangan batu dan panah. Mengontrol perpindahan karakter pemain dengan menekan dua kontrol di kiri dan kanan. Tetuko memiliki user interface yang mudah, grafik yang simpel tapi sarat makna, serta level permainan yang tak terbatas. Secara umum,

game ini dapat dikategorikan sebagai permainan yang sangat sederhana. Namun cukup sulit memainkannya karena butuh refleks yang tinggi dan pemain merasa cepat bosan karena serangan dan cara bertahan yang sama.

3. Scoring

Cara mendapatkan skor berdasarkan dari berapa banyak karakter pemain dapat menghindari serangan dari arah-arah yang berlawanan, dengan skor 10 point setiap selamat dari serangan.

3.1.3 Analisis *game* yang akan dibangun

Game yang akan dibangun merupakan *game* yang berjenis *adventure*, dimana karakter utama berpetualang mengeliling 5 pulau besar di Indonesia. Karakter pada *game* ini diambil dari salah satu cerita rakyat atau legenda Jawa Barat yaitu Sangkuriang yang menceritakan asal-mula Gunung Tangkuban Parahu

Fitur – fitur yang akan dibuat pada *game* ini adalah:

1. Sistem *single player.*
2. Karakter utama yang diambil dari cerita rakyat Jawa Barat.
3. Karakter utama memiliki kemampuan untuk melawan rintangan untuk mendapatkan misi.
4. Memiliki 5 (lima) *level* dengan tingkat kesulitan yang berbeda-beda, dimana lokasi setiap *level* merupakan lima pulau besar di Indonesia (Pulau Sumatera, Kalimantan, Sulawesi, Papua dan Jawa).
5. Mengangkat tema tentang keanekaragaman budaya Indonesia dan legenda Sangkuriang.
6. *Control* karakter dengan menggunakan *touchscreen.*
7. *Grafik* yang digunakan merupakan *grafik game* 2D.

3.1.3.1 *Storyline*

Berawal dari kisah seorang puteri raja di Jawa Barat bernama Dayang Sumbi. Puteri raja tersebut membuat sumpah “Barang siapa yang menemukan

gulungan benang jika laki-laki akan dijadikan suami", dan yang menemukannya adalah Tumang, anjing kesayangannya. Ternyata Tumang adalah titisan dewa yang berubah wujud menjadi seekor anjing. Akhirnya mereka menikah dan mempunyai seorang anak laki-laki yang diberi nama Sangkuriang. Anak tersebut sangat gemar berburu. Ia berburu dengan ditemani oleh Tumang. Sangkuriang tidak tahu, bahwa anjing itu adalah titisan dewa dan juga bapaknya. Pada suatu hari Tumang tidak mau mengikuti perintahnya untuk mengejar hewan buruan. Maka anjing tersebut disembelih dan diambil hatinya. Ketika kembali ke rumah, Sangkuriang menceritakan kejadian itu pada ibunya. Bukan main marahnya Dayang Sumbi begitu mendengar cerita itu. Tanpa sengaja ia memukul kepala Sangkuriang dengan sendok nasi yang dipegangnya. Sangkuriang terluka. Ia sangat kecewa dan pergi mengembara.

Petualangan ini dimulai ketika Sangkuriang diusir oleh Dayang Sumbi (ibunya), karena telah membunuh si Tumang yang ternyata adalah ayahnya sendiri. Semenjak saat itu Sangkuriang mengembara dengan mengelilingi lima pulau besar yang ada di Indonesia dimulai dari Pulau Sumatera sampai nanti akhirnya pulang kembali ke Pulau Jawa.

1. *Level* 1 (Pulau Sumatera)

Intro:

Sangkuriang pergi ke Pulau Sumatera dengan tujuan untuk menguasai ilmu silek. Sebelum bertemu ahli silek, dia harus menghadapi rintangan selama perjalanan, yaitu harus mengalahkan seluruh harimau buas dan burung Endemik yang ada di hutan. Sampai akhirnya harus meghadapi ahli silek (kepala suku Minangkabau). Berikut ini unsur-unsur yang terdapat pada *level* 1, dapat dilihat pada Tabel 3.1.

Tabel 3.1. Unsur-unsur pada *level* 1

Setting	hutan, depan rumah adat Padang (rumah gadang)
Misi	menguasai ilmu silek dan mendapatkan kerambit Minang (senjata tradisional suku Minangkabau)

Rintangan	harimau Sumatera, burung Endemik dan ahli silek (kepala suku Minangkabau)
Apa yang harus dilakukan	mengambil makanan diperjalanan, mengalahkan musuh dan mengalahkan ahli silek Minang kemudian mengambil kerambit yang dimilikinya

Outro :

Setelah Sangkuriang berhasil mengalahkan ahli silek Minang, ilmu bela diri yang dimilikinya menjadi sempurna. Kemudian Sangkuriang melanjutkan perjalanan ke pulau Kalimantan.

2. *Level* 2 (Pulau Kalimantan)

Intro :

Setelah mengalahkan ahli silek di Pulau Sumatera, Sangkuriang melanjutkan perjalanan ke Kalimantan. Di Pulau Kalimantan, Sangkuriang harus menghadapi binatang buas yang berbeda sebelum bertemu dengan kepala suku Dayak untuk mendapatkan misi. Sampai akhirnya harus meghadapi kepala suku Dayak. Berikut ini unsur-unsur yang terdapat pada *level* 2, dapat dilihat pada Tabel 3.2.

Tabel 3.2. Unsur-unsur pada *level* 2

Setting	hutan berawa, depan rumah adat Banjar
Misi	mendapatkan mandau dan talawang
Rintangan	Buaya dan kepala suku Dayak
Apa yang harus dilakukan	mengambil makanan diperjalanan, mengalahkan musuh dan mengalahkan kepala suku Dayak kemudian mengambil mandau dan talawang

Outro :

Setelah berhasil mengalahkan kepala suku Dayak, Sangkuriang mendapatkan pusaka mandau dan talawang. Kemudian Sangkuriang melanjutkan perjalanan ke pulau Sulawesi .

3. *Level* 3 (Pulau Sulawesi)

Intro:

Sangkuriang pergi ke Pulau Sulawesi, disana dia harus melewati rintangan yang berbeda. Berikut ini unsur-unsur yang terdapat pada *level* 3, dapat dilihat pada Tabel 3.3.

Tabel 3.3. Unsur-unsur pada *level* 3

Setting	Di laut
Misi	mendapatkan senjata tradisional kawali dari Kepala Suku Bugis
Rintangan	pusaran air, terumbu karang, ikan paus
Apa yang harus dilakukan	mengambil makanan dan menghindari rintangan.

Outro :

Setelah Sangkuriang berhasil mendapatkan mendapatkan senjata tradisional kawali dari nahkoda (Kepala Suku Bugis). Sangkuriang melanjutkan perjalanan ke pulau Papua.

4. *Level* 4 (Pulau Papua)

Intro:

Sangkuriang pergi ke Pulau Papua. Berikut ini unsur-unsur yang terdapat pada *level* 4, dapat dilihat pada Tabel 3.4.

Tabel 3.4. Unsur-unsur pada *level* 4

Setting	hutan belantara
Misi	menguasai ilmu memanah dan mendapatkan busur dan panah
Rintangan	Kelelawar dan burung-burungan
Apa yang harus dilakukan	mengambil makanan diperjalanan, memanah musuh dan busur dan panah menjadi milik Sangkuriang.

Outro :

Setelah Sangkuriang berhasil, Sangkuriang pulang ke Pulau Jawa tepatnya ke tanah Sunda.

5. *Level* 5 (Pulau Jawa)

Intro:

Sebelum sampai di kampung halamannya, sangkuriang bertemu dengan siluman harimau yang menghalangi perjalanannya. Sangkuriang harus mengalahkan siluman tersebut agar dapat pulang. Berikut ini unsur-unsur yang terdapat pada *level* 5, dapat dilihat pada Tabel 3.5.

Tabel 3.5. Unsur-unsur pada *level* 5

Setting	gua, depan rumat adat kasepuhan
Misi	mengalahkan semua siluman, kemudian menjadikan jin-jin tersebut sebagai anak buahnya dan sangkuriang mendapatkan kujang.
Rintangan	siluman harimau, siluman kelelawar dan raja jin Sunda
Apa yang harus dilakukan	mengambil makanan diperjalanan, mengalahkan musuh dan menjadikan semua siluman sebagai anak buahnya, serta sangkuriang mendapatkan kujang

Jika berhasil menjalani semua misi. Sangkuriang kembali ke tanah Sunda, kemudian disana Sangkuriang dan Dayang Sumbi bertemu kembali dan saling mencintai. Namun ketika mereka akan menikah, Dayang Sumbi mengetahui bahwa Sangkuriang adalah anak kandungnya. Maka Dayang Sumbi mengajukan persyaratan yang sulit kepada Sangkuriang agar pernikahan terlarang itu dapat dibatalkan. Persyaratan tersebut diantaranya Dayang Sumbi ingin dibuatkan danau dan sebuah perahu dalam waktu satu malam. Sangkuriang menyanggupinya dengan mengerahkan seluruh kesaktian yang dimilikinya dan dibantu oleh anak buahnya (para siluman). Dayang Sumbi berusaha untuk menggagalkan pekerjaan Sangkuriang dan berdoa kepada dewa agar menerbitkan matahari. Sadar Sangkuriang gagal, dia naik pitam lalu menendang perahu yang hampir selesai dibuatnya. Perahu tersebut jatuh dengan posisi telungkup dan menjadi Gunung Tangkuban Perahu.

3.1.3.2 *Gameplay*

Pada permainan ini, tugas utama pemain adalah *actor* akan melakukan perjalanan untuk mendapat makanan dan menghadapi musuh, terdapat 5 *level* . Dalam setiap perjalanan akan terdapat beberapa musuh yang harus di kalahkan agar mencapai misi utama (memperoleh pusaka atau senjata tradisional) dan menemukan beberapa makanan untuk mendapatkan *score.*

1. *Level* 1
    a. Terdapat beberapa makanan yang terletak di tempat-tempat tertentu secara acak.
    b. Terdapat beberapa musuh pada *level* ini, yaitu harimau Sumatera dan burung Endemik. Terdapat musuh utama yang harus dikalahkan yaitu kepala Suku Minangkabau (ahli silek).
    c. Terdapat pusaka (senjata tradisional kerambit) yang harus diambil.
    d. Pemain hanya dapat memukul musuh yang dihadapinya.
    e. Permainan akan berakhir apabila karakter telah mendapatkan misi dan pergi ke misi berikutnya (*level* 2) atau karakter kalah menghadapi musuh.

2. *Level* 2
    a. Terdapat beberapa makanan yang terletak di tempat-tempat tertentu secara acak.
    b. Terdapat beberapa musuh pada *level* ini, yaitu buaya.
    c. Terdapat musuh utama yang harus dikalahkan yaitu Kepala Suku Dayak.
    d. Terdapat pusaka (senjata tradisional mandau dan talawang) yang harus diambil.
    e. Pemain hanya dapat memukul musuh yang dihadapinya.
    f. Permainan akan berakhir apabila karakter telah mendapatkan misi dan pergi ke misi berikutnya (*level* 3) atau karakter kalah menghadapi musuh.

3. *Level* 3
    a. Terdapat beberapa makanan yang terletak di tempat-tempat tertentu secara acak.
    b. Terdapat beberapa musuh pada *level* ini, yaitu pusaran air, terumbu karang, dan paus.
    c. Terdapat pusaka (senjata tradisional kawali atau badik bugis) yang harus diambil.
    d. Pemain hanya dapat memukul musuh yang dihadapinya.
    e. Permainan akan berakhir apabila karakter telah mendapatkan misi dan pergi ke misi berikutnya (*level* 4) atau karakter kalah menghadapi musuh.

4. *Level* 4
    a. Terdapat beberapa makanan yang terletak di tempat-tempat tertentu secara acak.
    b. Terdapat beberapa musuh pada *level* ini, yaitu kelelawar dan burung-burungan.
    c. Terdapat pusaka (senjata panah dan busur) yang harus diambil.
    d. Pemain hanya dapat memukul musuh yang dihadapinya.
    e. Permainan akan berakhir apabila karakter telah mendapatkan misi dan pergi ke misi berikutnya (*level* 5) atau karakter kalah menghadapi musuh.

5. *Level* 5
    a. Terdapat beberapa makanan yang terletak di tempat-tempat tertentu secara acak.
    b. Terdapat beberapa musuh pada *level* ini, yaitu siluman harimau.
    c. Terdapat pusaka (senjata tradisional golok) yang harus diambil.
    d. Pemain hanya dapat memukul musuh yang dihadapinya.
    e. Permainan akan berakhir apabila karakter telah mendapatkan misi karakter kalah menghadapi musuh

3.1.3.3 Scoring

Pada bagian ini akan dijelaskan bagaimana pemberian skor yang ada pada *game*. Berikut cara memperoleh nilai atau *score* dari *game adventure* "Sangkuriang *and Indonesian five largest islands*" yaitu :

1. Jika berhasil mengambil makanan akan mendapatkan *score* 1.
2. Jika berhasil mengambil semua makanan yang ada tanpa ada yang terlewatkan, maka *score* dikalikan 2 dari *score* seharusnya.
3. Jika berhasil mengalahkan musuh, maka memperoleh *score* 20.
4. Jika berhasil menyelesaikan *level* 1 akan mendapatkan *score* 1000.
5. Jika berhasil menyelesaikan *level* 2 akan mendapatkan *score* 2000.
6. Jika berhasil menyelesaikan *level* 3 akan mendapatkan *score* 3000.
7. Jika berhasil menyelesaikan *level* 4 akan mendapatkan *score* 4000.
8. Jika berhasil menyelesaikan *level* 5 akan mendapatkan *score* 5000.

3.1.4 Analisis Algoritma

Agar permainan *game* menjadi lebih menarik dapat dilakukan dengan meningkatkan kualitas *gameplay* bagi pengguna yang memainkannya. AI merupakan faktor yang penting dalam meningkatkan *gameplay* pada semua jenis permainan *game*, termasuk pada *game* yang akan dibangun. Pathfinding dalam *game adventure* ini digunakan diterapkan pada objek musuh dalam melakukan pencarian karakter pemain di arena permainan. Dengan *pathfinding*, objek musuh

saat menuju ke karakter pemain menjadi lebih agresif. Dalam *game* adventure ini, arena permainan atau *game world* direpresentasikan dalam bentuk kumpulan *tile* atau *grid* yang memiliki koordinat X dan Y atau dengan istilah lain disebut *grid-based*.

Algoritma A* dalam *game* adventure ini akan menganggap *grid* tersebut sebagai simpul-simpul yang membentuk jalan (*path*) dari titik awal hingga ke titik tujuan. Lalu kemudian menghitung biaya dan mengambil jalan yang memakan biaya paling sedikit.

Kinerja algoritma A* bergantung pada fungsi heuristik yang digunakan, *game* adventure ini menggunakan fungsi *heuristic* Non-Manhattan Distance, dengan fungsi *heuristic* yang digunakan adalah sebagai berikut:

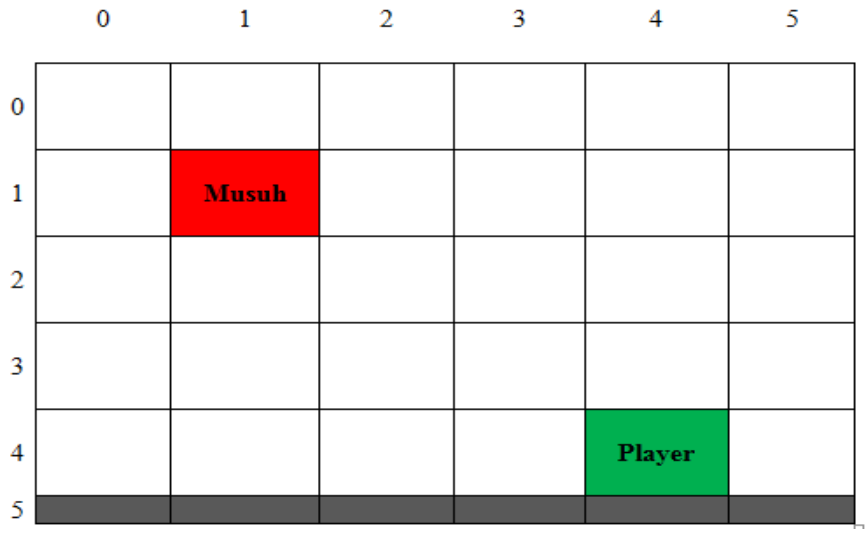
h_diagonal(n) = min(abs(n.x-goal.x) + abs(n.y-goal.y))

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)))

Berikut gambaran proses *pathfinding* dengan A* dalam *game* adventure "Sangkuriang and Indonesian Five Largest Islands" yang digunakan objek musuh untuk menuju ke karakter pemain (*Player*) :

Objek musuh di posisi (1,1) akan menuju ke karakter pemain yang ada di posisi (4,4), sebagaimana terlihat pada gambar 3.3.



Gambar 3.3. Posisi Musuh dan *Play*er dalam grid

Setelah mengetahui titik awal dan titik tujuan, pertama-tama titik awal yang ada di Open List (daftar simpul yang akan dievaluasi), dipindahkan ke Closed List (daftar simpul yang telah dievaluasi). Selanjutnya, diantara simpul (tile) yang telah dievaluasi tersebut, diambil simpul yang memiliki nilai f paling rendah. Nilai f ini diperoleh dari penjumlahan nilai g (jarak dari titik awal ke simpul aktif atau *current node*) dan nilai h (jarak dari simpul aktif atau *current node* ke titik tujuan).

Simpul yang telah dievaluasi kemudian dimasukkan ke Closed List, dan simpul dengan nilai f terendah akan dijadikan simpul aktif untuk proses evaluasi lebih lanjut.

Berikut ini nilai cost / biaya terkecil sebagai Best Node sebagai rute yang akan dipilih musuh berdasarkan fungsi *heuristic* Non-Manhattan Distance, dengan melakukan langkah perhitungan sebagai berikut :

1. Langkah Pertama

a) Koordinat n(0,0)

Nilai n.x = 0

Nilai n.y = 0

Simpul tujuan(4,4) sehingga goal.x = 4, goal.y = 4

g (0,0) = 1,4

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

h_orthogonal(0,0) = (abs(0 - 4) + abs(0 - 4))

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

= 8

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

h_diagonal(0,0) = -(abs(0 - 4)+abs(0 - 4))

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

= -8

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

h(0,0) = (-8) + (8-(2*(-8)))

= (-8) + 24

= 16

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

= 1,4 + 16

= 16,4

b) Koordinat n(0,1)

Nilai n.x = 0

Nilai n.y = 1

Simpul tujuan (4,4) sehingga goal.x = 4, goal.y = 4

g (n) = 1

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

h_orthogonal(0,1) = (abs(0 - 4) + abs(1 - 4))

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

= 7

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

h_diagonal(0,1) = min(abs(0 - 4) + abs(1 - 4))

= min(abs(-4) + abs(-3))

= -7

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

h(0,1) = (-7) + (7-(2*(-7)))

= -7 + 21

= 14

f (0,1) = g (1,0) + h (1,0)

= 1 + 14

= 15

c) Koordinat n(0,2)

Nilai n.x = 0

Nilai n.y = 2

Simpul tujuan (4,4) sehingga goal.x = 4, goal.y = 4

g(n) = 1,4

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

h_orthogonal(0,2) = (abs(0 - 4) + abs(2 - 4))

= (abs(-4) + abs(-2))

= 6

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

h_diagonal(0,2) = min(abs(0 - 4) + abs(2 - 4))

= min(abs(-4) + abs(-2))

= -6

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

h(0,2) = (-6) + (6-(2*(-6)))

= -6 + 18

= 12

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

= 1,4 + 12

= 13,4

d) Koordinat n(2,1)

Nilai n.x = 2

Nilai n.y = 1

Simpul tujuan (4,4) sehingga goal.x = 4, goal.y = 4

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

h_orthogonal(2,1) = (abs(2 - 4) + abs(1 - 4))

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

= 5

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

h_diagonal (2,1) = min(abs(2 - 4) + abs(1 - 4))

= min(abs(-2) + abs(-3))

= -5

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

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

= -5 + 15

= 10

f (2,1) = g (2,1) + h (2,1)

= 1 + 10

= 11

e) Koordinat n(1,2)

Nilai n.x = 1

Nilai n.y = 2

Simpul tujuan(4,4) sehingga goal.x = 4, goal.y = 4

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

h_orthogonal(1,2) = (abs(1 - 4) + abs(2 - 4))

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

= 5

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

h_diagonal(1,2) = (abs(1 - 4) + abs(2 - 4))

= min(abs(-3) + abs(-2))

= -5

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

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

= -5 + 15

= 10

f (1,2) = g (1,2) + h (1,2)

= 1+ 10

= 11

f) Koordinat n(2,2)

Nilai n.x = 2

Nilai n.y = 2

Simpul tujuan(4,4) sehingga goal.x = 4, goal.y = 4

g(n) = 1,4

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

h_orthogonal(2,2) = (abs(2 - 4) + abs(2 - 4))

= (abs(-2) + abs(-2))

= 4

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

h_diagonal(2,2) = -(abs(2 - 4) + abs(2 - 4))

= -(abs(-2) + abs(-2))

= -4

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

h(2,2) = (-4) + (4-(2*(-4)))

= -4 + 12

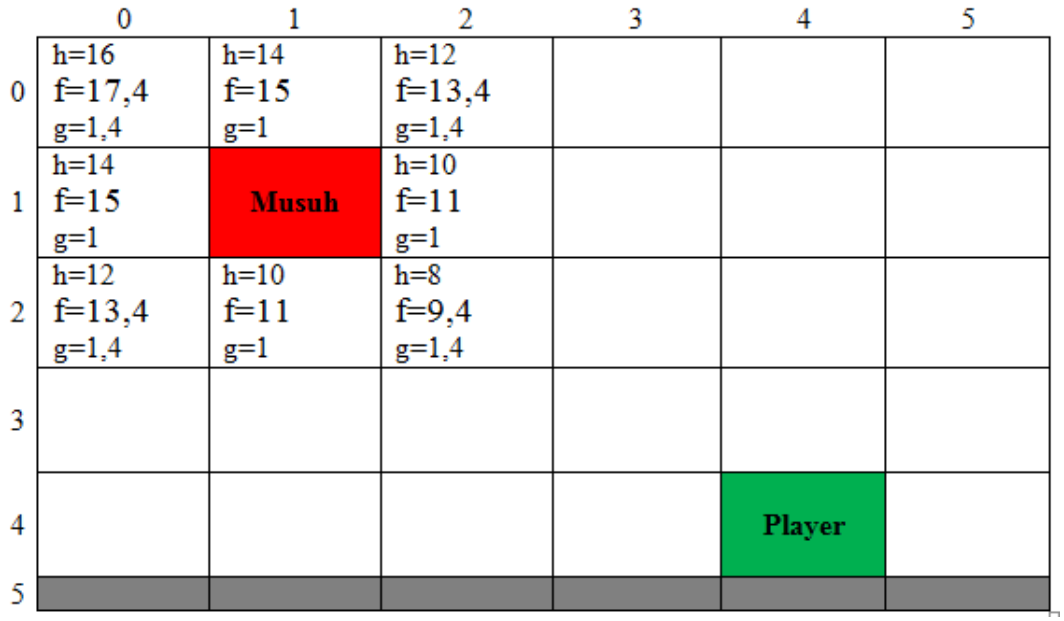
= 8

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

= 1,4 + 8

= 9,4

Lebih jelasnya perhatikan gambar di bawah ini, untuk perhitungan f= g(n) + h(n).



Gambar 3.4. Perhitungan F=g(n) + h(n)

Keterangan :

h : didapatdari rumus non-manhattan distance

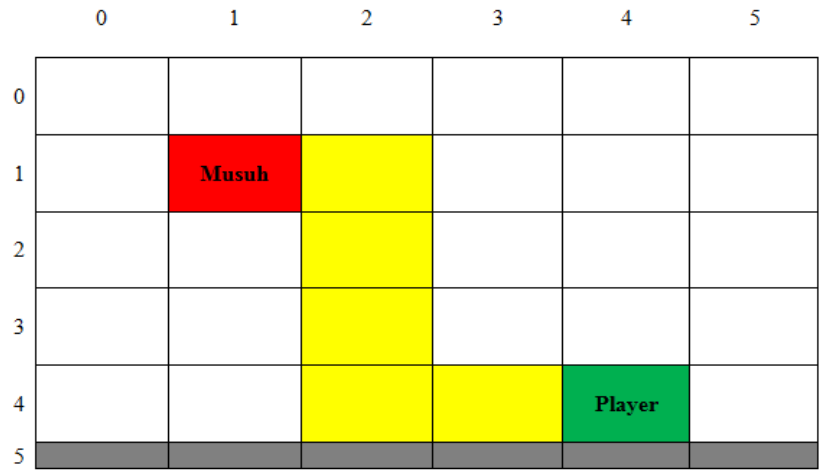
g : didefinisikan pergerakan horizontal dan vertikal= 1, jikadiagonal = 1,4

: tile yang tidak bisa dilewati

Sebagaimana terlihat pada gambar diatas, simpul yang akan dijadikan sebagai simpul aktif (*current node*) selanjutnya adalah simpul yang memiliki nilai f paling rendah. Namun karena ada lebih dari satu simpul dengan nilai f sama kecil, maka simpul yang dijadikan simpul aktif adalah simpul dengan nilai f kecil paling pertama yang diambil dari OpenList yaitu simpul (1,2).

Selanjutnya simpul tetangga dari simpul (1,2) akan dievaluasi dan dimasukkan ke OpenList. Setelah proses ini selesai, simpul aktif (1,2) dimasukkan

ke ClosedList. Lebih lanjut, karena memiliki nilai f kecil dan berada di posisi paling pertama OpenList, simpul (2,2) dijadikan simpul aktif selanjutnya. Simpul tetangga dari simpul (2,2) pun kemudian dievaluasi dan dimasukkan ke OpenList. Proses ini kemudian terus berulang hingga simpul aktif (*current node*) sama dengan titik tujuan. Berikut ini gambar hasil akhir pencarian langkah algoritm A Star.



Gambar 3.5. Hasil akhir pencarian langkah Algoritma A star

Keterangan :

: tile yang tidak bisa dilewati

: Jalan (*path*) optimal yang dapat digunakan musuh untuk menuju ke *Player*

Berikut ini kamus global algoritma A Star pada game adventure "Sangkuriang and Indonesian Five Largest Islands".

Program A_STAR

{Program yang membaca biaya perkiraan dan biaya sebenarnya dan menampilkan startnode dan endnode terbaik}

Kamus:

g : integer (biaya sebenarnya)
h : integer (biaya perkiraan)
startnode : integer (node awal)
endnode: integer (node akhir)
procedure bestPath
{prosedur untuk memproses startnode, endnode, biaya sebenarnya dan biaya perkiraan}

Algoritma:

Input (g,h)
bestPath
Output(startnode, endnode)

3.1.5 Analisis Kebutuhan Non-Fungsional

3.1.5.1 Analisis Kebutuhan Perangkat Keras

Agar aplikasi dapat berjalan dengan baik, maka dibutuhkan perangkat keras yang sesuai dengan kebutuhan aplikasi. Di bagian ini akan dijelaskan analisis kebutuhan perangkat keras dari sisi pengembang dan dari sisi pemain.

1. Dari sisi pengembang

Perangkat keras yang digunakan untuk membangun *game* ini adalah sebagai berikut:

1. Prosesor 2 GHz
2. Memori 4 GB
3. Space kosong di harddisk 5 GB
4. VGA card 512 MB
5. Dis*play* LCD Monitor dengan resolusi 1366 x 768

2. Dari sisi pemain

Spesifikasi minimum perangkat keras yang dibutuhkan agar dapat menjalankan *game* ini adalah sebagai berikut:

1. *Smartphone* Android *2.2 (froyo)* ke atas
2. Layar *touchscreen*
3. *Processor* 800 Mhz, 32 MB RAM

3.1.5.2 Analisis Kebutuhan Perangkat Lunak

Perangkat keras komputer tidak berarti tanpa perangkat lunak, begitu juga sebaliknya. Jadi perangkat lunak dan perangkat keras saling mendukung satu sama lain. Perangkat keras hanya berfungsi jika diberikan instruksi - intruksi kepadanya. Instruksi-instruksi inilah yang disebut dengan perangkat lunak.

1. Dari sisi pengembang

Perangkat lunak yang digunakan untuk mengembangkan *game* ini adalah sebagai berikut:

1. Sistem operasi *Windows*
2. Java *Development KIT versi* 6
3. *Eclipse Helios*
4. Android *SDK*
5. Android *Development Tool*
6. *Framework Kilobolt*

2. Dari sisi pemain

*Game adventure "*Sangkuriang *and Indonesian five largest islands"* ini dibutuhkan perangkat lunak untuk menjalankannya, yaitu Sistem Operasi Android Android *2.2 (froyo)* keatas.

3.1.5.3 Analisis Pengguna

Analisis pengguna gambaran tentang target maupun profil pengguna untuk *game* yang akan dibangun. Perangkat keras dan perangkat lunak yang ada tidak akan berguna apabila tidak ada pengguna yang mengoperasikannya. Berikut ini target maupun profil pengguna untuk *game adventure* "Sangkuriang and Indonesian five largest islands", dapat dilihat pada Tabel III.7.

1. Target Pengguna

Target pengguna (target user) untuk *game* ini adalah pengguna dengan karakteristik sebagai berikut :

Tabel 3.6. Target Pengguna

1	Peran	Pemain
2	Usia	10 tahun ke atas
3	Tingkat Keterampilan	Dapat menggunakan *smartphone* Android
4	Pengalaman	• Pernah memainkan aplikasi *game mobile* • Pernah memainkan aplikasi *game* dengan layar sentuh
5	Minat terhadap *game casual*	Baik

2. Profil Pengguna

Profil pengguna (*user profile*) untuk *game* ini dapat dikelompokkan sebagai berikut :

a. *Human Information Processing System*

Sistem pemroses informasi manusia yaitu berkaitan dengan antarmuka dan fungsionalitas. Antarmuka dan fungsionalitas yang ada dalam

game ini dibuat sederhana agar pengguna dapat dengan mudah memainkan *game* ini.

b. *User Pshycology Characteristic*

Karakteristik psikologi pengguna untuk *game* ini haruslah pengguna yang memiliki kegemaran atau setidaknya berminat untuk memainkan *game* berjenis *adventure*. Selain itu juga tertarik dengan kebudayaan dan ingin mempelajarinya.

c. *User Knowledge and Experience Characteristic*

Pengguna yang memainkan *game* ini harus memiliki pengetahuan dasar tentang cara pengoperasian *smartphone* android. Selain itu, pengguna juga diharapkan pernah bermain *game* berjenis *adventure* sebelumnya.

d. *User Job and Task Characteristic*

Untuk lebih memudahkan dalam memainkan *game* ini, Pengguna harus memiliki tingkat frekuensi penggunaan (*frequency of use*) yang tinggi. Kategori pekerjaan pengguna yang memainkan *game* ini mencakup siswa atau siapa saja yang berusia 10 tahun ke atas yang ingin mempelajari kebudayaan dan gemar bermain *game*.

e. *User Physical Characteristic*

Diharapkan tidak positif mengidap **buta warna** karena *game* ini adalah *game* yang *full color*, namun tidak mempengaruhi banyak saat memainkannya. ***Handedness user*** tidak mempengaruhi dalam memainkannya karena tidak akan berefek *human error* bila semua *skill handedness* di implementasikan, hanya perlu penyesuaian. Selain itu, Gender pengguna untuk *game* ini tidak dibatasi atau dengan kata lain, *game* ini dapat dimainkan oleh pengguna dengan gender apapun.

f. *User Physical Environment*

Dalam memainkan *game* ini, dibutuhkan cukup konsentrasi pengguna agar cerita dan pesan dalam *game* ini dapat tersampaikan dan dapat diingat dengan baik oleh pengguna. Selain itu, *game* ini tidak terlalu

mempermasalahkan mengenai privasi karena pada dasarnya *game* ini dibuat untuk umum.

g. *User Tools*

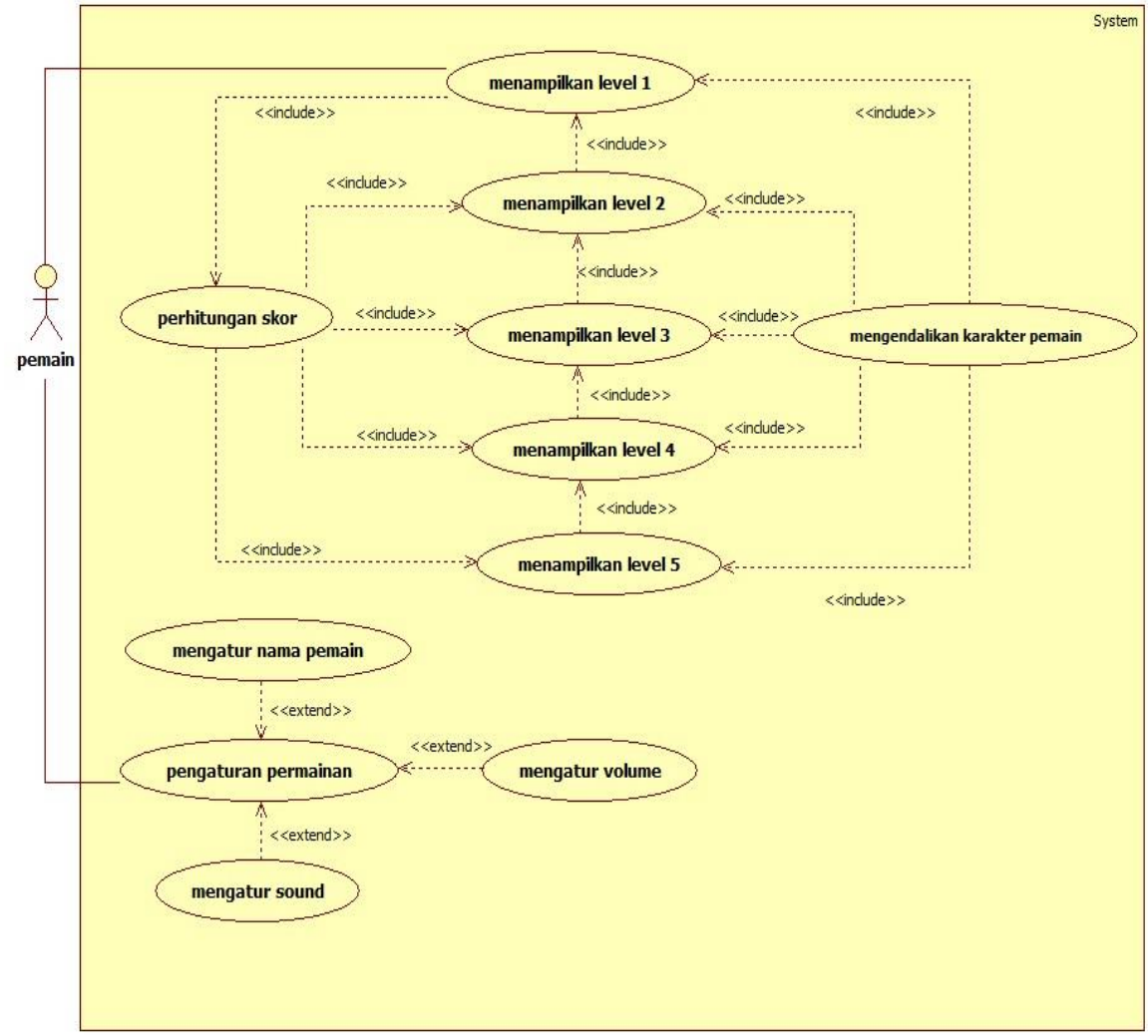
Selain *smartphone* android, pengguna tidak perlu menggunakan alat tambahan lainnya dikarenakan hanya dengan memiliki *smartphone* android pengguna dapat dengan mudah memainkan *game* ini.

3.1.6 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. Pada bagian ini akan dibahas tentang *use case diagram*, *activity diagram*, *class diagram*, dan *sequence diagram*.

3.1.6.1 *Use case diagram*

Pada sub bab berikut ini dijelaskan bagaimana *Use Case* yang terjadi dalam pembangunan *game* ini yang dimodelkan dalam sekumpulan *Use Case* dan Actor dan bagaimana hubungan-hubungannya yang akan dipetakan dalam diagram *Use Case*, dilengkapi dengan skenario untuk menjelaskan dari gambaran *Use Case* yang ada dengan menggambarkan skenario per *Use Case*. *Use case diagram* dapat dilihat pada Gambar 3.6 berikut ini.



Gambar 3.6. ***Use Case Diagram Game Adventure*** **"Sangkuriang** ***and Indonesian Five Largest Islands"***

1. ***Actor Definition***

   Actor Definition berfungsi untuk menjelaskan *Actor* yang terdapat pada *Use case diagram*. *Actor Definition* diterangkan pada Tabel 3.7 di bawah ini.

Tabel 3.7. ***Actor Definition***

No	*Actor*	Deskripsi
1	Pemain	Orang yang memainkan *game*

2. *Use Case Definition*

Use Case Definition berfungsi untuk menjelaskan fungsi *Use Case* yang terdapat pada *Use case diagram. Use Case Definition* diterangkan pada tabel di bawah ini.

Tabel 3.8. *Use Case Definition*

No	*Use Case*	Deskripsi
1	Menampilkan level 1	Proses untuk memulai permainan dan menampilkan level 1
2	Menampilkan level 2	Proses untuk menampilkan permainan level 2
3	Menampilkan level 3	Proses untuk menampilkan permainan level 3
4	Menampilkan level 4	Proses untuk menampilkan permainan level 4
5	Menampilkan level 5	Proses untuk menampilkan permainan level 5
6	Mengendalikan karakter pemain	Proses yang dilakukan oleh pemain untuk menggerakkan karakter pemain diarena permainan
7	Perhitungan skor	Proses pada permainan yang akan menghitung skor dari pemain
8	Pengaturan permainan	Proses untuk menampilkan pengaturan pada *game*
9	Mengatur nama pemain	Proses untuk mengubah nama pemain
10	Mengatur sound	Proses untuk mengatur *sound* pada *game*,

		yaitu on/ off
11	Mengatur volume	Proses untuk mengatur volume pada *game*

3. *Use Case Scenario*

Use Case Scenario mendeskripsikan urutan langkah-langkah dalam proses bisnis baik yang dilakukan aktor terhadap sistem maupun yang dilikakukan oleh sistem terhadap aktor. Berdasarkan *use case diagram* pada Gambar III.6 diatas maka *use case scenario* untuk aplikasi yang dibangun akan dijelaskan sebagai berikut.

***a. Use Case Scenario* Menampilkan Level 1**

Use Case Scenario dari *Use Case* Menampilkan Level 1 dijelaskan pada tabel di bawah ini.

Tabel 3.9. *Use Case Scenario* Menampilkan Level 1

Identifikasi	
Nomor	1
Nama	Menampilkan Level 1
Tujuan	Proses yang dilakukan pemain untuk memulai permainan dan menampilkan permainan pada level 1
Aktor	Pemain
Skenario Utama	
Kondisi Awal	Pemain berada di menu utama
Aksi Aktor	**Reaksi Sistem**
1. Memilih tombol menu *Play*	

	2. Sistem menampilkan prolog
	3. Sistem menampilkan arena permainan
	4. Sistem menginisialisasi posisi pemain
	5. Sistem menginisialisasi posisi musuh
6. Menekan Start	
	7. Memulai permainan
	8. Menampilkan permainan level 1
9. Mengendalikan karakter pemain	
	10. Karakter bergerak
	11. Cek makanan
	12. Cek collision saat mengambil makanan
	13. Perhitungan *score* makanan yang didapat
	14. Jalankan AI musuh bereaksi
15. Menyerang musuh	
	16. Perhitungan *score* saat mengalahkan musuh
	17. Jika musuh menabrak pemain, maka pemain mati

18. Menyelesaikan level permainan	
	19. Mengupdate informasi permainan
	20. Jika berhasil menyelesaikan level, pemain pindah ke *level* berikutnya
	21. Score bertambah
	22. Jika Tidak (*game* over), kembali ke awal permainan
Kondisi Akhir	Permainan lanjut ke *level* berikutnya atau kembali ke awal permainan.

b. Use Case Scenario Menampilkan Level 2

Use Case Scenario dari *Use Case* Menampilkan Level 2 dijelaskan pada tabel di bawah ini.

Tabel 3.10. *Use Case Scenario* Menampilkan Level 2

Identifikasi	
Nomor	2
Nama	Menampilkan Level 2
Tujuan	Proses untuk menampilkan permainan pada level 1
Aktor	Pemain
Skenario Utama	
Kondisi Awal	Pemain berada di arena permainan

Aksi Aktor	**Reaksi Sistem**
	1. Sistem menampilkan arena permainan level 2
	2. Sistem menginisialisasi posisi pemain
	3. Sistem menginisialisasi posisi musuh
4. Mengendalikan karakter pemain	
	5. Karakter bergerak
	6. Cek makanan
	7. Cek collision saat mengambil makanan
	8. Perhitungan *score* makanan yang didapat
9. Menyerang musuh	
	10. Perhitungan *score* saat mengalahkan musuh
	11. Jika musuh menabrak pemain, maka pemain mati
12. Menyelesaikan level permainan	
	13. Mengupdate informasi permainan
	14. Jika berhasil menyelesaikan level, pemain pindah ke *level* berikutnya

	15. Score bertambah
	16. Jika Tidak (*game* over), kembali ke awal permainan
Kondisi Akhir	Permainan lanjut ke *level* berikutnya atau kembali ke awal permainan.

c. Use Case Scenario Menampilkan Level 3

Use Case Scenario dari *Use Case* Menampilkan Level 3 dijelaskan pada tabel di bawah ini.

Tabel 3.11. *Use Case Scenario* Menampilkan Level 3

Identifikasi	
Nomor	3
Nama	Menampilkan Level 4
Tujuan	Proses untuk menampilkan permainan pada level 3
Aktor	Pemain
Skenario Utama	
Kondisi Awal	Pemain berada di arena permainan
Aksi Aktor	**Reaksi Sistem**
	1. Sistem menampilkan arena permainan level 3
	2. Sistem menginisialisasi posisi

	pemain
	3. Sistem menginisialisasi posisi musuh
4. Mengendalikan karakter pemain	
	5. Karakter bergerak
	6. Cek makanan
	7. Cek collision saat mengambil makanan
	8. Perhitungan *score* makanan yang didapat
9. Menyerang musuh	
	10. Perhitungan *score* saat mengalahkan musuh
	11. Jika musuh menabrak pemain, maka pemain mati
12. Menyelesaikan level permainan	
	13. Mengupdate informasi permainan
	14. Jika berhasil menyelesaikan level, pemain pindah ke *level* berikutnya
	15. Score bertambah
	16. Jika Tidak (*game* over), kembali ke awal permainan
Kondisi Akhir	Permainan lanjut ke *level* berikutnya

	atau kembali ke awal permainan.

d. Use Case Scenario Menampilkan Level 4

Use Case Scenario dari *Use Case* Menampilkan Level 4 dijelaskan pada tabel di bawah ini.

Tabel 3.12. *Use Case Scenario* Menampilkan Level 4

Identifikasi	
Nomor	4
Nama	Menampilkan Level 4
Tujuan	Proses untuk menampilkan permainan pada level 4
Aktor	Pemain
Skenario Utama	
Kondisi Awal	Pemain berada di arena permainan
Aksi Aktor	**Reaksi Sistem**
	1. Sistem menampilkan arena permainan level 3
	2. Sistem menginisialisasi posisi pemain
	3. Sistem menginisialisasi posisi musuh
4. Mengendalikan karakter pemain	

	5. Karakter bergerak
	6. Cek makanan
	7. Cek collision saat mengambil makanan
	8. Perhitungan *score* makanan yang didapat
9. Menyerang musuh	
	10. Perhitungan *score* saat mengalahkan musuh
	11. Jika musuh menabrak pemain, maka pemain mati
12. Menyelesaikan level permainan	
	13. Mengupdate informasi permainan
	14. Jika berhasil menyelesaikan level, pemain pindah ke *level* berikutnya
	15. Score bertambah
	16. Jika Tidak (*game* over), kembali ke awal permainan
Kondisi Akhir	Permainan lanjut ke *level* berikutnya atau kembali ke awal permainan.

e. Use Case Scenario Menampilkan Level 5

Use Case Scenario dari *Use Case* Menampilkan Level 5 dijelaskan pada tabel di bawah ini.

Tabel 3.13. *Use Case Scenario* Menampilkan Level 5

Identifikasi	
Nomor	5
Nama	Menampilkan Level 5
Tujuan	Proses untuk menampilkan permainan pada level 5
Aktor	Pemain
Skenario Utama	
Kondisi Awal	Pemain berada di arena permainan
Aksi Aktor	**Reaksi Sistem**
	1. Sistem menampilkan arena permainan level 5
	2. Sistem menginisialisasi posisi pemain
	3. Sistem menginisialisasi posisi musuh
4. Mengendalikan karakter pemain	
	5. Karakter bergerak
	6. Cek makanan
	7. Cek collision saat mengambil makanan
	8. Perhitungan *score* makanan yang didapat

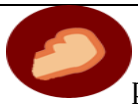
9. Menyerang musuh	
	10. Perhitungan *score* saat mengalahkan musuh
	11. Jika musuh menabrak pemain, maka pemain mati
12. Menyelesaikan level permainan	
	13. Mengupdate informasi permainan
	14. Jika berhasil menyelesaikan level, pemain pindah ke *level* berikutnya
	15. Score bertambah
	16. Jika Tidak (*game* over), kembali ke awal permainan
Kondisi Akhir	Permainan lanjut ke *level* berikutnya atau kembali ke awal permainan.

f. Use Case Scenario Mengendalikan Karakter Pemain

Use Case Scenario dari *Use Case* Mengendalikan Karakter Pemain dijelaskan pada tabel di bawah ini.

Tabel 3.14. *Use Case Scenario* Mengendalikan Karakter Pemain

Identifikasi	
Nomor	6
Nama	Mengendalikan Karakter Pemain

Tujuan	Proses yang dilakukan pemain untuk mengendalikan karakter pemain diarena permainan
Aktor	Pemain
Skenario Utama	
Kondisi Awal	Pemain berada di arena permainan
Aksi Aktor	**Reaksi Sistem**
1. Menekan tombol  pada layar sentuh	
	2. Mengerakkan karakter pemain kearah kiri
3. Menekan tombol  pada layar sentuh	
	4. Mengerakkan karakter pemain kearah kanan
5. Menekan tombol  pada layar sentuh	
	6. Mengerakkan karakter pemain kearah atas (meloncat)
7. Menekan tombol  pada layar sentuh	
	8. Mengerakkan karakter pemain untuk

	memukul
Kondisi Akhir	Pemain menekan tombol selain tombol  pada layar sentuh, maka karakter pemain tidak akan bergerak

g. Use Case Scenario Perhitungan Skor

Use Case Scenario dari *Use Case* Perhitungan Skor dijelaskan pada tabel di bawah ini.

Tabel 3.15. *Use Case Scenario* Perhitungan Skor

Identifikasi	
Nomor	7
Nama	Perhitungan Skor
Tujuan	Proses yang akan menginformasikan nilai skor pemain
Aktor	Pemain
Skenario Utama	
Kondisi Awal	Pemain berada di arena permainan
Aksi Aktor	**Reaksi Sistem**
1. Pemain memukul musuh	
	2. Objek musuh yang terpukul akan

	hilang dari arena permainan
	3. Skor Pemain akan bertambah 10
Skenario Alternatif	
1. Pemain memukul musuh	
	2. Objek musuh tidak terkena pukulan dan akan tetap mengejar pemain.
	3. Skor pemain tidak akan bertambah

h. Use Case Scenario Pengaturan Permainan

Use Case Scenario dari *Use Case* Pengaturan Permainan dijelaskan pada tabel di bawah ini .

Tabel 3.16. *Use Case Scenario* Pengaturan Permainan

Identifikasi	
Nomor	8
Nama	Pengaturan permainan
Tujuan	Proses untuk melakukan pengaturan permainan yang akan dimainkan
Aktor	Pemain
Skenario Utama	
Kondisi Awal	Pemain berada di menu utama
Aksi Aktor	**Reaksi Sistem**

4. Memilih tombol menu *Option*	
	5. Menampilkan menu pilihan pada pengaturan (*Player name*, *Sound* dan *Volume*)
Kondisi Akhir	Menampilkan tampilan pengaturan permainan

i. Use Case Scenario Mengatur Nama Pemain

Use Case Scenario dari *Use Case* Mengatur Nama Pemain dijelaskan pada tabel di bawah ini.

Tabel 3.17. *Use Case Scenario* Mengatur Nama Pemain

Identifikasi	
Nomor	9
Nama	Mengatur Nama Pemain
Tujuan	Untuk mengubah nama pemain yang menjalankan aplikasi *game* ini
Aktor	Pemain
Skenario Utama	
Kondisi Awal	Pemain berada di menu utama
Aksi Aktor	**Reaksi Sistem**
1. Memilih tombol menu Option	
	2. Menampilkan menu pengaturan *game*

	(Player name, Sound dan *Volume*)
3. Mengubah nama pada form menu *Player name*	
	4. Sistem mengatur nama pemain sesuai yang diinginkan pemain
	5. Menampilkan *Play*er Name yang baru
Kondisi Akhir	*Player Name* pada menu pengaturan berubah

j. Use Case Scenario Mengatur *Sound*

Use Case Scenario dari *Use Case* Mengatur *Sound* dijelaskan pada tabel di bawah ini.

Tabel 3.18. *Use Case Scenario* Mengatur Sound

Identifikasi	
Nomor	10
Nama	Mengatur *Sound*
Tujuan	Untuk mengatur *sound* pada *game*
Aktor	Pemain
Skenario Utama	
Kondisi Awal	Pemain berada di menu utama
Aksi Aktor	**Reaksi Sistem**

1. Memilih tombol menu *Option*	
	2. Menampilkan menu pengaturan *game (Player name*, *Sound* dan *Volume*)
3. Memilih menu pengaturan *Sound* (*on*/*off*)	
4. Jika pemain memilih *On*	
	5. Menghidupkan *sound game*
6. Jika pemain memilih *Off*	
	7. Mematikan *sound game*
Kondisi Akhir	Mengatur *sound* pada *game* sesuai yang diinginkan pemain

k. Use Case Scenario Mengatur *Volume*

Use Case Scenario dari *Use Case* Mengatur *Volume* dijelaskan pada tabel di bawah ini.

Tabel 3.19. *Use Case Scenario* Mengatur *Volume*

Identifikasi	
Nomor	11
Nama	Mengatur *Volume*
Tujuan	Untuk mengatur *volume* suara pada *game*
Aktor	Pemain

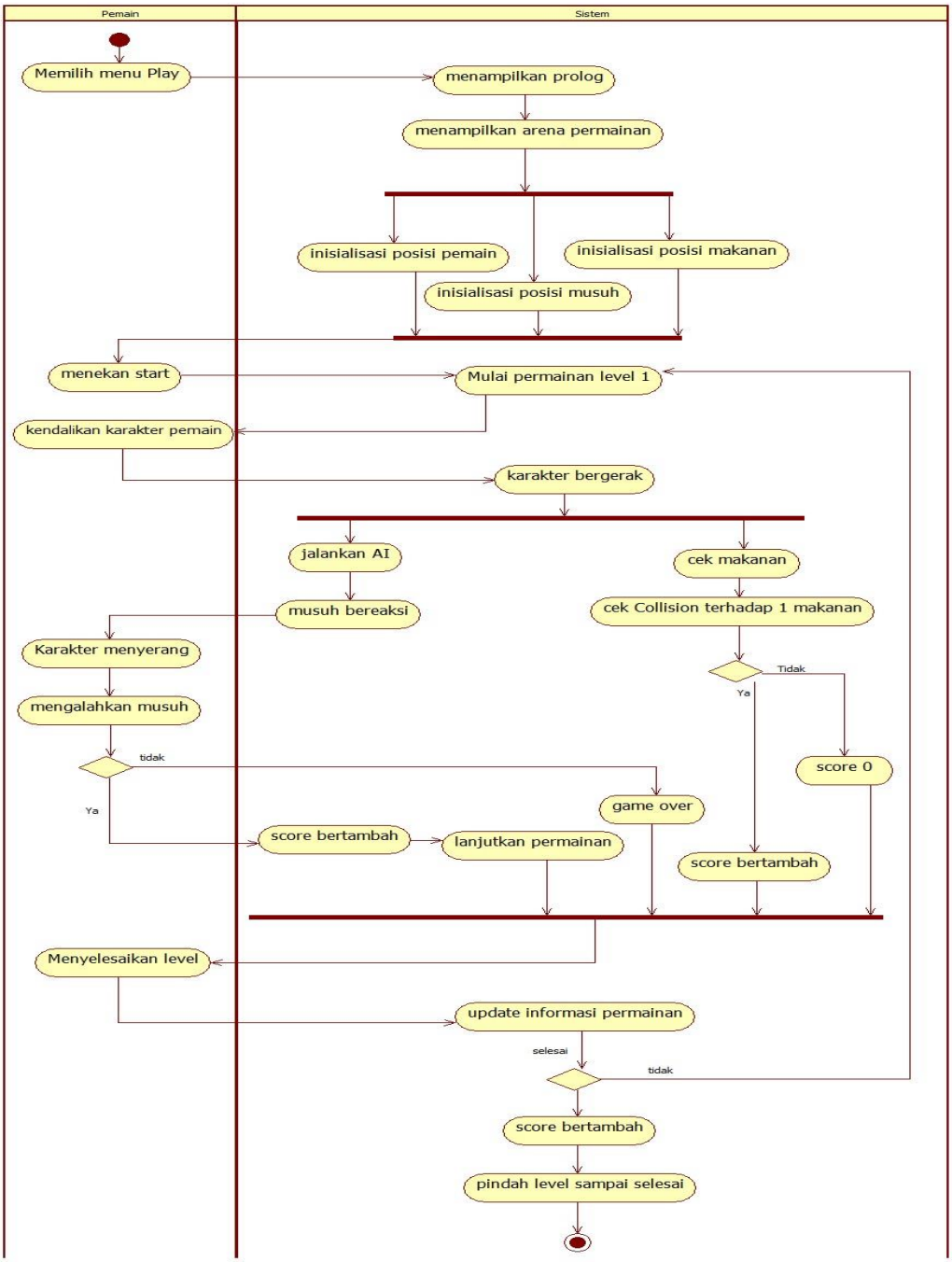
Skenario Utama	
Kondisi Awal	Pemain berada di menu utama
Aksi Aktor	**Reaksi Sistem**
1. Memilih tombol menu *Option*	
	2. Menampilkan menu pengaturan *game (Player name, Sound* dan *Volume*)
3. Memilih dan mengatur menu *Volume* suara	
4. Jika pemain menggeserkan ke kanan	
	5. Membesarkan volume suara pada permainan
6. Jika pemain menggeserkan ke kiri	
	7. Mengecilkan volume suara pada permainan
Kondisi Akhir	Mengatur *volume* suara pada *game* sesuai yang diinginkan pemain

3.1.6.2 Activity Diagram

Activity diagram menggambarkan berbagai alir aktivitas dalam sistem yang sedang dirancang, bagaimana masing-masing alir berawal, *decision* yang mungkin terjadi, dan bagaimana mereka berakhir. *Activity diagram* juga dapat menggambarkan proses paralel yang mungkin terjadi pada beberapa eksekusi. Berikut ini *activity diagram* pada aplikasi *game adventure "*Sangkuriang *and Indonesian Five Largest Islands".*

1. *Activity Diagram* Menampilkan level 1

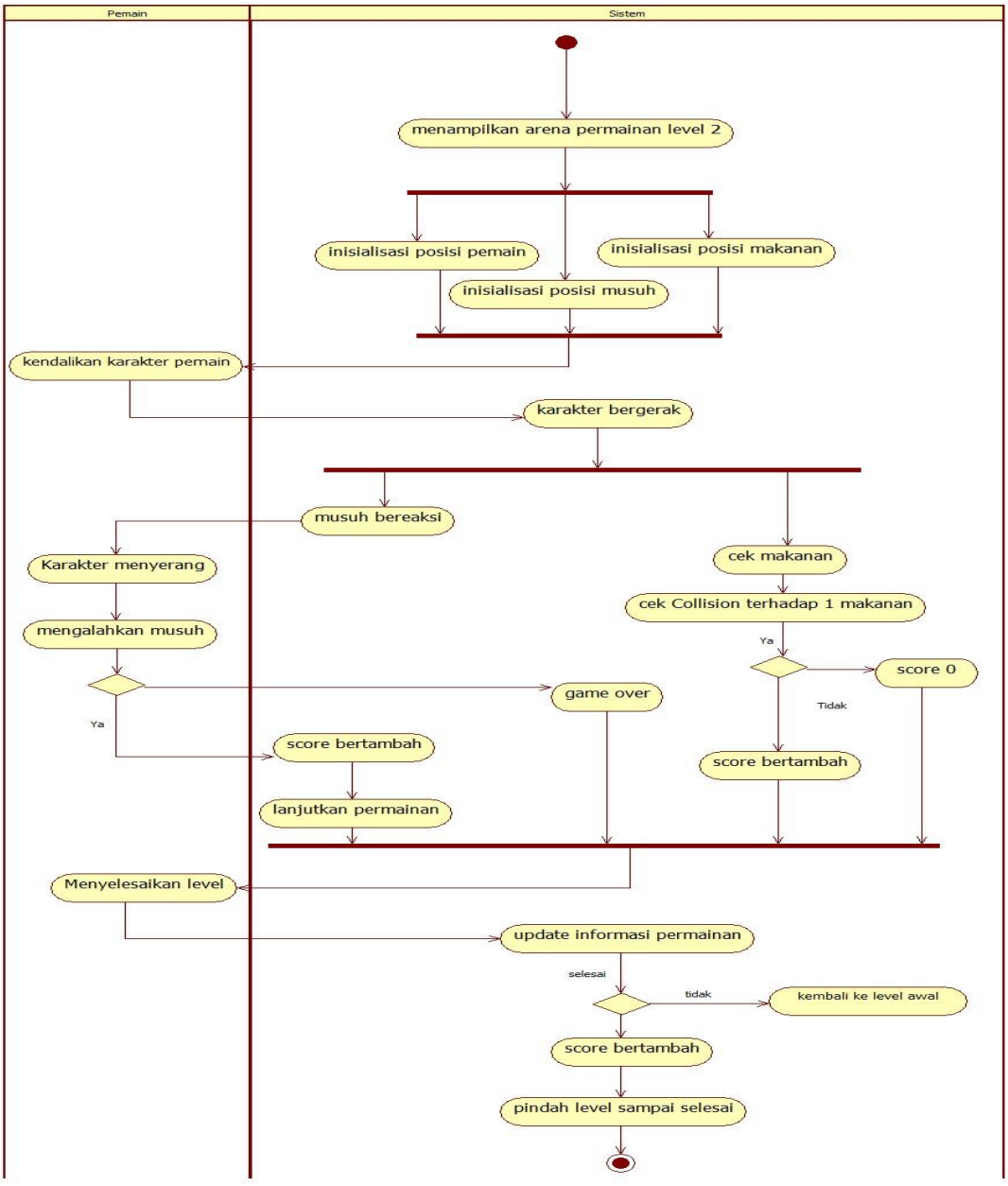
Berikut ini adalah diagram yang menunjukan alur aksi pada aktivitas menampilkan level 1 yang dapat dilihat pada gambar berikut ini.



Gambar 3.7. *Activity Diagram* Menampilkan Level 1

2. *Activity Diagram* Menampilkan Level 2

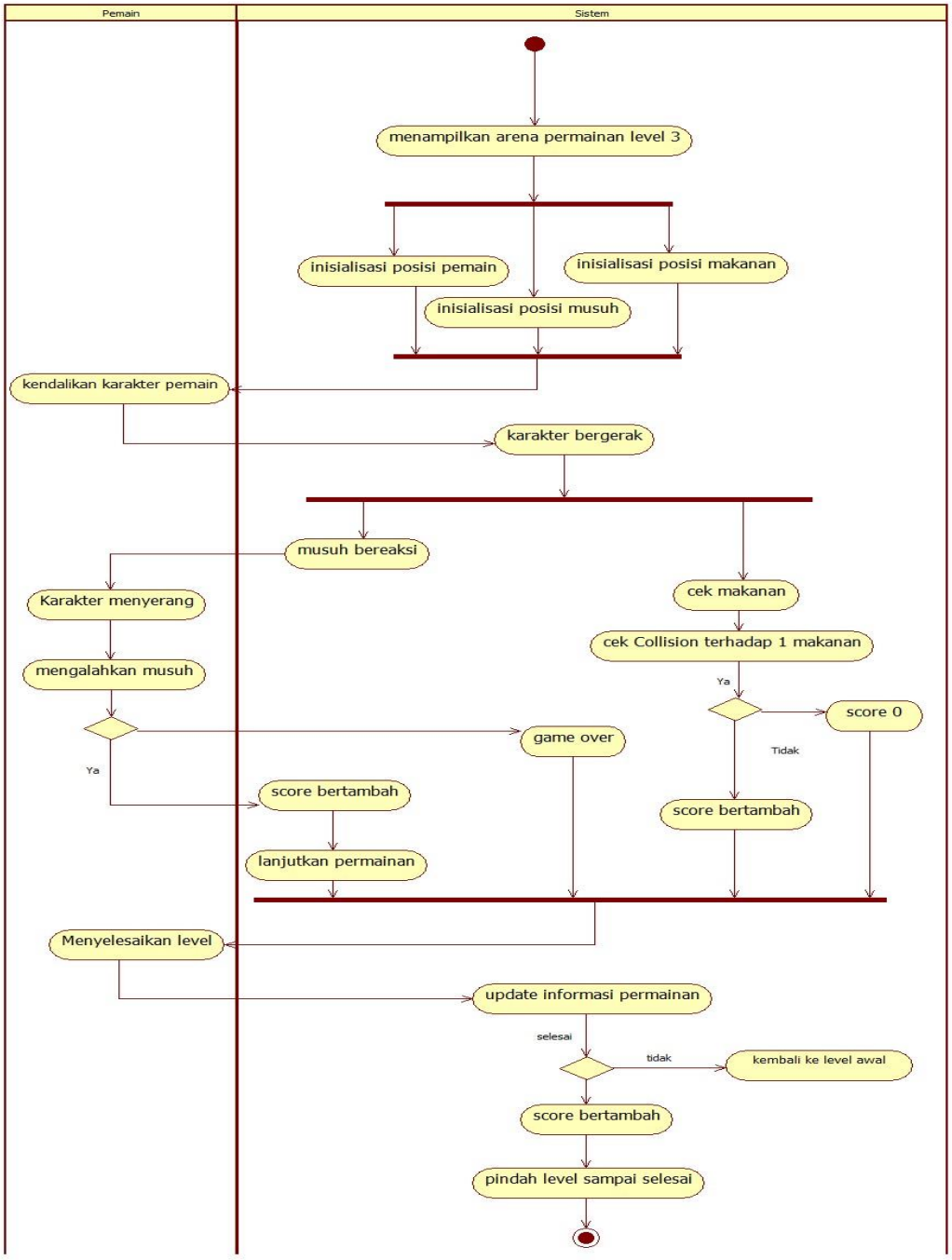
Berikut ini adalah diagram yang menunjukan alur aksi pada aktivitas menampilkan level 2 yang dapat dilihat pada gambar berikut ini.



Gambar 3.8. *Activity Diagram* Menampilkan Level 2

3. *Activity Diagram* Menampilkan Level 3

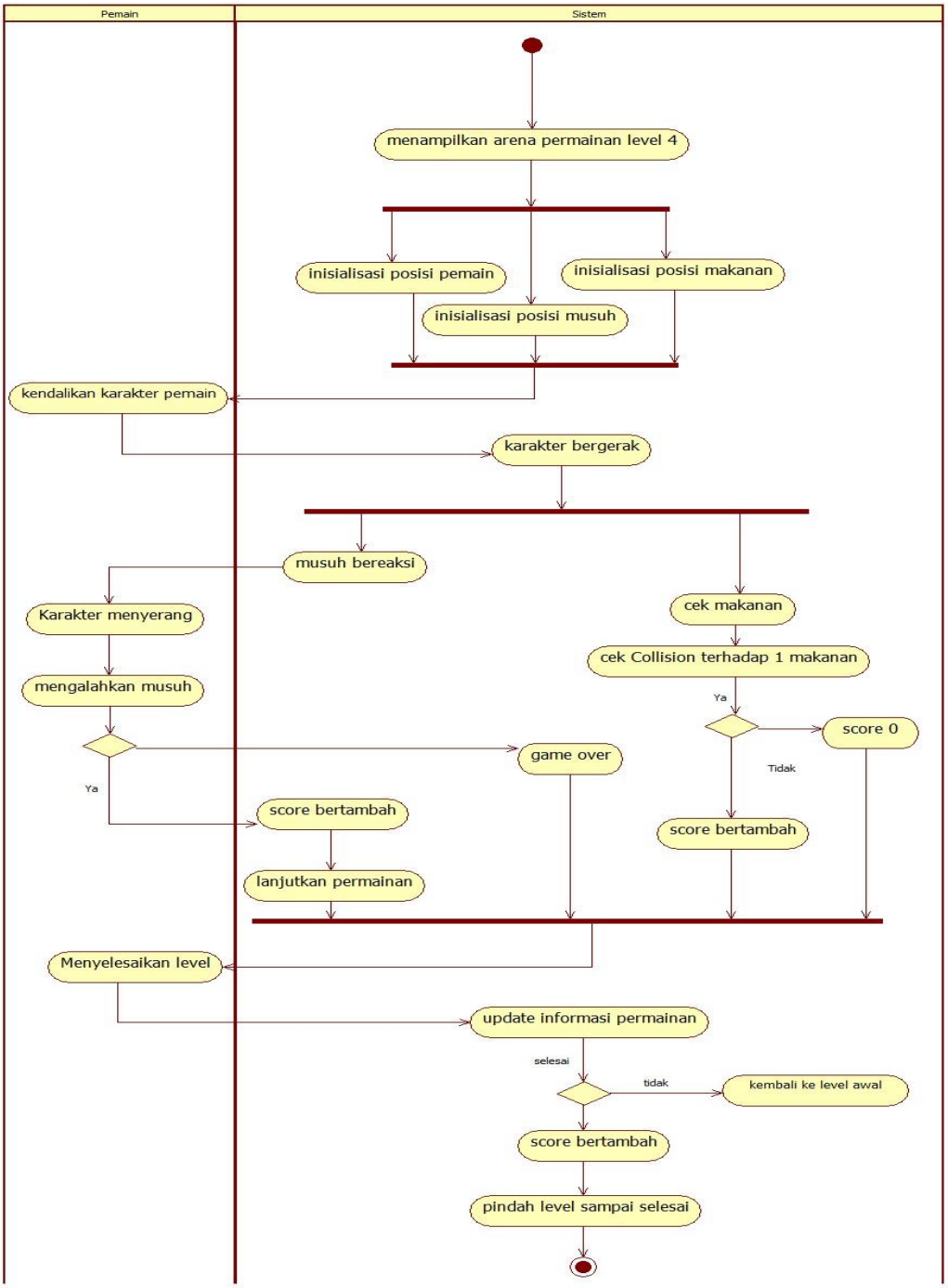
Berikut ini adalah diagram yang menunjukan alur aksi pada aktivitas menampilkan level 3 yang dapat dilihat pada gambar berikut ini.



Gambar 3.9. *Activity Diagram* Menampilkan Level 3

4. *Activity Diagram* Menampilkan Level 4

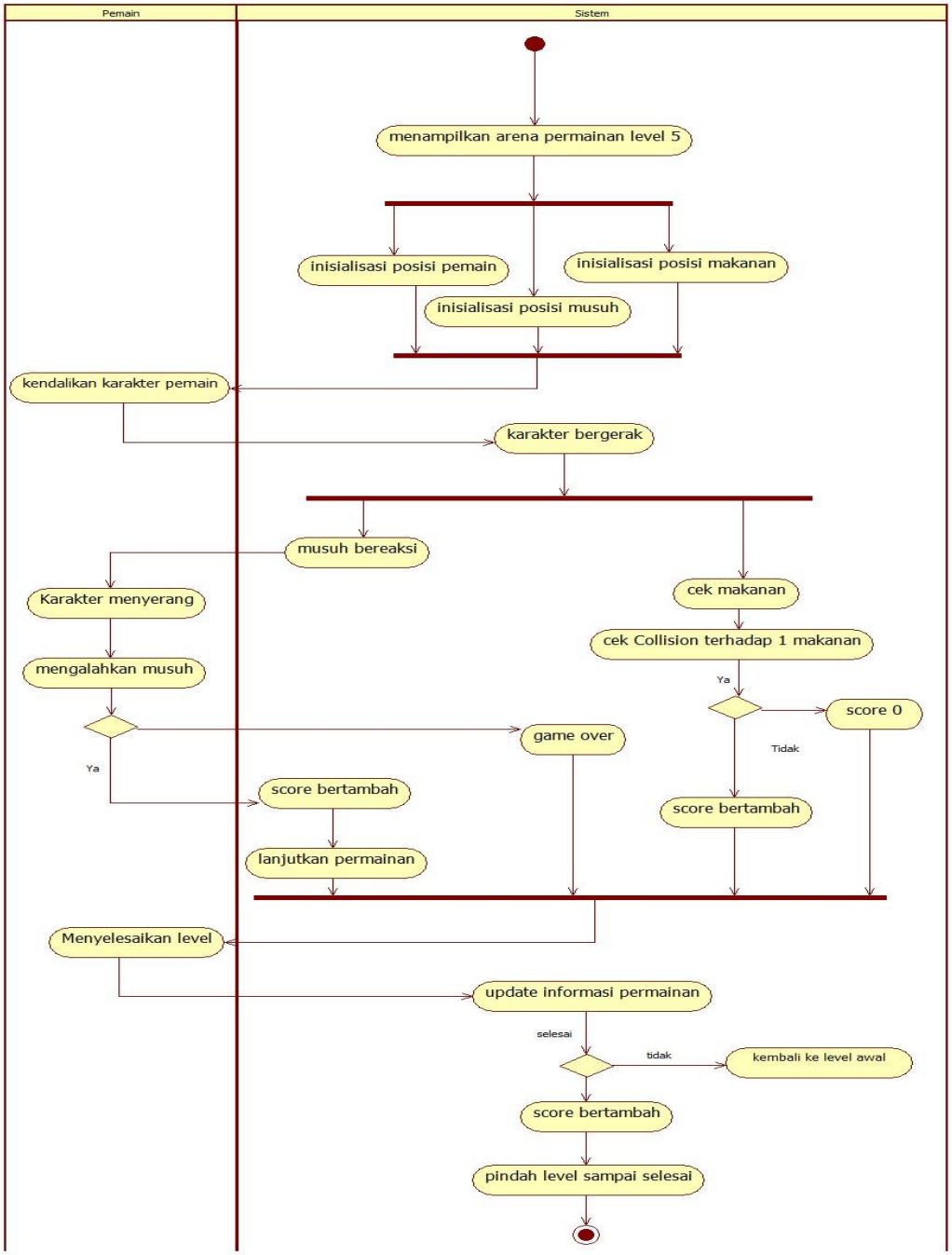
Berikut ini adalah diagram yang menunjukan alur aksi pada aktivitas menampilkan level 4 yang dapat dilihat pada gambar berikut ini.



Gambar 3.10. Activity *Diagram* Menampilkan Level 4

5. *Activity Diagram* Menampilkan Level 5

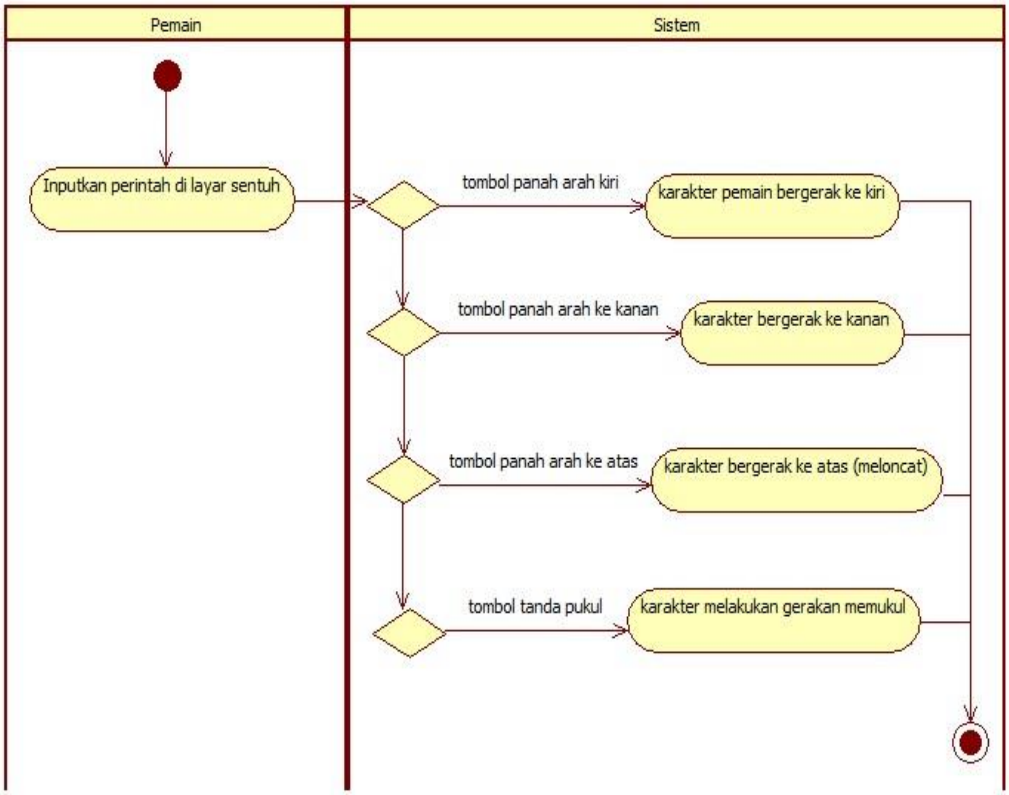
Berikut ini adalah diagram yang menunjukan alur aksi pada aktivitas menampilkan level 4 yang dapat dilihat pada gambar berikut ini.



Gambar 3.11. Activity *Diagram* Menampilk**an Level 5**

6. *Activity Diagram* Mengendalikan Karakter Pemain

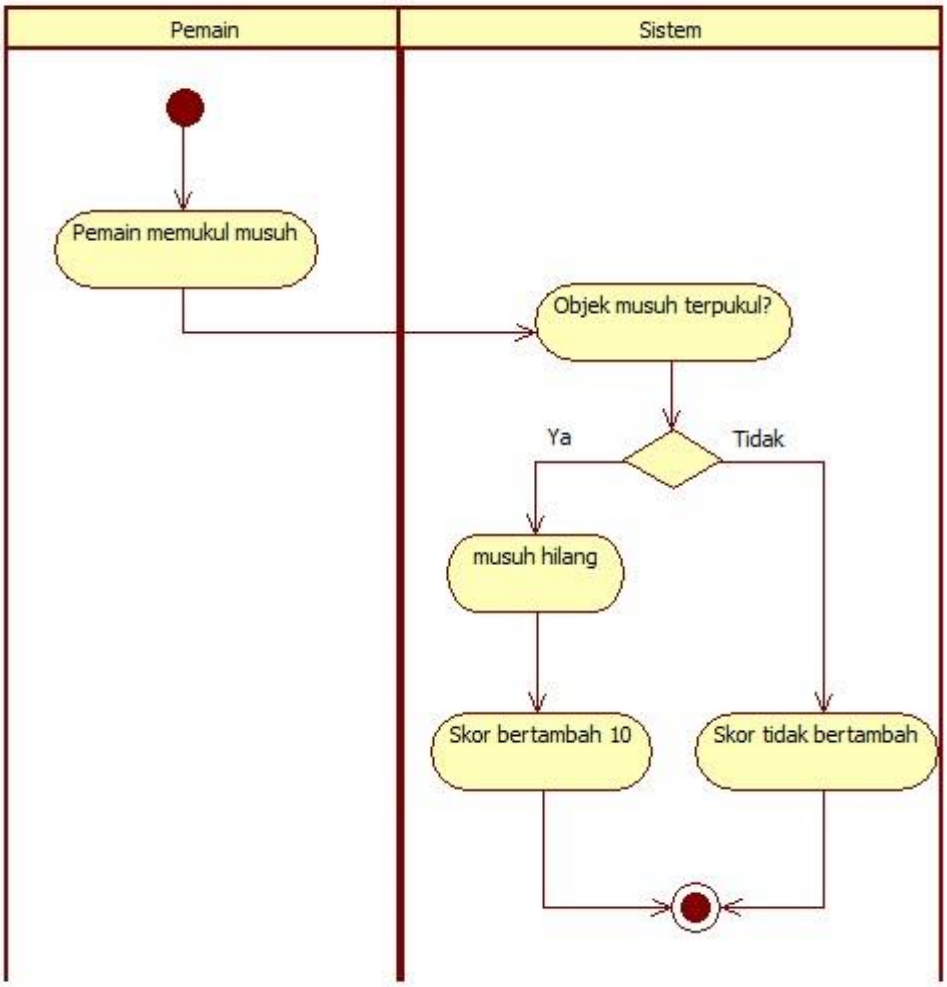
Berikut ini adalah diagram yang menunjukan alur aksi pada aktivitas Mengendalikan Karakter Pemain yang dapat dilihat pada gambar di bawah ini.



Gambar 3.12. *Activity Diagram* Mengendalikan Karakter Pemain

7. *Activity Diagram* Perhitungan Skor

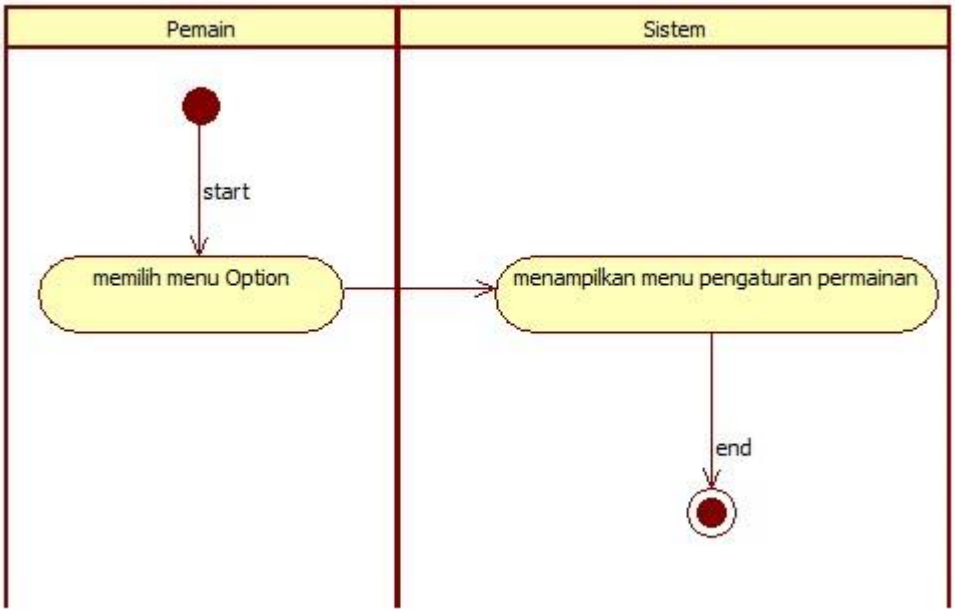
Berikut ini adalah diagram yang menunjukan alur aksi pada aktivitas Mengendalikan Karakter Pemain yang dapat dilihat pada gambar di bawah ini.



Gambar 3.13. *Activity Diagram* Perhitungan Skor

8. *Activity Diagram* Pengaturan Permainan

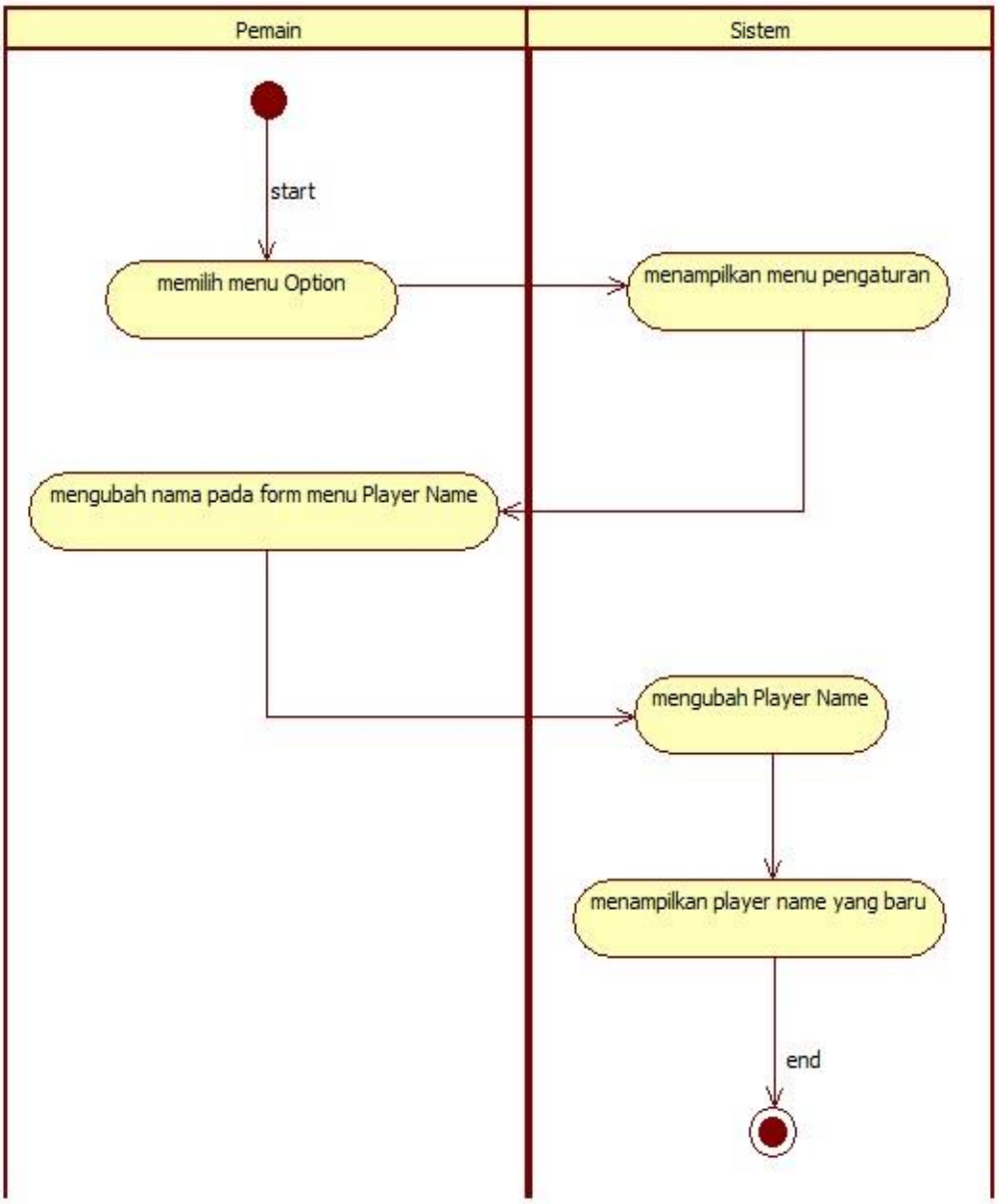
Berikut ini adalah diagram yang menunjukan alur aksi pada aktivitas Pengaturan permainan yang dapat dilihat pada gambar di bawah ini.



Gambar 3.14. *Activity Diagram* Pengaturan Permainan

9. *Activity Diagram* Mengatur Nama Pemain

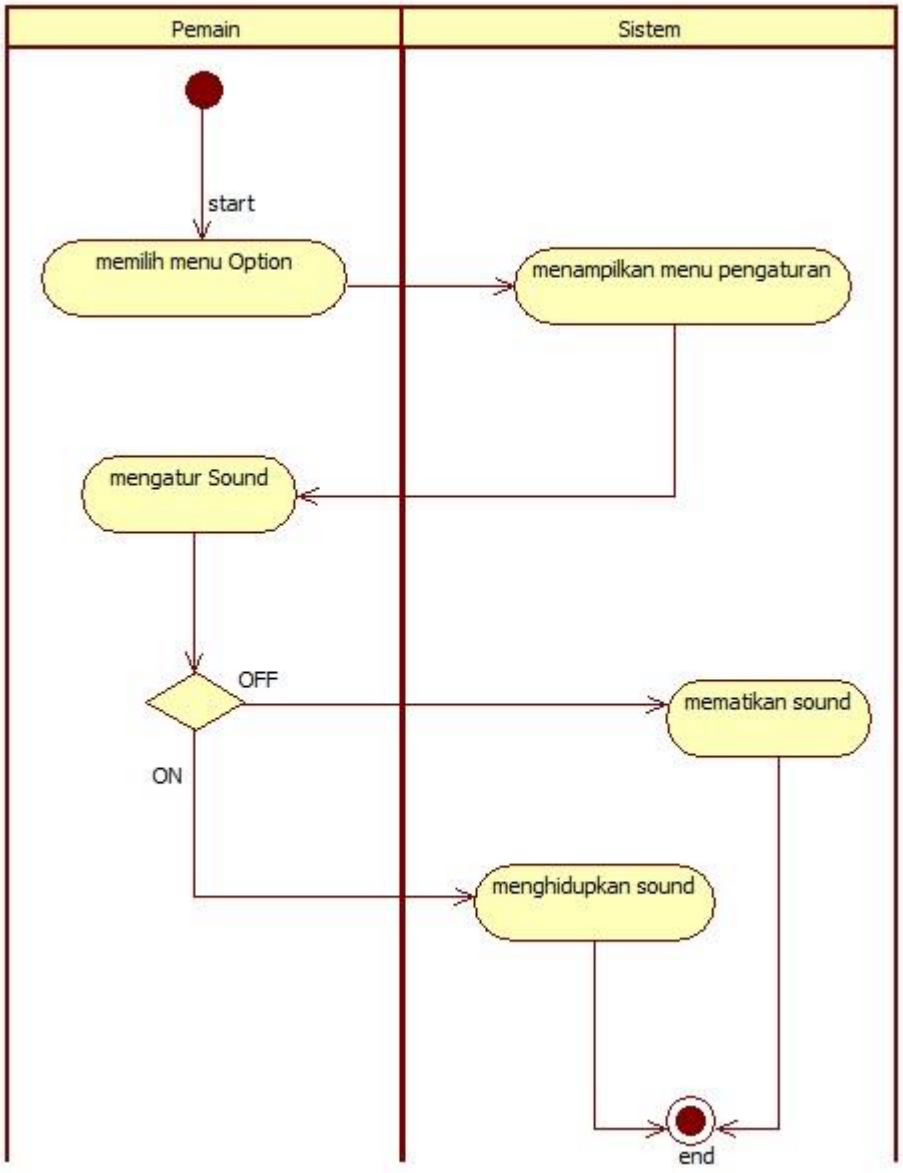
Berikut ini adalah diagram yang menunjukan alur aksi pada aktivitas Mengatur nama pemain yang dapat dilihat pada gambar di bawah ini.



Gambar 3.15. *Activity Diagram* Mengatur Nama Pemain

10. *Activity Diagram* Mengatur *Sound*

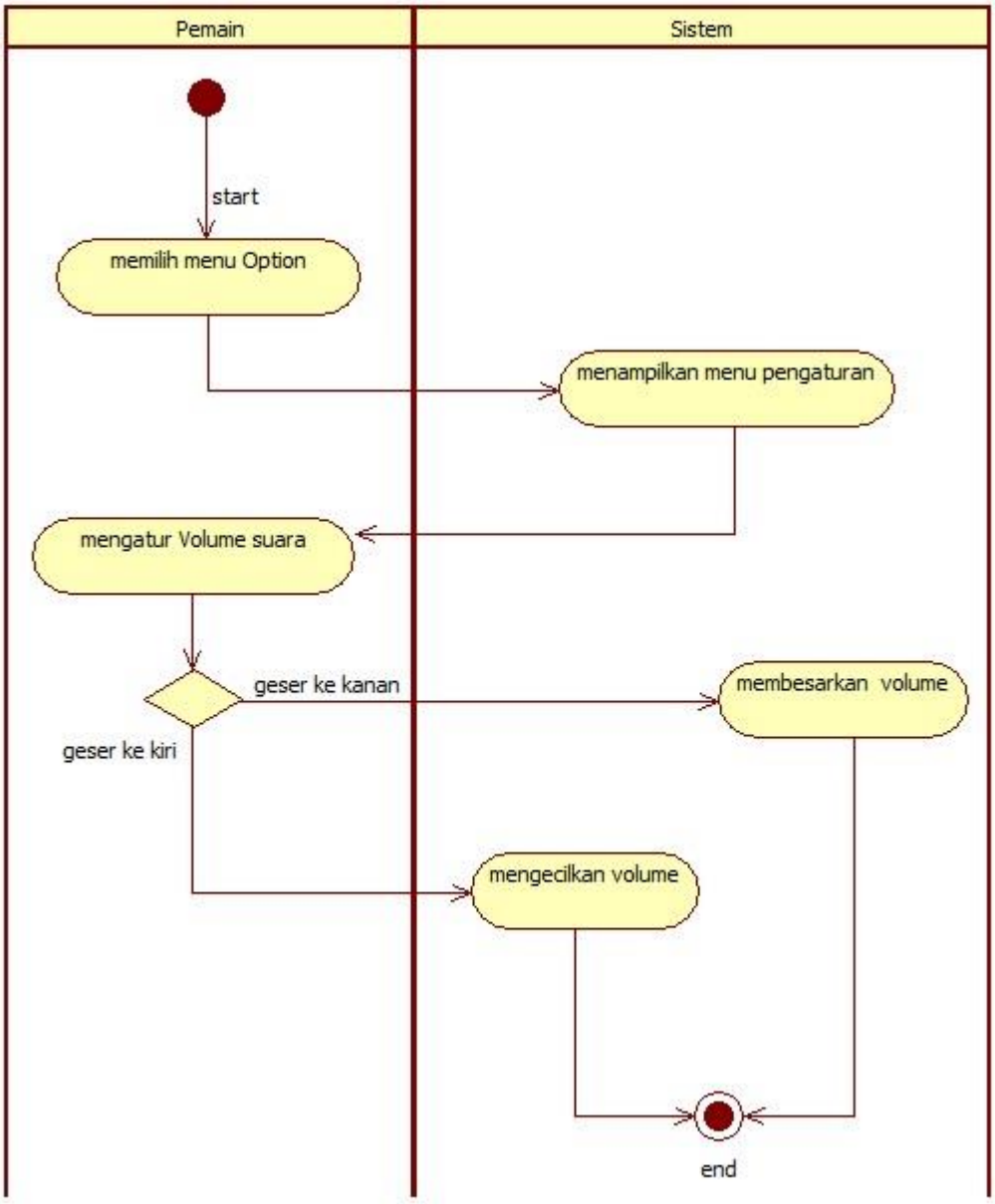
Berikut ini adalah diagram yang menunjukan alur aksi pada aktivitas Mengatur *Sound* yang dapat dilihat pada gambar berikut ini.



Gambar 3.16. *Activity Diagram* Mengatur *Sound*

11. *Activity Diagram* Mengatur *Volume*

Berikut ini adalah diagram yang menunjukan alur aksi pada aktivitas Mengatur *Volume* yang dapat dilihat pada gambar berikut ini.

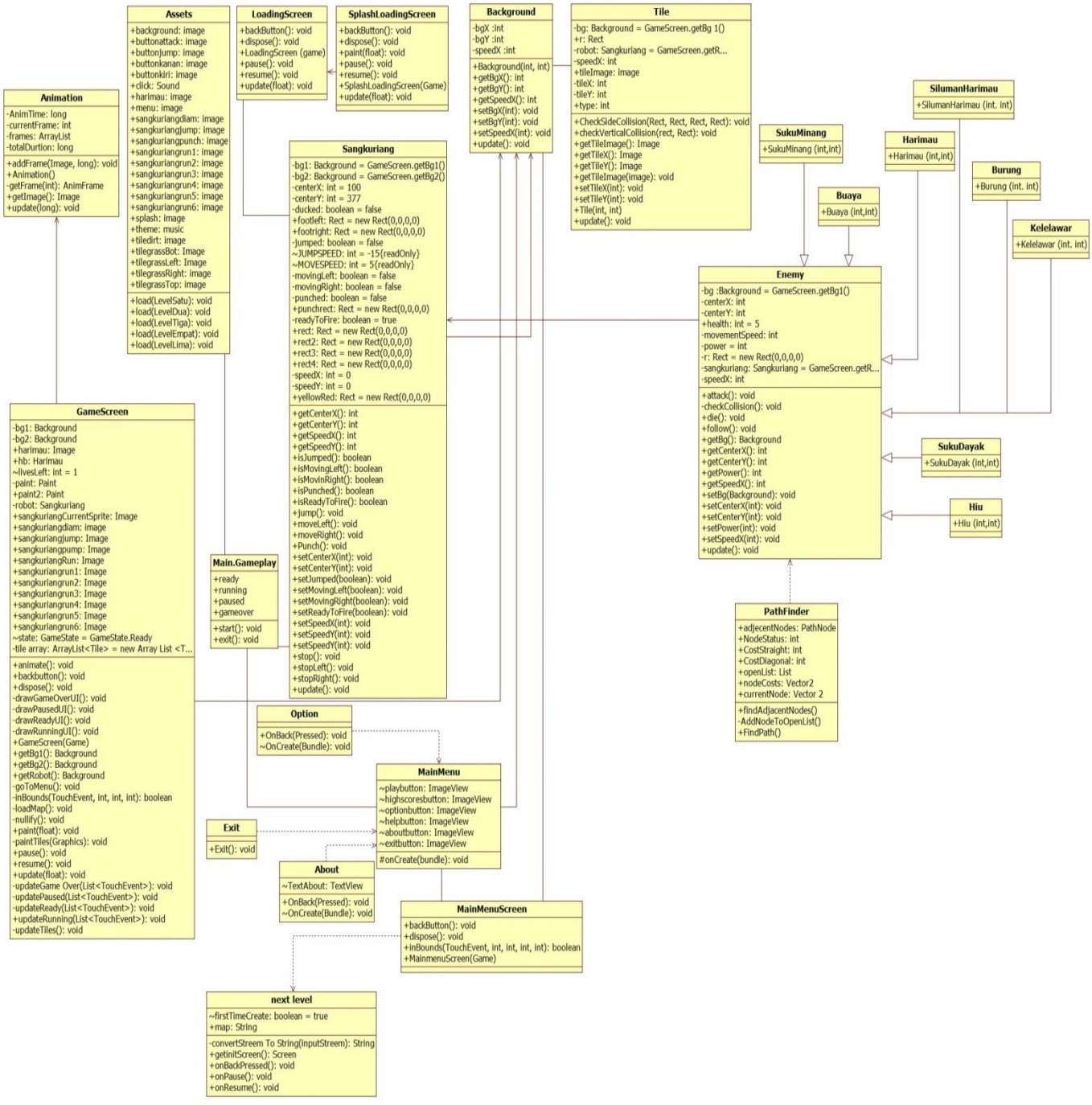


Gambar 3.17. *Activity Diagram* Mengatur Volume

3.1.6.3 Class Diagram

Class Diagram menggambarkan struktur dan hubungan antar objek-objek yang ada pada sistem. Struktur itu meliputi atribut dan *method* yang ada pada masing-masing *class*. Hubungannya dari masing-masing *class* yang ada tersebut

digambarkan dengan menggunakan pewarisan dan generalisasi seperti terlihat pada gambar berikut ini.



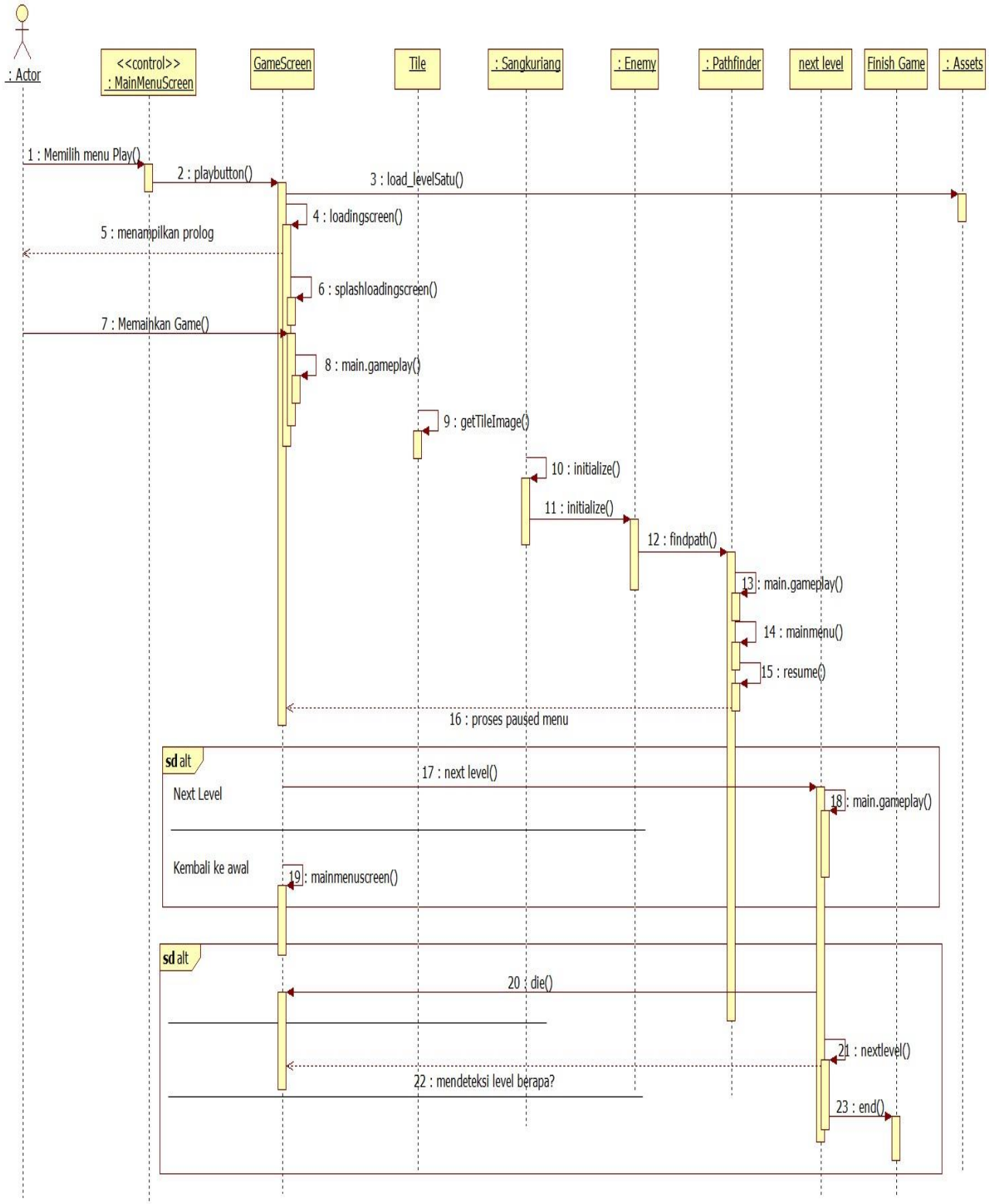
Gambar 3.18. ***Class Diagram Game Adventure "Sangkuriang and Indonesian Five Largest Islands"***

3.1.6.4 Sequence Diagram

Sequence Diagram menggambarkan interaksi antar masing-masing objek pada setiap *use case* dalam urutan waktu. Interaksi ini berupa pengiriman serangkaian data antar objek-objek yang saling berinteraksi. *Sequence Diagram* pada *game adventure "Sangkuriang and Indonesian Five Largest Islands"* terdiri dari *sequence diagram* menampilkan *score*, *sequence diagram* pengaturan permainan, *sequence diagram* petunjuk permainan, *sequence diagram* menampilkan *About* dan *sequence diagram* menutup aplikasi.. Berikut ini penjelasan dari masing-masing sequence diagram:

1. *Sequence Diagram* Menampilkan Level 1

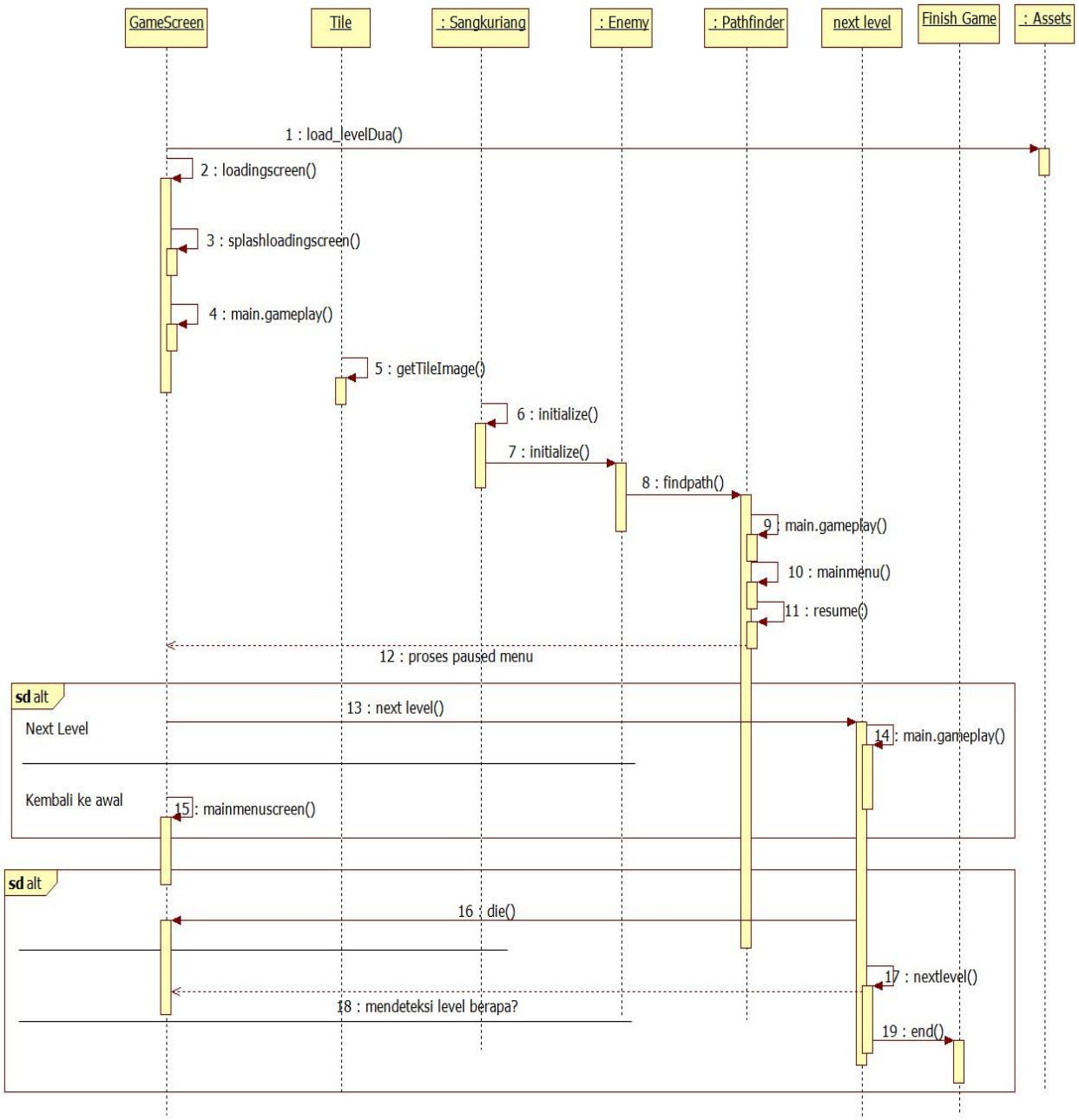
Diagram ini menjelaskan interaksi antar partisipan di dalam aktivitas menampilkan level 1 dalam urutan waktu. Untuk lebih jelasnya lihat gambar di bawah ini.



Gambar 3.19. *Sequence Diagram* Menampilkan Level 1

2. *Sequence Diagram* Menampilkan Level 2

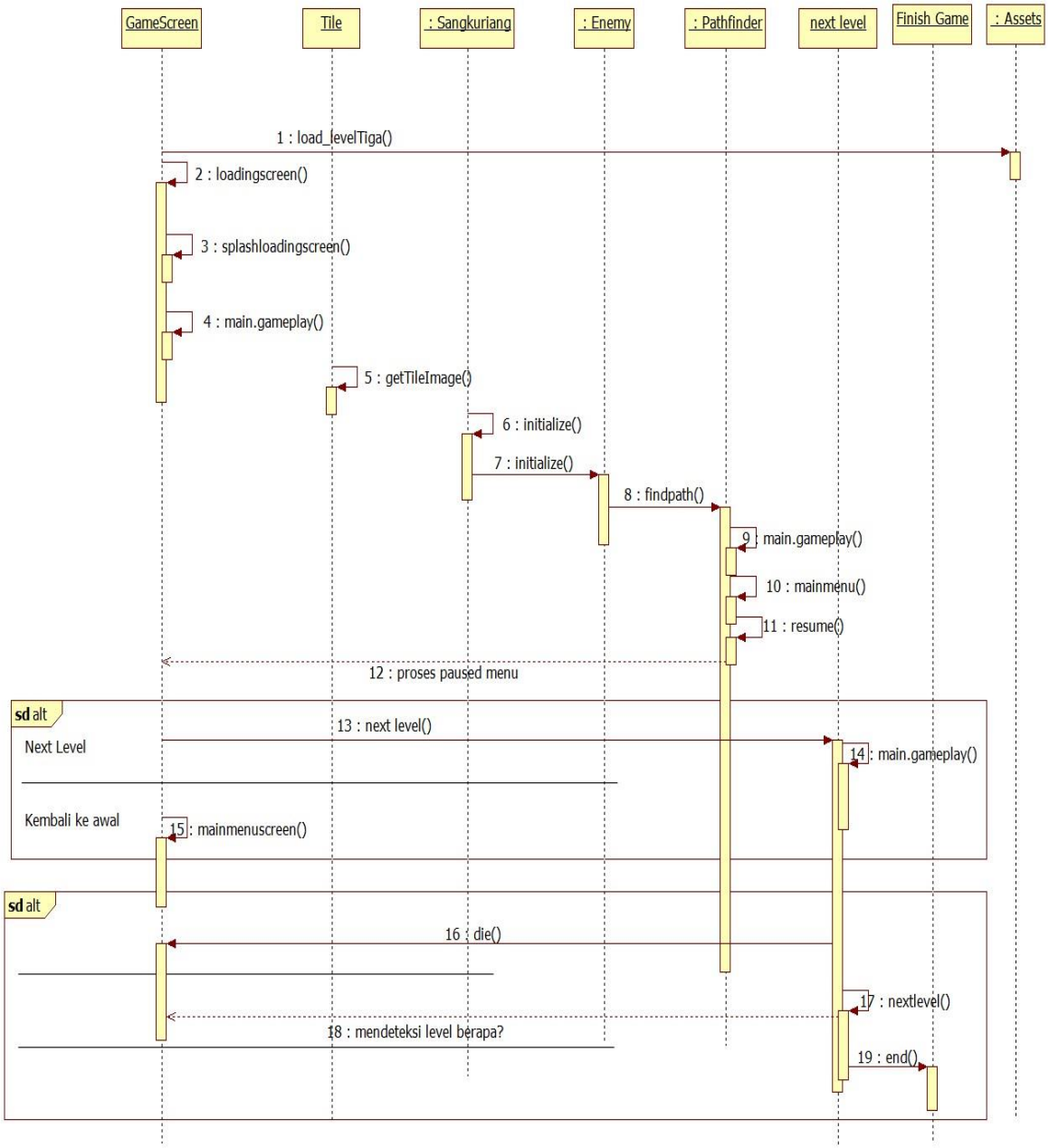
Diagram ini menjelaskan interaksi antar partisipan di dalam aktivitas menampilkan level 2 dalam urutan waktu. Untuk lebih jelasnya lihat gambar di bawah ini.



Gambar 3.20. *Sequence Diagram* Menampilkan Level 2

3. Sequence *Diagram* Menampilkan Level 3

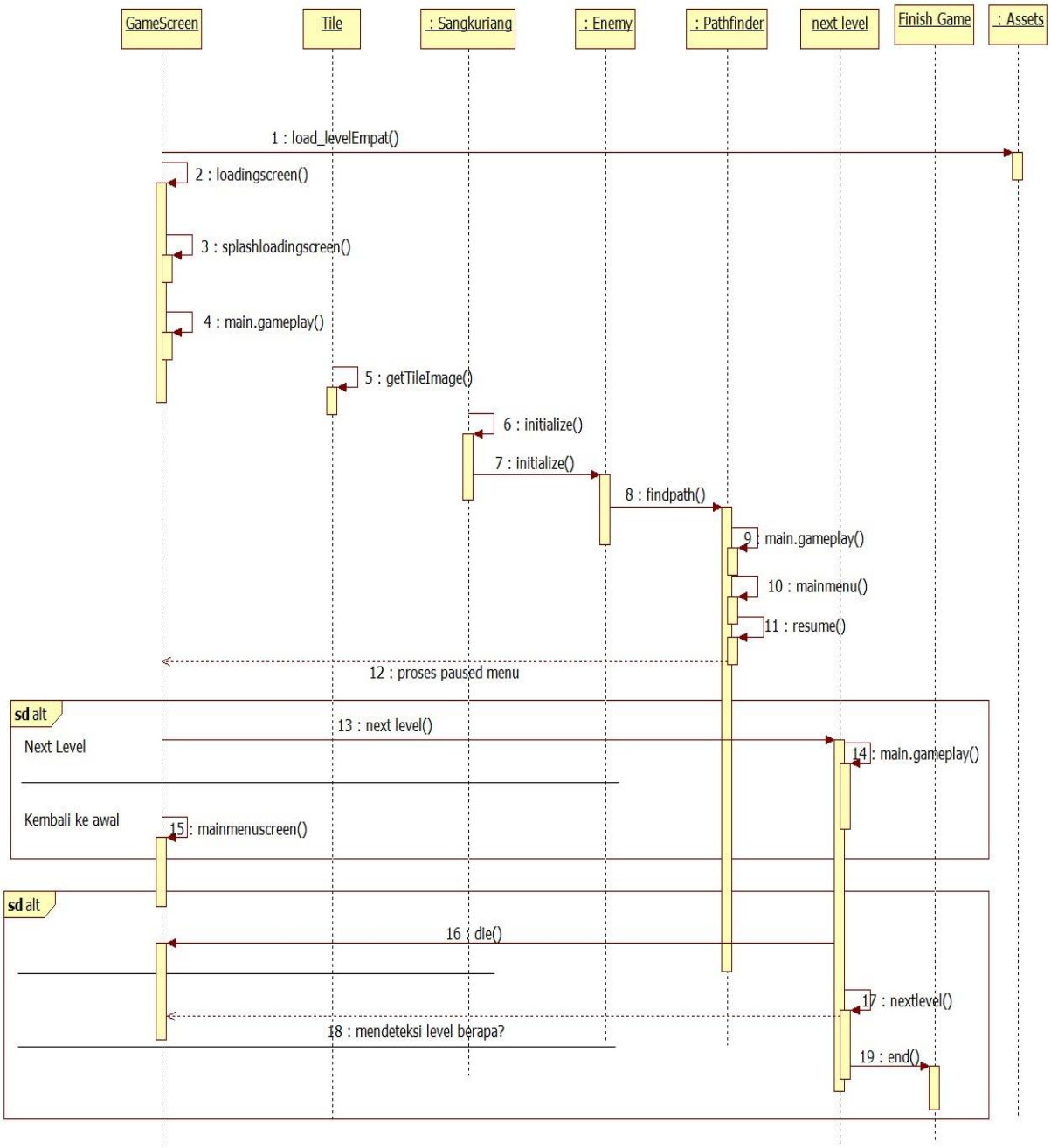
Diagram ini menjelaskan interaksi antar partisipan di dalam aktivitas menampilkan level 3 dalam urutan waktu. Untuk lebih jelasnya lihat gambar di bawah ini.



Gambar 3.21. *Sequence Diagram* Menampilkan Level 3

4. *Sequence Diagram* Menampilkan Level 4

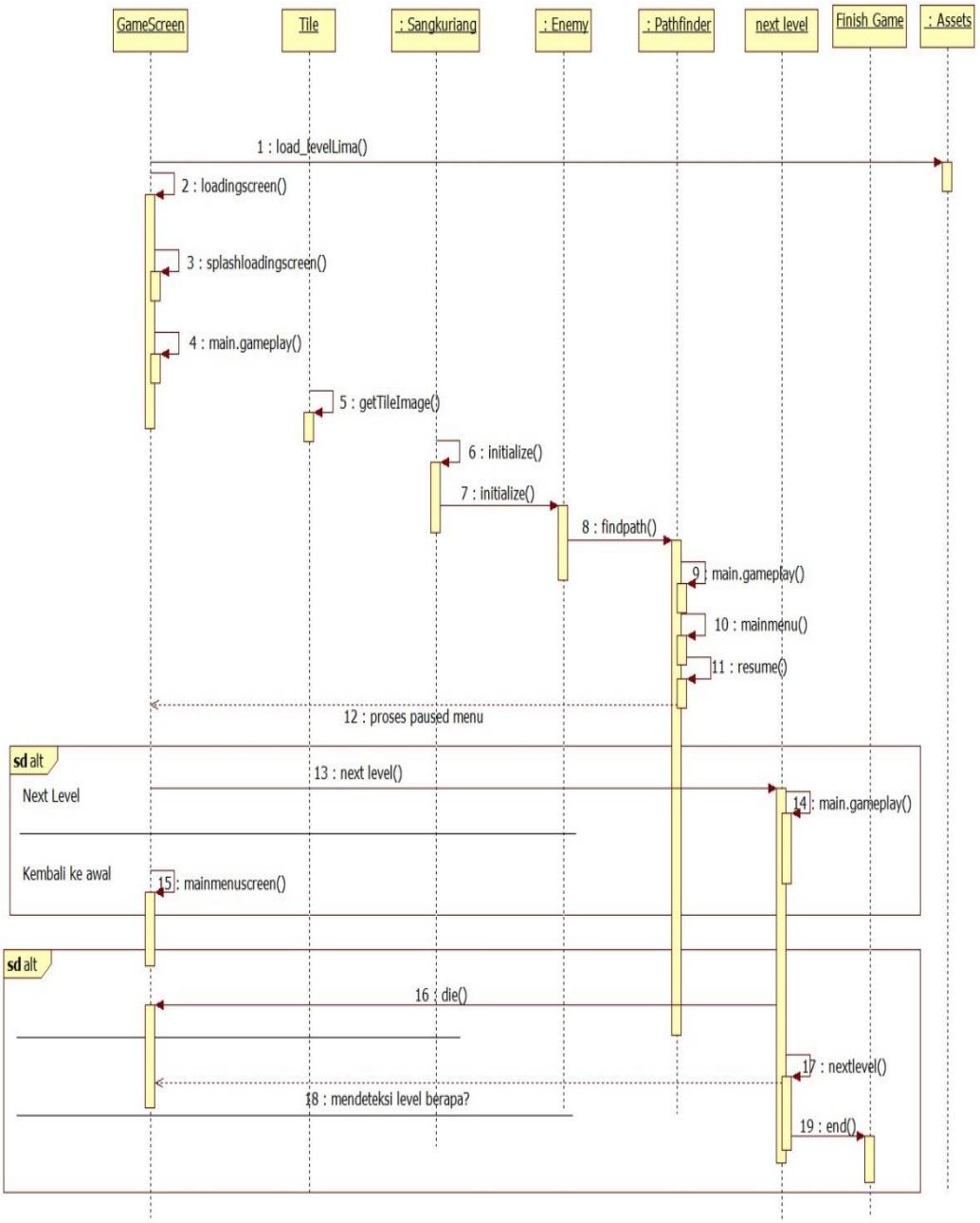
Diagram ini menjelaskan interaksi antar partisipan di dalam aktivitas menampilkan level 4 dalam urutan waktu. Untuk lebih jelasnya lihat gambar di bawah ini.



Gambar 3.22. *Sequence Diagram* Menampilkan Level 4

5. *Sequence Diagram* Menampilkan Level 5

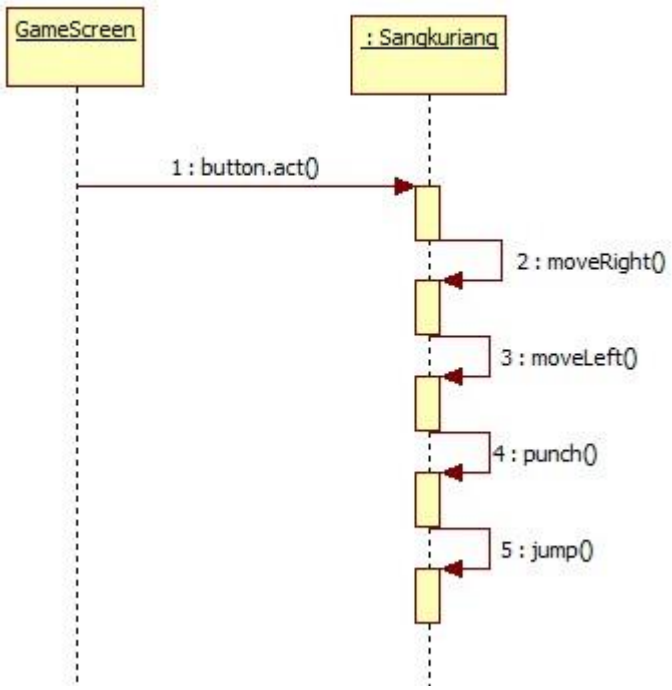
Diagram ini menjelaskan interaksi antar partisipan di dalam aktivitas menampilkan level 5 dalam urutan waktu. Untuk lebih jelasnya lihat gambar di bawah ini.



Gambar 3.23. *Sequence Diagram* Menampilkan Level 5

6. *Sequence Diagram* Mengendalikan Karakter Pemain

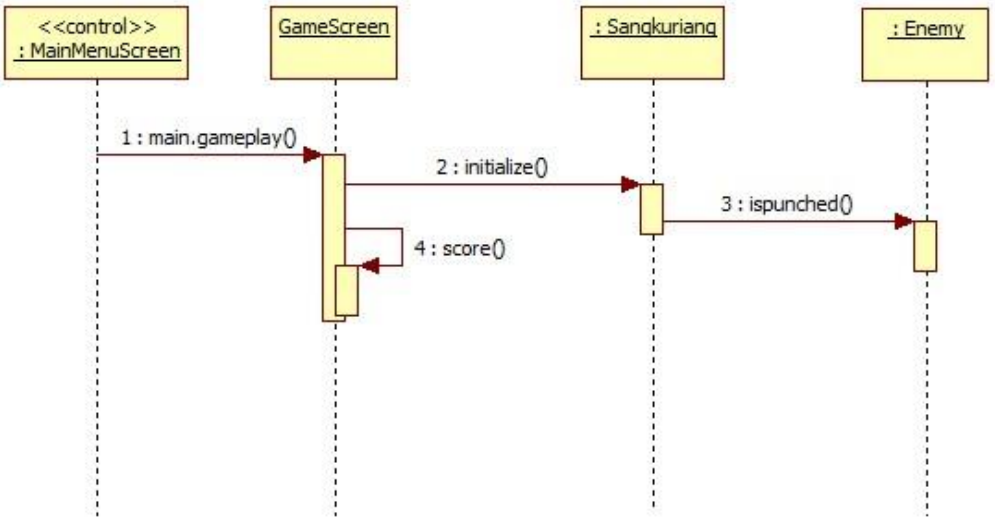
Diagram ini menjelaskan interaksi antar partisipan di dalam aktivitas mengendalikan karakter pemain dalam urutan waktu. Untuk lebih jelasnya lihat gambar di bawah ini.



Gambar 3.24. *Sequence Diagram* Mengendalikan Karakter Pemain

7. *Sequence Diagram* Perhitungan Skor

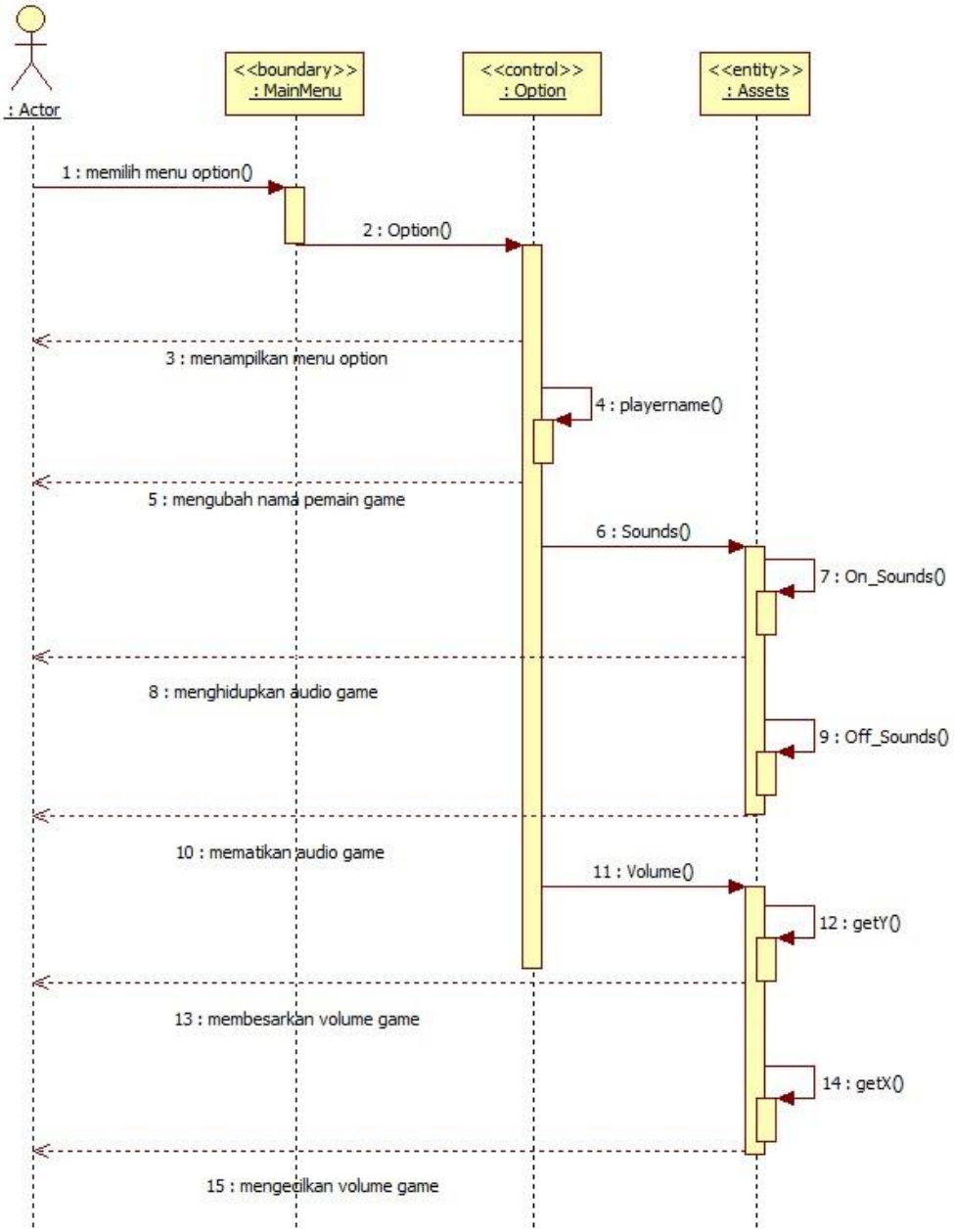
Diagram ini menjelaskan interaksi antar partisipan di dalam aktivitas perhitungan skor dalam urutan waktu. Untuk lebih jelasnya lihat gambar di bawah ini.



Gambar 3.25. *Sequence Diagram* Perhitungan Skor

8. *Sequence Diagram* Pengaturan permainan

Diagram ini menjelaskan interaksi antar partisipan di dalam aktivitas Pengaturan Permainan dalam urutan waktu. Untuk lebih jelasnya lihat gambar di bawah ini.

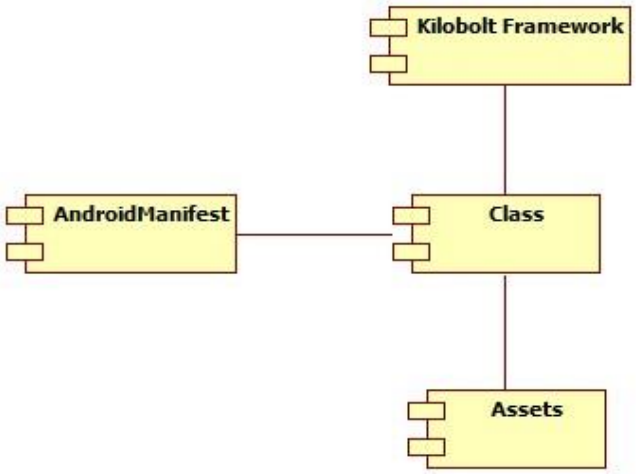


Gambar 3.26. *Sequence Diagram* Pengaturan permainan

3.1.6.5 *Component Diagram*

Component Diagram menggambarkan struktur dan hubungan antar komponen piranti lunak termasuk ketergantungan (*dependency*). Komponen piranti lunak yang yang dimaksud adalah modul yang berisi *source code* atau

binary code, yang ada *library atau excutetable* yang muncul pada compile time, link time ataupun pada runtime. Component Diagram pada *game* ini bisa dilihat pada gambar berikut ini.



Gambar 3.27. *Component Diagram*

3.2 Perancangan Sistem

Perancangan sistem merupakan kegiatan untuk merancang aplikasi yang akan di bangun, tahapannya dimulai dari perancangan komponen permainan, perancangan antarmuka, perancangan pesan dan jaringan semantik dari *game adventure* "Sangkuriang *and Indonesian Five Largest Island*s".

3.2.1 Perancangan Komponen Permainan

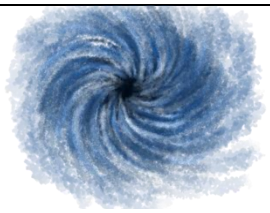
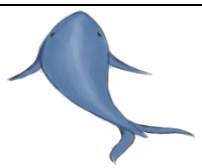
Di dalam bagian ini akan dijelaskan karakter, pergerakan karakter, dan item yang terdapat pada *game* dan *adventure* "Sangkuriang *and Indonesian Five Largest Island*s".

3.2.1.1 Karakter *Game*

Di dalam bagian ini akan dijelaskan karakter-karakter yang terdapat pada *game* ini seperti yang bisa dilihat pada tabel berikut ini.

Tabel 3.20. Karakter *Game*

No	Karakter	Keterangan
1		Nama karakter : **Sangkuriang** merupakan karakter utama yang berpetualang mengelilingi 5 pulau besar Indonesia.
2		Nama karakter : **Harimau Sumatera** sebagai musuh Sangkuriang di *level* 1 (Pulau Sumatera).
3		Nama karakter : **Burung Endemik** sebagai musuh Sangkuriang di *level* 1 (Pulau Sumatera).
4		Nama karakter : **Ahli Silek (Kepala Suku Minangkabau)** sebagai musuh utama Sangkuriang di *level* 1 (Pulau Sumatera).
5		Nama karakter : **Buaya** sebagai musuh Sangkuriang di *level* 2 (Pulau Kalimantan).

		
6		Nama karakter : **Kepala Suku Dayak** sebagai musuh utama Sangkuriang di *level* 2 (Pulau Kalimantan).
7		Nama karakter : **Pusaran air** sebagai musuh Sangkuriang di *level* 3 (Pulau Sulawesi).
8		Nama karakter : **Terumbu Karang** sebagai musuh Sangkuriang di *level* 3 (Pulau Sulawesi).
9		Nama karakter : **Paus** sebagai musuh Sangkuriang di *level* 3 (Pulau Sulawesi).
10		Nama karakter : **Kelelawar** sebagai musuh Sangkuriang di *level* 4 (Pulau Papua).
11		Nama karakter : **Burung-burungan** sebagai musuh Sangkuriang di *level* 4 (Pulau Papua).

12		Nama karakter : **Siluman Harimau** sebagai musuh Sangkuriang di *level* 5 (Pulau Jawa).

3.2.1.2 *Item Game*

Item pendukung yang digunakan pada aplikasi *game* "Sangkurang *and Indonesian Five Largest Islands*". Berikut ini tabel deskripsi item pada aplikasi *game* "Sangkurang *and Indonesian Five Largest Islands*".

Tabel 3.21. Item *Game*

No	*Item*	Keterangan
1		Nama *item*: Kerambit yang merupakan alat tradisional dari Pulau Sumatera, sebagai misi yang harus dicapai untuk menyelesaikan *level* 1.
2		Nama *item*: Mandau dan Talawang yang merupakan alat tradisional dari Pulau Kalimantan, sebagai misi yang harus dicapai untuk menyelesaikan *level* 2.
3		Nama *item*: Kawali yang merupakan alat tradisional dari Pulau Sulawesi, sebagai misi yang harus dicapai untuk menyelesaikan *level* 3.

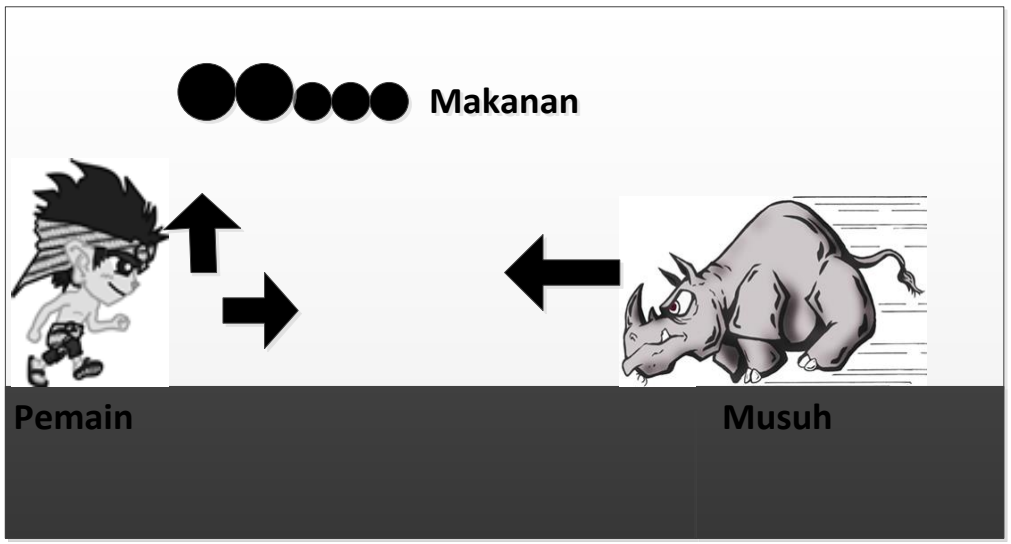
4		Nama *item*: Panah dan Busur yang merupakan alat tradisional dari Pulau Papua, sebagai misi yang harus dicapai untuk menyelesaikan *level* 4.
5		Nama *item*: Kujang yang merupakan alat tradisional dari Pulau Jawa, sebagai misi yang harus dicapai untuk menyelesaikan *level* 5.

3.2.1.3 *Storyboard*

Storyboard adalah sketsa gambar yang disusun berurutan sesuai dengan naskah, dengan storyboard ide cerita dapat disampaikan kepada orang lain dengan lebih mudah, karena dapat menggiring khayalan seseorang mengikuti gambar-gambar yang tersaji sehingga menghasilkan persepsi yang sama. Berikut merupakan story board dari *game adventure* "*Sangkuriang and Indonesian Five Largest Islands"*:

1. *Storyboard level* 1

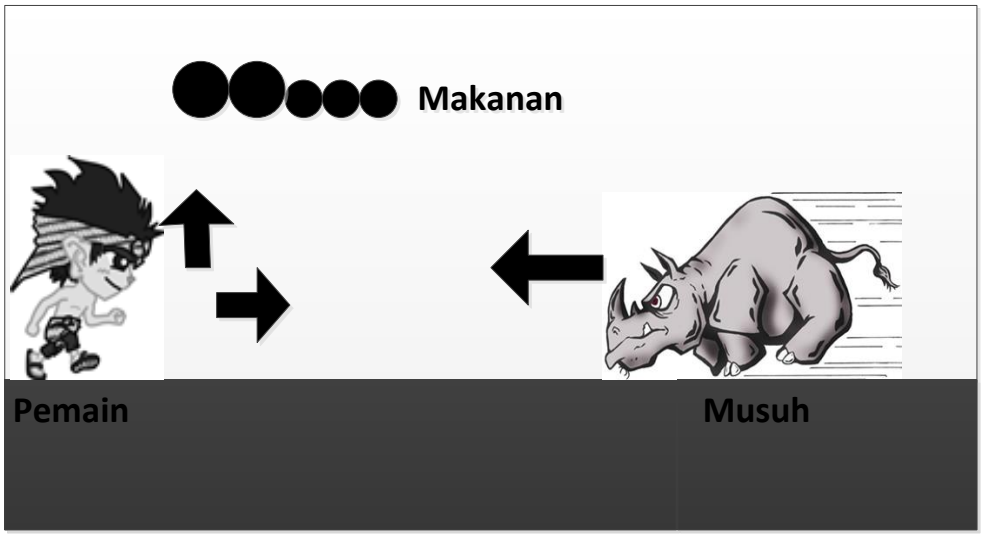
Story board untuk *level* 1 pada aplikasi ini dapat dilihat sebagai berikut :



Gambar 3.28. ***Storyboard level*** **1**

2. ***Storyboard level 2***

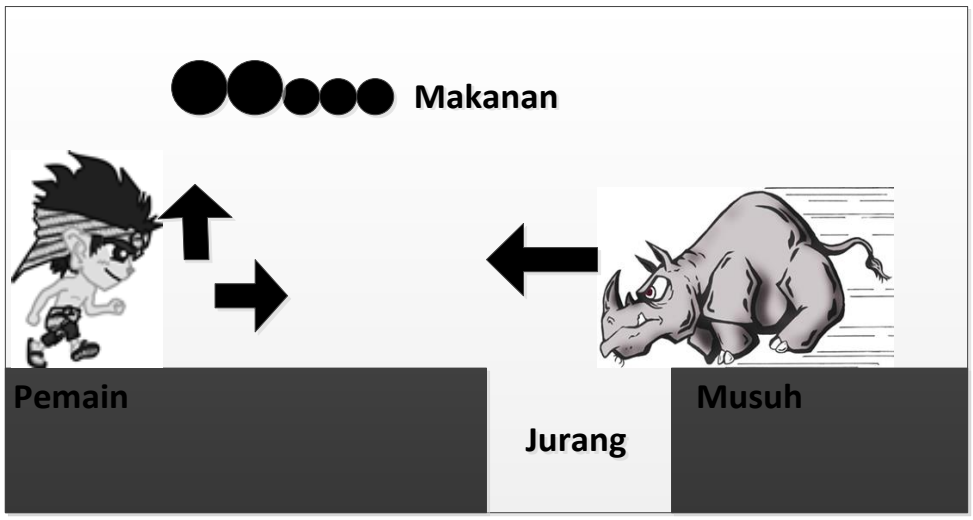
Story board untuk level 2 pada aplikasi ini dapat dilihat sebagai berikut :



Gambar 3.29. ***Storyboard level*** **2**

3. *Storyboard level* 3

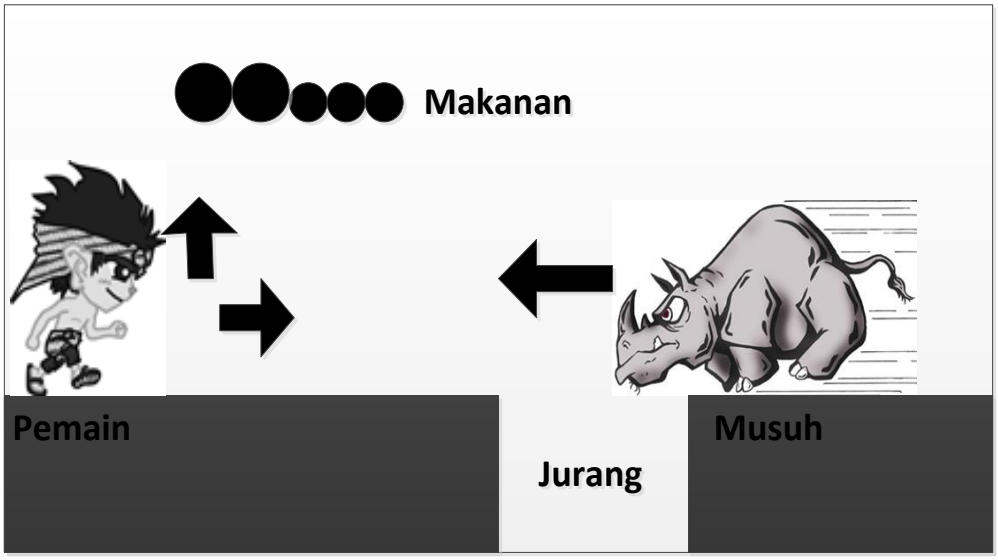
Story board untuk *level* 3 pada aplikasi ini dapat dilihat sebagai berikut :



Gambar 3.30. *Storyboard level* 3

4. *Storyboard level* 4

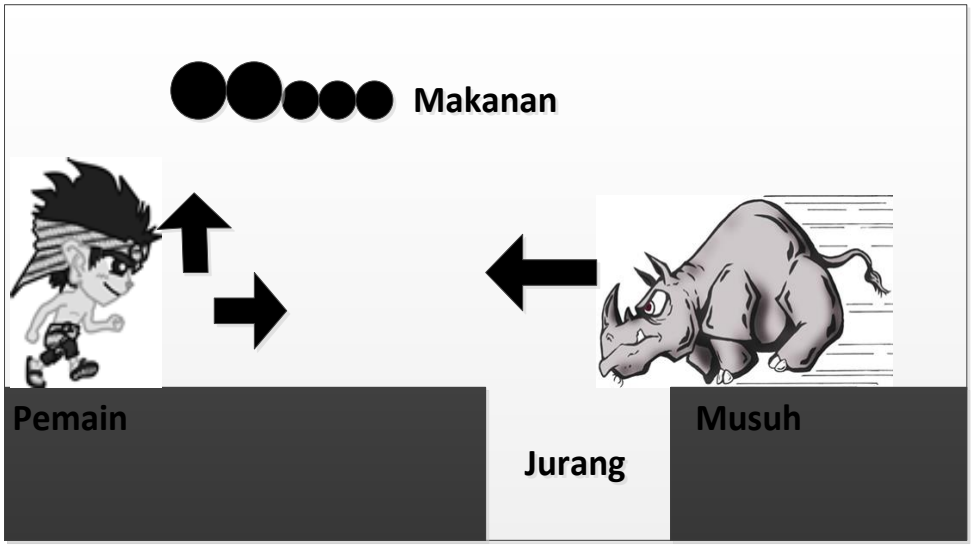
Story board untuk *level* 4 pada aplikasi ini dapat dilihat sebagai berikut :



Gambar 3.31. *Storyboard level* 4

5. *Storyboard level* 5

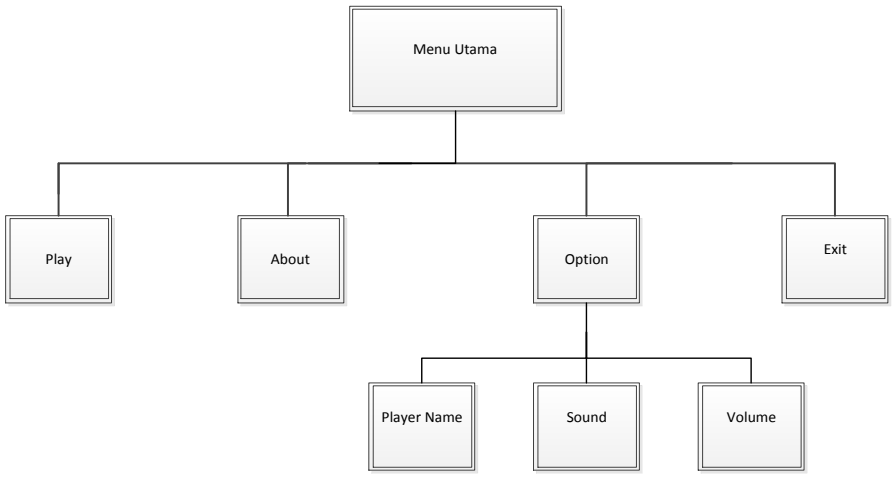
Story board untuk *level* 5 pada aplikasi ini dapat dilihat sebagai berikut :



Gambar 3.32. *Storyboard level* 5

3.2.2 Perancangan Struktur Menu

Perancangan struktur menu pada *game adventure "*Sangkuriang *and five largest islands"* pada gambar berikut :



Gambar 3.33. Perancangan Struktur Menu Utama

Keterangan:

1. Menu *Play* : merupakan submenu untuk memulai permainan baru
2. Menu *Option* : merupakan submenu yang menampilkan menu pengaturan
3. Menu *Player Name* : merupakan submenu untuk mengubah nama pemain
4. Menu *Volum*e : merupakan submenu untuk mengubah volume suara
5. Menu *Sound* : merupakan submenu yang mengubah sound ON/OFF
6. Menu *About* : merupakan submenu yang menampilkan profil tentang pembuat *game*
7. Menu *Exit* : merupakan submenu untuk keluar dari aplikasi

3.2.3 Perancangan Antarmuka

Perancangan antarmuka bertujuan untuk memberikan gambaran tentang aplikasi yang akan dibangun, sehingga akan mempermudah dalam mengimplementasikan aplikasi serta akan memudahkan pembuatan aplikasi.

Pada bagian ini akan dijelaskan perancangan antarmuka menu utama, perancangan antarmuka *menu play*, perancangan antarmuka menu *highscores*, perancangan antarmuka menu *option*, perancangan antarmuka menu *help*, perancangan antarmuka menu *About* dan perancangan antarmuka menu *exit.*

3.2.3.1 Perancangan Antarmuka Menu Utama

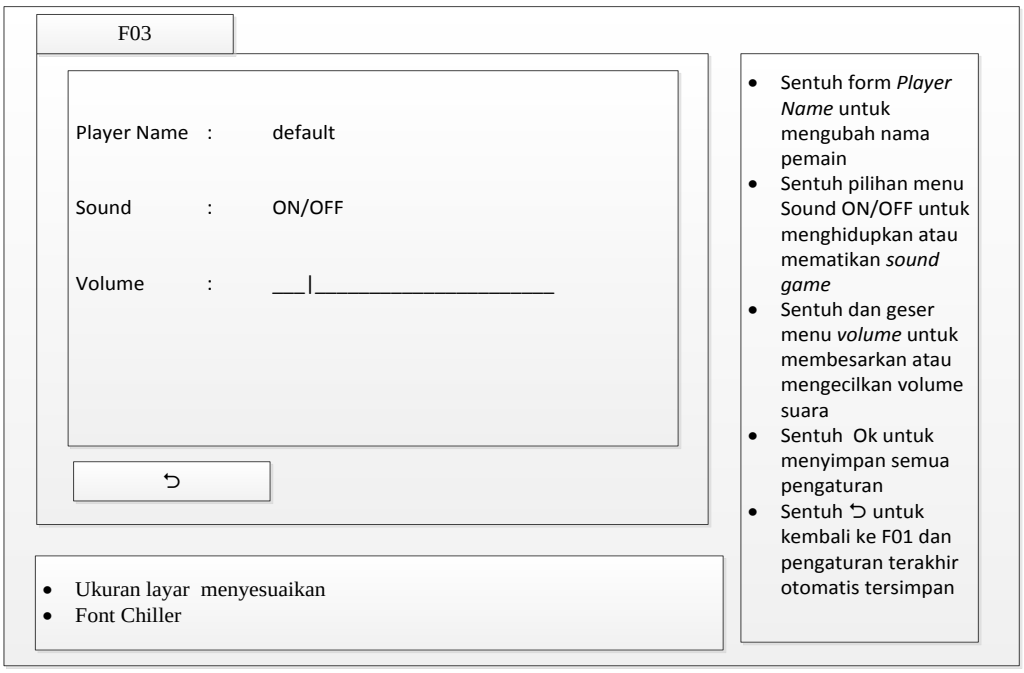
Menu utama merupakan menu yang akan muncul pertama kali pada saat pengguna membuka *game*. Gambar di bawah ini menunjukan perancangan antarmuka Menu Utama.



Gambar 3.34. Perancangan Antarmuka Menu Utama

3.2.3.2 Perancangan Antarmuka *Option*

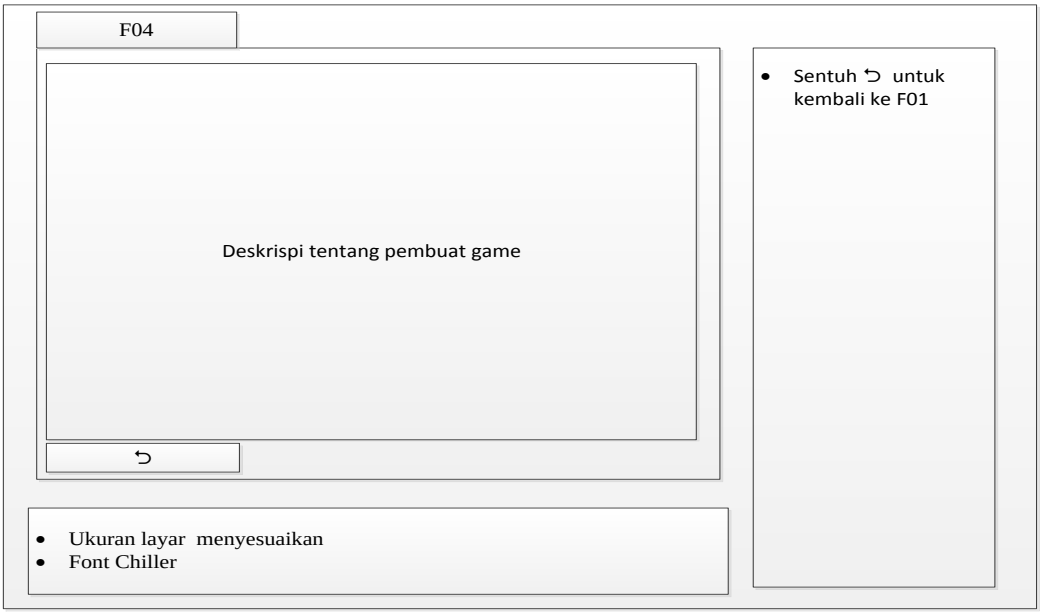
Menu *option* merupakan menu yang menampilkan pengaturan *game*, seperti pengaturan *Play*er Name, pengaturan Sound dan pengaturan Volume suara. Gambar di bawah ini menunjukan perancangan antarmuka Menu *Option*.



Gambar 3.35. Perancangan Antarmuka *Option*

3.2.3.3 Perancangan Antarmuka *About*

Menu *About* merupakan menu yang menampilkan profil penulis. Gambar di bawah ini menunjukan perancangan antarmuka Menu *About.*



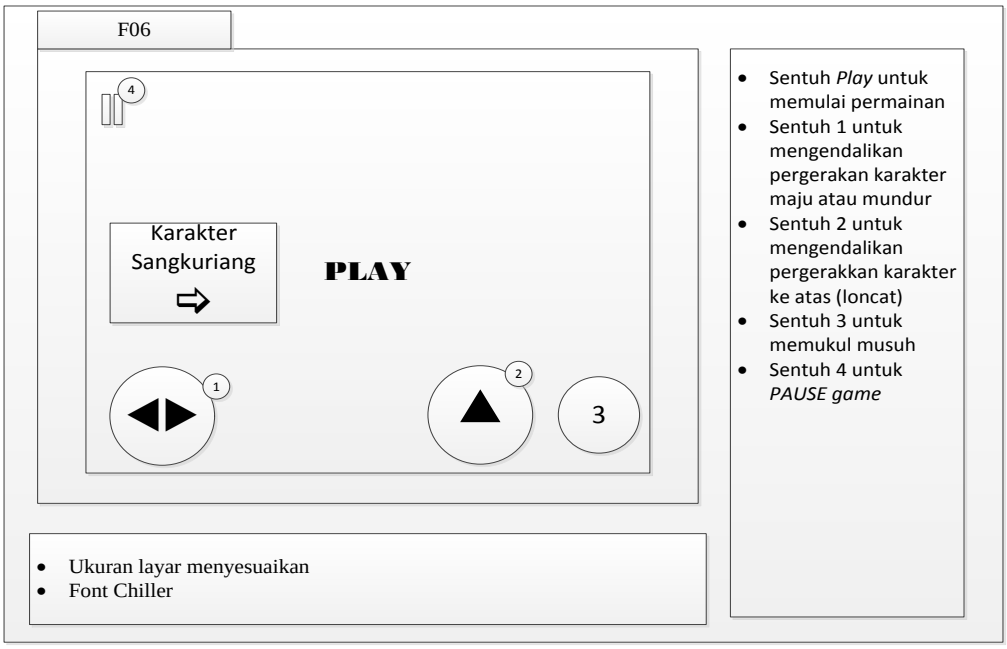
Gambar 3.36. Perancangan Antarmuka *Option*

3.2.3.4 Perancangan Antarmuka Menu *Play*

Antarmuka Menu *Play* merupakan perancangan antarmuka yang menampilkan permainan *game adventure* "Sangkuriang *and Indonesian Five Largest Island*". Berikut ini perancangan antarmuka menu *play* yang terdiri dari *level* 1, *level* 2, *level* 3, *level* 4 dan *level* 5.

3.2.3.5 Perancangan Antarmuka Menu *Play Level* 1

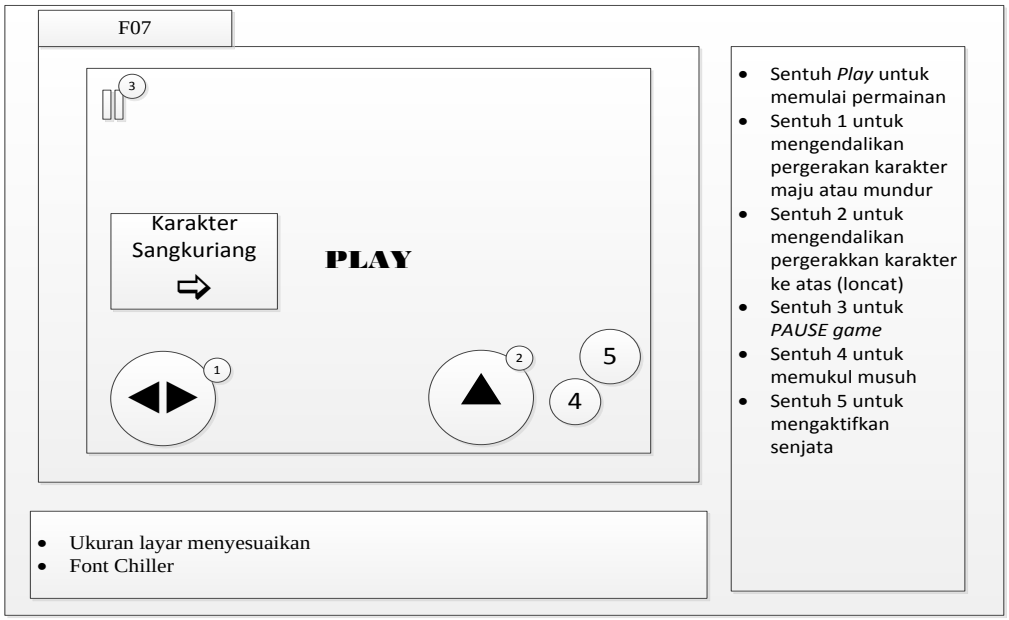
Berikut ini perancangan antarmuka *game adventure "Sangkuriang and Indonesian five largest islands"* di *level* 1, yaitu pada gambar di bawah ini.



Gambar 3.37. Perancangan Antarmuka *Play* Level 1

1. Perancangan Antarmuka Menu *Play Level* 2

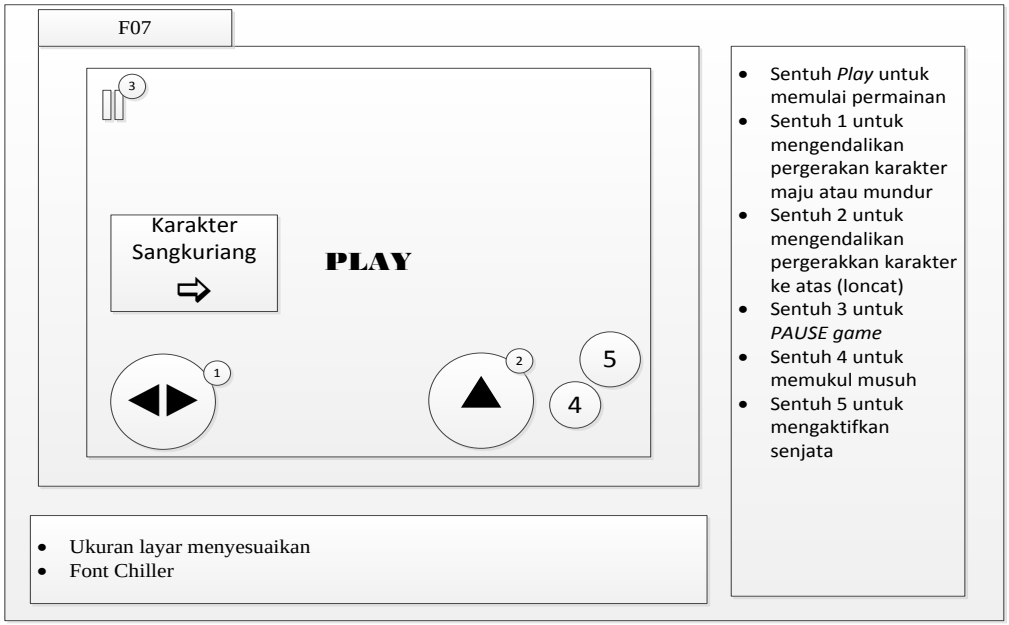
Berikut ini perancangan antarmuka *game adventure "Sangkuriang and Indonesian five largest islands"* di *level* 2, yaitu pada gambar di bawah ini.



Gambar 3.38. Perancangan Antarmuka *Play* Level 2

2. Perancangan Antarmuka Menu *Play Level* 3

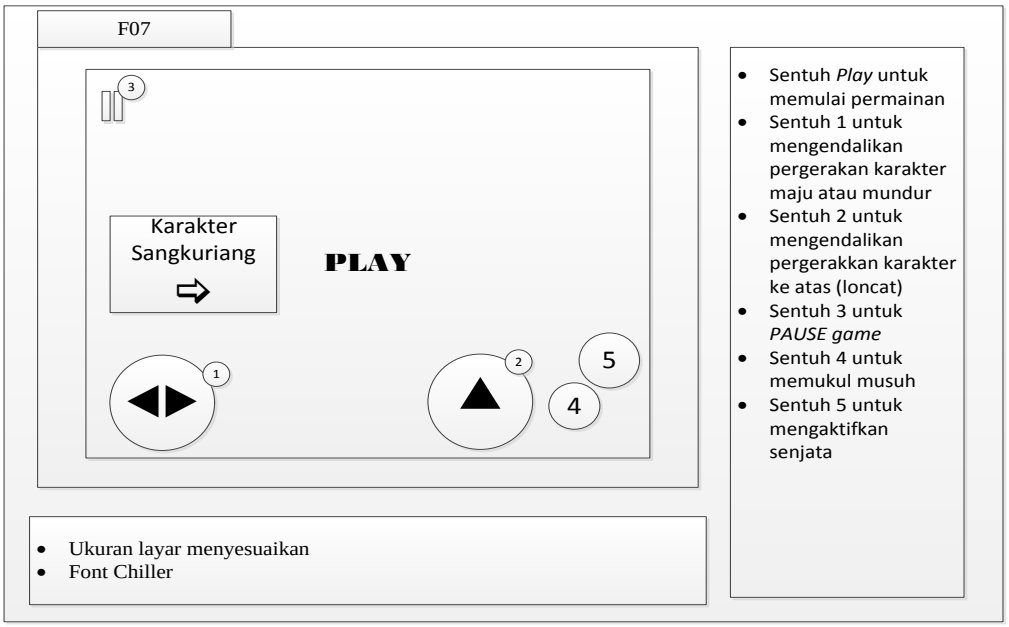
Berikut ini perancangan antarmuka *game adventure "Sangkuriang and Indonesian five largest islands"* di *level* 3, yaitu pada gambar di bawah ini.



Gambar 3.39. Perancangan Antarmuka *Play Level* 3

3. Perancangan Antarmuka Menu *Play Level* 4

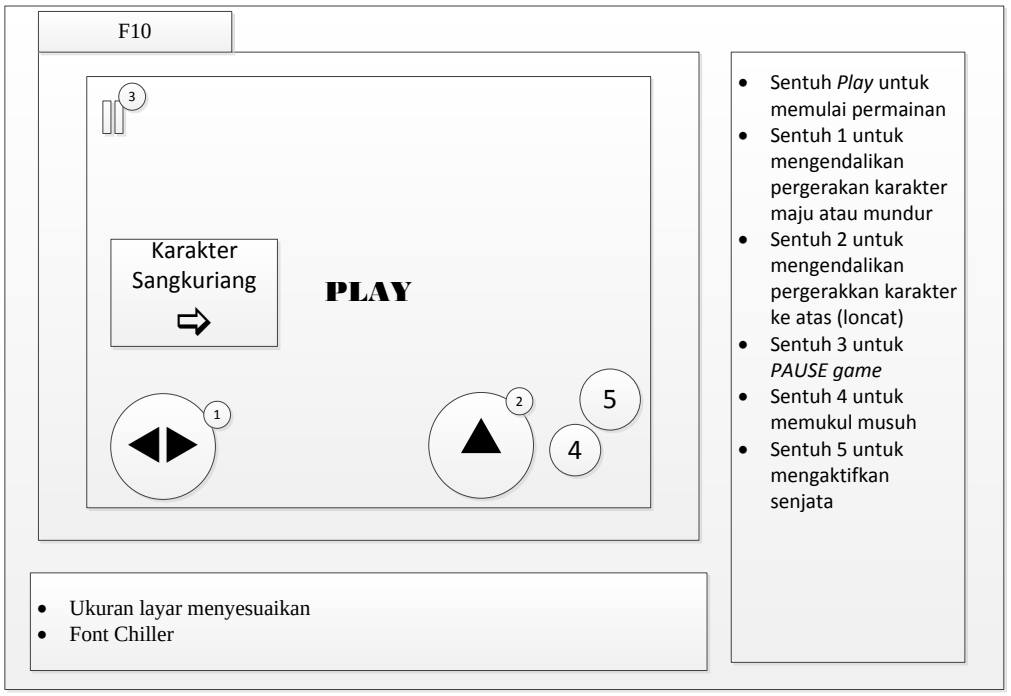
Berikut ini perancangan antarmuka *game adventure "Sangkuriang and Indonesian five largest islands"* di *level* 4, yaitu pada gambar di bawah ini.



Gambar 3.40. Perancangan Antarmuka *Play Level* 4

4. Perancangan Antarmuka Menu *Play Level* 5

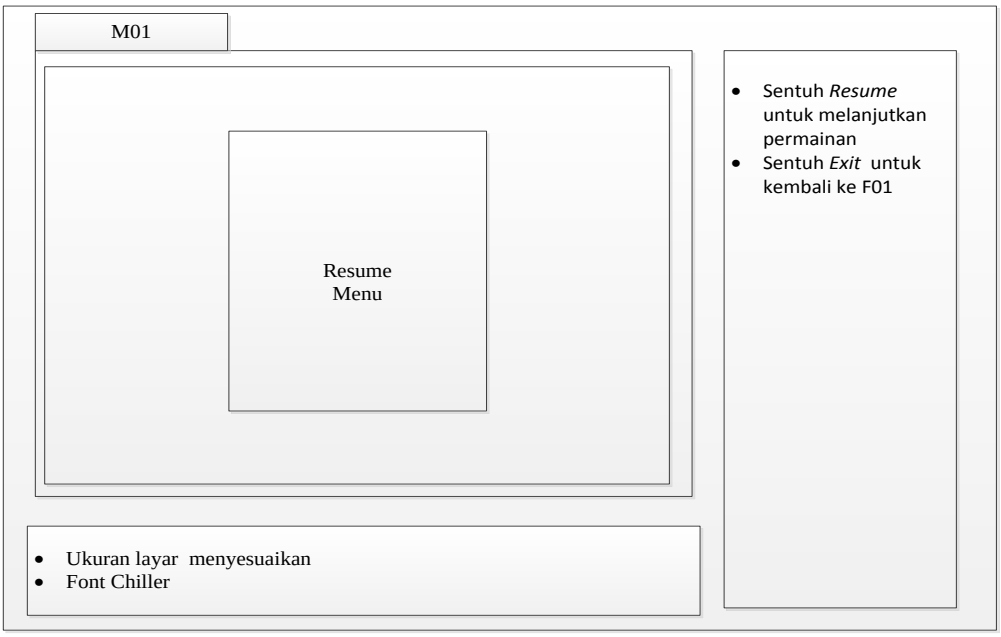
Berikut ini perancangan antarmuka *game adventure "Sangkuriang and Indonesian five largest islands"* di *level* 5, yaitu pada gambar di bawah ini..



Gambar 3.41. Perancangan Antarmuka *Play Level* 5

3.2.4 Perancangan Pesan

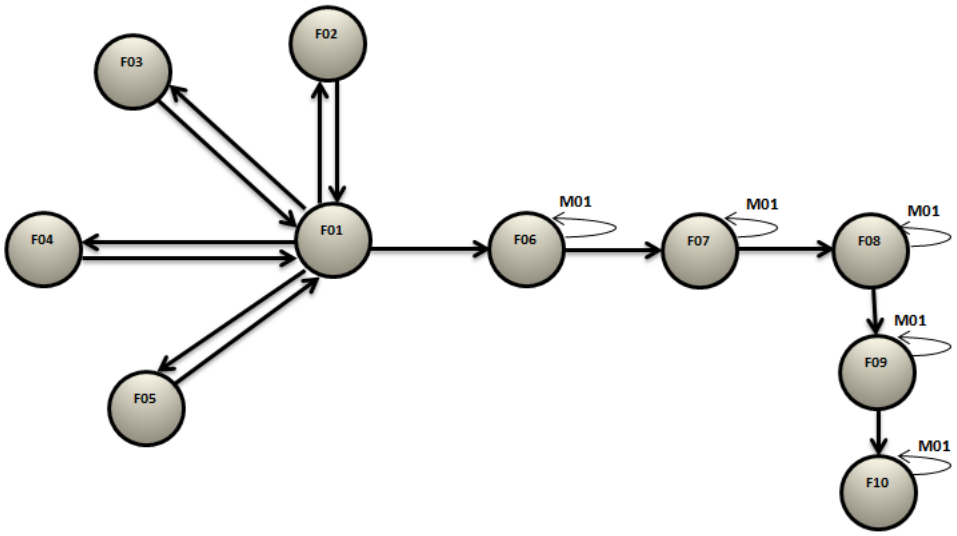
Pada bagian ini akan dijelaskan perancangan pesan yang terdapat pada *game adventure "*Sangkuriang *and five largest islands"* yaitu perancangan pesan pa*use* menu. Pesan Pa*use* Menu merupakan pesan yang muncul ketika pengguna menampilkan pa*use* menu. Untuk lebih jelasnya dapat dilihat pada gambar gambar di bawah ini.



Gambar 3.42. Perancangan Pesan *Pause Menu*

3.2.5 Jaringan Semantik

Untuk mengetahui keterhubungan dari setiap tampilan yang dibuat dapat menggunakan jaringan semantik. Gambar di bawah ini adalah jaringan semantik pada *game adventure "*Sangkuriang *and Indonesian five largest islands"*.



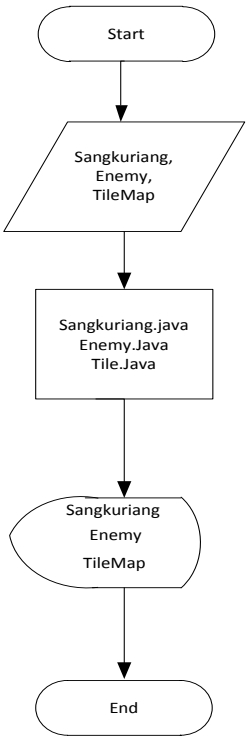
Gambar 3.43. Jaringan Semantik

3.2.6 Perancangan Metode

Perancangan metode merupakan perancangan yang dilakukan untuk menetapkan detail algoritma yang akan dinyatakan ke dalam suatu program. Perancangan metode menjelaskan tentang bagaimana program aplikasi berjalan.

1. Method StartGame()

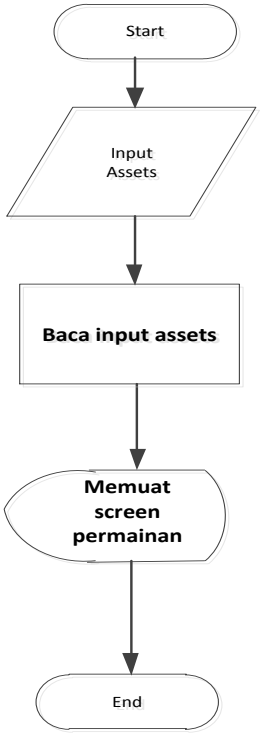
Method ini dipanggil ketika menu *play* dipilih. *Method* ini akan menginisialisasi posisi awal dari pemain. Peracangan metode ini bisa dilihat pada gambar berikut.



Gambar 3.44. *Method* Start *Game*()

2. Method LoadScreen()

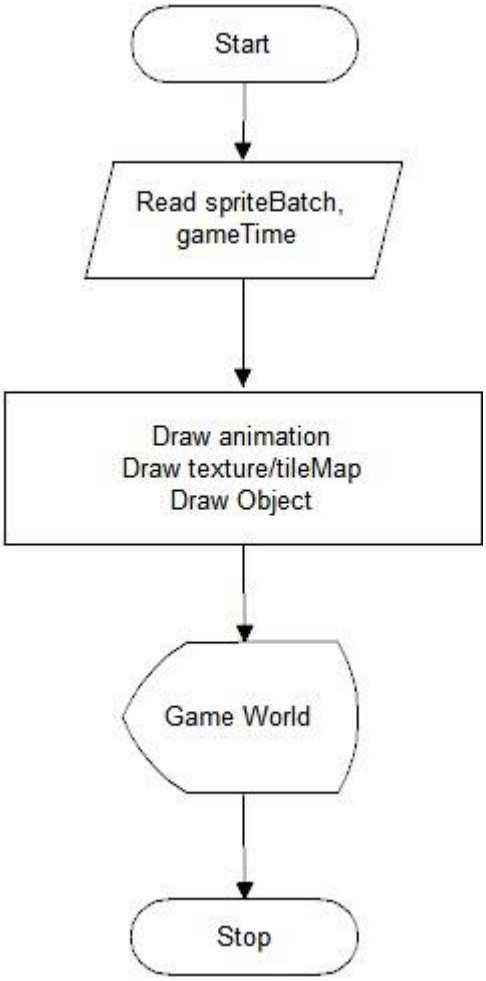
Method ini akan mengambil source yang dibutuhkan dalam *game* ini, seperti gambar dari karakter pemain, musuh dan arena permainan. Perancangan *method* ini bisa dilihat pada gambar berikut.



Gambar 3.45. Perancangan *method LoadScreen()*

3. *Method Draw()*

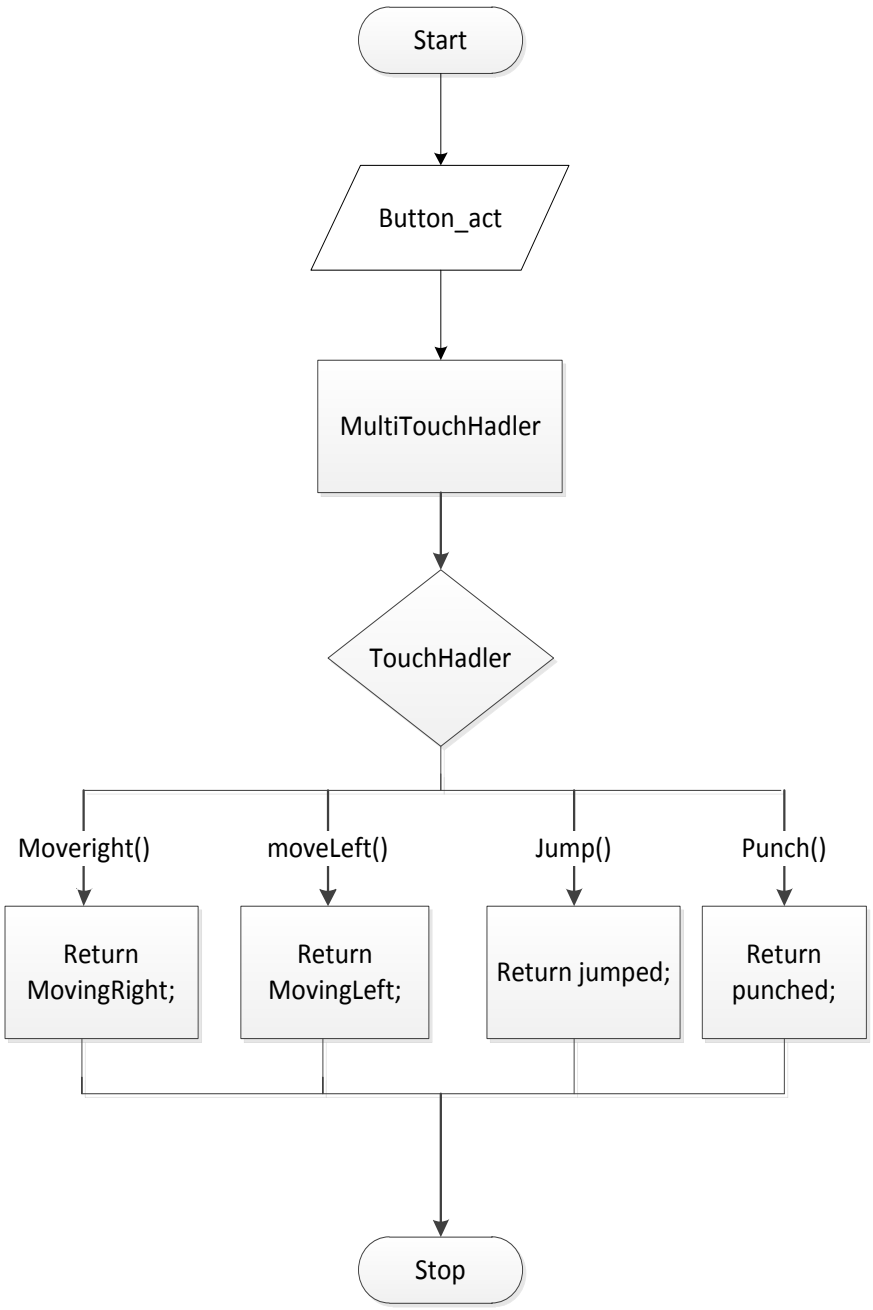
Method ini adalah *game* loop yang digunakan untuk memperbaharui tampilan *game* seperti animasi karakter, musuh, dan dunia permainan.



Gambar 3.46. Perancangan *Method Draw()*

4. Method SetMoving()

Method ini digunakan untuk membaca input dari pengguna, input tesebut berupa tombol yang di tekan pada touchscreen. Peracangan *method* ini bisa dilihat pada gambar berikut.

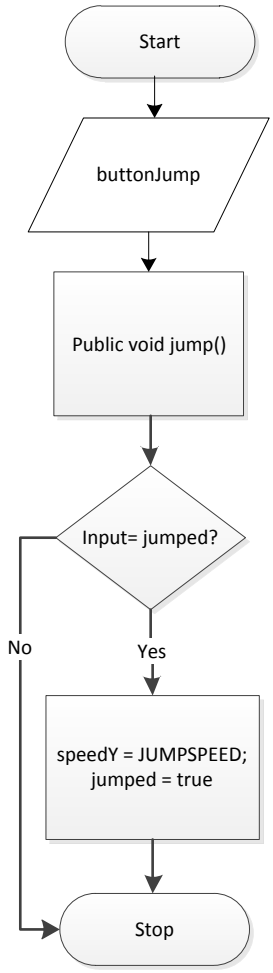


Gambar 3.47. Perancangan *method SetMoving()*

5. Method Jump()

Method jump merupakan *method* yang berfungsi agar aktor melakukan gerakan meloncat. *Method* ini bekerja ketika pengguna menekan tombol jumped

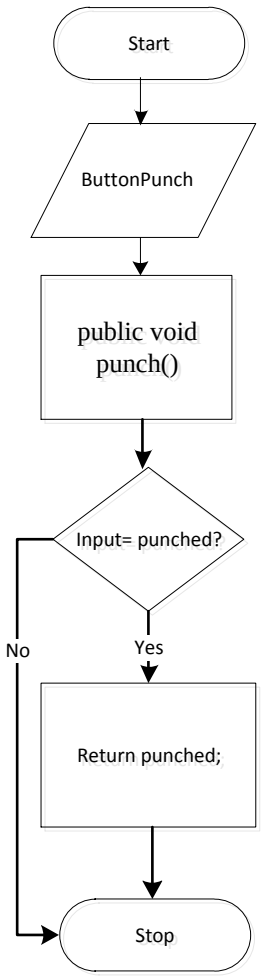
pada touchscreen dan program membacanya. Perancangan *method* jump() bisa dilihat pada gambar berikut ini:



Gambar 3.48. Perancangan *method Jump()*

6. Method Punch()

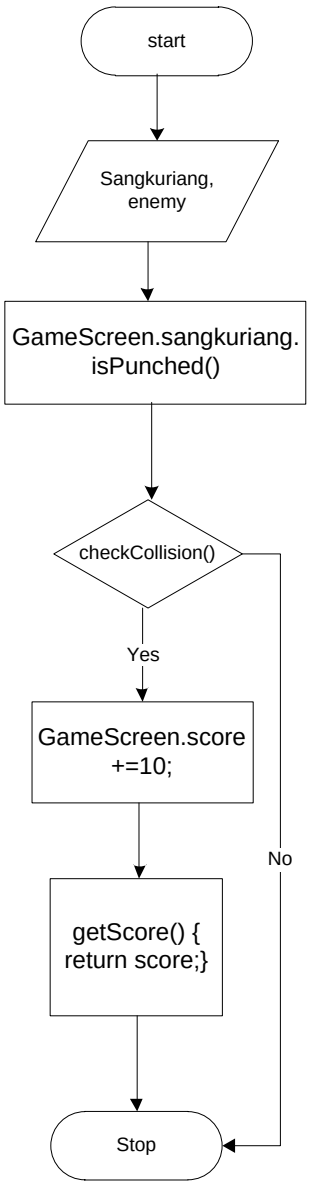
Method punch merupakan *method* yang berfungsi agar aktor melakukan gerakan memukul kepada musuh yang dihadapinya. *Method* ini bekerja ketika pengguna menekan tombol punched pada touchscreen dan program membacanya. Perancangan *method* punch() bisa dilihat pada gambar berikut ini.



Gambar 3.49. Perancangan *method punch*()

7. *Method GetScore()*

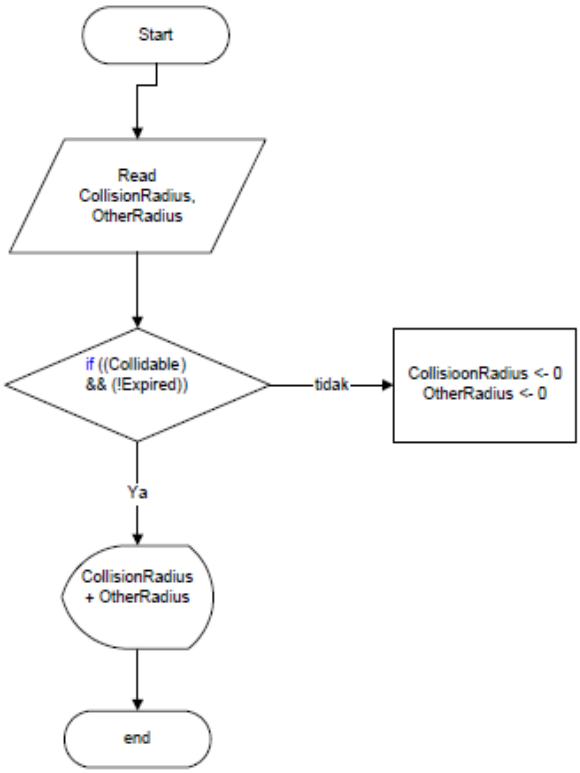
Method GetScore() yang dijelaskan pada sub bab berikut ini merupakan *method* yang bekerja ketika pemain mendapatkan skor. Perancangan *method GetScore()* bisa dilihat pada gambar berikut ini.



Gambar 3.50. Perancangan *method GetScore()*

8. Method CheckCollision()

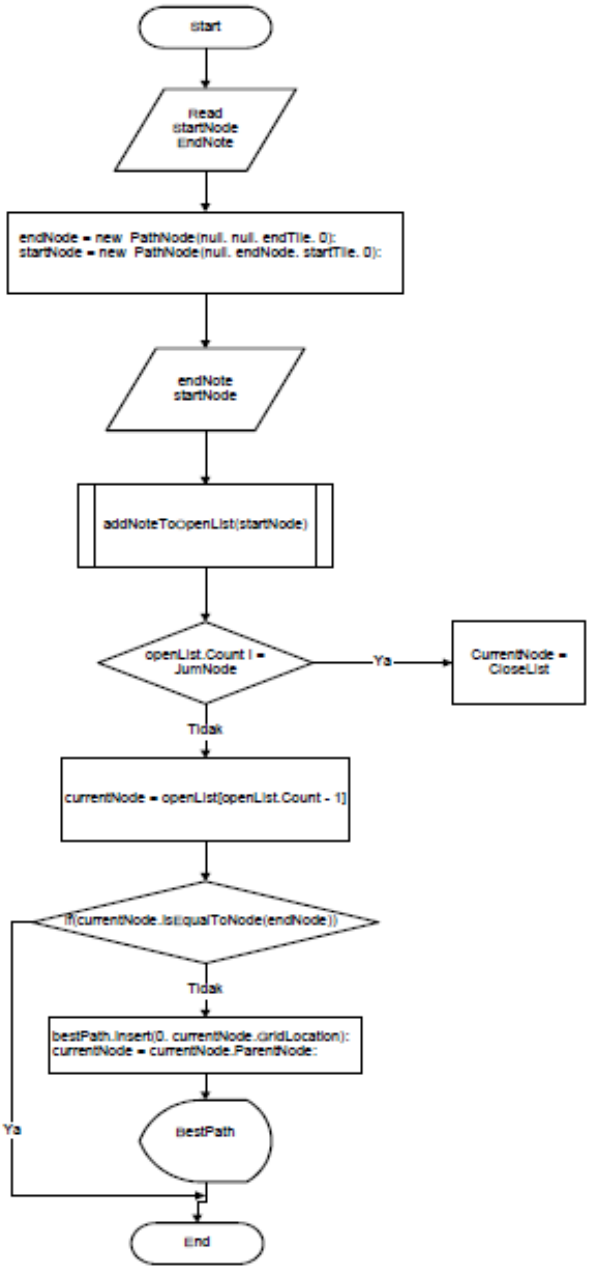
Method ini akan melakukan pengecekan terhadap suatu objek apabila ada kedua objek yang saling bertabrakan. Perancangan method check collision bisa dilihat pada gambar berikut ini.



Gambar 3.51. Perancangan *method CheckCollision()*

9. *Method* FindPath()

Method ini digunakan pada saat objek musuh mendekati karakter pemain yang menggunakan algoritma A Star. Perancangan *method* findpath bisa dilihat pada gambar berikut ini.



Gambar 3.52. Perancangan *method FindPath()*