BAB 3
ANALISIS DAN PERANCANGAN

3.1. Analisis Sistem

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

3.1.1 Analisis masalah

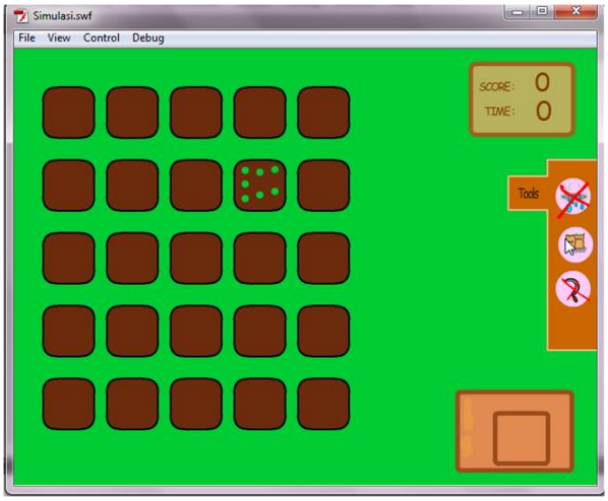
Tanaman sebagai sumber daya alam memiliki banyak sekali manfaatnya bagi kehidupan manusia, diantaranya adalah sebagai penyedia makanan dan sebagai bahan pengobatan tradisional. Untuk mendapatkan hasil yang maksimal dalam pemanfaatan tanaman maka diperlukan pemeliharaan yang tepat. Ada juga beberapa faktor yang dapat mempengaruhi pertumbuhan tanaman, diantaranya makanan dan lingkungan. Ada juga beberapa tanaman yang dapat diolah menjadi obat herbal. Sayangnya tidak sedikit masyarakat Indonesia yang kurang mengetahui tentang bercocok tanam dan jenis-jenis tanaman obat.

Perkembangan dunia teknologi saat ini memungkinkan bagi kehidupan masyarakat cepat mengenal suatu permainan menyenangkan dari sebuah *game* yang dapat dia mainkan. Baik itu melalui komputer atau *gadget* yang dimilikinya. Walaupun banyak yang berpendapat bahwa game itu memiliki dampak negatif, namun game bisa dimanfaatkan menjadi sarana edukasi. Game edukasi merupakan alat bantu yang sangat efektif dalam melakukan pengajaran jika penerapannya tepat.

3.1.2 Analisis *Game* Sejenis

Analisis game sejenis merupakan observasi dari game sejenis untuk melihat cara bermain, tujuan dan ada tidaknya kekurangan atau kelebihan game itu sendiri. Hasil observasi tersebut untuk membantu dalam penelitian.

Game Simulasi Menanam ini adalah game dengan genre simulasi. *Game* ini menggunakan *platform flash*(swf). Tampilan *game* simulasi menanam dapat dilihat pada gambar III.1



Gambar 3-1 Tampilan Game Simulasi Menanam

1. *Gameplay Game Simulasi Menanam*

   Beberapa aturan dalam *Game Simulasi Menanam* adalah sebagai berikut:

   a. Pemain dituntut untuk merawat tanaman hingga dewasa dan siap dipanen.
   b. Pemain akan disediakan bibit, air dan alat untuk memanen.
   c. Tujuan dari *game* ini adalah untuk memberikan pengetahuan kepada pemain tentang bagaimana cara menanam tanaman dari bentuk bibit hingga bisa dipanen.
   d. Pada awalnya pemain akan diberikan lahan untuk menanam.
   e. Setelah itu pemain akan mulai menanam benih yang telah disediakan.
   f. Tugas pemain untuk merawatnya adalah dengan memberikan air dan pupuk agar tanaman dapat tumbuh dengan baik.

g. Setelah tanaman berbuah maka pemain berhak untuk memanen hasil tanamannya.
h. *Score* akan bertambah jika pemain dapat memanen buah dengan tepat waktu, namun jika buah dari tanaman tidak segera diambil maka buah tersebut akan membusuk dan apabila buah yang telah busuk tersebut dipanen maka akan mengurangi *score*.
i. Pemain akan diberi waktu 50 detik untuk mendapatkan *score* setinggi-tingginya.

2. Kekurangan pada *game* tersebut yaitu :
   a. Hanya menonjolkan sisi menanam tanaman saja.
   b. Level/stage tidak ada.
   c. Gambar objek, layar kurang tertata rapi.
   d. Pemain hanya menggerakan alat yang digunakan untuk menanam tanaman.

Perbandingan game sejenis dengan game yang akan dikembangkan dijelaskan pada tabel berikut ini :

Tabel 3-1 Perbandingan game sejenis dengan game yang akan dibangun

No.	Perbandingan	Game Simulasi Menanam	Game yang akan dikembangkan
1.	Jenis Game	Simulasi	Simulasi
2.	Tema Game	Bercocok Tanam	Bercocok Tanam dan jenis-jenis tanaman obat
3.	Karakter	-	2 karakter utama
4.	Tampilan antar muka	2 dimensi	2 dimensi
5.	Tingkat kesulitan	1 Stage	3 Stage
6.	Platform	Desktop	Mobile

7.	Player	1 orang pengguna atau single player	1 orang pengguna atau single player

3.1.3 Analisis Game yang Akan Dibangun

3.1.3.1 Deskripsi

Game yang akan dibangun merupakan *game* edukasi yang berjeniskan *Simulation Game*. *Game* ini dibangun dengan mengaplikasikan teknologi sebagai jembatan untuk menyajikan materi mengenai cara bercocok tanam dan pengenalan jenis-jenis tanaman obat secara interaktif dan edukatif.. Pembangunan game ini dengan grafis 2D(2 Dimensi). *Game* ini terbagi kedalam objek-objek karena menggunakan pemrograman berorientasi objek. Deskripsi penerapan dalam game adalah sebagai berikut :

1. *Game* akan dimainkan dengan cara *tapping* yaitu teknik interaksi dengan menyentuh layar dengan satu jari lalu melepaskan kembali. Teknik ini sebagai pengganti fungsi klik pada mouse.
2. *Game* terdiri dari 2 stage.
3. Setiap stage memiliki kesulitan yang berbeda.
4. *Game* akan selalu disimpan ketika pemain berhasil memasuki ke stage kedua.
5. Pembelajaran terdiri dari cara bercocok tanam, kandungan dan manfaat tanaman obat.
6. *Game* bersifat bermain sambil belajar, karena dalam *game* disajikan materi-materi mengenai cara bercocok tanam dan pengenalan jenis-jenis tanaman obat.

3.1.3.2 Storyline

Game ini menceritakan perjalanan seorang anak laki-laki bernama Aji dan seorang anak perempuan bernama Siska berangkat untuk liburan dari rumahnya menuju rumah kakeknya. Ketika sampai dirumah kakeknya, pemain akan

membantu kakeknya untuk bercocok tanam dan belajar mengenal jenis-jenis tanaman yang bisa dimanfaatkan menjadi obat herbal. Pada setiap stage akan diberikan kesulitan yang berbeda.

3.1.3.3 *Game*play

Pada aplikasi *game* ini, pemain memiliki tugas untuk membantu sang kakek pada masing-masing stage. Pemain harus menyelesaikan stage pertama untuk dapat melanjutkan ke stage berikutnya. Bagi pemula, stage yang dapat dimainkan berawal dari stage 1. Sebelum masuk ke permainan, pemain dapat memilih menu belajar yaitu pengenalan tentang kandungan dan manfaat dari tanaman obat. Setiap game dimainkan dengan cara Tapping yaitu teknik interaksi dengan menyentuh layar dengan satu jari lalu melepaskannya kembali.

Pada game ini terdapat dua stage, stage tersebut adalah sebagai berikut :

1. Stage pertama

   Pada stage pertama pemain akan memakai karakter Siska untuk menjawab enam pertanyaan dari sepuluh pertanyaan yang diacak yang berhubungan dengan jenis-jenis tanaman obat. Tanaman yang akan menjadi pertanyaan pada stage pertama ini adalah lidah buaya, jahe merah, cengkeh, mengkudu, daun katuk, kaktus, daun dewa, zaitun, bunga matahari dan bunga melati. Pemain harus menjawab dengan benar pertanyaan-pertanyaan yang ada untuk dapat melanjutkan ke stage berikutnya.
2. Stage kedua

   Pada stage kedua pemain akan memakai karakter aji, pemain akan melakukan proses bercocok tanam pada lahan yang sudah disiapkan. Tanaman yang akan ditanam adalah tomat. Sebelum memasuki permainan, pemain akan dijelaskan tentang manfaat dari buah tomat. Rintangan yang dihadapi oleh pemain adalah menghindari tanaman menjadi rusak akibat hama tanaman.

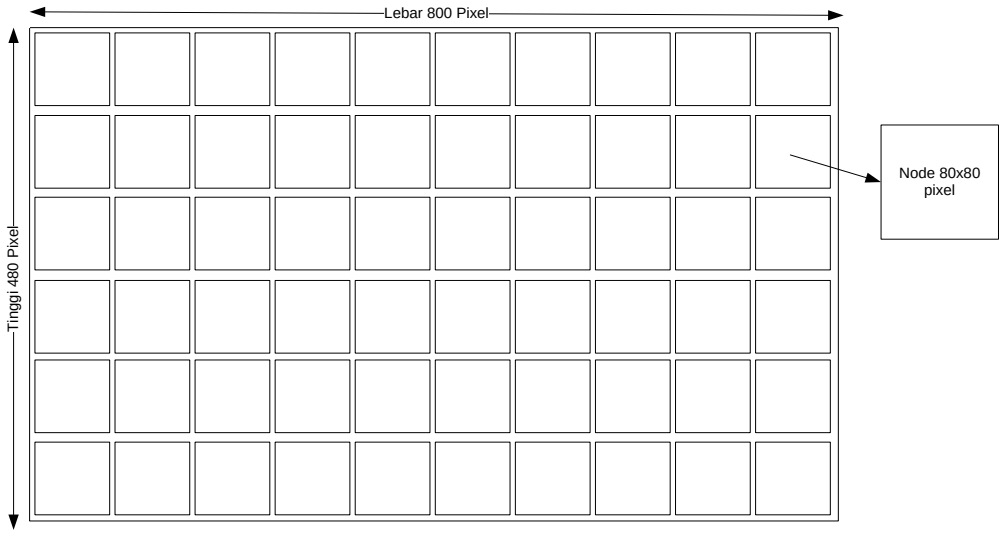
3.1.4 Analisis Algoritma

Algoritma yang akan diterapkan pada analisis pencarian jalur terpendek pada *game Berlibur Bersama Kakek* adalah Algoritma A*(*A Star*. Algoritma A* akan digunakan untuk mencari letak tanah yang akan ditanam oleh karakter.

3.1.4.1 Analisis Penggunaan Algoritma A*(*A Star)* terhadap pergerakan karakter

Algoritma A*(*A Star*) akan diterapkan pada analisis kasus pencarian jalan terpendek menuju tujuan untuk mencari jalur dengan cepat. Dalam penelitian ini, tujuan dari karakter berupa tanah-tanah yang akan ditanami oleh pemain dan menggerakan karakter menuju tanah yang akan dituju oleh pemain.

Room game Berlibur Bersama Kakek ini dibuat dengan skala *pixel* yang berukuran lebar 800 dan tinggi 480. Dari satu *Room* tersebut disimpan beberapa *node* (kotak-kotak kecil), masing-masing ukuran tinggi dan lebarnya sebesar 80 *pixel*, sebanyak 10 *node* horizontal dan 6 *node* vertikal.



Gambar 3-2 Keterangan ukuran room dan node pada stage 1

Algoritma A*(*A Star)* melakukan pencarian jalur terpendek dengan biaya yang diperhitungkan dari penjumlahan biaya sebenarnya ditambah dengan biaya perkiraan. Dalam notasi matematika algoritma ini dapat ditulis dengan rumus sebagai berikut :

$$f(n) = g(n) + h(n)$$

Keterangan :

f(n) = Fungsi evaluasi

g(n) = Biaya yang sudah dikeluarkan dari keadaan awal sampai keadaan n

h(n) = Estimasi biaya untuk sampai pada suatu tujuan mulai dari n

Dikarenakan menggunakan *manhattan distance* dalam mencari nilai *heuristic* maka pergerakan karakter didalam map akan bergerak secara horizontal dan vertikal. untuk menentukan *h(n)*, maka digunakan rumus *manhattan distance* sebagai berikut ini :

$$h(xy) = |x_1 - x_2|\ |y_1 - y_2|)$$

Keterangan :

h(xy) = nilai *heuristic* untuk simpul *xy*

x_1 = posisi koordinat x dari simpul x_1

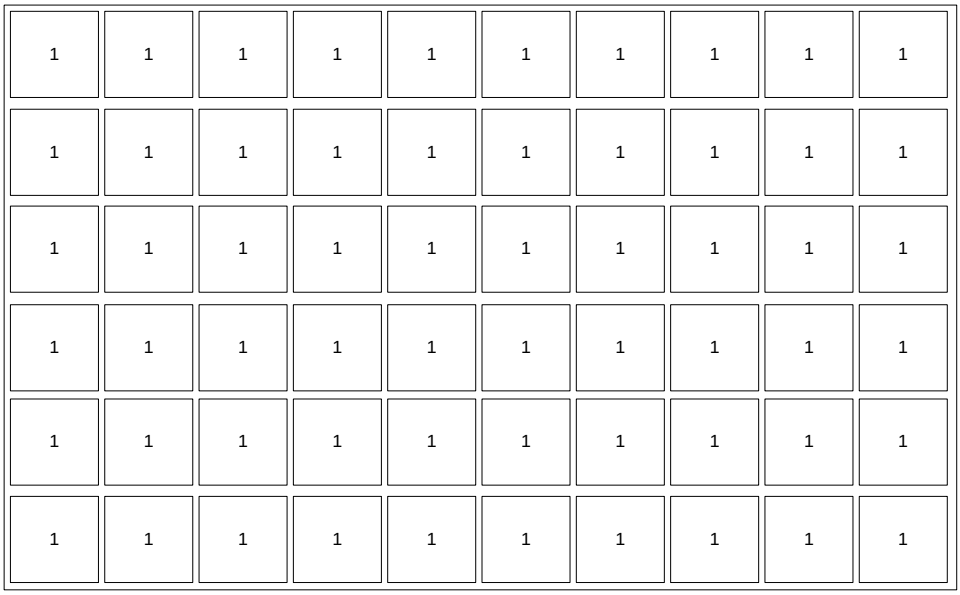
x_2 = posisi koordinat x tujuan

y_1 = posisi koordinat y dari simpul y_1

y_2 = posisi koordinat y tujuan

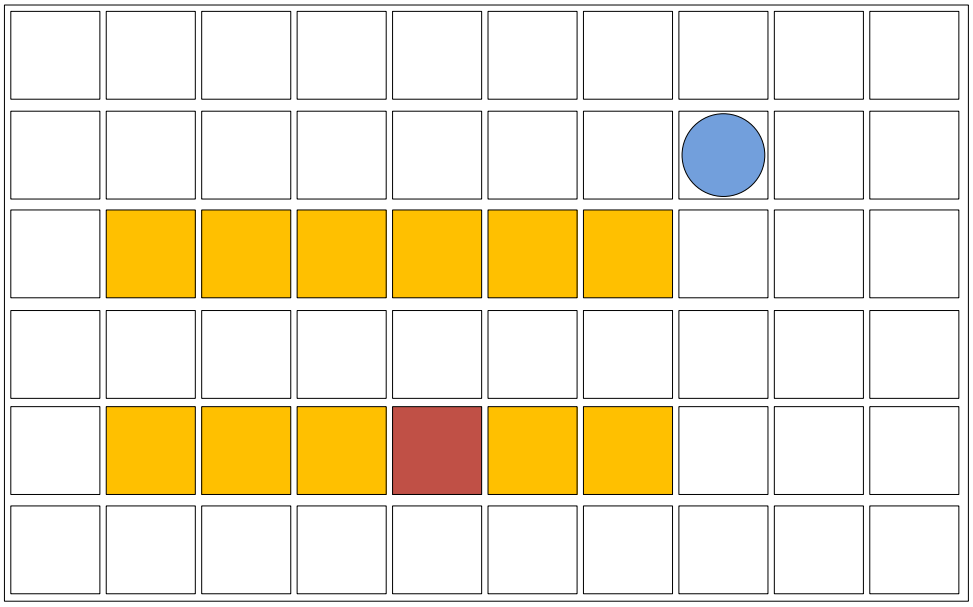
Dalam penelitian ini, untuk menentukan *cost* dari masing-masing node adalah 1. Dengan asumsi perpindahan dari node satu ke node lainnya satu kali

perpindahan. Maka untuk masing-masing node yang ada pada *room game Berlibur Bersama Kakek* ini memiliki *cost* sebesar 1.



Gambar 3-3 Nilai g(n) dari masing-masing node

Berikut ini adalah penjelasan penggunaan algoritma a* pada pergerakan karakter aji menuju tanah yang menjadi tujuannya.

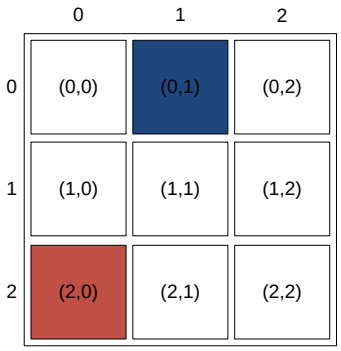


Gambar 3-4 Posisi karakter dan tanah pada game

Tabel 3-2 Keterangan pada gambar III-4

No	Gambar	Keterangan
1.		Lingkaran biru adalah node awal(posisi awal karakter aji)
2.		Kotak merah adalah node tujuan(tanah yang menjadi tujuan aji)
3.		Kotak kuning adalah obstacle(tanah yang bukan tujuan aji)

Untuk mencari nilai *h(n)*, maka akan digunakan rumus *manhattan distance* dengan contoh sebagai berikut :



Gambar 3-5 Contoh mencari nilai Heuristic menggunakan Manhattan Distance

Misalnya Node biru merupakan Node awal, node merah merupakan node tujuan, maka nilai *heuristic*nya sebagai berikut :

$h(n) = (abs(x_i(n)-x)+abs(y_i(n)-y))$

$h(n) = (abs(0-2)+abs(1-0))$

$h(n) = 3$

Dalam mencari nilai heuristik pada room stage 1 ini, ada nilai yang akan disederhanakan untuk mempermudah perhitungan. Karena ukuran dari *node* pada room berukuran 80x80 pixel maka akan disederhanakan menjadi 1x1.

	0	80	160	240	320	400	480	560	640	720
0	(0,0)	(80,0)	(160,0)	(240,0)	(320,0)	(400,0)	(480,0)	(560,0)	(640,0)	(720,0)
80	(0,80)	(80,80)	(160,80)	(240,80)	(320,80)	(400,80)	(480,80)	(560,80)	(640,80)	(720,80)
160	(0,160)	(80,160)	(160,160)	(240,160)	(320,160)	(400,160)	(480,160)	(560,160)	(640,160)	(720,160)
240	(0,240)	(80,240)	(160,240)	(240,240)	(320,240)	(400,240)	(480,240)	(560,240)	(640,240)	(720,240)
320	(0,320)	(80,320)	(160,320)	(240,320)	(320,320)	(400,320)	(480,320)	(560,320)	(640,320)	(720,320)
400	(0,400)	(80,400)	(160,400)	(240,400)	(320,400)	(400,400)	(480,400)	(560,400)	(640,400)	(720,400)

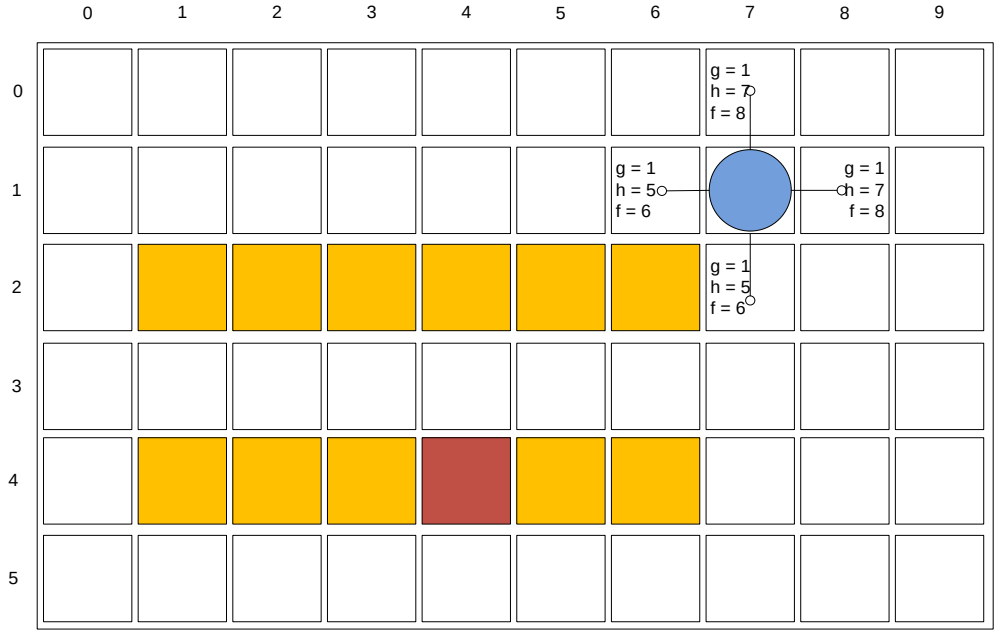
Gambar 3-6 Jarak x dan y pada room Berlibur Bersama Kakek sebelum di sederhanakan

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

Gambar 3-7 Jarak x dan y pada room Berlibur Bersama Kakek sesudah di sederhanakan

Dalam pencarian jalur terpendek menggunakan Algoritma A*(A Star) akan dijelaskan sebagai berikut :

1. Pada langkah pertama nilai dari node awal akan dimasukan ke *open list* dan akan menjadi *current node*.
2. Setelah dimasukan ke *open list*, periksa semua *neighbor node* atau *node* yang berdekatan dengan node awal apakah *passable* atau tidak.
3. kemudian tambahkan node yang *passable* ke *open list* dan node awal akan disimpan menjadi *parent node* yang nantinya akan digunakan untuk menelusuri jalan.



Gambar 3-8 Penjelasan langkah pertama

Pada langkah pertama, node (7,1) dimasukan ke dalam *open list* dan selanjutnya memeriksa *neighbor node* atau *node* yang berdekatan dengan *node* (7,1) yaitu *node* (7,2), (6,1), (7,0), (8,1). Dalam mencari nilai *f* dari *node* (7,2) menggunakan Algoritma A*(A Star) akan dijelaskan pada perhitungan sebagai berikut :

Neighbor Node (7,2)

neighbor.x = 7

neighbor.y = 2

Destination Node (4,4)

destination.x = 4

destination.x = 4

g(7,2) = 1 //*cost g* dari node awal ke node tetangga

h(7,2) = (abs(neighbor.x - destination.x) + abs(neighbor.y – destination.y)

h(7,2) = (abs(7 - 4) + abs(2 - 4))

h(7,2) = (abs(3) + abs(-2))

h(7,2) = 5

f(7,2) = *g*(7,2) + *h*(7,2)

f(7,2) = 1 + 5

f(7,2) = 6

Pada langkah pertama, setiap *neighbor node* akan dihitung menggunakan perhitungan ini hingga didapatkan nilai yang akan dijelaskan pada tabel III-3.

Tabel 3-3 Hasil Perhitungan langkah pertama

Node	Nilai g	Nilai h	Nilai f
(7,2)	1	5	6
(6,2)	1	5	6
(7,0)	1	7	8
(8,1)	1	7	8

Setelah diperiksa *node-node* tersebut akan dimasukan ke dalam *open list* lalu hapus *node* (7,1) dari open list, kemudian memasukan *node* (7,1) kedalam *closed list.* Karena terdapat dua nilai *f* yang paling rendah pada *open list*, sistem akan melakukan pengecekan ulang untuk mendapat *node* yang memiliki jalur

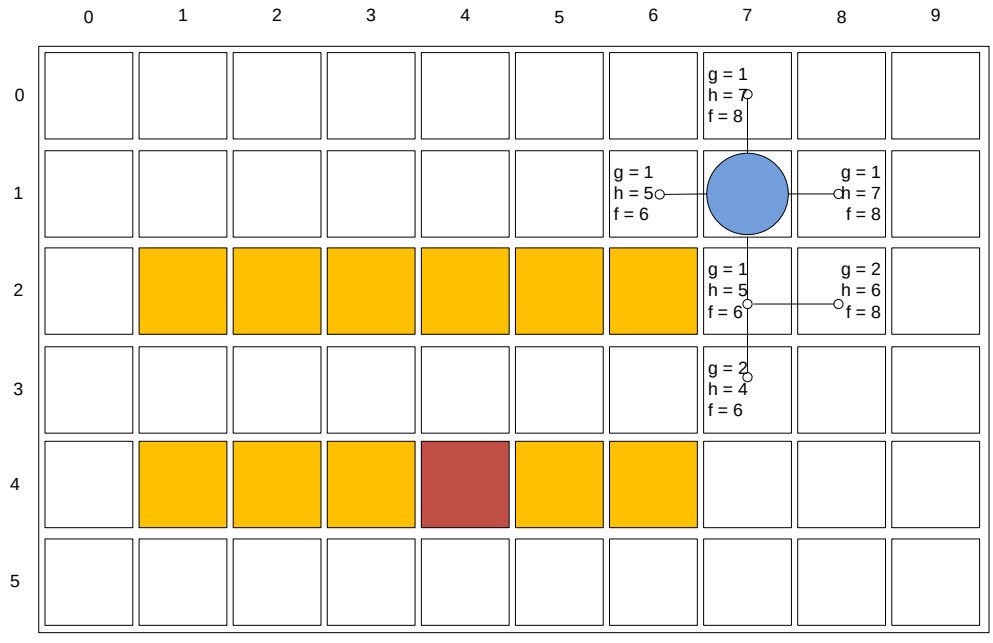
lebih cepat, maka akan dipilih *node* (7,2) sebagai nilai *f* terkecil untuk dijadikan *current node*.

Tabel 3-4 Tabel Perhitungan langkah pertama

Open List	(7,2), (6,1), (7,0), (8,1).
Closed List	(7,1)
Parent	(7,1)
Current Node	(7,2)
Destination Node	(4,4)

Proses tersebut akan dilakukan berulang-ulang hingga ditemukan jalur ke tujuan. Pada langkah kedua akan dijelaskan sebagai berikut :

1. Pada langkah kedua, hapus *current node* lalu masukan yang ada pada *open list* ke *closed list.*
2. Periksa *neighbor node* yang *passable* pada *current node*, jika tidak ada pada *open list*, tambahkan ke *open list* untuk diperiksa, lalu masukan *parent node* dari *current node* sebelumnya.



Gambar 3-9 Penjelasan langkah kedua

Pada langkah kedua, *node* (7,2) memiliki dua *neighbor* yaitu *node* (7,3) dan *node* (8,2). Kemudian lakukan perhitungan pada masing-masing *neighbor node* menggunakan Algoritma A*(A Star).

Tabel 3-5 Hasil perhitungan langkah kedua

Node	Nilai g	Nilai h	Nilai f
(7,3)	2	4	6
(6,2)	2	6	8

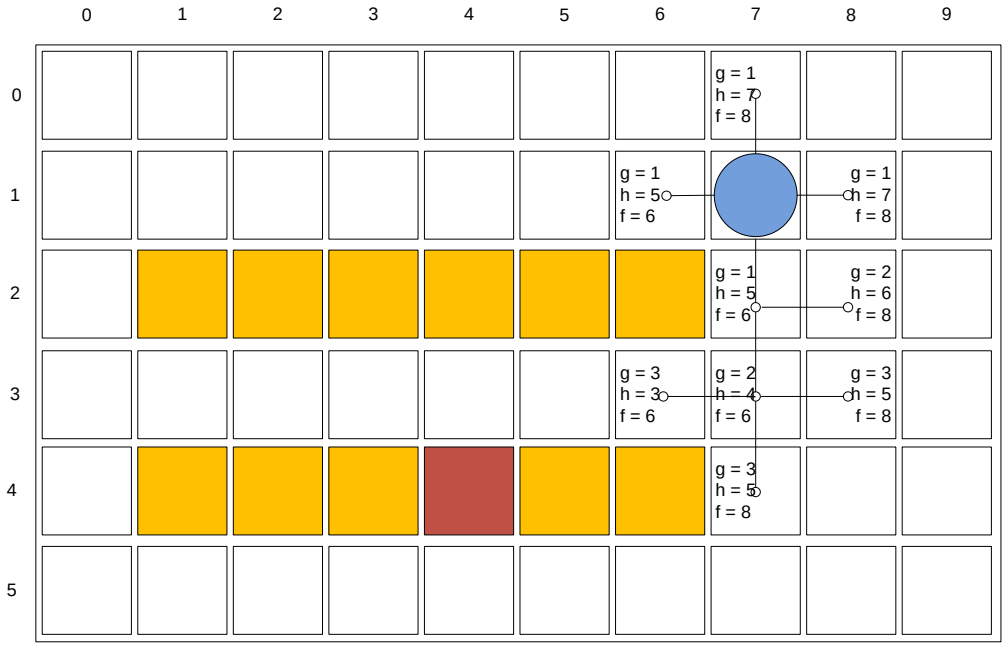
Setelah diperiksa *node-node* tersebut akan dimasukan ke dalam *open list* lalu hapus *node* (7,2) dari open list, kemudian memasukan *node* (7,2) kedalam *closed list.* Ambil nilai *f* terkecil untuk dijadikan *current node*.

Tabel 3-6 Tabel perhitungan langkah kedua

Open List	(7,3), (6,2)
Closed List	(7,1), (7,2)
Parent	(7,2)
Current Node	(7,3)
Destination Node	(4,4)

Pada langkah ketiga akan dijelaskan sebagai berikut :

1. Pada langkah ketiga, hapus *current node* lalu masukan yang ada pada *open list* ke *closed list.*
2. Periksa *neighbor node* yang *passable* pada *current node*, jika tidak ada pada *open list*, tambahkan ke *open list* untuk diperiksa, lalu masukan *parent node* dari *current node* sebelumnya.



Gambar 3-10 Penjelasan langkah ketiga

Pada langkah ketiga, *node* (7,3) memiliki tiga *neighbor* yaitu *node* (7,4), (6,3),(8,3). Kemudian lakukan perhitungan pada masing-masing *neighbor node* menggunakan Algoritma A*(A Star).

Tabel 3-7 Hasil perhitungan langkah ketiga

Node	Nilai g	Nilai h	Nilai f
(7,4)	3	5	8
(6,3)	3	3	6
(8,3)	3	5	8

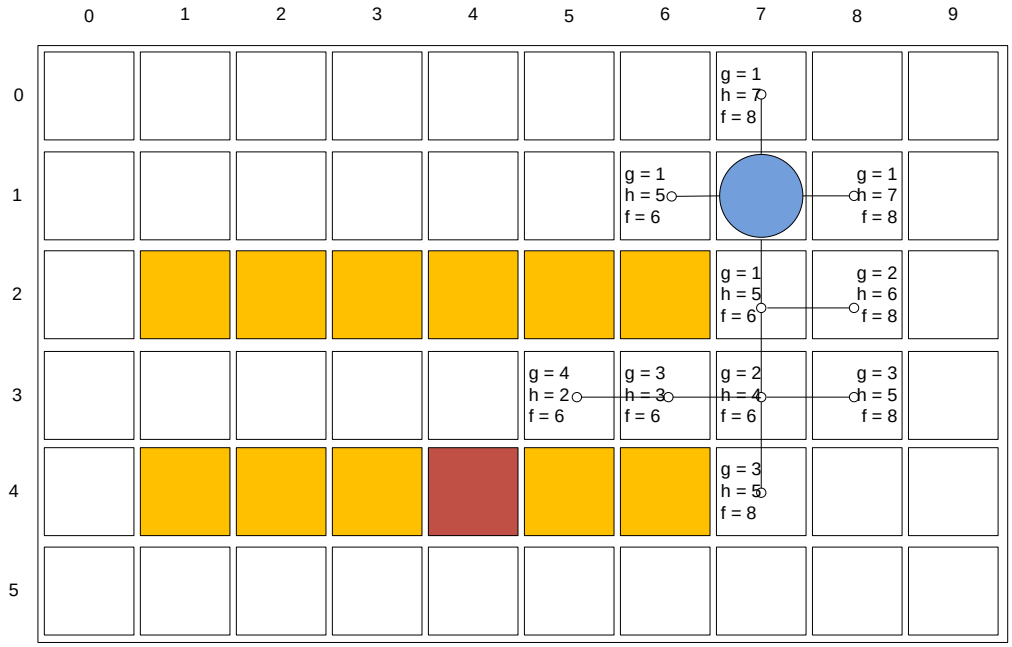
Setelah diperiksa *node-node* tersebut akan dimasukan ke dalam *open list* lalu hapus *node* (7,3) dari open list, kemudian memasukan *node* (7,3) kedalam *closed list.* Ambil nilai *f* terkecil untuk dijadikan *current node* untuk diperiksa hingga ditemukan *node* tujuan.

Tabel 3-8 Tabel perhitungan langkah ketiga

Open List	(7,4),(6,3),(8,3)
Closed List	(7,1),(7,2),(7,3)
Parent	(7,3)
Current Node	(6,3)
Destination Node	(4,4)

Pada langkah keempat akan dijelaskan sebagai berikut :

1. Pada langkah keempat, hapus *current node* lalu masukan yang ada pada *open list* ke *closed list.*
2. Periksa *neighbor node* yang *passable* pada *current node*, jika tidak ada pada *open list*, tambahkan ke *open list* untuk diperiksa, lalu masukan *parent node* dari *current node* sebelumnya.



Gambar 3-11 Penjelasan langkah keempat

Pada langkah keempat, *node* (6,3) memiliki satu *neighbor* yaitu *node* (5,3). Kemudian lakukan perhitungan pada masing-masing *neighbor node* menggunakan Algoritma A*(A Star).

Tabel 3-9 Hasil perhitungan langkah keempat

Node	Nilai g	Nilai h	Nilai f
(5,3)	4	2	6

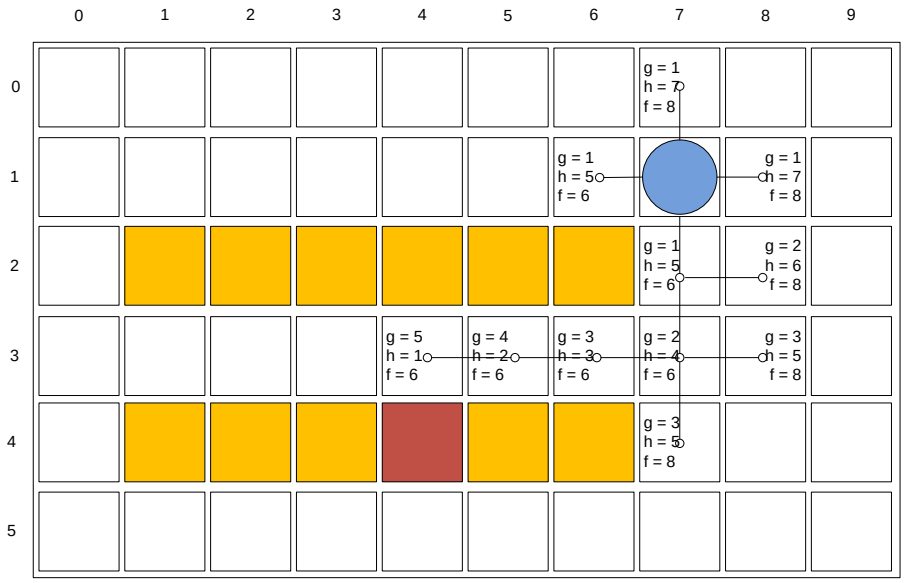
Setelah diperiksa *node* tersebut akan dimasukan ke dalam *open list* lalu hapus *node* (6,3) dari open list, kemudian memasukan *node* (6,3) kedalam *closed list.* Karena hanya ada satu nilai *f* terkecil, maka *node* tersebut dijadikan *current node* untuk diperiksa hingga ditemukan *node* tujuan.

Tabel 3-10 Tabel perhitungan langkah keempat

Open List	(5,3)
Closed List	(7,1),(7,2),(7,3),(6,3)
Parent	(6,3)
Current Node	(5,3)
Destination Node	(4,4)

Pada langkah kelima akan dijelaskan sebagai berikut :

1. Pada langkah kelima, hapus *current node* lalu masukan yang ada pada *open list* ke *closed list.*
2. Periksa *neighbor node* yang *passable* pada *current node*, jika tidak ada pada *open list*, tambahkan ke *open list* untuk diperiksa, lalu masukan *parent node* dari *current node* sebelumnya.



Gambar 3-12 Penjelasan langkah kelima

Pada langkah kelima, *node* (5,3) memiliki satu *neighbor* yaitu *node* (4,3). Kemudian lakukan perhitungan pada masing-masing *neighbor node* menggunakan Algoritma A*(A Star).

Tabel 3-11 Hasil perhitungan langkah kelima

Node	Nilai g	Nilai h	Nilai f
(4,3)	5	1	6

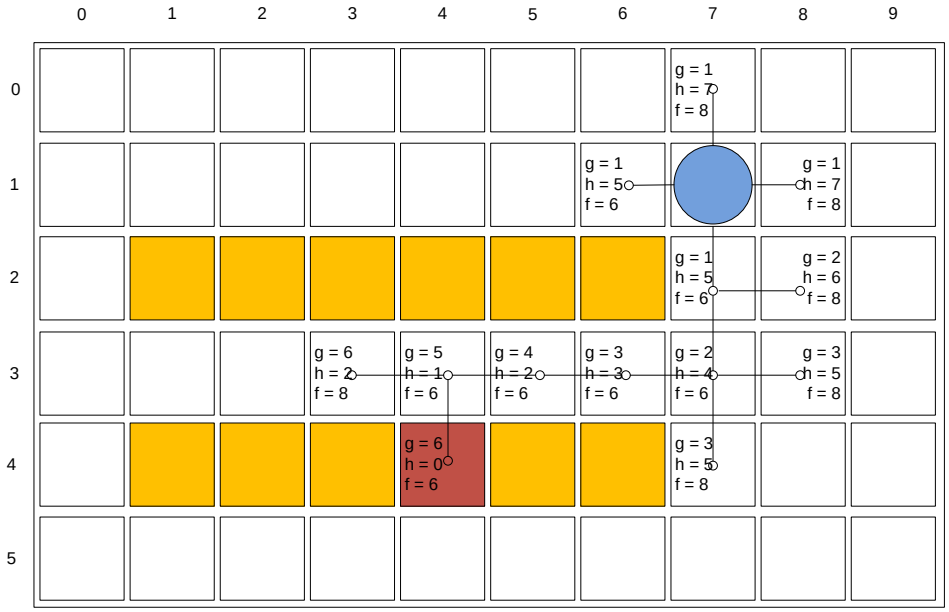
Setelah diperiksa *node* tersebut akan dimasukan ke dalam *open list* lalu hapus *node* (5,3) dari open list, kemudian memasukan *node* (5,3) kedalam *closed list.* Karena hanya ada satu nilai *f* terkecil, maka *node* tersebut dijadikan *current node* untuk diperiksa hingga ditemukan *node* tujuan.

Tabel 3-12 Tabel perhitungan langkah kelima

Open List	(4,3)
Closed List	(7,1),(7,2),(7,3),(6,3),(5,3)
Parent	(5,3)
Current Node	(4,3)
Destination Node	(4,4)

Pada langkah keenam akan dijelaskan sebagai berikut :

1. Pada langkah keenam, hapus *current node* lalu masukan yang ada pada *open list* ke *closed list.*
2. Periksa *neighbor node* yang *passable* pada *current node*, jika tidak ada pada *open list*, tambahkan ke *open list* untuk diperiksa, lalu masukan *parent node* dari *current node* sebelumnya.



Gambar 3-13 Penjelasan Langkah keenam

Pada langkah keenam, *node* (4,3) memiliki dua *neighbor* yaitu *node* (4,4) dan *node* (3,3). Kemudian lakukan perhitungan pada masing-masing *neighbor node* menggunakan Algoritma A*(A Star).

Tabel 3-13 Hasil perhitungan langkah keenam

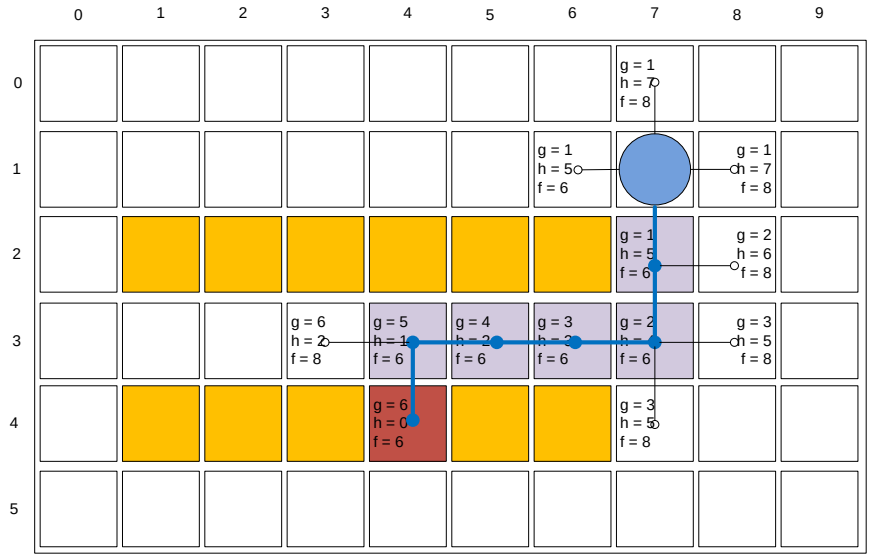
Node	Nilai g	Nilai h	Nilai f
(4,4)	6	0	6
(3,4)	6	2	8

Setelah diperiksa *node* tersebut akan dimasukan ke dalam *open list* lalu hapus *node* (4,3) dari open list dan telah menemukan jalur terpendek dari *node* awal ke node tujuan karena *current node* sama dengan *destination node*.

Tabel 3-14 Tabel perhitungan langkah keenam

Open List	(4,4),(3,3)
Closed List	(7,1),(7,2),(7,3),(6,3),(5,3),(4,3)
Parent	(4,3)
Current Node	(4,4)
Destination Node	(4,4)

Maka hasil dari pencarian jalan terpendek menggunakan Algoritma A*(A Star), karakter akan bergerak melalui *node* (7,1),(7,2),(7,3),(6,3),(5,3),(4,3) untuk sampai ke tanah yang menjadi tujuannya akan dijelaskan pada gambar III-14



Gambar 3-14 Hasil dari pencarian jalur terpendek menggunakan Algoritma A*(A Star)

Jalan yang akan dilewati karakter untuk mencapai tujuan akhir adalah sebagai berikut :

Tabel 3-15 Jalan yang akan dilewati oleh karakter aji untuk mencapai tujuan

Jalur	(7,1),(7,2),(7,3),(6,3),(5,3),(4,3)

3.1.4.2 *Pseudocode* Algoritma A*

Berikut adalah pseudocode Algoritma A*

```
Deklarasi
node awal : real
node tujuan : real
openlist  : array
closelist : array
nilai G : int
temp : boolean
Algoritma
Cari nilai f pada node awal
Masukan nilai f pada node awal ke openlist
do
      Cari nilai f terkecil pada openlist
      Hapus nilai node sekarang yang ada pada openlist
      Memasukan nilai node sekarang ke closedlist
```

```
      For pengecekan node sebanyak node tetangga
           Cari nilai node tetangga yang belum diperiksa
           If node tetangga bisa dilewati
               If node tetangga tidak ada pada openlist
               Cari nilai f dari node tetangga
               Memasukan nilai node tetangga ke openlist
           endif
           else
           memasukan nilai baru dari G
               if nilai G lebih kecil dari nilai G node tetangga
                  menghitung kembali nilai nilai G node tetangga
                  menghitung nilai f node tetangga
               endif
           endelse
       endfor
until langkah hingga node tujuan masuk ke closed list atau tidak ditemukan jalur lagi

if node tujuan tidak ditemukan
    node terdekat sama dengan node awal
    for pengecekan setiap node yang sudah diperiksa
        pilih node kandidat dari closed list
        cari nilai f dari kandidat
        if kandidat lebih kecil dari node sekarang
           node sekarang sama dengan node kandidat
        endif
    node tujuan sama dengan node sekarang
endif
```

3.1.5 Analisis Kebutuhan Non Fungsional

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

3.1.5.1 Analisis Kebutuhan Perangkat Lunak

Analisis kebutuhan perangkat lunak pada game ini terbagi menjadi dua bagian yaitu analisis kebutuhan perangkat lunak untuk pengembang dan analisis keutuhan perangkat lunak untuk pengguna.

1. Analisis kebutuhan perangkat lunak untuk pengembang

   Perangkat lunak yang digunakan dalam pengembangan aplikasi game ini adalah sebagai berikut :

1. Sistem Operasi Windows 8 4 bit
2. Gamemaker:Studio
3. Android SDK
4. Adobe Photoshop
5. Corel Draw

2. Analisis kebutuhan perangkat lunak untuk pengguna

   Perangkat lunak yang dibutuhkan pengguna dalam memainkan aplikasi game ini adalah sebagai berikut :

   a. Sistem Operasi Android 2.3.3 ke atas

3.1.5.2 Analisis Kebutuhan Perangkat Keras

Analisis kebutuhan perangkat keras pada *game* ini terbagi menjadi dua bagian yaitu analisis kebutuhan perangkat keras untuk pengembang dan analisis kebutuhan perangkat keras untuk penguna.

1. Analisis Kebutuhan Perangkat Keras untuk Pengembang.

   Pada aplikasi ini, analisis perangkat perangkat keras yang digunakan pengemban adalah sebagai berikut :

   a. Processor 1.80GHz
   b. RAM 4GB
   c. VGA 2GB
   d. Harddisk 500GB
   e. Monitor
   f. Mouse dan Keyboard

2. Analisis Kebutuhan Perangkat Keras untuk Pengguna.

   Perangkat keras yang dibutuhkan pengguna dalam memainkan aplikasi game adalah *smartphone*. Spesifikasi *smartphone* yang diperlukan untuk memainkan *game* ini adalah sebagai berikut :

   a. Sistem Operasi Android 2.3.3 keatas
   b. Processor 1,00Ghz
   c. Kapasitas RAM 256MB

d. Kapasitas penyimpanan 500MB

3.1.5.3 Analisis dan Kebutuhan Perangkat Pikir

Analisis kebutuhan perangkat pikir dilakukan untuk mengetahui kemampuan *player*(pemain) yang dapat memainkan game ini. Analisis kebutuhan perangkat pikir digambarkan dalam Tabel kebutuhan *player*. Kebutuhan *player* dijabarkan pada tabel berikut ini.

Tabel 3-16 Tabel Kebutuhan Player

Karakteristik Player	Keterangan
Computer literacy	Dapat menggunakan smartphone berbasis Android
Age	Semua umur
Application Experience	Tidak dibutuhkan
Gender	Pria dan wanita
Disabilities	Buta total dan buta warna

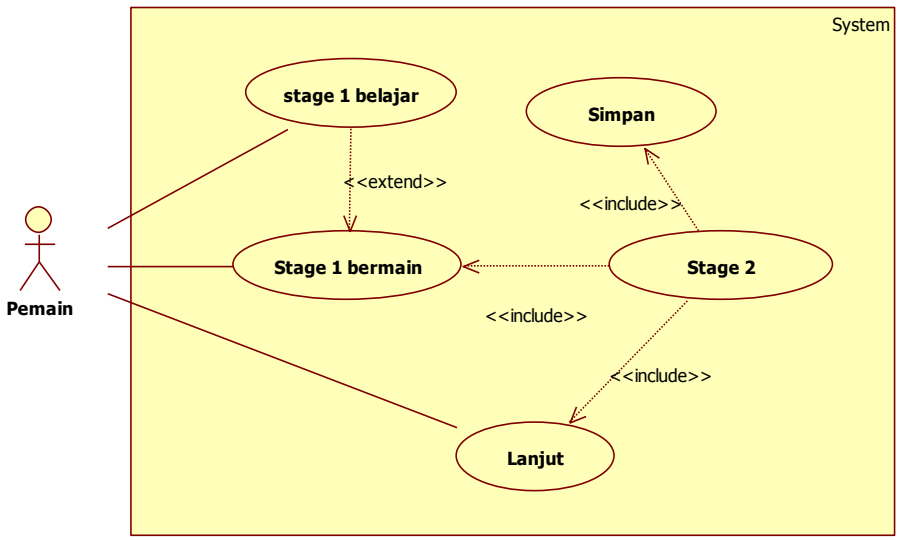
3.1.6 Analisis Kebutuhan Fungsional

Analisis kebutuhan fungsional merupakan proses penggambaran, perencanaan dan pembuatan sketsa atau pengaturan dari beberapa elemen yang terpisah ke dalam satu kesatuan yang utuh dan berfungsi. Alat bantu untuk menggambarkan sistem menggunakan *Unified Modeling Language*(UML), dimana tahapan-tahapan yang terdapat pada *game* ini adalah *Use Case Diagram, Class Diagram* dan *Sequence Diagram.*

3.1.6.1 Use Case Diagram

Use Case atau diagram *use case* merupakan pemodelan untuk kelakuan (*behavior*) sistem informasi yang akan dibuat. *Use case* mendeskripsikan sebuah interaksi antara satu atau lebih aktor dengan sistem informasi yang akan dibuat.

Perancangan proses-proses yang terdapat pada *game* Berlibur Bersama Kakek, yang digambarkan dengan *use case diagram* pada gambar III.15



Gambar 3-15 Use Case Diagram

3.1.6.2 Definisi Aktor

Definisi Aktor berfungsi untuk menjelaskan *actor* yang terlibat pada *use case diagram*. Berikut ini adalah tabel III.3 yang menerangkan definisi *actor*.

Tabel 3-17 Definisi Aktor

No.	Aktor	Deskripsi
1.	Pemain	Merupakan orang yang akan memainkan game.

3.1.6.3 Definisi *Use* Case

Definisi *use case* menjelaskan fungsi *use case* yang terdapat pada *use case diagram*. Berikut ini adalah tabel III.4 menerangkan definisi *use case*.

Tabel 3-18 Definisi Use Case

No.	Use Case	Deskripsi
1.	Stage 1 belajar	Proses pembelajaran pada stage 1
2.	Stage 1 bermain	Proses permainan pada stage 1
3.	Stage 2	Proses permainan pada stage 2
4.	Lanjut	Proses untuk melanjutkan permainan yang sebelumnya sudah disimpan
5.	Simpan	Proses untuk menyimpan permainan

3.1.6.4 Skenario Use Case

Skenario *use case* menggambarkan alur penggunaan sistem dimana setiap skenario digambarkan dari sudut pandang aktor, seseorang atau peranti yang berinteraksi dengan perangkat lunak dalam berbagai cara.

Tabel 3-19 Skenario Use Case Stage 1 Belajar

Identifikasi	
Nomor	1.
Nama	Stage 1 belajar
Tujuan	Mulai pembelajaran pada stage 1 belajar
Deskripsi	Proses untuk memulai pembelajaran pada stage 1 belajar
Aktor	Pemain
Trigger	☐ internal ☒ Eksternal
Skenario Utama	
Kondisi Awal	Pemain berada di tampilan stage 1 belajar

Aksi Aktor	Reaksi Sistem
1. Memilih stage1 belajar	2. Menampilkan stage1 belajar
3. Memilih tanaman	4. Menampilkan info tanaman
5. Menekan kembali	6. Menampilkan stage 1 belajar
7. Menekan tombol lanjut	8. Menampilkan tampilan stage 1 bermain
9. Menekan kembali	10. Menampilkan tampilan menu utama
Skenario Tambahan	
Kondisi Awal	Pemain berada di pesan keluar dari permainan
1. Menekan tombol Ya	2. Menampilkan Menu Utama
3. Menekan tombol Tidak	4. Menampilkan Tampilan Stage 1 belajar

Tabel 3-20 Skenario Use Case Stage 1 Bermain

Identifikasi	
Nomor	2.
Nama	Stage 1 bermain
Tujuan	Mulai permainan pada stage 1 bermain
Deskripsi	Proses untuk memulai permainan pada stage 1 bermain
Aktor	Pemain
Trigger	☐internal ☒ Eksternal
Skenario Utama	
Kondisi Awal	Pemain berada di tampilan stage 1 bermain
Aksi Aktor	Reaksi Sistem
1. Memilih stage1	2. Menampilkan stage1

3. Memilih tampilkan pertanyaan	4. Mengacak pertanyaan
	5. Menampilkan pertanyaan
6. Memilih jawaban	7. Cek jawaban
	8. Cek pertanyaan
	9. Jika pertanyaan belum terjawab semua maka menampilkan pertanyaan berikutnya
	10. Jika pertanyaan sudah terjawab maka stage 1 telah selesai
	11. Menampilkan pesan berhasil
12. Menekan tombol lanjut	13. Menampilkan tampilan stage 2
Skenario Tambahan	
Kondisi Awal	Pemain berada di pesan keluar dari permainan
1. Menekan tombol Ya	2. Menampilkan Menu Utama
3. Menekan tombol Tidak	4. Menampilkan Tampilan Stage 1 bermain

Tabel 3-21 Skenario Use Case Stage 2

Identifikasi	
Nomor	3.
Nama	Stage 2
Tujuan	Memainkan stage 2
Deskripsi	Proses permainan pada stage 2
Aktor	Pemain

Trigger	☐internal ☒ Eksternal
Skenario Utama	
Kondisi Awal	Pemain berada di halaman menu utama

Aksi Aktor	Reaksi Sistem
1. Memulai stage 2	2. Menampilkan tampilan stage 2
3. Memilih tanah	4. Karakter bergerak menuju tanah yang dipilih
5. Memilih Alat-alat yang akan digunakan dengan memilih tangan, cangkul, bibit, penyiram tanaman, pot dan pestisida	6. Jika memilih tangan, maka dapat mengambil hasil dari tanaman
	7. Jika memilih cangkul, maka dapat mencangkul tanah untuk dapat ditanam
	8. Jika memilih bibit, maka dapat menanam tanaman dengan bibit ke tanah
	9. Jika memilih penyiram tanaman, maka dapat menyiram tanaman yang sedang ditanam
	10. Jika memilih pestisida, maka dapat menyemprotkan pestisida ke tanaman
11. Memilih keranjang	12. Karakter bergerak menuju keranjang dan dapat menyimpan hasil tanaman

Skenario Tambahan	
Kondisi Awal	Pemain berada di pesan keluar dari permainan
1. Menekan tombol Ya	2. Menampilkan Menu Utama
3. Menekan tombol Tidak	4. Menampilkan Tampilan Stage 2

Tabel 3-22 Skenario Lanjut

Identifikasi	
Nomor	4.
Nama	Lanjut
Tujuan	Melanjutkan permainan
Deskripsi	Proses untuk melanjutkan permainan yang sudah disimpan sebelumnya
Aktor	Pemain
Trigger	☐ internal ☒ Eksternal
Skenario Utama	
Kondisi Awal	Pemain berada di tampilan menu
Aksi Aktor	Reaksi Sistem
1. Memilih lanjut	2. Menampilkan stage permainan sesuai yang disimpan

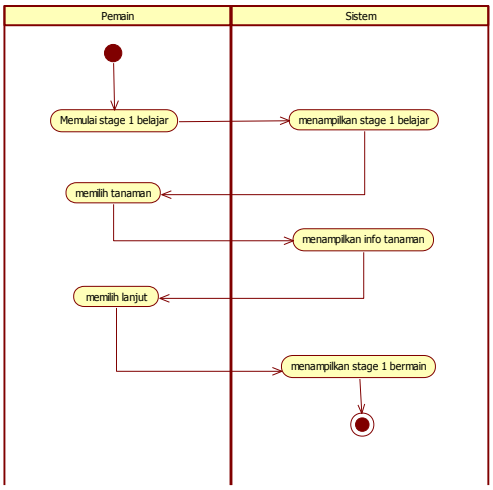
Tabel 3-23 Skenario Simpan

Identifikasi	
Nomor	5.
Nama	Simpan
Tujuan	Menyimpan permainan

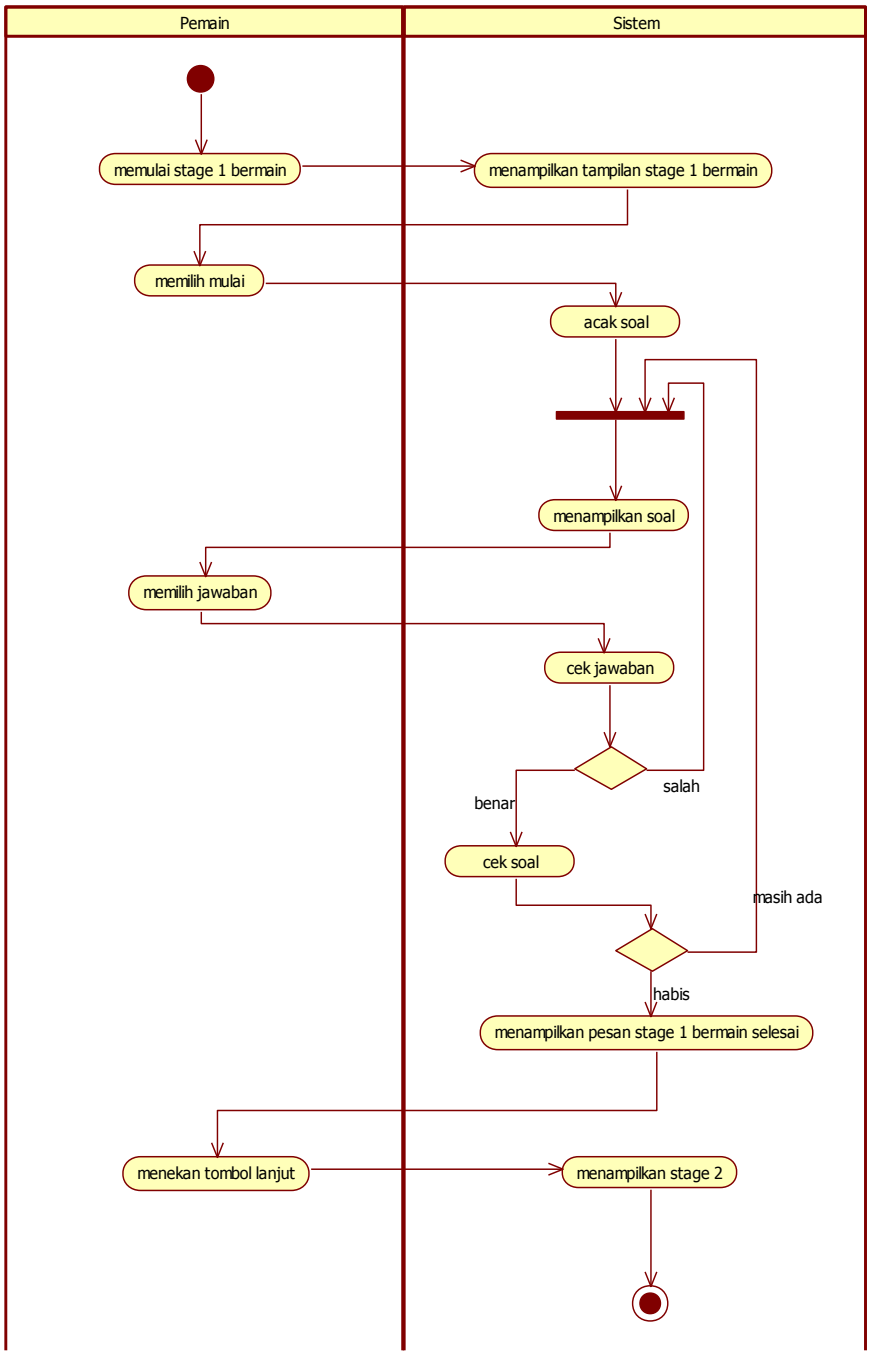
Deskripsi	Proses untuk menyimpan permainan
Aktor	Pemain
Trigger	☐internal ☒Eksternal
Skenario Utama	
Kondisi Awal	Pemain berada di tampilan stage yang dimainkan
Aksi Aktor	Reaksi Sistem
1. Memasuki stage yang dimainkan	2. Menyimpan permainan

3.1.6.5 Activity diagram

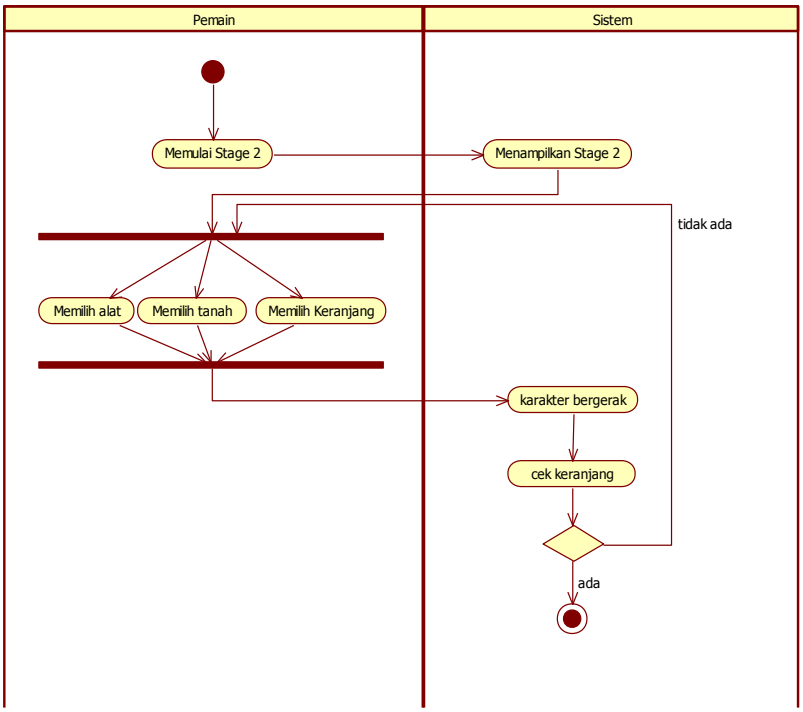
Diagram aktivitas atau *activity diagram* menggambarkan *workflow* (aliran kerja) atau aktivitas dari sebuah sistem atau proses bisnis. Dalam diagram aktivitas yang terlibat adalah antara *user* dan sistem. Terdapat tujuh aktivitas yang tergambar sesuai prosesnya masing-masing yaitu *activity diagram* stage 1 bermain, *activity diagram* stage 1 belajar, *activity diagram* stage 2, *activity diagram* lanjut, *activity diagram* simpan.



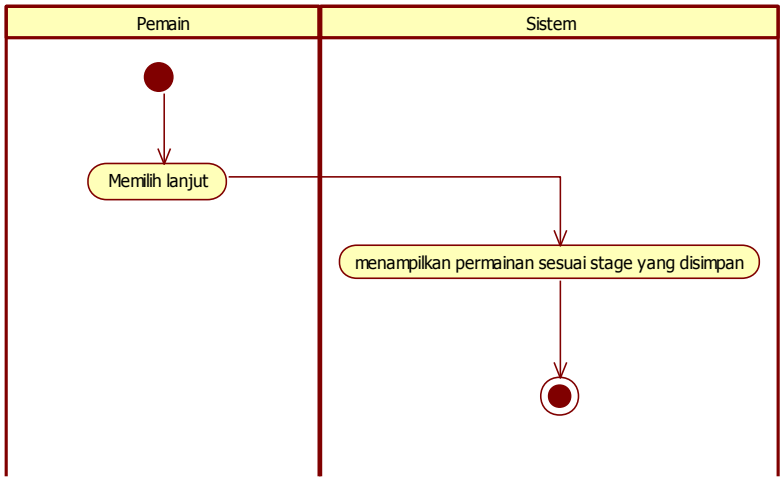
Gambar 3-16 Activity Diagram Stage 1 Belajar



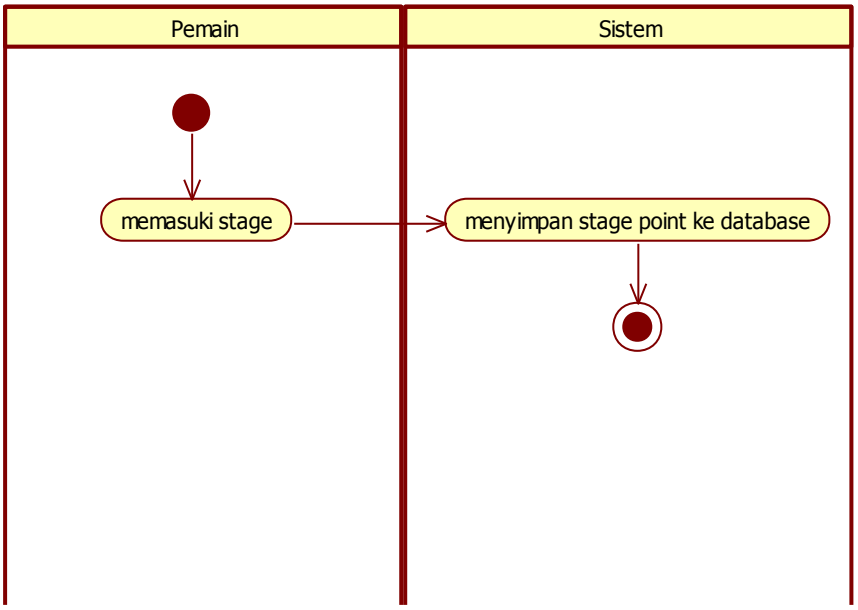
Gambar 3-17 Activity Diagram Stage 1 Bermain



Gambar 3-18 Activiry Diagram Stage 2



Gambar 3-19 Activity Diargam Lanjut permainan



Gambar 3-20 Activity Diagram Simpan

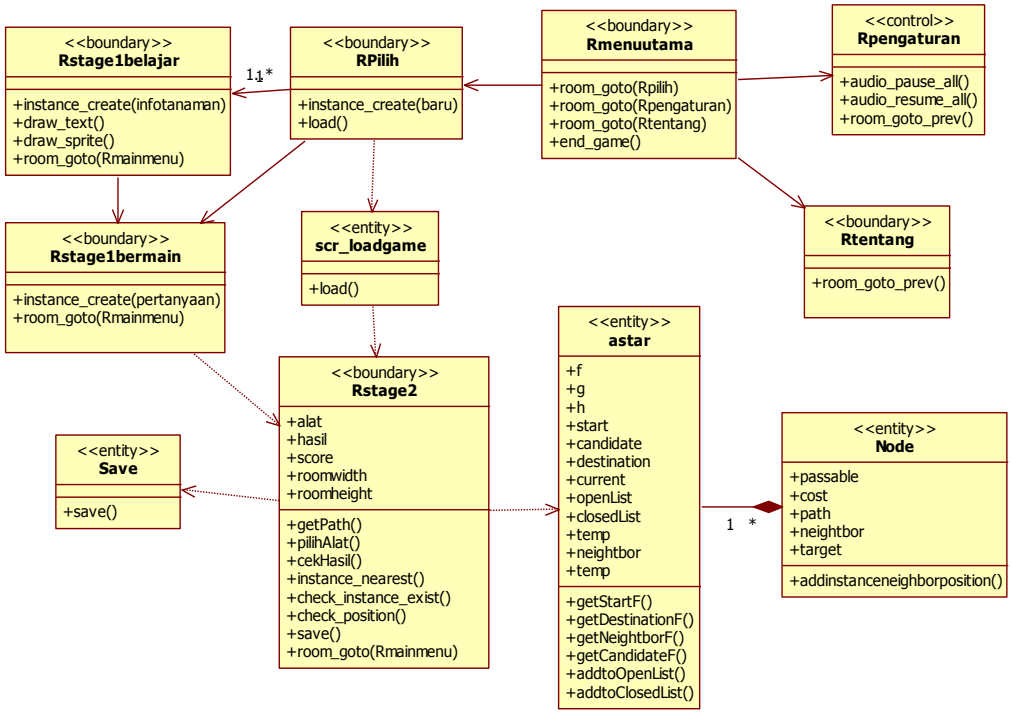
3.1.6.6 Class Diagram

Diagram kelas atau *class diagram* menggambarkan struktur sistem dari segi pendefinisian kelas-kelas yang akan dibuat untuk membangun sistem. Kelas memiliki apa yang disebut atributdan metode atau operasi. *Class diagram* ditunjukan pada Gambar III-29. Pendefinisian kelas ditunjukan pada Tabel III-12.

Tabel 3-24 Tabel pendefinisian kelas

Jenis kelas	Nama Kelas
Boundary	Rmenuutama
	Rtentang
	Rpilih
	Rstage1bermain
	Rstage1belajar
	Rstage2

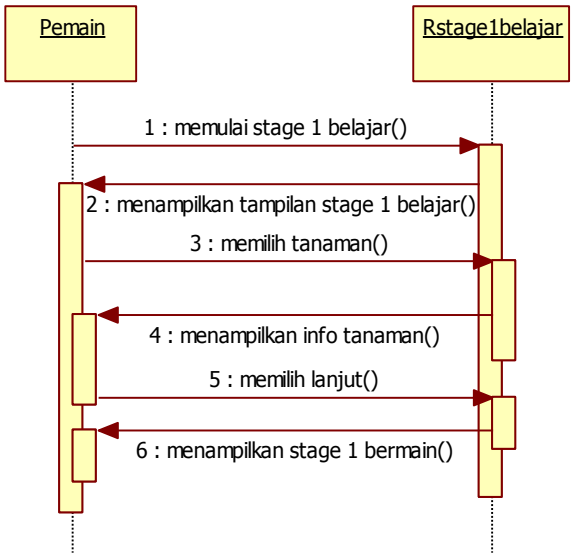
Entity	Astar
	Node
	Scr_loadgame
	Save
Control	Pengaturan



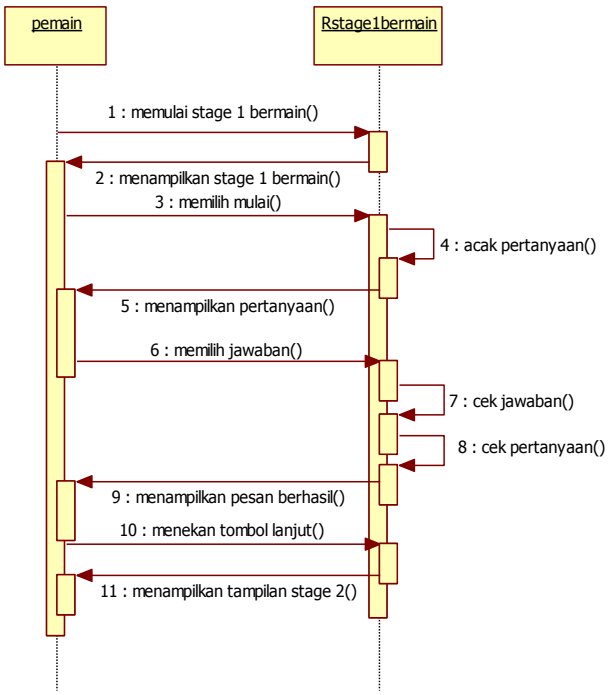
Gambar 3-21 Class Diagram

3.1.6.7 Sequence diagram

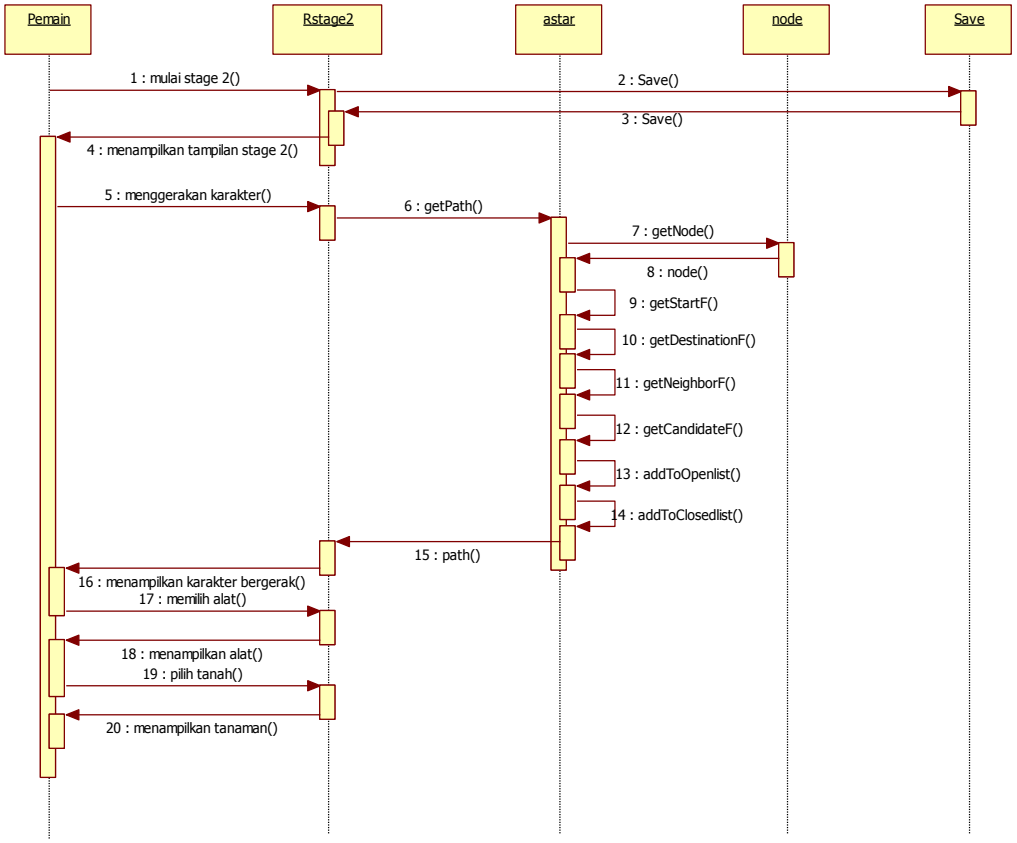
Diagram sekuen atau *sequence diagram* menggambarkan kelakuan objek pada *use case* dengan mendeskripsikan waktu hidup objek dan *message* yang dikirimkan dan diterima antar objek.



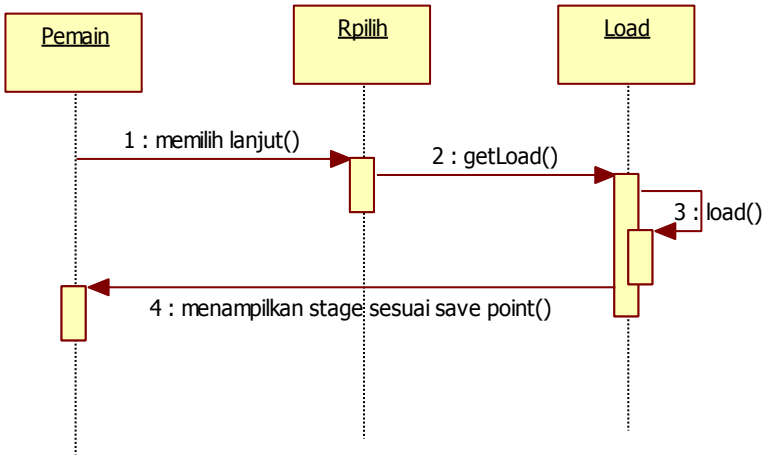
Gambar 3-22 Sequence Diagram Stage 1 Belajar



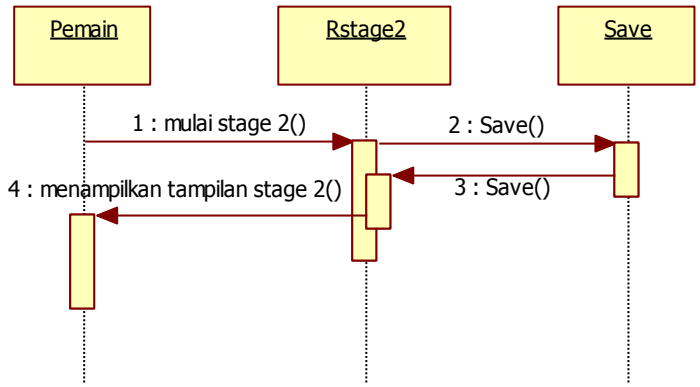
Gambar 3-23 Sequence Diagram Stage 1 Bermain



Gambar 3-24 Sequence Diagram Stage 2



Gambar 3-25 Sequence Diagram Lanjut



Gambar 3-26 Sequence Diagram Simpan

3.2. Perancangan Sistem

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

Perancangan *game* ini meliputi perancangan struktur menu, perancangan antarmuka, perancangan pesan, jaringan semantik, semua perancangan yang telah disebutkan diatas harus saling berhubungan agar menciptakan suatu sistem yang baik.

3.2.1 Perancangan Komponen Permainan

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

3.2.1.1 Karakter dan Alat-Alat Bercocok Tanam

Perancangan karakter dan alat-alat bercocok tanam adalah bagian dimana karakter dan alat-alat bercocok tanam pada *game* dijelaskan ciri-cirinya baik itu secara fungsionalitas peran maupun secara desain.

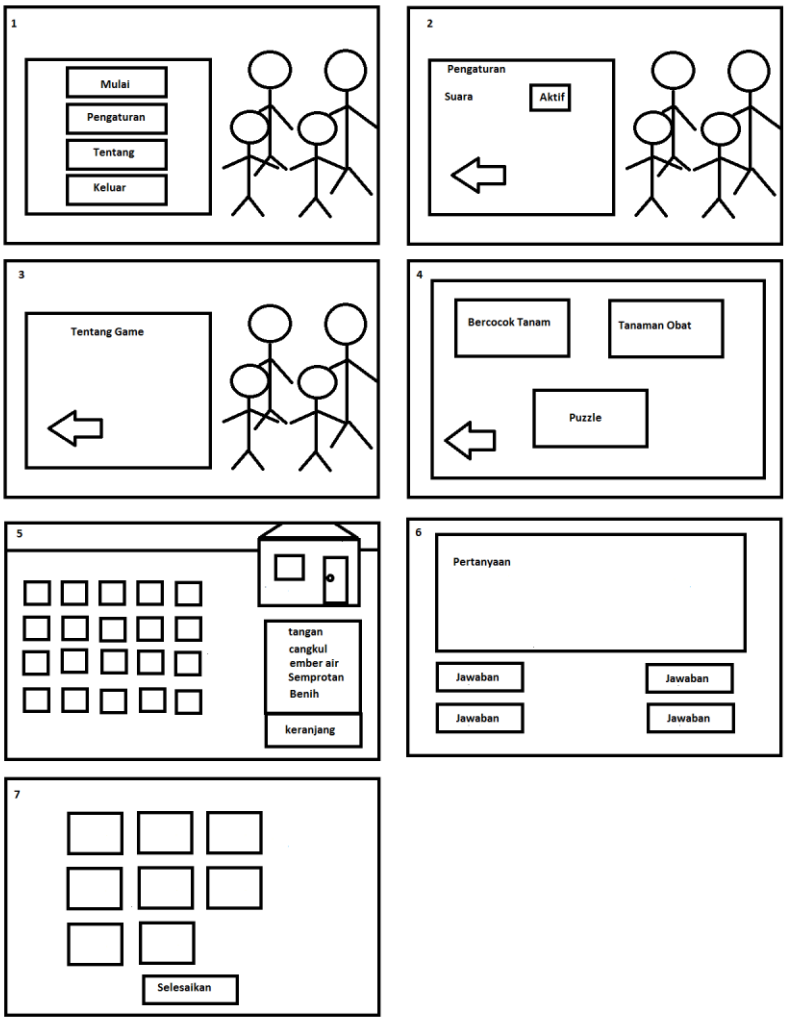
Tabel 3-25 Karakter dan Alat-alat

No.	Nama	Desain	Keterangan
1.	Aji		a. Karakter utama dalam permainan b. Mempunyai rasa ingin tahu yang tinggi c. Ingin membantu kakek dalam bercocok tanam
2.	Siska		a. Karakter utama dalam permainan b. Mempunyai rasa ingin tahu yang tinggi c. Ingin mengetahui tentang tanaman-tanaman obat
3.	Kakek		a. Kakek dari Aji dan Siska b. Senang berkebun
4.	Nenek		a. Istri sang kakek b. Mengetahui cara pengolahan jenis-jenis tanaman obat

5.	Tangan		Dalam permainan ini tangan berfungsi untuk mengambil tanaman yang berbuah ke dalam keranjang
6.	Cangkul		Dalam permainan ini cangkul berfungsi untuk mencangkul tanah
7.	Penyiram Tanaman		Dalam permainan ini Penyiram Tanaman berfungsi untuk menyiram air
8.	Benih		Benih yang akan digunakan untuk menanam tanaman.
9.	Semprotan pestisida		Dalam permainan semprotan pestisida digunakan untuk menghindari tanaman rusak akibat hama

3.2.1.2 Story Board

Storyboard merupakan sketsa gambar yang disusun secara berurutan berdasarkan naskah untuk menggambarkan lingkungan pada *game*.

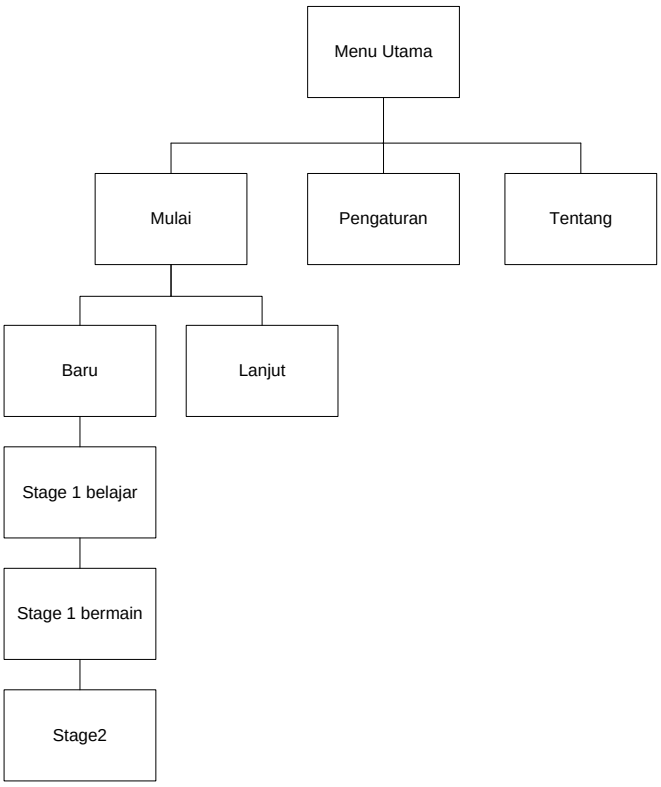


Gambar 3-27 Storyboard Game Berlibur Bersama Kakek

Pada gambar 1 menampilkan tampilan menu game dan ada pilihan untuk mulai *game*, pengaturan pada *game* dan tentang *game.* Pada gambar 2 merupakan tampilan pengaturan. Pada gambar 3 merupakan tampilan tentang *game*. Pada gambar 4 merupakan tampilan dimana pemain memilih stage yang akan dimainkannya. Pada gambar 5 merupakan tampilan dimana pemain memilih bercocok tanam dan menanam tanaman dengan alat-alat yang sudah dipersiapkan. Pada gambar 6 merupakan tampilan dimana pemain memilih stage tentang tanaman obat dan menjawab pertanyaan-pertanyaan yang ada. Pada gambar 7 merupakan tampilan dari stage 3 dimana pemain akan menyusun puzzle.

3.2.2 Perancangan Struktur Menu

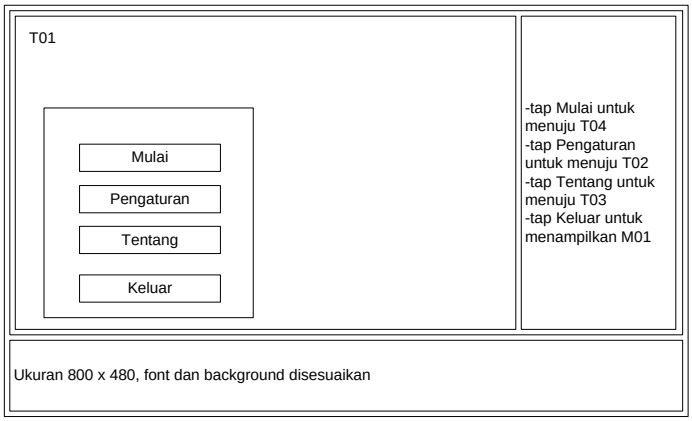
Struktur menu merupakan cara yang mudah untuk memahami menu yang ada di dalam sebuah program. Berikut adalah gambar dari struktur menu permainan Berlibur Bersama Kakek :



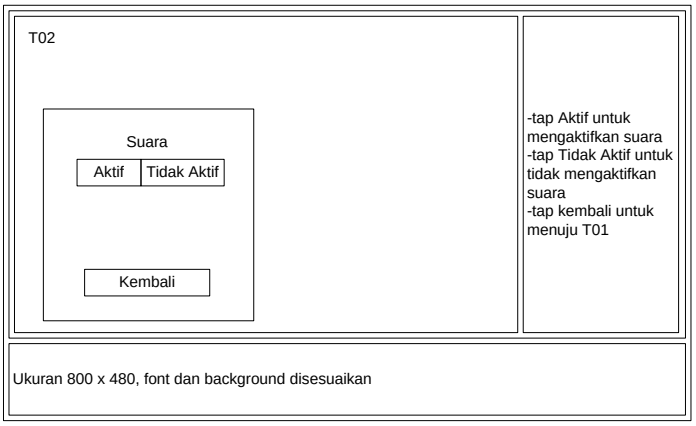
Gambar 3-28 Struktur Menu pada Game

3.2.3 Perancangan Antar Muka

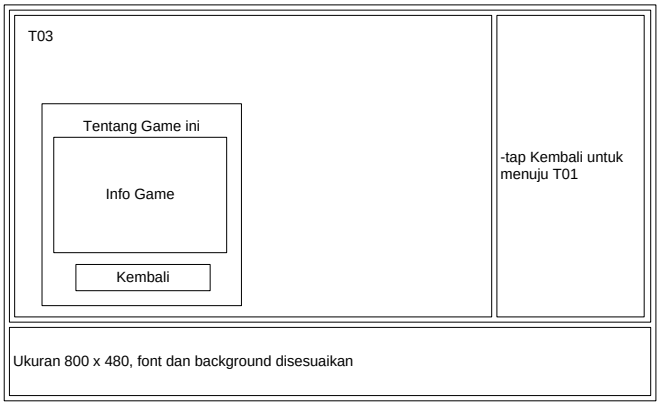
Perancangan antarmuka merupakan salah satu bagian penting dalam perancangan sistem karena nantinya antarmuka tersebut akan menjadi fasilitas yang menjembatani interaksi manusia dengan sistem. Perancangan antarmuka yang akan dibangun pada game ini adalah sebagai berikut :



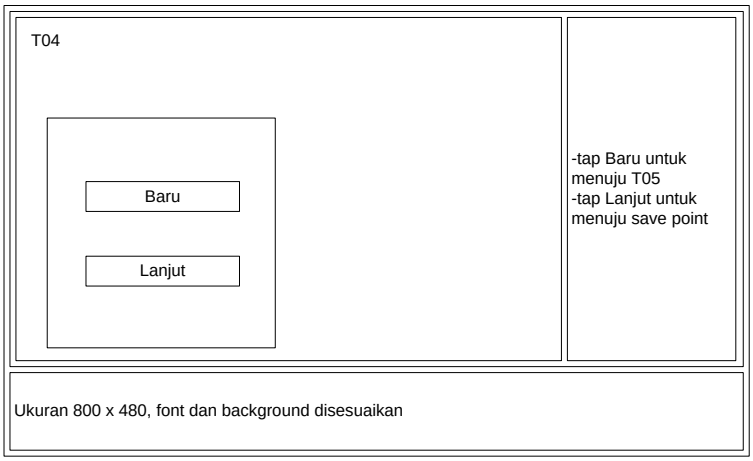
Gambar 3-29 Tampilan Menu utama



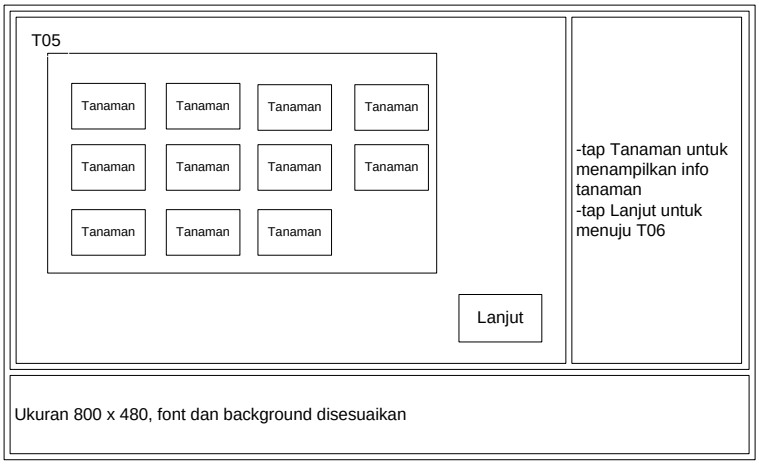
Gambar 3-30 Tampilan Pengaturan



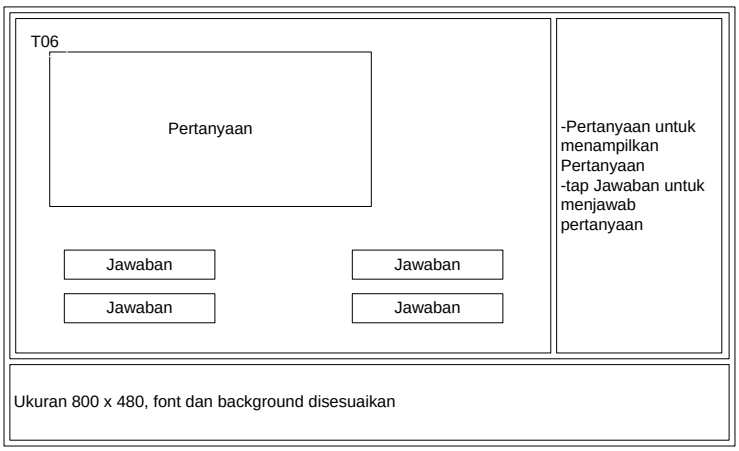
Gambar 3-31 Tampilan Tentang Game



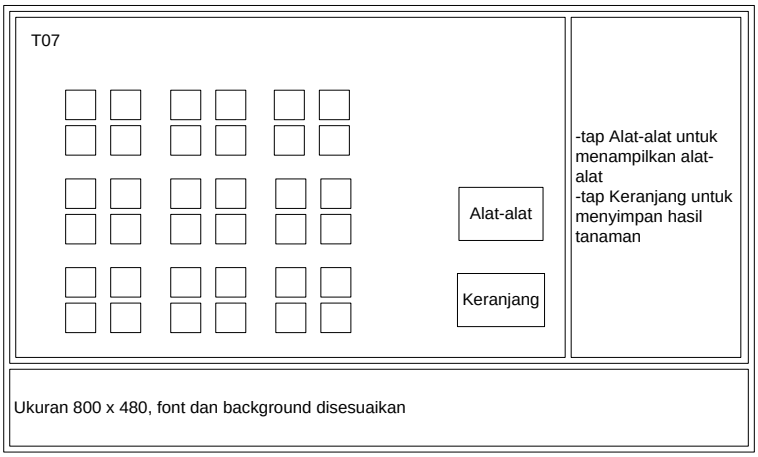
Gambar 3-32 Tampilan Pilih



Gambar 3-33 Tampilah Stage 1 Belajar



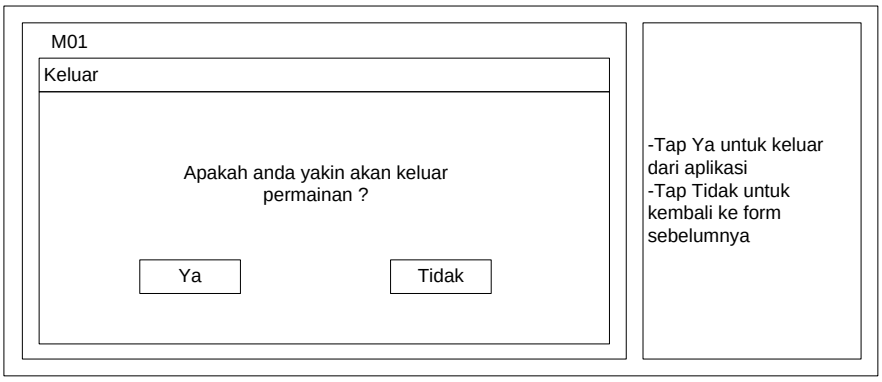
Gambar 3-34 Tampilan Stage 1 bermain



Gambar 3-35 Tampilan Stage 2

3.2.4 Perancangan Pesan

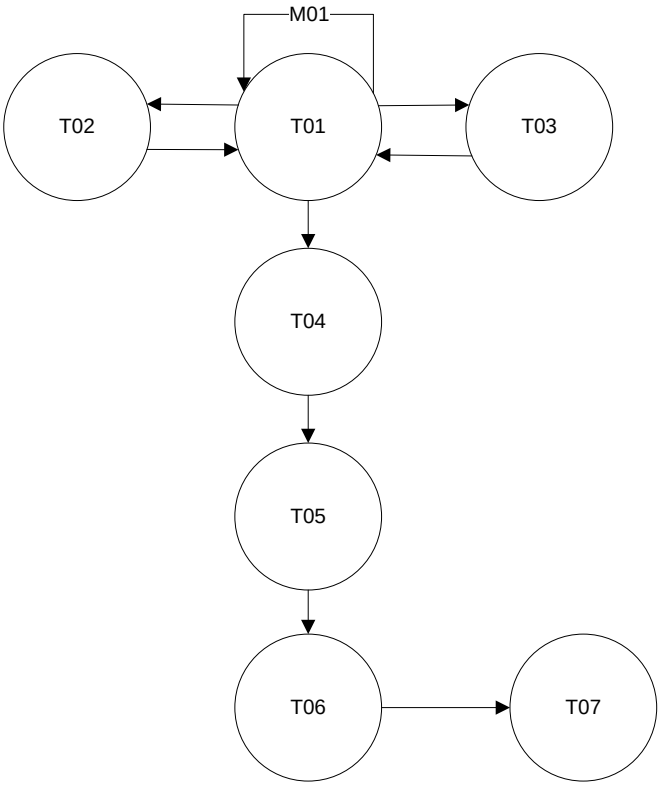
Perancangan pesan adalah rancangan antarmuka dari pesan yang akan ditampilkan apabila terjadi kesalahan dan pemberitahuan dalam proses yang dilakukan. Rancangan pesan dalam *game* yang akan dibangun adalah sebagai berikut :



Gambar 3-36 Tampilan Pesan keluar Permainan

3.2.5 Jaringan Semantik

Jaringan semantik merupakan alat efektif untuk mempresentasikan pemetaan data, yang bertujuan mencegah terjadinya duplikasi data. Berikut adalah jaringan semantis pada game ini :



Gambar 3-37 Jaringan Semantik

3.2.6 Perancangan Method

Perancangan *method* menjelaskan penjabaran alur proses-proses yang ada pada sistem dan dibangun menggunakan diagram *flowchart*. *Method* yang ada pada aplikasi *game* ini terdiri dari dua jenis *method*, yaitu *method* yang sudah ada di dalam *tools* pembangun aplikasi (hanya perlu dipanggil sesuai dengan kebutuhan) dan *method* yang dirancang berdasarkan kebutuhan aplikasi.

3.2.6.1 Penggunaan Method yang Tersedia di Dalam Tools Pembangun

Method yang digunakan dari *tools* pembangun aplikasi dalam pembangunan *game berlibur bersama kakek* ini adalah sebagai berikut.

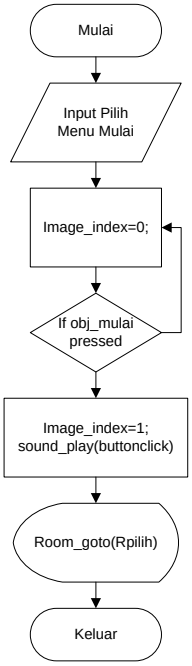
Tabel 3-26 Method yang tersedia di dalam tools pembangun

No.	Nama Method	Keterangan
1.	room_goto()	Pindah ke halaman tertentu, sesuai nama room
2.	room_goto_prev()	Pindah ke halaman sebelumnya
3.	load_background()	Menampilkan background dari room
4.	audio_pause_all()	Menghentikan musik latar untuk sementara
5.	audio_resume_all()	Melanjutkan musik latar
6.	end_game()	Menutup aplikasi
7.	instance_create()	Membuat objek
8	instance_nearest()	Memberikan id dari posisi objek
9.	instance_destroy()	Membuang objek
10.	check_instance_exist()	Memeriksa keberadaan objek di dalam room
12.	check_position()	Memeriksa posisi objek
13.	draw_sprite()	Menampilkan sprite dari objek
14	draw_text()	Menampilkan tulisan
15.	draw_seft()	Menampilkan objek

3.2.6.2 Perancangan Method yang Akan Digunakan di Dalam Game

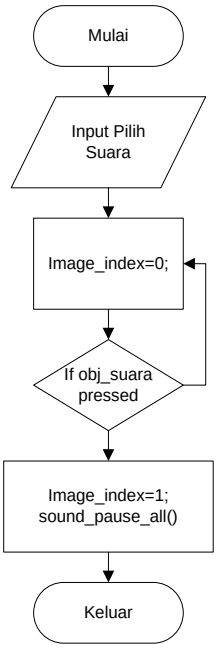
Method yang dirancang berdasarkan kebutuhan yang ada pada *game berlibur bersama kakek* ini adalah sebagai berikut :

1. Perancangan *Method* Memulai Permainan



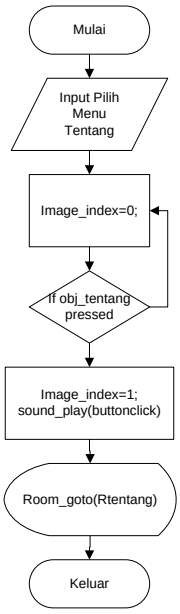
Gambar 3-38 Perancangan Method Memulai Permainan

2. Perancangan *Method* Pengaturan Suara



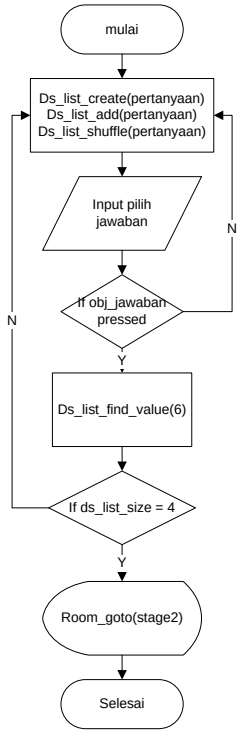
Gambar 3-39 Perancangan Method Pengaturan Suara

3. Perancangan *Method* Menu Tentang



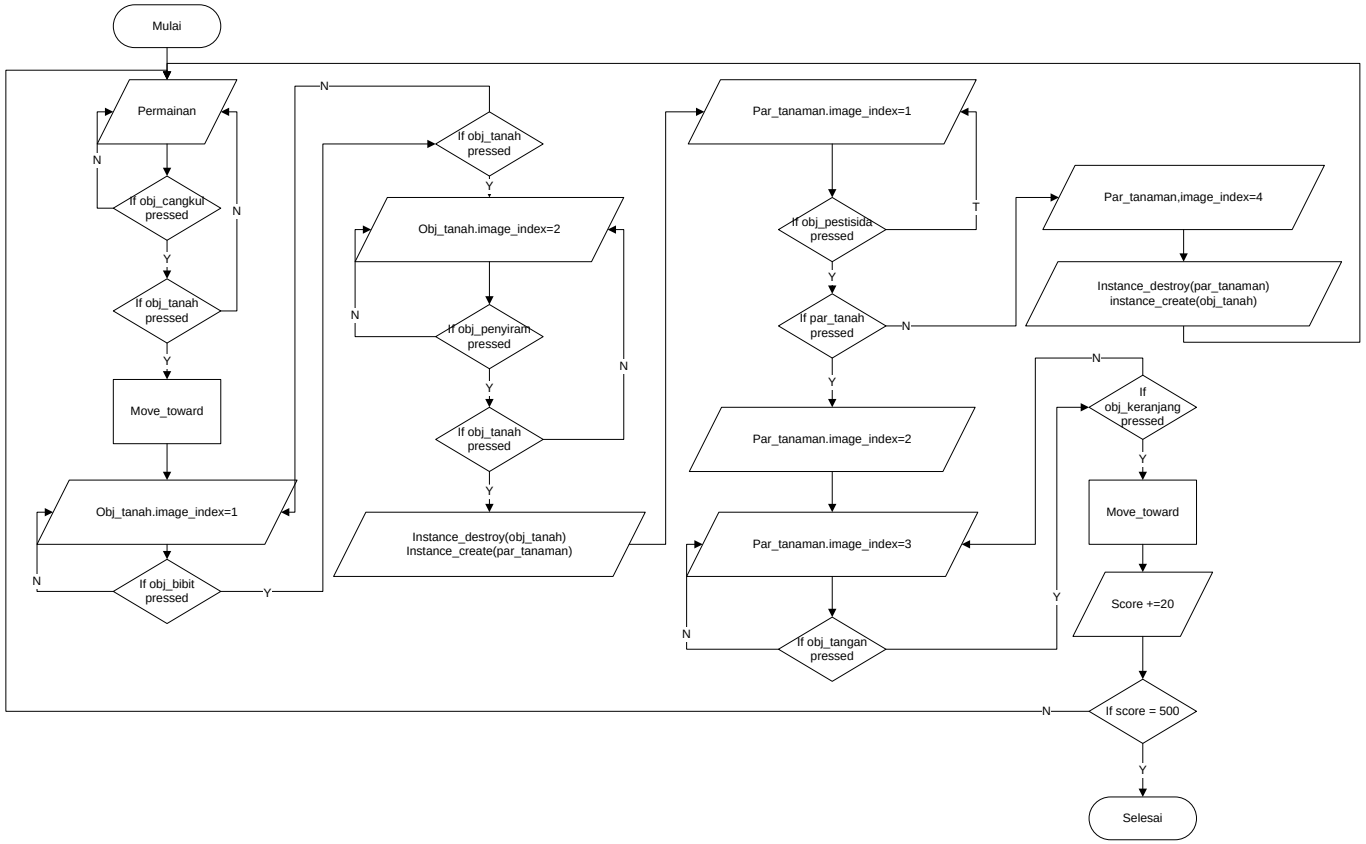
Gambar 3-40 Perancangan Method Menu Tentang

4. Perancangan *Method* Pertanyaan



Gambar 3-41 Perancangan Method Pertanyaan

5. Perancangan *Method* Menanam Tanaman



Gambar 3-42 Perancangan Method Menanam Tanaman