BAB 3
ANALISIS DAN PERANCANGAN SISTEM

3.3 Analisis Sistem

Analisis sistem merupakan kegiatan penguraian suatu sistem informasi yang utuh dan nyata ke dalam bagian-bagian atau komponen-komponen komputer yang bertujuan untuk mengidentifikasi serta mengevaluasi masalah-masalah yang muncul, hambatan-hambatan yang mungkin terjadi dan kebutuhan-kebutuhan yang diharapkan sehingga mengarah kepada solusi dengan kebutuhan serta perkembangan teknologi.

3.3.1 Analisis Masalah

Adanya beberapa faktor pada *game*simulasi bercocok tanam menimbulkan beberapa masalah, diantaranya adalah bagaimana menerapkan tanaman hidup atau mati, hama, cuaca, sistem penanggalan, dan karakter utama pada *game* simulasi berccok tanam?

Solusi untuk memecahkan masalah tersebut adalah dengan membangun sebuah *game* yang didalamnya terdapat tanaman yang memiliki kondisi dimana tanaman dapat tetap hidup atau mati,NPC hama dengan diterapkan kecerdasan buatan dengan algoritma koloni semut untuk mencari tanaman yang akan diserang, kondisi perubahan cuaca yang dapat mempengaruhi tanaman seperti cuaca hujan dapat memberikan sumber air untuk tanaman, sistem penanggalan yang dapat menunjukan waktu tumbuh tanaman dengan jelas dan dapat mengatur waktu penanaman dengan baik, dan karakter utama yang dapat digerakkan untuk dapat melakukan aksi dalam bercocok tanam.

3.3.2 Analisis *Game* Sejenis

Analisis yang dilakukanmerupakan observasi dari game sejenis untuk melihat cara bermain, tujuan dan ada tidaknya kekurangan atau kelebihan game itu sendiri.Hasil observasi tersebut untuk membantu dalam penelitian.*Game* sejenis yang dianalisis yaitu *Farmerama, MyFreeFarm,* dan *MyWonderfullFarm*.

3.3.2.1 *Farmerama*

Farmerama adalah *game* simulasi *online*.Sistem menyediakan ladang untuk bercocok tanam untuk pemain pada awal permainan.*Game*ini tidak memiliki karakter utama yang dapat digerakkan oleh pemain karena menggunakan media interaksi *mouse*untuk menggerakkan penunjuk. Tanaman yang tumbuh memiliki beberapa kondisi yang berpengaruh pada poin permainan.Tanaman tumbuh dengan durasi waktu nyata akan tetapi tidak terlihat adanya tanaman yang mati. *Game* ini tidak memiliki hama yang menyerang tanaman.Tujuan *game* ini hanya mengatur lahannya seindah mungkin. Tampilan *gameFarmerama* dapat dilihat pada gambar 3.1.

Beberapa hal yang tidak terdapat pada game ini adalah :

1. Karakter utama yang dapat digerakkan oleh pemain.
2. Hama yang menyerang tanaman.
3. Tidak terlihat adanya tanaman yang mati.



Gambar 3.1 *Farmerama* [13]

3.3.2.2 *My Free Farm*

My Free Farm adalah *game* simulasi *online*.Sistem menyediakan ladang untuk bercocok tanam untuk pemain pada awal permainan.*Game*ini tidak memiliki karakter utama yang dapat digerakkan oleh pemain karena menggunakan media interaksi *mouse*untuk menggerakkan penunjuk. Tanaman yang tumbuh memiliki beberapa kondisi yang berpengarup pada poin permainan.Tanaman tumbuh dengan durasi waktu nyata akan tetapi tidak terlihat adanya tanaman yang mati. *Game* ini tidak memiliki hama yang menyerang tanaman. Tujuan *game* ini hanya mengatur lahannya seindah mungkin. Tampilan *gameMy Free Farm* dapat dilihat pada gambar 3.2.

Beberapa hal yang tidak terdapat pada game ini adalah :

1. Karakter utama yang dapat digerakkan oleh pemain.
2. Hama yang menyerang tanaman.
3. Tidak terlihat adanya tanaman yang mati.



Gambar 3.2 *My Free Farm* [14]

3.3.2.3 *My Wonderful Farm*

My Wonderfull Farm adalah *game* simulasi *online*.Sistem menyediakan ladang untuk bercocok tanam untuk pemain pada awal permainan.*Game*ini memiliki karakter utama,tetapi bergerak sesuai tujuan yang dipilih oleh pemain dengan menggunakan media interaksi *mouse*untuk menggerakkan penunjuk untuk menentukan tujuan. Tanaman yang tumbuh memiliki beberapa kondisi yang berpengarup pada poin permainan.Tanaman tumbuh dengan durasi waktu nyata akan tetapi tidak terlihat adanya tanaman yang mati. *Game* ini tidak memiliki hama yang menyerang tanaman. Tujuan *game* ini menanam tanaman yang diminta dalam misi permainan.Tampilan*gameMy Free Farm* dapat dilihat pada gambar 3.3.

Hal yang tidak terdapat dalam game ini sebagai berikut :

1. Hama yang menyerang tanaman.
2. Tidak terlihat tanaman mati.



Gambar 3.3 *My Wonderful Farm* [15]

3.3.3 Analisis *Game* yang akanDibangun

Analisis *game* yang akan dibangun merupakanbagian yang mendeskripsikan *game* yang akan dibangun. Pada bagian ini terdiri dari arsitektur, *storyline*, tingkat kesulitan, *gameplay*

3.3.3.1 Arsitektur

Aplikasi ini dibangun dalam bentuk dua dimensi (2D), serta memiliki *fitur* seperti pada Tabel 3.1.

Tabel 3.1 *FiturGame*

FiturGame
1. Sistem *single player*. 2. Terdapat 2 cuaca, yaitu cerah dan hujan. 3. Terdapat hama. 4. Terdapat 3 alat yang dapat digunakan. 5. Terdapat karakter utama, 2 NPC (*Non-Playable Character*) manusia, 1 NPC hama. 6. Terdapat 5 arena permainan. 7. Terdapatsimpan dan lanjut permainan.

3.3.3.2 *Storyline*

Storyline merupakan cerita pada game dimana pada *storyline* sebuah game akan ditentukan bagaimana desainnya dan aturan permainannya. Pada game ini, pemain akan berperan sebagai seorang pemuda yang diberi lahan perkebunan kakeknya oleh kepala desa di sebuah desa bernama desa bunga. Lahan itu telah ditinggalkan oleh kakek pemuda tersebut, sebenarnya akan dijual untuk dibangun taman bermain apabila tidak ada yang mengurusinya. Karena pemuda itu memiliki kenangan bersama kakeknya di perkebunan tersebut, maka pemuda itu bersedia mengurus lahan perkebunan kakeknya.Kepala desa menyetujuinya dan memberikan beberapa syarat untuk tinggal disana.Apabila pemuda tersebut dapat mengurusi lahan selama sebulan dengan baik dan berprilaku baik dengan warga sekitar, maka lahan maka pemuda itu diijinkan melanjutkan mengurus lahan tersebut.Pemuda tersebut akhirnya merubah kehidupannya dengan bercocok tanam untuk mempertahankan ladang kakenya itu. Berdasarkan cerita yang telah diuraikan maka game ini diberi judul "*Farming Life*".

3.3.3.3 *Gameplay*

*Gameplay*dilakukan untuk menggambarkan aturan-aturan dalam *game*.*Gameplaygame* yang akan dibangun terdapat pada tabel 3.2.

Tabel 3.2 *Gameplay*

Gameplay
1. Karakter utama dapat digerakkan oleh pemain setelah intro dan dimulai dari ladang. 2. Pada awal permainan karakter utama diberi 100 koin dan 3 benih. 3. Koin digunakan sebagai alat tukar pada *game*. 4. Karakter utama memiliki poin stamina terbatas untuk bekerja setiap harinya. 5. Karakter utama diberi 3 alat utama yang dapat dipakai dalam bercocok tanam, yaitu : Cangkul, Siraman, dan Sabit. 6. Karakter utama dapat bergerak empat arah, yaitu : atas, kanan, kiri, bawah. 7. Karakter utama dapat menuju depan rumah untuk melanjutkan ke hari berikutnya dan mengembalikan poin stamina sampai penuh. 8. Poin stamina berkurang apabila karakter utama melakukan aksi. 9. Aksi karakter utama menggunakan alat dan barang, yaitu : menggemburkan tanah, menaburkan benih, menyiram, menyabit tanaman dan memetik hasil panen. 10. Karakter utama dapat berbicara dengan NPC Thomas setiap harinya untuk mendapatkan koin tambahan. 11. Karakter utama dapat berbicara dengan NPC Kim untuk membeli benih dan pestisida. 12. Pemain diberi batasan waktu selama 30 hari pada *game* untuk menyelesaikan *game*. 13. Sepuluh menit pada *game* berlangung selama 10 detik. 14. Waktu pada *game*akan berhenti ketika membuka menu permainan. 15. Tanah yang dapat dipakai berkebun hanya tanah yang berwana coklat yang berada di ladang. 16. Tanah harus digemburkan dahulu dengan cangkul sehingga tanah dapat dipakai untuk menanam. 17. Tanah yang sudah digemburkan dan tidak ditanami bibit keesokan harinya akan menjadi rata kembali. 18. NPC Thomas dan Kim akan muncul pada pukul 8 sampai pukul 18 pada *game*. 19. NPC hama akan muncul ketika tamanan tumbuh dan menyerangnya. 20. Tanaman yang diserang hama akan mati keesokan harinya apabila tanaman tersebut tidak segera diberi pestisida. 21. Tanaman yang tidak disiram akan mati pada keesokan harinya. 22. Tanaman yang sudah mati harus disabit untuk menghilangkannya. 23. Tanaman yang telah panen tidak akan mati keesokan harinya.

24. Memetik hasil panen akan menambah koin.
25. *Rating game* didapat dari perhitungan banyaknya hasil panen.
26. *Rating game* menentukan bagian akhir dalam *game*.
27. *Rating game* baik akan memperbolehkan pemain melanjutkan permainan tanpa batasan waktu.
28. *Rating game* buruk akan menghentikan permainan.

3.3.3.4 Misi

Misi yang harus diselesaikan yaitu mengumpulkan poin *ratinggame*sebanyak mungkin.Rating game menentukan layak tidaknya pemain untuk melanjutkan permainan.Terdapat 2 kondisi pada akhir permainan, yaitu permainan dilanjutkan tanpa adanya batasan waktu atau permainan dihentikan.

3.3.4 Sasaran Pemain dan Kegunaan*GameFarming Life*

Sasaran pemain menentukan layak tidaknya pemain yang memainkan *game* ini. Sasaran pemain yang layak memiliki kriteria :

1. Umur 12tahun ke atas,karena *game* tidak memiliki tindak kekerasan.
2. Dapat menggunakan komputer, karena game berbentuk aplikasi *desktop*.
3. Tidak buta total maupun tidak buta warna, karena *game* memiliki perbedaan warna yang berpengaruh dalam *game*.

Kegunaan *game* menunjukan kegunaan *game* untuk pemain. Kegunaan dalam *game* ini untuk meningkatkan daya pikir pemain dalam segi memanajemen waktu dan sebagai sarana hiburan karena pemain bebas bermain sesuai keinginan mereka untuk dapat menyelesaikan misi permainan.

3.3.5 Analisis Algoritma

Analisis algoritma yang dilakukan dalam penelitian ini adalah meneliti bagaimana cara kerja algoritma koloni semutpada NPC hama di dalam *game*. Kondisi dimana hama muncul berdasarkan kondisi tanaman yang tumbuh.

3.3.5.1 Penerapan Algoritma Koloni Semut

Algoritma koloni semut adalah algortima yang dapat mencari dan menentukan jalur terbaik dari titik awal sampai titik akhir. Cara kerja algoritma koloni semut yang diterapkan pada NPC hamauntuk mencari tanaman yang akan diserang.Kondisi tanaman menentukan hama akan muncul atau tidak.

Pada game ini, parameter yang digunakan untuk menjalankan algoritma yaitu terdiri dari lima parameter yaitu sasaran sebagai kondisi tanaman, hama sebagai semut, tanaman sebagai sumber makanan, sarang sebagai sarang semut dan jejak hama sebagai jejak semut.

Secara keseluruhan algoritma ini terbagi menjadi beberapa bagian yaitu *Initialization*, *Construction*, *Trails Update*, *Terminating Condition*.Penerapan bagian-bagian tersebut pada game ini adalah sebagai berikut:

1. *Initialization*

   Proses dimana menentukan nilai awal posisi sarang dan posisi tanaman
2. *Construction*

   Proses pencarian jalur ke tujuan yaitu jalur dari posisi sarang sampai posisi tanaman dengan carameletakkan hama pada posisi sarang lalu menyebarkan hama secara acak.
3. *Trails Update*

   Proses dimana hamayang disebar sudah menemukan posisi tanaman. Hama kembali ke posisi sarang dari posisi tanaman dengan meninggalkan jejak hama.
4. *Terminating Condition*

   Proses penutupan algoritma bila kondisi telah terpenuhi, jika belum kembali ke bagian 2.

Alur dari algortima koloni semut pada gambar 3.4 yang diterapkan pada NPC hama dalam game ini sebagai berikut:

1. Periksa kondisi tanaman.

   Langkah pertama adalah memeriksa apakah sasaran memiliki status *true* yang memiliki arti tanaman dapat diserang atau *false* yang memiliki arti tanaman tidak dapat diserang. Jika ya, maka akan lanjut ke langkah kedua. Jika tidak, maka hama tidak akan muncul.
2. Inisialisasi

   Langkah ini merupakan proses inisialisasi posSarang merupakan posisi sarang, posTanaman merupakan posisi tanaman yang dapat diserang dan posJejakHama merupakan posisi jejak hama.

3. Perulangan sebarkan hama

   Langkah ini merupakan proses menyebarkan hama sampai batas jumlah hama yang akan disebar.Hama yang disebar diposisikan pada posisi sarang.

4. Perulangan cek posisi

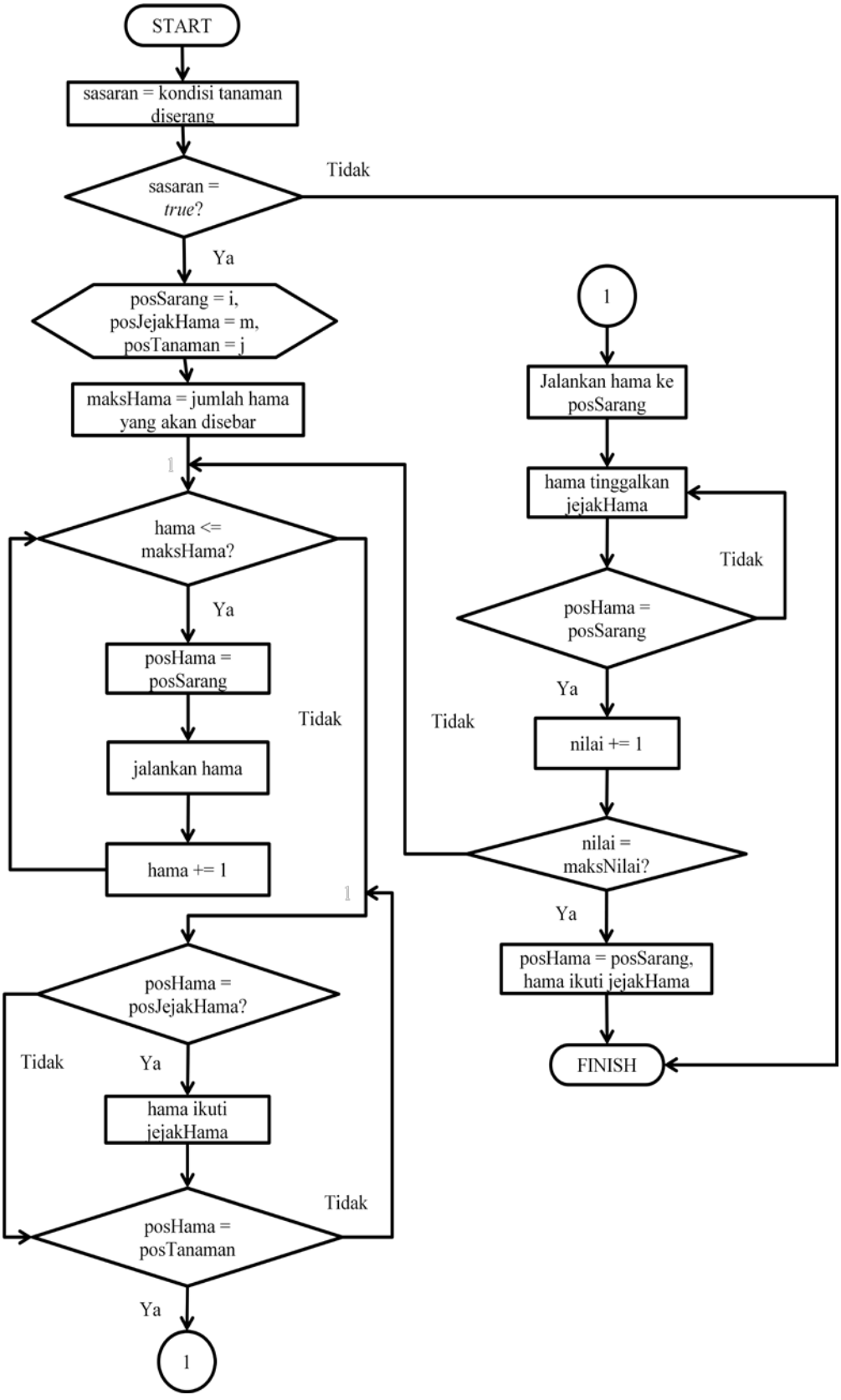
   Langkah ini merupakan proses pengecekan setiap hama yang disebar apakah menemukan jejak. Jika menemukan jejakHama maka hama tersebut mengikuti jejakHama sampai ke posisi tanaman.Jika tidak, maka hama melakukan pengecekan setaip hama yang disebar menemukan posisi tanaman. Jika ya, makalanjut ke langkah ke-5 dan jika tidak, maka ulangi langkah ke-4.

5. Pemberian jejakHama

   Langkah ini merupakan proses pemberian jejakHama bila hama sudah menemukan posisi tanaman maka hama dijalankan ke posisi sarang dengan meninggalkan jejakHama.

6. Pengecekan kondisi penutup

   Langkah ini merupakan proses pengecekan ketika hama telah kembali ke posisi sarang. Pengecekan apakah nilai sudah mencukupi. Jika ya, maka hama dijalankan dari posisi sarang dengan mengikuti jejakHama. Jika tidak, kembali ke langkah ke-3.



Gambar 3.4 *Flowchart*Penerapan Algoritma Koloni Semut

3.3.6 Analisis Kebutuhan Non-Fungsional

Analisis kebutuhan non-fungsional dilakukan untuk mengetahui spesifikasi kebutuhan untuk sistem.Spesifikasi kebutuhan melibatkan analisis perangkat keras/*hardware*, analisis perangkat lunak/*software*, analisis perangkat pikir/*user*.

3.3.6.1 Analisis Kebutuhan Perangkat Keras

Analisis kebutuhan perangkat keras digunakan untuk mengetahui spesifikasi minimun perangkat keras dalam membangun *gameFarming Life*pada tabel 3.3 dan menjalankan *gameFarming Life*pada tabel 3.4.

Tabel 3.3 Spesifikasi Perangkat Keras Pembangun

Nama Perangkat	Spesifikasi
Prosesor	1.80Ghz
RAM	4GB
Harddisk	50GB
Display	Resolusi 1366 x 768
Keyboard	Standar
Mouse	Standar

Tabel 3.4 Spesifikasi Perangkat Keras Pengguna

Nama Perangkat	Spesifikasi
Prosesor	1.80Ghz
RAM	512MB
Harddisk	100MB
Display	Resolusi 800 x 600
Keyboard	Standar
Mouse	Standar

3.3.6.2 Analisis Kebutuhan Perangkat Lunak

Analisis kebutuhan perangkat lunak digunakan untuk mengetahui spesifikasi perangkat lunak dalam menjalankan dan membangun *game Farming Life*, seperti pada tabel 3.5 dan tabel 3.6.

Tabel 3.5 Spesifikasi Perangkat Lunak Pembangun

Nama Perangkat Lunak	Spesifikasi
Sistem Operasi	Microsoft Windows 7
Tools	JDK 6
	Greenfoot 2.2.1
	Adobe Photoshop CS5

Tabel 3.6 Spesifikasi Perangkat Lunak Pengguna

Nama Perangkat Lunak	Spesifikasi
Sistem Operasi	Microsoft Windows XP, Win 7
Tools	JDK 6

3.3.6.3 Analisis Kebutuhan Perangkat Pikir

Analisis kebutuhan perangkat pikir digunakan untuk mengetahui spesifikasi pengguna untuk dapat memainkan *game Farming Life*, seperti pada tabel 3.7.

Tabel 3.7 Spesifikasi Pengguna

Peran	Pemain
Usia	12 tahun ke atas
Tingkat Keterampilan	Dapat menggunakan komputer
Kondisi Fisik	Tidak buta total dan tidak buta warna.

3.3.7 Analisis Kebutuhan Fungsional

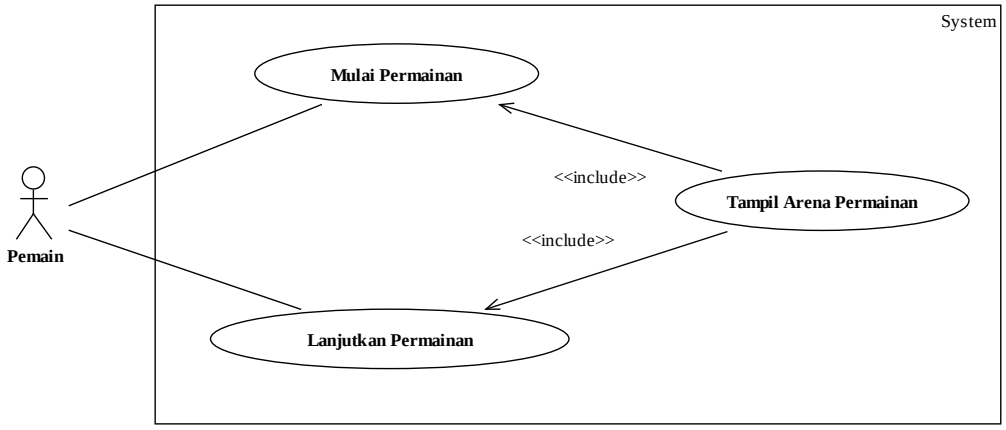
Analisis kebutuhan fungsional dilakukan untuk mengetahui gambaran sistem yang akan dibangun. Analisis kebutuhan fungsional terdiri dari pendefinisan fungsi game, penggambaran alir aktivitas pada game dan penggambaran interaksi antar objek pada game.

3.3.7.1 Pendefinisian Fungsi Game

Pendefinisian fungsi game digunakan untuk menggambarkan fungsi-fungsi pada game. Pendefinisian fungsi pada game ini digambarkan dengan menggunakan *use case diagram*, definisi aktor,definisi*use case* dan skenario *use case*.

1. Use Case Diagram

Penggambaran fungsi game berdasarkan interaksi antar aktor dan objek pada sistem digambarkan dengan menggunakan diagram use case yang terdapat pada Gambar3.5.



Gambar 3.5 ***Use Case Diagram***

2. Definisi Aktor

Definisi aktor berfungsi menjelaskan peran setiap aktor yang terdapat pada use case. Definisi aktor ditunjukan pada Tabel 3.8.

Tabel 3.8 Definisi Aktor

No.	Aktor	Deskripsi
1.	Pemain	User / pengguna / orang yang memainkan game ini

3. Definisi *Use Case*

Definisi *use case* berfungsi menjelaskan tiap-tiap proses dan objek yang terdapat pada *use case*. Definisi *use case* ditunjukan pada Tabel 3.9.

Tabel 3.9 Definisi *Use Case*

No.	Use Case	Deskripsi
1	Mulai Permainan	Proses memulai permaian
2	Lanjutkan Permainan	Proses melanjutkan permainan
3	Tampil Arena Permainan	Proses menampilkan arena permainan

4. Skenario *Use Case*

Skenario *use case* menunjukan proses apa yang terjadi pada setiap bagian di dalam *use case* tersebut, dimana *user* memberikan perintah pada bagian-bagian *use case* dan respon apa yang diberikan oleh sistem kepada *user* setelah *user* memberikan perintahnya. Skenario *use case* ditujukan pada Tabel 3.10 sampai dengan Tabel 3.12.

Tabel 3.10 Skenario *Use Case*Mulai Permainan

Identifikasi		
Nomor	1	
Nama	Mulai Permainan	
Tujuan	Memulai permaian dari awal	
Deskripsi	Proses memulai permainan	
Aktor	Pemain	
Skenario Utama		
Kondisi Awal	Aktor berada pada menu utama	
Aksi Aktor		**Reaksi Sistem**
1. Memilih mulai		2. Menampilkan Intro
		3. Menampilkan Kotak Nama
4. Memasukan Nama		5. Menampilkan Nama
6. Menekan Tombol ENTER		7. Menampilkan Petunjuk
8. Memilih keluar		9. Menampilkan pesan keluar aplikasi
Skenario Tambahan		
Kondisi Awal	Aktor berada di pesan keluar permainan	
1. Memilih Ya		2. Keluar aplikasi
3. Memilih Tidak		4. Menampilkan Menu Utama

Tabel 3.11 Skenario *Use Case*Lanjutkan Permainan

Identifikasi		
Nomor	2	
Nama	Lanjutkan Permainan	
Tujuan	Melanjutkan permainan sebelumnya	
Deskripsi	Proses melanjutkan permainan	
Aktor	Pemain	
Skenario Utama		
Kondisi Awal	Menu Utama ditampilkan	
Aksi Aktor		**Reaksi Sistem**
1. Memilih lanjut		2. Menampilkan Data Permainan
3. Memilih Data		4. Menampilkan Petunjuk
5. Memilih keluar		6. Menampilkan pesan keluar aplikasi
Skenario Tambahan		
Kondisi Awal	Aktor berada di pesan keluar permainan	
1. Memilih Ya		2. Keluar aplikasi
3. Memilih Tidak		4. Menampilkan Menu Utama

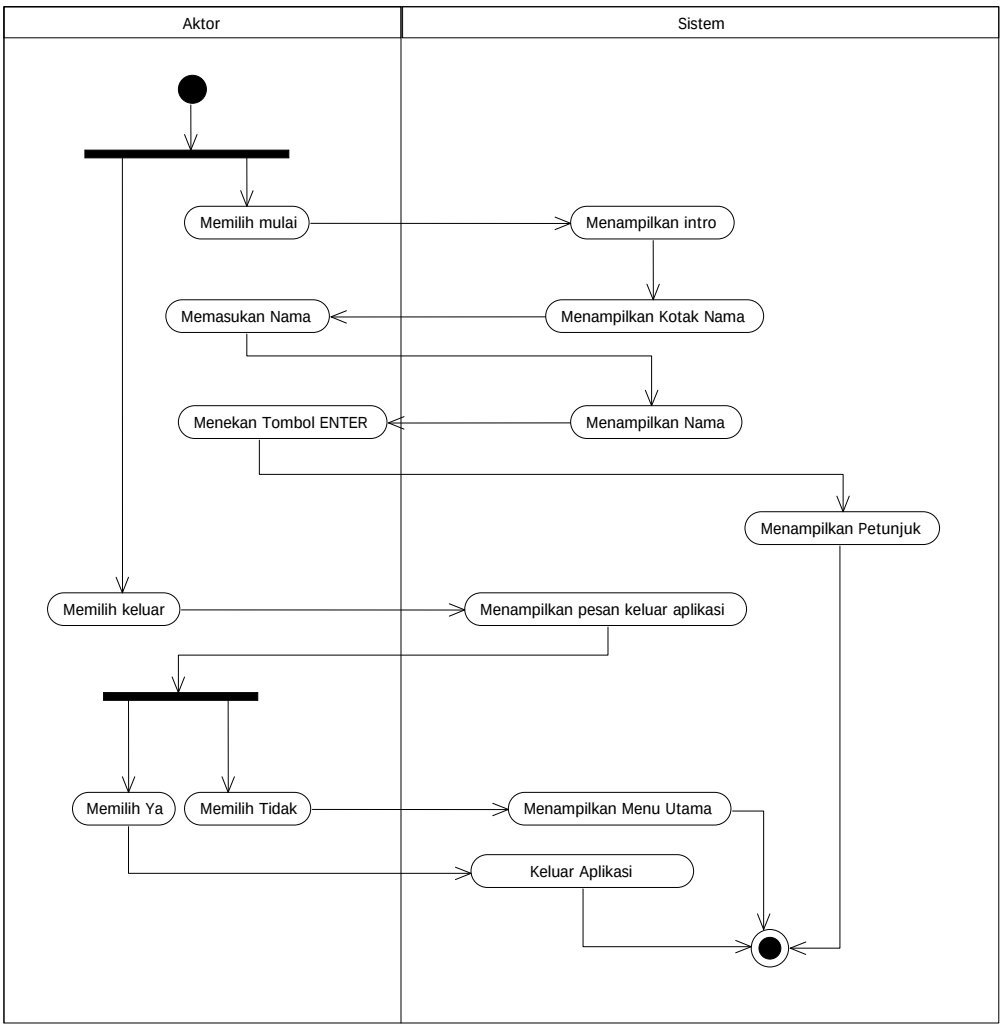
Tabel 3.12 Skenario *Use Case*Tampil Arena Permainan

Identifikasi	
Nomor	3
Nama	Tampil Arena Permainan

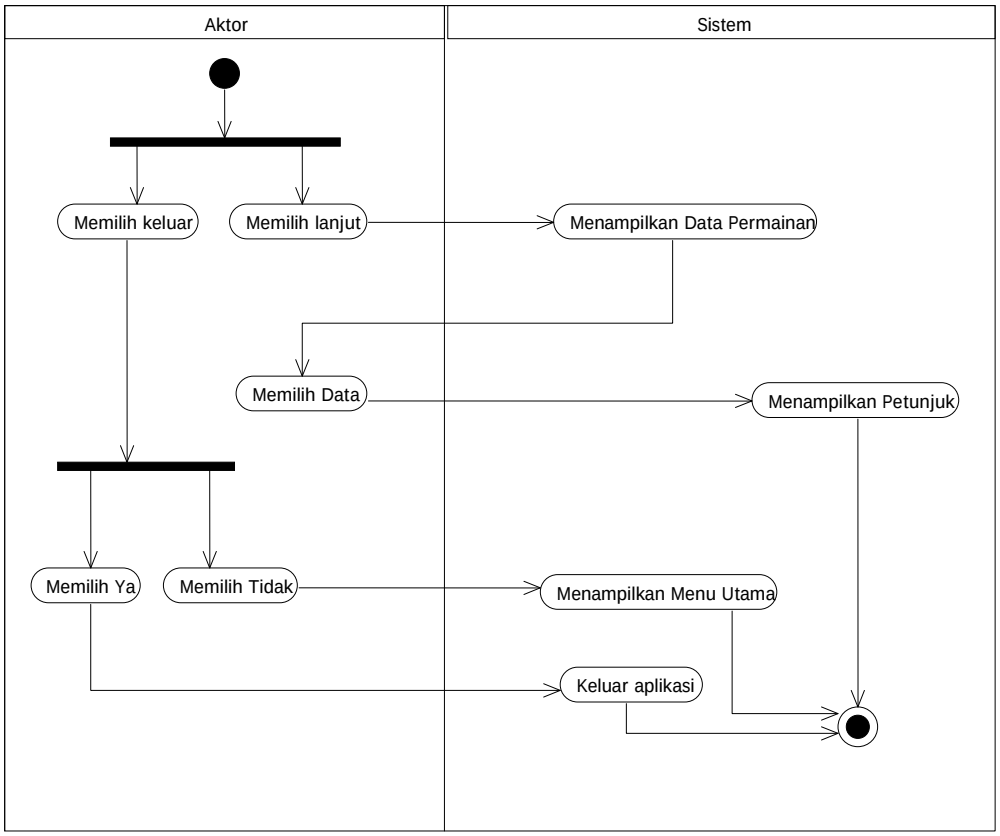
Tujuan	Menampilkan Arena permainan	
Deskripsi	Proses menampilkan arena permainan	
Aktor	Pemain	
Skenario		
Kodisi Awal	Arena permainan ditampilkan	
Aksi Aktor		**Reaksi Sistem**
Skenario Normal		
1. Menggerakan Karakter		2. Menampilkan Karakter Bergerak
3. Menekan Tombol Aksi		4. Menampilkan Aksi
5. Menekan Tombol ENTER		6. Menampilkan Dialog
7. Menekan Tombol ESC		8. Menampilkan Menu Permainan
9. Memilih Status Permainan		10. Menampilkan Status Permainan
11. Memilih Simpan Data		12. Menampilkan Data Permainan
13. Memilih Data		14. Menyimpan Data
15. Memilih Keluar		16. Menampilkan pesan keluar permainan
Skenario Tambahan		
Kondisi Awal	Aktor berada di pesan keluar permainan	
1. Memilih Ya		2. Menampilkan Menu Utama
3. Memilih Tidak		4. Menampilkan Menu Permainan

3.3.7.2 Penggambaran Alir Aktivitas Pada *Game*

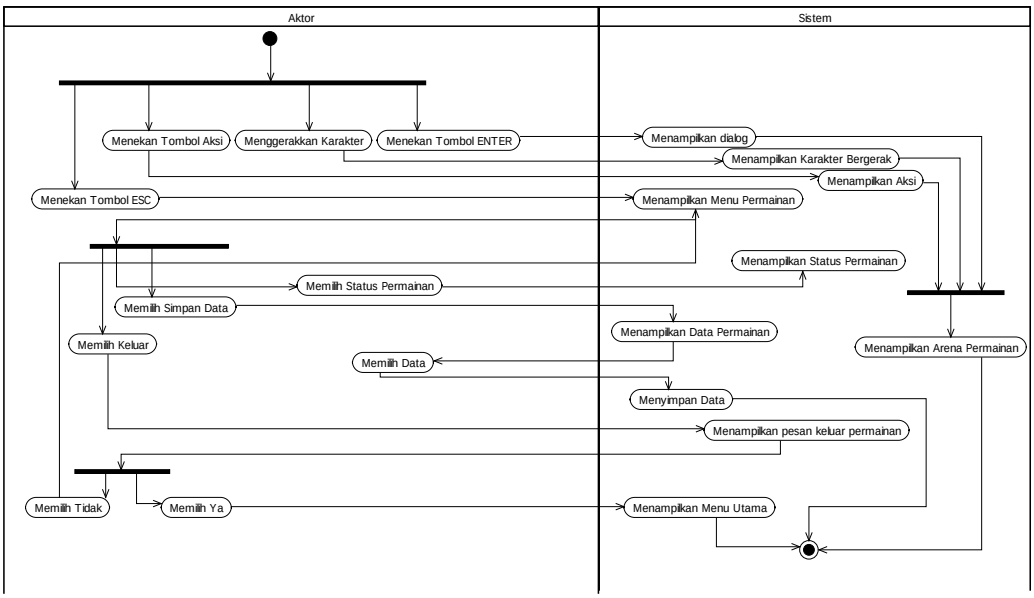
Penggambaran alir aktivitas pada game menggambarkan berbagai alir aktivitas dalam game yang sedang dirancang, bagaimana masing-masing alir berawal, *decision* yang mungkin terjadi, dan bagaimana mereka berakhir. Penggambaran alir aktivitas pada *game* digambarkan menggunakan *Activity Diagram*. *Activity diagram* merupakan *state diagram* khusus, dimana sebagian besar *state* adalah *action* dan sebagian besar transisi di-*trigger* oleh selesainya *state* sebelumnya (*internal processing)*. Oleh karena itu *activity diagram* tidak menggambarkan behavior internal sebuah sistem (dan interaksi antar subsistem) secara eksak, tetapi lebih menggambarkan proses-proses dan jalur-jalur aktivitas dari level atas secara umum. Sebuah aktivitas dapat direalsasikan oleh satu *Use Case* atau lebih. Aktivitas menggambarkan proses yang berjalan, sementara *Use Case* menggambarkan bagaimana actor menggunakan sisitem untuk melakukan aktivitas. *Activity diagram* ditunjukan pada Gambar 3.13 sampai dengan Gambar 3.14.



Gambar 3.6 *Activity Diagram*Mulai Permainan



Gambar 3.7 *Activity Diagram*Lanjutkan Permainan



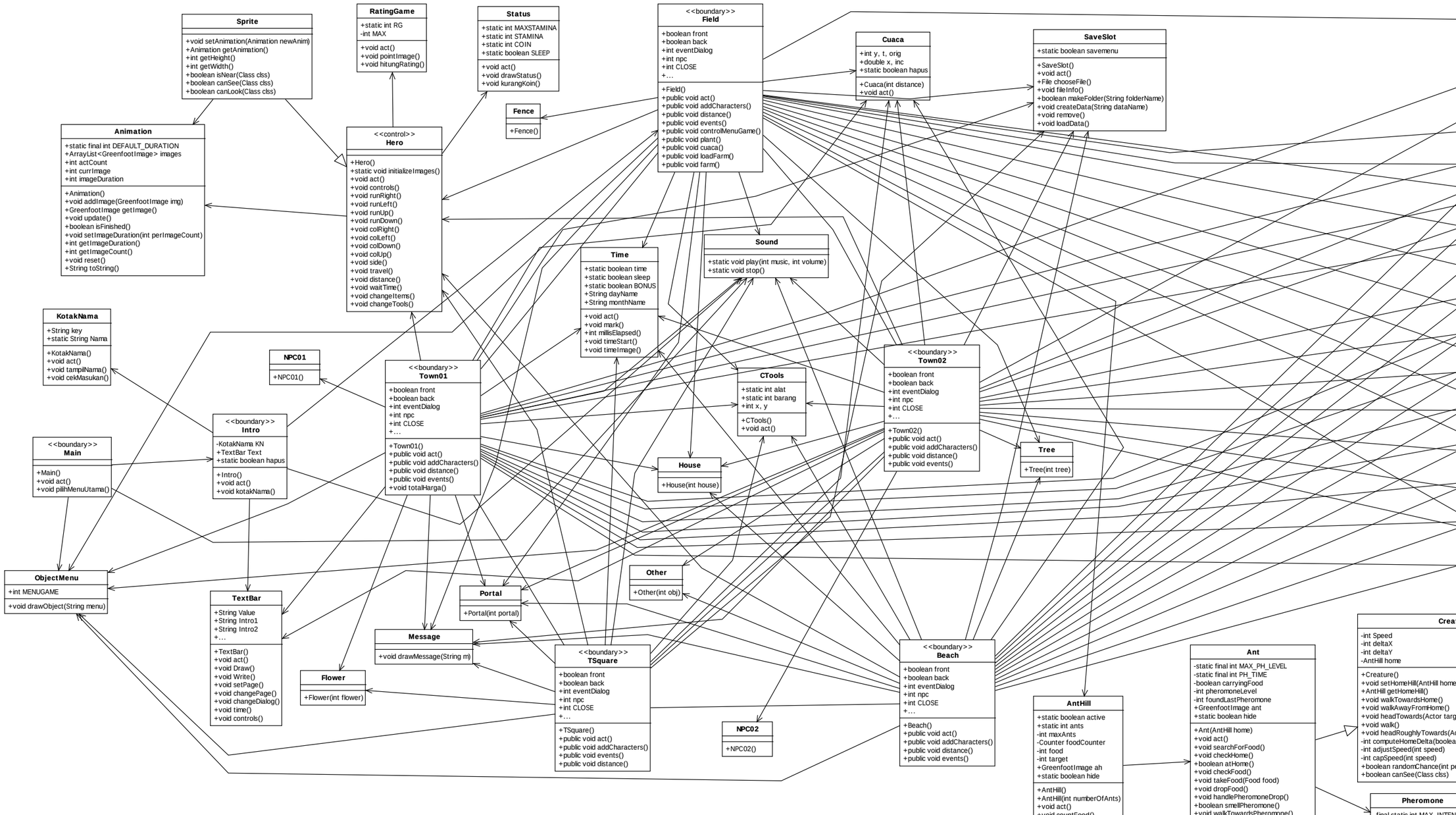
Gambar 3.8 *Activity Diagram*Arena Permainan

3.3.7.3 Penggambaran Kelas

Kelas digambarkan dengan *class diagram*.Diagram kelas atau *classdiagram* menggambarkan struktur sistem dari segi pendefinisian kelas-kelas yang akan dibuat untuk membangun sistem. Kelas memiliki apa yang disebut atribut dan metode atau operasi.*Class diagram* ditunjukan pada Gambar 3.9.Pendefinisian kelas ditunjukan pada Tabel 3.13.

Tabel 3.13 Definisi *Class*

Jenis *Class*	**Nama *Class***
Boundary	Main
	Intro
	Field
	Town01
	Town02
	Tsquare
	Beach
Control	Hero
	TextBar
	Main
Entity	SaveSlot


Sprite
+void setAnimation(Animation newAnim)
+Animation getAnimation()
+int getHeight()
+int getWidth()
+boolean isNear(Class clss)
+boolean canSee(Class clss)
+boolean canLook(Class clss)
RatingGame
+static int RG
-int MAX
+void act()
+void pointImage()
+void hitungRating()
Status
+static int MAXSTAMINA
+static int STAMINA
+static int COIN
+static boolean SLEEP
+void act()
+void drawStatus()
+void kurangKoin()
<<boundary>>
Field
+boolean front
+boolean back
+int eventDialog
+int npc
+int CLOSE
+…
+Field()
+public void act()
+public void addCharacters()
+public void distance()
+public void events()
+public void controlMenuGame()
+public void plant()
+public void cuaca()
+public void loadFarm()
+public void farm()
Cuaca
+int y, t, orig
+double x, inc
+static boolean hapus
+Cuaca(int distance)
+void act()
SaveSlot
+static boolean savemenu
+SaveSlot()
+void act()
+File chooseFile()
+void fileInfo()
+boolean makeFolder(String folderName)
+void createData(String dataName)
+void remove()
+void loadData()
Animation
+static final int DEFAULT_DURATION
+ArrayList<GreenfootImage> images
+int actCount
+int currImage
+int imageDuration
+Animation()
+void addImage(GreenfootImage img)
+GreenfootImage getImage()
+void update()
+boolean isFinished()
+void setImageDuration(int perImageCount)
+int getImageDuration()
+int getImageCount()
+void reset()
+String toString()
<<control>>
Hero
+Hero()
+static void initializeImages()
+void act()
+void controls()
+void runRight()
+void runLeft()
+void runUp()
+void runDown()
+void colRight()
+void colLeft()
+void colDown()
+void colUp()
+void side()
+void travel()
+void distance()
+void waitTime()
+void changeItems()
+void changeTools()
Fence
+Fence()
Time
+static boolean time
+static boolean sleep
+static boolean BONUS
+String dayName
+String monthName
+void act()
+void mark()
+int millisElapsed()
+void timeStart()
+void timeImage()
Sound
+static void play(int music, int volume)
+static void stop()
KotakNama
+String key
+static String Nama
+KotakNama()
+void act()
+void tampilNama()
+void cekMasukan()
NPC01
+NPC01()
<<boundary>>
Town01
+boolean front
+boolean back
+int eventDialog
+int npc
+int CLOSE
+…
+Town01()
+public void act()
+public void addCharacters()
+public void distance()
+public void events()
+void totalHarga()
CTools
+static int alat
+static int barang
+int x, y
+CTools()
+void act()
<<boundary>>
Town02
+boolean front
+boolean back
+int eventDialog
+int npc
+int CLOSE
+…
+Town02()
+public void act()
+public void addCharacters()
+public void distance()
+public void events()
<<boundary>>
Intro
-KotakNama KN
+TextBar Text
+static boolean hapus
+Intro()
+void act()
+void kotakNama()
<<boundary>>
Main
+Main()
+void act()
+void pilihMenuUtama()
House
+House(int house)
Tree
+Tree(int tree)
ObjectMenu
+int MENUGAME
+void drawObject(String menu)
TextBar
+String Value
+String Intro1
+String Intro2
+…
+TextBar()
+void act()
+void Draw()
+void Write()
+void setPage()
+void changePage()
+void changeDialog()
+void time()
+void controls()
Portal
+Portal(int portal)
Other
+Other(int obj)
Message
+void drawMessage(String m)
Flower
+Flower(int flower)
<<boundary>>
TSquare
+boolean front
+boolean back
+int eventDialog
+int npc
+int CLOSE
+…
+TSquare()
+public void act()
+public void addCharacters()
+public void distance()
+public void events()
NPC02
+NPC02()
<<boundary>>
Beach
+boolean front
+boolean back
+int eventDialog
+int npc
+int CLOSE
+…
+Beach()
+public void act()
+public void addCharacters()
+public void distance()
+public void events()
AntHill
+static boolean active
+static int ants
-int maxAnts
-Counter foodCounter
-int food
-int target
+GreenfootImage ah
+static boolean hide
+AntHill()
+AntHill(int numberOfAnts)
+void act()
+void countFood()
+void sebarSemut()
+void checkTarget()
Ant
-static final int MAX_PH_LEVEL
-static final int PH_TIME
-boolean carryingFood
-int pheromoneLevel
-int foundLastPheromone
+GreenfootImage ant
+static boolean hide
+Ant(AntHill home)
+void act()
+void searchForFood()
+void checkHome()
+boolean atHome()
+void checkFood()
+void takeFood(Food food)
+void dropFood()
+void handlePheromoneDrop()
+boolean smellPheromone()
+void walkTowardsPheromone()
+void checkGetFood()
+void removeAnt()
Crea
-int Speed
-int deltaX
-int deltaY
-AntHill home
+Creature()
+void setHomeHill(AntHill home
+AntHill getHomeHill()
+void walkTowardsHome()
+void walkAwayFromHome()
+void headTowards(Actor targ
+void walk()
+void headRoughlyTowards(Ac
-int computeHomeDelta(boolea
-int adjustSpeed(int speed)
-int capSpeed(int speed)
+boolean randomChance(int pe
+boolean canSee(Class clss)
Pheromone
-final static int MAX_INTEN
-int intensity
+static boolean hide

BAB 4

BAB 5

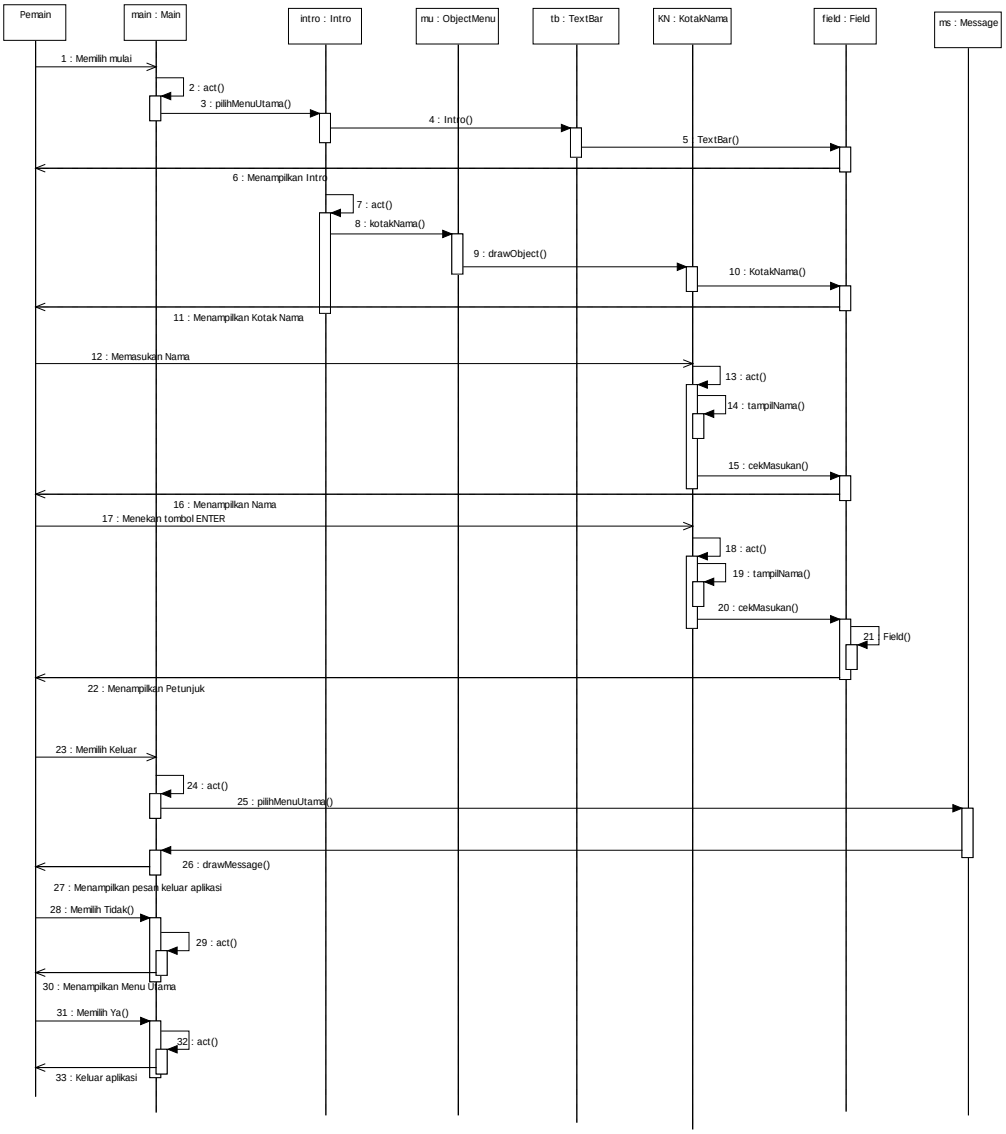
BAB 6

BAB 7

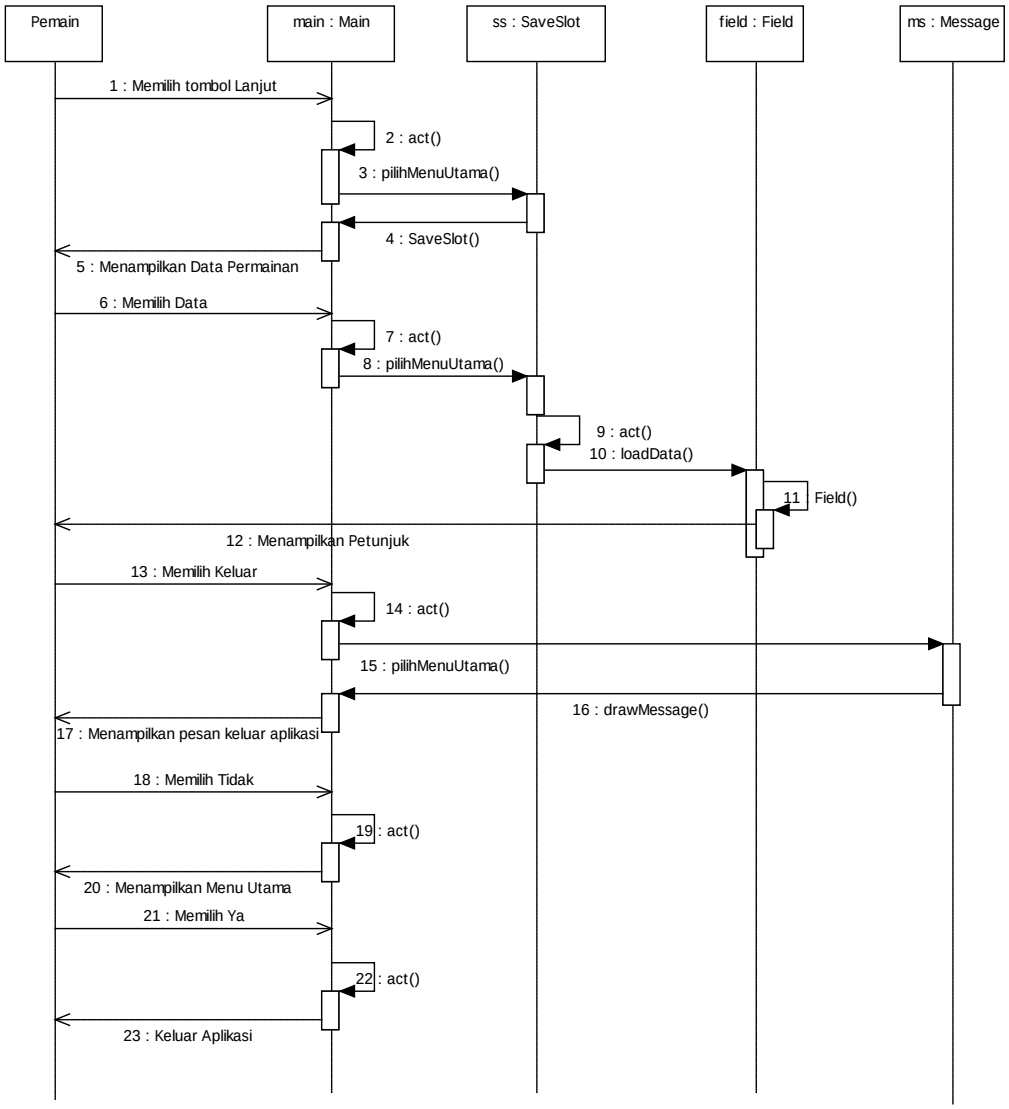
BAB 8

8.3.1.1 Penggambaran Interaksi Antar Objek Pada Game

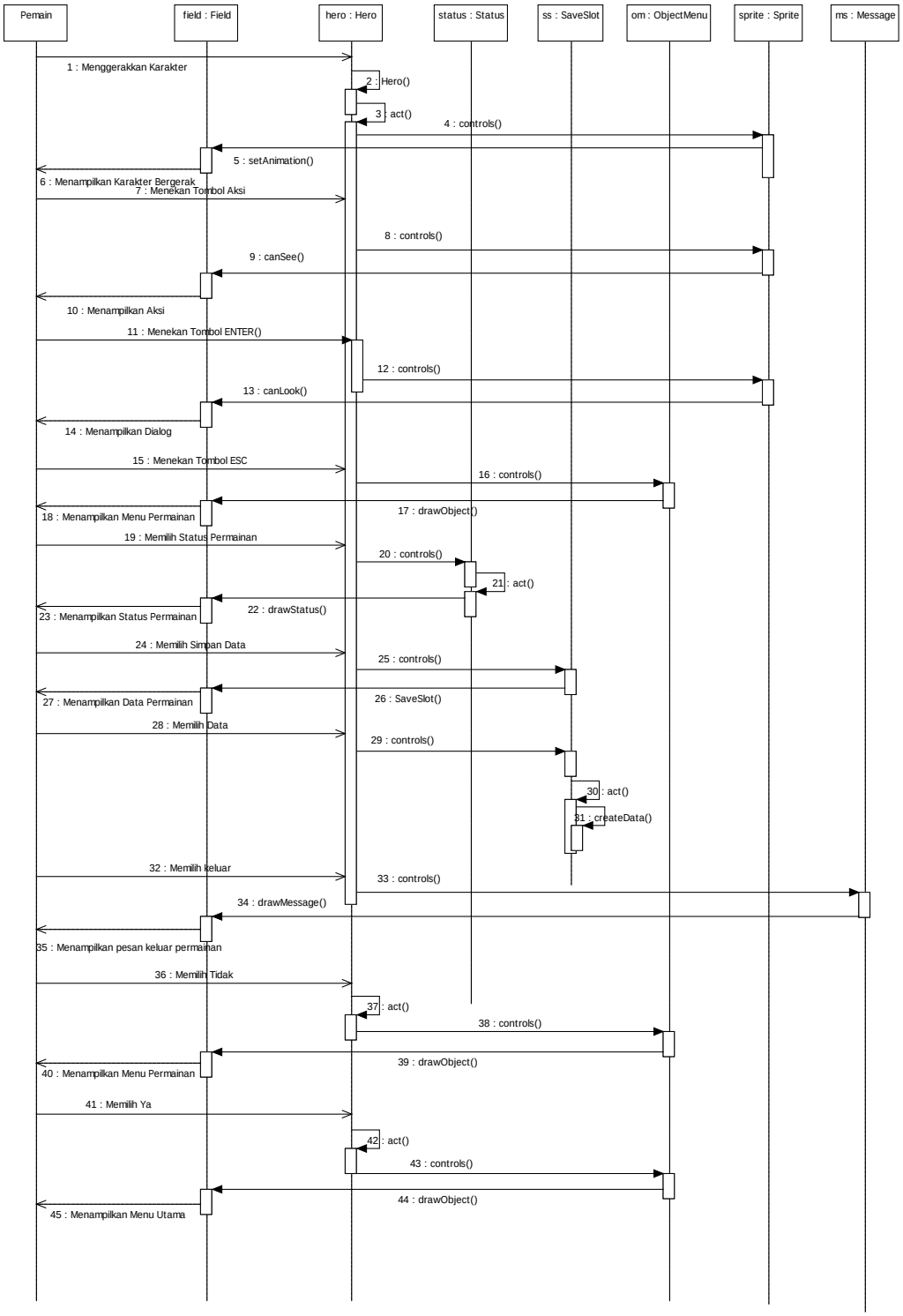
Penggambaran interaksi antar objek menggambarkan interaksi antar objek yang terjadi di dalam dan di luar sistem pada game (termasuk *player*, *display*, dan sebagainya).Penggambaran interaksi antar objek pada game menggunakan *Sequence Diagram*.*Sequence diagram* terdiri atas dimensi *vertical* (waktu) dan dimensi *horizontal* (objek-objek yang terkait). Pada *sequence diagram* ini menggambarkan scenario atau rangkaian langkah-langkah yang dilakukan pada sistem sebagai respon dari sebuah *event* untuk menghasilkan *output* tertentu. *Sequence diagram* ditunjukan pada Gambar 3.1 sampai dengan Gambar 3.3.



Gambar 8.1 *Sequence Diagram*Mulai Permainan



Gambar 8.2 *SequenceDiagram*LanjutkanPermainan



Gambar 8.3 *Sequence Diagram*Tampil Arena Permainan

8.4 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.

8.4.1 Perancangan Komponen Permainan

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

8.4.1.1 Perancangan Karakter

Perancangan karakter adalah bagian dimana karakter pada game dijelaskan ciri-cirinya baik itu secara fungsionalitas peran maupun secara desain. Perancangan karakter manusia pada *gameFarming Life* dijabarkan pada Tabel 3.1.

Tabel 8.1 Perancangan Karakter Manusia

No	Nama	Gambar	Keterangan
1.	Karakter Utama		Seorang pemuda yang mengurusi ladang peninggalan kakeknya
2.	Kim		NPC Penjual barang
3.	Thomas		NPC Kepala Desa Bunga

Berikut ini karakter berupa hama, tanaman, alat, dan barang yang terdapat pada *game Farming Life* dijabarkan pada Tabel 3.2.

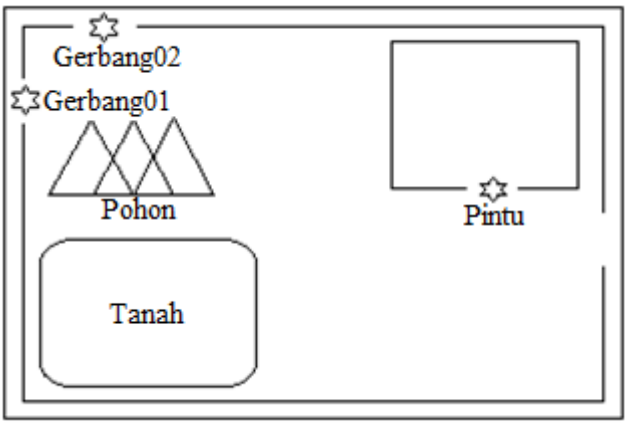
Tabel 8.2 Perancangan Karakter Hama, Tanaman, Alat, dan Barang

No.	Nama	Gambar	Type	Kondisi	Harga (koin)
1.	Ulat		Hama	Penyerangan satu hari dan hari berikutnya tanaman akan mati apabila tidak diberi pestisida. Penyerangan acak.	
2.	Lobak		Tanaman	Waktu tumbuh : 3hari. Panen : sekali	Jual : 180 beli : 30
3.	Kentang		Tanaman	Waktu tumbuh : 3hari. Panen : sekali	Jual : 200 beli : 40
4.	Bawang		Tamanan	Waktu tumbuh : 3hari. Panen : sekali	Jual : 160 beli : 20
5.	Terong		Tanaman	Waktu tumbuh : 7hari. Panen : 2hari sekali.	Jual : 750 beli : 250
6.	Wortel		Tanaman	Waktu tumbuh : 3hari. Panen : sekali	jual : 200 beli : 40
7.	Ubi Manis		Tanaman	Waktu tumbuh : 4hari. Panen : 2hari sekali.	jual :160 beli : 20
8.	Cangkul		Alat Cangkul		
9.	Alat siram		Alat Siraman		
10.	Sabit		Alat Sabit		
11.	Pestisida		Barang Pestisida		beli : 20

8.4.1.2 Perancangan Storyboard

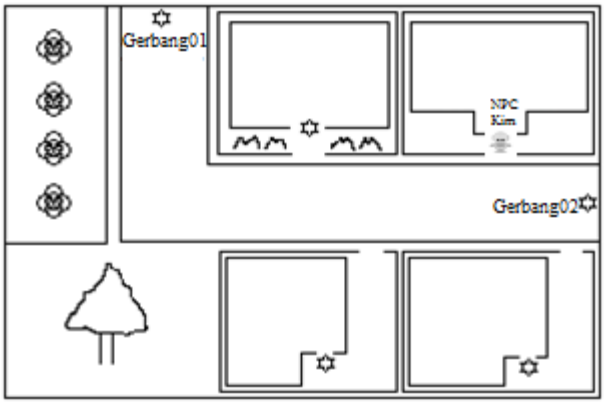
Storyboard merupakan sketsa gambar yang disusun secara berurutan berdasarkan naskah untuk menggambarkan lingkungan pada game. Pada game ini, storyboard merupakan sebuah *map*(peta) yang dijadikan sebagai arena permainan.

Map pada game ini digambarkan pada Gambar 3.13 sampai Gambar 3.18. Berikut ini merupakan *storyboard*dari aplikasi *Game Farming Life* :



Gambar 8.4 Ladang

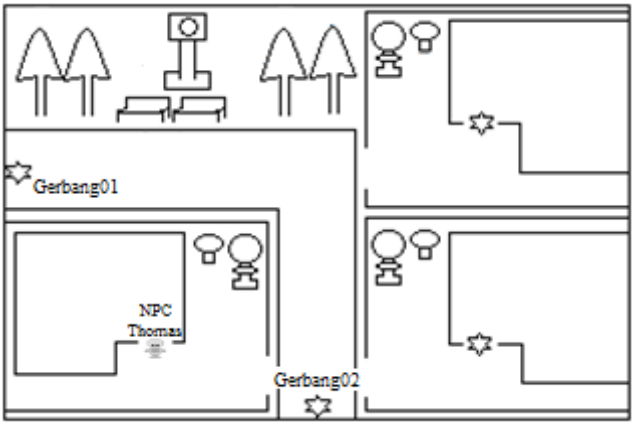
Map **Ladang** merupakan tempat yang hanya dapat digunakan untuk bercocok tanam. **Tanah** adalah lahan yang dapat digunakan untuk menanam.Diatas terdapat beberapa pepohonan.**Pintu** adalah tempat untuk karakter utama mengakhiri hari dan tidur.**Gerbang01** merupakan jalur menuju map **Pemukiman01**. **Gerbang02**merupakan jalur menuju map **Pemukiman02**.



Gambar 8.5 Pemukiman01

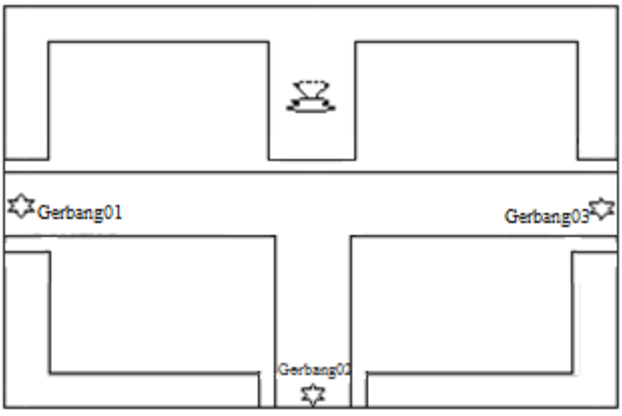
Map **Pemukiman01** merupakan tempat pemukiman penduduk pertama dan terdapat 4 bangunan. Di kiri bawah terdapat pohon dan di samping kiri terdapat

kebun bunga. Jalan terhubung dari**Gerbang01** yang merupakan jalur menuju map **Alun-Alun** hingga **Gerbang02** yang merupakan jalur menuju map **Ladang**. Terdapat **NPC Kim** yang merupakan NPC penjual barang.



Gambar 8.6 Pemukiman02

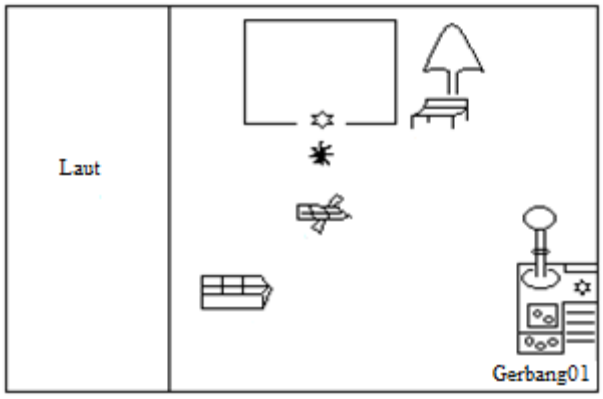
Map **Pemukiman02** merupakan tempat pemukiman penduduk kedua dan terdapat 3 bangunan. Di atas terdapat menara jam yang diapit beberapapohon dan di depannya terdapat kursi. Jalan terhubung dari**Gerbang01** yang merupakan jalur menuju map **Taman Kota** hingga **Gerbang02** yang merupakan jalur menuju map **Ladang**.



Gambar 8.7 Alun-Alun

Map **Alun-Alun** merupakan tempat mengadakan festival. Di tengah terdapat air mancur dikelilingi rumput di tepi map. Terdapat 3 Gerbang yaitu **Gerbang01**merupakan jalur menuju map**Pantai**, **Gerbang02** merupakan jalur

menuju map **Pemukiman01** dan **Gerbang03** merupakan jalur menuju map **Pemukiman02**.

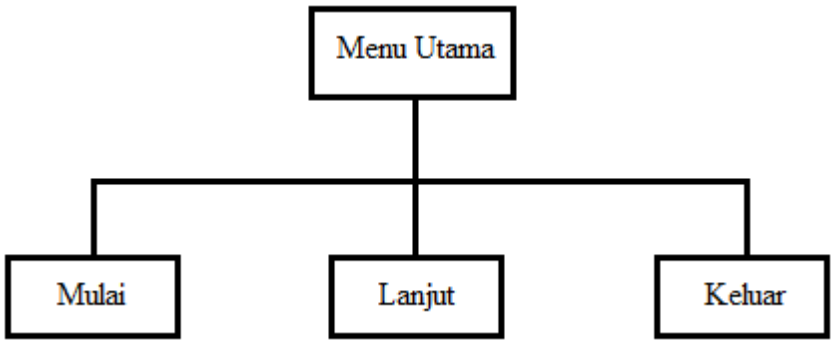


Gambar 8.8 Pantai

Map **Pantai** merupakan pantai satu-satunya yang terdapat pada *Farming Life*. Hanya terdapat 1 bangunan yang berada di map ini. Di samping bangunan terdapat pohon dan kursi dan di depannya terdapat tempat api unggun, batang pohon , dan perahu. Di sebelah kiri terdapat **Laut** yang terbentang dari atas sampai ke bawah.**Gerbang01** merupakan jalur menuju map **Alun-Alun**.

8.4.2 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 Farming Life* dapat dilihat pada gambar 3.38.



Gambar 8.9 Arsitektur Menu

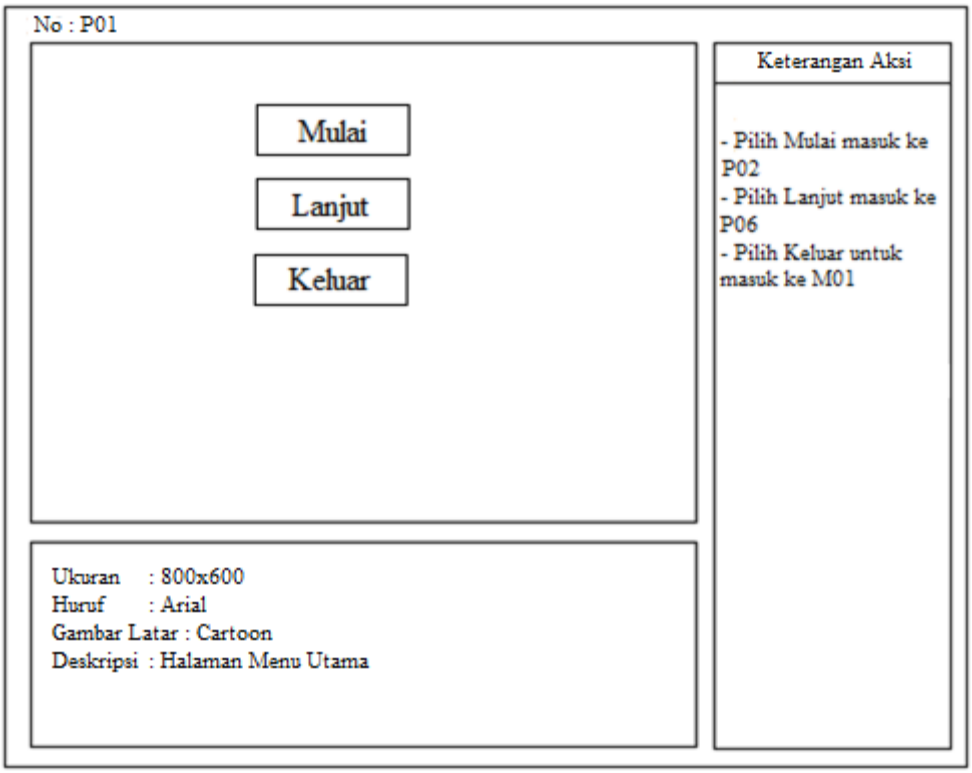
A. **Mulai** : Merupakan submenu untuk memulai permainan baru.

B. **Lanjut**: Merupakan submenu untuk memulai permainan berdasarkan data yang tersimpan.

C. **Keluar**: Merupakan submenu untuk keluar dari aplikasi

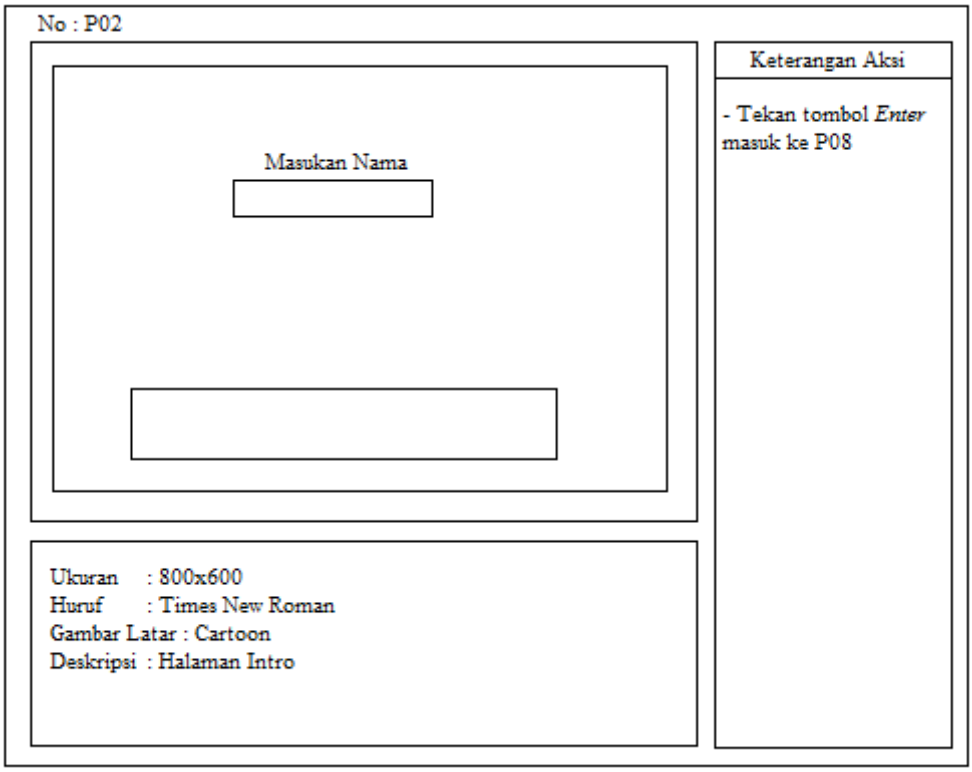
8.4.3 Perancangan Antarmuka

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 pada Gambar 3.19 sampai dengan Gambar 3.26.



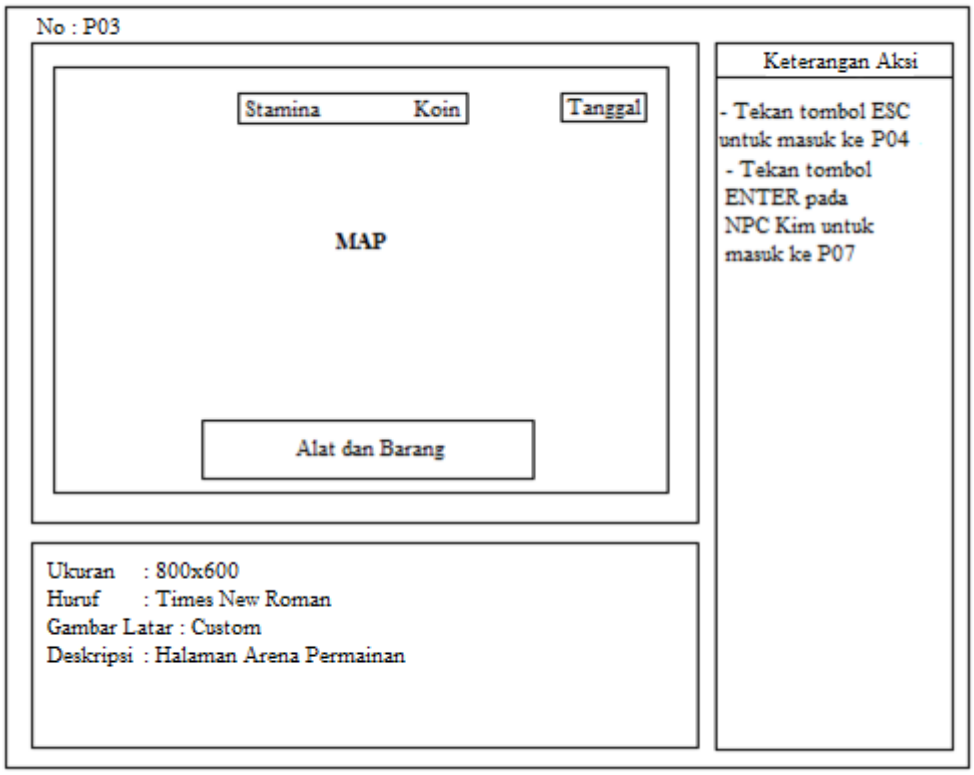
Gambar 8.10 Perancangan Antarmuka Menu Utama

Perancangan antarmuka Menu Utama memiliki 3 pilihan submenu yaitu, **Mulai** untuk memulai permainan dari awal, **Lanjut** untuk melanjutkan permainan dari data yang tersimpan, dan **Keluar** untuk keluar dari aplikasi.



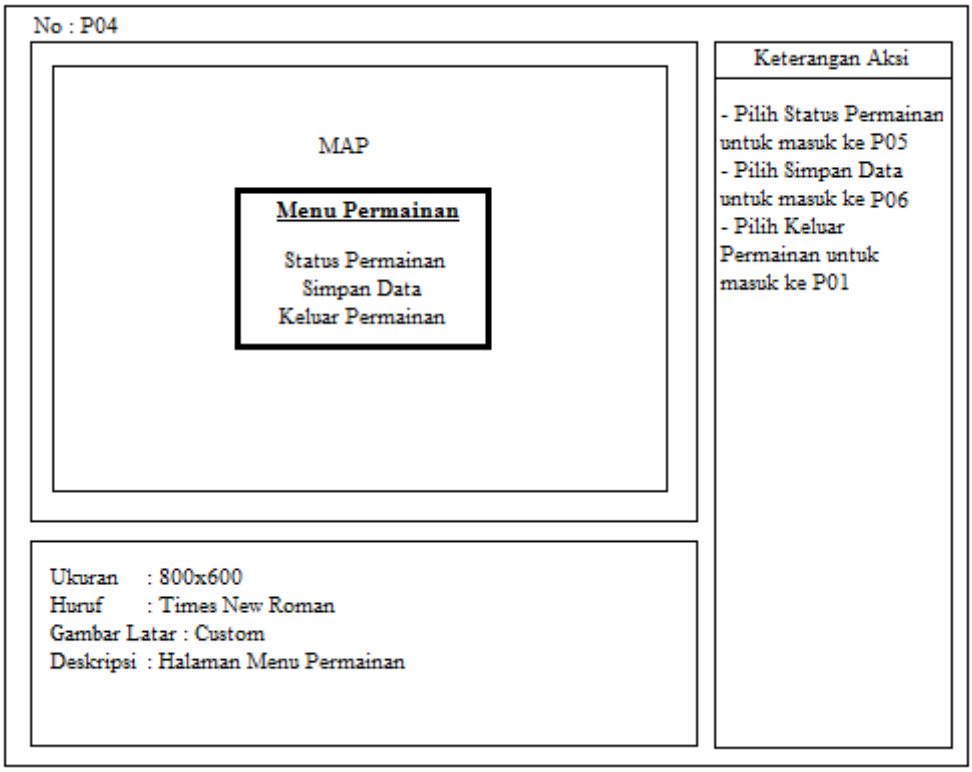
Gambar 8.11 Perancangan Antarmuka Intro

Perancangan antarmuka Intro menampilkan kotak dialog yang berada di bawah dan kotak nama yang berada di tengah. Kotak nama akan muncul ketika dialog berhenti. Memasukan nama karakter dengan mengetik huruf yang berada di *keyboard*. Tekan ENTER untuk masuk ke arena permainan.



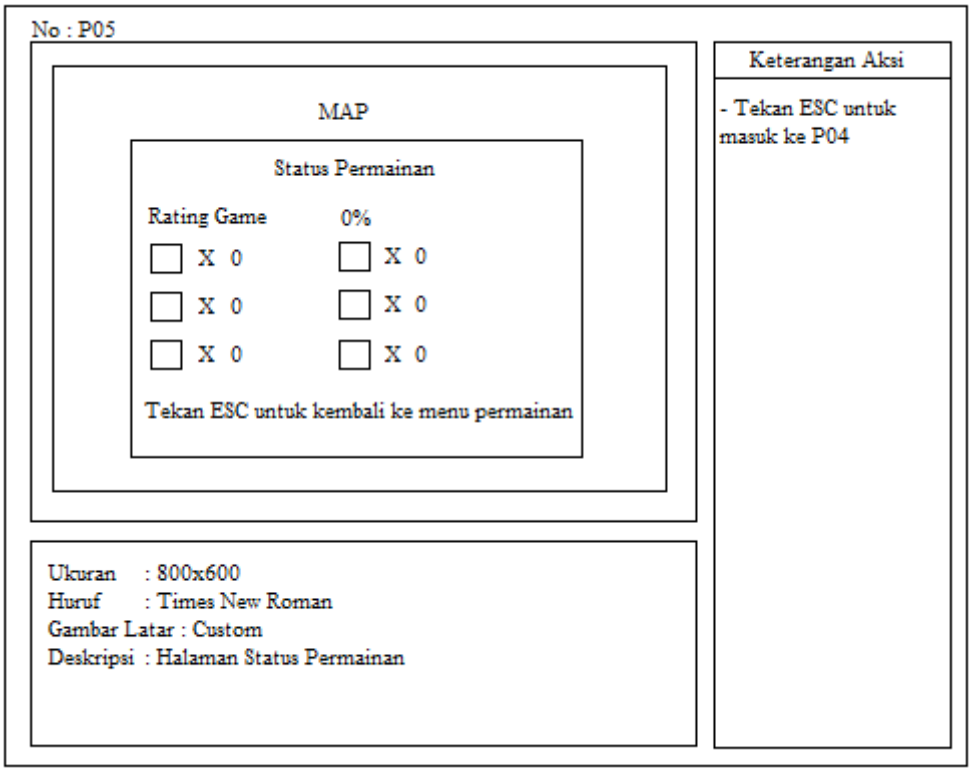
Gambar 8.12 Perancangan Antarmuka Arena Permainan

Perancangan antarmenu Arena Permainan memunculkan map yang dikunjungi oleh karakter utama. Tanggal terdapat pada pojok kanan atas.Stamina dan Koin terdapat pada pojok kiri atas.Daftar alat dan barang terdapat pada pojok kanan bawah.Menekan ESC untuk menampilkan menu permainan.Menekan ENTER ketika karakter utama berinteraksi dengan NPC Kim untuk menampilkan daftar barang.



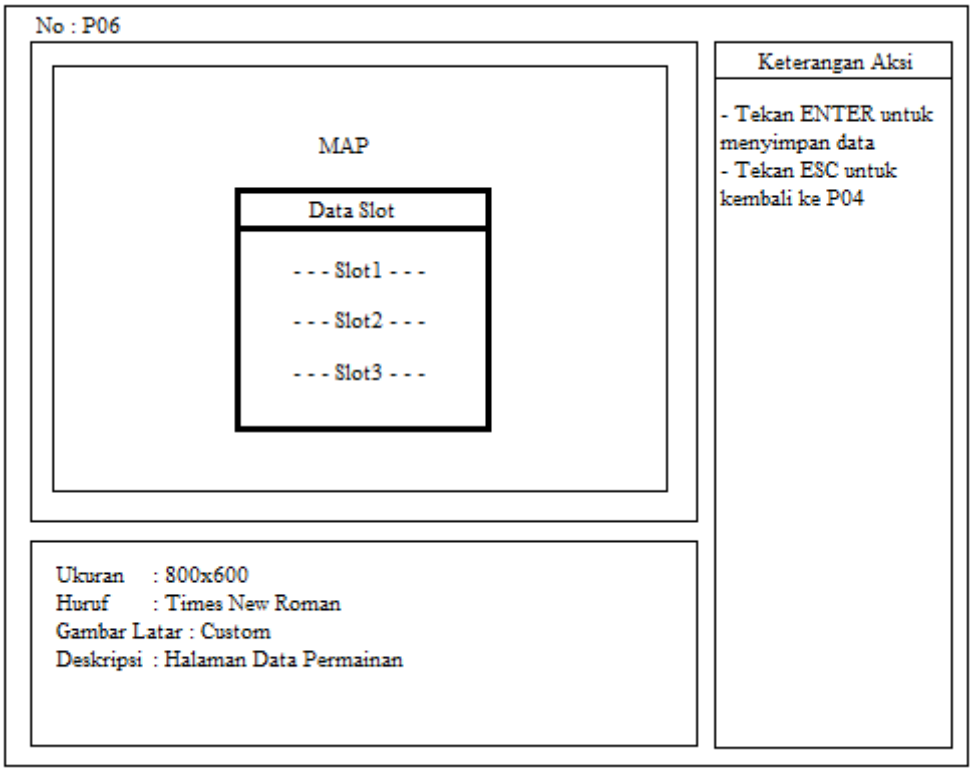
Gambar 8.13 Perancangan Antarmuka Menu Permainan

Perancangan antarmuka Menu Permainan memiliki 3 submenu yaitu, **Status Permainan** untuk menampilkan Status Permainan berlangsung, **Simpan Data** untuk menampilkan Data Slot, **Keluar Permainan** untuk kembali ke menu utama. Tekan tombol ESC untuk kembali ke Arena Permainan.



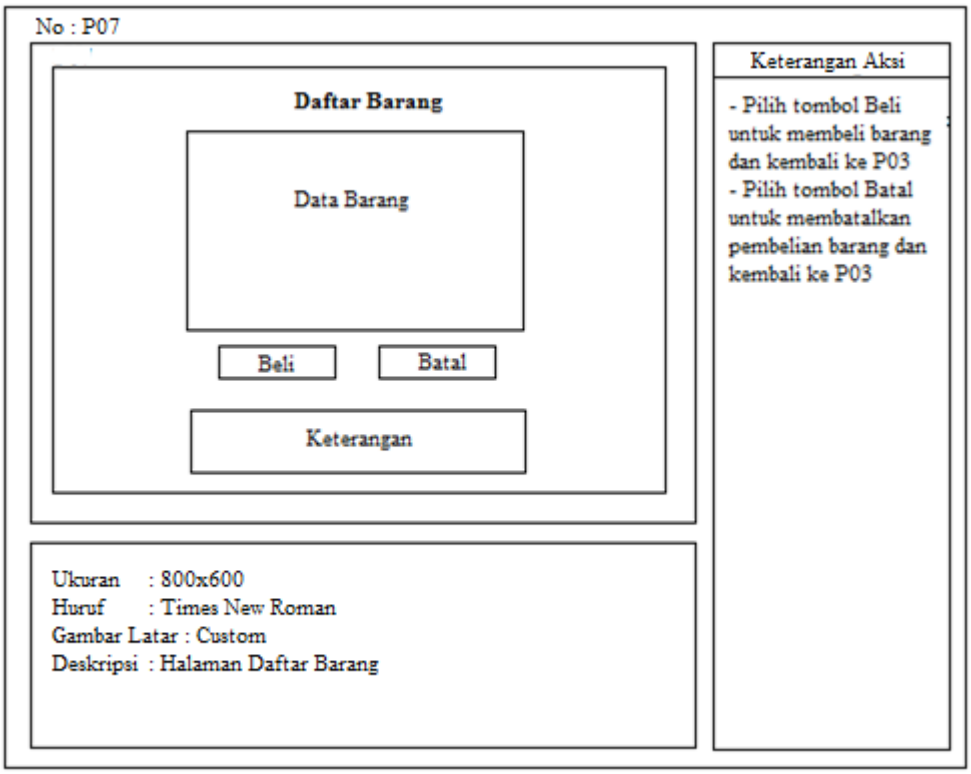
Gambar 8.14 Perancangan Antarmuka Status Permainan

Perancangan antarmuka Status Permainan menampilkan *Rating Game* dan hasil panen ketika *game* berlangsung.Menekan ESC kembali ke menu permainan.



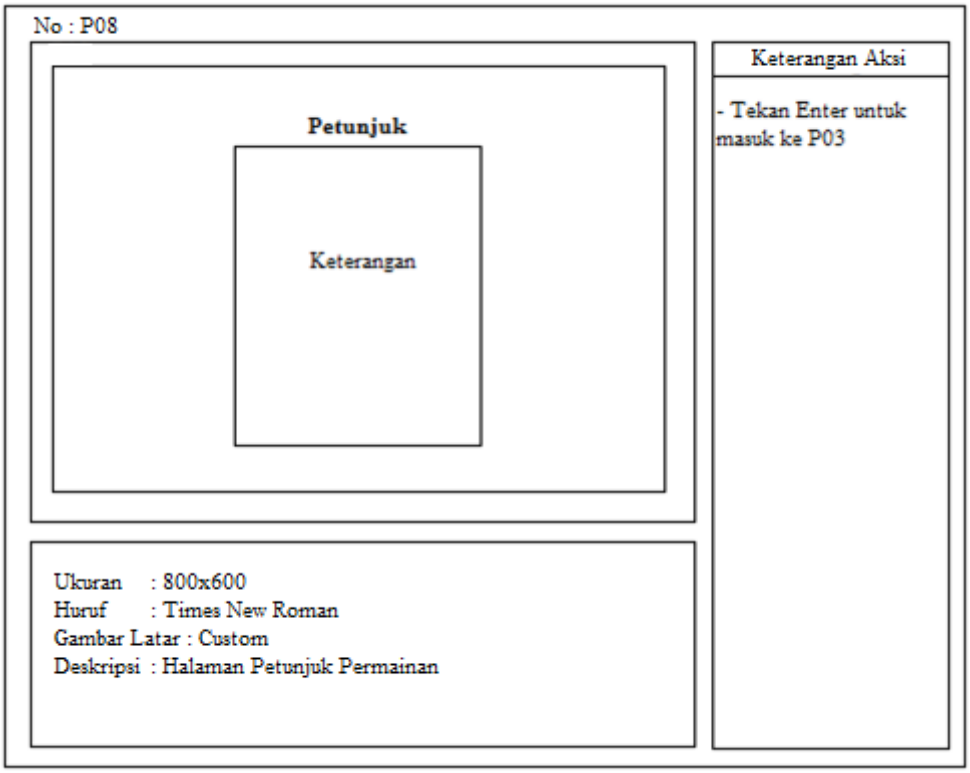
Gambar 8.15 Perancangan Antarmuka Data Permainan

Perancangan antarmuka Data Slot menampilkan 3 slot untuk menyimpan data permainan. Tekan ESC untuk kembali ke menu permainan.



Gambar 8.16 Perancangan Antarmuka Daftar Barang

Perancangan antarmuka Daftar Baranguntuk membeli barang yang diinginkan.Pilih beli untuk membeli barang dan kembali ke Arena Permainan.Pilih batal untuk membatalkan pembelian dan kembali ke arena permainan.

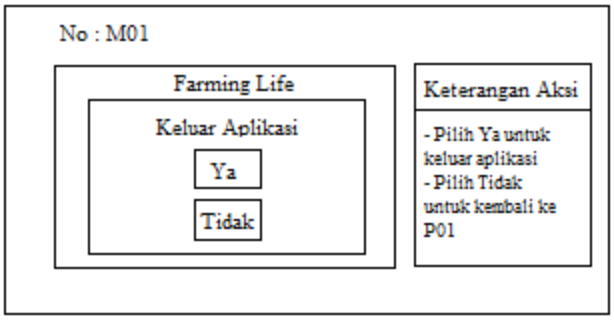


Gambar 8.17 Perancangan Antarmuka Petunjuk

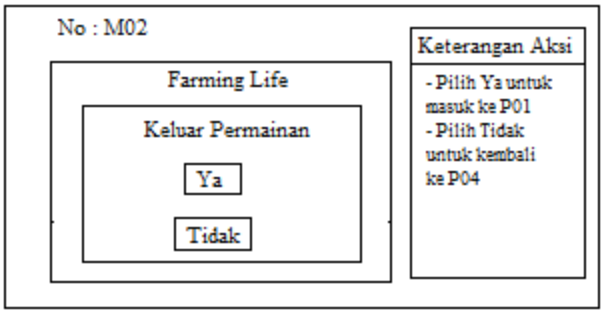
Perancangan antarmuka Petunjukuntuk memberikan petunjuk cara permainan.Tekan ENTER untuk masuk ke arena permainan.

8.4.4 Perancangan Pesan

Perancangan pesan adalah rancangan antar muka dari pesan yang akan ditampilkan apabila terjadi kesalahan dan pemberitahuan dalam proses yang dilakukan. Rancangan pesan dari gameini digambarkan pada Gambar 3.27 sampai dengan Gambar 3.28.



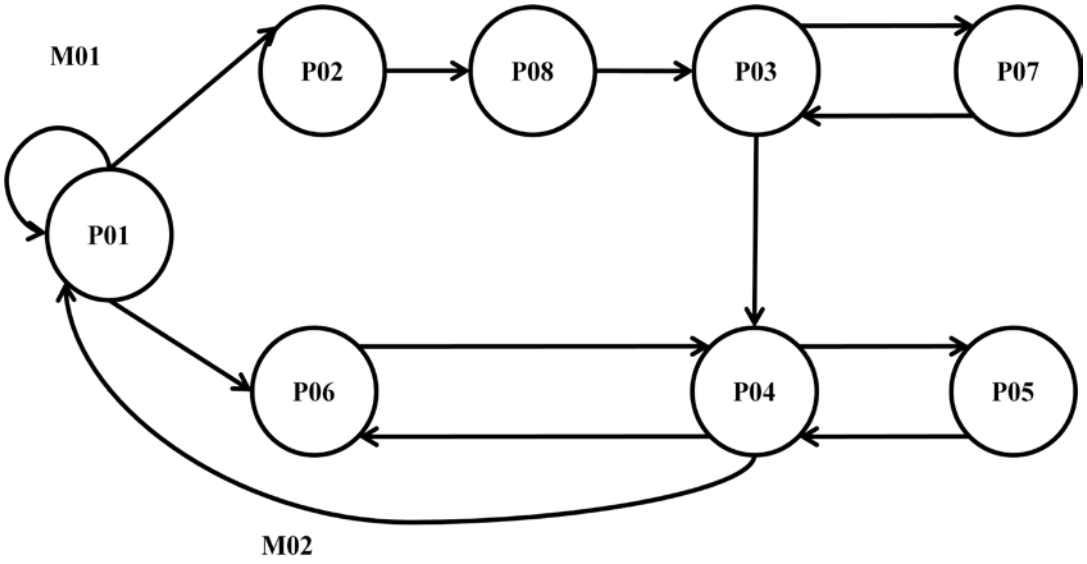
Gambar 8.18 Perancangan Pesan Peringatan Keluar Aplikasi



Gambar 8.19 Perancangan Pesan Peringatan Keluar Permainan

8.4.5 Jaringan Semantik

Jaringan semantik merupakan alat yang mangkus untuk merepresentasikan pemetaan data, yang bertujuan mencegah terjadinya duplikasi data.Jaringan semantik pada game ini digambarkan pada Gambar 3.29.



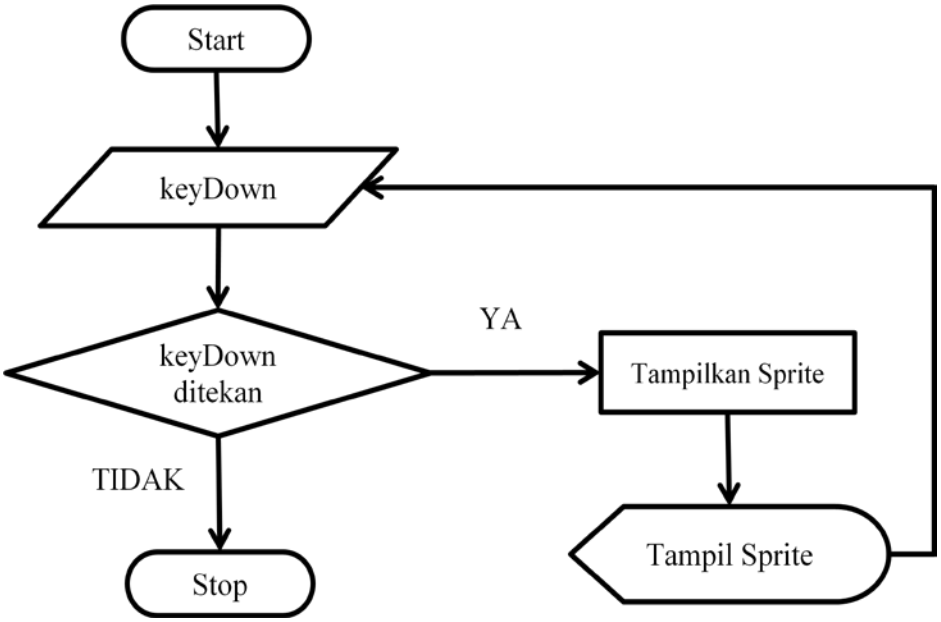
Gambar 8.20 Jaringan Semantik

8.4.6 Perancangan Method

Perancangan method merupakan pemaparan proses pada method-method yang digunakan pada setiap kelas. Perancangan method digambarkan dengan menggunakan Flow Chart.

A. Method Controls()

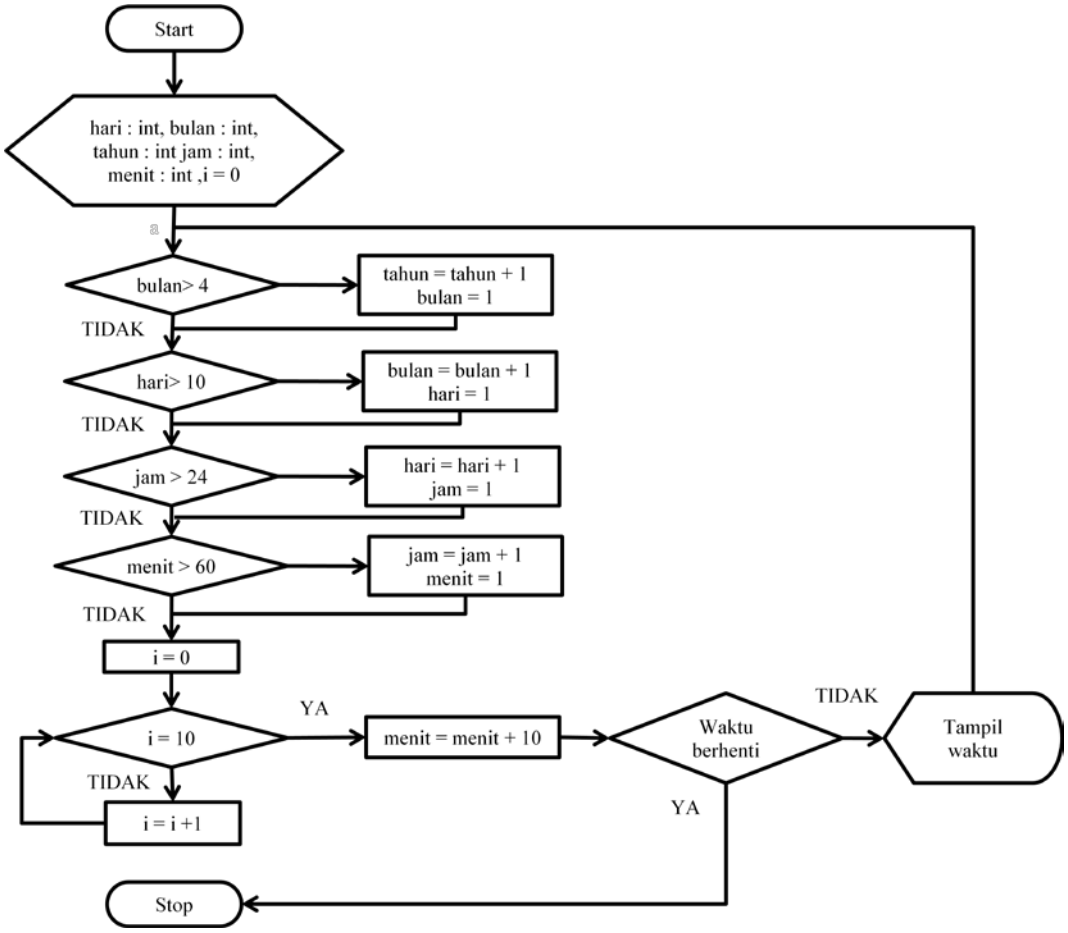
Method controls() dipanggil untuk menggerakkan objek karakter, dapat dilihat pada gambar 3.30.



Gambar 8.21 Method controls()

B. Method Time()

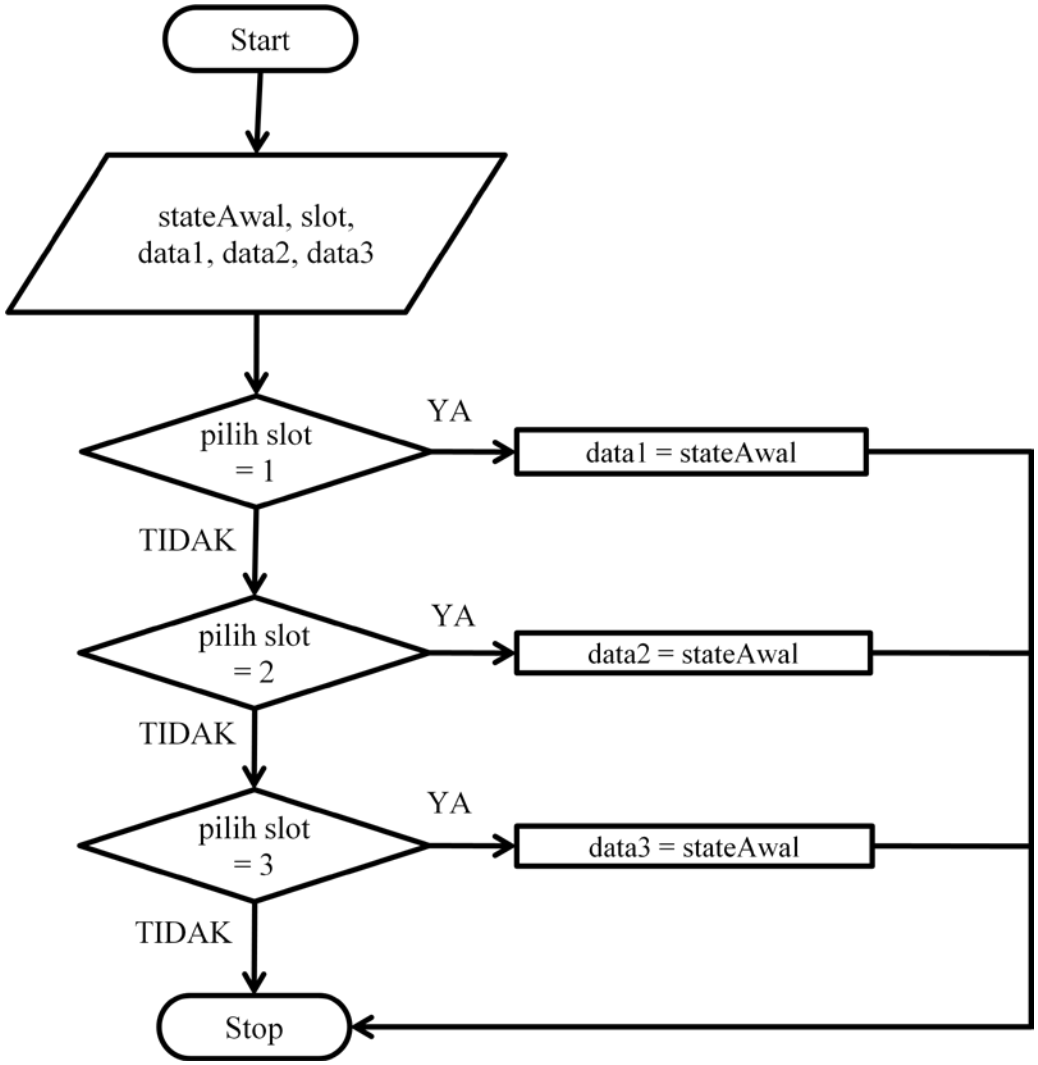
Method time() akan dipanggil ketika telah memasuki arena permainan, dapat dilihat pada gambar 3.31.



Gambar 8.22 Method timeStart()

C. Method CreateData()

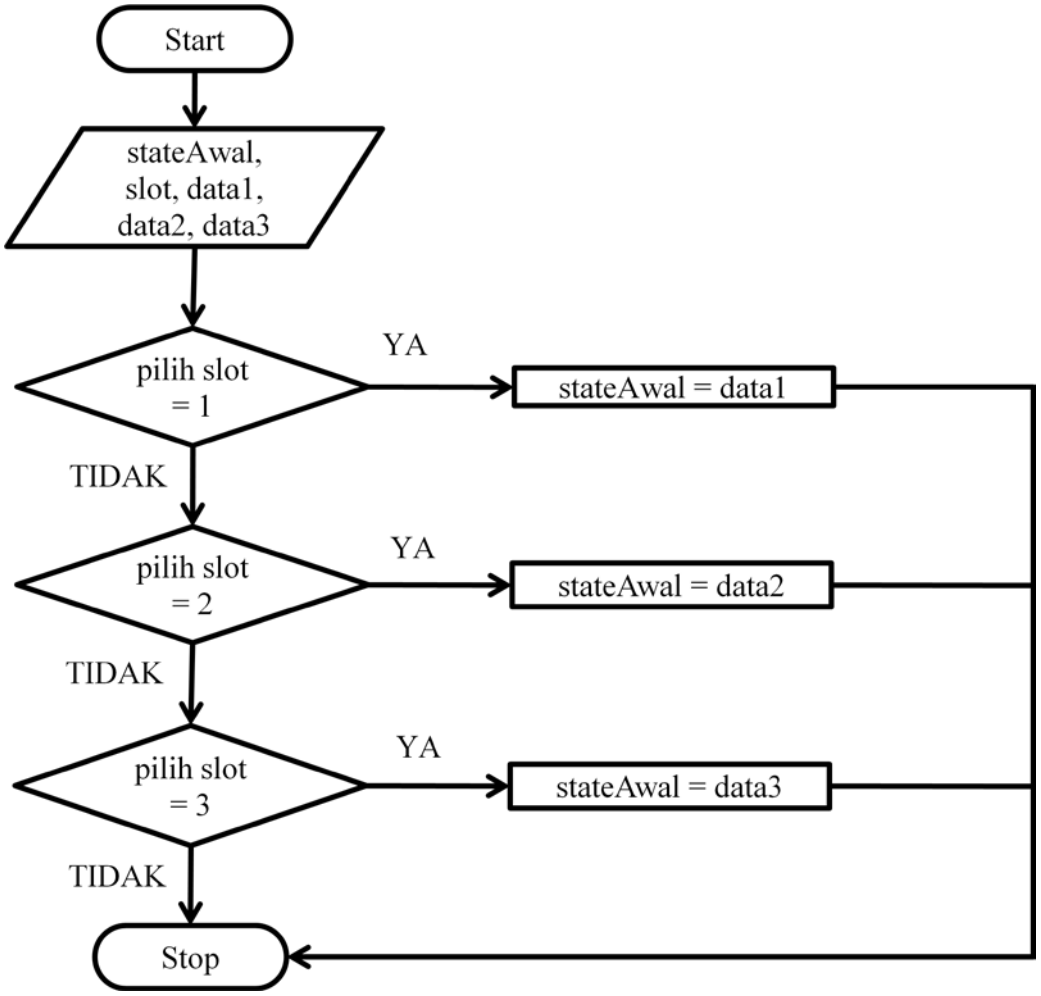
Method createData() dipanggil ketika *save data slot* dipilih, dapat dilihat pada gambar 3.32.



Gambar 8.23 Method createData()

D. Method LoadData()

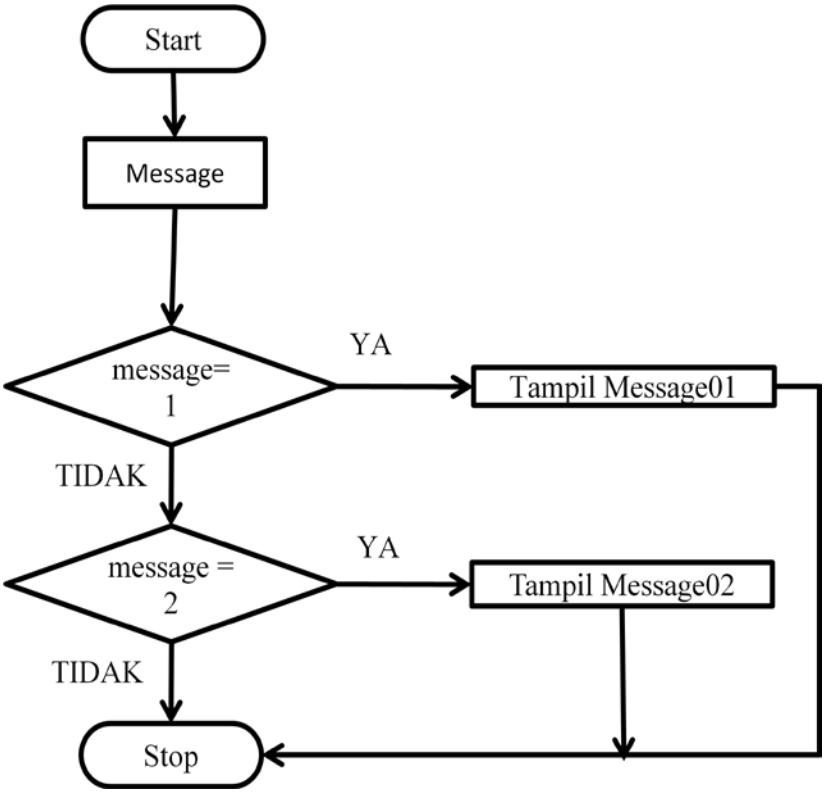
Method loadData() dipanggil ketika load data slot dipilih, dapat dilihat pada gambar 3.33.



Gambar 8.24 MethodloadData()

E. Method drawMessage()

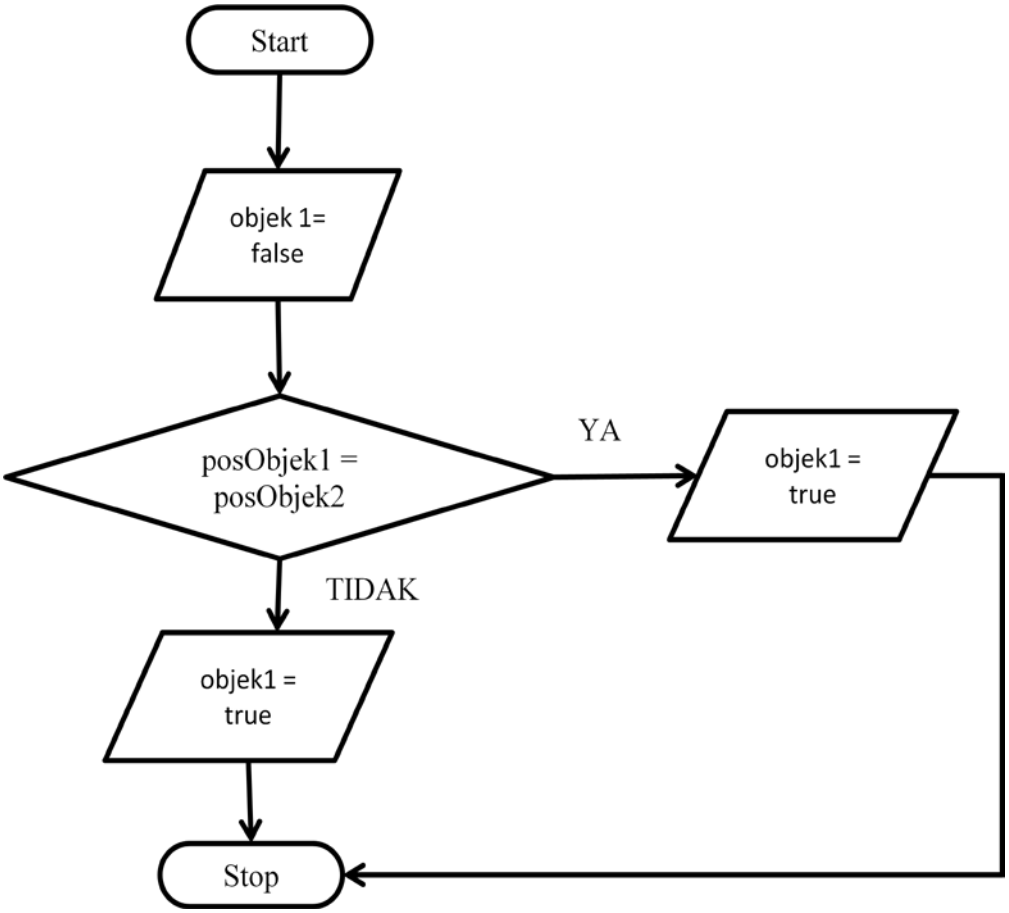
Method draw Message() dipanggil ketika sistem memunculkan pesan, dapat dilihat pada gambar 3.34.



Gambar 8.25 MethoddrawMessage()

F. Method canSee()

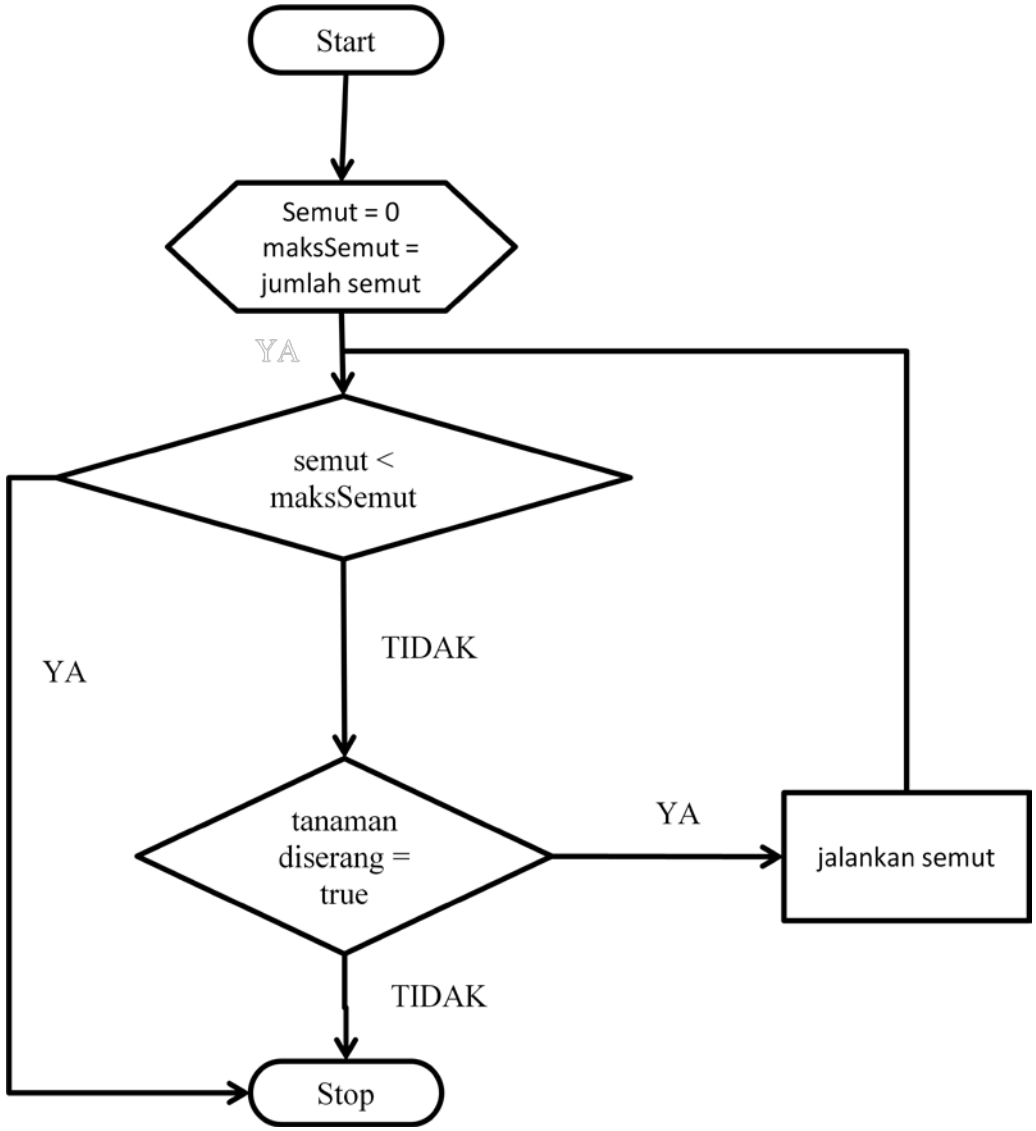
Method draw canSee() dipanggil ketika objek yang satu dengna yang lainnya bersinggungan, dapat dilihat pada gambar 3.35.



Gambar 8.26 MethodcanSee()

G. Method sebarSemut()

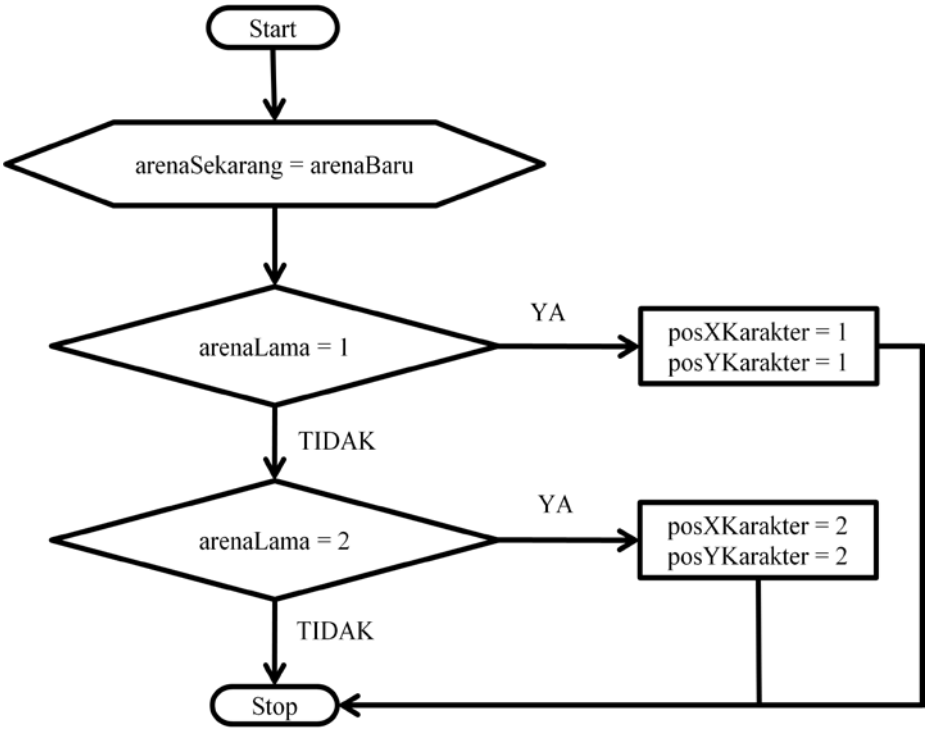
Method draw sebarSemut() dipanggiluntuk menyebarkan semut ke area permainan, dapat dilihat pada gambar 3.36.



Gambar 8.27 MethodsebarSemut()

H. Method addHero()

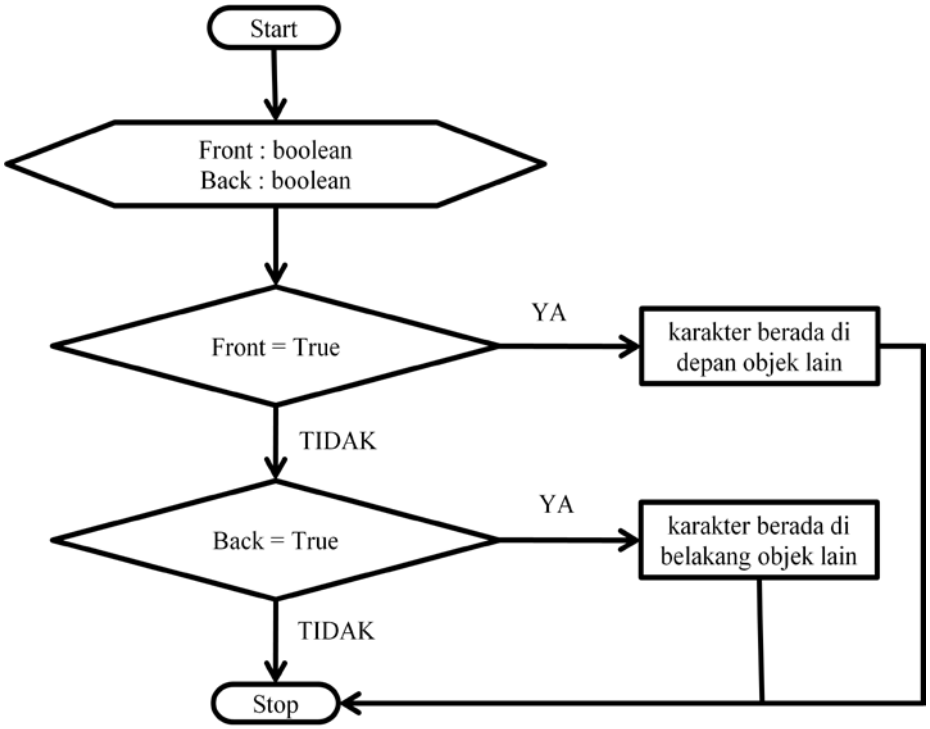
Method draw addHero() dipanggil untuk meletakan karakter utama ketika menampilkan arena permainan, dapat dilihat pada gambar 3.37.



Gambar 8.28 MethodaddHero ()

I. Method distance()

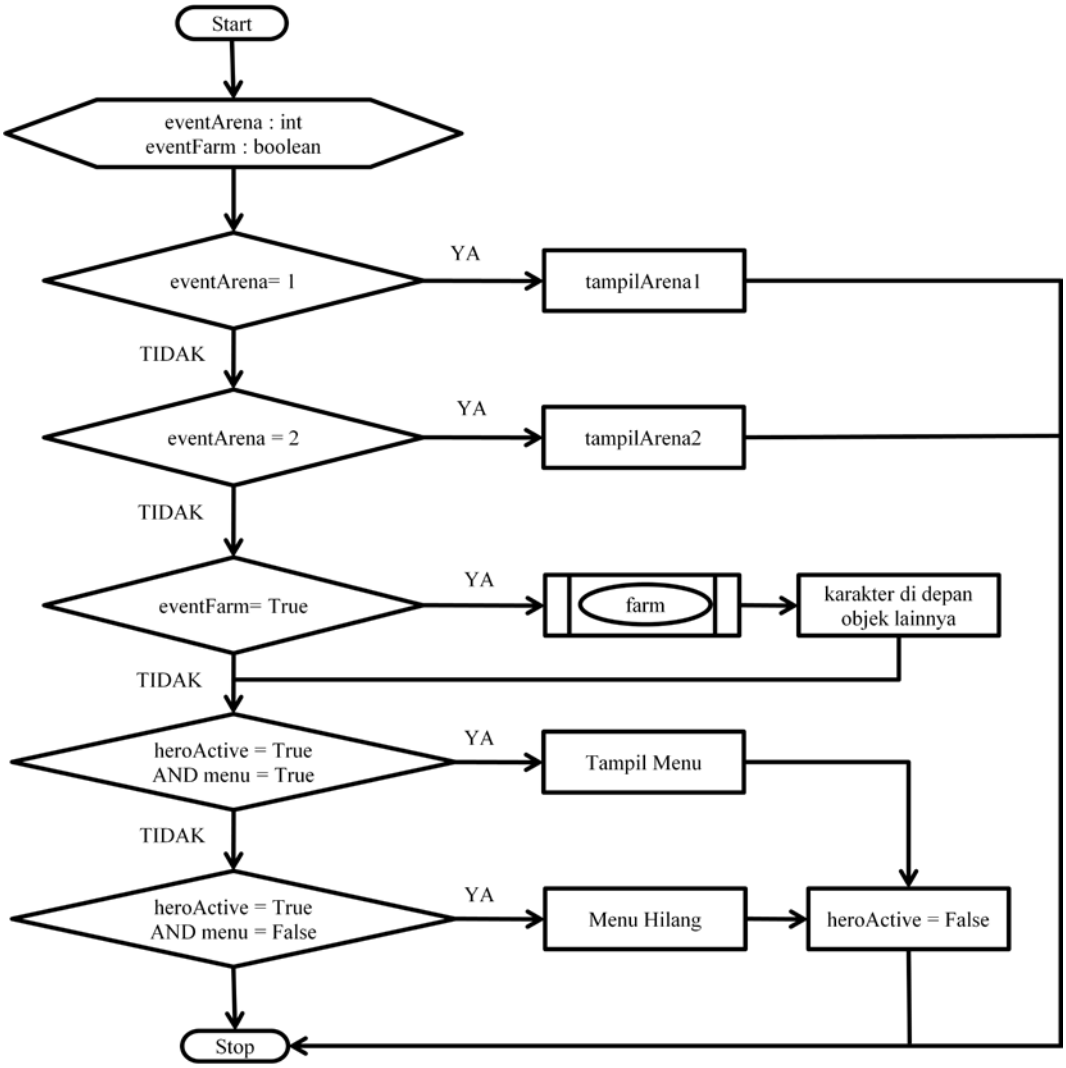
Method draw distance() dipanggil untuk meletakan karakter utama agar berada di depan atau di belakang objek lainnya, dapat dilihat pada gambar 3.38.



Gambar 8.29 Methoddistance ()

J. Method events()

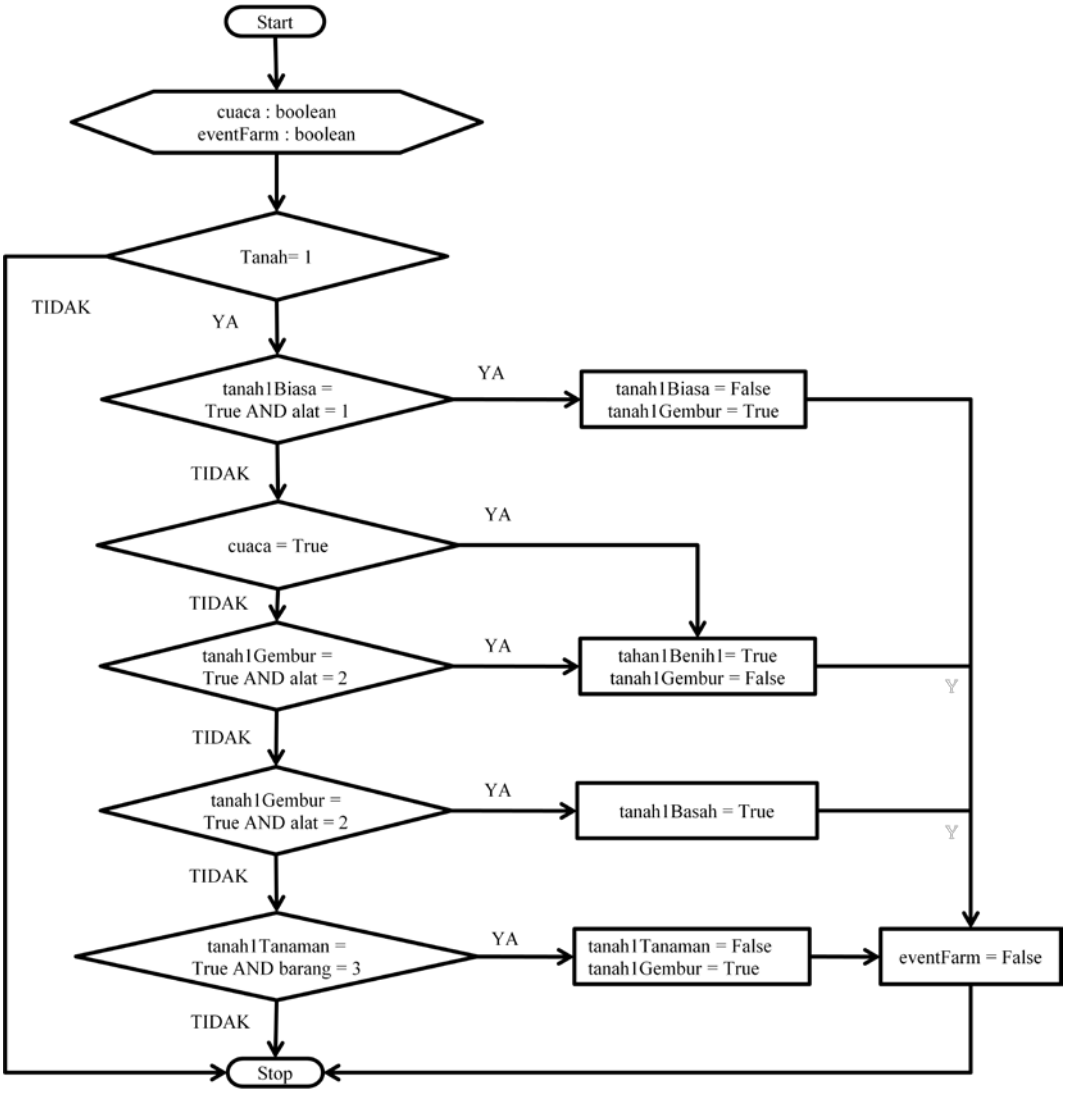
Method draw events() dipanggil untuk mengeksekusi kejadian yang terjadi pada arena permainan, dapat dilihat pada gambar 3.39.



Gambar 8.30 Methodevents ()

K. Method farm()

Method draw farm() dipanggil untuk melakukan aksi menanam, dapat dilihat pada gambar 3.40.



Gambar 8.31 Methodfarm()