BAB 3
ANALISIS DAN PERANCANGAN SISTEM

3.1 Analisis Sistem

Analisis adalah proses mengurai konsep ke dalam bagian-bagian yang lebih sederhana, sedemikian rupa sehingga struktur logisnya menjadi jelas. Analisis merupakan metode untuk menguji, menilai, dan memahami sistem pemikiran yang kompleks dengan memecahnya ke dalam unsur-unsur yang lebih sederhana sehingga hubungan antar unsur-unsur itu menjadi jelas.

3.1.1 Analisis Masalah

Dalam merancang dan mengembangkan sebuah aplikasi *game* ada beberapa permasalahan yang perlu diperhatikan. Seperti menganalisa aplikasi *game* yang serupa atau yang telah ada dengan cara membandingkan kelebihan dan kekurangan *game* tersebut, dalam membandingkan sebuah aplikasi *game* dapat dilihat dari perbedaan-perbedaan yang ada pada aplikasi *game* yang serupa dengan aplikasi *game* yang akan dibangun. Dengan melihat kekurangan dari sebuah *game* yang serupa maka suatu permasalahan didapat. Suatu aplikasi *game* dapat dikembangkan dan dibangun dengan melihat beberapa perbandingan yang ada pada sebuah aplikasi *game*, sehingga *game* yang serupa dapat dikembangkan dan menjadi lebih baik lagi dan lebih realistis.

Dari beberapa game yang sering ditemui pada aplikasi desktop tentang peternakan dengan *genre Time Management Game*, umumnya tidak menampilkan tokoh karakter dari peternak tersebut, melainkan fokus terhadap hewan ternak dan produksi yang dihasilkan. Seperti pada beberapa game yang sudah ada.

Tabel 3.1 Analisis Perbandingan Game yang sejenis

Game yang sejenis (Farm Frenzy)	**Game Ternak**
Tidak adanya karakter peternak	Ada karakter peternak
Tiap naik level berulang ke posisi awal (dimana hewan ternak masih kosong)	Selalu meneruskan dari level sebelumnya
Tidak ada kandang	Disediakan kandang
Jenis makanan sama	Jenis makanan hewan ternak dibedakan
Makanan hewan tidak dibeli	Makanan hewan dibeli
Musuh berjalan acak	Musuh mengejar hewan ternak
Menu upgrade ditampilkan setiap naik level	Menu upgrade ditampilkan didalam permainan

3.1.2 Analisis *Game* Ternak

Game yang akan dibangun ini berjudul Ternak, dan *genre* yang dipakai dalam *game* ini adalah *Time Management Game*. Dalam kisahnya nanti *game* ini akan menceritakan tentang peternakan, dari mulai beternak hewan sampai mengembangbiakkan dan menjual hasil ternak tersebut.

Game ini dibangun berdasarkan salah satu aktifitas dari sebagian banyak warga Indonesia yang dikemas kedalam sebuah media permainan desktop. *Game* ini merupakan pengembangan dari *game* yang sudah ada dengan menggabungkan fitur-fitur yang ada pada permainan online. Yang dibangun dengan grafis 2D (2 *Dimensi*). Cara permainannya sendiri menggunakan sistem *single player*. AI (*Artificial Intelligence*) yang digunakan untuk musuh dalam *game* ini adalah algoritma A* (*A Star*) untuk mencari solusi terbaik dengan *cost* paling murah.

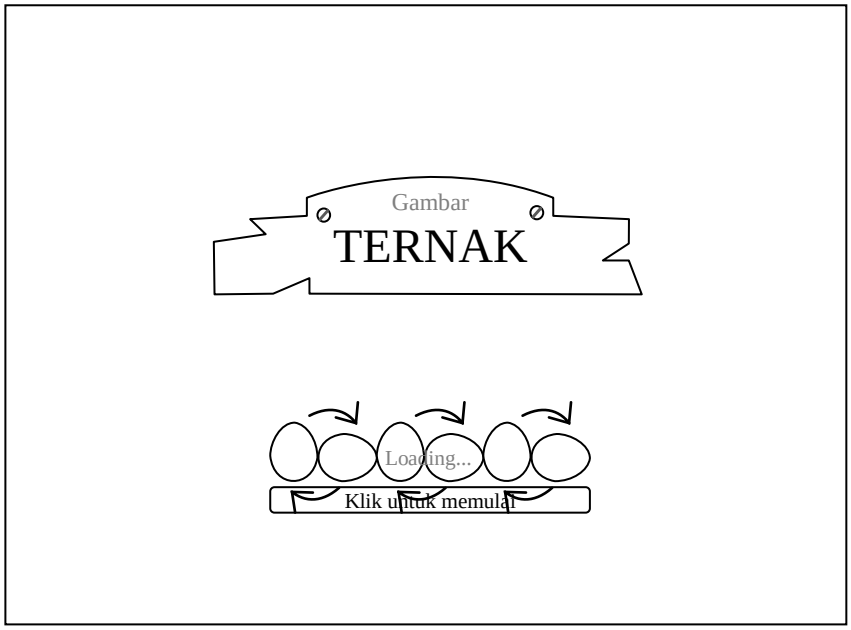
3.1.2.1 *Storyline*

Game Ternak menceritakan sebuah aktifitas peternak hewan yang selalu berusaha mengelola ternaknya (berupa ikan, ayam, itik, kambing, dan sapi). Agar menghasilkan produksi yang baik dan maksimal, yang bermula dari modal yang cukup untuk bisa beternak ikan dan ayam, sampai menghasilkan keuntungan yang banyak sehingga dapat beternak hewan dengan modal besar seperti sapi. Memberi makan hewan ternak adalah suatu hal yang utama agar hewan ternak yang dimiliki tidak mati, dan dapat berkembang biak sehingga populasinya meningkat. Untuk menjaga hewan ternak dari ancaman pemangsa dan pencuri maka salah satu cara antisipasinya yaitu dengan membuatkan kandang atau pagar bagi hewan ternak yang dimiliki, setidaknya dengan antisipasi itulah dapat menghambat pergerakan pemangsa dan pencuri ternak. Sampai akhirnya ternak yang dimiliki berjumlah sangat besar sehingga dapat mengambil keuntungan dari produksi hewan ternak yang dimiliki.

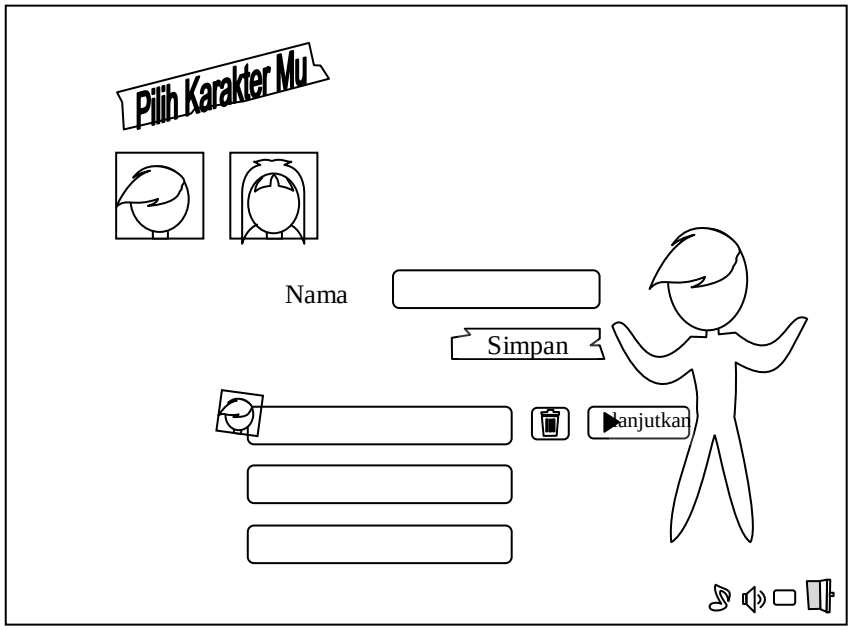
3.1.2.2 *Storyboard*

Storyboard adalah sketsa gambar yang disusun berurutan sesuai dengan naskah, dengan *storyboard* ide cerita yang dibuat dapat disampaikan dengan lebih mudah kepada orang lain, karena dengan ini imajinasi seseorang dapat dituangkan kedalam bentuk gambar, sehingga dapat lebih mudah dipahami meski dengan melihat gambar-gambar yang telah tersedia.

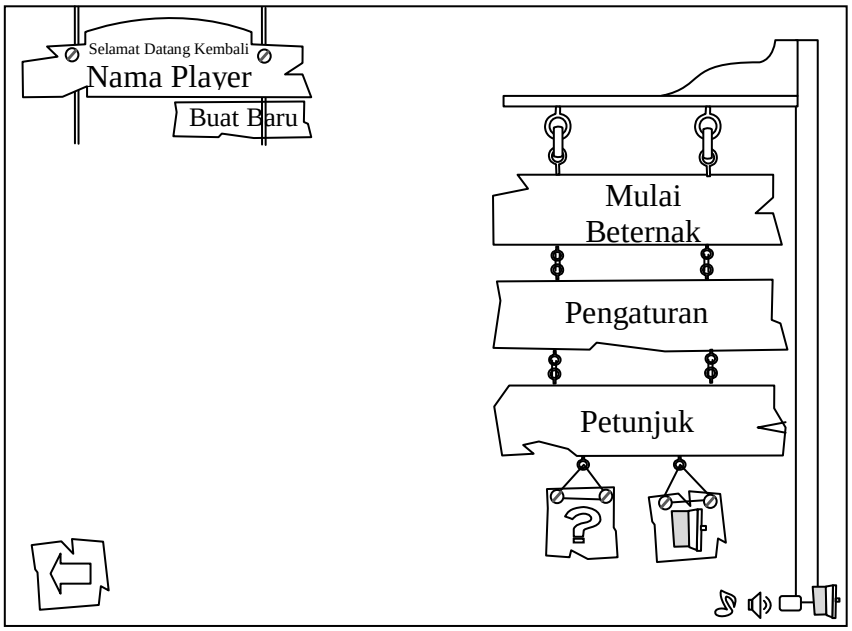
Berikut adalah pemaparan *storyboard* tentang Aplikasi *game* Ternak :



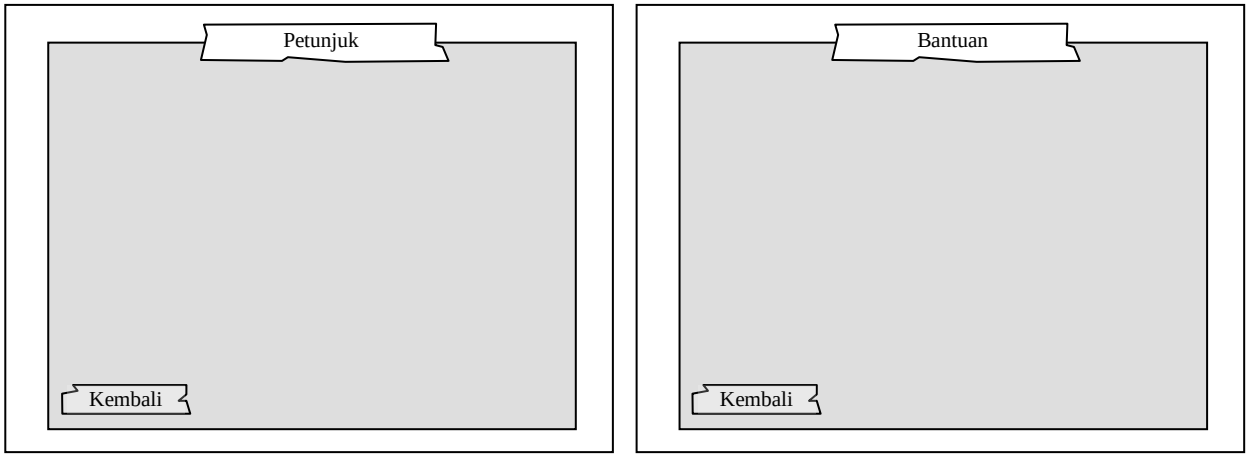
Gambar 3.1 *Story Board* Tampilan Pembuka



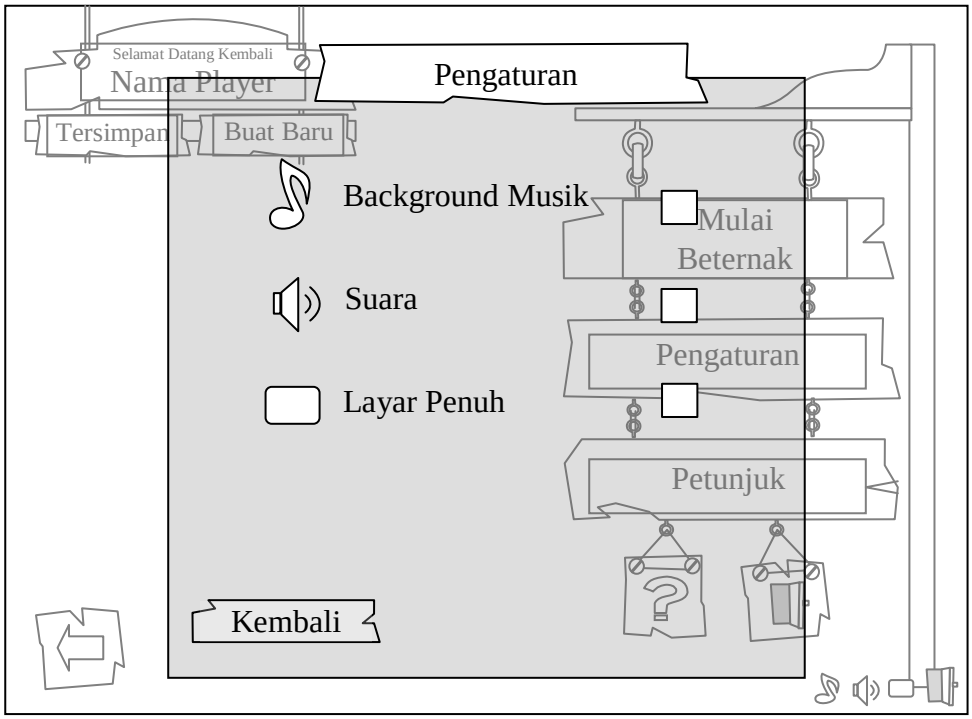
Gambar 3.2 *Story Board* Pilih Karakter



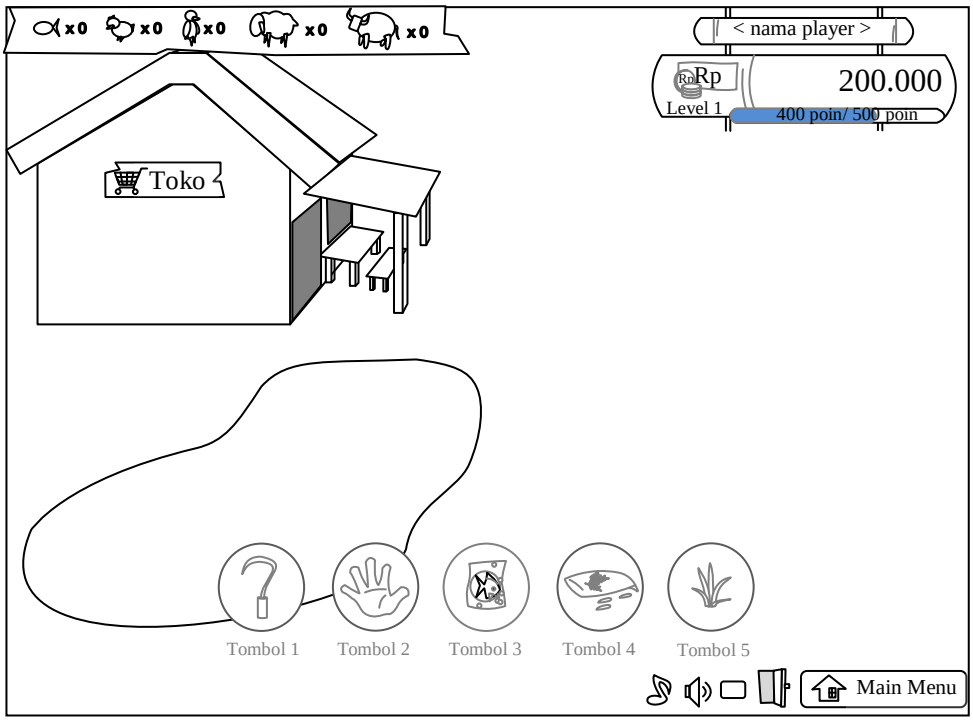
Gambar 3.3 *Story Board* Tampilan Menu



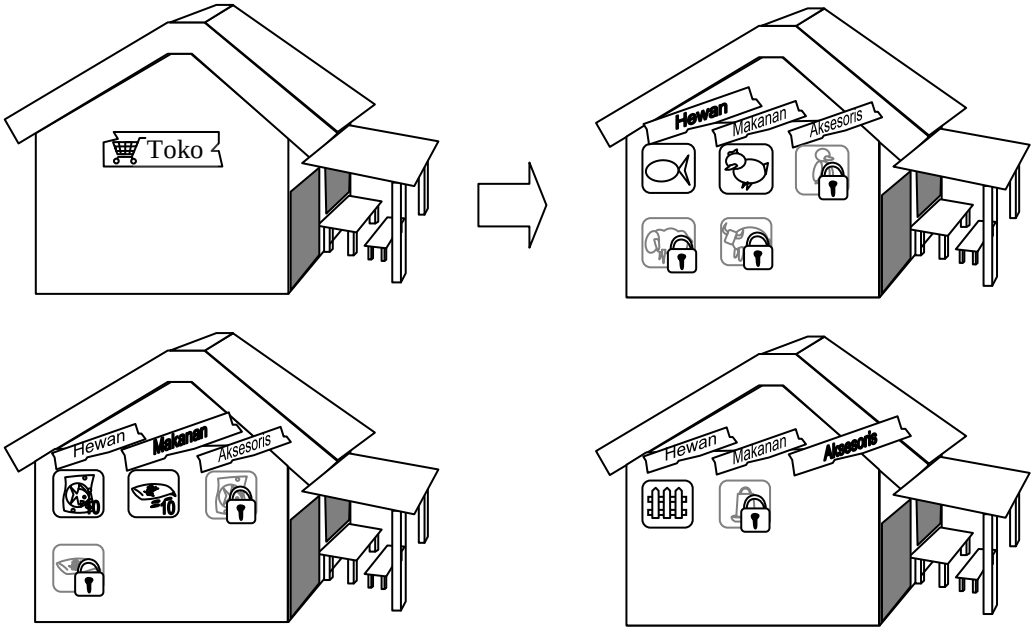
Gambar 3.4 *Story Board* Tampilan Petunjuk dan Bantuan



Gambar 3.5 *Story Board* Tampilan Pengaturan



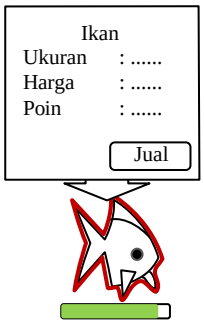
Gambar 3.6 *Story Board* Awal Permainan



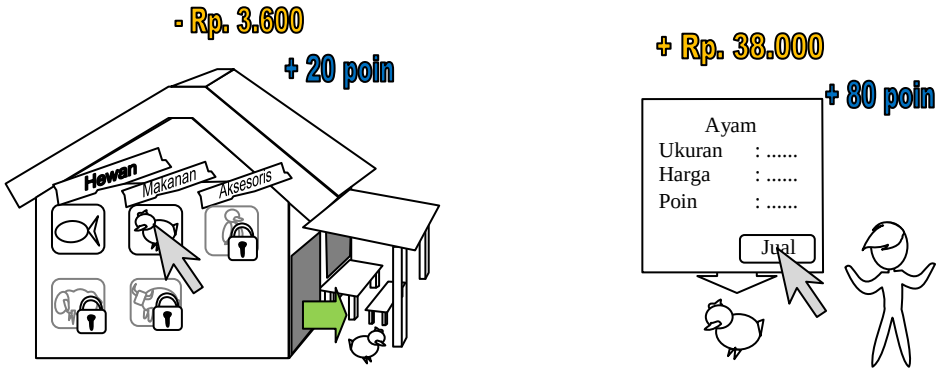
Gambar 3.7 *Story Board* Menu Pembelian



Gambar 3.8 *Story Board Lifebar*



Gambar 3.9 *Story Board* Menu Tampilan Keterangan pada Hewan



Gambar 3.10 *Story Board* Pembelian dan Penjualan Hewan

3.1.2.3 *Gameplay*

Tugas utama pemain (*player*) dalam *game* ini adalah mengembangkan modal utama dengan cara beternak untuk bisa meraih keuntungan dengan memperbanyak ternak sehingga dapat menghasilkan produksi yang banyak. Dalam pengelolaan ternak pun pemain harus menjaga agar ternak yang dimiliki tidak berkurang dikarenakan mati ataupun hilang, sebagai salah satu antisipasi

kehilangan yaitu dengan memasangkan pagar pada wilayah yang dimiliki, untuk menahannya dari berbagai ancaman musuh.

Dalam menjalankan aplikasi ini pemain (*player*) akan langsung di tujukan kepada pemilihan karakter dan pemberian nama karakter tersebut. Setelah itu klik mulai untuk melanjutkan ke tahap permainan, disana akan di tampilkan menu untuk memulai permainan level pertama. Pada level pertama *player* diberi modal uang sebesar Rp. 100.000, dari hasil pengelolaannya peternak akan mendapatkan poin yang mana poin tersebut digunakan untuk naiknya level, pada setiap level tingkatan poin yang harus dicapai itu berbeda, semakin bertambahnya level poin yang harus dicapaipun semakin tinggi, sehingga kesulitan yang dimiliki pemain akan lebih bertambah. Pada setiap level ditanamkan musuh sebagai rintangan yang akan dihadapi. Permainan akan berakhir ketika menang dan kalah, pertama apabila menang keadaan point yang dicapai adalah 5000, dan apabila kalah pada keadaan uang dan ternak habis.

Pada pilihan menu utama juga disediakan pilihan Petunjuk, guna untuk menampilkan informasi petunjuk permainan.

3.1.3 Analisis Algoritma

Algoritma yang akan diterapkan pada analisis pencarian jalur terpendek pada *game* Ternak adalah Algoritma A* (*A star*). Algoritma ini akan digunakan untuk pencarian makan pada hewan ternak, dan pergerakan musuh didalam mengejar mangsanya.

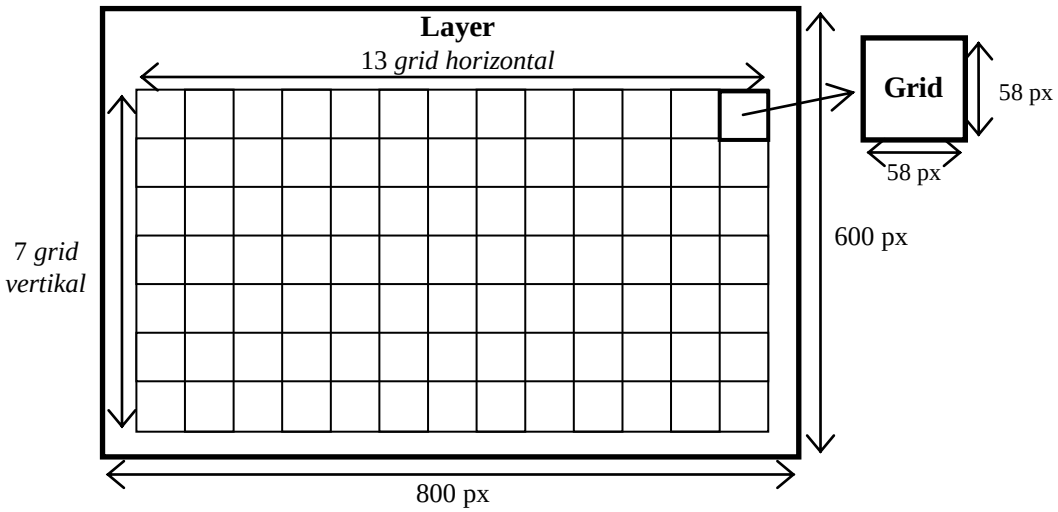
Dalam mencari jalur terpendek sangatlah perlu untuk diterapkannya algoritma, karena algoritma dapat mempermudah cara kerja pada suatu sistem.

3.1.3.1 Analisis Algoritma A* (*A Star*)

Algoritma A* *(A Star)* akan diterapkan pada analisis kasus pencarian jalan terpendek menuju tujuan pada *game pathfinding* untuk mencari jalur dengan cepat, jarak langkah menuju tujuan dan simpul diperiksa.

Sebelum menghitung menggunakan algoritma A* *(A Star)* alangkah baiknya ketahui dulu nilai *cost* yang terdapat pada jalur *game* itu sendiri.

Layer game Ternak dibuat dengan skala *pixel* yang berukuran lebar 800 dan tinggi 600. Dari satu *layer* tersebut disimpan beberapa *grid* (suatu kotak-kotak kecil) kedalamnya dengan ukuran lebar dan tingginya 58 *pixel*, sebanyak 13 *grid horizontal* dan 7 *grid vertikal* jika dijumlahkan keseluruhan *grid*nya menjadi 91 *grid*. Lihat Gambar 3.11.



Gambar 3.11 Keterangan ukuran *layer* dan *grid* (*node*) pada *game*

Dalam kondisi jarak yang ditempuh sangat jauh dan memiliki banyak rintangan algoritma A* cocok untuk mencari solusi yang terbaik. Algoritma ini melakukan pencarian jalur terpendek dengan biaya yang di perhitungkan dari penjumlahan biaya sebenarnya ditambah dengan biaya perkiraan. Dalam notasi matematika algoritma ini dapat dituliskan dengan $f(n) = g(n) + h(n)$.

$f(n)$ = Total biaya yang diperhitungkan

$g(n)$ = Biaya sebenarnya

$h(n)$ = Biaya perkiraan

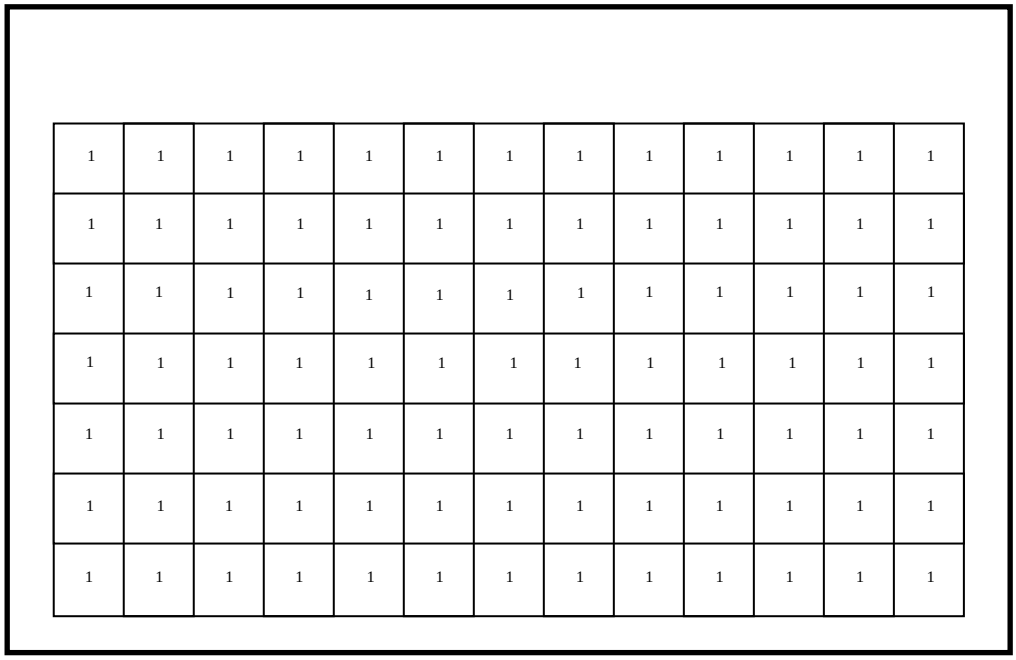
Untuk menentukan $h(n)$ diantara dua jalur maka digunakan rumus *euclidean/phytagoras* :

$$h(n) = \sqrt{(n.x - tujuan.x)^2 + (n.y - tujuan.y)^2}$$

Dengan,

h(n)	= nilai *heuristic* untuk simpul *n*
n.x	= nilai koordinat *x* dari simpul *n*
n.y	= nilai koordinat *y* dari simpul *n*
x-tujuan	= nilai koordinat *x* dari simpul tujuan
y-tujuan	= nilai koordinat *y* dari simpul tujuan

Untuk memudahkan perhitungan maka ada beberapa nilai yang akan diubah (disederhanakan). Pertama nilai *node* (*grid*) yang tadinya 58x58px diubah menjadi 1x1, yang mengibaratkan jarak *node* 1 ke *node* yang lain adalah 1, jadi setiap *node* memiliki *cost* yang sama yaitu 1. Lihat Gambar 3.12.



Gambar 3.12 Nilai *g*(*n*) pada sebuah *node horizontal* dan *vertical*

Kedua perhitungan *euclidean heuristic* nya yang digunakan untuk mencari jarak terpendek antara dua buah *node* (*grid*) *horizontal* dan *vertikal*. Nilai yang diberikan setiap *node*nya yaitu kelipatan dari 58 sedangkan euclidian/diagonalnya 82,02 yang didapat dari $\sqrt{2}\ x\ 58 = 82{,}02$.

Misal nilai x,y = 0 dan tujuan.x, tujuan.y = 58 yang diambil dari jarak *node* pada *game*. 1 *grid* = 58 *pixel* ke arah *horizontal* / 58 *pixel* ke arah *vertical*.

(0,0)	(58,0)	(116,0)	(174,0)	(232,0)	(290,0)	(348,0)	(406,0)	(464,0)	(522,0)	(580,0)	(638,0)	(696,0)
(0,58)	(58,58)	(116, 58)	(174, 58)	(232, 58)	(290, 58)	(348, 58)	(406, 58)	(464, 58)	(522, 58)	(580, 58)	(638, 58)	(696,58)
(0,116)	(58,116)	(116,116)	(174, 116)	(232, 116)	(290, 116)	(348, 116)	(406, 116)	(464, 116)	(522, 116)	(580, 116)	(638, 116)	(696, 116)
(0,174)	(58,174)	(116,174)	(174, 174)	(232, 174)	(290, 174)	(348, 174)	(406, 174)	(464, 174)	(522, 174)	(580, 174)	(638, 174)	(696, 174)
(0,232)	(58,232)	(116,232)	(174,232)	(232,232)	(290, 232)	(348, 232)	(406, 232)	(464, 232)	(522, 232)	(580, 232)	(638, 232)	(696, 232)
(0,290)	(58,290)	(116,290)	(174,290)	(232,290)	(290,290)	(348, 290)	(406, 290)	(464, 290)	(522, 290)	(580, 290)	(638, 290)	(696, 290)
(0,348)	(58,348)	(116,348)	(174,348)	(232,348)	(290,348)	(348,348)	(406,348)	(464,348)	(522,348)	(580,348)	(638,348)	(696,348)

Gambar 3.13 Jarak *x* dan *y* pada game Ternak

Dan untuk penjelasannya sebagai berikut:

$$\sqrt{(\mathrm{x} - \mathrm{tujuan.\,x})^2 + (\mathrm{y} - \mathrm{tujuan.\,y})^2}$$

$$= \sqrt{(0 - 58)^2 + (0 - 58)^2}$$

$$= \sqrt{(-58)^2 + (-58)^2}$$

$$= \sqrt{3364 + 3364}$$

$$= \sqrt{6728}$$

$$= 82{,}02$$

Karena nilainya terlalu besar maka dibuat penyederhanaan seperti pada penyederhanaan yang pertama kelipatan 58 diubah menjadi kelipatan 1, misal nilai x,y = 0 dan tujuan.x, tujuan.y = 1. Lihat Gambar 3.14.

(0,0)	(1,0)	(2,0)	(3,0)	(4,0)	(5,0)	(6,0)	(7,0)	(8,0)	(9,0)	(10,0)	(11,0)	(12,0)
(0,1)	(1,1)	(2, 1)	(3, 1)	(4,1)	(5, 1)	(6, 1)	(7, 1)	(8, 1)	(9, 1)	(10, 1)	(11, 1)	(12,1)
(0,2)	(1,2)	(2,2)	(3, 2)	(4, 2)	(5, 2)	(6, 2)	(7, 2)	(8, 2)	(9, 2)	(10, 2)	(11, 2)	(12, 2)
(0,3)	(1,3)	(2,3)	(3,3)	(4, 3)	(5, 3)	(6, 3)	(7, 3)	(8, 3)	(9, 3)	(10, 3)	(11, 3)	(12, 3)
(0,4)	(1,4)	(2,4)	(3,4)	(4,4)	(5, 4)	(6, 4)	(7, 4)	(8, 4)	(9, 4)	(10, 4)	(11, 4)	(12, 4)
(0,5)	(1,5)	(2,5)	(3,5)	(4,5)	(5,5)	(6, 5)	(7, 5)	(8, 5)	(9, 5)	(10, 5)	(11, 5)	(12, 5)
(0,6)	(1,6)	(2,6)	(3,6)	(4,6)	(5,6)	(6,6)	(7,6)	(8,6)	(9,6)	(10,6)	(11,6)	(12,6)

Gambar 3.14 Penyederhanaan nilai jarak *x* dan *y*

Jika x,y = 0 dan tujuan.x, tujuan.y = 1 maka perhitungannya seperti berikut:

$$\sqrt{(\mathrm{x} - \mathrm{tujuan.\,x})^2 + (\mathrm{y} - \mathrm{tujuan.\,y})^2}$$

$$= \sqrt{(0-1)^2 + (0-1)^2}$$

$$= \sqrt{(-1)^2 + (-1)^2}$$

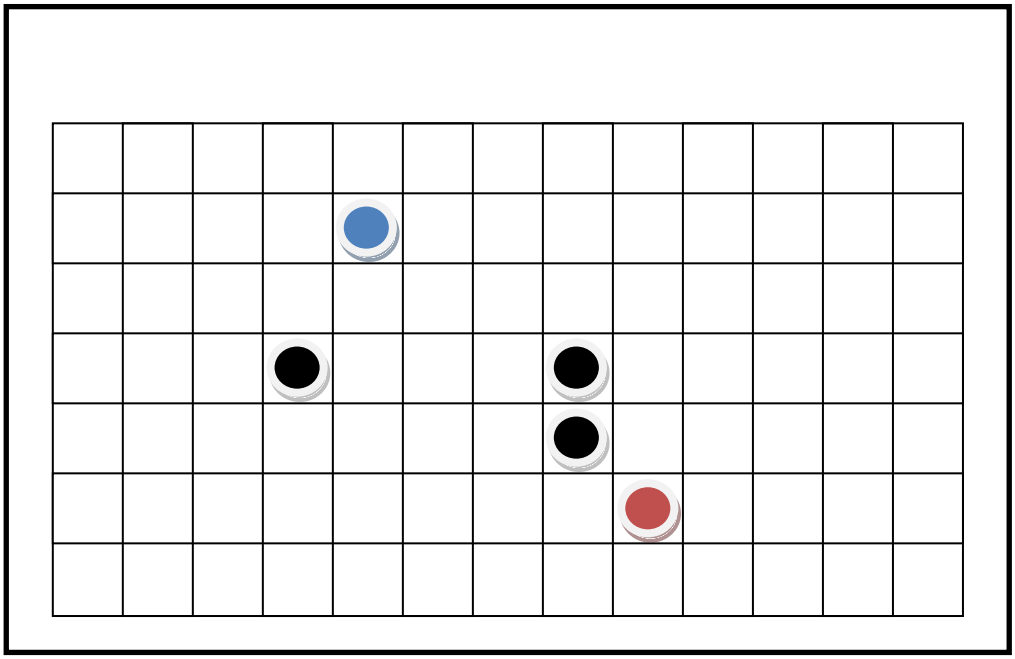
$$= \sqrt{1+1}$$

$$= \sqrt{2}$$

$$= 1{,}41$$

Jadi untuk setiap perhitungan diagonalnya memiliki nilai 1,41

Untuk lebih memahami maka akan dibuat perumpamaan cara kerja perhitungan algoritma A* (*A star*) pada *game* Ternak. Lihat Gambar 3.15.

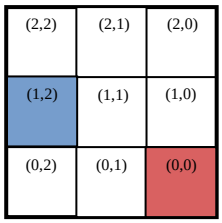


Gambar 3.15 Contoh kasus algoritma A* (*A star*) *pathfinding euclidean distance*

Keterangan pada Gambar 3.1

- Lingkaran biru adalah *node* awal (Hewan ternak)
- Lingkaran merah adalah *node* tujuan (Makanan)
- Lingkaran hitam adalah *obstacle* (Halangan)

Nilai *h(n)/heuristic* di dapat dari jarak perkiraan dari *node* biru (hewan ternak) ke *node* merah (makanan), setiap jarak perkiraannya bernilai kelipatan 1. dan untuk jarak yang arahnya *diagonal* mempunyai nilai kelipatan 1,41 dari $\sqrt{2}$, dan untuk menghitung jarak yang tidak membentuk garis lurus maka digunakan rumus *euclidean distance* dengan mengacu pada nilai koordinat *x* dan *y*. Untuk lebih jelasnya dapat dilihat pada Gambar 3.16 dan 3.17.



Misal dari *node* yang nilainya (0,0) menuju *node* dengan nilai (1,2), perhitungannya :

$= \sqrt{(0-1)^2 + (0-2)^2}$

$= \sqrt{(-1)^2 + (-2)^2}$

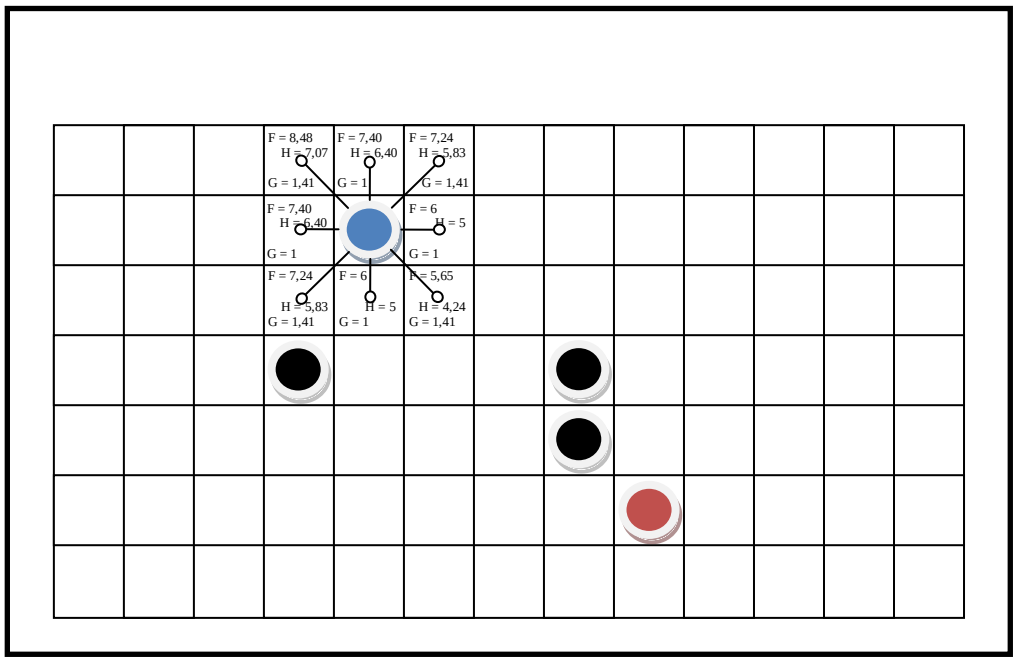
$= \sqrt{1+4}$ $= \sqrt{5} = 2{,}23$

Gambar 3.16 Nilai *heuristic* pada kasus *diagonal* tidak lurus

H = 9,43	H = 8,60	H = 7,81	H = 7,07	H = 6,40	H = 5,83	H = 5,38	H = 5,09	H = 5	H = 5,09	H = 5,38	H = 5,83	H = 6,40
H = 8,94	H = 8,06	H = 7,21	H = 6,40	H = 5,65	H = 5	H = 4,47	H = 4,12	H = 4	H = 4,12	H = 4,47	H = 5	H = 5,65
H = 8,54	H = 7,61	H = 6,70	H = 5,83	H = 5	H = 4,24	H = 3,60	H = 3,16	H = 3	H = 3,16	H = 3,60	H = 4,24	H = 5
H = 8,24	H = 7,28	H = 6,32		H = 4,47	H = 3,60	H = 2,82		H = 2	H = 2,23	H = 2,82	H = 3,60	H = 4,47
H = 8,06	H = 7,07	H = 6,08	H = 5,09	H = 4,12	H = 3,16	H = 2,23		H = 1	H = 1,41	H = 2,23	H = 3,16	H = 4,12
H = 8	H = 7	H = 6	H = 5	H = 4	H = 3	H = 2	H = 1	H = 0	H = 1	H = 2	H = 3	H = 4
H = 8,06	H = 7,07	H = 6,08	H = 5,09	H = 4,12	H = 3,16	H = 2,23	H = 1,41	H = 1	H = 1,41	H = 2,23	H = 3,16	H = 4,12

Gambar 3.17 Nilai *heuristic* pada kasus pencarian jalur terpendek

Untuk pergerakan secara *horizontal* dan *vertikal* (*x*,*y*) nilai *cost*nya adalah 1 sedangkan untuk nilai *diagonal* (*euclidean heurstic*) nya adalah 1,41 diambil dari $\sqrt{2} = 1{,}41$. Di bawah ini merupakan langkah-langkah pencarian jalur terpendek menggunakan algoritma A* (*A Star*). Langkah pertama lihat pada Gambar 3.18.

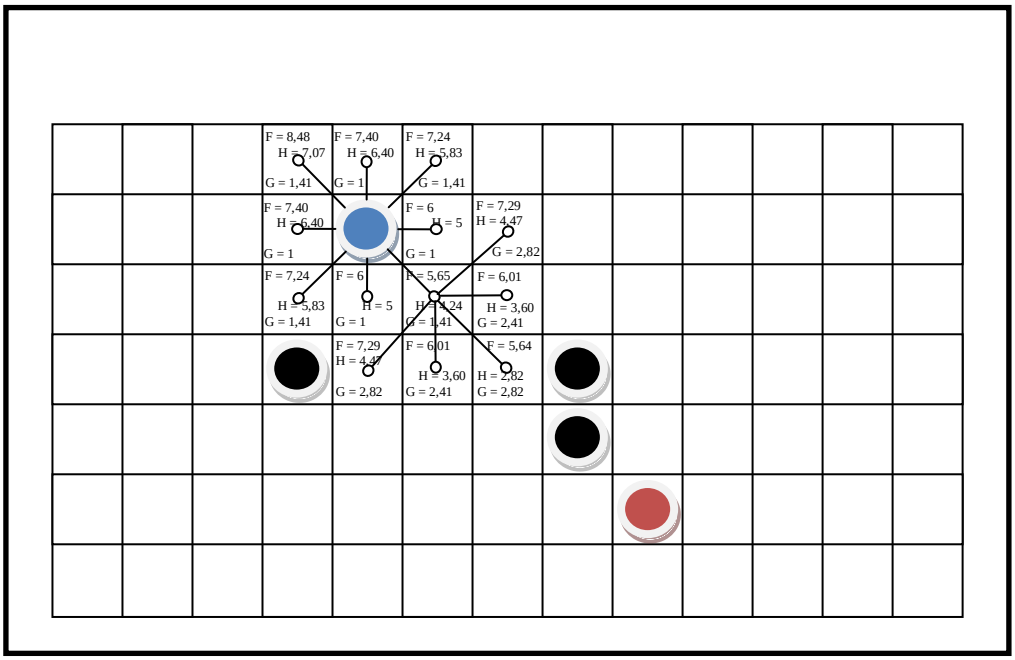


Gambar 3.18 Pencarian jalur terpendek dengan algoritma A* (*A Star*) 1

Dari *node* biru (Hewan) akan melakukan pencarian secara *horizontal*, *vertical* dan juga *diagonal* menuju *node* merah (Makanan). Pertama-tama pencarian dilakukan ke arah :

- *Vertical* atas : $g = 1$, $h = 6{,}40$ dan $f = g + h = 7{,}40$.
  bawah : $g = 1$, $h = 5$ dan $f = g + h = 6$.
- *Horizontal* kanan : $g = 1$, $h = 5$ dan $f = g + h = 6$.
  kiri : $g = 1$, $h = 6{,}40$ dan $f = g + h = 7{,}40$.
- *Diagonal* kanan atas : $g = 1{,}41$, $h = 5{,}83$ dan $f = g + h = 7{,}24$.
  kanan bawah : $g = 1{,}41$, $h = 4{,}24$ dan $f = g + h = 5{,}65$.
  kiri atas : $g = 1{,}41$, $h = 7{,}07$ dan $f = g + h = 8{,}48$.
  kiri bawah : $g = 1{,}41$, $h = 5{,}83$ dan $f = g + h = 7{,}24$.

Berikut ini adalah langkah kedua pencarian jalur terpendek dengan algoritma A* (*A Star*) yang dapat dilihat pada gambar 3.19.



Gambar 3.19 Pencarian jalur terpendek dengan algoritma A* (*A Star*) 2

Karena *node* dengan nilai *f* = 5,65 (*diagonal* kanan bawah) adalah *cost* yang paling murah, maka kini pencarian di mulai dari *node* ini. Ke arah :

- *Vertical* bawah : *g* = 2,41, *h* = 3,60 dan *f* = *g* + *h* = 8.
- *Horizontal* kanan : *g* = 2,41, *h* = 3,60 dan *f* = *g* + *h* = 8.
- *Diagonal* kanan atas : *g* = 2,82, *h* = 4,47 dan *f* = *g* + *h* = 7,29.
  kanan bawah : *g* = 2,82, *h* = 2,82 dan *f* = *g* + *h* = 5,64.
  kiri bawah : *g* = 2,82, *h* = 4,47 dan *f* = *g* + *h* = 7,29.

Nilai *g* = 2,41 pada *node vertical* dan *horizontal* didapat dari nilai *g* sebelumnya yaitu 1,41 ditambah 1 (*cost* sebuah *node*), dan *g* = 2,82 pada *node*

diagonal didapat dari nilai *g* sebelumnya yaitu 1,41 ditambah 1,41 (*cost* sebuah *node*). Atau perhitungannya seperti berikut :

$$\sqrt{(0-2)^2+(0-2)^2}$$

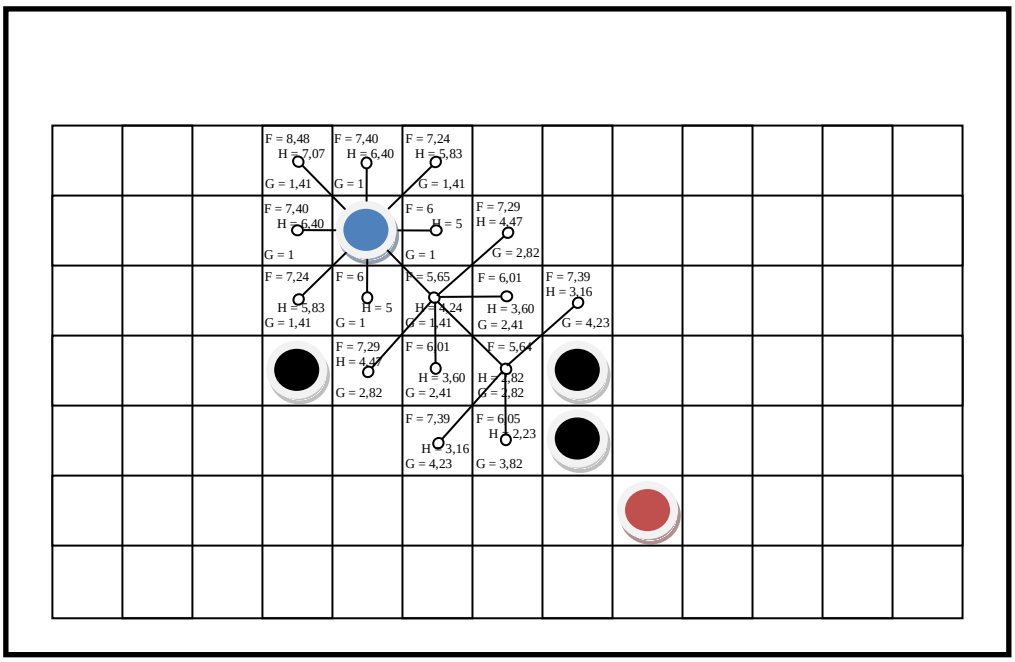
$$=\sqrt{(-2)^2+(-2)^2}$$

$$=\sqrt{4+4}$$

$$=\sqrt{8}$$

$$=2{,}82$$

Berikut ini adalah langkah ketiga pencarian jalur terpendek dengan algoritma A* (*A Star*) yang dapat dilihat pada gambar 3.20.



Gambar 3.20 Pencarian jalur terpendek dengan algoritma A* (*A Star*) 3

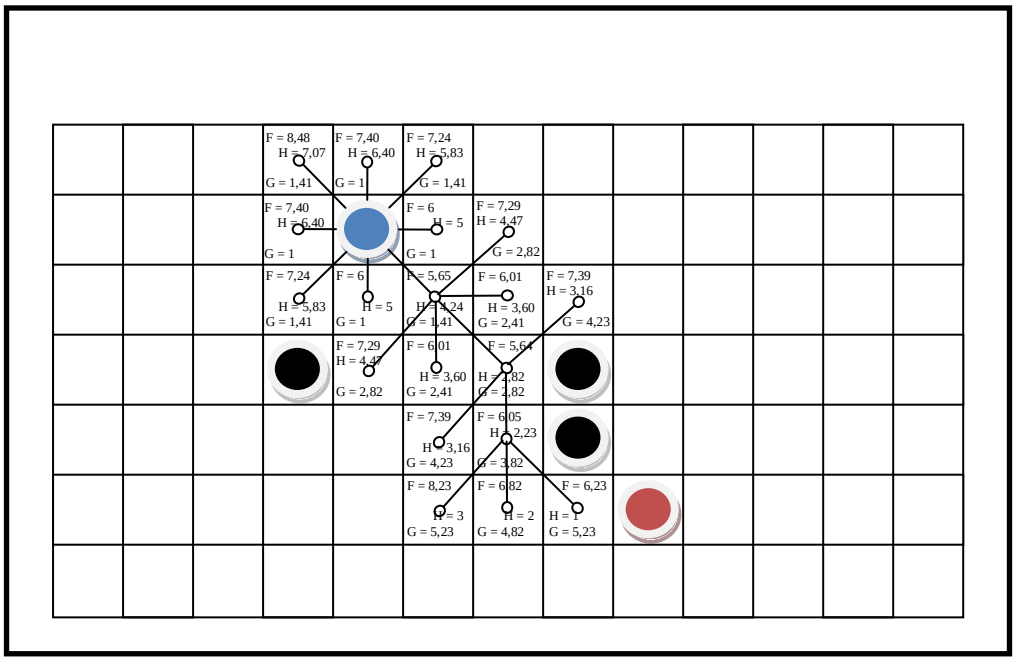
Karena *node* dengan nilai *f* = 5,64 (*diagonal* kanan bawah) adalah *cost* yang paling murah, maka kini pencarian di mulai dari *node* ini. Ke arah :

- *Vertical* bawah : *g* = 3,82, *h* = 2,23 dan *f* = *g* + *h* = 6,05.
- *Diagonal* kanan atas : *g* = 4,23, *h* = 3,16 dan *f* = *g* + *h* = 7,39.
  kiri bawah : *g* = 4,23, *h* = 3,16 dan *f* = *g* + *h* = 7,39.

Nilai *g* = 3,82 pada *node vertical* dan *horizontal* didapat dari nilai *g* sebelumnya yaitu 2,82 ditambah 1 (*cost* sebuah *node*), dan *g* = 4,23 pada *node diagonal* didapat dari nilai *g* sebelumnya yaitu 2,82 ditambah 1,41 (*cost* sebuah *node*). Atau perhitungannya seperti berikut :

$$\sqrt{(0-3)^2+(0-3)^2}$$
$$=\sqrt{(-3)^2+(-3)^2}$$
$$=\sqrt{9+9}$$
$$=\sqrt{18}$$
$$=4{,}24$$

Berikut ini adalah langkah keempat pencarian jalur terpendek dengan algoritma A* (*A Star*) yang dapat dilihat pada gambar 3.21.



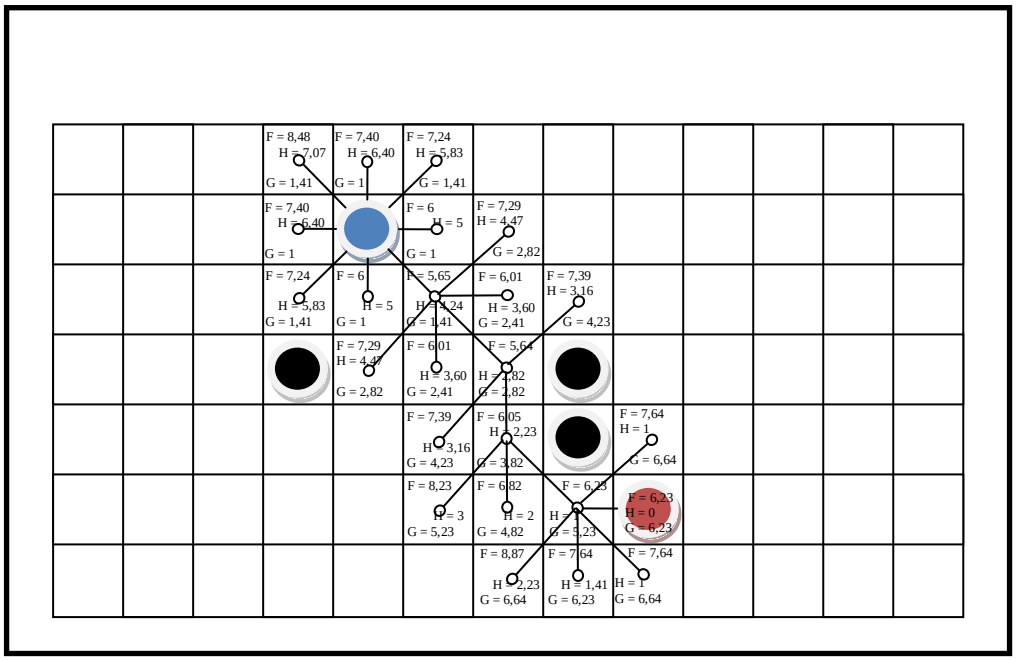
Gambar 3.21 Pencarian jalur terpendek dengan algoritma A* (*A Star*) 4

Karena *node* dengan nilai *f* = 6,05 (*vertical* bawah) adalah *cost* yang paling murah, maka kini pencarian di mulai dari *node* ini. Ke arah :

- *Vertical* bawah : $g = 4{,}82$, $h = 2$ dan $f = g + h = 6{,}82$.
- *Diagonal* kanan bawah : $g = 5{,}23$, $h = 1$ dan $f = g + h = 6{,}23$.
  kiri bawah : $g = 5{,}23$, $h = 3$ dan $f = g + h = 8{,}23$.

Nilai *g* = 4,82 pada *node vertical* dan *horizontal* didapat dari nilai *g* sebelumnya yaitu 3,82 ditambah 1 (*cost* sebuah *node*), dan *g* = 5,23 pada *node diagonal* didapat dari nilai *g* sebelumnya yaitu 3,82 ditambah 1,41 (*cost* sebuah *node*).

Berikut ini adalah langkah kelima pencarian jalur terpendek dengan algoritma A* (*A Star*) yang dapat dilihat pada gambar 3.22.



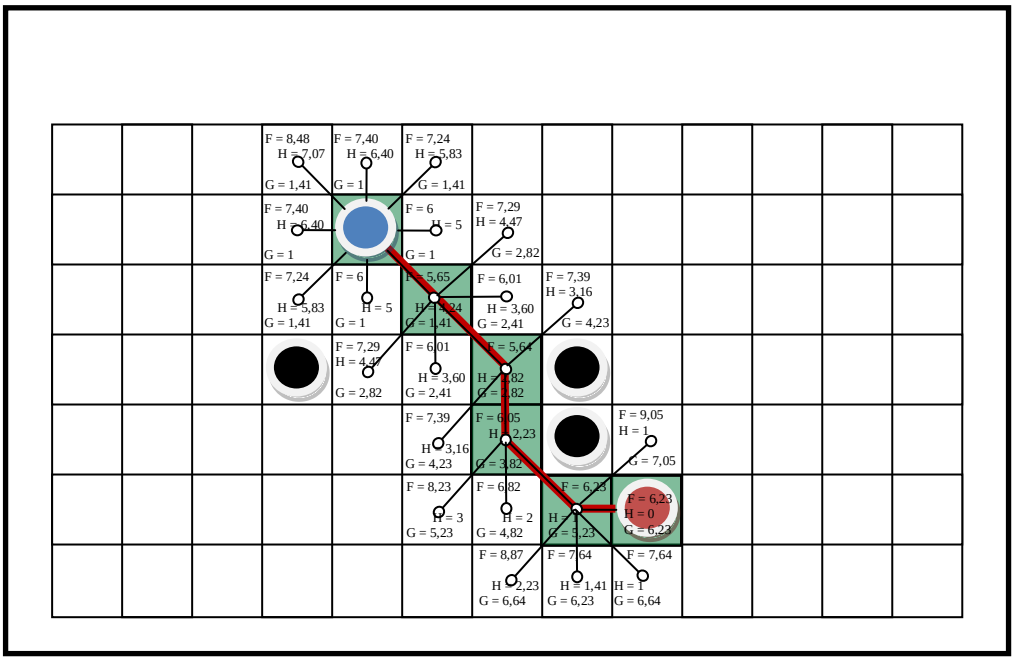
Gambar 3.22 Pencarian jalur terpendek dengan algoritma A* (*A Star*) 5

Karena *node* dengan nilai *f* = 6,23 (*diagonal* kanan bawah) adalah *cost* yang paling murah, maka kini pencarian di mulai dari *node* ini. Ke arah :

- *Vertical* bawah : g = 6,23, h = 1 dan $f = g + h$ = 7,23.
- *Horizontal* kanan : g = 6,23, h = 0 dan $f = g + h$ = 6,23.
- *Diagonal* kanan atas : g = 6,64, h = 1 dan $f = g + h$ = 7,64.
  kanan bawah : g = 6,64, h = 1,41 dan $f = g + h$ = 7,64.
  kiri bawah : g = 6,64, h = 2,23 dan $f = g + h$ = 8,87.

Nilai *g* = 6,23 pada *node vertical* dan *horizontal* didapat dari nilai *g* sebelumnya yaitu 5,23 ditambah 1 (*cost* sebuah *node*), dan *g* = 7,05 pada *node diagonal* didapat dari nilai *g* sebelumnya yaitu 5,23 ditambah 1,41 (*cost* sebuah *node*).

Dari perhitungan diatas maka hasil jalur terpendek dengan algoritma A* (*A Star*) terdapat pada *node horizontal* kanan dengan nilai *f* = 6,23 yang dapat dilihat pada gambar 3.23.



Gambar 3.23 Hasil pencarian jalur terpendek dengan algoritma A* (*A Star*)

Hasil pencarian jalur terpendek menggunakan algoritma A* (*A Star*) dari *node* berwarna biru (Hewan) menuju *node* berwarna merah (Makanan) yang didapatkan adalah sebagai berikut :

1. Dari *node* berwarna biru (Hewan) mengambil jalur ke arah *node diagonal* kanan bawah dengan nilai *f* = 5,65.
2. Dari *node diagonal* kanan bawah bernilai *f* = 5,65 mengambil jalur ke arah *node diagonal* kanan bawah dengan nilai *f* = 5,64.
3. Dari *node diagonal* kanan bawah bernilai *f* = 5,64 mengambil jalur ke arah *node vertical* bawah dengan nilai *f* = 6,05.
4. Dari *node vertical* bawah bernilai *f* = 6.05 mengambil jalur ke arah *node diagonal* kanan bawah dengan nilai *f* = 6,23.
5. Dari *node diagonal* kanan bawah bernilai *f* = 6,23 mengambil jalur ke arah *node horizontal* kanan dan sampailah di *node* berwarna merah (Makanan).dengan nilai *f* = 6,23.

3.2 Analisis Kebutuhan Non-Fungsional

Analisis kebutuhan non-fungsional merupakan analisis yang digunakan untuk menentukan spesifikasi kebutuhan didalam perancangan aplikasi. Spesifikasi ini meliputi elemen atau komponen-komponen yang dibutuhkan untuk membangun aplikasi hingga pengimplementasiannya.

3.2.1 Analisis Kebutuhan Perangkat Lunak

Perangkat lunak digunakan dalam sebuah sistem merupakan perintah-perintah yang diberikan kepada perangkat keras agar bisa saling berinteraksi diantara keduanya. Perangkat lunak yang dibutuhkan untuk membangun aplikasi *game* Ternak ini adalah sebagai berikut :

1. Sistem Operasi Windows XP, Windows Vista, Windows 7.
2. Adobe Flash CS3, CS4, CS5 Professional
3. Adobe Flash Player 9,10.

3.2.2 Analisis Kebutuhan Perangkat Keras

Komputer terdiri dari perangkat keras dan perangkat lunak yang saling berinteraksi. Perangkat lunak memberikan instruksi-instruksi kepada perangkat keras untuk melakukan suatu tugas tertentu, sehingga dapat menjalankan suatu sistem di dalamnya.

Perangkat keras yang dibutuhkan untuk mendukung *game* Ternak ini adalah sebagai berikut :

1. Processor : 1,8 Ghz ke atas.
2. Hardisk : 40 Gb.
3. Memory : 512 Mb.
4. VGA : 128 Mb.
5. Monitor : CRT/LCD 15 inch.
6. Mouse & Keyboard.

3.2.3 Analisis dan Kebutuhan *User*

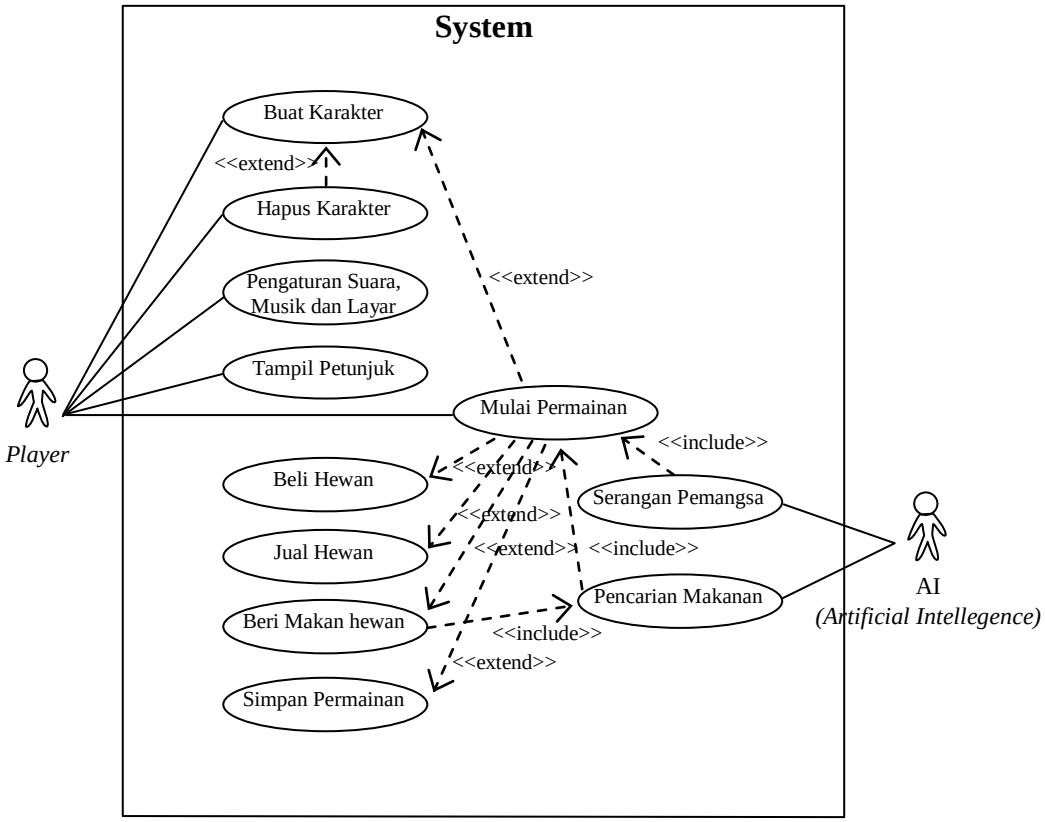
Game yang dibuat ditujukan untuk berbagai usia khususnya anak-anak dan remaja, mulai dari umur 5 tahun sampai 18 tahun, alasannya karena pada usia itu anak-anak ataupun remaja mampu meningkatkan ketajaman dan akurasi berpikir. Usia itu juga merupakan usia yang efektif untuk mempelajari berbagai hal dan merupakan kurun waktu yang sangat penting dan kritis dalam hal tumbuh kembangnya fisik, mental, dan psikososial. Maka banyak pula orang-orang menyebut hal demikian itu sebagai masa keemasan (*the golden age*).

3.3 Analisis Kebutuhan Fungsional

Analisis kebutuhan fungsional adalah analisis yang digunakan untuk membuat aliran data terstruktur. Analisis sistem yang dilakukan menggunaka tools UML. Dengan tahapan perancangan *use case diagaram*, *use case scenario*, *activity diagram*, dan *squence diagram* yang akan digambarkan pada sub bab berikut ini.

3.3.1 Use Case Diagram

Berikut ini adalah perancangan Use Case Diagram pada *game* Ternak :



Gambar 3.24 Use Case Diagram *Game* Ternak

Definisi Use Case :

Tabel 3.2 Definisi Use Case

No	Use Case	Deskripsi
1	Buat Karakter	Proses untuk membuat karakter peternak pada permainan
2	Hapus Karakter	Proses untuk menghapus karakter yang telah dibuat/disimpan
3	Pengaturan Suara, Musik dan Layar	Proses untuk mengatur suara, musik dan tampilan layar

4	Tampil Petunjuk	Proses untuk menampilkan tampilan petunjuk
5	Mulai Permainan	Proses untuk permainan Ternak
6	Beli Hewan	Proses untuk membeli hewan ternak
7	Jual Hewan	Proses untuk menjual hewan ternak
8	Beri Makan Hewan	Proses pemberian makanan hewan ternak
9	Simpan Permainan	Proses menyimpan permainan

3.3.2 Use Case Scenario

Skenario setiap bagian pada use case menunjukkan proses apa yang terjadi pada setiap bagian didalam use case tersebut, dimana user memberikan perintah pada setiap bagian dan respon apa yang diberikan oleh sistem kepada user setelah user memberikan perintah pada setiap bagian-bagian use case.

Tabel 3.3 Use Case Scenario Buat Karakter

Identifikasi		
Nomor	1	
Nama	Buat Karakter	
Tujuan	Membuat karakter peternak	
Deskripsi	Proses untuk membuat karakter peternak pada permainan	
Aktor	Player	
Trigger	☐ Internal ☑ Eksternal	
Skenario Utama		
Kondisi awal	Player berada pada tampilan Pilih Karakter	
Aksi Aktor		**Reaksi Sistem**
a. Memilih karakter		

	b. Menampilkan karakter laki-laki atau perempuan
c. Mengisi nama	
	d. Nama di tuliskan
e. Memilih menu Simpan	
	f. Menyimpan nama dan karakter yang telah dipilih
Kondisi akhir	Aplikasi menyimpan nama dan karakter yang dipilih

Tabel 3.4 Use Case Scenario Hapus Karakter

Identifikasi	
Nomor	2
Nama	Hapus Karakter
Tujuan	Menghapus karakter peternak
Deskripsi	Proses untuk menghapus karakter yang telah dibuat/disimpan
Aktor	Player
Trigger	☐ Internal ☑ Eksternal
Skenario Utama	
Kondisi awal	Player berada pada tampilan Pilih Karakter
Aksi Aktor	**Reaksi Sistem**
a. Memilih menu Hapus	
	b. Menghapus Karakter yang telah dibuat/disimpan
Kondisi akhir	Aplikasi menghapus karakter yang ada

Tabel 3.5 Use Case Scenario Pilihan Pengaturan Suara, Musik dan Layar

Identifikasi	
Nomor	3
Nama	Pengaturan Suara, Musik dan Layar
Tujuan	Mengatur efek suara, musik dan layar
Deskripsi	Proses untuk mengatur suara, musik dan tampilan layar
Aktor	Player
Trigger	☐ Internal ☑ Eksternal
Skenario Utama	
Kondisi awal	Player berada pada tampilan menu utama
Aksi Aktor	**Reaksi Sistem**
a. Memilih menu Pengaturan	
	b. Menampilkan tampilan pengaturan
c. Mengatur suara	
	d. Mengaktifkan suara
e. Mengatur musik	
	e. Mengaktifkan musik
f. Mengatur Layar	
	g. Mengaktifkan layar penuh
Kondisi akhir	Aplikasi berada pada tampilan Pengaturan

Tabel 3.6 Use Case Scenario Tampil Petunjuk

Identifikasi	
Nomor	4
Nama	Tampil Petunjuk
Tujuan	Menampilkan informasi petunjuk
Deskripsi	Proses untuk menampilkan tampilan petunjuk
Aktor	Player
Trigger	☐ Internal ☑ Eksternal
Skenario Utama	
Kondisi awal	Player berada pada tampilan menu utama
Aksi Aktor	**Reaksi Sistem**
a. Memilih menu Petunjuk	
	b. Menampilkan informasi petunjuk permainan
Kondisi akhir	Aplikasi berada pada tampilan Petunjuk

Tabel 3.7 Use Case Scenario Mulai Permainan

Identifikasi	
Nomor	5
Nama	Mulai Permainan
Tujuan	Bermain permainan Ternak
Deskripsi	Proses untuk permainan Ternak
Aktor	Player
Trigger	☐ Internal ☑ Eksternal

Skenario Utama		
Kondisi awal	Player berada pada tampilan Permainan	
Aksi Aktor		**Reaksi Sistem**
a. Memilih menu Toko		
		b. Menampilkan menu pembelian
c. Membeli hewan Ternak		
		d. Menambahkan hewan ternak pada permainan
e. Memberi makan hewan		
		f. Hewan mencari makanan yang telah di simpan
g. Menjual hewan		
		h. Mengurangi hewan ternak pada permainan
Kondisi akhir	Aplikasi berada pada Permainan	

Tabel 3.8 Use Case Scenario Beli Hewan

Identifikasi	
Nomor	6
Nama	Beli Hewan
Tujuan	Membeli hewan ternak
Deskripsi	Proses untuk membeli hewan ternak
Aktor	Player
Trigger	☐ Internal ☑ Eksternal

Skenario Utama		
Kondisi awal	Player berada pada tampilan Permainan	
Aksi Aktor		**Reaksi Sistem**
a. Memilih menu Toko		
		b. Menampilkan menu pembelian
c. Membeli hewan		
		d. Menambahkan hewan ternak pada permainan
Kondisi akhir	Aplikasi berada pada tampilan menu toko	

Tabel 3.9 Use Case Scenario Jual Hewan

Identifikasi		
Nomor	7	
Nama	Jual Hewan	
Tujuan	Menjual hewan ternak	
Deskripsi	Proses untuk menjual hewan ternak	
Aktor	Player	
Trigger	☐ Internal ☑ Eksternal	
Skenario Utama		
Kondisi awal	Player berada pada tampilan Permainan	
Aksi Aktor		**Reaksi Sistem**
a. Memilih hewan yang akan dijual		
		b. Menampilkan informasi hewan

c. Memilih menjual hewan	
	d. Mengurangi hewan ternak pada permainan
Kondisi akhir	Aplikasi berada pada tampilan Permainan

Tabel 3.10 Use Case Scenario Beri Makan Hewan

Identifikasi	
Nomor	8
Nama	Beri Makan Hewan
Tujuan	Memberi makan hewan ternak
Deskripsi	Proses pemberian makanan hewan ternak
Aktor	Player
Trigger	☐ Internal ☑ Eksternal
Skenario Utama	
Kondisi awal	Player berada pada tampilan Permainan
Aksi Aktor	**Reaksi Sistem**
a. Memilih tombol pemberian makanan	
	b. Tombol pemberian makanan aktif
c. Menyimpan makanan	
	d. Hewan mencari makanan yang telah di simpan
Kondisi akhir	Aplikasi berada pada tampilan Permainan

Tabel 3.11 Use Case Scenario Simpan Permainan

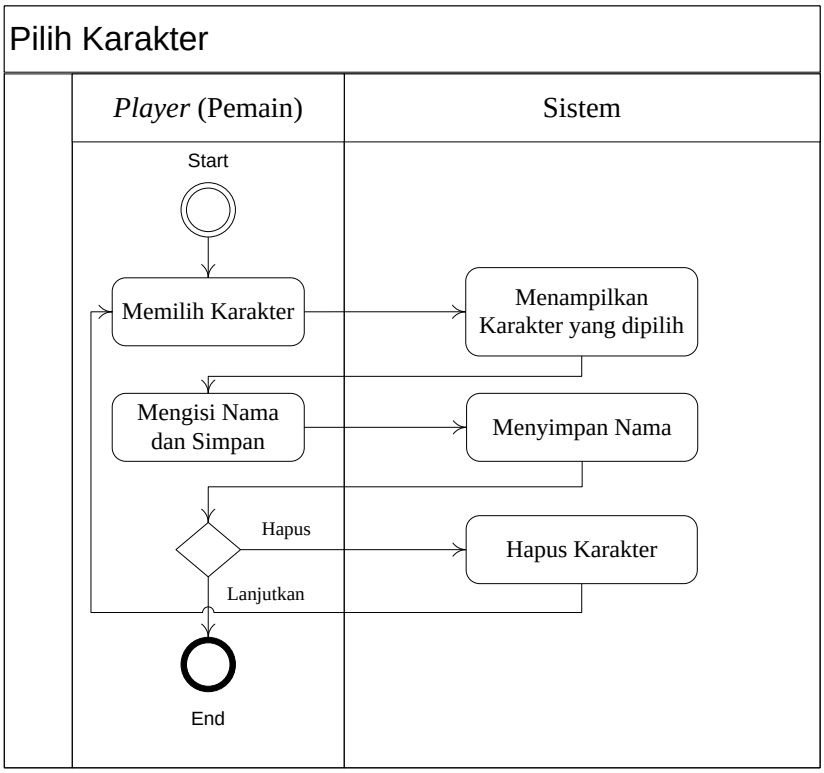
Identifikasi		
Nomor	9	
Nama	Simpan Permainan	
Tujuan	Menyimpan permainan	
Deskripsi	Proses menyimpan permainan	
Aktor	Player	
Trigger	☐ Internal ☑ Eksternal	
Skenario Utama		
Kondisi awal	Player berada pada tampilan Permainan	
Aksi Aktor		**Reaksi Sistem**
a. Memilih tombol menu utama		
		b. Aplikasi menampilkan pesan Simpan dan kembali ke menu utama
c. Memilih Simpan		
		d. Menyimpan Permainan
Kondisi akhir	Aplikasi kembali ke menu utama	

3.3.3 Activity Diagram

1. Proses Pilih Karakter

Tabel 3.12 Penjelasan View Pilih Karakter

Deskripsi	Tampilan ini merupakan pilihan untuk membuat karakter
Normal Flow	1. *Player* memilih Karakter 2. *Player* mengisi nama dan Simpan 3. *Player* memilih hapus 4. *Player* memilih lanjutkan

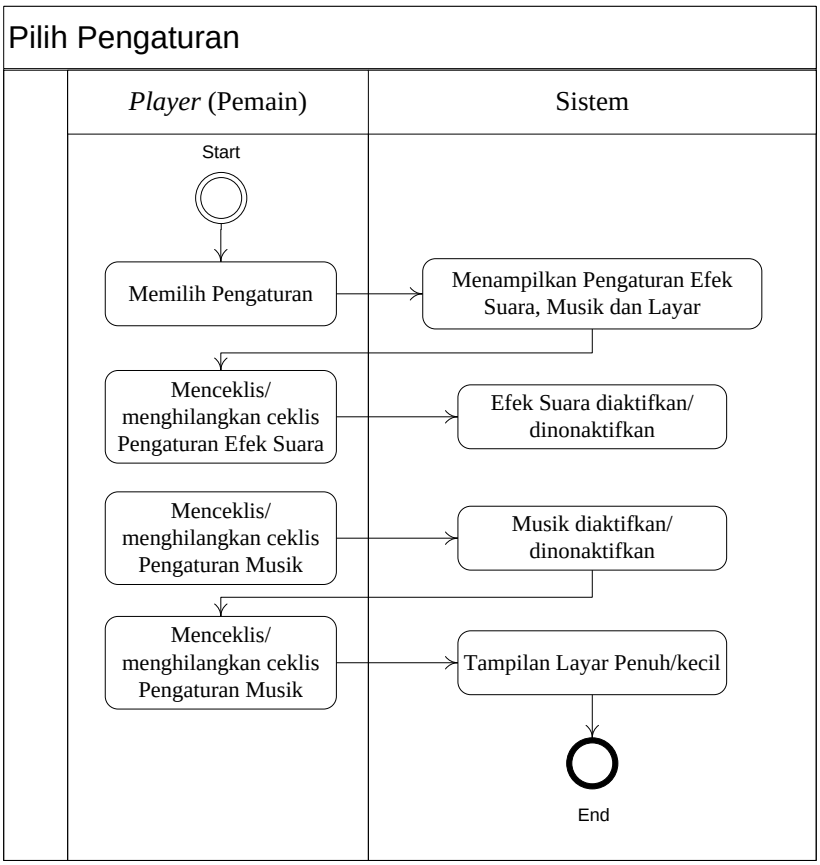


Gambar 3.25 *Activity Diagram* Pada Proses Pilih Karakter

2. Proses Pilih Pengaturan

Tabel 3.13 Penjelasan View Pilih Pengaturan

Deskripsi	Tampilan ini merupakan pilihan untuk mengatur Efek suara, Musik dan Tampilan Layar
Normal Flow	1. *Player* memilih pilihan Pengaturan 2. *Player* memilih menceklis/menghilangkan ceklis pengaturan Efek suara 3. *Player* memilih menceklis/menghilangkan ceklis pengaturan Musik 4. *Player* memilih menceklis/menghilangkan ceklis pengaturan Layar

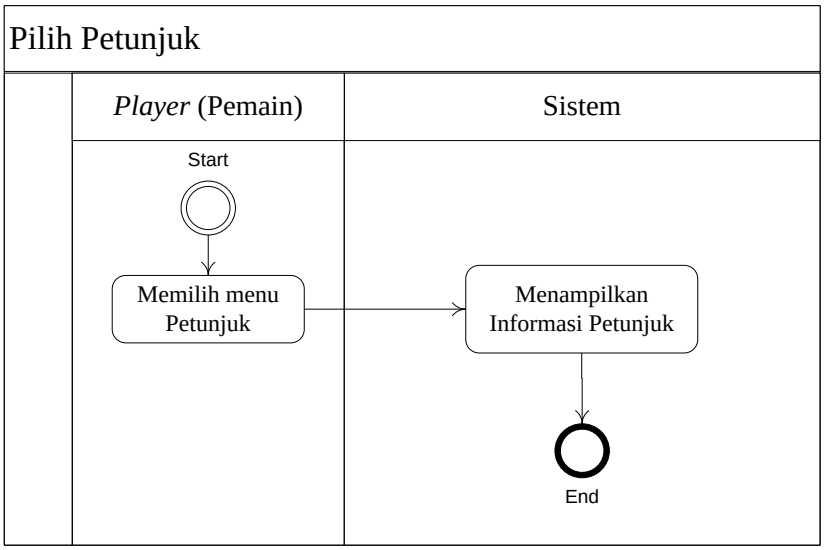


Gambar 3.25 *Activity Diagram* Pada Proses Pilih Pengaturan

3. Proses View Petunjuk

Tabel 3.14 Penjelasan View Petunjuk

Deskripsi	Tampilan ini merupakan pilihan untuk melihat informasi *gameplay* dan cara menjalankan permainan
Normal Flow	1. *Player* memilih pilihan Petunjuk untuk melihat informasi *gameplay* dan tata cara permainan 2. Keluar dari tampilan petunjuk

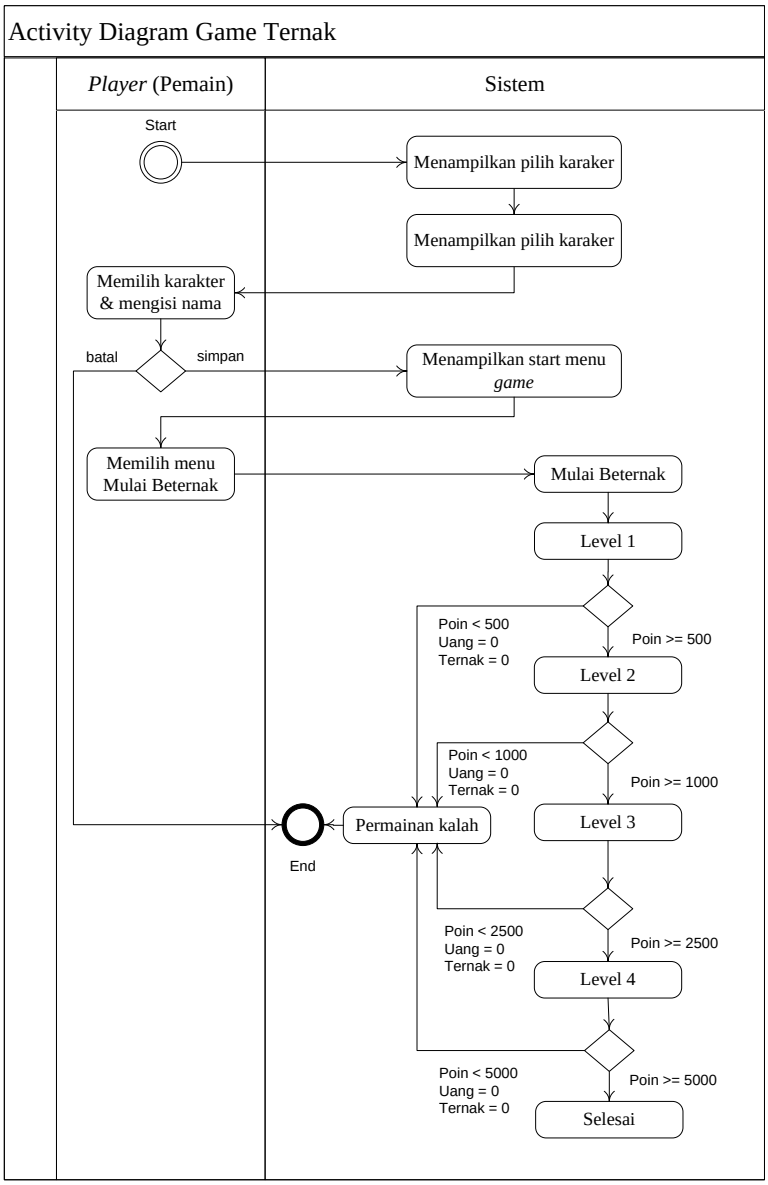


Gambar 3.27 *Activity Diagram* Pada Proses View Petunjuk

4. Proses Mulai Game Ternak

Tabel 3.15 Penjelasan Mulai Game Ternak

Deskripsi	Tampilan ini merupakan pilihan untuk masuk dan memulai permainan
Normal Flow	1. *Player* memilih karakter dan memberi nama 2. *Player* memilih menu Mulai Beternak 3. *Player* langsung memulai permainan Ternak dari Level 1 sampai Level 4

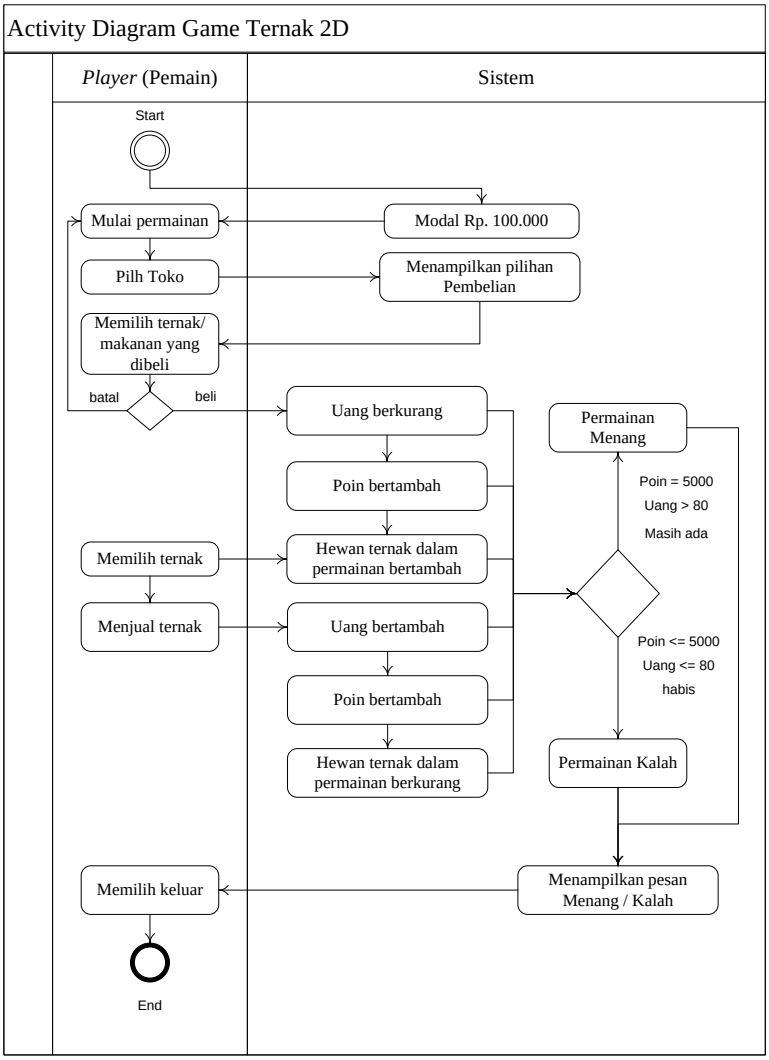


Gambar 3.28 *Activity Diagram* Pada Proses Mulai Permainan

4. Proses Mulai Permainan

Tabel 3.16 Penjelasan Mulai Permainan

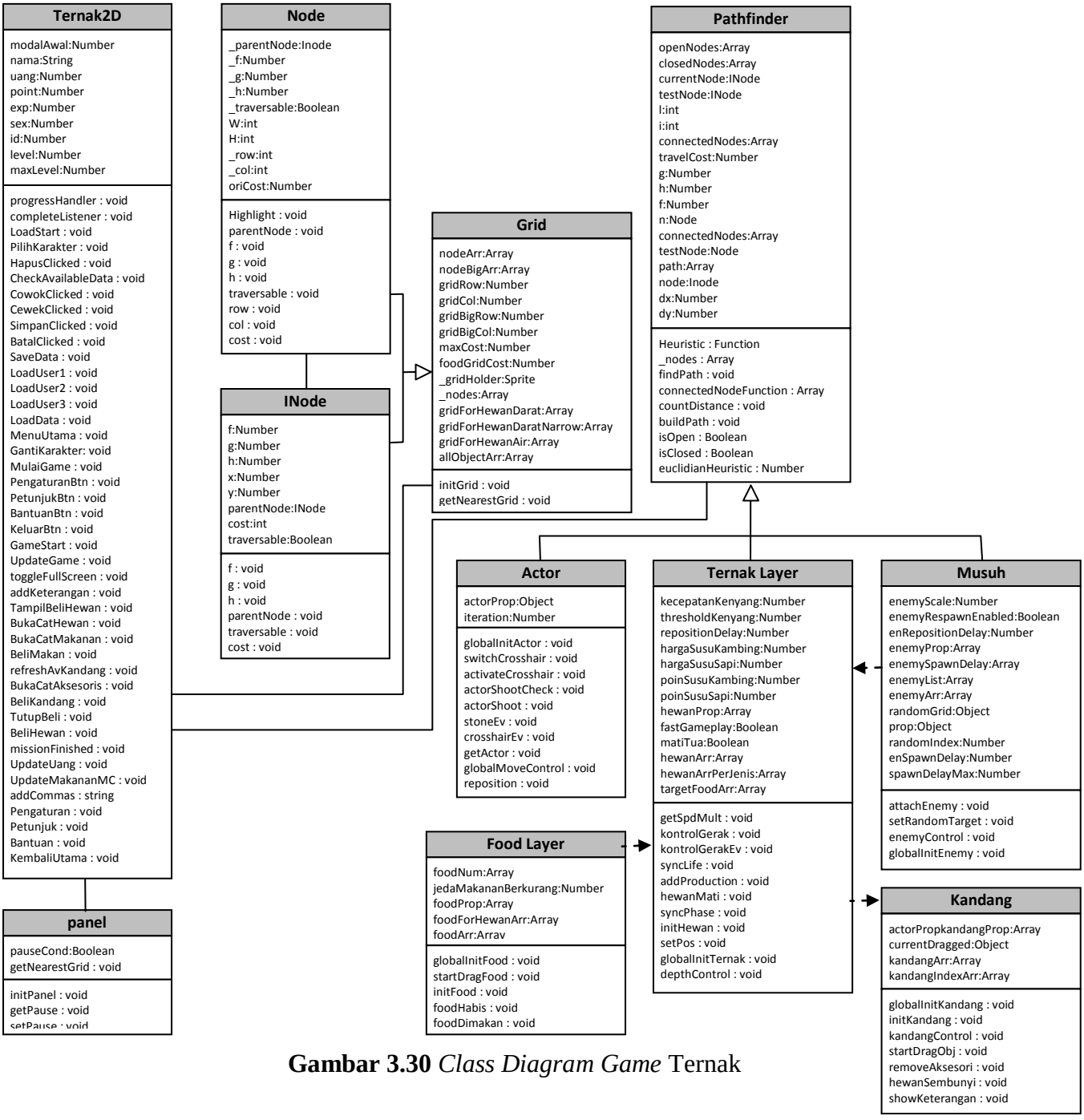
Deskripsi	Tampilan ini merupakan aktifitas permainan
Normal Flow	1. *Player* memilih Toko 2. *Player* memilih Ternak/Makanan yang dibeli 3. *Player* memilih ternak dan menjualnya 4. *Player* keluar



Gambar 3.29 *Activity Diagram* Pada Proses Mulai Permainan

3.3.4 *Class Diagram*

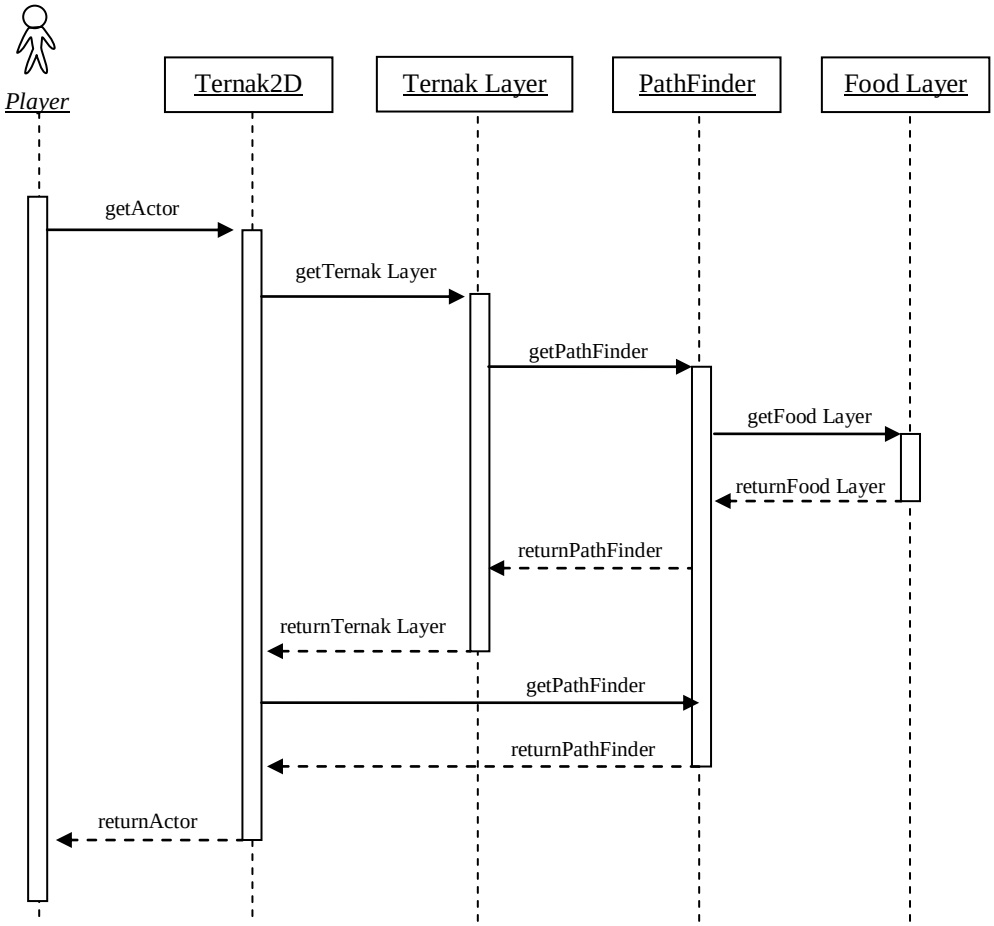
Class diagram adalah suatu diagram yang memperlihatkan struktur dari sebuah sistem. Sistem tersebut akan menampilkan sistem kelas, atribut dan hubungan antar kelas. *Class* diagram *game* dapat dilihat pada gambar 3.24.



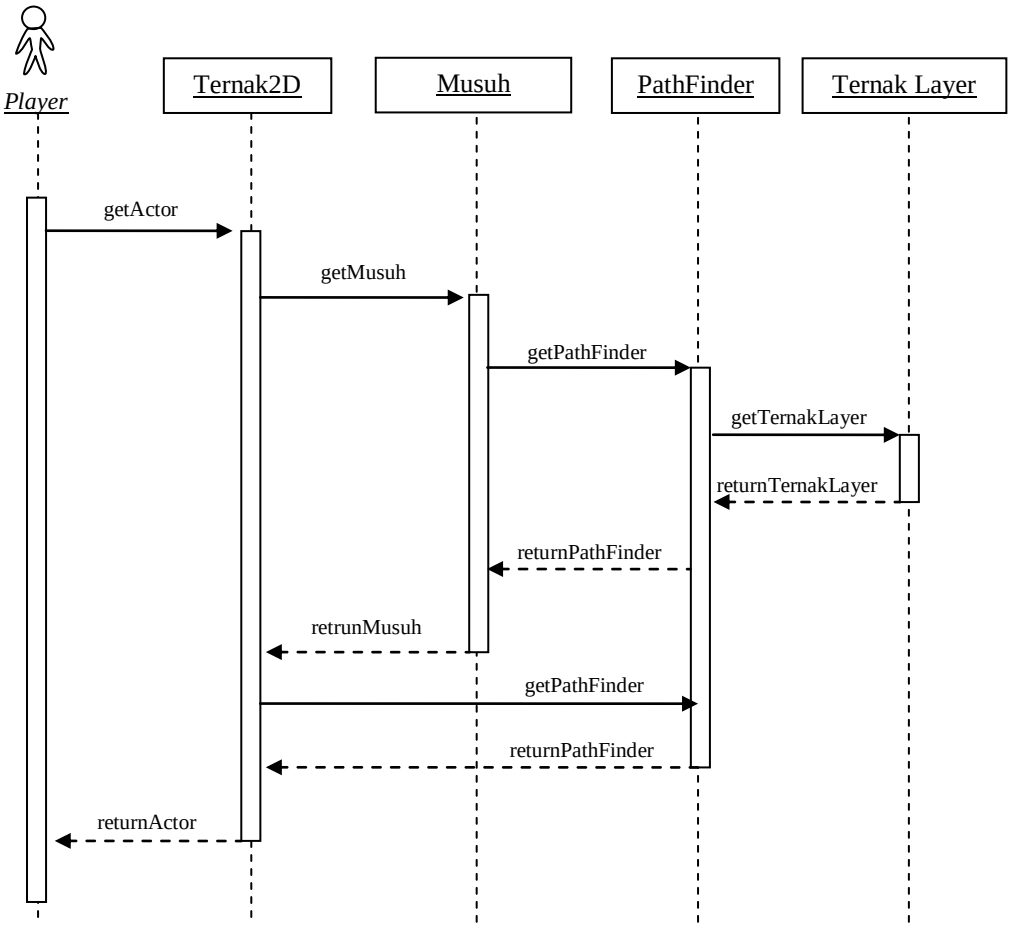
Gambar 3.30 *Class Diagram Game* Ternak

3.3.5 Sequence Diagram

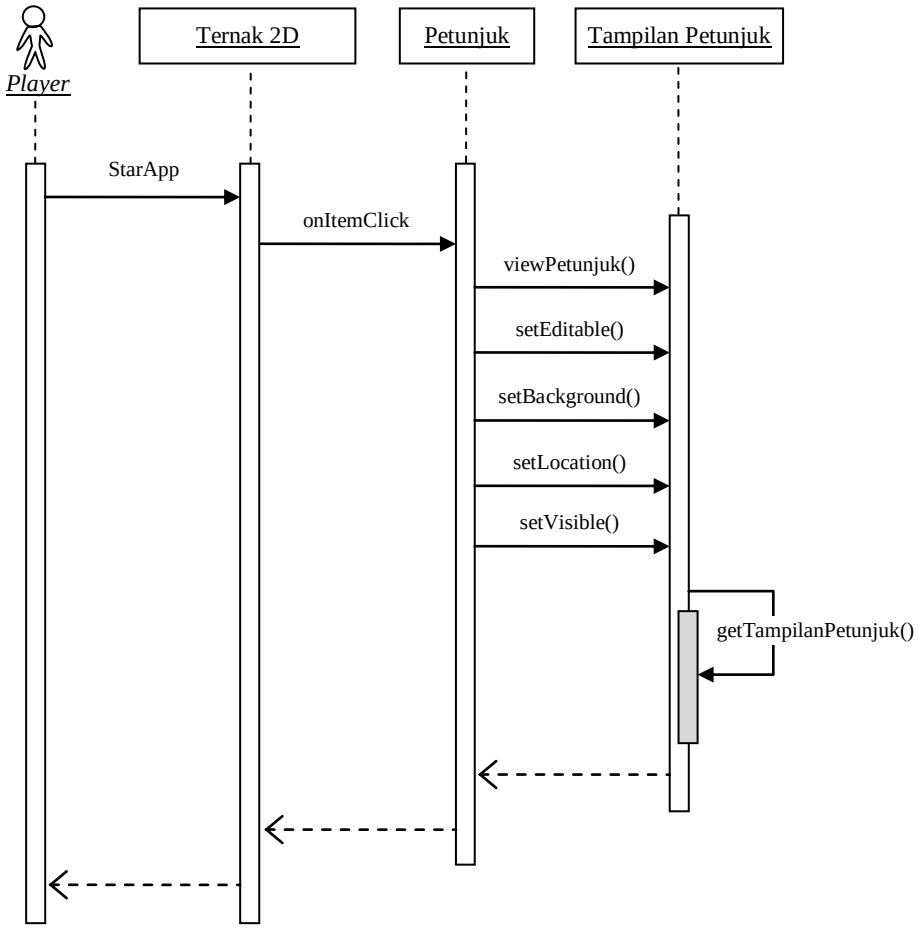
Sequence diagram menjelaskan secara detail urutan proses yang dilakukan dalam sistem untuk mencapai tujuan dari use case: interaksi yang terjadi antar *class*, operasi apa saja yang terlibat, urutan antar operasi, dan informasi yang diperlukan oleh masing-masing operasi.



Gambar 3.31 *Sequence Diagram Game* Ternak Pemberian makan



Gambar 3.32 *Sequence Diagram Game* Ternak Musuh memangsa Ternak



Gambar 3.33 *Sequence Diagram* Petunjuk

3.4 Perancangan

3.4.1 Perancangan Game Ternak

Tabel 3.17 Perancangan Management waktu perkembangan hewan beserta poin

Nama	Gambar	Status	Jeda Waktu	Waktu Lapar	Waktu Berkembang	Waktu Optimal	Reproduksi	Harga Beli	Poin	Harga Jual	Poin
Ikan		Telur	-	-	60 detik			-	-	-	0
		Kecil	40 detik	10 detik	9 x makan	450 dtk		80 (800/10)	30	80	10
		Sedang	45 detik	15 detik	8 x makan	480 dtk		-	-	10.000	20
		Besar	45 detik	15 detik	~	~	1,5 menit 5 telur	-	-	18.000	30
Ayam		Telur	-	-	2 menit			-	-	1.500	20
		Kecil	50 detik	15 detik	8 x makan	504 dtk		3.600	40	3.600	20
		Sedang	50 detik	15 detik	10 x makan	630 dtk		-	-	24.000	30
		Besar	60 detik	20 detik	~	~	1,5 menit 2 telur	-	-	38.000	80
Itik		Telur	-	-	3 menit			-	-	1.500	20
		Kecil	50 detik	20 detik	9 x makan	603 dtk		4.500	50	4.500	30
		Sedang	60 detik	20 detik	8 x makan	624 dtk		-	-	30.000	80
		Besar	75 detik	20 detik	~	~	2 menit 2 telur	-	-	42.000	100
Kambing		Kecil	90 detik	25 detik	8 x makan	912 dtk		375.000	160	350.000	150
		Sedang	90 detik	25 detik	10 x makan	1146 dtk		-	-	800.000	180
		Besar	120 detik	30 detik	~	~	3 menit 1-2 anak 2 menit 1 ltr susu	-	-	1.200.000	200
Sapi		Kecil	90 detik	25 detik	10 x makan	1146 dtk		2.500.000	280	2.000.000	250
		Sedang	120 detik	30 detik	8 x makan	1200 dtk		-	-	5.000.000	300
		Besar	120 detik	30 detik	~	~	4 menit 1 anak 3 menit 2-4 ltr susu	-	-	8.500.000	450

Tabel 3.18 Perancangan Kekuatan & Score pada Musuh

Nama	Gambar	Ternak yg dituju	Kekuatan	Maximal Serang	Jeda Waktu	Uang	Poin
Kucing	-	Ikan	5 x lemparan	3 ternak	60 detik	10.000	80
Musang	-	Ayam & itik	8 x lemparan	5 ternak	90 detik	40.000	100
Pencuri	-	Semua Ternak	10 x lemparan	10 ternak	60 detik	200.000	260

Tabel 3.19 Perancangan Management waktu kecepatan berjalan

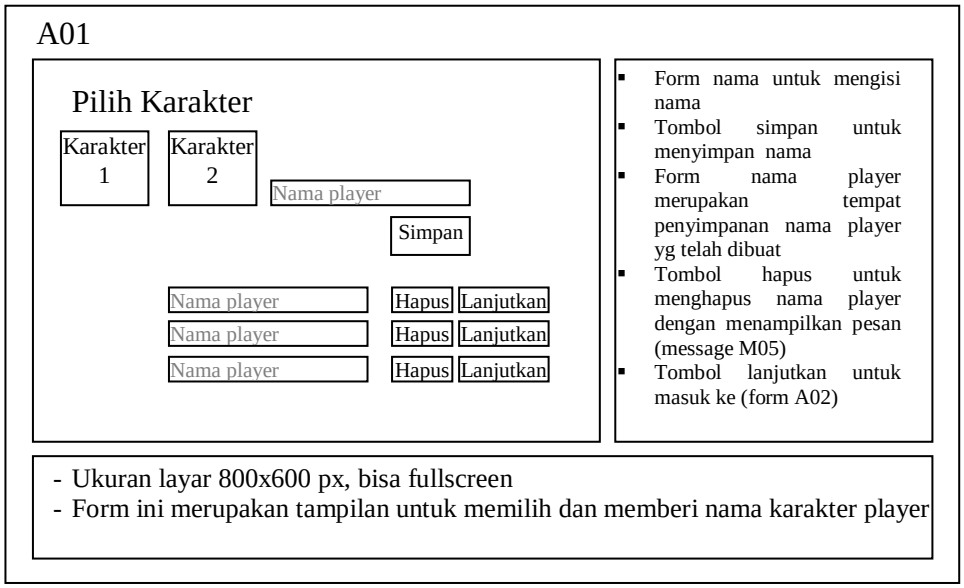
Nama	Gambar	Status	Kecepatan	Waktu Berjalan	Waktu Jeda (Beristirahat)	Waktu Dikejar Musuh
Ikan		Kecil	2	5 detik	3 detik	Kecepatan + 1
		Sedang	2	6 detik	4 detik	Kecepatan + 1
		Besar	3	7 detik	4 detik	Kecepatan + 1
Ayam		Kecil	3	8 detik	5 detik	Kecepatan + 1
		Sedang	3	10 detik	6 detik	Kecepatan + 2
		Besar	4	10 detik	6 detik	Kecepatan + 2
Itik		Kecil	3	10 detik	6 detik	Kecepatan + 1
		Sedang	4	11 detik	6 detik	Kecepatan + 2
		Besar	4	11 detik	6 detik	Kecepatan + 2
Kambing		Kecil	4	11 detik	6 detik	Kecepatan + 2
		Sedang	5	12 detik	7 detik	Kecepatan + 3
		Besar	5	12 detik	7 detik	Kecepatan + 3
Sapi		Kecil	4	12 detik	7 detik	Kecepatan + 2
		Sedang	4	12 detik	8 detik	Kecepatan + 3
		Besar	4	12 detik	8 detik	Kecepatan + 3
Kucing	-	Pemangsa ikan	3	6 detik	2 detik	Kecepatan + 1
Musang	-	Pemangsa unggas	5	8 detik	3 detik	Kecepatan + 2
Pencuri	-	Musuh dari semuanya	4	8 detik	3 detik	Kecepatan + 3

Tabel 3.20 Perancangan Tingkatan Level

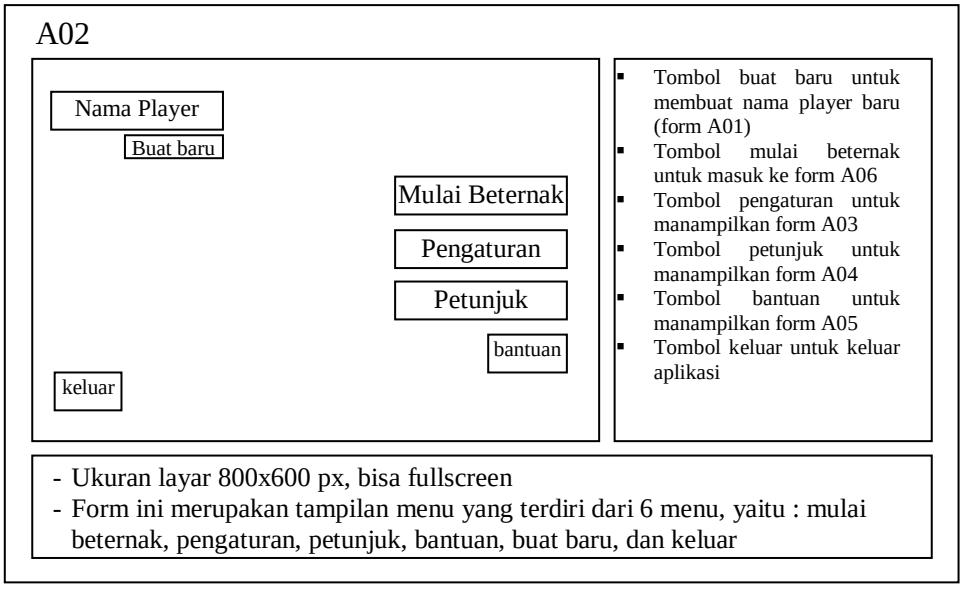
Level	Keterangan				
	Poin yg harus ditempuh	Menu pembelian			Musuh
		Hewan	Makanan	Aksesoris	
1 Modal Rp. 100.000	500	▪ Ikan ▪ Ayam	▪ Ikan (paket10) ▪ Unggas (paket10)		1
2	1.000	▪ Ikan ▪ Ayam ▪ Itik	▪ Ikan (paket10,30) ▪ Unggas (paket10,30)	▪ Kandang Unggas ▪ Tempat makan Ayam	1-2
3	2.500	▪ Ikan ▪ Ayam ▪ Itik ▪ Kambing	▪ Ikan (paket10,30) ▪ Unggas (paket10,30)	▪ Kandang Unggas ▪ Tempat makan Ayam ▪ Kandang Kambing & Sapi	1-2
4	5.000	▪ Ikan ▪ Ayam ▪ Itik ▪ Kambing ▪ Sapi	▪ Ikan (paket10,30) ▪ Unggas (paket10,30)	▪ Kandang Unggas ▪ Tempat makan Ayam ▪ Kandang Kambing & Sapi	2

3.4.2 Perancangan Antar Muka

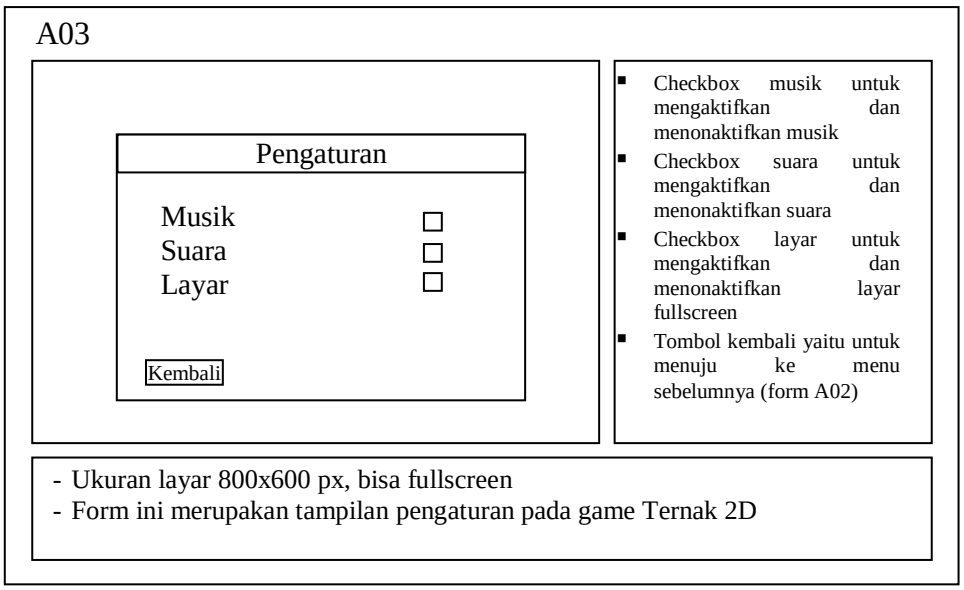
Perancangan antarmuka dibutuhkan untuk mewakili keadaan sebenarnya dari aplikasi yang akan dibangun, berikut akan disajikan perancangan antarmuka dari aplikasi yang akan dibangun:



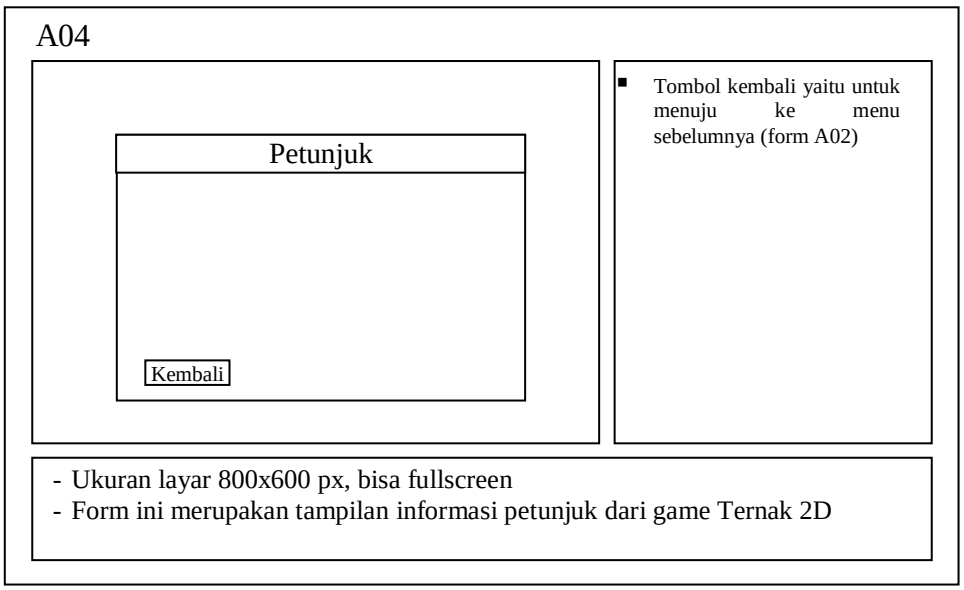
Gambar 3.34 Antar Muka Pilih karakter A01



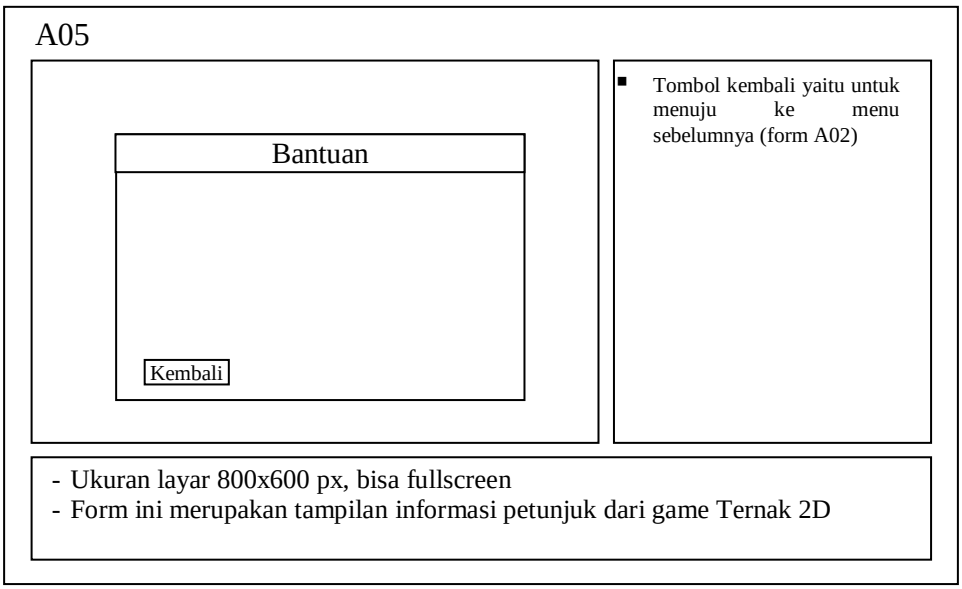
Gambar 3.35 Antar Muka Tampilan Menu A02



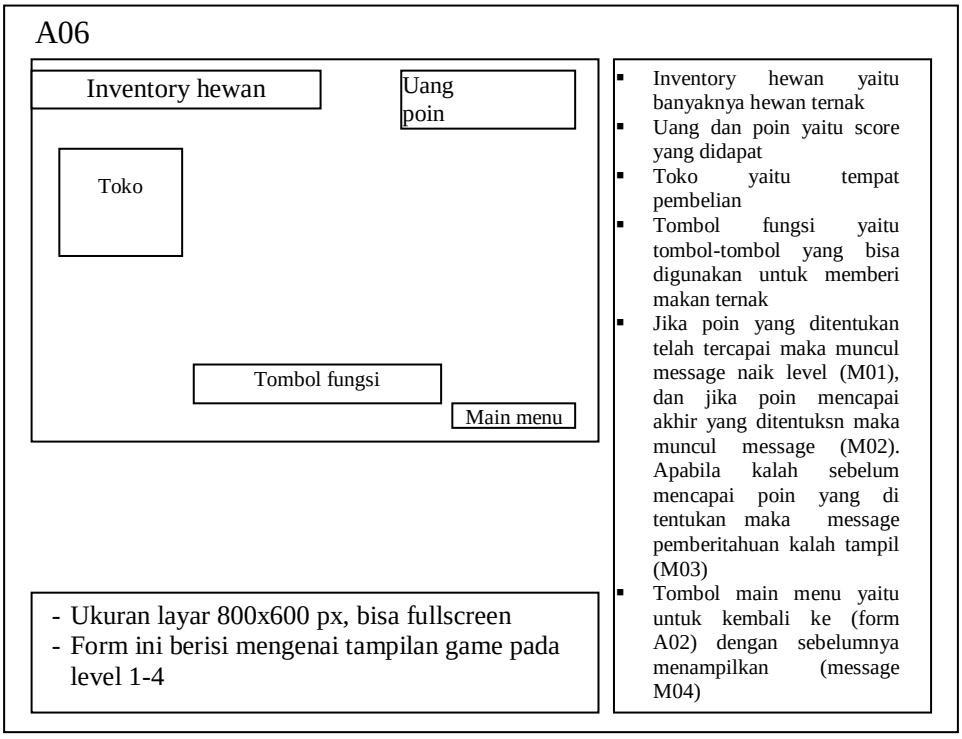
Gambar 3.36 Antar Muka Pengaturan *Game* A03



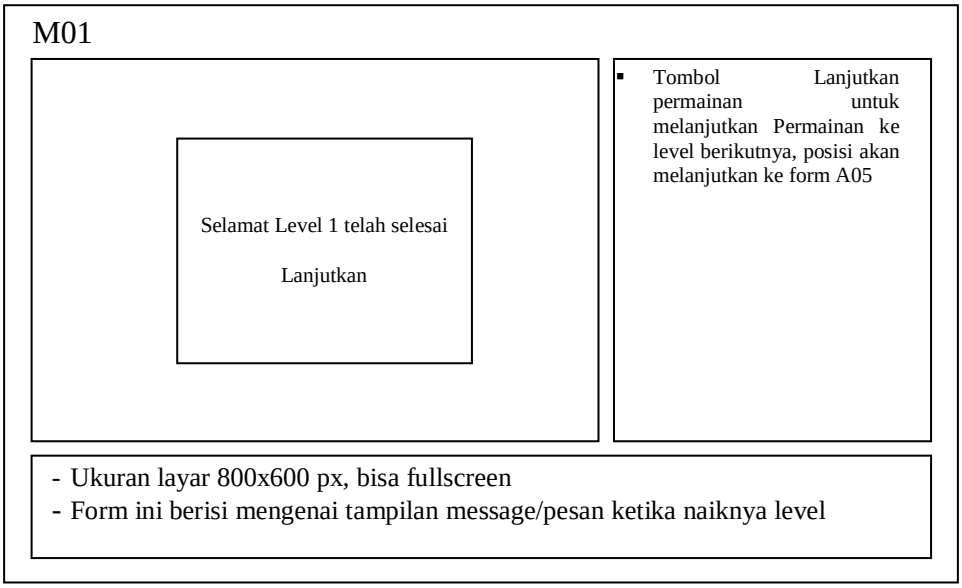
Gambar 3.37 Antar Muka Petunjuk *Game* A04



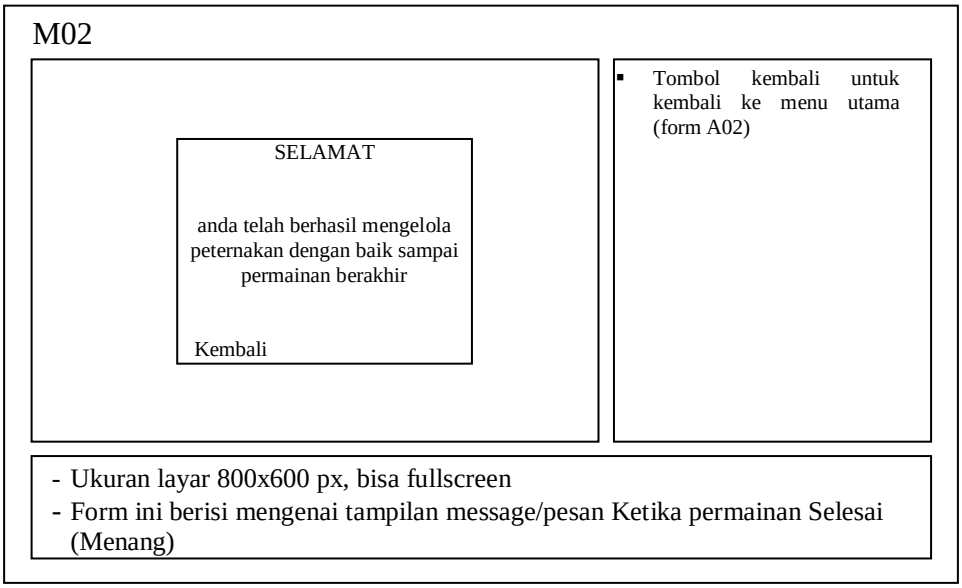
Gambar 3.38 Antar Muka Bantuan *Game* A05



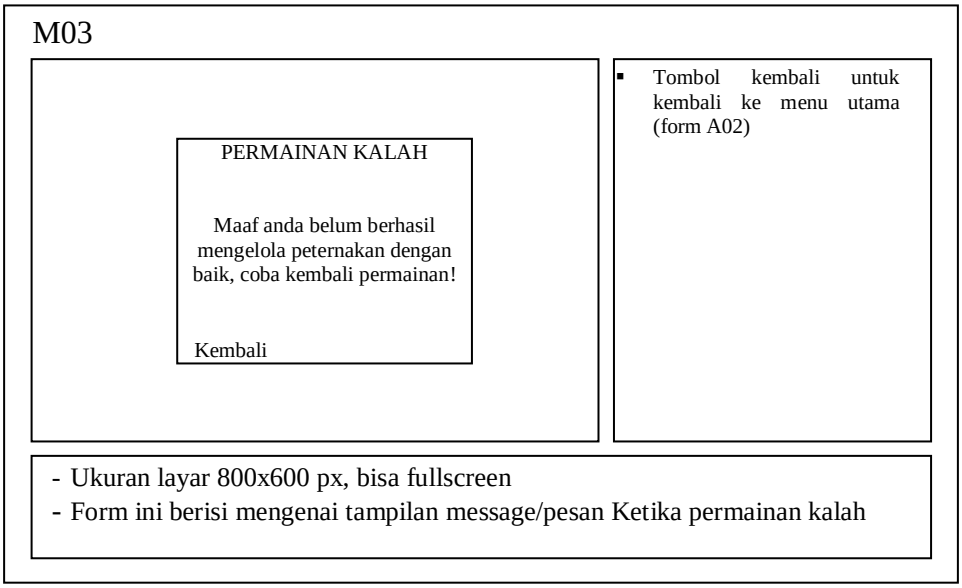
Gambar 3.39 Antar Muka *Game* Ternak level 1-4 A06



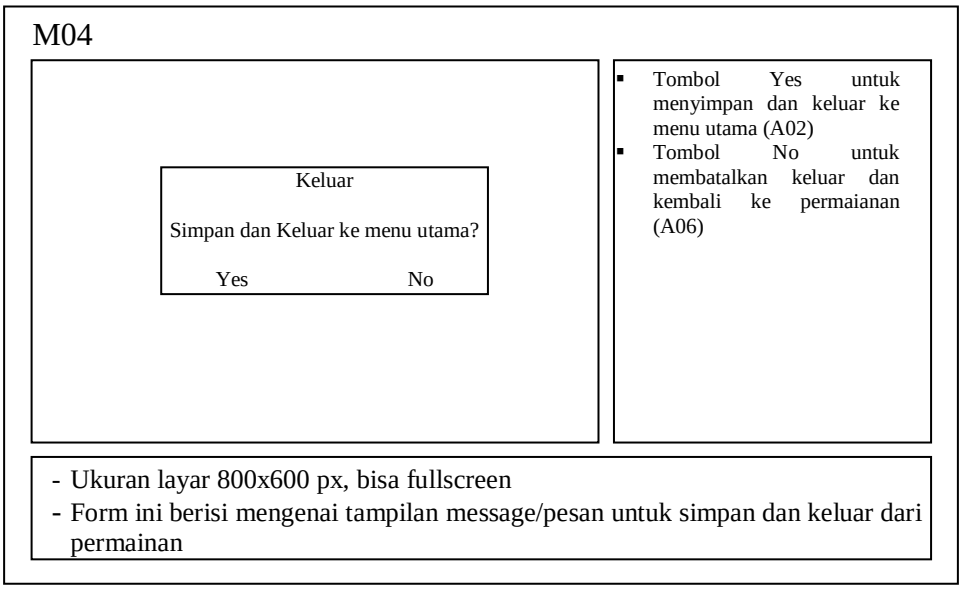
Gambar 3.40 Antar Muka Tampilan Pesan Naik Level M01



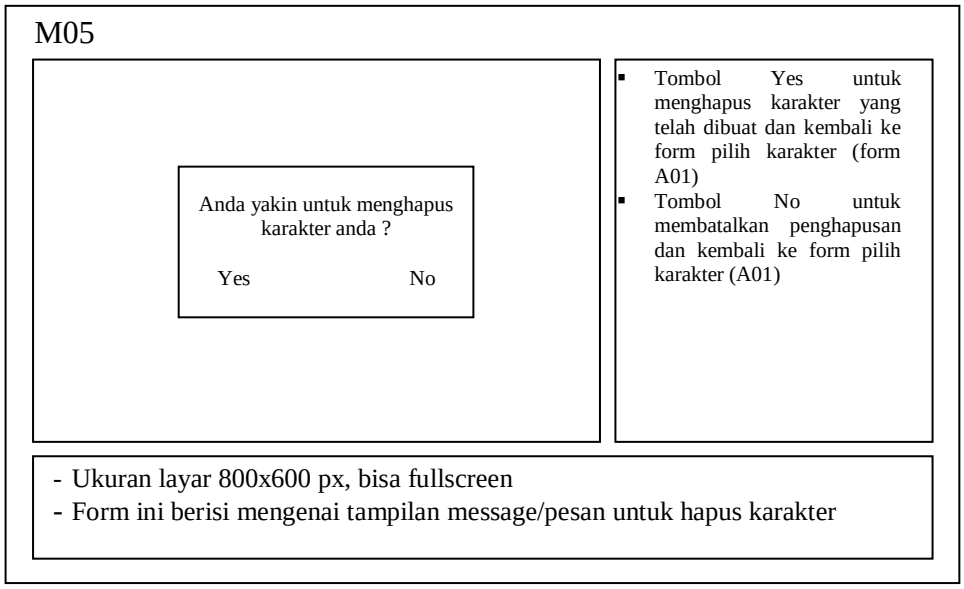
Gambar 3.41 Antar Muka Tampilan Pesan Permainan Selesai (Menang) M02



Gambar 3.42 Antar Muka Tampilan Pesan Permainan Kalah M03



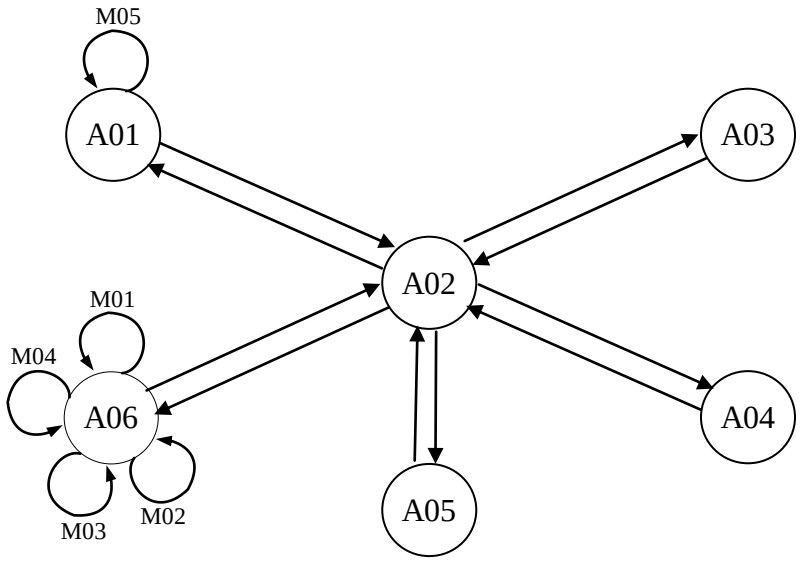
Gambar 3.43 Antar Muka Tampilan Pesan Keluar dari Permainan M04



Gambar 3.44 Antar Muka Tampilan Pesan Hapus Karakter M05

Jaringan Semantik

Jaringan semantik *Game* Ternak adalah sebagai berikut :

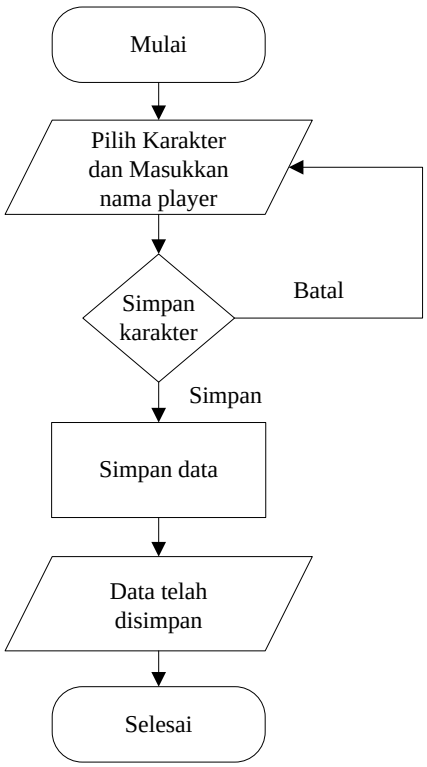


Gambar 3.45 Jaringan Semantik Ternak

3.4.3 Perancangan Prosedural

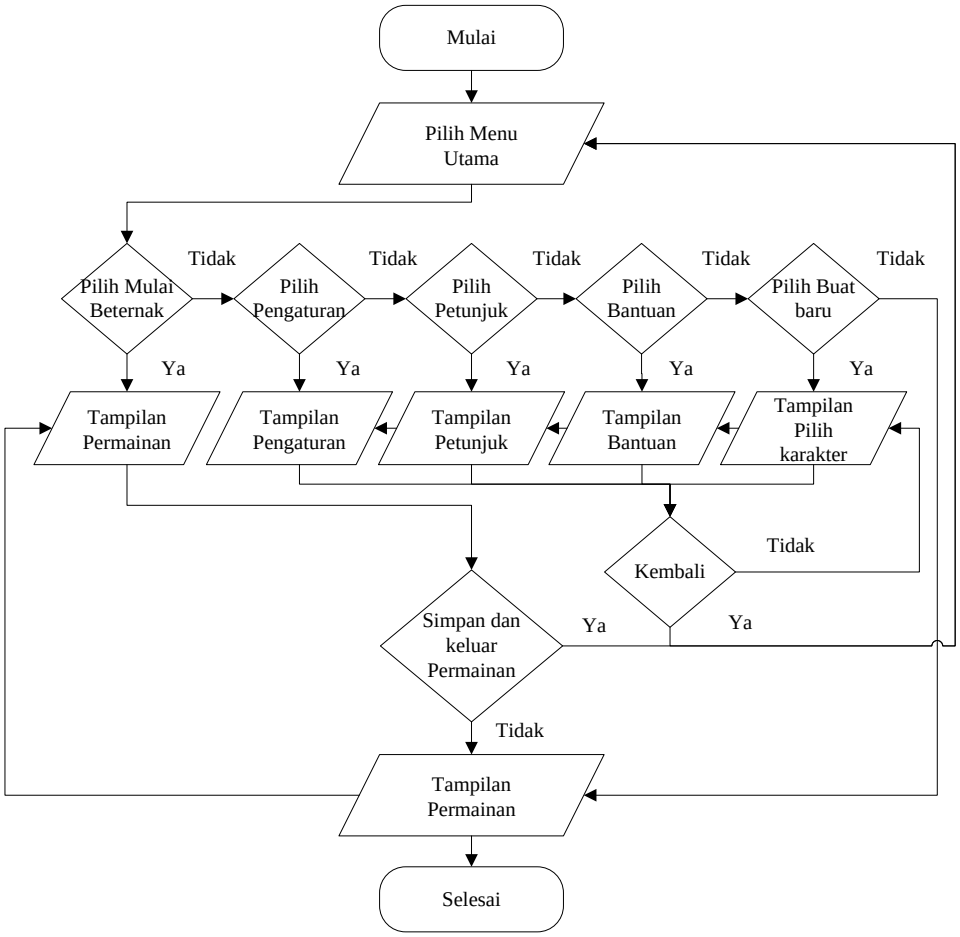
Perancangan prosedural merupakan perancangan yang dilakukan untuk menetapkan detail program algoritma yang akan dinyatakan kedalam suatu program. Adapun perancangan prosedural untuk aplikasi yang akan dibangun adalah :

1. Prosedur Pilih Karakter, merupakan prosedural yang terjadi ketika *User* akan mulai bermain *game*.



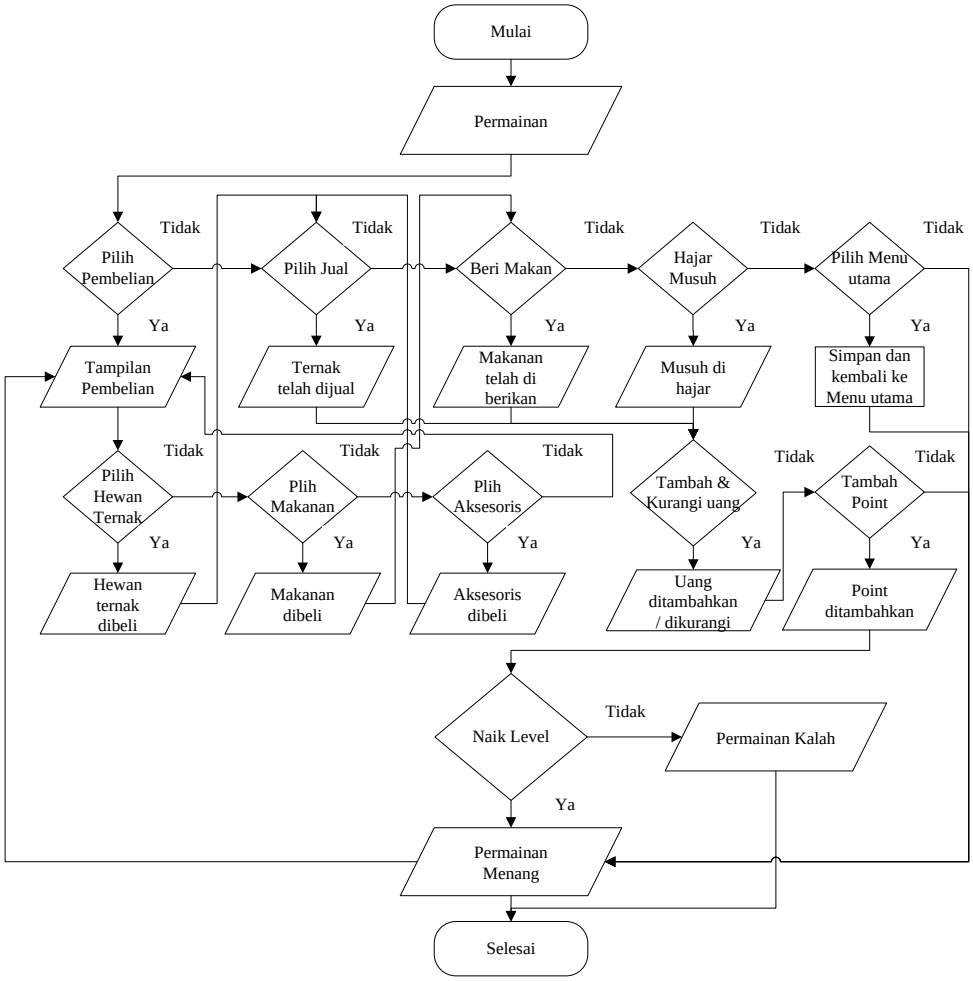
Gambar 3.46 Prosedural Pilih karakter Ternak

2. Prosedur pilihan Menu Utama Sesudah Pilih karakter, merupakan prosedur yang dilakukan oleh *player* untuk melanjutkan ke langkah permainan.



Gambar 3.47 Prosedural Menu Utama Ternak

3. Prosedur Permainan Ternak, merupakan prosedur yang dilakukan oleh *player* untuk menjalankan permainan.



Gambar 3.48 Prosedural *Game* Ternak