BAB III
ANALISIS DAN PERANCANGAN

III.1 Analisis Sistem

Pada bagian analisis sistem ini akan dibahas mengenai analisa-analisa masalah yang akan dilakukan terhadap pasar *game* khususnya *game* di Indonesia, analisa pengguna (user) dan juga kegunaan *game* yang akan dibangun. Berikut ini adalah analisa sistem yang ada pada *game* ini :

1. Sistem single player.
2. Mengangkat tema belajar berpuasa.
3. Grafik *game* 2D yang lebih berwarna.
4. Media interaksi menggunakan keyboard.
5. Teknik Interaksi Side Scrolling.
6. Aplikasi *game* bersifat edukatif.

III.1.1 Analisis Masalah

Berdasarkan pengamatan yang telah dilakukan, dapat diketahui bahwa jarang sekali ada aplikasi *game* pada suatu perangkat komputer yang memiliki nilai edukasi tentang berpuasa yang menerangkan tentang makna puasa dan memberikan nilai edukasi tentang kesabaran dalam menjalankan puasa itu sendiri. Biasanya kebanyakan dari *game* edukasi yang ada hanya bersifat normatif. Disamping itu banyak anak-anak yang belum paham dam mengerti tentang keharusan untuk memulai belajar berpuasa.

Oleh karena itu untuk mengatasi masalah tersebut akan dibangun sebuah aplikasi *Game* yang diharapkan dapat menjadi suatu media yang bisa menggambarkan atau menceritakan tentang pentingnya pemahaman anak untuk memulai belajar berpuasa, dan bisa memberikan edukasi untuk kepada anak tentang apa yang sebenarnya harus dilakukan pada saat berpuasa agar puasa yang dilakukan sesuai dengan syariat Islam.

III.1.2 Analisis *Game* Sejenis

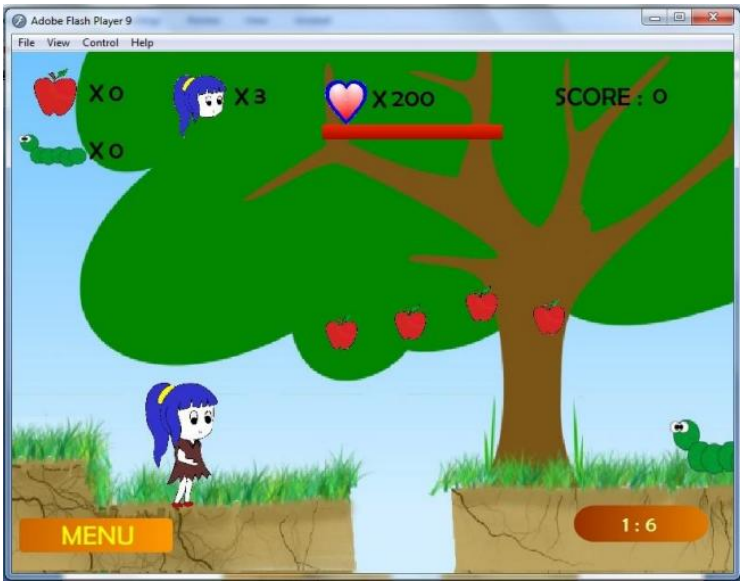
Analisis *game* sejenis dilakukan untuk memberikan gambaran sistem agar lebih mudah dalam pembangunan dan sebagai pembanding untuk kesempurnaan *game* yang akan dicapai.

Game Catch The Apple menceritakan tentang seorang gadis yang mempunyai sebuah perkebunan apel. Pada saat panen apel tiba ia memanen apel-apel tersebut dan mematikan hama-hama yang mengincar apel yang sudah matang.



Gambar III. 1 *Game* Catch The Apple [15].

Dalam *game* ini *player* berperan sebagai Seorang gadis yang mempunyai kebun apel, dan kebunnya tersebut sudah siap dipanen. *Player* diharuskan melewati mengumpulkan apel untuk mendapatkan poin. Dapat dilihat di gambar 3.2 dan analisis yang terdapat *game Catch The Apple* dapat dilihat pada Tabel 3.1.



Gambar III.2 Tampilan game *Catch The Apple*

Tabel III.1 Analisis *game* sejenis

Nama *Game*	***Genre Game***	**Desain Grafis**	***Storyline* dan *gameplay***	***Multi player***	**Basis**
Catch The Apple	*Side Scrolling*	2D kartun dan penuh warna.	Gadis yang mempunyai sebuah perkebunan apel yang mematikan hama saat hama tersebut mengganggunya.	Tidak.	*Desktop*

a. Gameplay

Pada *game Catch The Apple* , pemain harus mengumpulkan beberapa buah apel yang mengantung di pohon-pohon dan membunuh hama yang mengganggu panen apel tersebut. Pemain diharuskan mencapai finish dalam waktu 1 menit 10 detik untuk masuk ke level berikutnya.

b. **Skenario *Game***

Aturan permainannya yaitu:[15]

1. Karakter yang digunakan dalam *game* ini adalah seorang gadis.
2. Protokol komunikasi yang digunakan adalah keyboard.
3. *Game* terdiri dari 2 level.
4. Dalam setiap level, pemain harus mengambil apel yang tersebar dilayar sebanyak-banyaknya di setiap levelnya. Waktunya 1 menit 10 detik pada setiap levelnya. Jadi jika pemain tidak bisa menyelesaikan permainan sampai dengan waktu yang ditentukan maka permainan akan berakhir/*game* over.
5. Bila permainan berakhir sampai dengan level 2, pemain akan melihat skor terakhirnya, dan bisa melakukan permainan dari awal lagi

III.1.3 Analisis *Game* Yang Akan Dikembangkan

Analisis *Game* yang akan dikembangkan adalah meliputi tentang Story Line dan *Game* Play *game* Ayo Belajar Berpuasa.

III.1.3. 1 Story Line

Skenario : Stage 1 (Kuliah Subuh)

Intro :

Naofal merupakan seorang anak yang sedang belajar perpuasa untuk pertama kalinya. Naofal bangun untuk makan sahur dengan gembira, pada saat sahur Naofal dinasehati oleh orang tuanya tentang makna puasa itu sendiri. Naofal mengerti tentang apa yang dijelaskan oleh orang tuanya, dan bertekad bahwa dia harus menamatkan puasa hari ini dan menahan kesabaran agar disayangi oleh Allah. Naofal berjanji akan menjadi anak baik dengan mematuhi segala perintah dan menjauhi semua larangan Allah. Tantangan pertama Naofal adalah saat dia akan pergi kuliah subuh di mesjid dekat rumahnya.

Setting : Rumah, Masjid dan Lingkungan dekat Rumah Naofal

Misi : Naofal menghindari makanan yang sangat menggoda dan menghindari para Lita (sekelompok balita yang sedang makan).

Rintangan : Makanan dan para Lita (sekelompok balita yang sedang makan).
Apa yang harus dilakukan : Menghindari makanan dan para Lita untuk mempertahankan poin, dan mengambil ikon hati untuk menambah Health Point.
Outro :
Naofal berhasil menghindari makanan dan Para Lita. Naofal berhasil sampai ke Masjid untuk Kuliah subuh. Sampai di Masjid Naofal mendengarkan ceramah yang disampaikan oleh Pak Ustad tentang puasa, dan Naofal bahagia karena berhasil menahan diri saat ada makanan enak yang menggodanya.

Skenario : Stage 2 (Berangkat Sekolah)
Intro :
Naofal berada di rumah dan sedang bersiap-siap untuk berangkat sekolah, dan Naofal ingin melanjutkan misinya yaitu berpuasa dan bersabar agar di sayang Allah. Saat akan berangkat ke sekolah, Naofal kembali digoda oleh berbagai macam makanan yang kelihatan enak dan Para Lita juga semakin membuat Naofal ingin mencicipi makanan tersebut. Saat hampir sampai di Sekolah, Naofal diganggu oleh Anak Nakal bernama Ibro yang suka menyerangnya. .
Setting : Rumah, Sepanjang jalan menuju ke Sekolah
Misi : Naofal menghindari makanan yang sangat menggoda, menghindari para Lita (sekelompok balita yang sedang makan). dan menahan sabar saat anak nakal bernama Ibro (Bos level 2) menyerangnya dengan tidak balik menyerang, karena dengan menyerang akan mengurangi poin.
Rintangan : Makanan dan para Lita (sekelompok balita yang sedang makan).dan Ibro (Boss level 2)
Apa yang harus dilakukan : Menghindari makanan, para Lita dan Ibro (Bos Level 2)untuk mempertahankan poin serta tidak balik menyerangnya karena apabila Naofal memukulny a maka akan mengurangi poin, dan mengambil ikon hati untuk menambah Health Point.

Outro :

Naofal berhasil menahan sabar saat anak nakal bernama Ibro (Bos level 2) menyerangnya dan menghindari makanan dan Para Lita. Naofal berhasil sampai ke Sekolah.

Skenario : Stage 3 (Rumah)

Intro :

Naofal pulang sekolah dan sampai di rumah, di rumah ternyata Bunda sedang memasak makanan yang sangat disukai Naofal, disini terjadi perjuangan batin Naofal dan kesabaran pun diuji agar Naofal tidak sampai tergoda untuk mencicipi makanan yang sangat disukainya sampai Adzan Maghrib berkumandang

Setting : Rumah Naofal

Misi : menghindari makanan kesukaan dan Evil Naofal (Boss di level 3)

Rintangan : Makanan yang sangat Naofal dan Evil Naofal.

Apa yang harus dilakukan : menghindari makanan kesukaan yang sudah di pasak Bunda dan menghindari Evil Naofal sampai saat Adzan Magrib tiba.

Outro :

Naofal berhasil menghindari makanan dan Evil Naofal (Refleksi dirinya sendiri yang sedang memakan makanan kesukaannya). Akhirnya Adzan Magrib berkumandang. Naofal dan Orang Tuanya makan dengan lahap, Naofal diberi pujian oleh orang tuanya karena telah menamatkan Puasa dan menahan sabar hari ini.

Skenario : *Game* Over

“Jangan Menyerah, Ayo Bantu Naofal untuk Belajar Berpuasa”

III.1.3.2 *Game* Play

Tugas utama pemain (*game*r) dalam memainkan Aplikasi ini yaitu melalui stage 1 sampai stage 3, dimana setiap bagian stage menyediakan petualangan dengan menampilkan beberapa karakter stage tersebut maupun karakter musuh yang harus dihindari (sebagai indikator kesabaran), sehingga pemain memperoleh score (nilai).

Pemain (*game*r) dapat memulai permainan dengan memilih pilihan mulai baru, dimana nantinya pemain memasuki stage pertama dari permainan. Sedangkan pilihan lainnya yaitu pilihan melanjukan, pilihan ini akan menanpilkan stage terakhir di mana pemain tersebut bermain. Apabila pemain belum bermain sedikitpun tetapi memilih "melanjutkan", maka aplikasi akan menampilkan stage pertama.

Setelah pemain telah memilih salah satu stage, maka pemain langsung dapat memulai petualangannya. Selain pilihan "Mulai baru" dan "Melanjutkan" pada tampilan awal juga disediakan pilihan "High score" yaitu pilihan untuk melihat score-score yang sudah tercetak sebagai hasil dari permainan yang telah dilakukan oleh pemain dan pilihan lainnya adalah pilihan "Pembuat", sebagai keterangan yang menyediakan informasi mengenai pembuat *game* ini.

III.1.3.3 Analisis Tingkat Kesulitan

Tingkat kesulitan dari tiap *level* akan dibedakan dari jumlah musuh dan *delay* tembakan musuh. Perbandingan tingkat kesulitan dari tiap *level* dapat dilihat pada tabel III.2.

Tabel III.2 Tingkat Kesulitan tiap *Level*

No	***Level* 1**	***Level* 2**	***Level* 3**
Jumlah Musuh	10	15	20
Kecepatan Pergerakan Bos	-	2	5

Berdasarkan table diatas dapat diketahui bahwa tingkat kesulitan dalam *game* "Ayo Belajar Puasa" ini, terdiri dari 3 level dengan spesifikasi sebagai berikut:

1. Level Pertama

Jumlah musuh dalam level ini adalah 10 buah yang terdiri dari para Lita dan berbagai macam makanan kesukaan Naofal (Pemain). Naofal harus menghindari makanan dan para lita tersebut, pada awal permainan Naofal akan diberikan skor 100 dan skor akan terus berkurang bila Naofal bersentuhan langsung dengan musuh.

2. Level Kedua

Jumlah musuh dalam level ini adalah 15 buah yang terdiri dari para Lita, berbagai macam makanan kesukaan Naofal (Pemain) dan Ibro (Boss Level 2). Naofal harus menghindari makanan dan para lita tersebut, dan menghindari lemparan dari Boss Ibro pada akhir level. Pada awal permainan Naofal akan diberikan skor 100 dan skor akan terus berkurang bila Naofal bersentuhan langsung dengan musuh.

Pada level kedua kecepatan pergerakan boss diatur sebanyak 2. Itu artinya, boss level 2 berpindah posisi sebanyak dua kali posisi sebelumnya. Misalnya, koordinat awal boss adalah (4,4) maka setiap melakukan pergerakan ke kiri (mengurangi nilai X) maka dia akan berpindah sebanyak minus dua (-2). Jadi pada pergerakan awal dari koordinat pertama boss, dia akan bergerak ke koordinat (4,2) untuk langkah pertama. Namun demikian, karena pengulangan berlangsung sangat cepat, maka pergerakan tiap koordinatnya tidak akan terlalu terlihat, dan yang paling jelas terlihat adalah pergerakan boss dua yang semakin cepat jika variabel speed tersebut diperbesar.

Implementasi dari perbedaan kecepatan musuh di kode program diterapkan pada variabel *speed,* yang diimplentasikan pada kelas ibro.

3. Level Ketiga

Jumlah musuh dalam level ini adalah 20 buah yang terdiri dari para Lita, berbagai macam makanan kesukaan Naofal (Pemain) dan Evil Naofal (Boss Level 3). Naofal harus menghindari makanan dan para lita tersebut, dan menghindari lemparan makanan kesukaan dari Evil Naoval pada akhir level. Pada awal

permainan Naofal akan diberikan skor 100 dan skor akan terus berkurang bila Naofal bersentuhan langsung dengan musuh.

Pada level ketiga, kecepatan (*speed*) pergerakan boss diatur sebanyak 5. Hal tersebut memiliki arti bahwa boss ketiga akan berpindah sebanyak 5 kali dari posisi sebelumnya. Perpindahan boss tiga bisa memiliki berbagai macam bentuk, diantarnya yaitu horizontal (mengurangi/menambah nilai X), vertikal (mengurangi/menambah nilai Y), serta diagonal (mengurangi/menambah nilai X dan Y).

Untuk level 3 implementasi dari perbedaan kecepatan musuh di kode program diterapkan pada variabel *speed*, yang diimplentasikan pada kelas naofal_evil.

III.1.3.4 Analisis Scoring

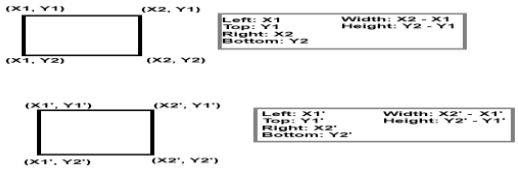
Game "Ayo Belajar Puasa" ini tidak memiliki score, satu-satunya penilaian yang ada adalah sukses atau tidaknya misi dalam setiap level.

III.1.3.5 Analisis Pengunaan Collision Detection

Dalam pembuatan *game* "Ayo Belajar Puasa" ini diperlukan adanya penggunaan *collision detection*. Penggunaan diantaranya untuk memicu event pada pergerakan aktor dengan makanan atau aktor dengan musuh.

Teknik yang digunakan disini adalah *Rectangles Collision Detection*. Teknik ini merupakan pendeteksian tabrakan menggunakan bound berbentuk persegi, dimana setiap sprite atau objek yang dibuat dalam *game* memiliki bound masing-masing. Cara ini cukup mudah karena hanya perlu membandingkan apakah ada *intersection* (irisan) pada kedua objek pada kordinat tertentu.

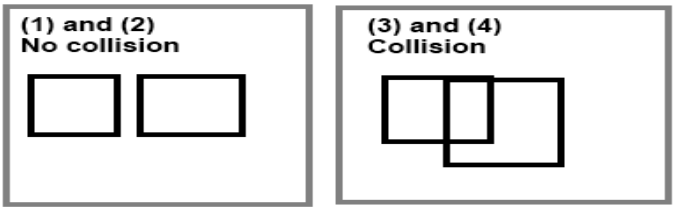
Cara mengetahui bagaimana tabrakan terjadi, pertama adalah menentukan titik koordinat dari masing-masing objek. Contoh, ada dua buah persegi panjang dengan nama R1 dan R2. R1 mempunyai koordinat: (X1,Y1), (X1,Y2), (X2,Y1), (X2,Y2). Sedangkan koordinat R2: (X1', Y1'), (X1', Y2'), (X2',Y1'), (X2',Y2'). Untuk penjelasan lebih jelasnya dapat dilihat dalam gambar di bawah ini.



Gambar III.2.1 Koordinat Persegi Panjang

Pertama, memakai skenario *Horizontal Collision*. Seperti di bawah ini:

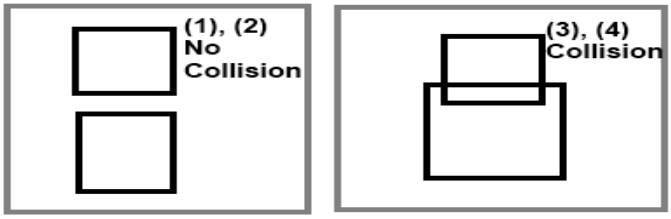
1. Melakukan pemeriksaan apakah tepi kiri R2 ada diantara tepi kiri dan kanan R1. Artinya, jika X1 <X1'<X2 maka terjadi tabrakan. X1 <X1', tapi X1'> X2 maka tidak ada tabrakan.
2. Melakukan pemeriksaan apakah tepi kanan R2 ada diantara tepi kiri dan kanan R1. Artinya, jika X1 <X2'<X2 maka terjadi tabrakan.
3. (1), tepi ada tabrakan karena X1 <X1'<X2.
4. (2), tepi ada tabrakan karena X1 <X2'<X2.



Gambar III.2.2 Horizontal Collision

Selanjutnya adalah *Vertical Collision*, skenarionya sebagai berikut:

1. Melakukan pemeriksaan tepi atas R2, tabrakan jika Y1> Y1' > Y2. Tidak ada tabrakan dalam hal ini karena Y1' > Y2.
2. Melakukan pemeriksaan tepi bawah R2, tabrakan jika Y1> Y2' > Y2. Tidak ada tabrakan dalam hal ini karena Y2' < Y1.
3. Mirip dengan no. (1), tepi kali ini Y1> Y1' > Y2 sehingga ada tabrakan.
4. Seperti no.(2), tepi Y1> Y2' > Y2 sehingga ada tabrakan.



Gambar III.2.3 Vertical Collision

III.4. Analisis Algoritma Pencarian

Pada *game* Edukasi 2D "Ayo Belajar Puasa", algoritma pencarian diterapkan pada musuh di level 2 dan 3 dengan beberapa *node* pada peta. Struktur data yang digunakan adalah *array* dua dimensi dengan menentukan bobot sesuai koordinat setiap *node*. Pada penentuan bobot setiap *node* akan diberikan nilai sesuai dengan jarak terdekat ke tujuan. *Node* terjauh dari tujuan akan diberikan bobot yang kecil, sedangkan *node* terdekat dengan tujuan akan diberikan bobot yang lebih besar berdasarkan koordinat dari tiap *node*. Penggunaan algoritma pencarian pada *game* ini akan disimulasikan dengan menggunakan matriks dengan ordo 3x3.

Untuk melihat contoh kasusnya dapat dilihat pada gambar:



Gambar III.3 Kasus pencarian jalur terpendek dengan algoritma pencarian

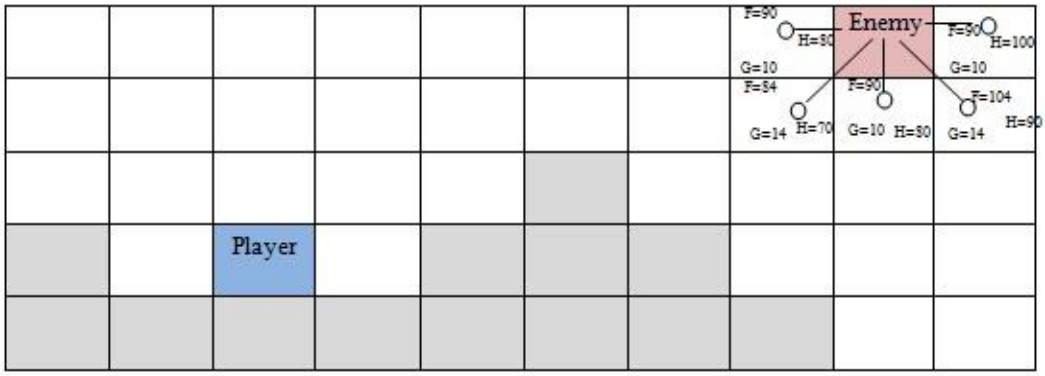
Pada gambar III.1 bisa dilihat bahwa node yang berwarna merah melambangkan enemy sedangkan node yang berwarna biru melambangkan player dan untuk node yang berwarna abu-abu merupakan penghalang pada *game*. Nilai heuristic atau h didapat dari jarak perkiraan dari node biru atau player ke node merah atau enemy.

Untuk lebih jelasnya dapat dilihat pada gambar III.2

H=10	H=20	H=30	H=40	H=50	H=60	H=70	H=80	Enemy	H=100
	H=10	H=20	H=30	H=40	H=50	H=60	H=70	H=80	H=90
		H=10	H=20	H=30	H=40	H=50	H=60	H=70	H=80
		Player	H=10	H=20	H=30	H=40	H=50	H=60	H=70
		H=10	H=20	H=30	H=40	H=50	H=60	H=70	H=80

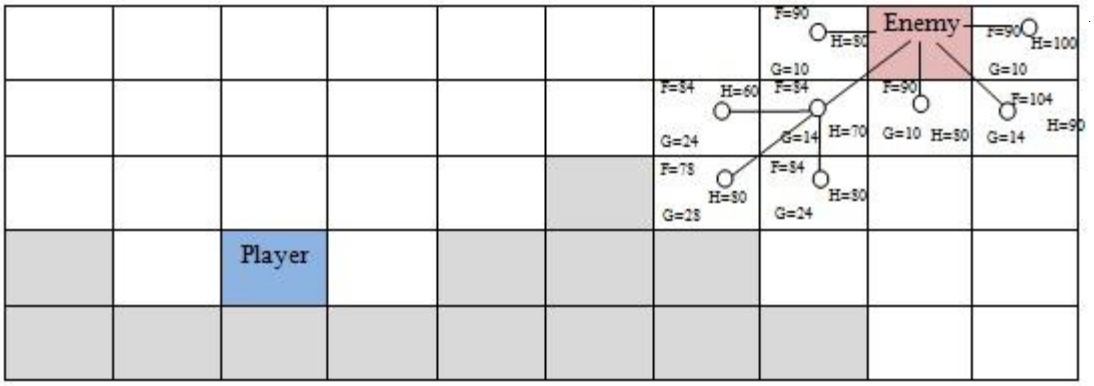
Gambar III.4 Nilai Heuristic pada kasus pencarian jalur terpendek

Terdapat 3 nilai yaitu g,h dan f. Nilai g adalah biaya sebenarnya, nilai h adalah biaya perkiraan sedangkan f adalah total biaya yang diperhitungkan. Untuk pergerakan secara horizontal cost-nya adalah 10 sedangkan untuk biaya diagonal cost-nya adalah 14, dan notasi matematika untuk mendapatkan nilai heuristic *diagonal distance* adalah *h_diagonal*(n)=min(abs(n.x-goal.x)+abs(n.y-goal.y)). Di bawah ini merupakan langkah-langkah pencarian jalur terpendek menggunakan algoritma pencarian (A Star) dapat di lihat pada gambar III.3:



Gambar III.5 Pencarian Jalur Terpendek dengan algoritma pencarian

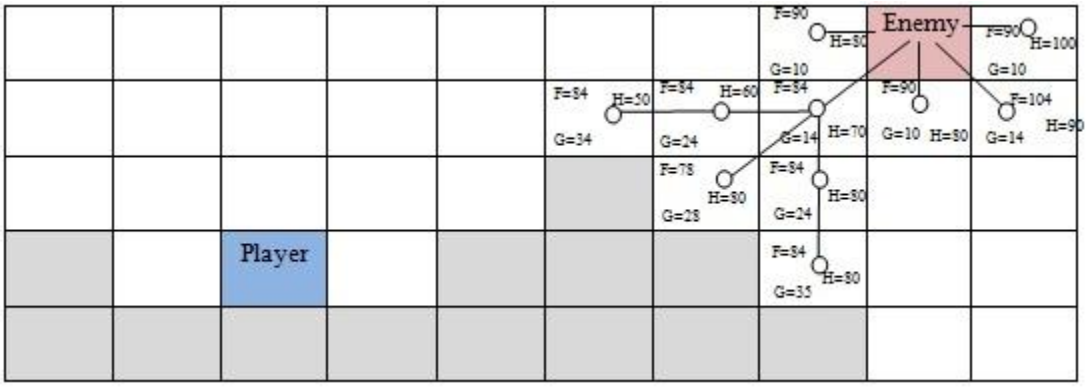
Dari node enemy akan melakukan pencarian secara horizontal, vertikal dan juga diagonal menuju ke arah node player. Pertama-tama untuk pencarian arah vertikal bawah, nilai g=10, h=80 dan f=g+h=90. Untuk horizontal kanan nilai g=10, h=90 dan f=g+h=100. Sedangkan untuk horizontal kiri nilai g=10, h=80 dan f=g+h=90. Lalu untuk yang diagonal kanan bawah g=14, h=90 dan f=g+h=104. Dan untuk diagonal kiri bawag nilai g=14, h=70 dan f=g+h=84. Berikut ini adalah langkah kedua pencarian jalur terpendek dengan algoritma pencarian dapat dilihat pada gambar III.4 di bawah ini:



Gambar III.6 Pencarian Jalur Terpendek dengan algoritma pencarian 2

Karena node dengan nilai f=84 adalah cost yang paling murah, maka kini pencarian di mulai dari node ini. Pertama-tama pencarian dimulai ke arah node vertika bawah, nilai g=24 yang di dapat dari g=14 pada node sebelumnya di tambah 10, h=60, dan f=g+h=84. Untuk yang horizontal kiri nilai g=24 yang didapat dari g=14 pada node sebelumnya ditambah 10, h=60, dan f=g+h=84. Dan untuk diagonal kiri bawah nilai g=28 yang di dapat dari nilai g=14 pada node sebelumnya ditambah dengan 14, h=50 dan f=g+h=78.

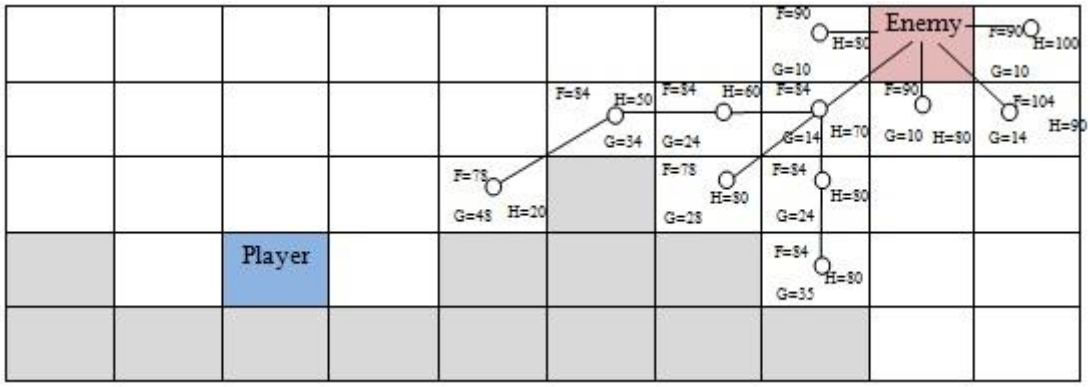
Berikut ini adalah langkah ketiga pencarian jalur terpendek dengan algoritma pencarian yang dapat dilihat pada gambar III.5 di bawah ini:



Gambar III.7 Pencarian Jalur Terpendek dengan algoritma Pencarian

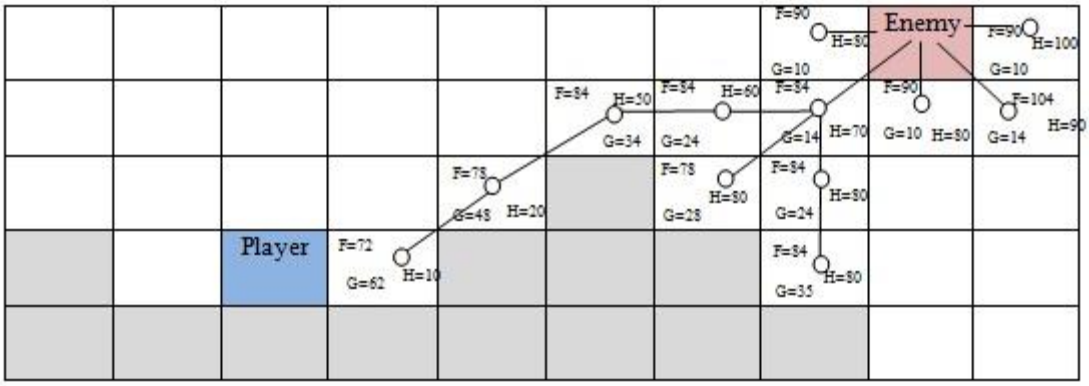
Meski node dengan nilai f=78 adalah lebih kecil dibanding nilai f=84, tapi pencarian tidak bisa dimulai dari node ini karena terhalang oleh node berwarna abu-abu atau penghalang maka sekarang node pencarian dialihkan ke node yang lain. Untuk pencarian yang dimulai dari node vertikal bawah dengan nilai f=84, nilai g=34 yang didapatkan dari g=24 pada node sebelumnya ditambah 10, h=50, dan f=g+h=84. Sedangkan untuk pencarian dari node horizontal kiri dengan nilai f=84, nilai g=34 yang didapat dari g=24 pada node sebelumnya di tambah 10,

h=50, dan f=g+h=84. Berikut ini adalah langkah keempat pencarian jalur terpendek dengan algoritma pencarian yang dilihat dalam tabel III.6 dibawah ini:



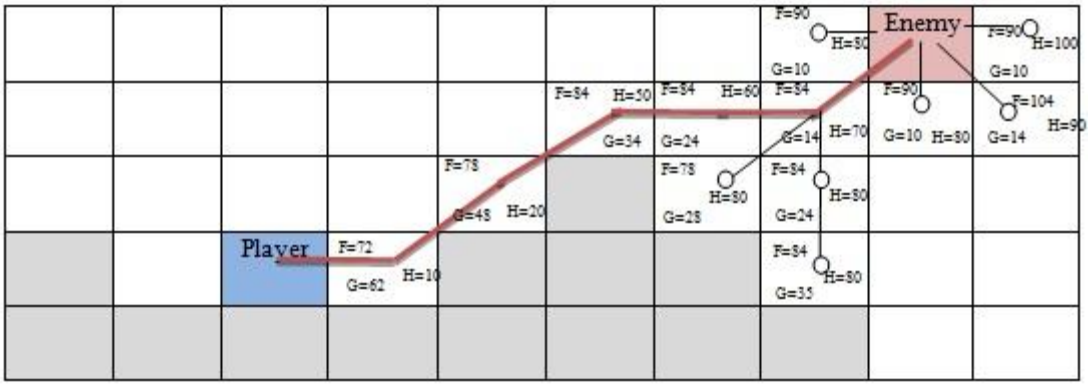
Gambar III.8Pencarian Jalur Terpendek dengan algoritma pencarian

Dari node horizontal kiri dengan nilai f=84 di mulai lagi pencarian ke arah node diagonal kiri bawah, nilai g=48 yang di dapat dari g=34 pada node sebelumnya, h=30, dan f=g+h=78. Berikut ini adalah langkah kelima pencarian jalur terpendek dengan algoritma pencarian (A Star) yang dapat dilihat pada gambar III.7 di bawah ini:



Gambar III.9 Pencarian Jalur Terpendek dengan algoritma pencarian

Dari node diagonal kiri bawah dengan nilai f=78 di mulai lagi pencarian ke arah node diagonal kiri bawah lagi, nilai g=62 yang dudapat dari g=48, h=10, dan f=g+h=72. Berikut ini adalah langkah keenam pencarian jalur terpendek dengan algoritma pencarian dapat dilihat pada gambar III.8 di bawah ini:



Gambar III.10 Pencarian Jalur Terpendek dengan algoritma pencarian 6

Dari node berwarna merah (enemy) menuju node perwarna biru (player) maka di dapatkan jalur terpendek dengan menggunakan algoritma pencarian sebagai berikut:

1. Dari node merah (enemy) mengambil jalur ke arah node diagonal kiri bawah dengan nilai f=84
2. Dari node diagonal kiri bawah bernilai f=84 mengambil jalur ke arah node horizontal kiri dengan nilai f=84
3. Dari node horizontal kiri bernilai f=84 mengambil jalur ke arah node doagonal kiri bawah dengan nilai f=78.
4. Dari node diagonal kiri bawah bernilai f=78 mengambil jalur ke arah node diagonal kiri bawah dengan nilai f=72.
5. Dari node diagonal kiri bawah bernilai f=72 mengambil jalur ke arah horizontal kiri sampailah ke node berwarna biru atau player.

Untuk menghitung nilai heuristic diagonal distance bisa dilihat pada kasus di bawah ini, berikut adalah penjelasannya:

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

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

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

Bahwa ordo dapat disesuaikan dengan rentang X x Y, salah satu contoh perhitungan pada *game* pathfinding ini akan dijelaskan dengan ordo minimal yaitu 3x3 dikarenakan perhitungan pada ordo berapapun akan sama. Pada penentuan bobot setiap node akan diberikan nilai sesuai jarak terdekat ke tujuan, misal nome yang terjauh dari tujuan maka akan diberi bobot yang kecil, sedangkan node yang terdekat dengan tujuan diberi bobot yang lebih besar. Perhitungan yang dilakukan

Dari penjelaskan Algoritma pencarian tersebut, dapat kita tuangakn sebuah pseudo-code yang nantinya akan menjadi acuan untuk penerapan Algoritma tersebut dalam kode program. Pseudo-code algoritma pencarian untuk *game* Ayo Belajar Puasa dapat dilihat pada gambar berikut

```
function a_star return solusi
        open = S
        closed = array
while sukses or open=null
        if open=null then
                gagal
        else
                BestNode=simpul yang ada di open dengan f paling minmal
                Pindahan simpul terbaik tersebut dari open menjadi closed
        If BestNode==goal then
                Sukses
        else
                Bangkitkan semua suksesor BestNode tapi jangan buat pointer
                for suksesor=0,suksesor<suksesor.length
                        hitung g (suksesor)=g (BestNode)+actual cost (dari
                    BestNode ke suksesor)
                        {periksa suksesor}
                If suksesor ada di open {sudah pernah dibangkitkan tapi belum
                    diproses}
```

```
                Old=simpul di open yang sama dengan suksesor tersebut
                BestNode=Old
                Buat pointer dari Old ke BestNode
        If g (Old) lebih baik then
                Ubah parent Old ke BestNode
                Ubah nilai g dan f yang ada pada Old
        end
end
        if suksesor ada di closed then {sudah pernah dibangkitkan tapi
            belum diproses}
                Old=simpul di closed yang sama dengan suksesor tersebut
                Tambahkan Old sebagai suksesor BestNode
                If g (Old) lebih baik then
                        Ubah parent  Old ke BestNode
                        Ubah nilai g dan f yang ada pada Old
                        Propagasi untuk semua suksesor Old dengan
                    penelusuran DFS
                        dengan aturan
                        while suksesor tidak ada di open atau simpul tidak
                    punya suksesor
                                if suksesor ada di Open then
                                        Propagasi diteruskan
                                else
                                        if suksesor ada di open then
```

```
                                        propagasi diteruskan
                                    else
                                        if nilai g via suksesor lebih
                                      baik then
                                            propagasi diteruskan
                                        else
                                            propagasi dihentikan
                                        end
                                    end
                                end
                            else suksesor tidak ada di open dan closed
                                masukan suksesor ke open
                                tambahkan suksesor tersebut sebagai
                              suksesornya BestNode
                                F=g(suksesor)+h(suksesor)
                            End
                        End
                    End
End
```

Gambar III.14 Pseudocode Algoritma pencarian

III.1.5 Analisis Kebutuhan Non Fungsional

Analisis kebutuhan sistem non fungsional dilakukan untuk mengetahui spesifikasi kebutuhan untuk sistem. Spesifikasi kebutuhan melibatkan analisis pengguna, analisis perangkat keras, analisis perangkat lunak.

III.1.4.1 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. Pada aplikasi *game* "Ayo Belajar Berpuasa", perangkat keras yang digunakan adalah sebagai berikut :

1. Processor minimal 2,6 GHZ
2. RAM : 1GB
3. VGA 512 MB
4. Harddisk 320 GB
5. Monitor 17'
6. Keyboard

III.1.4.2 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 aplikas "Ayo Belajar Berpuasa" ini adalah sebagai berikut :

1. Sistem Operasi Windows XP.
2. Greenfoot 2.2.1
3. Java

III.1.4.3 Sasaran User

Game yang akan dibuat ditujukan untuk user dengan usia dari kelompok anak-anak berusia antara 5-12 tahun, alasannya adalah karena *Game* ini cocok sekali bagi anak yang akan memulai balajar berpuasa karena *Game* ini berusaha memberikan edukasi tentang makna berpuasa dan kesabaran dalam menjalankan puasa, sehingga tanpa disadari *game*r telah mempelajari pesan moral yang di sampaikan, tetapi tidak dengan cara yang membosankan. Usia ini juga merupakan usia yang efektif untuk mempelajari berbagai hal dan merupakan kurun waktu yang sangat penting dan kritis dalam hal tumbuh kembang fisik, mental, dan

psikososial, yang berjalan sedemikian cepatnya. Pada masa usia ini juga sering disebut dengan masa usia emas (the golden age).

Game yang dibuat merupakan *game* PC (Personal Computer) atau bisa disebut juga *game desktop*.

III.1.6. Analisis Kebutuhan Fungsional

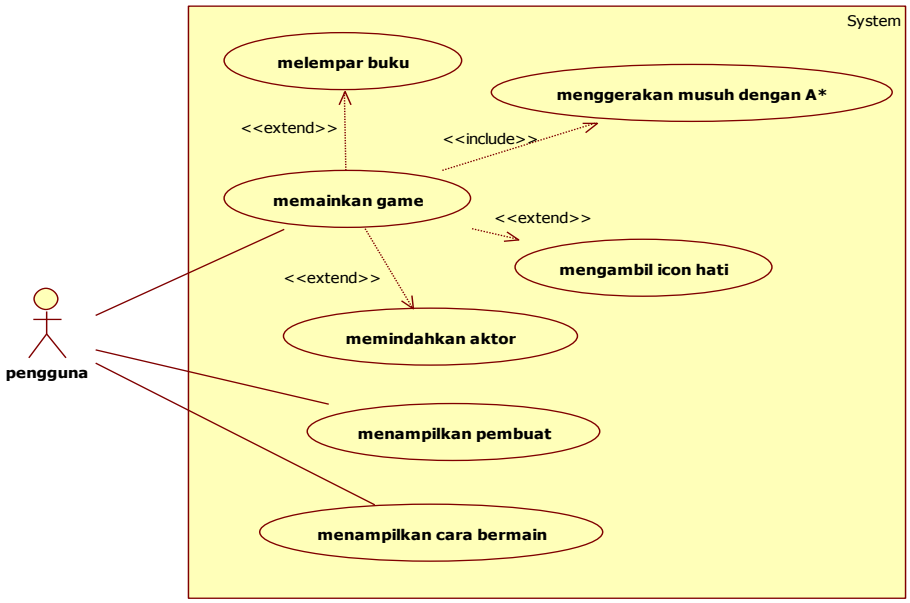
Analisis kebutuhan fungsional merupakan tindak lanjut dari analisis sistem. Tujuannya adalah untuk memberikan gambaran mengenai sistem yang akan dibuat. Tahap awal yang dilakukan dalam melakukan analisis berorientasi objek mengunakan UML (Unified Modeling Language). Dimana tahap-tahap perancangan yang dilakukan dalam membangun *Game* Ayo Belajar Puasa antara lain Use Case Diagram, Class Diagram, Sequence Diagram, Activity Diagram.

II.I.5.1 Identifikasi Aktor

Tahap pertama yang dilakukan dalam melakukan analisis berorientasi objek menggunakan UML adalah menentukan aktor atau pengguna sistem. Kata aktor dalam konteks UML, menampilkan peran (roles) yang pengguna (atau sesuatu di luar sistem yang dikembangkan yang dapat berupa perangkat keras, end user, sistem yang lain, dan sebagainya).

III.1.5.2 Use Case Diagram

Berikut ini adalah perancangan proses-proses yang terdapat pada *game* Ayo Belajar Puasa:



Gambar III.15 Use Case Diagram *Game* Ayo Belajar Puasa

1. Definisi Use Case

Definisi Use Case berfungsi untuk menjelaskan proses yang terdapat pada setiap Use Case. Definisi use case pada *game* ini padat dilihat dalam tabel III.3 di bawah ini:

Tabel III.3Definisi Use Case

No.	Use Case	Deskripsi.
1.	Melemparkan buku	Memberikan action kepada aktor untuk melemparkan buku kepada musuh, tetapi apabila pemain melakukannya dan buku tersebut mengenai musuh maka poin pemain akan berkurang.
2	Memainkan *Game*	Setiap proses yang berkaitan dengan menggerakan aktor, mengambil buku dan interaksi antara aktor dan sistem.
3.	Menggerakan aktor	Pemain bisa menggerakan aktor maju, mundur dan loncat.

4.	Menampilkan Pembuat	Menampilkan informasi pembuat *game*
5.	Menampilkan Cara Bermain	Menampilkan informasi cara memainkan *game*
6.	Mengambil *Icon* Hati	Aktor bisa mengambil *icon* hati dengan tujuan untuk menambah *health point.*
7.	Menggerakan Musuh Denga pencarian	Otomatisasi pergerakan musuh dengan menggunakan Algoritma pencarian

2. Sekenario Use Case

Skenario *Usecase* mendeskripsikan urutan langkah-langkah dalam proses bisnis, baik yang dilakukan aktor terhadap sistem maupun yang dilakukan oleh sistem terhadap aktor

a. Melempar Buku

Interaksi antara aktor pengguna dengan *use case,* Melempar buku dijelaskan dalam *use case* skenario dalam tabel III.4 di bawah:

Tabel III.4 Usecase Skenario Melempar Buku

Identifikasi	
Nama	Melempar Buku
Tujuan	Agar dapat menampilkan/menggerakan aktor agar melemparkan buku
Deskripsi	Menampilkan/menggerakan aktor agar melemparkan buku.
Aktor	User
Skenario Utama	
Kondisi awal	User menekan tombol F
Aksi Aktor	**Reaksi Sistem**
User menekan tombol F pada keyboard	Sistem akan membaca kondisi terakhir dari arah aktor.

	Jika arah aktor menghadap ke kiri, maka posisi buku yang dilempar oleh aktor akan berpindah ke arah kiri.
	Jika aktor menghadap ke kanan, maka posisi buku yang dilempar oleh aktor akan berpindah ke arah kanan.
Kondisi akhir	Buku yang dilempar berpindah tempat sesuai dengan arah lemparan.

b. Memainkan *Game*

Interaksi antara aktor pengguna dengan *use case*, Memainkan *Game* dijelaskan dalam *use case* skenario dalam tabel III.5 di bawah ini :

Tabel III.5 Usecase Skenario Memainkan *Game*

Identifikasi	
Nama	Memainkan *Game*
Tujuan	Agar dapat memastikan user untuk memulai permainan baru
Deskripsi	Menu ini akan menampilkan permainan yang dimainkan oleh user
Aktor	User
Skenario Utama	
Kondisi awal	User menjalankan aplikasi naofal.jar
Aksi Aktor	**Reaksi Sistem**
User mengklik Tombol Mulai	Sistem akan menampilkan komik dari stage pertama
User klik tombol selanjutnya	Sistem melakukan proses *loading* semua komponen *game*

	Sistem menampilkan seluruh komponen *game* yang telah diinisialiasi pada kelas naofalworld pada layar.
Kondisi akhir	User memulai permainan stage pertama.

c. Memindahkan Aktor

Interaksi antara aktor pengguna dengan *use case* menampilkan memindahkan aktor dijelaskan dalam *use case* skenario dalam Tabel III.6 di bawah ini:

Tabel III.6 *Usecase Skenario* Memindahkan Aktor

Identifikasi	
Nama	Menggerakan Aktor
Tujuan	Agar dapat menampilkan menggerakan aktor ke kanan, kiri, loncat, lempar dan lain-lain.
Deskripsi	Menampilkan aktor akan menggerakan aktor ke kanan, kiri, loncat, lempar dan lain-lain
Aktor	User
Skenario Utama	
Kondisi awal	User menekan tombol atas, bawah, kiri, kanan dan enter.
Aksi Aktor	**Reaksi Sistem**
User menekan tombol	Sistem membaca masukan yang diperoleh dari *keyboard*. Apakah masukan berupa *key_left*, *key_right*, atau *key_up*
	Jika *input user*, diluar tombol tersebut, maka sistem mengabaikan *input.*
	Jika *input* berupa *key_left*, maka pindahkan *background* ke arah kanan dan pindahkan koordinat X aktor ke arah kiri.

	Jika *input* berupa *key_right*, maka pindahkan *background* ke arah kiri dan pindahkan koordinat X aktor ke arah kanan.
	Jika *input* berupa *key_up*, maka kurangi koordinat Y dari aktor
Kondisi akhir	Aktor bergerak.

d. Menampilkan Pembuat

Interaksi antara aktor pengguna dengan *use case* Menampilkan Pembuat dijelaskan dalam *use case* skenario dalam Tabel III.7 di bawah ini:

Tabel III.7 *Usecase Skenario* Menampilkan Pembuat

Identifikasi	
Nama	Menampilkan Pembuat
Tujuan	Agar dapat menampilkan pembuat aplikasi *game* edukasi ini kepada user
Deskripsi	Menu ini akan menampilkan data diri Tampilan Pembuat
Aktor	User
Skenario Utama	
Kondisi awal	User membuka menu utama
Aksi Aktor	**Reaksi Sistem**
User memilih Tampilan Pembuat	Sistem akan menampilkan form Tampilan Pembuat kepada *user*.
Kondisi akhir	Sistem menampilkan Tampilan Pembuat pada user

e. Cara Bermain.

Interaksi antara aktor pengguna dengan *use case* Memilih Cara Bermain dijelaskan dalam *use case* skenario dalam tabel III.8 di bawah ini:

Tabel III.8 *Usecase Skenario* Memilih Cara Main

Identifikasi	
Nama	Memilih Cara Bermain
Tujuan	Agar dapat menampilkan petunjuk permainan kepada user
Deskripsi	Menu ini akan menampilkan Tampilan Petunjuk Permainan
Aktor	User
Skenario Utama	
Kondisi awal	*User* menampilkan halaman utama
Aksi Aktor	**Reaksi Sistem**
User klik menu panduan	Sistem akan menampilkan form cara bermain kepada *user*.
Kondisi akhir	User dapat melihat konten yang ada di dalam form cara bermain.

g. Menggerakan Musuh Dengan pencarian

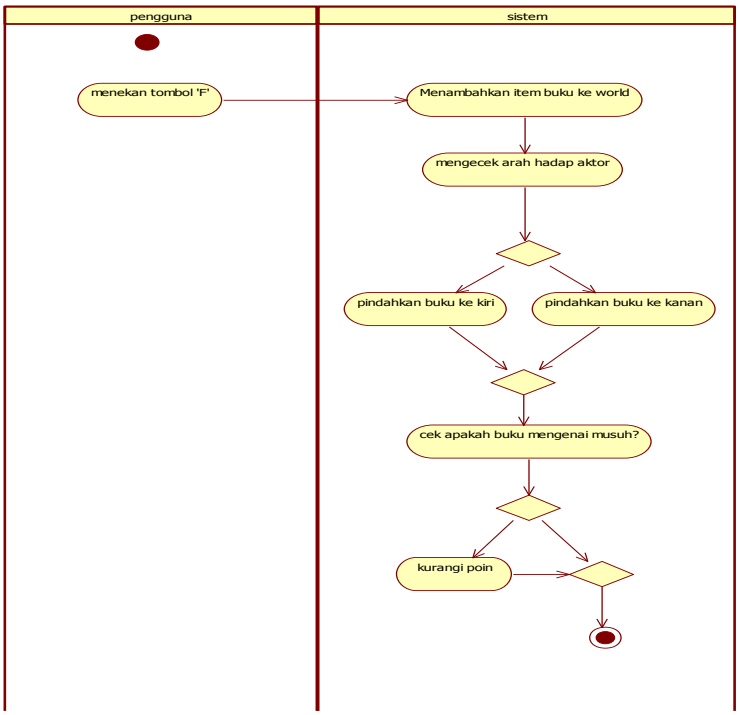
Proses ini dilakukan secara otomatis yang artinya bukan diaktifkan oleh pengguna. Proses dari pergerakan musuh menggunakan algoritma pencarian yang memiliki tujuan guna mendekatkan musuh kepada aktor.

Tabel III.9 *Use case* Skenario Menggerakan Musuh Dengan pencarian

Identifikasi	
Nama	Menggerakan musuh dengan pencarian
Tujuan	Memungkinkan musuh untuk mendekati aktor.
Deskripsi	Memberikan gerakan otomatis/terprogram yang memungkinkan musuh untuk mendekati aktor. Tidak semua musuh memiliki kemampuan ini. Hanya boss saja yang diberi kemampuan ini.
Aktor	Pemain
Skenario Utama	
Kondisi awal	Pengguna sudah berada di area bos
Aksi Aktor	**Reaksi Sistem**
Pengguna mengerakan aktor	Sistem akan membaca letak atau koordinat dari aktor.
	Sistem menjalankan algoritma pencarian
	Sistem memindahkan Boss mendekati aktor.
Kondisi akhir	Aktor dan boss saling berdekatan.

III.1.5.3 Activity Diagram

a. Melempar Buku



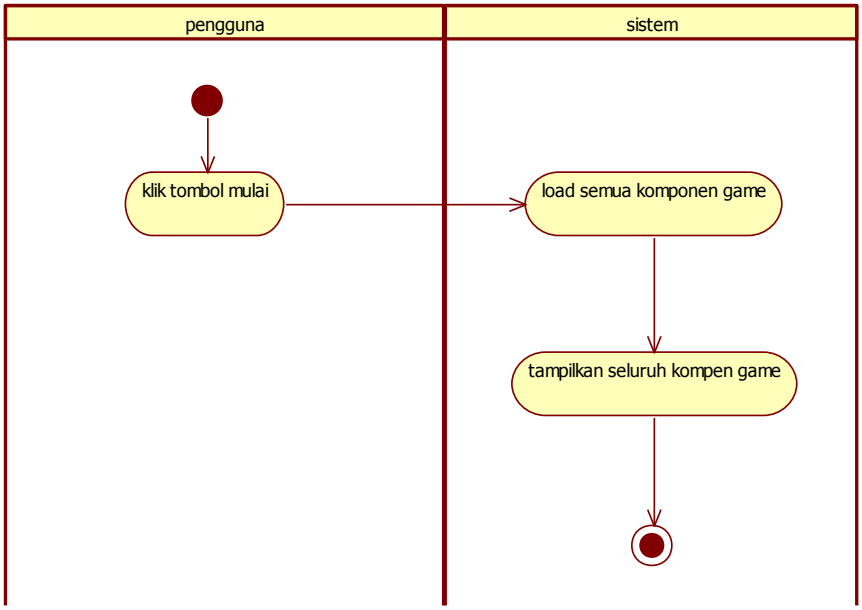
Gambar III.16 Activity Diagram- Melempar Buku

Deskripsi: Memberikan action kepada aktor untuk melemparkan buku kepada musuh, tetapi apabila pemain melakukannya maka poin pemain akan berkurang.

Normal Flow:

1. Pemain menekan tombol F
2. Sistem menambahkan item buku ke world
3. Sistem mengecek arah hadap aktor.
4. Apabila aktor menghadap ke kiri, maka sistem akan memindahkan buku ke arah kiri. Begitu juga sebaliknya.
5. Sistem mengurangi poin jika buku yang dilemparnya menyenai musuh.

b. Memainkan *Game*



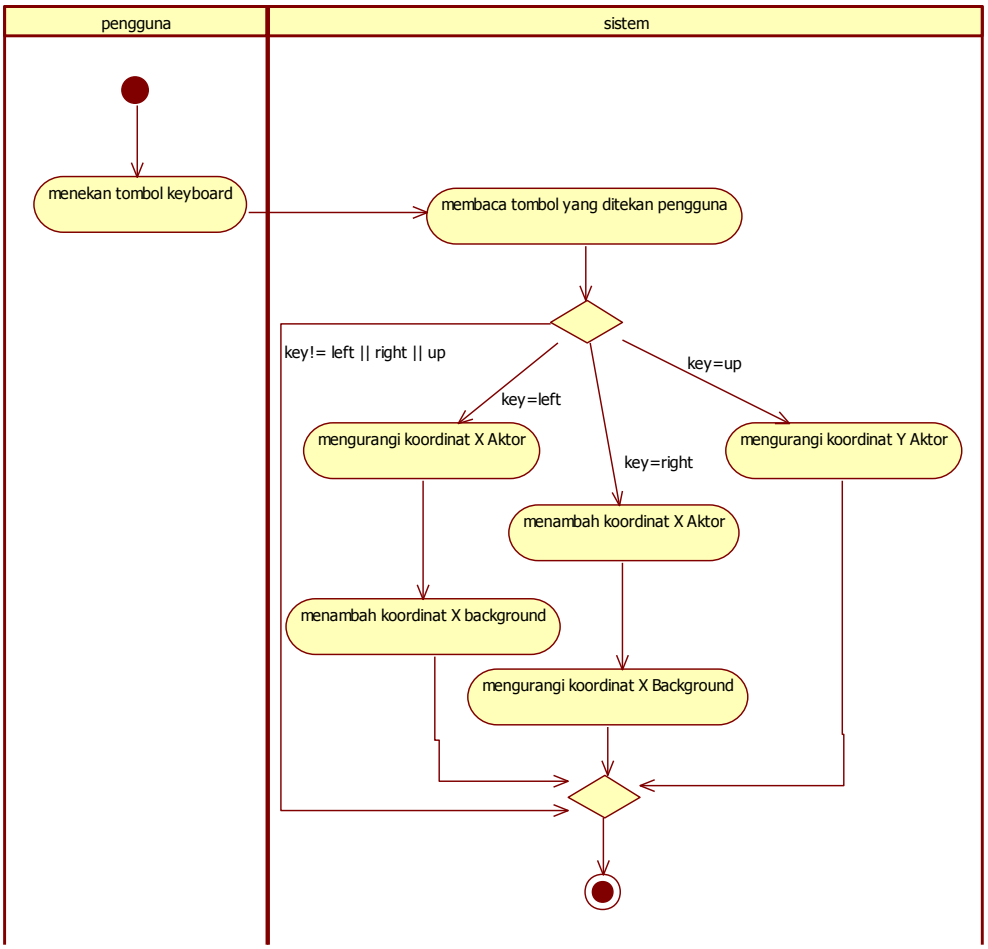
Gambar III.17 Activity Diagram-Memainkan *Game*

Deskripsi: Tampilan ini menampilkan pilihan Mulai Baru, dimana pemain apabila memilih pilihan "Memulai *Game*", akan langsung memasuki stage awal di permainan.

Normal Flow:

1. Pemain memilih pilihan Memulai *Game*.
2. Sistem *load* seluruh komponen *game*
3. Sistem menampilkan seluruh komponen.

c. Memindahkan Aktor



Gambar III.18 Activity Diagram-Menggerakan Aktor

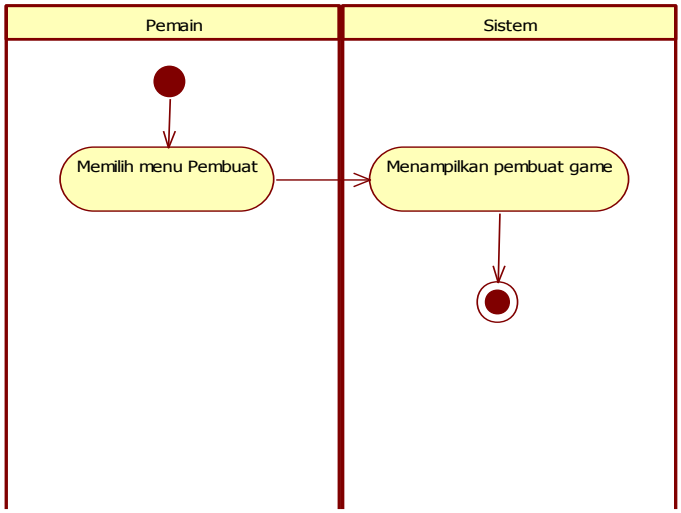
Deskripsi: Pemain bisa menggerakan aktor maju, mundur dan loncat.

Normal Flow:

1. Pemain menekan Keyboard.
2. Sistem membaca tombol yang ditekan oleh pemain.
   - Jika pemain menekan tombol kiri, maka sistem akan mengurangi koordinat x dari aktor dan memindahkan backround ke kanan.
   - Jika pemain menekan tombol kanan, maka sistem akan menambah koordinat x dari aktor dan memindahkan backround ke kiri.

- Jika pemain menekan tonbol up, maka sistem akan mengurangi koordinat y aktor.
- Jika sistem tidak menekan salah satu diantara tombol tersebut, maka sistem membiarkannya.

d. Menampilkan Pembuat.



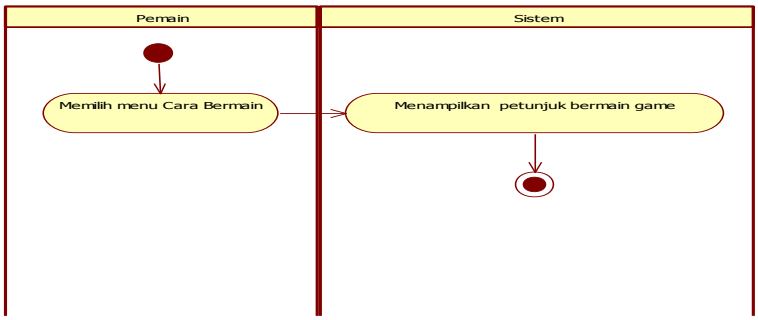
Gambar III.19 Activity Diagram-Menampilkan pembuat

Deskripsi: Menampilkan informasi pembuat *game*.

Normal Flow:

1. Pemain memilih menu pembuat.
2. Sistem menampilkan pembuat *game*.

e. Cara Bermain



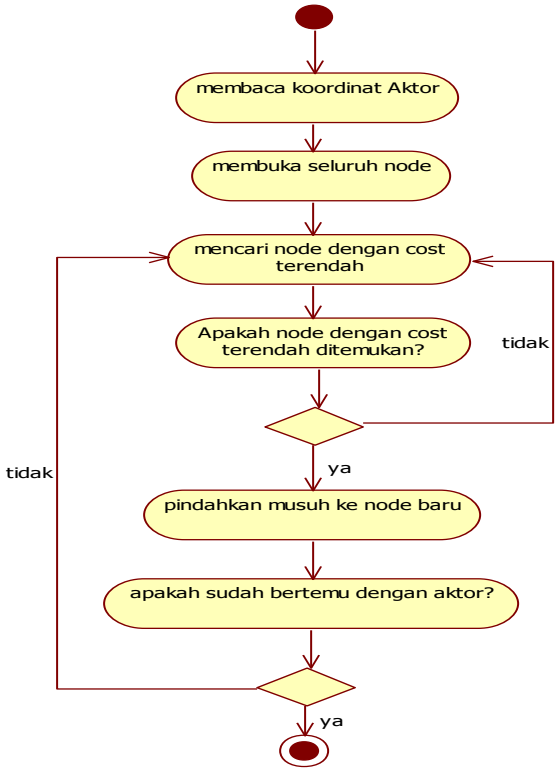
Gambar III.20 Activity Diagram-Cara Bermain

Deskripsi: Menampilkan informasi cara bermain *game*.

Normal Flow:

1. Pemain memilih menu cara bermain.
2. Sistem menampilkan petunjuk dalam permainan.

f. Menggerakan Musuh dengan pencarian



Gambar III.21 *Activity Diagram* menggerakan aktor dengan pencarian

Deskripsi: Menggerakan aktor dengan pencarian

Normal Flow:

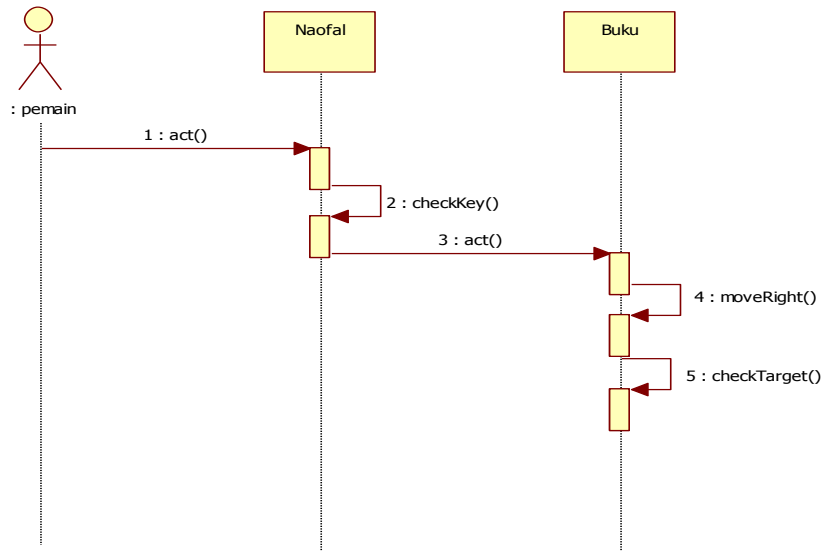
1. Sistem membaca koordinat dari Aktor
2. Sistem membuka seluruh node yang memungkinkan untuk dilalui
3. Sistem mencari node dengan *cost* terendah
4. Node dengan *cost* terendah ditemukan
5. Pindahkan musuh ke node terbaru
6. Musuh bertemu dengan aktor

III.1.5.4 Sequence Diagram

Sequence diagram menggambarkan kelakuan objek pada use case dengan mendeskripsikan waktu hidup objek dan *message* yang dikirimkan dan diterima antar objek.

a. Melemparkan Buku

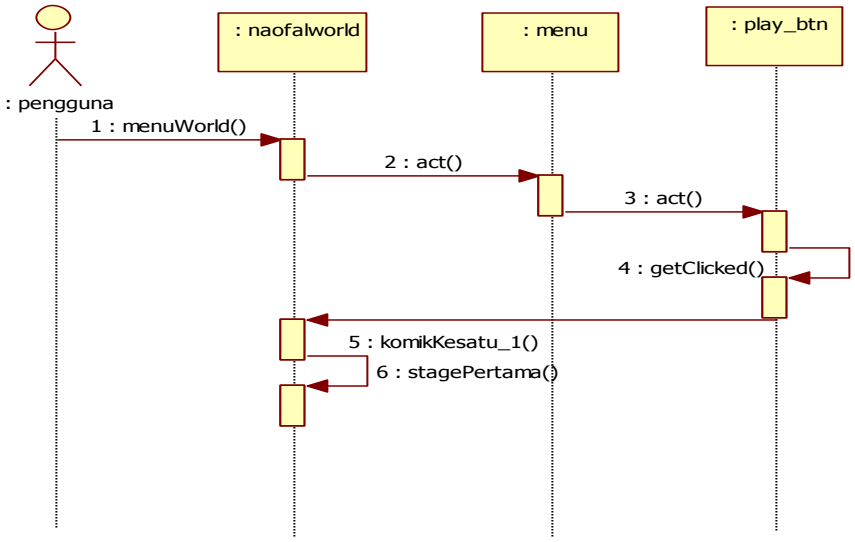
Sequence diagram untuk Use Case melemparkan buku menggambarkan interaksi antara objek dari class Page melemparkan buku dan objek yang berkaitan dengan class lainya. Keseluruhan sequence melemparkan buku dapat dilihat pada Gambar III.22 di bawah ini:



Gambar III.22 Sequence Diagram- Melempar Buku

b. Memainkan *Game*

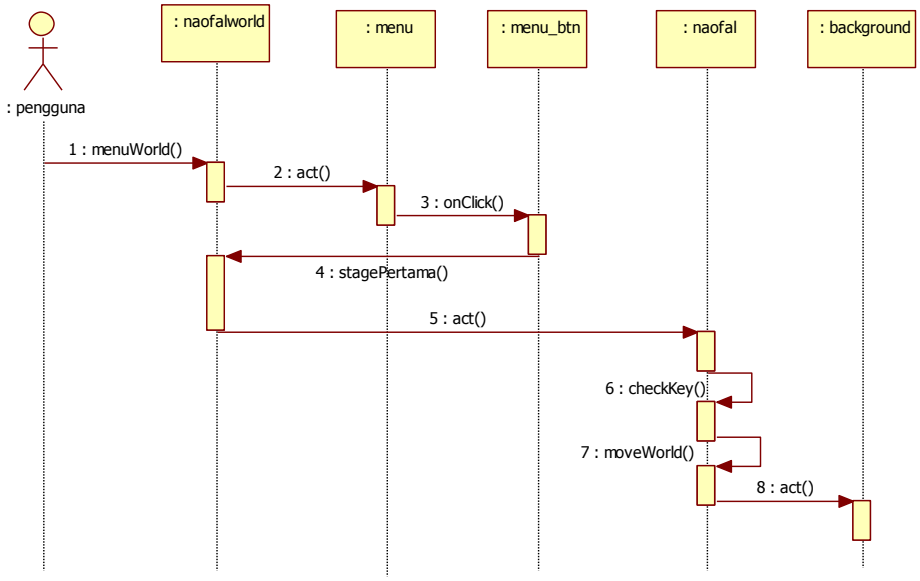
Sequence diagram untuk Use Case Memainkan *Game* menggambarkan interaksi antara objek dari class Page Memainkan *Game* dan objek yang berkaitan dengan class lainya. Keseluruhan sequence Memainkan *Game* dapat dilihat pada Gambar III.23 di bawah ini:



Gambar III.23 Sequence Diagram- Memainkan *Game*

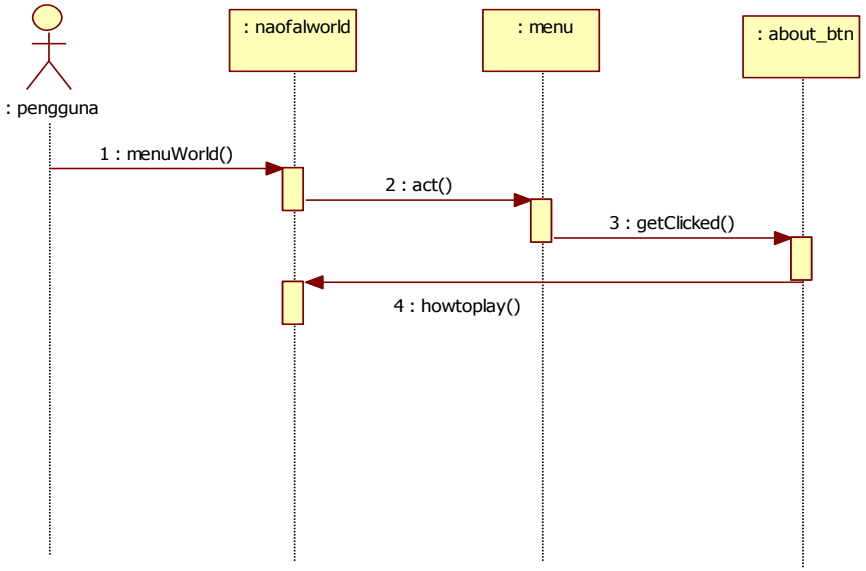
c. Memindahkan Aktor

Sequence diagram untuk *Use Case* Memindahkan Aktor menggambarkan interaksi antara objek dari class Page Menggerakan Aktor dan objek yang berkaitan dengan class lainya. Keseluruhan sequence Menggerakan Aktor dapat dilihat pada Gambar III.24 di bawah ini



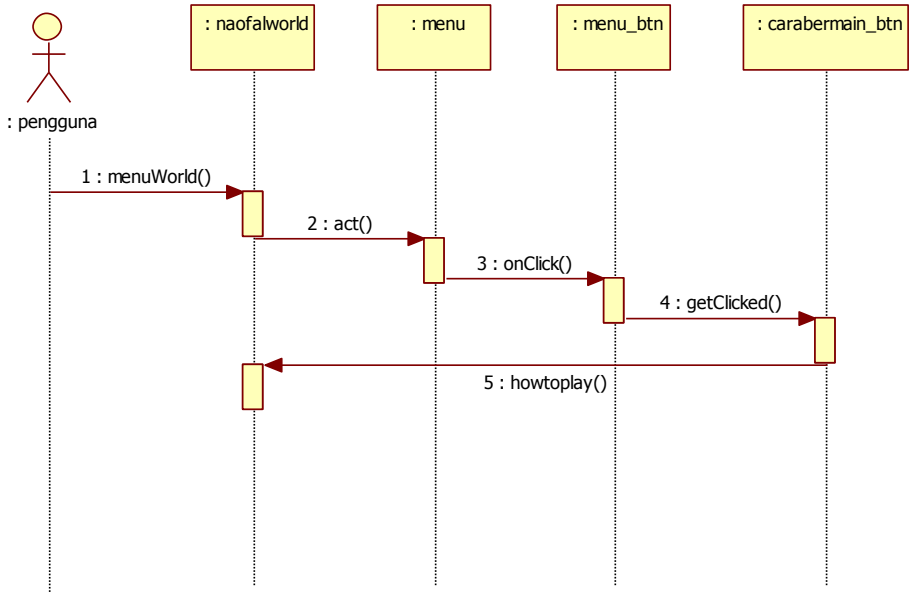
Gambar III.24 Sequence Diagram- Menggerakan Karakter.

d. Menampilkan Pembuat



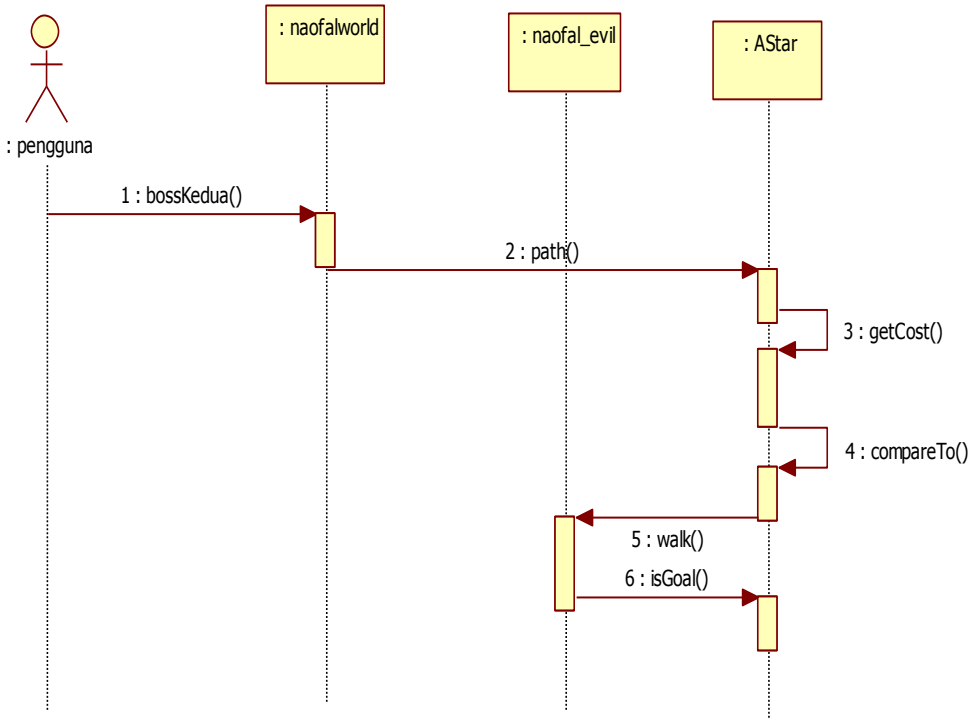
***Gambar III.25 Sequence Diagram* menampilkan tentang aplikasi**

e. Cara Bermain



***Gambar III.26 Sequence Diagram* menampilkan cara bermain**

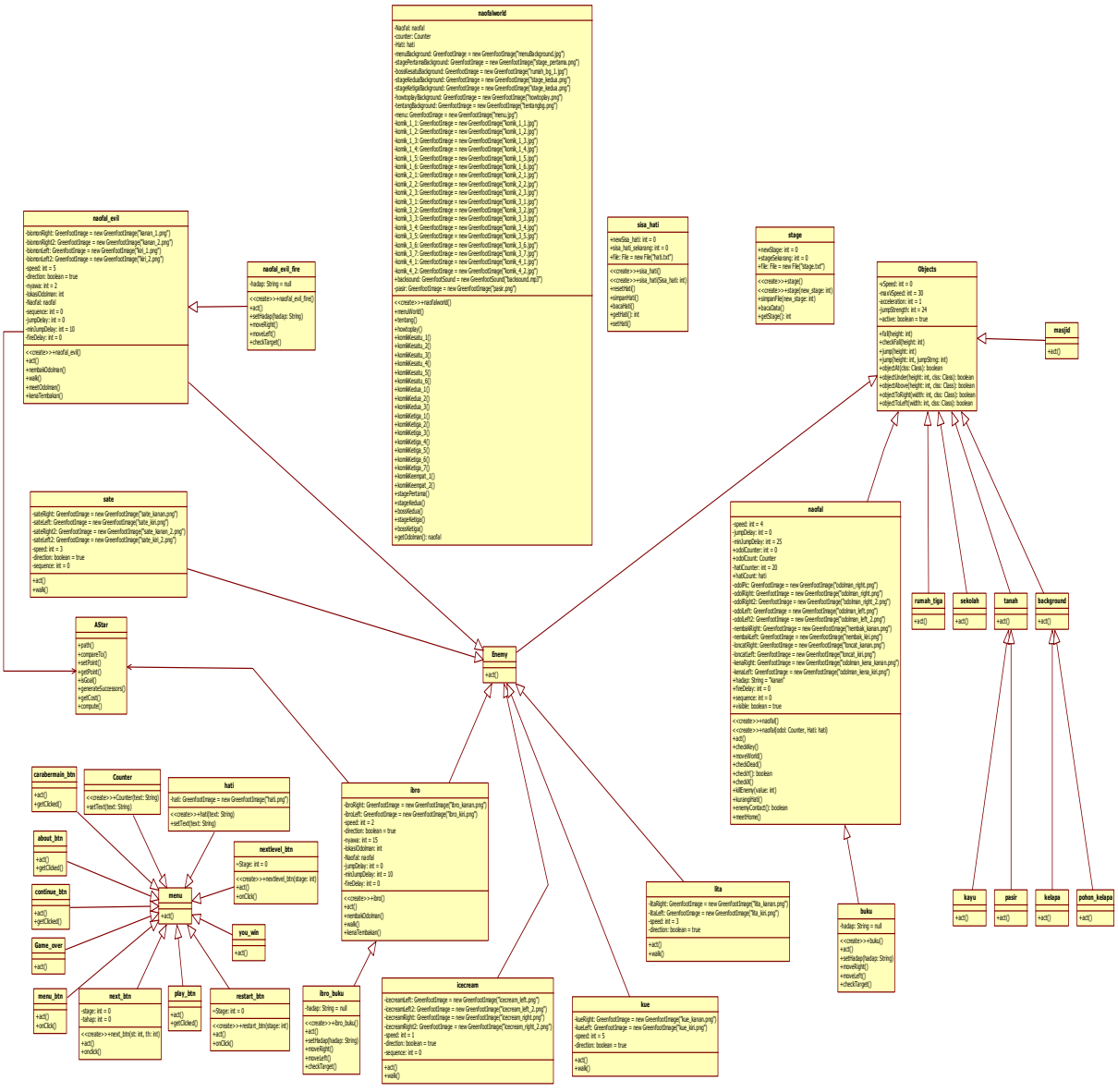
f. Menggerakan Musuh dengan pencarian



***Gambar III.27 Sequence Diagram* Menggerakan Musuh**

III.1.5.5 Class Diagram

Diagram kelas atau atau class diagram menggambarkan struktur sistem dari segi pendefinisian kelas-kelas yang dibuat untuk membangun sistem. Gambar III.19 di bawah ini adalah perancangan struktur sistem yang terdapat pada *game* Ayo Belajar Puasa.



Gambar III.28 Class Diagram Ayo Belajar Puasa

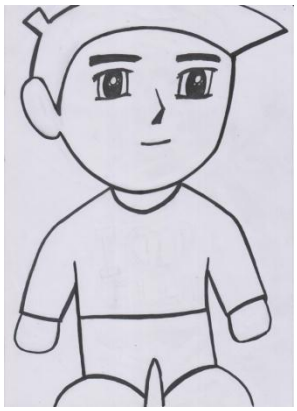
III.2 Perancangan Sistem

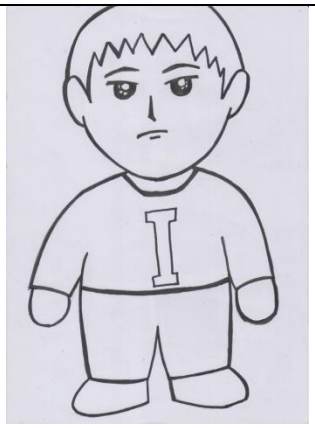
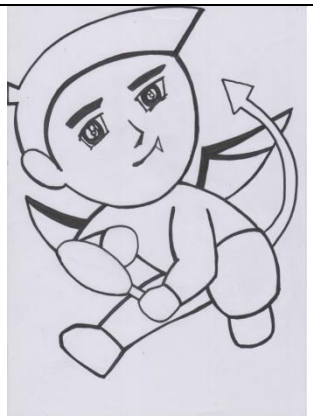
III.2.1 Pengenalan Karakter

Pengenalan karakter merupakan pembahasan mengenai karakter-karakter yang terlibat dalam *game* Ayo Belajar Puasa. Pengenalan karakter dapat dilihat pada Tabel III.10 di bawah ini:

Tabel III.10

Pengenalan Karakter

No	Nama	Gambar	Keterangan
1	Karakter Utama		✓ Karakter ini merupakan karakter yang akan digunakan oleh pengguna untuk memainkan permainan. ✓ Karakter ini digambarkan sebagai seorang anak laki-laki berusia 6 tahun.
2	Lita (Karakter musuh level 1)		✓ Karakter musuh merupakan karakter yang terdapat setiap stage, karakter ini harus dihindari oleh pemain apabila pemain ingin melanjutkan ke stage selanjutnya. ✓ Karakter musuh level 1 adalah Para Lita anak balita yang banyak berkeliaran dan mengganggu konsentrasi karakter utama dalam

			menjalankan puasa.
3	Ibro (Musuh Level 2)		✓ Karakter musuh level 2 adalah seorang anak nakal yang selalu mengganggu karakter utama. Ibro akan melempari mengejar-ngejar Naofal dan melempari Naofal dengan makanan.
4	Evil Naofal (Musuh Utama)		✓ Karakter musuh utama dan paling kuat. Evil Naofal adalah cerminan dari tokoh utama dalam bentuk yang jahat dan nakal. Evil Naofal adalah gambaran diri tokoh utama sendiri yang sedang menikmati makanan kesukaannya. ✓ Evil Naofal akan mengejar dan melempari Naofal dengan makanan kesukaannya.

III.2.2 Story Board

Storyboard adalah sketsa gambar yang disusun berurutan sesuai dengan naskah, dengan storyboard kita dapat menyampaikan ide cerita kita kepada orang lain dengan lebih mudah, karena kita dapat menggiring khayalan seseorang

mengikuti gambar-gambar yang tersaji, sehingga menghasilkan persepsi yang sama pada ide cerita kita.

Berikut ini adalah *storyboard* dari *Game* "Ayo Belajar Puasa":

a. *Storyboard* menu utama



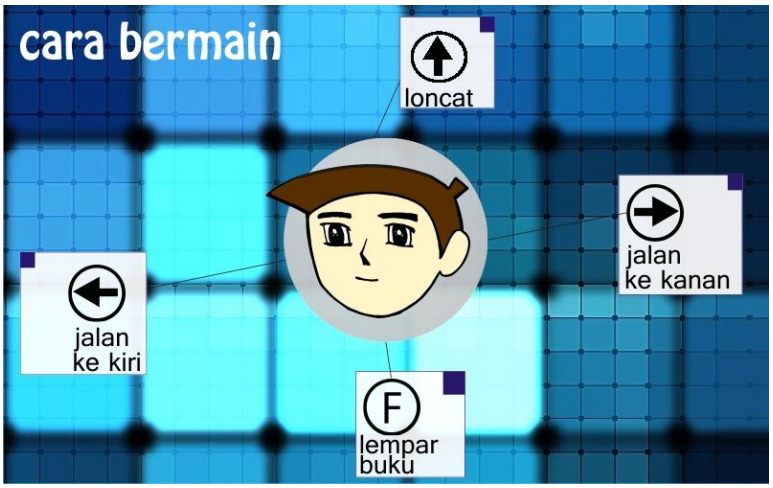
Gambar III.29 Storyboard Menu Utama

b. *Storyboard* cerita



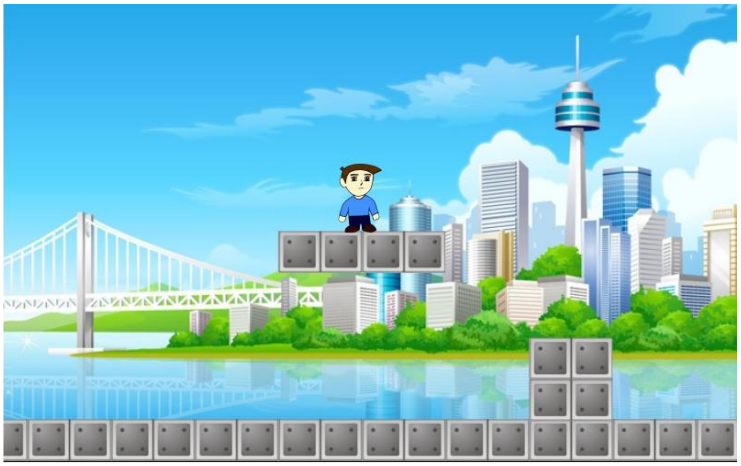
Gambar III.30 Storyboard cerita

c. *Storyboard* menampilkan cara bermain



***Gambar. III.31 Storyboard* cara bermain**

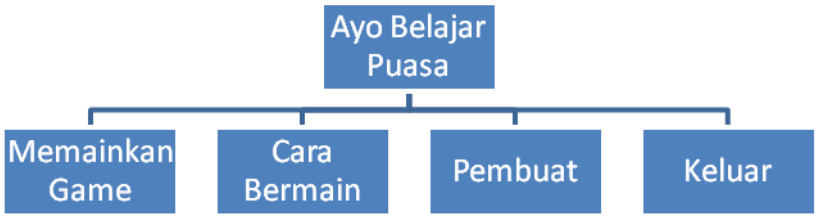
d. *Storyboard* tampilan permainan



***Gambar III.32 Storyboard* tampilan permainan**

III. 2. 3 Perancangan Stuktur Menu

Perancangan menu dibuat dengan harapan agar pengguna dapat mengunakan *Game* dengan mudah tanpa kesulitan, sehingga memudahkan pemakai dalam memilih menu aplikasi yang sedang berjalan. Stuktur menu dapat dilihat pada Gambar III.33 di bawah ini:

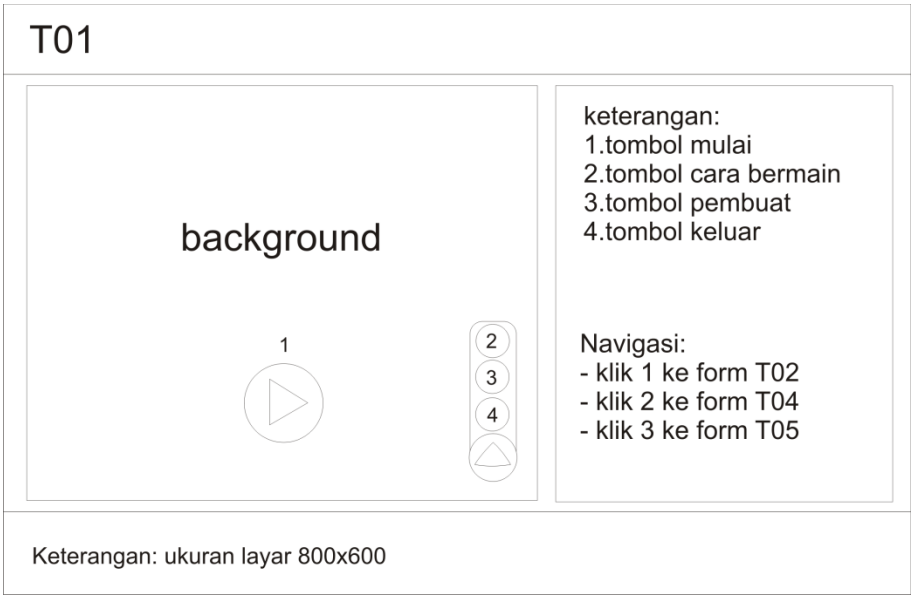


Gambar III.33 Perancangan Struktur Menu *Game* Ayo Belajar Puasa

III.2.4 Perancangan Antar Muka

Antarmuka (*interface*) merupakan bentuk tampilan dari program yang tampil pada layar komputer yang bertujuan memberikan gambaran tentang aplikasi yang akan dibangun, sehingga akan memudahkan pengimplementasikan aplikasi sesuai ukuran layar dan memudahkan dalam pembuatan aplikasi.

a. Rancangan Menu Utama



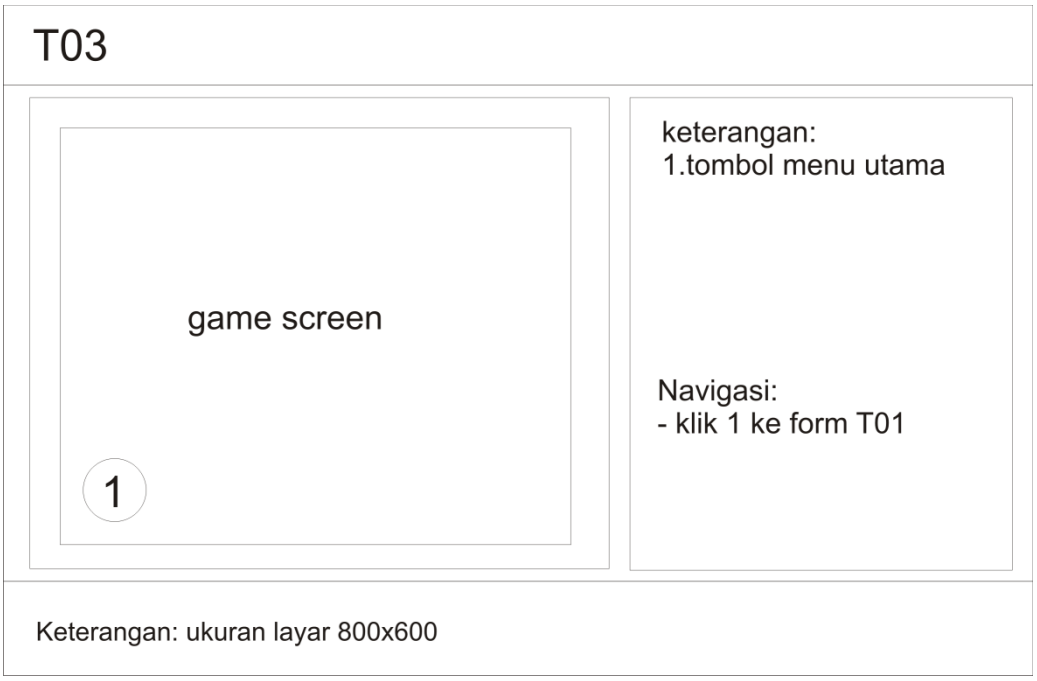
Gambar III.34 Rancangan Menu Utama

b. Rancangan Menu Ilustrasi Cerita



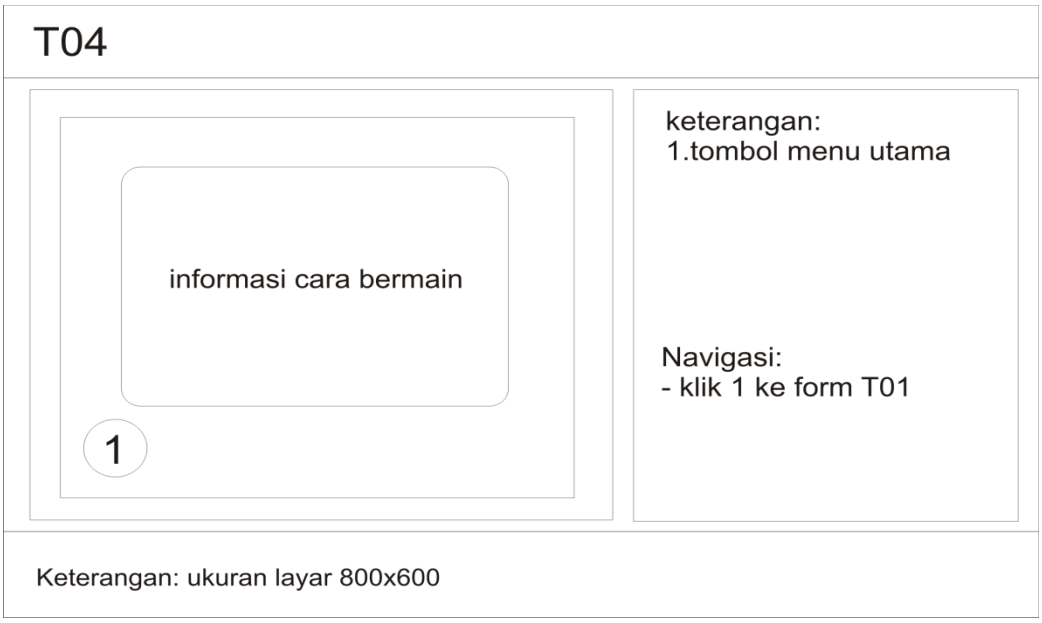
Gambar III.35 Rancangan Ilustrasi Cerita

c. Rancangan Menu Memainkan *Game*



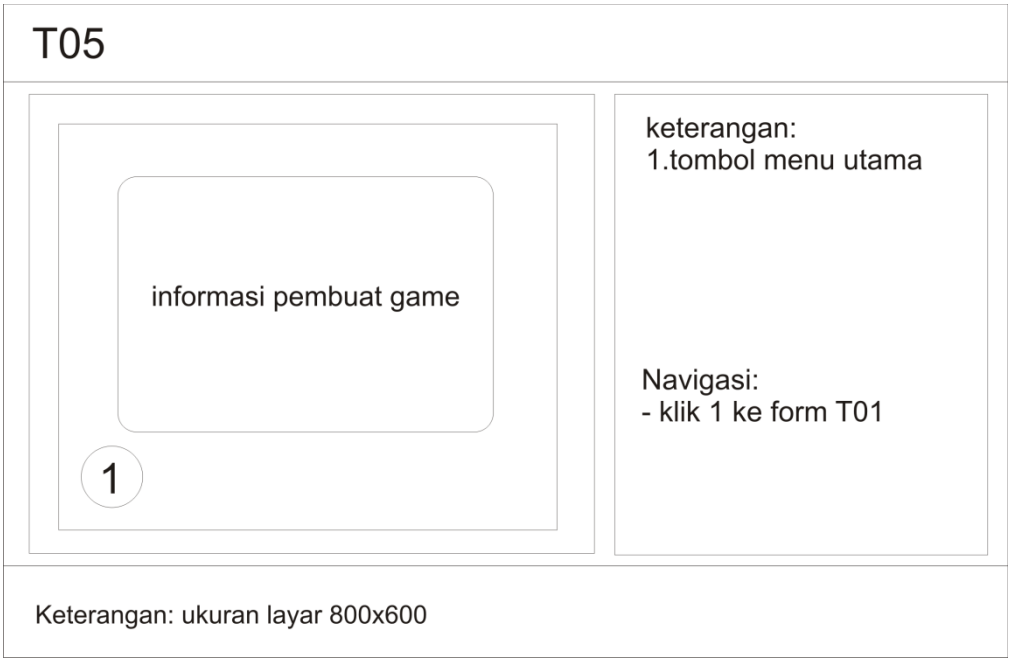
Gambar III.36 Rancangan Menu Memainkan *Game*

d. Rancangan Menu Cara Bermain



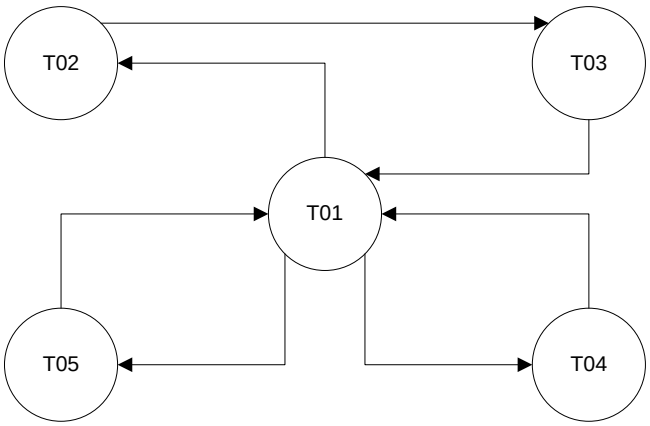
Gambar III.37 Rancangan Menu Cara Bermain

e. Rancangan Menu Tentang Pembuat



Gambar III.38 Rancangan Menu Tentang Pembuat

III.2.5 Jaringan Semantik



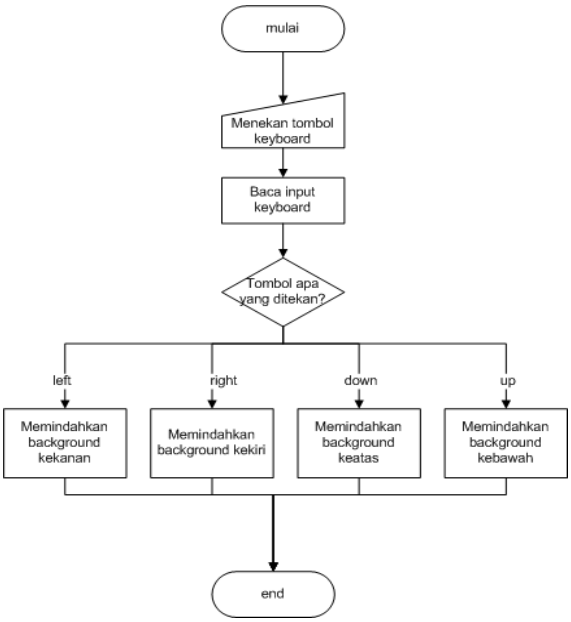
Gambar III.39 Jaringan Semantik

III.2.6 Perancangan Method

Perancangan method memberikan gambaran tentang alur dari method-method yang ada pada *game* Ayo Belajar Puasa.

A. Method *Checkkey*

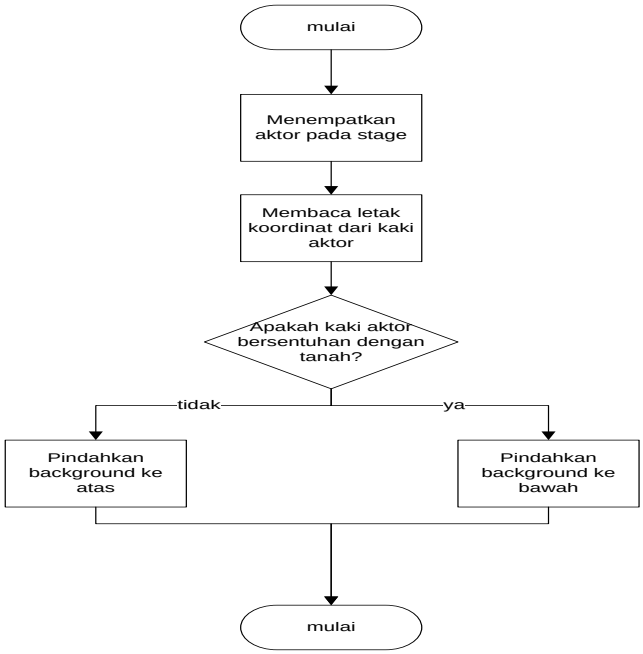
Method *checkkey* digunakan untuk membaca input dari pengguna, input tesebut berupa tombol yang di tekan pada *keyboard.* Peracangan method *checkkey* bisa dilihat pada gambar



Gambar III.40 Perancangan method *checkkey()*

B. Method cekJatuh()

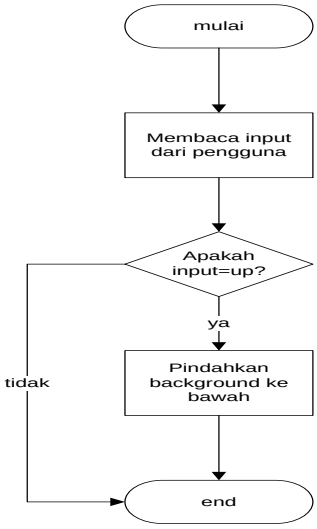
Method cekJatuh berfungsi untuk melakukan pengecekan apakah aktor/karakter ada diatas tanah atau tidak. Jika aktor ada diatas tanah maka nilai jatuh dari aktor tersebut adalah *false*, sedangkan jika tidak ada diatas tanah maka nilai jatuh dari aktor tersebut adalah *true.* Perancangan method *cekJatuh* bisa dilihat pada gambar III.41.



Gambar III.41 Perancangan method *cekJatuh()*

C. Method loncat()

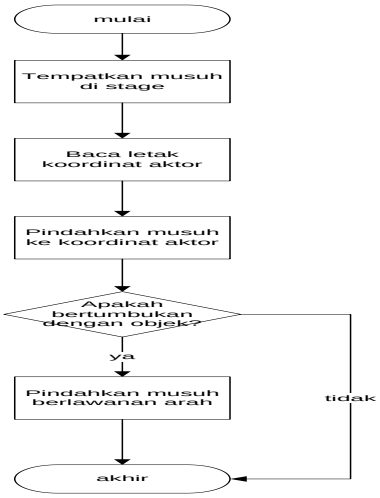
Method loncat merupakan method yang berfungsi untuk memberikan efek loncat pada aktor. Method ini bekerja ketika pengguna menekan tombol up pada keyboard dan program membacanya melalui method *checkkey()* yang telah dijelaskan pada sub bab sebelumnya. Setelah dibaca dengan menggunakn method *checkkey()* program memindahkan background ke sebelah bawah. Perancangan method loncat() bisa dilihat pada gambar III.29 berikut ini:



Gambar III. 42 Perancangan method loncat()

D. Method gerak()

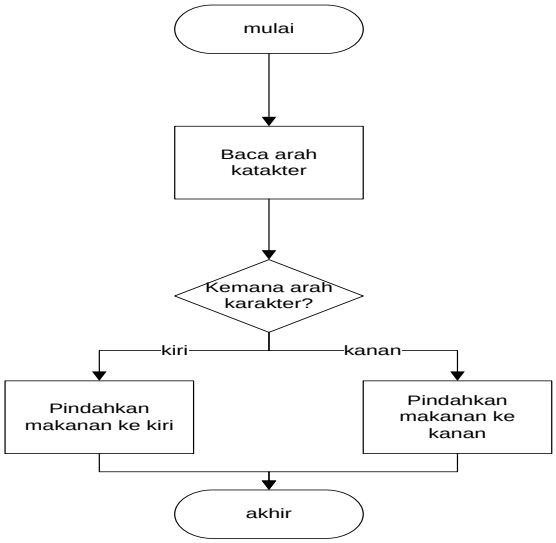
Method gerak() yang dijelaskan pada sub bab berikut ini merupakan method gerak yang ada pada *class* musuh. Method gerak ini berfungsi untuk memberikan efek hidup pada *game* dengan memberikan kemampuan kepada musuh untuk bergerak maju mundur maupun mendekati aktor.



Gambar III. 43 Perancangan method gerak

E. Method makanan()

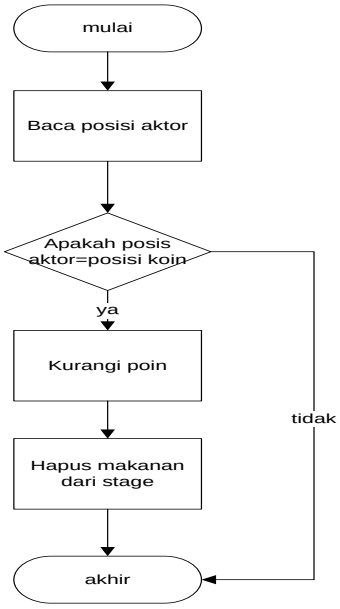
Method makanan() merupakan method yang memiliki kemampuan untuk memindahkan makanan yang dikeluarkan oleh musuh untuk mengarah ke kiri maupun ke kanan dari stage.



Gambar III.44 Perancangan method makanan

F. Method ambilMakanan()

Method yang akan dieksekusi ketika aktor berada tepat didepan makanan (dalam ini divisualkan dengan roti). Perancangan method dari ambil Makanan bisa dilihat pada gambar



Gambar III.45 Perancangan method ambil Makanan