BAB 3
ANALISIS DAN PERANCANGAN

3.1 Analisis Sistem

Analisis sistem ini merupakan cakupan dari analisis yang berhubungan dengan *game* yang akan dibangun diantaranya analisis masalah terhadap *game* yang sejenis, analisis fungsional dan analisis non fungsional.

3.1.1 Analisis Masalah

Berdasarkan latar belakang masalah yang sudah diuraikan, maka masalah yang ada adalah perkembangan teknologi yang berkembang saat ini seharusnya bisa dimanfaatkan dengan baik, karena dengan memanfaatkan teknologi bisa membantu untuk melestarikan permainan tradisional. Salah satu teknologi yang terus berkembang dan dianggap bisa memperkenalkan permainan tradisional dengan cara yang menyenangkan adalah dengan melibatkan *game*. *Game* merupakan salah satu media yang disukai. Karena selain sifatnya adalah permainan, *game* juga memiliki unsur *multimedia* yang menarik.

Dalam pembahasan, penelitian ini akan membahas *game* dengan *genre sports*. *Game* yang menjadi acuan disini ialah *game* Gobak Sodor *Beta Version*, secara dasar *gameplay* game Gobak Sodor *Beta Version* serupa dengan *gameplay game* Permainan Tradisional Galah yang akan dibangun, akan tetapi pada *game* Permainan Tradisional Galah akan ditambahkan elemen *game* yang dapat membuat skenario *game* menjadi lebih menarik. Selanjutnya rumusan masalah dari permasalahan di atas adalah sebagai berikut:

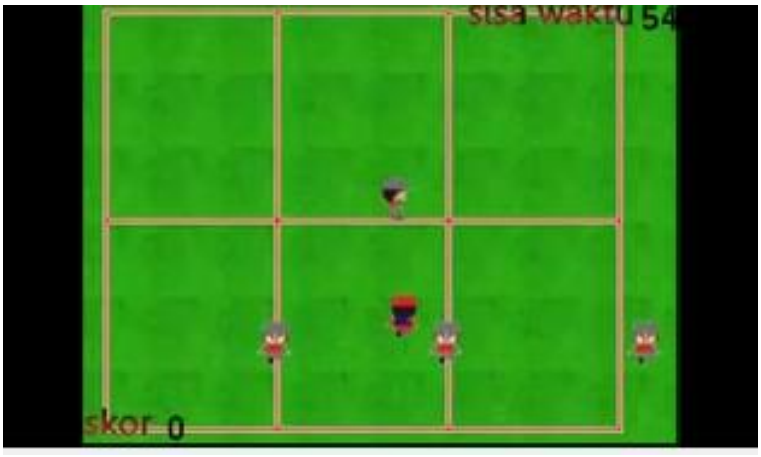
1. Bagaimana membuat *game* yang bertemakan permainan tradisional Indonesia yang dapat memperkenalkan salah satu permainan tradisional kepada pemainnya.
2. Bagaimana membuat pergerakan musuh lebih dinamis serta memberikan tingkatan *level* pada *game*nya.

3.1.2 Analisis *Game* Sejenis

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.

3.1.2.1 Gobak Sodor

Game Gobak Sodor merupakan *game* sejenis dari *game* galah, *game* ini dibangun untuk PC yang dikembangkan oleh sebuah Studio Pengembang *game*. Tujuan utama dari *game* ini ialah untuk mempertahankan diri dari hambatan musuh yang berjalan diatas garis yang telah ditentukan.



Gambar 3.1 *Screenshoot* Gobak Sodor

Game gobak sodor ini terdiri dari lima orang dan enam kolom, dua dari lima orang tersebut dijadikan pemain yang harus melewati hambatan musuh yang terdiri dari tiga orang yang lain dengan kurun waktu yang sudah ditentukan. Kedua pemain tersebut harus melewati hadangan musuh yang berjalan diatas garis kolom tanpa ada sentuhan terhadap musuh. Jika kedua pemain tersebut berhasil melewati musuh dengan kurun waktu yang telah ditentukan dan tanpa ada sentuhan dari musuh, maka pemain tersebut dinyatakan sebagai pemenang. Namun, jika pemain tersebut tidak dapat melewati musuh dengan kurun waktu yang telah ditentukan atau terkena hadangan musuh, maka pemain tersebut dinyatakan kalah.

Kelebihan *game* gobak sodor:

- Mudah digunakan anak-anak
- Aturannya mudah dimengerti

Kekurangan *game* gobak sodor:

- Pergerakan lawan yang masih statis
- Tidak adanya tingkat *level*
- Kualitas grafis kurang atraktif

3.1.3 Analisis *Game* Yang Akan Dibangun

Game permainan tradisional galah merupakan *game* berjenis *sports* yang berlatar permainan tradisional Indonesia, dimana pemain harus menghindari tangkapan musuh untuk mencapai garis akhir dan memenangkan pertandingan.

3.1.3.1 Pengenalan

Game yang akan dibangun berjudul Permainan Tradisional Galah yang ber*genre sports*. *Game* ini dibangun untuk memperkenalkan salah satu kebudayaan atau mengintrepretasikan salah satu game yang sudah ada sejak lama kepada anak-anak masa kini.

Batasan-batasan pada *game* permainan tradisional galah ini adalah:

1. *Game* yang dibuat berbasis desktop dengan tampilan jenis 2 Dimensi.
2. *Game* berupa *single player game*.
3. *Game* berupa *offline game*.
4. *Genre game* adalah *sports.*
5. *Game* terdiri dari 3 *level*.

3.1.3.2 *Storyline*

Game permainan tradisional galah menceritakan tentang sekelompok anak yang bosan karena terus menerus hanya bermain *playstation* di rumah. Akan tetapi, suatu hari salah satu dari mereka menemukan artikel mengenai permainan tradisional di sebuah website. Lantas mereka mencoba untuk memainkan permainan yang ada di

website tersebut, karena jumlah anak yang ingin bermain permainan tradisional jumlahnya banyak akhirnya mereka memutuskan untuk bermain permainan galah yang secara peraturan membutuhkan banyak orang untuk memainkannya. Sebagai permulaan mereka mencoba memainkannya di dekat pantai, kemudian di pekarangan rumah dan juga di sekolah ketika waktu istirahat tiba.

3.1.3.3 *Gameplay*

Gameplay dilakukan untuk menggambarkan aturan-aturan dalam Game. *Gameplay* yang terdapat pada *game* Permainan Tradisional Galah adalah sebagai berikut:

1. Pemain dapat langsung memulai *game* saat memilih menu mulai.
2. Sebelum memulai *game* akan ada intro yang tampil di awal dan pemain harus memilih tombol lanjut untuk memulai permainan.
3. Pemain menggerakkan *hero* dengan tombol W, A, S, D pada *keyboard.*
4. Pemain bertugas menghindari hadangan musuh yang ada, jika terkena maka *lifepoint hero* akan berkurang.
5. Jika *lifepoint hero* telah habis maka *hero* mati dan *game* akan berakhir.
6. Jika waktu telah habis maka *hero* mati dan *game* akan berakhir.
7. Pemain bisa mendapatkan 1 poin apabila mencapai garis akhir.
8. Setiap mendapatkan 3 poin maka arena akan berubah dan *level* musuh akan berbeda.
9. Pemain dapat melihat cara bermain *game* setelah intro.
10. Pemain akan selalu memulai *game* dari *level* pertama saat memilih tombol mulai.
11. *Game* terdiri dari 3 *level*, yaitu:
    a. *Level* 1

       Di *level* 1 pemain bermain galah di arena pantai, pemain harus mengumpulkan 3 poin agar bisa meneruskan ke *level* selanjutnya. Pada *level* 1 ini musuh belum diterapkan kecerdasan buatan.

b. *Level* 2

Di *level* 2 pemain bermain galah di arena pekarangan rumah, pemain harus mengumpulkan 6 poin agar bisa melanjutkan ke *level* ke 3. Pada *level* 2 ini sebagian musuh sudah diterapkan kecerdasan buatan.

c. *Level* 3

Pada *level* 3 pemain bermain galah di arena lapangan sekolah, pemain harus mengumpulkan 9 poin agar bisa menyelesaikan *game*. Pada *level* 3 ini semua musuh sudah diterapkan kecerdasan buatan.

3.1.3.4 Misi Permainan

Misi yang harus diselesaikan yaitu melewati garis kolom tanpa bersentuhan dengan musuh dengan batas waktu yang telah ditentukan.

3.1.4 Analisis Algoritma *Greedy*

Analisis *greedy* digunakan sebagai pengendali untuk menentukan jalur musuh agar tetap berada didalam garis kolom. Dalam kaidah *greedy*, sistem dapat melakukan penalaran dengan prinsip seperti manusia melakukan hadangan tanpa melewati garis kolom.

3.1.4.1 Algoritma *Greedy*

Persoalan karakter musuh dalam menentukan arah mana yang harus dijalaninya untuk semakin mendekatkan dirinya kepada karakter pemaindapat dikategorikan sebagai persoalan optimasi, dan persoalan optimasi cukup efektif dipecahkan dengan menggunakan algoritma *greedy*. Persoalan optimasi pada musuh ini termasuk persoalan minimasi, yaitu mencari rute yang harus dilalui oleh musuh.

Dalam bahasa Indonesia, *greedy* adalah rakus. Prinsip dari algoritma *greedy* adalah memilih keputusan yang terbaik pada setiap langkahnya (optimum lokal). Keputusan – keputusan lokal yang diambil pada akhirnya menjadi solusi optimum yang mutlak (*optimum global*).

3.1.4.2 Penerapan Algoritma *Greedy*

Dalam penerapannya, algoritma *greedy* digunakan saat pemilihan jalur terpendek untuk tiap lawan.

Secara garis besar, alur kerja algoritma adalah mencari jalur terpendek antara karakter dengan tiap lawan menggunakan optimasi *greedy*. Berikut *pseudo* algoritma *greedy* bisa dilihat pada gambar 3.2.

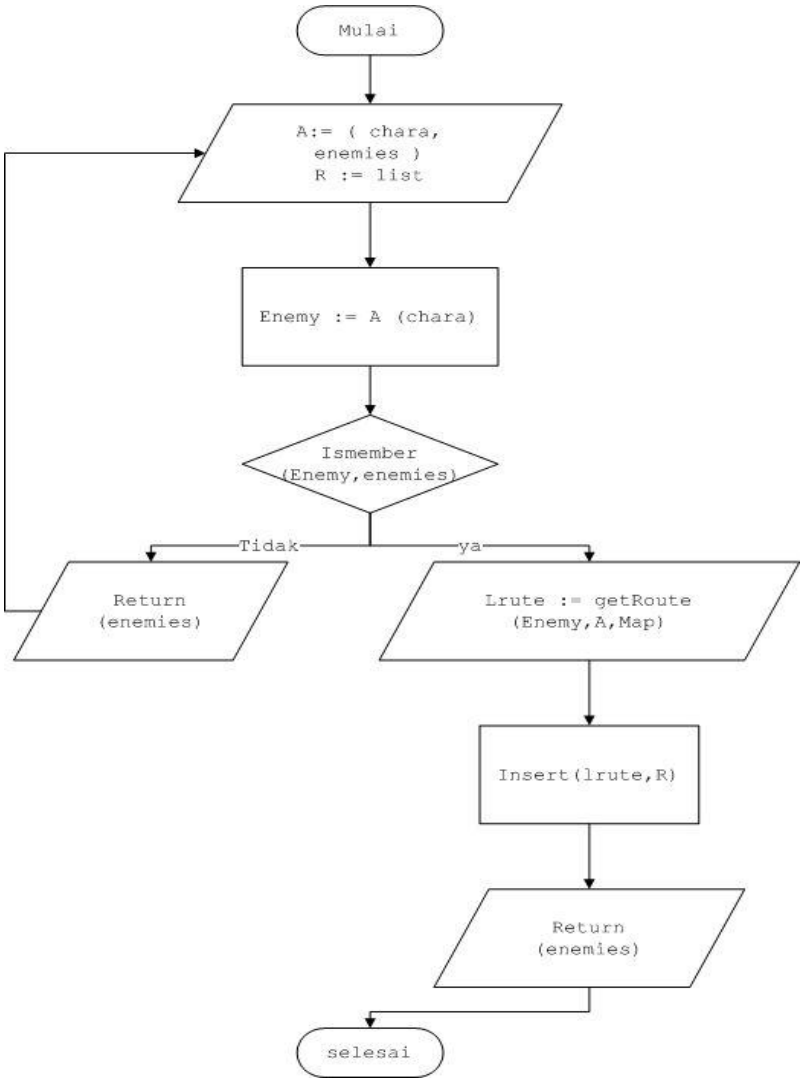
```
procedure enemyMinRoutes (input M :
map, A : chara, enemies : list of
chara, output R : list of route)
{program menerima masukan berupa
matriks map, karakter yang akan
digerakkan, dan list musuh}

Deklarasi
enemy := chara;
lrute := list of route;
rute := route;

Algoritma
enemy := first(enemies);
while (isMember(enemy,enemies)) do
{fungsi seleksi}
 rute := shortest(lrute);
{fungsi kelayakan}
 insert(rute,R);
{fungsi obyektif}
 lrute := getRoutes(enemy,A,M);
 enemy := next(enemies);
```

Gambar 3.2 *Pseudo* Algoritma *Greedy*

Dalam penerapannya algoritma *greedy* digunakan saat pemilihan jalur terpendek untuk setiap musuh, berikut adalah algoritma *greedy* bisa dilihat pada gambar 3.3.

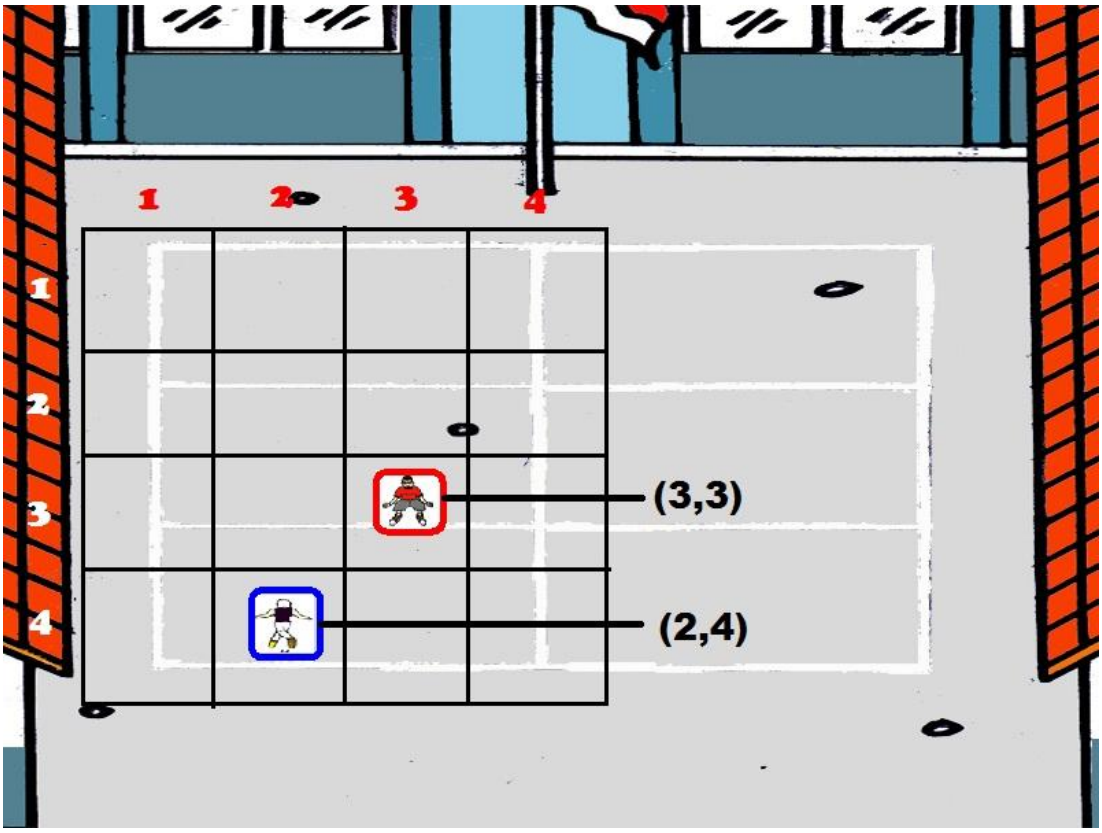


Gambar 3.3 Algoritma *Greedy*

Dimulai dari titik awal karakter yaitu chara dan enemies, chara adalah titik awal pemain dan enemies adalah titik awal musuh. Setelah titik awal diinisialisasi titik pemain dimasukan ke dalam *list* lalu diproses oleh sistem dengan fungsi kelayakan, jika titik chara dan enemies terdapat dalam *list* maka akan lanjut ke proses selanjutnya, jika tidak sistem akan menginisialisasi kembali. Setelah titik objek dimasukan list maka sistem akan menyesuaikan titik dengan titik-titik *map* sehingga sistem bisa mendapatkan rute terpendek dan dapat ditelusuri.

3.1.4.3 Contoh Kasus Algoritma *Greedy*

Pada metode *greedy* yang terdapat dalam *game* Permainan Tradisional Galah dapat diilustrasikan pada gambar 3.4 sebagai berikut:



Gambar 3.4 Ilustrasi Algoritma *Greedy*

Pada ilustrasi ini terdapat *node* awal atau posisi musuh yang berada pada koordinat (3,3), sedangkan *node* tujuan atau posisi pemain berada pada kooordinat (2,4). Selanjutnya musuh akan mulai mencari node-node yang berdekatan dengan pemain. *Node* yang mungkin diperiksa adalah (4,3), (3,4), (2,3), (3,2). *Node* yang mungkin diperiksa tersebut akan dilakukan perhitungan sebagai berikut:

1. *Node* (4,3) dan *Node* tujuan (2,4)

   Dengan fungsi heuristik seperti f(n) = h(n)

   Dimana f(n) adalah fungsi heuristik dari n, dan h(n) adalah perkiraan jarak tersisa ke tujuan.

h_orthogonal(n) = (abs(n.x – goal.x)+abs(n.y – goal.y))
h_orthogonal(4,3) = (abs(4 – 2) + abs(3 – 4))
= (abs(2)+abs(-1))
= 3
h_diagonal(n) = min(abs(n.x – goal.x) + abs(n.y – goal.y))
h_diagonal(4,3) = min(abs(4 – 2) + abs(3 – 4))
= min(abs(2) + abs(-1))
= min 3
h(n) = h_diagonal + (h_orthogonal(n) – (2*h_diagonal(n)))
h(4,3) = (-3) + (3 – (2*(-3)))
= -3 + 9
= 6

2. *Node* (3,4) dan *Node* tujuan (2,4)

h_orthogonal(n) = (abs(n.x – goal.x) + abs(n.y – goal.y))
h_orthogonal(3,4) = (abs(3 – 2) + abs(4 – 4))
= (abs(1) + abs(0))
= 1
h_diagonal(n) = min(abs(n.x – goal.x) + abs(n.y – goal.y))
h_diagonal(3,4) = min(abs(3 – 2) + abs(4 – 4))
= min(abs(1) + abs(0))
= min 1
h(n) = h_diagonal + (h_orthogonal(n) – (2*h_diagonal(n)))
h(1,1) = (-1) + (1 – (2*(-1)))
= -1 + 3
= 2

3. *Node* (2,3) dan *Node* tujuan (2,4)

h_orthogonal(n) = (abs(n.x – goal.x) + abs(n.y – goal.y))
h_orthogonal(2,3) = (abs(2 – 2) + abs(3 – 4))
= (abs(0) + abs(-1))

= 1

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

h_diagonal(2,3) = min(abs(2 – 2) + abs(3 – 4))

= min(abs(1) + abs(0))

= min 1

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

h(2,3) = (-1) + (1 – (2*(-1)))

= -1 + 3

= 2

4. *Node* (3,2) dan *Node* tujuan (2,4)

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

h_orthogonal(3,2) = (abs(3 – 2) + abs(2 – 4))

= (abs(1) + abs(-2))

= 3

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

h_diagonal(3,2) = min(abs(3 – 2) + abs(2 – 4))

= min(abs(1) + abs(-2))

= min 3

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

h(3,2) = (-3) + (1 – (2*(-3)))

= -3 + 7

= 4

Dari hasil yang didapat, *greedy* akan memperkirakan biaya yang paling kecil untuk mencapai tujuan nya, dari hasil yaitu (4,3) = 6 , (3,4) = 2, (2,3) = 2, (3,2) = 4. Dengan hasil tersebut untuk mencapai tujuan algoritma *greedy* akan memilih *node* (2,3) dikarenakan biaya estimasi yang kecil untuk mencapai node (2,4).

3.1.5 Analisis Kebutuhan Non-Fungsional

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

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

3.1.5.1 Analisis Kebutuhan Perangkat Lunak (Pembangun)

Perangkat lunak digunakan dalam sebuah sistem merupakan perintah yang diberikan kepada perangkat keras agar dapat saling berinteraksi diantaranya. Perangkat lunak yang dibutuhkan untuk pembangunan *game* Permainan Tradisional Galah ini adalah sebagai berikut:

Tabel 3.1 Spesifikasi Perangkat Lunak (Pembangun)

Sistem operasi	Windows 7 *Ultimate* 32-bit
Compiler	Mozilla Firefox
Framework	Construct 2
Development toolkit	Construct 2
Design tools	Adobe Photoshop CS 3
Design analysis	StarUML

3.1.5.2 Analisis Kebutuhan Perangkat Keras (Pembangun)

Komputer terdiri dari perangkat keras dan perangkat lunak yang saling berinteraksi. Perangkat lunak memberikan instruksi kepada perangkat keras untuk melakukan tugas tertentu. Adapun perangkat keras yang digunakan dalam membangun *game* Permainan Tradisional Galah adalah sebagai berikut:

Tabel 3.2 Spesifikasi Perangkat Keras (Pembangunan)

Nama Perangkat	Spesifikasi
Processor	Dual Core
RAM	2000 MB
Harddisk	500 GB
Display	Resolusi 800 x 600
Keyboard	Standar
Mouse	Standar

3.1.5.3 Analisis Kebutuhan Perangkat Lunak (Pengguna)

Pada sisi pengguna, perangkat lunak yang dibutuhkan untuk menjalankan *game* ini adalah sebagai berikut:

Tabel 3.3 Spesifikasi Perangkat Lunak (Pengguna)

Sistem Operasi	Windows XP
Perangkat Lunak	Mozilla Firefox

3.1.5.4 Analisis Kebutuhan Perangkat Keras (Pengguna)

Pada sisi pengguna, perangkat keras yang dibutuhkan untuk menjalankan *game* ini adalah sebagai berikut:

Tabel 3.4 Spesifikasi Perangkat Keras (Pengguna)

Nama perangkat	Spesifikasi
Processor	Minimum Pentium 4
RAM	Minimum 512 MB
Harddisk	Minimum 200 MB
Display	Resolusi 800 x 600
Keyboard	Standar
Mouse	Standar

3.1.5.5 Analisis Kebutuhan Pengguna

Spesifikasi minimal bagi pengguna yang dapat menggunakan *game* ini adalah sebagai berikut:

Tabel 3.5 Spesifikasi Pengguna

Peran	Pemain
Usia	6 tahun ke atas
Tingkat Keterampilan	Dapat menggunakan *computer*

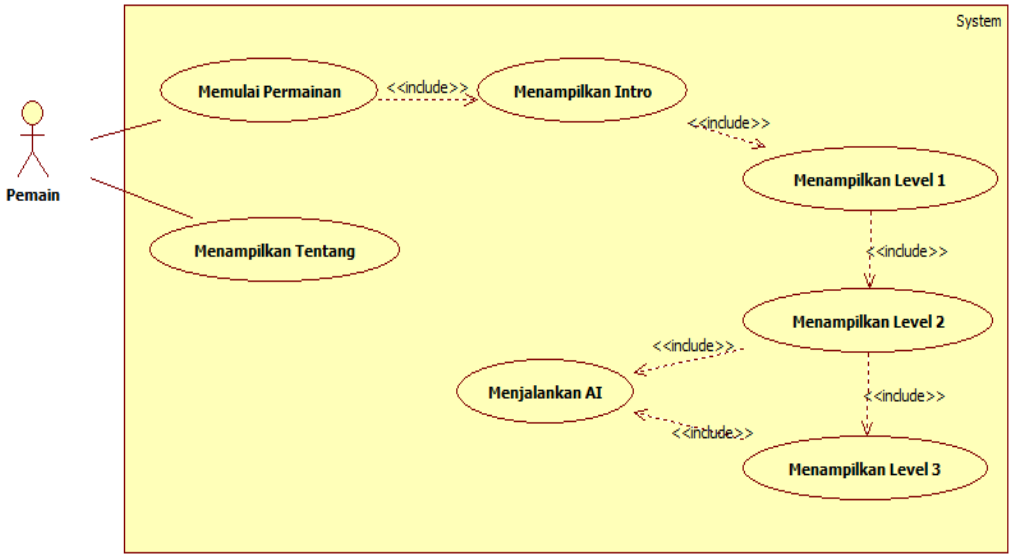
3.1.6 Analisis Kebutuhan Fungsional

Tahap pertama yang pertama dalam melakukan analisis berorientasi objek menggunakan *UML* adalah menentukan aktor atau pengguna sistem, dimana aktor menampilkan peran (*roles*) yang dilakukan pada sistem sebagai aksi dan respon dari sistem.

3.1.6.1 *Use Case Diagram*

Use case diagram merupakan gambaran fungsional dari sebuah sistem sehingga pengguna sistem mengerti mengenai kegunaan sistem yang akan dibangun.

Berikut ini adalah interaksi antara pengguna dengan *game* Permainan Tradisional Galah yang akan digambarkan sebagai berikut:



Gambar 3.5 *Use Case Diagram*

3.1.6.2 Definisi *Use Case*

Definisi *use case* berfungsi untuk menjelaskan fungsi dari *use case* yang terdapat pada diagram *use case*. Definisi *use case* dijelaskan pada tabel berikut:

Tabel 3.6 Definisi *Use Case*

No	*Use Case*	Keterangan
1.	Memulai Permainan	Proses untuk memulai permainan
2.	Menampilkan Tentang	Proses menampilkan tentang pembuatan
3.	Menampilkan Intro	Proses menampilkan intro
4.	*Level* 1	Proses permainan pada *level* 1
5.	*Level* 2	Proses permainan pada *level* 2
6.	*Level* 3	Proses permainan pada *level* 3

No	*Use Case*	Keterangan
7.	AI (kecerdasan buatan)	Proses pergerakan musuh yang menghadang pemain

3.1.6.3 Skenario *Use Case*

Skenario pada diagram *use case* merupakan penjelasan mengenai alur kejadian yang di dalamnya menjelaskan urutan interaksi antara aktor dengan *use case* tersebut. Adapun skenario *use case* yang terdapat pada *game* Permainan Tradisional Galah adalah sebagai berikut:

Tabel 3.7 Skenario *Use Case* Memulai Permainan

Identifikasi	
Nomor	1
Nama	Memulai Permainan
Tujuan	Masuk ke halaman intro
Deskripsi	Proses untuk memulai permainan
Aktor	Pemain
Skenario Utama	
Kondisi awal	Pemain berada di menu utama *game* permainan tradisional galah
Aksi Aktor	Reaksi Sistem
1. Memilih menu Mulai	
	2. Menampilkan tampilan Intro
3. Menekan tombol lanjut	
	4. Menampilkan Intro selanjutnya

Tabel 3.8 Skenario *Use Case* Menampilkan Tentang

Identifikasi	
Nomor	2
Nama	Menampilkan Tentang
Tujuan	Menampilkan Tentang pembuatan
Deskripsi	Proses untuk menampilkan Tentang
Aktor	Pemain
Skenario Utama	
Kondisi Awal	Pemain berada di menu utama *game* permainan tradisional galah
Aksi Aktor	Reaksi Sistem
1. Memilih menu tentang	
	2. Menampilkan Tentang pemebuat
3. Menekan tombol kembali	
	4. Menampilkan halaman utama

Tabel 3.9 Skenario *Use Case* Menampilkan Intro

Identifikasi	
Nomor	3
Nama	Tampilkan Intro
Tujuan	Menampilkan Intro permainan
Deskripsi	Proses untuk menampilkan Intro permainan
Aktor	Pemain
Skenario Utama	
Kondisi Awal	Pemain berada di halaman Intro permainan

Aksi Aktor	Reaksi Sistem
1. Menekan tombol lanjut	
	2. Menampilkan info
	3. Menampilkan awal permainan
Kondisi Akhir	Masuk ke tampilan awal permainan

Tabel 3.10 Skenario *Use Case Level* 1

Identifikasi	
Nomor	4
Nama	Tampil *Level* 1
Tujuan	Menampilkan permainan *level* 1
Deskripsi	Proses untuk memainkan permainan *level* 1
Aktor	Pemain
Skenario Utama	
Kondisi Awal	Pemain berada di tampilan permainan *level* 1
Aksi Aktor	Reaksi Sistem
	1. Menggerakan musuh yang bergerak diatas garis kolom
2. Menekan tombol navigasi pada keyboard untuk menggerakan *hero* pemain	
	3. Menjalankan waktu permainan
	4. Mereaksi tombol yang ditekan

	pengguna
5. Berhasil melewati musuh dengan batas waktu yang telah ditentukan	
	6. Mengakhiri permainan
Skenario Tambahan	
1. Terkena sentuhan musuh atau tidak dapat menyelesaikan game dari batas waktu yang telah ditentukan	
	2. Mengakhiri permainan
Skenario Tambahan	
1. Tidak dapat menyelesaikan permainan dengan batas waktu yang ditentukan	
	2. Mengakhiri permainan

Tabel 3.11 Skenario *Use Case Level* 2

Identifikasi	
Nomor	5
Nama	Tampil *Level* 2
Tujuan	Menampilkan permainan *level* 2
Deskripsi	Proses untuk memainkan permainan *level* 2
Aktor	Pemain

Skenario Utama	
Kondisi Awal	Pemain berada di tampilan permainan level 2
Aksi Aktor	Reaksi Sistem
	1. Menggerakan musuh yang bergerak diatas garis kolom dengan lebih cepat dari *level* 1
2. Menekan tombol navigasi pada keyboard untuk menggerakan *hero* pemain	
	3. Menjalankan waktu permainan
	4. Mereaksi tombol yang ditekan pengguna
5. Berhasil melewati musuh dengan batas waktu yang telah ditentukan atau terkena sentuhan musuh atau tidak dapat menyelesaikan game dari batas waktu yang telah ditentukan	
	6. Mengakhiri permainan
Skenario Tambahan	
3. Terkena sentuhan musuh atau tidak dapat menyelesaikan game dari batas waktu yang telah ditentukan	

	4. Mengakhiri permainan
Skenario Tambahan	
3. Tidak dapat menyelesaikan permainan dengan batas waktu yang ditentukan	
	4. Mengakhiri permainan

Tabel 3.12 Skenario *Use Case Level* 3

Identifikasi	
Nomor	6
Nama	Tampil *Level* 3
Tujuan	Menampilkan permainan *level* 3
Deskripsi	Proses untuk memainkan permainan *level* 3
Aktor	Pemain
Skenario Utama	
Kondisi Awal	Pemain berada di tampilan permainan *level* 3
Aksi Aktor	Reaksi Sistem
	1. Menggerakan musuh yang bergerak diatas garis kolom dengan lebih cepat dari *level* 2
2. Menekan tombol navigasi pada keyboard untuk menggerakan *hero* pemain	
	3. Menjalankan waktu permainan
	4. Mereaksi tombol yang ditekan

	pengguna
5. Berhasil melewati musuh dengan batas waktu yang telah ditentukan atau terkena sentuhan musuh atau tidak dapat menyelesaikan game dari batas waktu yang telah ditentukan	
	6. Mengakhiri permainan
Skenario Tambahan	
5. Terkena sentuhan musuh atau tidak dapat menyelesaikan game dari batas waktu yang telah ditentukan	
	6. Mengakhiri permainan
Skenario Tambahan	
5. Tidak dapat menyelesaikan permainan dengan batas waktu yang ditentukan	
	6. Mengakhiri permainan

Tabel 3.13 Skenario *Use Case AI*

Identifikasi	
Nomor	7
Nama	AI

Tujuan	Menggerakan musuh	
Deskripsi	Proses pergerakan musuh	
Aktor	Pemain	
Skenario Utama		
Kondisi Awal	Pemain berada di arena permainan	
Aksi Aktor		Reaksi Sistem
1. Berada di *level* 2		2. Menggerakan musuh yang bergerak diatas garis kolom
3. Menggerakan karakter pemain.		
		4. Melakukan pergerakan diagonal untuk menjaga pergerakan karakter pemain
		5. Mereaksi tombol yang ditekan
6. Berhasil melewati musuh dengan batas waktu yang telah ditentukan.		
		7. Mengakhiri permainan
Skenario Tambahan		
	1. Mengenai karakter pemain	
	2. Mengakhiri permainan	

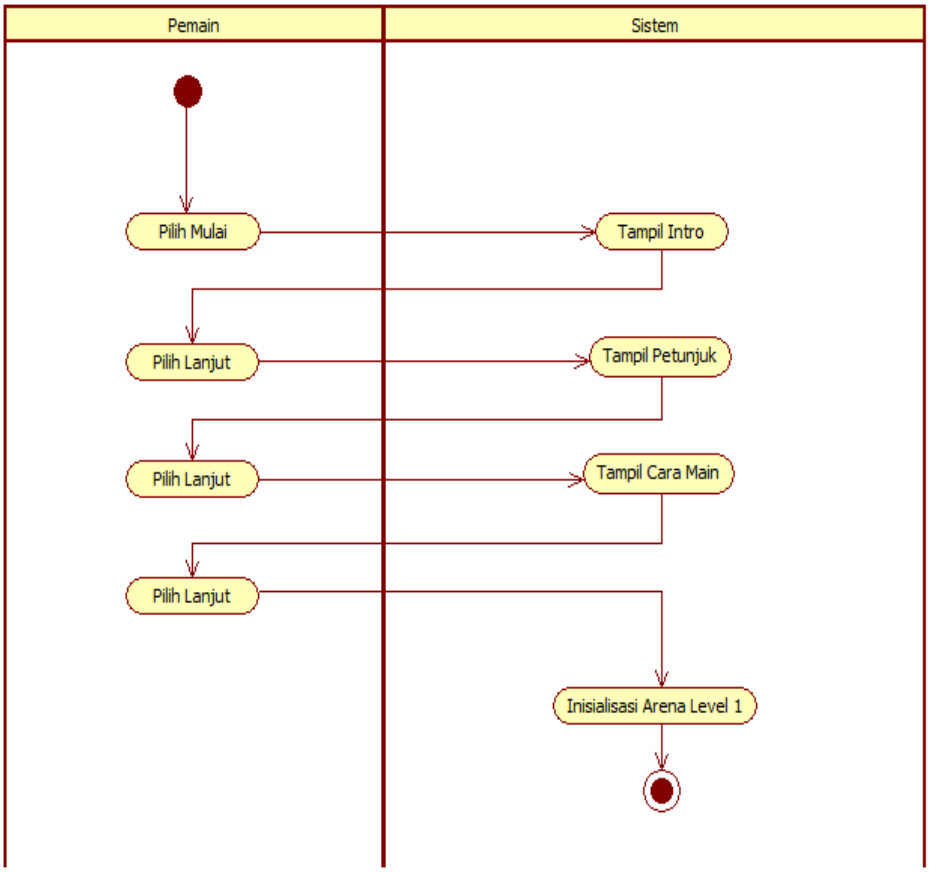
3.1.6.4 *Activity Diagram*

Activity diagram mempunyai penjelasan lebih mengenai urutan suatu proses yang terjadi antara aktor dan sistem. Aktivitas yang pertama dimulai dari aktor,

kemudian akan mendapat respon dari sistem yang nantinya akan menampilkan hasil dari proses tersebut.

1. *Activity Diagram* Mulai

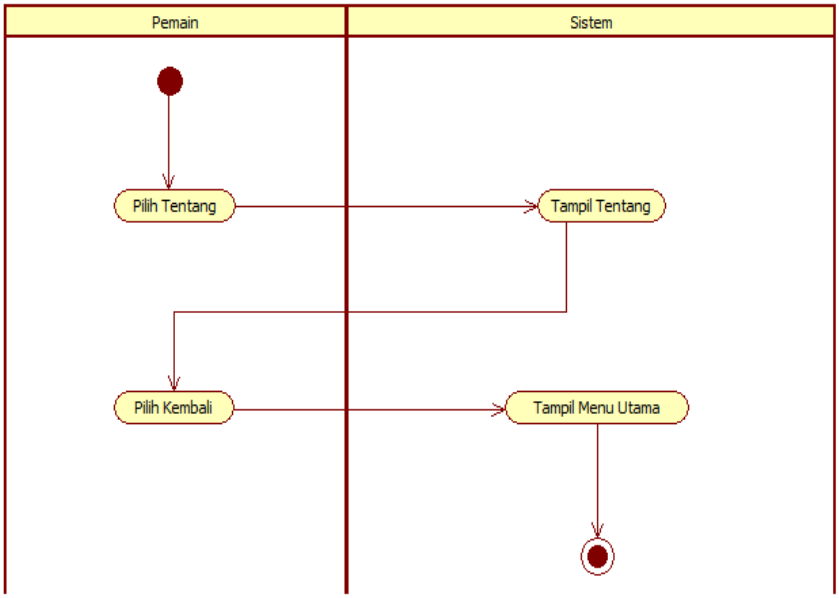
Activity diagram ini menjelaskan alur dari aktivitas mulai antar aktor dan sistem yang dijelaskan pada gambar 3.6.



Gambar 3.6 *Activity Diagram* Mulai

2. *Activity Diagram* Tentang

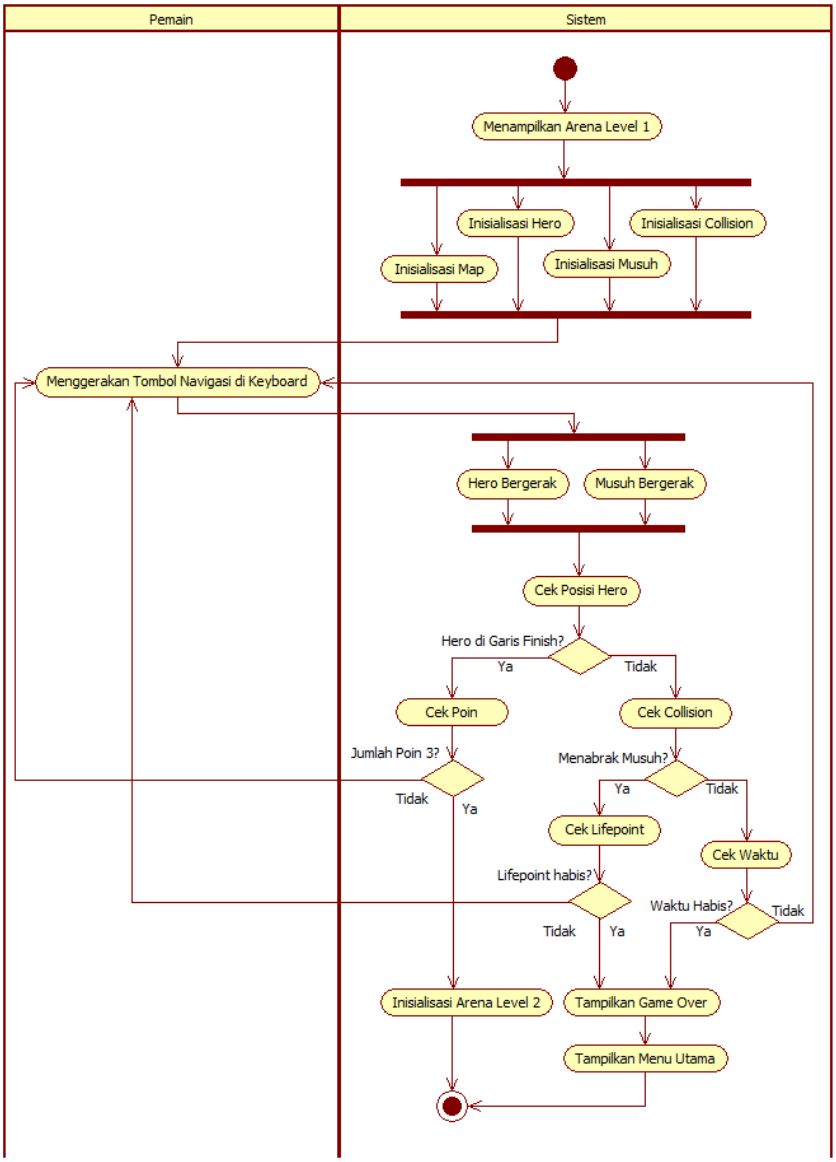
Activity diagram ini menjelaskan alur dari aktivitas Tentang antar aktor dan sistem yang dijelaskan pada gambar 3.7.



Gambar 3.7 *Activity Diagram* Tentang

3. *Activity Diagram Level* 1

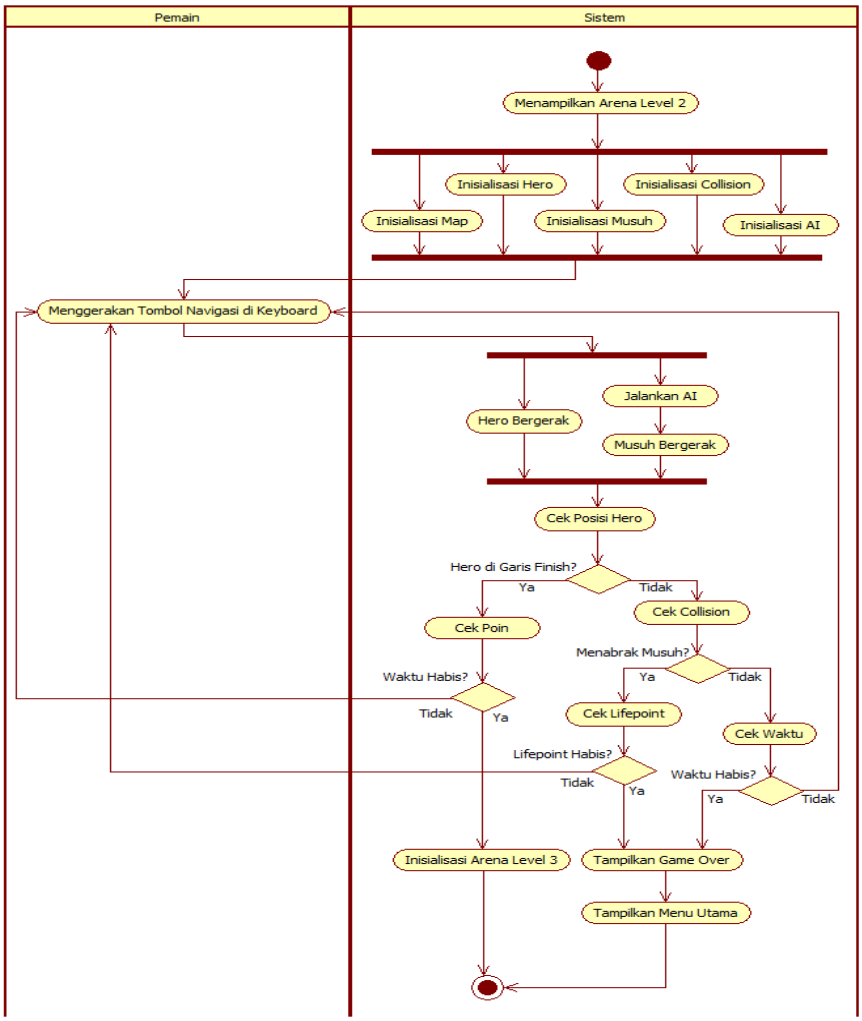
Activity diagram ini menjelaskan alur dari aktivitas *level* 1 antar aktor dan sistem yang dijelaskan pada gambar 3.8.



Gambar 3.8 *Activity Diagram level* 1

4. *Activity Diagram Level* 2

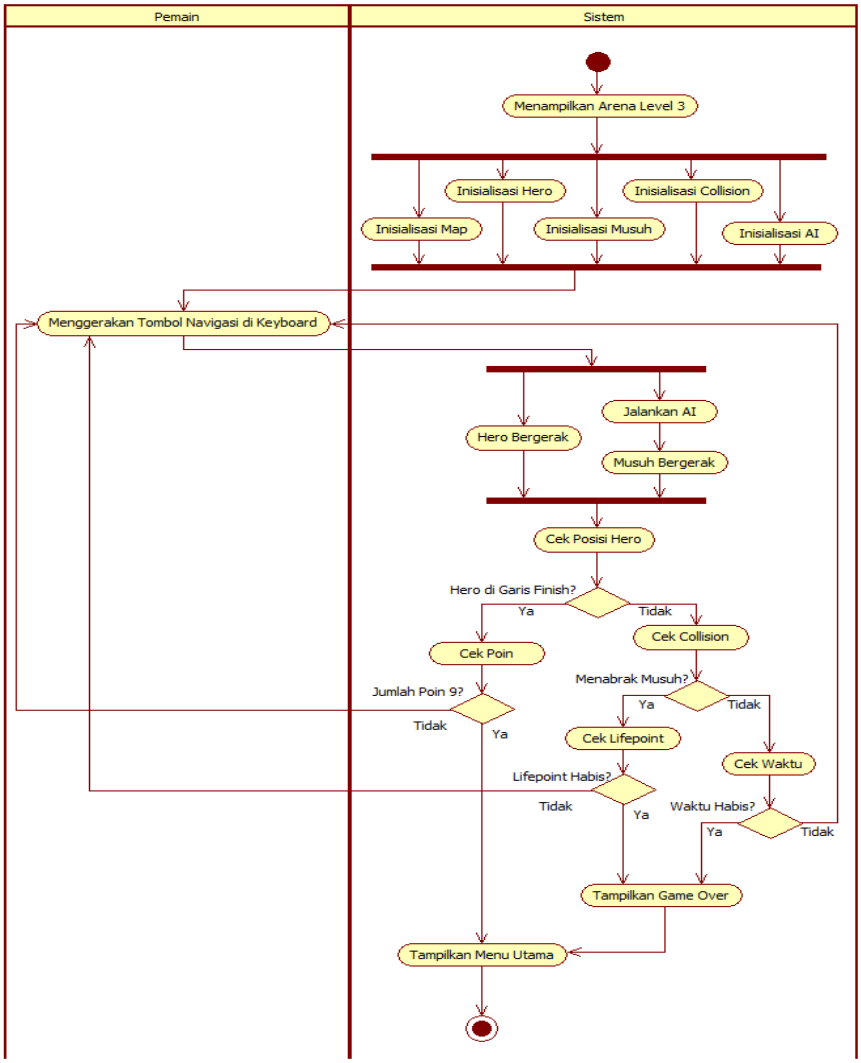
Activity diagram ini menjelaskan alur dari aktivitas *level* 2 antar aktor dan sistem yang dijelaskan pada gambar 3.9.



Gambar 3.9 *Activity Diagram level* 2

5. *Activity Diagram Level* 3

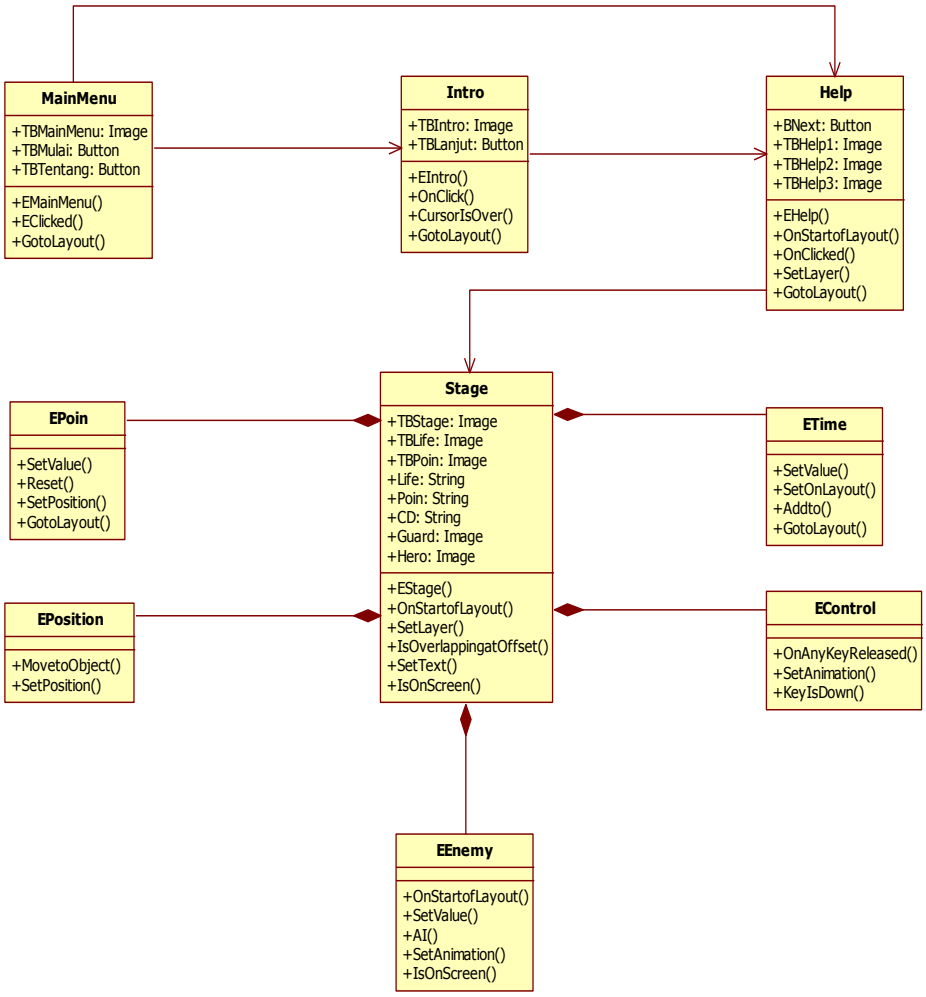
Activity diagram ini menjelaskan alur dari aktivitas *Level* 3 antar aktor dan sistem yang dijelaskan pada gambar 3.10.



Gambar 3.10 *Activity Diagram level* 3

3.1.6.5 *Class Diagram*

Class diagram digunakan untuk menampilkan kelas-kelas dan paket-paket di dalam sistem. *Class diagram* memberikan gambaran sistem berupa relasi antar kelas-kelas ataupun paket-paketnya.



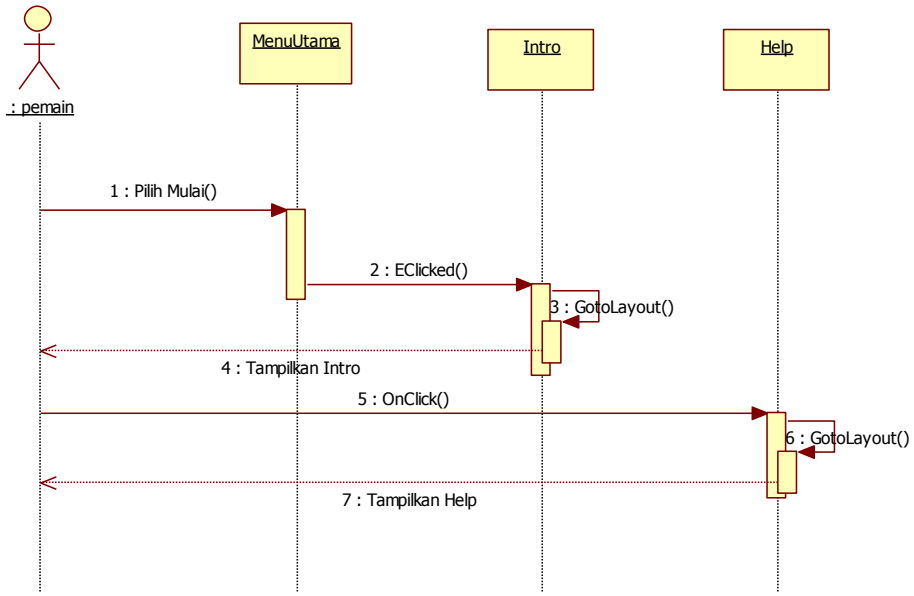
Gambar 3.11 *Class Diagram*

3.1.6.6 *Sequence Diagram*

Sequence diagram menjelaskan kelakuan objek pada *use case* dengan mendeskripsikan berdasarkan waktu hidup dari pesan yang dikirim dan diterima antar objek.

1. *Sequence Diagram* mulai

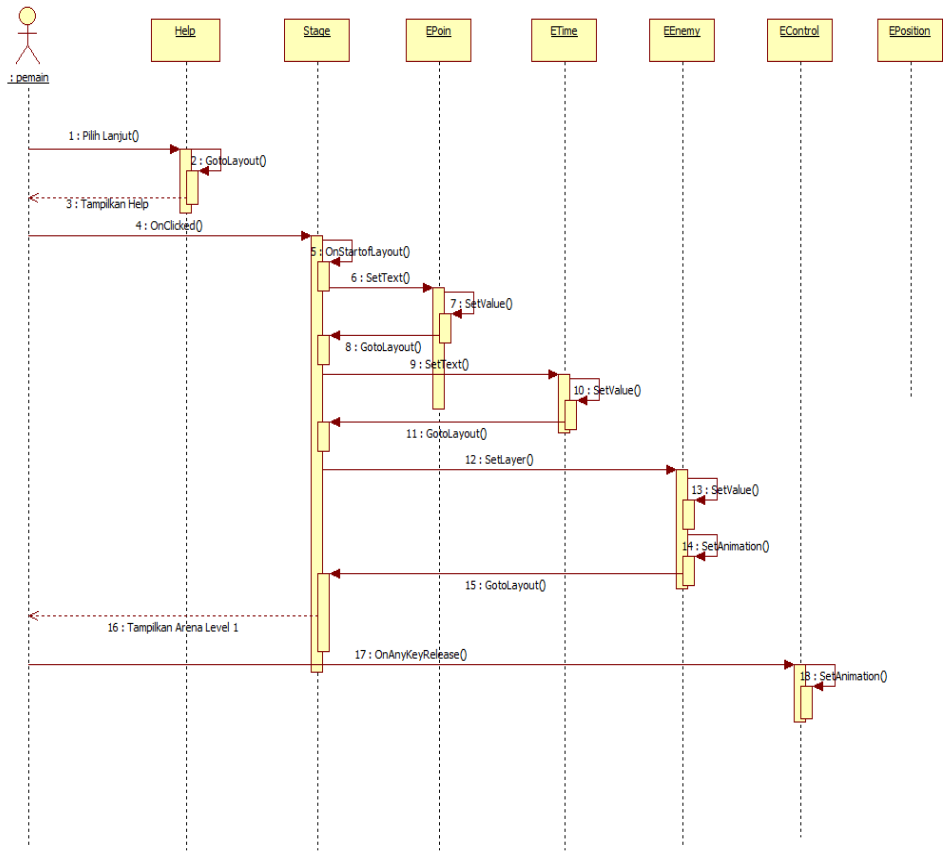
Sequence diagram ini menjelaskan urutan interaksi dari *use case* memulai permainan, dimana antar objek dan pesan yang dijelaskan pada gambar 3.12.



Gambar 3.12 *Sequence Diagram* memulai permainan

2. *Sequence Diagram Level* 1

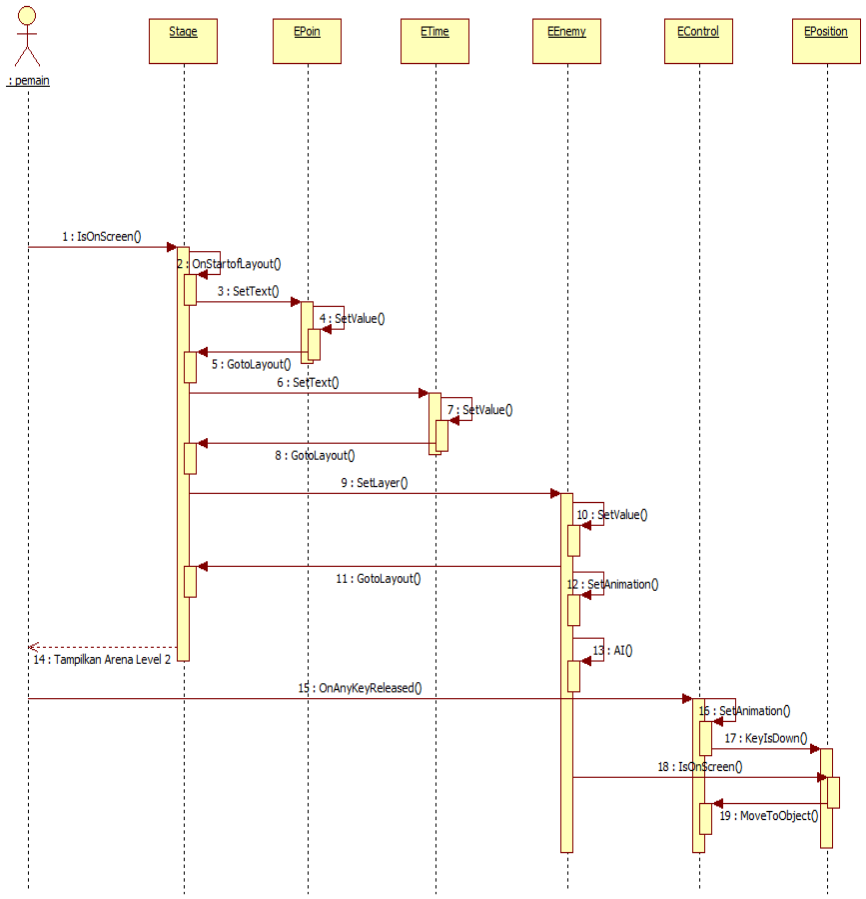
Sequence diagram ini menjelaskan urutan interaksi dari *use case level* 1, dimana antar objek dan pesan yang dijelaskan pada gambar 3.13.



Gambar 3.13 *Sequence Diagram level* 1

3. *Sequence Diagram Level* 2

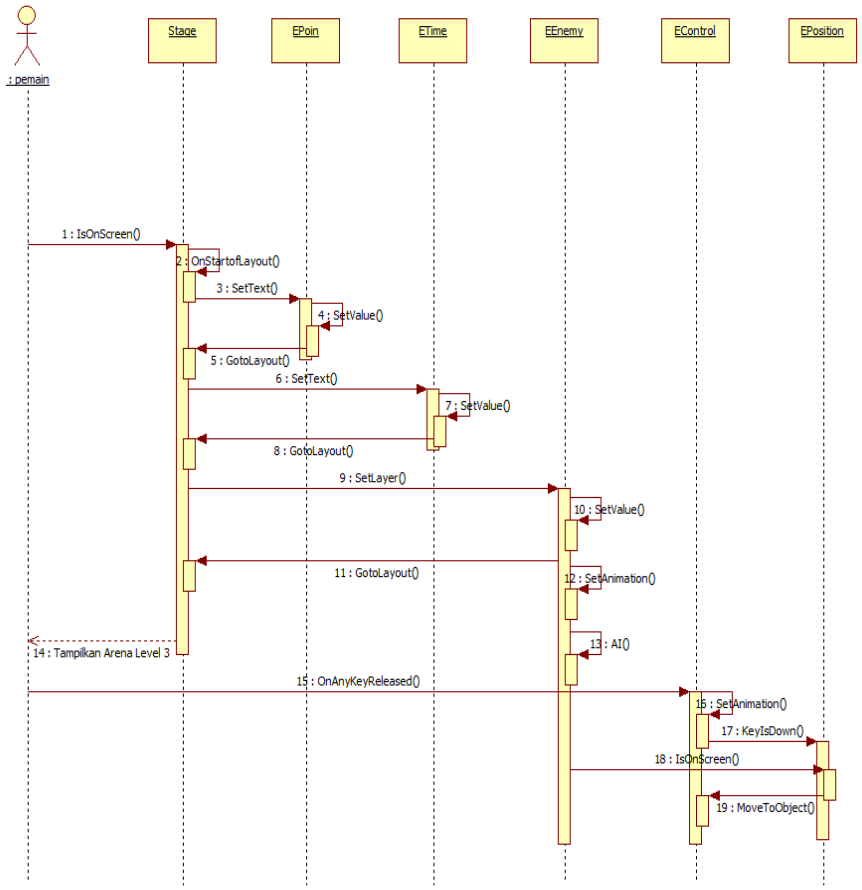
Sequence diagram ini menjelaskan urutan interaksi dari *use case level 2*, dimana antar objek dan pesan yang dijelaskan pada gambar 3.14.



Gambar 3.14 *Sequence Diagram level* 2

4. *Sequence Diagram Level* 3

Sequence diagram ini menjelaskan urutan interaksi dari *use case level* 3, dimana antar objek dan pesan yang dijelaskan pada gambar 3.15.



Gambar 3.15 *Sequence Diagram level* 3

3.2 Perancangan Sistem

Tahap perancangan merupakan kelanjutan dari proses analisis dimana dilakukan perubahan-perubahan terhadap sistem yang sedang berjalan. Hal ini dilakukan untuk mengatasi kekurangan yang ada, memudahkan pekerjaan yang dilakukan oleh orang yang terlibat dan menghemat waktu pekerjaan.

Perancangan *game* ini meliputi perancangan struktur menu, perancangan antarmuka, perancangan pesan, jaringan semantik, perancangan prosedural. Semua

perancangan yang telah disebutkan diatas harus saling berhubungan agar menciptakan suatu sistem yang baik.

3.2.1 Perancangan Komponen Permainan

Perancangan komponen permainan merupakan bagian pendeskripsian setiap komponen yang ada pada *game* baik karakter, bangunan, dan *storyboard*. Perancangan komponen permainan terdiri dari perancangan karakter dan perancangan *storyboard*.

3.2.1.1 Perancangan Karakter

Perancangan karakter adalah bagian dimana karakter pada game dijelaskan ciri-cirinya baik itu secara fungsionalitas peran maupun secara desain. Berikut ini adalah perancangan karakter manusia pada *game* Permainan Tradisional Galah.

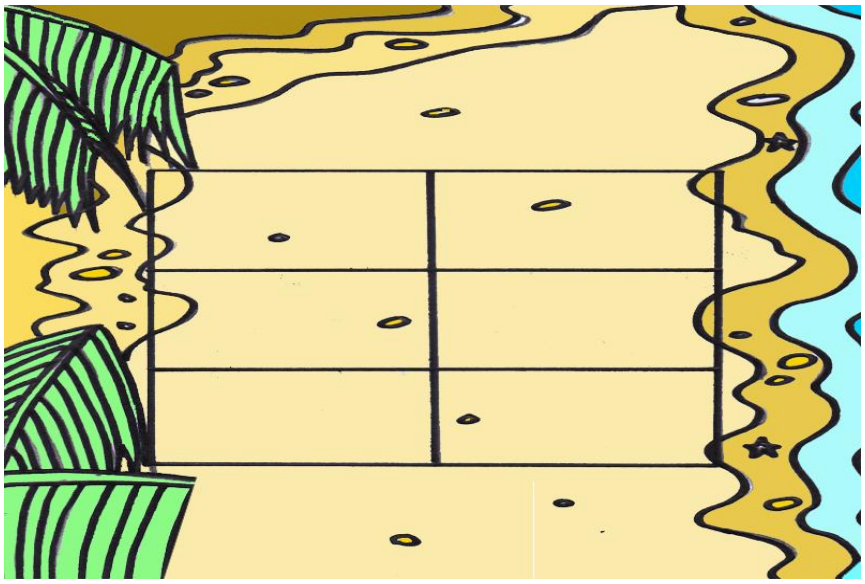
Tabel 3.14 Keterangan Karakter

No.	Nama	Gambar	Keterangan
1.	Musuh		NPC penghambat atau penghalang karakter pemain yang berjalan diatas garis kolom
2.	Hero		Karakter yang harus menyelesaikan misi untuk melewati musuh.

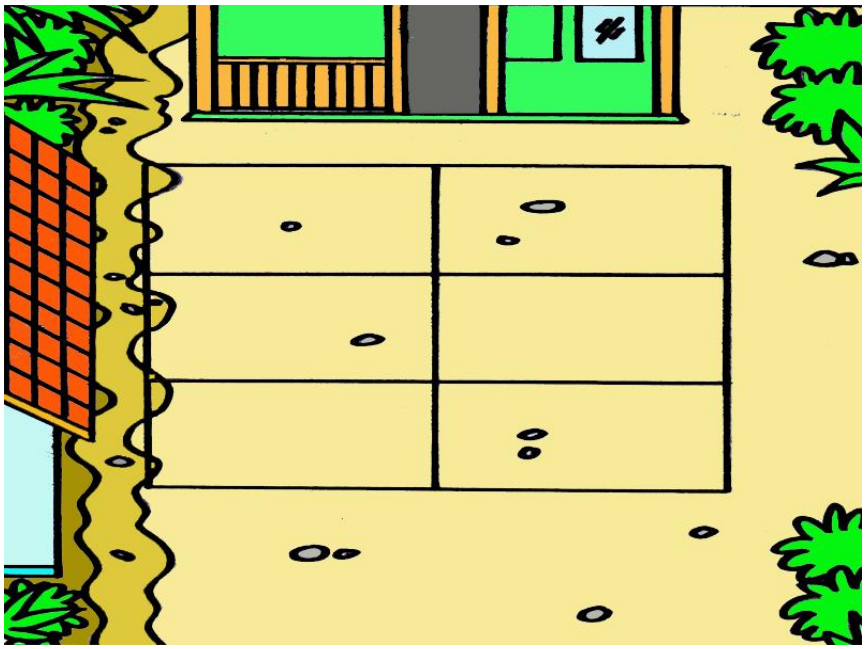
3.2.1.2 Perancangan *Storyboard*

Storyboard merupakan sketsa gambar yang disusun secara berurutan berdasarkan naskah untuk menggambarkan lingkungan pada game. Pada game ini, storyboard merupakan sebuah *map*(peta) yang dijadikan sebagai arena permainan.

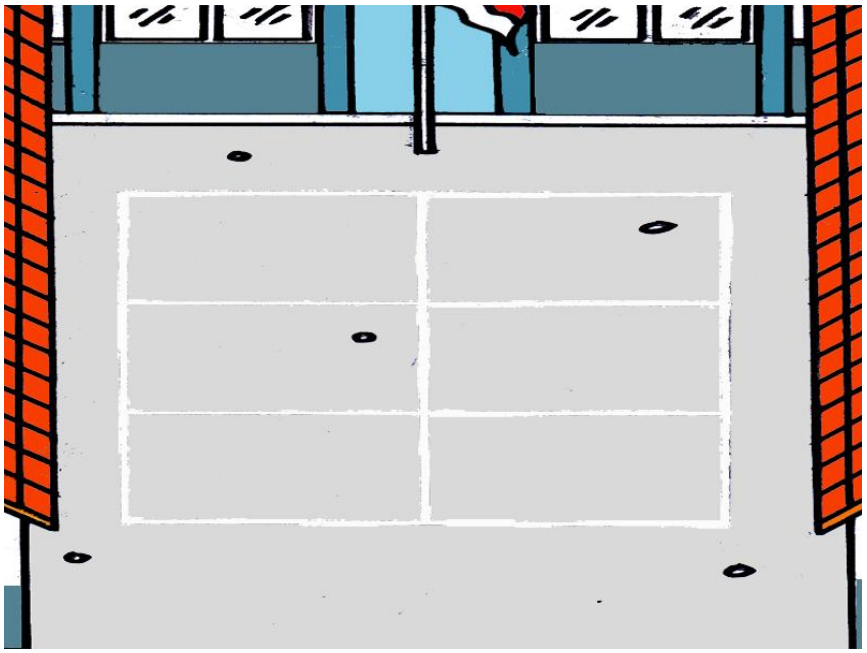
Map pada *game* ini digambarkan pada gambar 3.16 sampai gambar 3.18. Berikut ini merupakan *storyboard* dari aplikasi *game* Permainan Tradisional Galah:



Gambar 3.16 *Storyboard level* 1



Gambar 3.17 *Storyboard level* 2

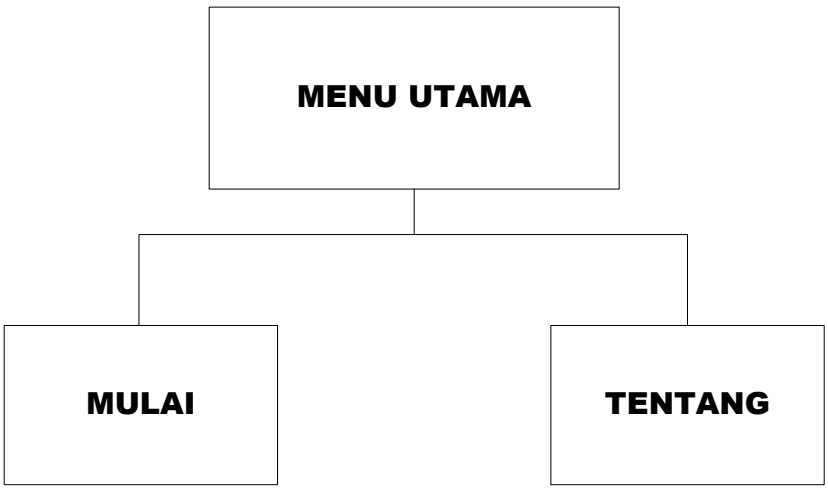


Gambar 3.18 *Storyboard level* 3

Map merupakan tempat yang digunakan pemain untuk memainkan permainan. Enam kotak yang ada merupakan tempat utama untuk pemain dan musuh memainkan perinan, pemain harus dapat berjalan diatas 6 kotak yang ada secara vertikal dan horizontal. Sementara musuh bergerak di atas garis kolom yang. Diatas terdapat beberapa kotak untuk melihat scpre pada kotak atas kanan dan waktu permainan pada kotak atas kiri.

3.2.2 Perancangan Struktur Menu

Untuk mempermudah dalam pembuatan sistem, diperlukan perancangan struktur menu program yang akan dibangun. Perancangan struktur menu program ini membantu dalam merancang bagian-bagian dari sistem yang sebenarnya dan untuk mengetahui bagian mana yang nantinya akan diakses terlebih dahulu setelah program selesai dibuat. Perancangan arsitektur menu pada aplikasi *game* Permainan Tradisional Galah dapat dilihat pada gambar 3.19.



Gambar 3.19 Struktur Menu

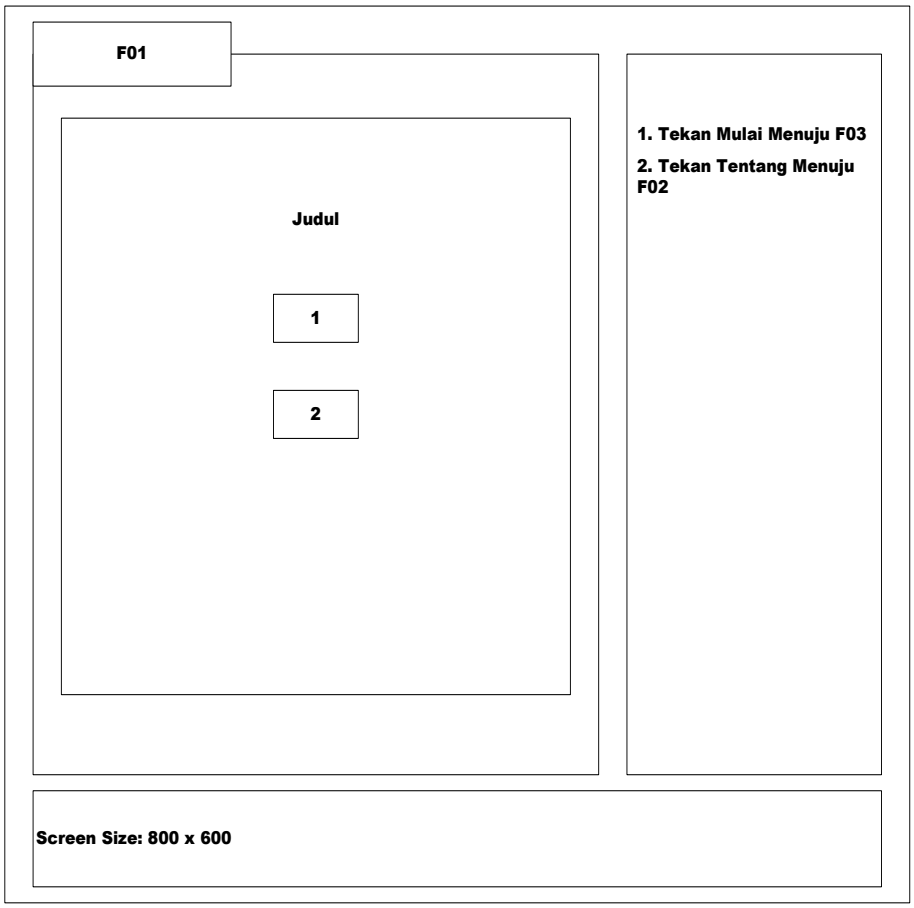
A. **Mulai** : Merupakan submenu untuk memulai permainan baru.

B. **Tentang**: Merupakan submenu mengenai pembuatan aplikasi.

3.2.3 Perancangan Antarmuka

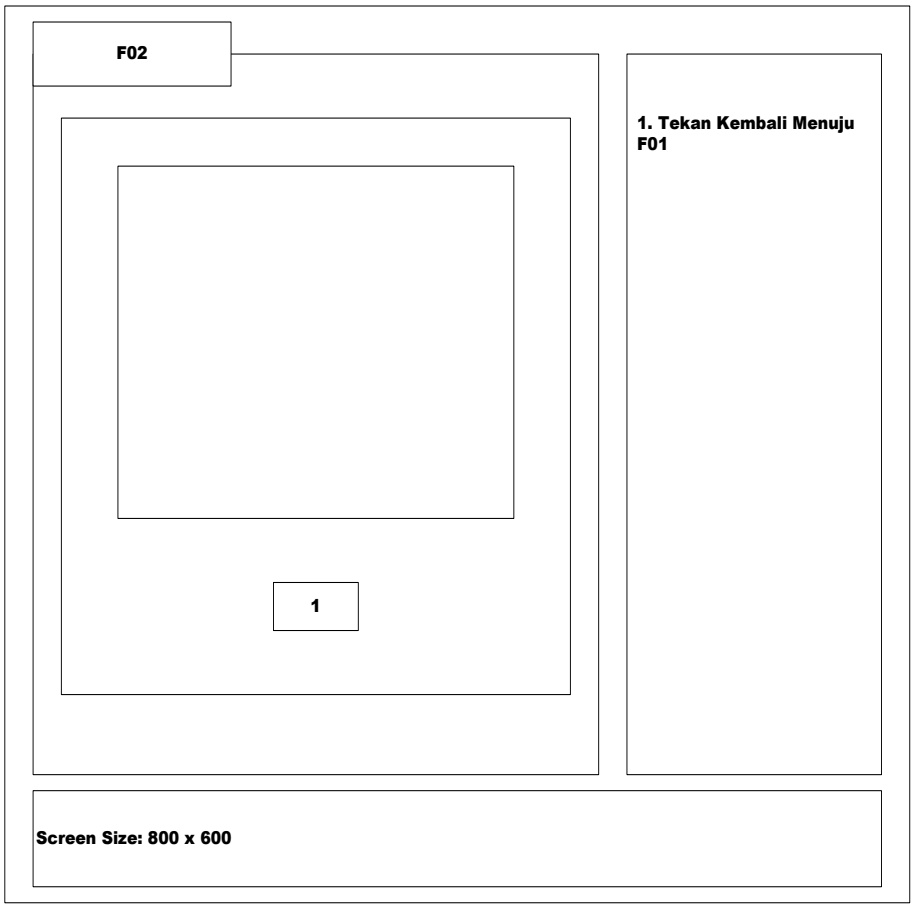
Perancangan antarmuka merupakan salah satu bagian penting dalam perancangan sistem karena nantinya antarmuka tersebut akan menjadi fasilitas yang menjembatani interaksi manusia dengan sistem. Perancangan antarmuka pada *game* ini digambarkan seperti berikut.

1. Perancangan Antarmuka Menu Utama



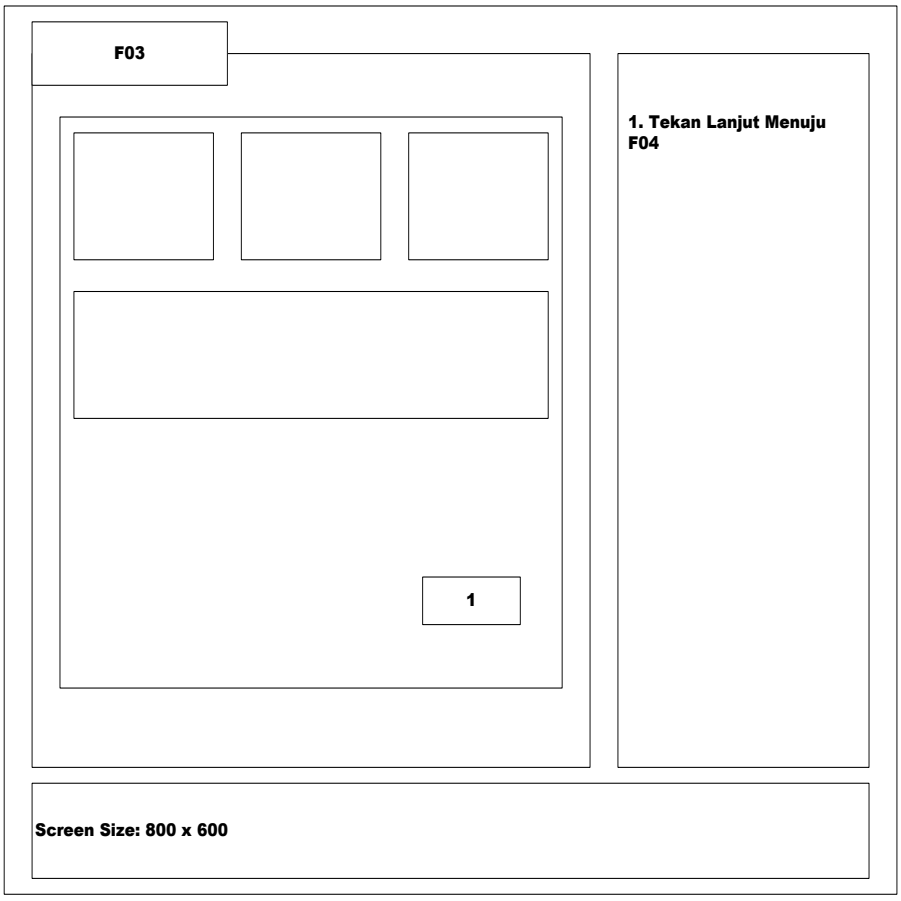
Gambar 3.20 (F01) Perancangan Antarmuka Menu Utama

2. Perancangan Antarmuka Tentang



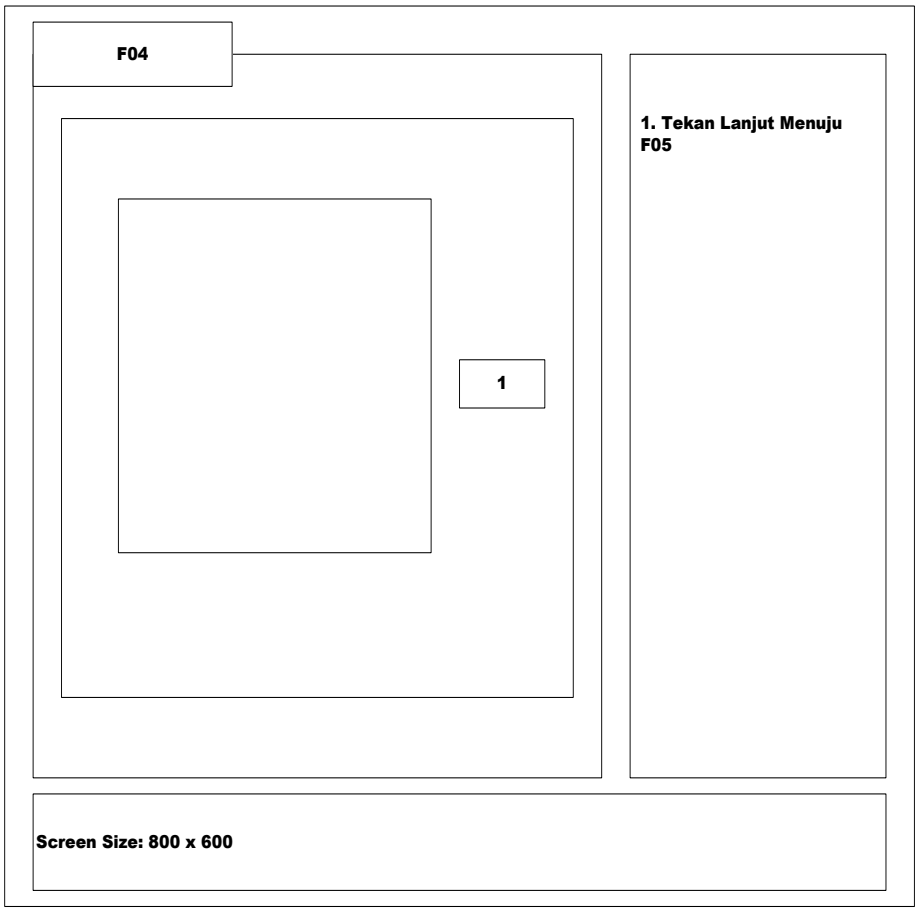
Gambar 3.21 (F02) Perancangan Antarmuka Tentang

3. Perancangan Antarmuka Intro



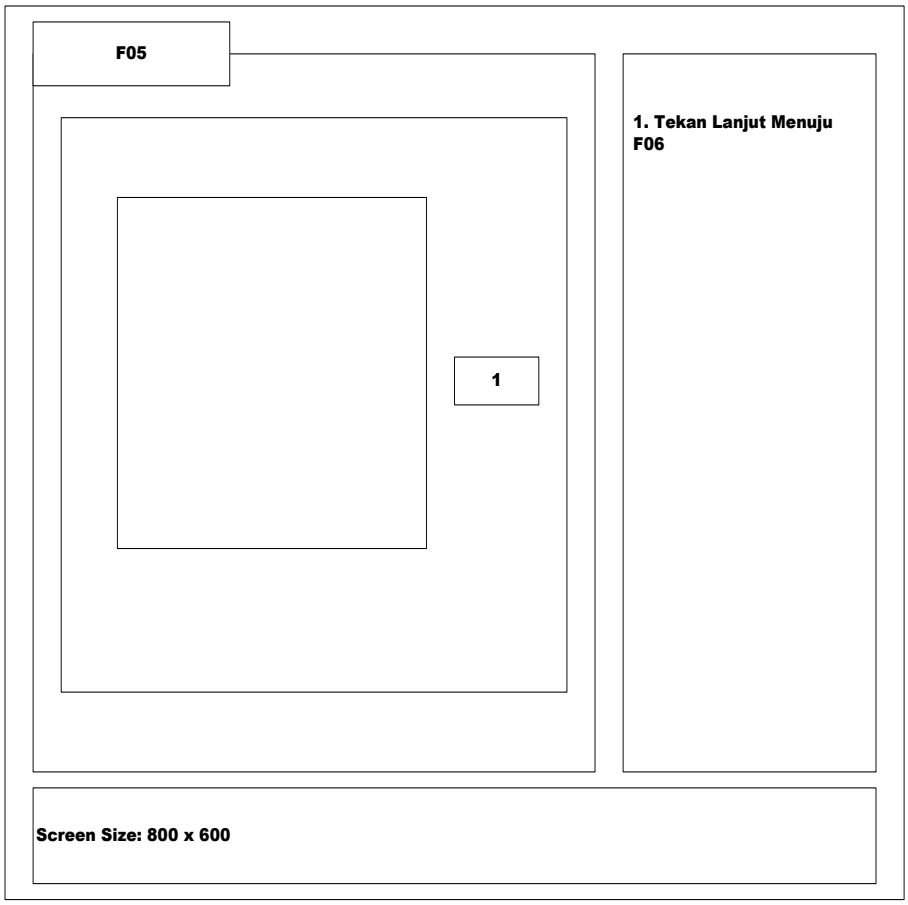
Gambar 3.22 (F03) Perancangan Antarmuka Intro

4. Perancangan Antarmuka Petunjuk 1



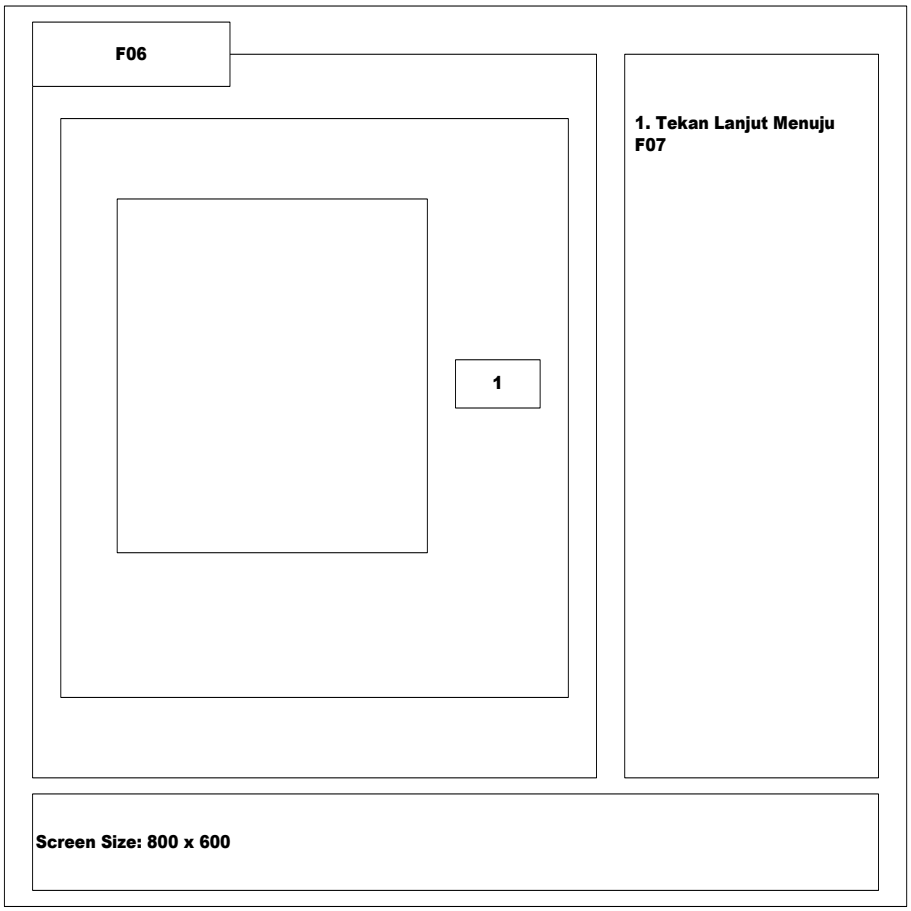
Gambar 3.23 (F04) Perancangan Antarmuka Petunjuk 1

5. Perancangan Antarmuka Petunjuk 2



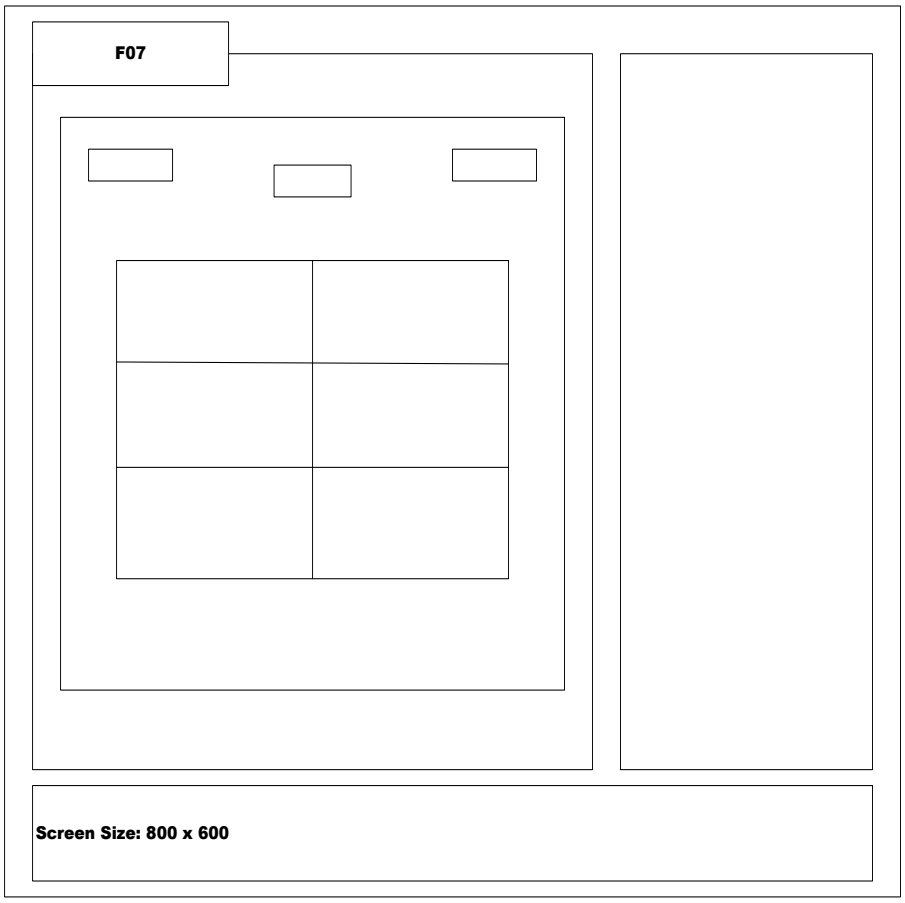
Gambar 3.24 (F05) Perancangan Antarmuka Petunjuk 2

6. Perancangan Antarmuka Cara Main



Gambar 3.25 (F06) Perancangan Antarmuka Cara Main

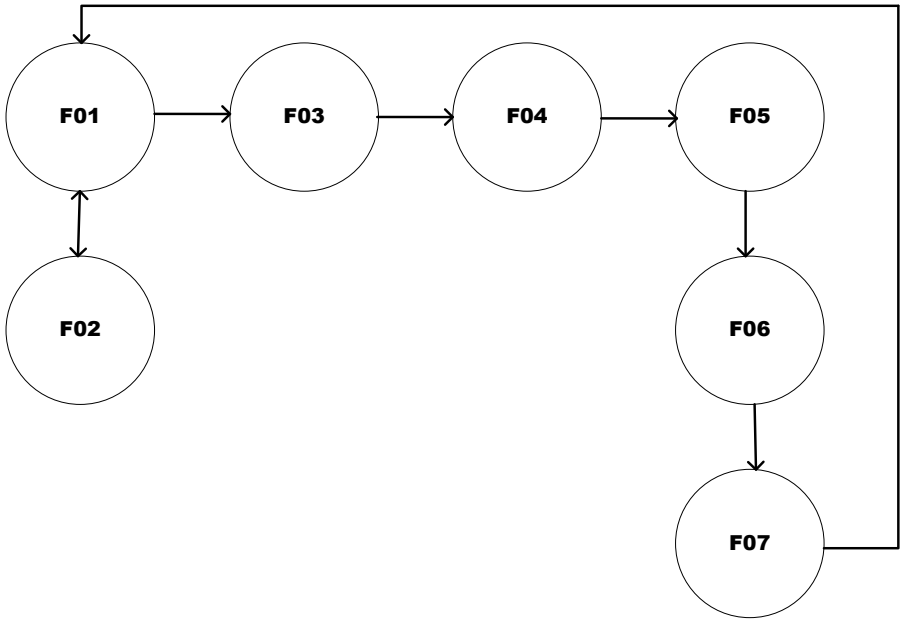
7. Perancangan Antarmuka Main



Gambar 3.26 (F07) Perancangan Antarmuka Main

3.2.4 Jaringan Semantik

Jaringan semantik merupakan alat yang mangkus untuk merepresentasikan pemetaan data, yang bertujuan mencegah terjadinya duplikasi data. Berikut ini adalah jaringan semantik pada *game* ini.



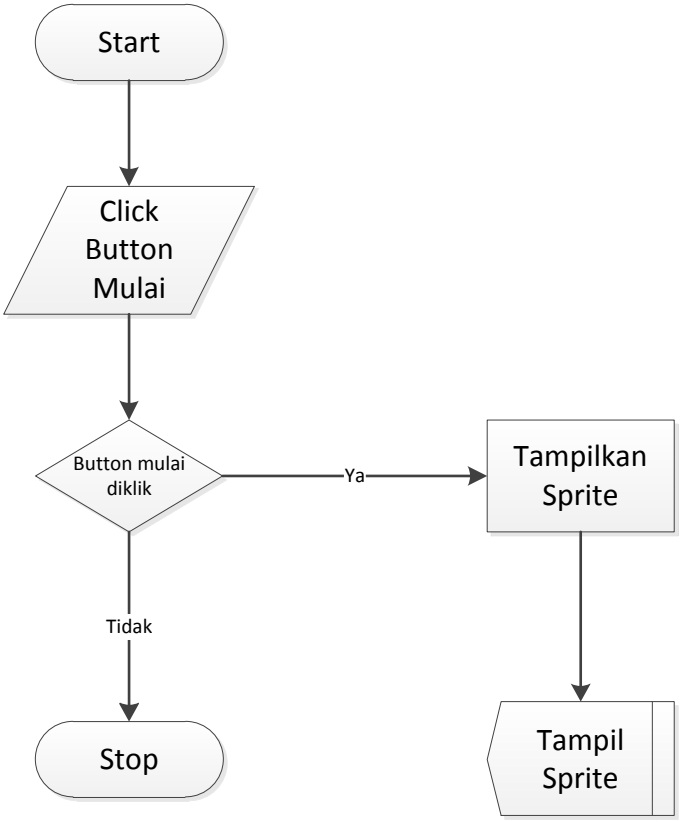
Gambar 3.27 Jaringan Semantik

3.2.5 Perancangan *Method*

Perancangan *method* merupakan pemaparan proses pada *method* yang digunakan pada setiap kelas. Perancangan method digambarkan dengan menggunakan *flow chart*.

1. *Method Controls*()

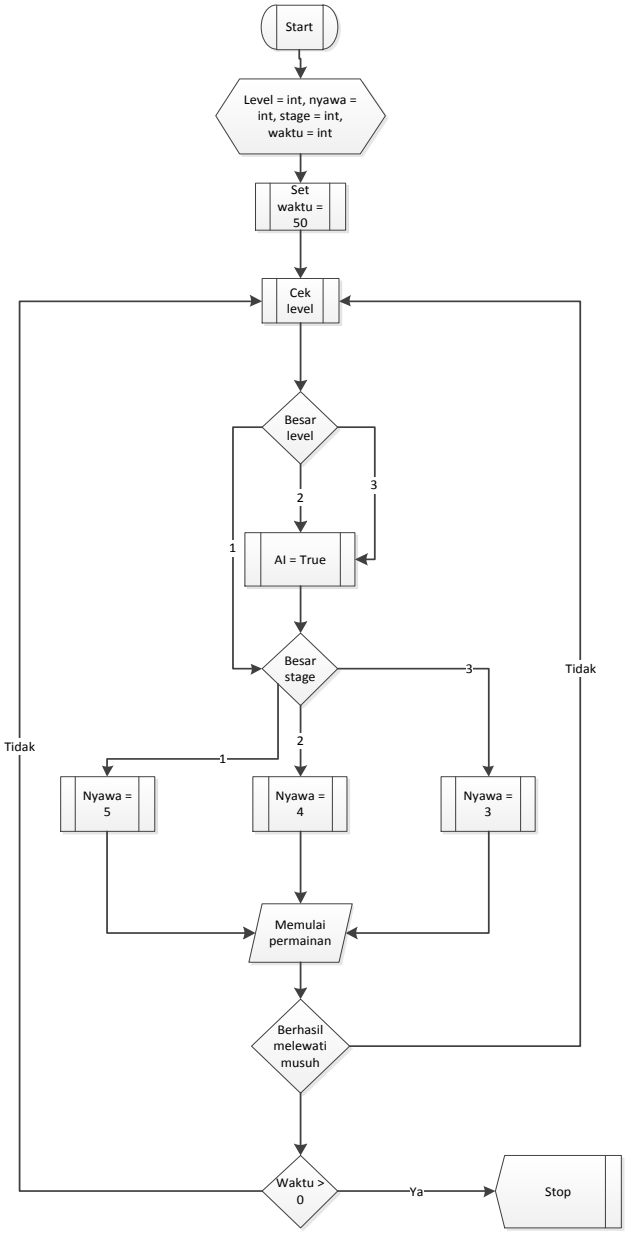
Method controls() dipanggil untuk menggerakkan objek karakter, berikut ini adalah gambar *method control*.



Gambar 3.28 Perancangan *Method Control*

2. *Method Play Game*

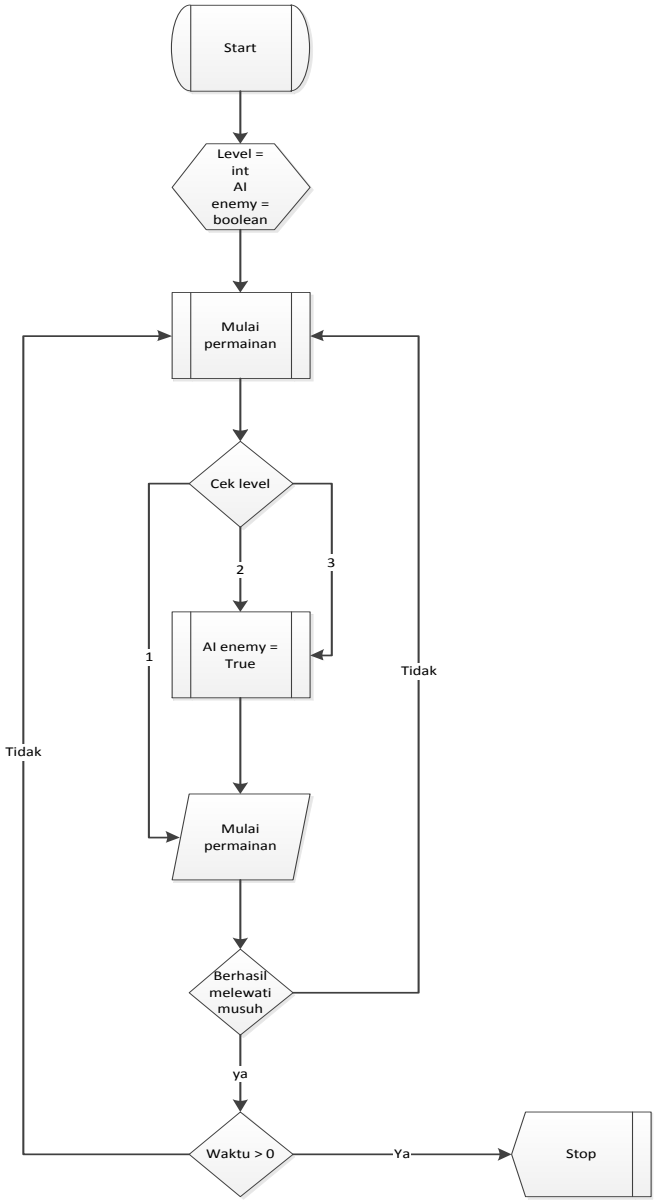
Berikut ini adalah *method play game* yang dipanggil ketika pemain sedang memainkan permainan.



Gambar 3.29 Perancangan *Method Play Game*

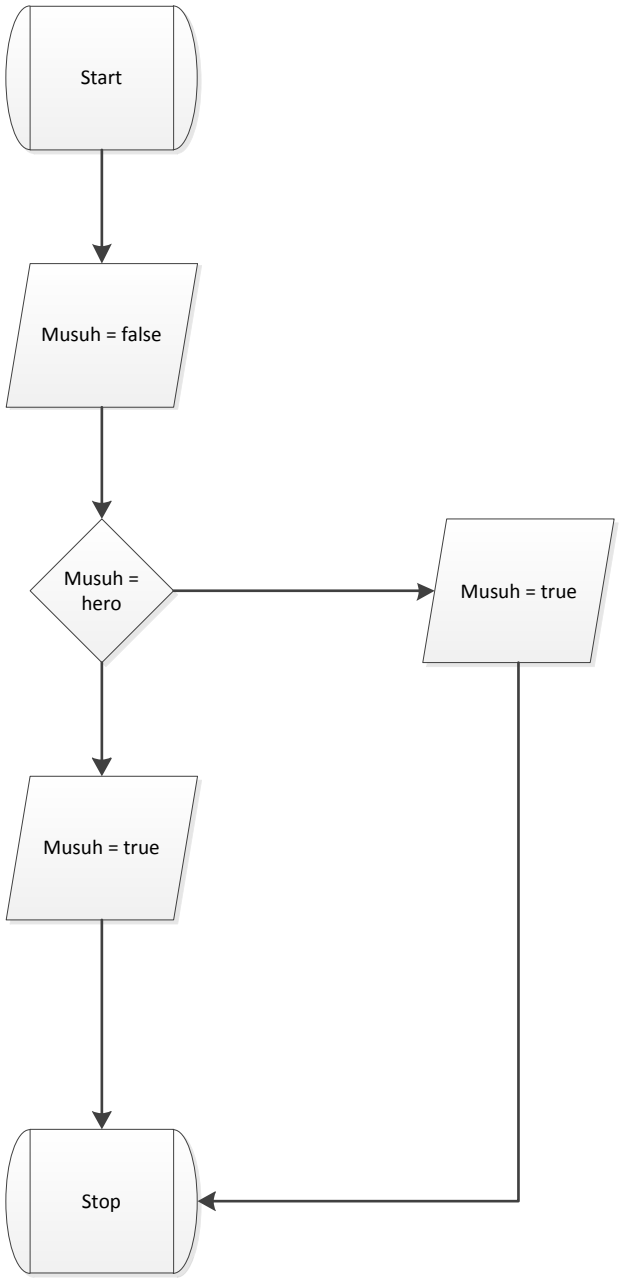
3. *Method Enemy*()

Method enemy() ini dipanggil ketika pemain memulai permainan di setiap *level*.



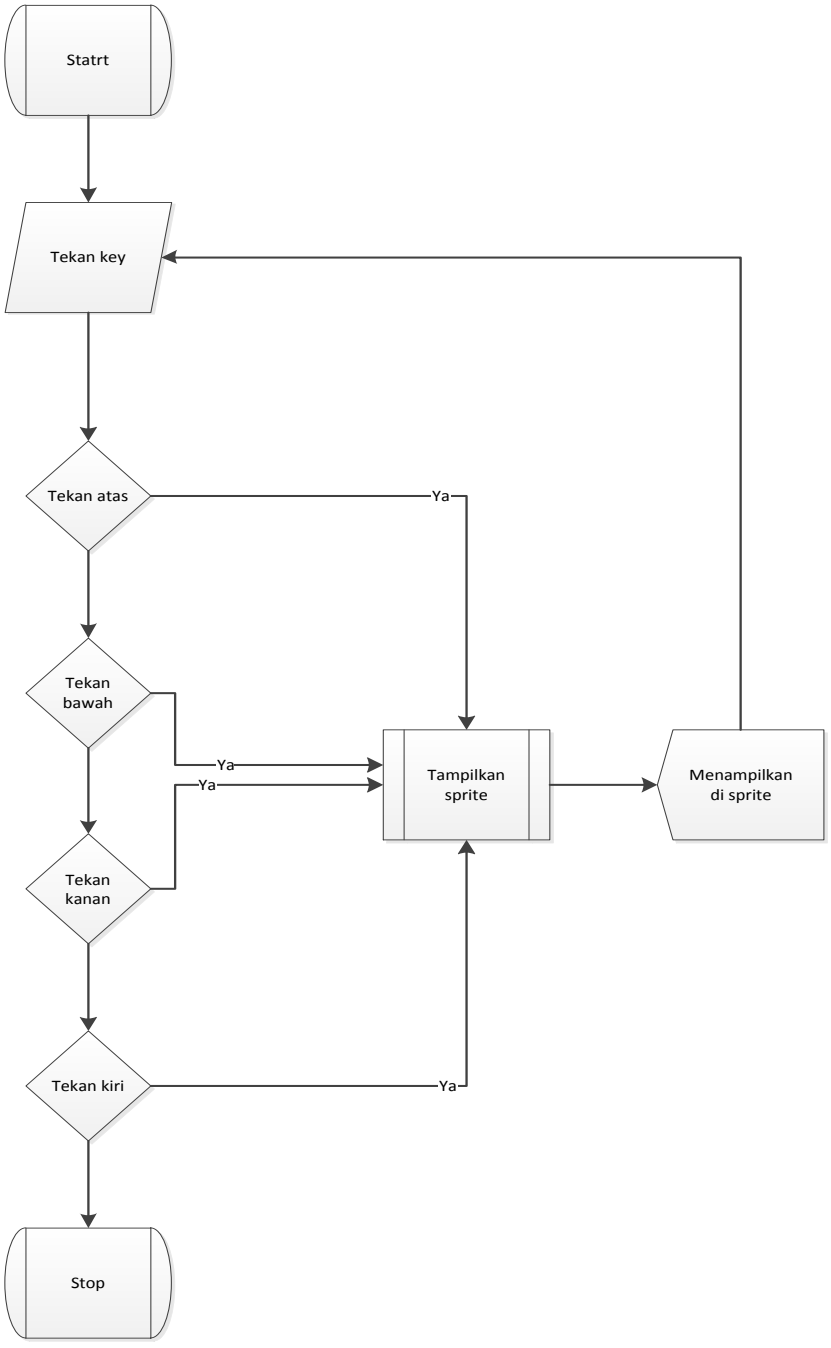
Gambar 3.30 Perancangan *Method Enemy*

4. *Method Collision*()



Gambar 3.31 Perancangan *Method Collision*

5. *Method Input Keyboard*()



Gambar 3.32 Perancangan *Input Keyboard*