BAB 3

ANALISIS DAN PERANCANGAN SISTEM

3.1 Analisis Sitem

Pada bagian ini akan dijelaskan tentang analisis terhadap sistem yang akan dibangun. Bagian ini menjelaskan analisis-analisis yang dilakukan.

3.1.1 Analisis Masalah

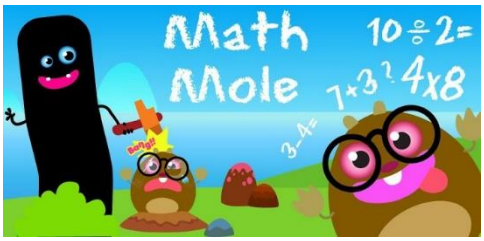
Berdasarkan analisa yang dilakukan, maka permasalahan yang dihadapi adalah bagaimana pembelajaran dalam operasi hitung bilangan dasar matematika lebih interaktif dengan menggunakan tokoh Kabayan sebagai tokoh utama *game*.

3.1.2 Analisis *Game* Sejenis

Analisis *game* sejenis merupakan analisis yang akan membahas mengenai game yang menjadi acuan dalam pembangunan sebuah *game*. Dalam analisis *game* sejenis ini akan membahas mengenai komponen yang berperan penting untuk membangun *game* edukasi. Komponen *game* yang akan di analisis yaitu *story line*, *game play*, dan edukasi.

3.1.2.1 Analisis *Game* Mole Math LITTLE

Game Mole Math LITTLE merupakan *game* edukasi untuk pembelajaran matematika terutama dalam hal berhitung pertambahan, pengurangan, perkalian, pembagian.



Gambar 3.1 *Game* Mole Math LITTLE

1. *Story Line* Mole Math LITTE

Game ini menceritakan tentang mole yang menyerang ke kebun Tuba. Mole yang datang harus diusir secepat mungkin diusir. Hanya ada satu cara untuk menyingkirkan mereka. Mencari dan memukul mole yang menampilkan jawaban yang tepat dari pertanyaan yang tampil di papan tulis.

2. *Game Play* Mole Math LITTLE

Pada *game* Mole math versi LITTLE hanya ada level "medium" dan permainan "20 pertanyaan". *Game play* dari *game* ini sangat sederhana namun membutuhkan konsentrasi tinggi. Pemain *game* ini harus mencari jawaban dari pertanyaan yang tampil di papan tulis dengan menyentuh Mole yang keluar dengan jawaban yang tepat. Pemain akan dituntut untuk menjawab 20 pertanyaan dengan yang singkat karena, penilaian atau *scoring* berpengaruh terhadap waktu.

3. Edukasi Mole Math LITTLE

Pada *game* Mole math versi LITTLE memiliki konten edukasi tentang penjumlahan, pengurangan, perkalian, dan pembagian. Bentuk penyampaian edukasi dengan mengunnakan pertanyaan.

3.1.2.2 Analisis *Game* Pick a Path

Game Pick a Path merupakan game edukasi tentang pembelajaran matematika dalam perbandingan antara besar dan kecil.



Gambar 3.2 *Game* Pick a Path

1. *Storyline* Pick a Path

Game ini menceritakan tokoh Okta yang mencari jalur dalam graph yang tertampil. Tokoh Okta dapat keluar jika bisa mencapai targetnya.

2. *Game Play* Pick a Path

Game ini mempunyai 7 level dengan tingkat kesulitan yang berbeda dalam setiap level ataupun sub level nya tetapi, mempunyai game play yang sama. *Game play* pada *game* ini sama disetiap level nya yaitu, pemain harus memilih jalur untuk membantu tokoh Okta dengan mencari jalur (maksimum, minimum, atau nilai tertentu).

3.1.2.2.1 Edukasi Pick a Path

Pada *game* Pick a Path terdapat edukasi dalam menentukan perbandingan antara besar atau kecil. Bentuk penyampaian edukasi dengan menggunakan graph dimana pemain harus mencari jalur dengan membandingkan setiap jalur yang akan dilewati.

3.1.3 Analisis Game yang Akan Dibangun

Pada bagian ini akan dijelaskan analisis untuk membangun *game* Kabayan Berhitung.

3.1.3.1 Storyline

Game Kabayan Berhitung menceritakan tentang tokoh Kabayan. Pada game ini menceritakan Kabayan yang memiliki kekurangan untuk berhitung, sehingga membuat Kabayan tidak bisa bermain dengan tokoh Nyi Iteung karena tokoh Abah ingin agar anaknya yaitu Nyi Iteung bermain dengan anak yang bisa berhitung. Untuk membuat Kabayan bisa bermain lagi dengan Nyi Iteung, Kabayan akan diberikan pelatihan berhitung oleh Abah sebelum Kabayan dapat bermain lagi dengan Nyi Iteung.

Pelatihan yang diberikan Abah akan terdapat 3 pelatihan, yaitu :

1. Pelatihan pertama

Pada pelatihan pertama ini Abah akan melatih Kabayan di Ladang. Pelatihan ini dilakukan Abah agar Kabayan dapat melakukan pertambahan dan pengurangan.

2. Pelatihan kedua

Pada pelatihan kedua, Abah akan membawa Kabayan untuk mencari jalan terdekat menuju rumah Nyi Iteung. Setiap Kabayan menemui titik akhir jalan yang dicari, Abah akan bertanya perkalian dan pembagian kepada Kabayan untuk melanjutkan perjalanan.

3. Pelatihan ketiga

Pada pelatihan ini Kabayan sudah berada di depan rumah Nyi Iteung. Pelatihan ketiga ini Kabayan harus membuka setiap pintu untuk bertemu dengan Nyi Iteung. Di setiap Kabayan membuka pintu, Abah akan memberikan kunci untuk membuka pintu dengan memberikan pertanyaan percampuran antara

pertambahan, pengurangan, perkalian dan pembagian. Setelah Kabayan bisa menjawab pertanyaan dari Abah, Kabayan harus memasukan kunci tersebut dengan memecahkan hasil pertanyaan kedalam puluhan dan satuan. Kabayan harus membuka pintu sebanyak 3 kali pada pelatihan ini.

Setelah 3 pelatihan selesai maka Kabayan diijinkan Abah untuk bermain dengan Nyi Iteung kembali. Permainan selesai ketika semua pelatihan sudah di bereskan Kabayan.

3.1.3.2 Analisis Tingkat Kesulitan

Analisis tingkat kesulitan merupakan analisis untuk menguji *user* untuk menyelesaikan *game*.

1. Tingkat 1 (*level* 1)

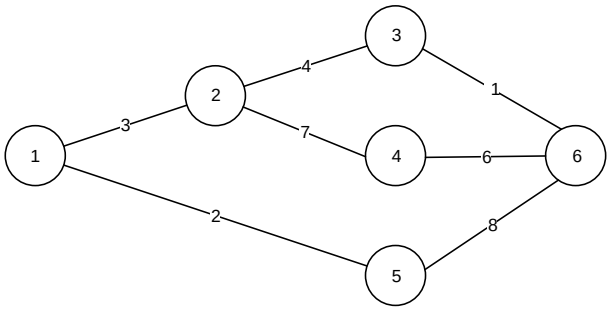
Materi yang disajikan adalah pertambahan dan pengurangan dengan pembatasan nilai 1 – 9. Pembatasan nilai diadakan karena pertambahan dan pengurangan dari 1 – 9 merupakan kewajiban dari setiap siswa untuk di hafal. Contoh soal :

a. 3 + 2 = ?
b. 9 – 1 = ?

2. Tingkat 2 (*level* 2)

Materi yang disajikan yaitu :

a. Pertambahan dan pengurangan dengan menggunakan pemilihan dalam kasus mencari jarak. Sebagai contohakan tertampil graph sebagai berikut :



Gambar 3.3 Contoh Graph

Jarak yang harus ditempuh adalah 8. Pemain harus mencari jalur dengan jarak yang ditanyakan yaitu 8.

b. Perkalian dan pembagian dengan pembatasan nilai 1 – 9. Pembatasan nilai diadakan karena perkalian dan pembagian dari 1 – 9 merupakan kewajiban dari setiap siswa untuk di hafal. Contoh soal sebagai berikut:

i. 3 x 8 = ?

ii. 24 : 6 = ?

3. Tingkat 3 (*level* 3)

Materi yang disajikan yaitu :

a. Penggabungan antara pertambahan, pengurangan, perkalian, sebagai contoh :

(4 x 4) – 1 = ?

b. Pemecahan bilangan ke puluhan dan satuan, sebagai berikut :

		puluhan		satuan
42	=	?	+	?

3.1.3.3 Game Play

Tugas utama pemain dalam memainkan *game* ini adalah untuk menyelesaikan misi-misi yang ada pada setiap bagian atau *level* pada *game.* Pada *game* ini akan dibagi dalam 3 *level* atau pelatihan. Setiap pelatihan memiliki misi yang berbeda. Kesulitan setiap *level* atau pelatihan akan berada pada pertanyaan yang tertampil. Berikut ini akan dijelaskan misi-misi dari setiap pelatihannya :

1. Pelatihan 1 (*level* 1)
   a. Pemain harus mencangkul ladang untuk mencari jawaban dari pertanyaan yang tampil.
   b. Jika pemain merasa jawaban benar dan waktu menjawab belum habis, maka pemain memeriksakan jawaban.
   c. Pemain harus menjawab sebanyak 5 pertanyaan dan kesempatan belum habis untuk melanjutkan ke pelatihan 2.
2. Pelatihan 2 (*level* 2)
   a. Pemain pada awal pelatihan 2 harus mencari jalur terpendek pada graph pada graph ke 1 dengan jarak yang ditanyakan.
   b. Setelah menyelesaikan graph ke 1, pemain harus menjawab pertanyaan.

c. Jika pemain menjawab dengan benar, pemain harus mencari jalur terpendek pada graph ke 2 dengan jarak yang ditanyakan.
d. Setelah menyelesaikan graph ke 2, pemain harus menjawab pertanyaan.
e. Jika pemain menjawab dengan benar, pemain harus mencari jalur terpendek pada graph ke 3 dengan jarak yang ditanyakan.
f. Setelah menyelesaikan graph ke 3, pemain harus menjawab pertanyaan.
g. Jika pemain menjawab dengan benar, pemain akan melanjutkan permainan ke pelatihan 3.

3. Pelatihan 3 (*level* 3)
   a. Pemain harus menyentuh pintu ke 1
   b. Pemain harus menjawab pertanyaan agar mendapatkan sandi untuk membuka pintu ke 1.
   c. Pemain harus memasukan sandi dengan benar untuk membuka pintu ke 1 dan menuju pintu ke 2.
   d. Pemain harus menyentuh pintu ke 2
   e. Pemain harus menjawab pertanyaan agar mendapatkan sandi untuk membuka pintu ke 2.
   f. Pemain harus memasukan sandi dengan benar untuk membuka pintu ke 2 dan menuju pintu ke 3.
   g. Pemain harus menyentuh pintu ke 3
   h. Pemain harus menjawab pertanyaan agar mendapatkan sandi untuk membuka pintu ke 3.
   i. Pemain harus memasukan sandi dengan benar untuk membuka pintu ke 3 dan pemain berhasil.

3.1.3.4 Analisis *Scoring*

Analisis *scoring* merupakan analisa untuk memberikan penilaian kepada *user* yang menyelesaikan *game*. *Scoring* pada game Kabayan Berhitung akan berupa nilai. Dengan keterangan, sebagai berikut :

Tabel 3.1 Analisa Scoring

Pelatihan (*Level*)	Keterangan	Score	Jumlah
1	Terdapat 5 pertanyaan	5 x 10 (score yang diberi)	50
2	Terdapat 3 kali penyelesaian jalan tercepat	3 x 15 (score yang diberi)	45
2	Terdapat 3 pertanyaan	3 x 10 (score yang diberi)	30
3	Terdapat 3 pertanayaan	3 x 15 (score yang diberi)	45
3	Terdapat 3 pemecahan hasil	3 x 10 (score yang diberi)	30
Total nilai			200

Pada *game* Kabayan Berhitung penilaian atau *scoring* tidak berpengaruh terhadap kesalahan. Jika pemain melakukan kesalahan disetiap pelatihan atau kejadian, maka kesempatan pemain akan berkurang.

3.1.3.5 Edukasi

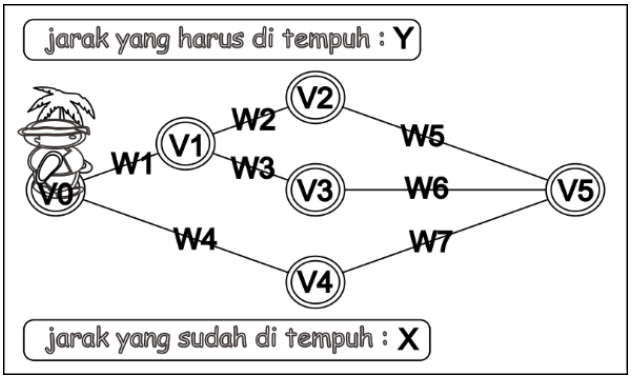
Game Kabayan Berhitung akan mengemas edukasi tentang pembelajaran matematika dalam operasi hitung bilangan yaitu penjumlahan, pengurangan, perkalian dan pembagian. Pembagian materi akan dibedakan dalam setiap *level* di *game* Kabayan Berhitung. Dimulai dari *level* 1 yaitu tentang penjumlahan dan pengurangan. Dilanjutkan *level* 2 tentang pencarian jalur dengan menggunakan pertambahan dan pengurangan, dan materi perkalian dan pembagian. Terakhir pada *level* 3 akan memberikan materi operasi hitung campuran dari pertambahan, pengurangan, perkalian, dan pembagian, juga pemecahan bilangan.

3.1.4 Analisis Algoritma

Analisis Algoritma merupakan suatu analisis untuk menerapkan algoritma yang digunakan terhadap *game* yang akan dibangun.

3.1.1.1 Analisis Algoritma *Dijkstra*

Algoritma *Dijkstra* merupakan algoritma untuk mencari jarak minimum dan *rute* terpendek dari graph berbobot tidak negatif dan berarah. Dalam pembangunan *Game* Kabayan Berhitung, algoritma ini digunakan untuk mencari jarak minminum dari kombinasi graph yang tertampil pada level 2. Hasil dari algoritma ini akan diperlihatkan sebagai nilai Y di gambar 3.4. Berikut adalah contoh kombinasi graph yang tertampil di level 2 :

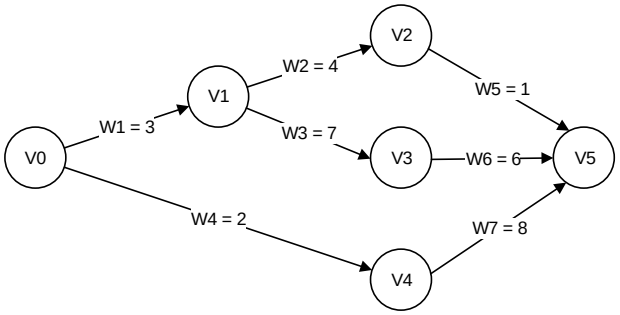


Gambar 3.4 Contoh graph yang tertampil di level 2

Keterangan :

1. V1-5 : merupakan titik tujuan yang nantinya akan ditempuh Kabayan (*vertex*).
2. W1-7 : merupakan jarak atau bobot antara setiap titik tujuan. Jarak yang tertampil akan selalu berubah-ubah.
3. Y : merupakan hasil dari algoritma Dijkstra yaitu jarak minimum pada graph yang tertampil yang menjadi parameter untuk pemain.
4. X : merupakan jarak yang sudah ditempuh pemain.

Dari contoh *graph* yang tertampil, akan dijelaskan mengenai langkah algoritma dari algoritma Dijkstra untuk menghitung jarak minimum. Berikut adalah gambar dari graph yang sudah di inisialisasi bobot atau jarak antara titik tujuannya (*vertex*) :



Gambar 3.5 Contoh *graph* yang sudah dinisialisasi jarak antara *vertex* nya

Graph akan direpresentasikan berupa array 2 dimensi (matriks) dengan indeks array baris dan kolom akan mengacu kepada jumlah *vertex* (array[0..jumlah veretex][0..jumlah vertex] of integer). Kumpulan *vertex* ini akan diimplementasikan sebagai graph[ ][ ], sebagai berikut :

Tabel 3.2 Matriks dari Contoh Graph

graph	V0	V1	V2	V3	V4	V5
V0	0	3	∞	∞	2	∞
V1	∞	0	4	7	∞	∞
V2	∞	∞	0	∞	∞	1
V3	∞	∞	∞	0	∞	6
V4	∞	∞	∞	∞	0	8
V5	∞	∞	∞	∞	∞	0

Keterangan :

1. Baris merupakan *vertex* awal dan kolom merupakan *vertex* tujuan.
2. Jika *vertex* awal menuju *vertex* tujuan tidak mempunyai arah maka akan di insialisasi dengan *infinity* (∞).
3. Jika vertex mengacu ke *vertex* itu sendiri maka akan di inisialisasi dengan 0.

Algoritma Dijkstra akan di jelaskan dalam *procedure* pada *Pseudocode* berikut :

```
Procedure dijkstra(input  d:array[0..JumlahVertex][0..JumlahVertex] of integer,
from:intger, output distance:array[0..JumlahVertex] of integer, predecessors :
array[0..JumlahVertex] of integer)

Deklarasi
    set : List
    closed : array[0..JumlahVertex] of Boolean
    minDistance : integer
    node : integer
    i : integer
Algoritma
    set ← add(from)
    {inisalisasi distances}
    for i ← 0 to i < JumlahGraph do
        if(i ≠ from)then
            distance[i] ← infinity
        else
            distance[i] ← 0
        endif
    endfor

    predecessors[from] ← -1

    while ( set ≠ kosong) do
        minDistance ← infinity
        node ← -1
        for each i in adj(set) do
            if distaces[i] < minDistance then
                minDistance ← distances[i]
                node ← i
            endif
        endfor
        set ← remove(node)
        closed[node]← true
        for i ← 0 to i < JumlahVertex do
            if d[node][i] ≠ infinity then
                if closed[i] = false then
                    distances[i] ← distances[node] + d[node][i]
```

```
                    predecessors[i] ← node
                    if i not in set then
                        set ← add(i)
                    endif
                endif
            endif
        endfor
    endwhile
```

Berikut penjelasan dari *procedure* Dijkstra :

Tabel 3.3 *Pseudocode* Algoritma Dijkstra

No	Algoritma	*Pseudocode*
1	Masukan **form** kedalam **set**	set ← **add**(from)

2	Insialisasi **distances**	**for** i ← 0 to JumlahGraph **do**
		if(i ≠ from)**then**
		distance[i] ← infinity
		else
		distance[i] ← 0
		endif
		endfor

3	Masukan nilai -1 ke **predecessors** indeks ke **from**	predecessors[from] ←-1
3	Ketika set tidak kosong maka lakukan proses selanjutnya	**while** (set ≠ kosong) **do**
4	Inisialisasi **minDistance** dengan **infinity**	minDistance ← infinity
5	Masukan -1 ke **node**	node ← -1

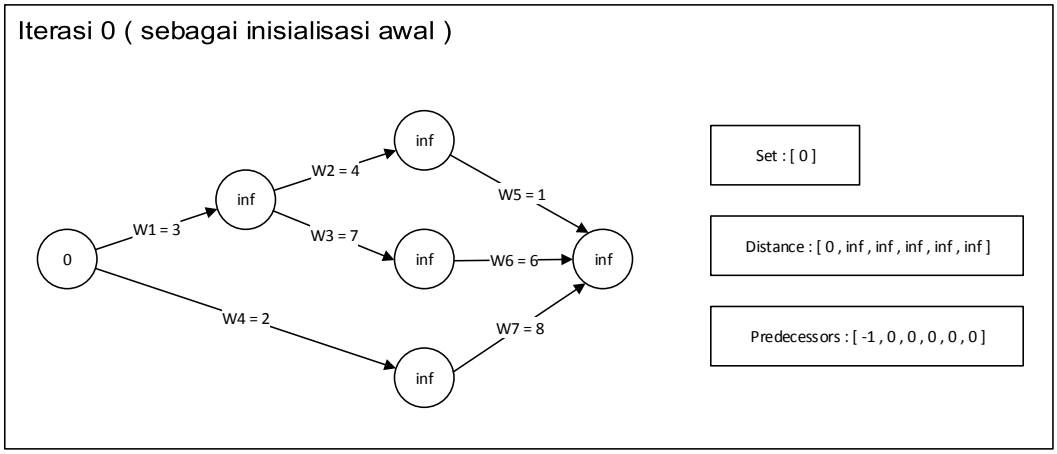
6	Proses untuk mencari jarak minimal di dalam **set**	**for** each i in adj(set) **do**
		if distaces[i] < minDistance **then**
		minDistance ← distances[i]
		node ← i
		endif
		endfor

7	Hapus **node** di dalam **set**	set ← **remove**(node)
8	Masukan nilai true pada **closed** dengan indeks ke **node**	closed[node]← true

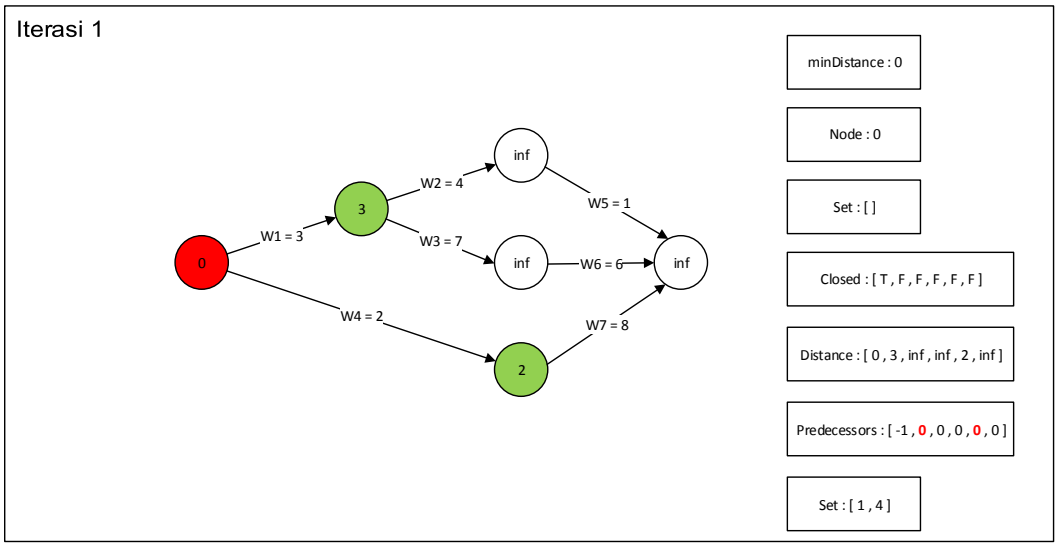
9	Proses perbandingan jarak tempuh dengan *vertex* yang dituju dan mencari nilai yang paling kecil kemudian dimasukan kedalam **set**.	**for** i ← 0 **to** JumlahVertex **do**
		if d[node][i] ≠ infinity **then**
		if closed[i] = false **then**
		distances[i] ← distances[node] + d[node][i]
		predecessors[i] ← node
		if i not in set **then**
		set ← **add**(i)
		endif
		endif
		endif
		endfor

10	Akhir dari while ketika **set** sudah kosong	**endwhile**

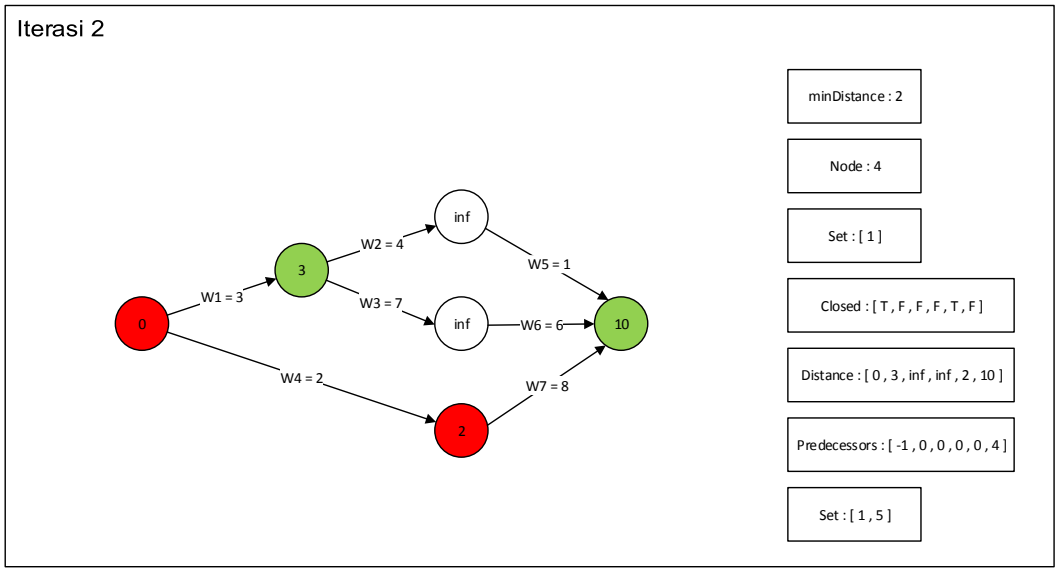
Berikut langkah-langkah algoritma pada graph yang tertampil :



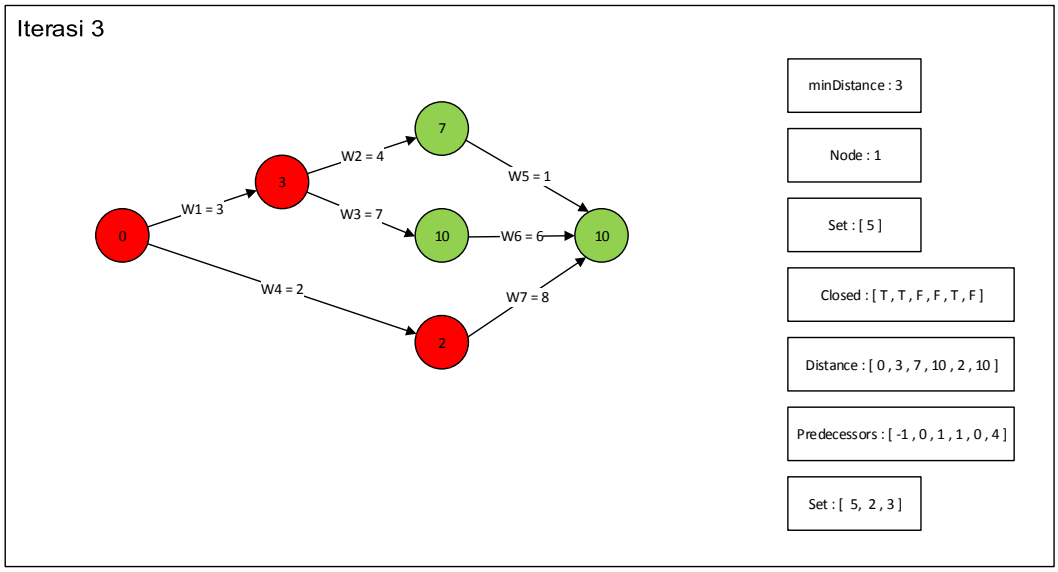
Gambar 3.6 Iterasi 0 Algoritma Dijkstra



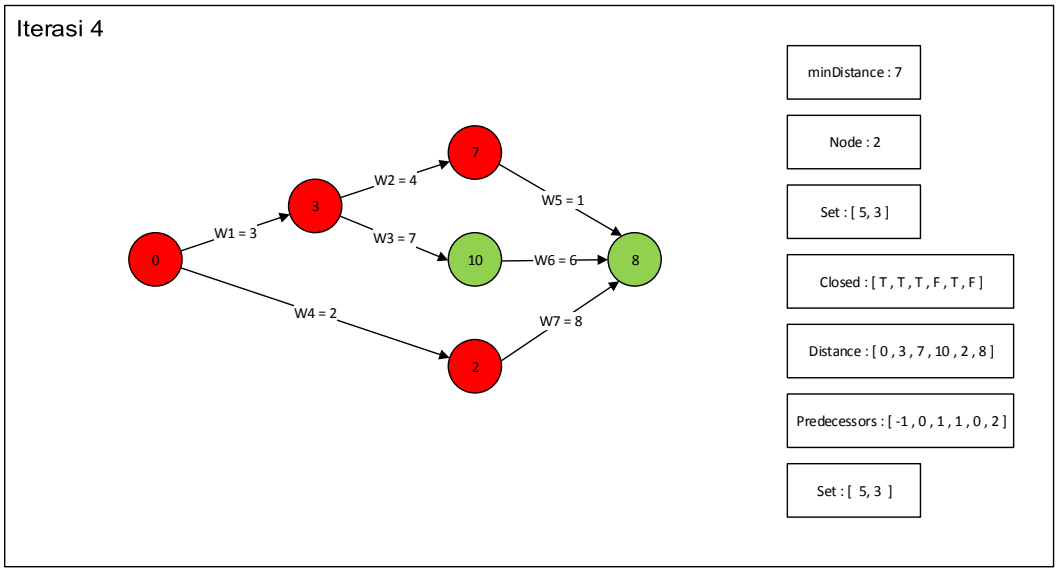
Gambar 3.7 Iterasi 1 Algoritma Dijkstra



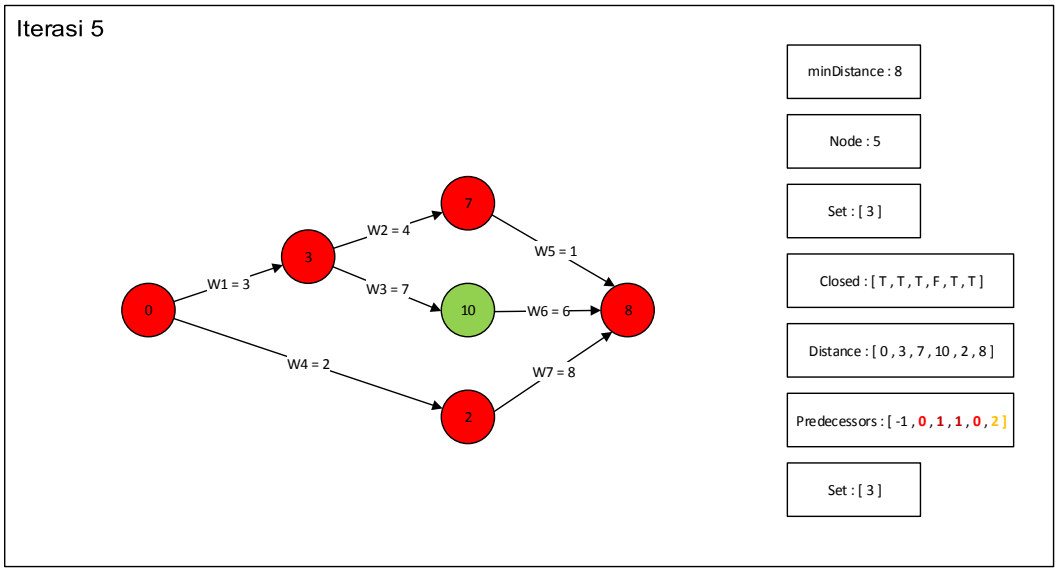
Gambar 3.8 Iterasi 2 Algoritma Dijkstra



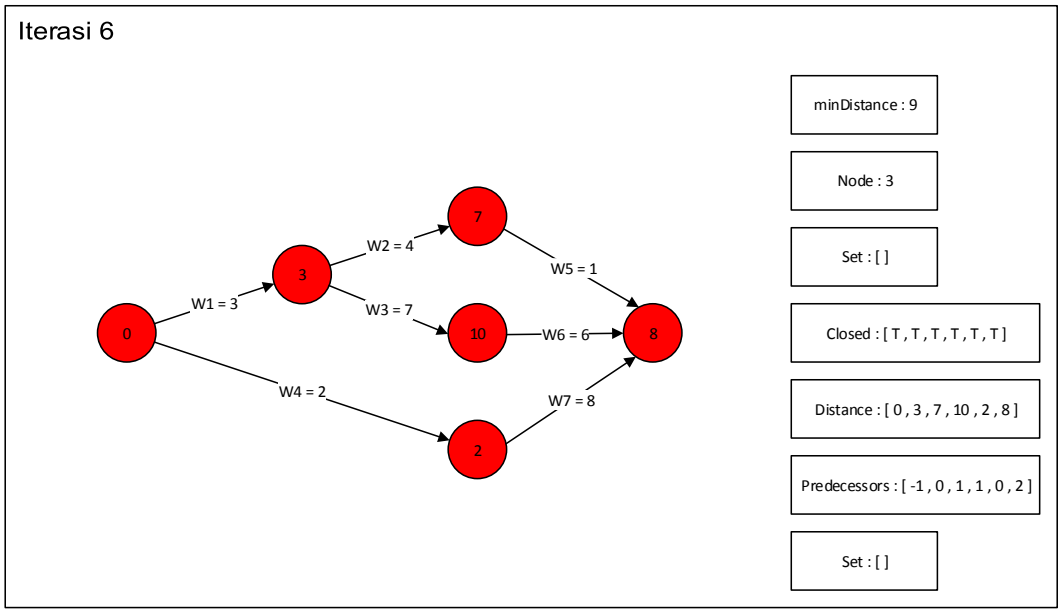
Gambar 3.9 Iterasi 3 Algoritma Dijkstra



Gambar 3.10 Iterasi 4 Algoritma Dijkstra



Gambar 3.11 Iterasi 5 Algoritma Dijkstra



Gambar 3.12 Iterasi 6 Algoritma Dijkstra

Berikut hasil berupa tabel hasil perhitungan :

Tabel 3.4 Hasil Perhitungan Algoritma Dijkstra

node	distances						predecessors	set
	V0	V1	V2	V3	V4	V5		
-1	(0,-)	(∞,-)	(∞,-)	(∞,-)	(∞,-)	(∞,-)	0,0,0,0,0,0	0
0	(0,-)	(3,V0)	(∞,-)	(∞,-)	(2,V0)	(∞,-)	-1,0,0,0,0,0	1,4
4	(0,-)	(3,V0)	(∞,-)	(∞,-)	(2,V0)	(10,V4)	-1,0,0,0,0,4	1,5
1	(0,-)	(3,V0)	(7,V1)	(10,V3)	(2,V0)	(10,V4)	-1,0,1,1,0,4	5,3,2
2	(0,-)	(3,V0)	(7,V1)	(10,V3)	(2,V0)	(8,V2)	-1,0,1,1,0,2	5,3
5	(0,-)	(3,V0)	(7,V1)	(10,V3)	(2,V0)	(8,V2)	-1,0,1,1,0,2	3
3	(0,-)	(3,V0)	(7,V1)	(10,V3)	(2,V0)	(8,V2)	-1,0,1,1,0,2	null

Untuk mendapatkan pembacaan yang benar dan mengetahui jarak minimal, dibutuhkan suatu *procedure Reconstructor Predecessor*. Berikut *Procedure Reconstructor Predecessor* yang digunakan untuk memperbaiki pembacaan :

```
Procedure reconstructorPredecessor(input  sumVertex : integer, s :
integer, output solutionPath : array[0..IndeksSolusi] of integer,
minDistanceGraph : integer )
DEKLARASI
    i : integer
ALGORITMA
    i ← 0
    currentVertex ← sumVertex – 1
    while currentVertex ≠ 0 do
        solutionPath[i] ← currentVertex
```

```
        currentVertex ← predecessors[currentVertex]
        i ← i + 1
    endwhile
    minDistanceGraph ← distances[panjang array distances - 1]
```

Berikut penjelasan dari procedure Reconstructor Predecessor :

Tabel 3.5 ***Pseudocode Procedure Reconstructor Predecessor***

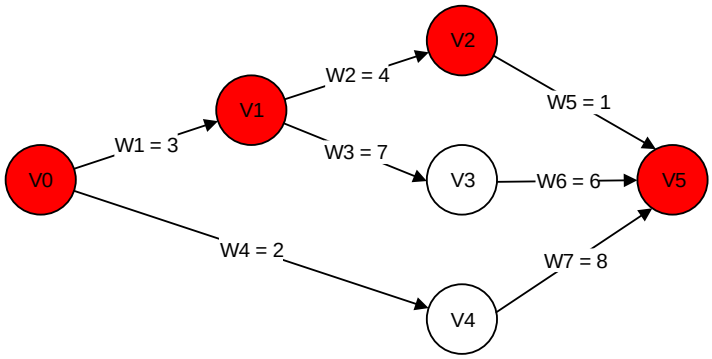
No	Algoritma	*Pseudocode*
1	Inisalisasi i = 0	i ← 0
2	Inisalisasi **currentVertex** dengan **sumVertex** – 1	currentVertex ← sumVertex – 1
3	Ulangi jika **currentVertex** tidak sama dengan 0 lakukan proses selanjutnya	**while** currentVertex ≠ 0 **do**
4	Masukan **currentVertex** kedalam **solutionPath** dengan indeks ke **i**	solutionPath[i] ← currentVertex
5	Insialisasi kembali **currenVertex** dengan **predecessors** indeks ke **currentVertex**	currentVertex ← predecessors[currentVertex]
6	Tambah nilai indeks **i**	i ← i + 1
7	Akhir dari **while** jika **currenVertex** = 0	**endwhile**
8	Masukan **distances** indeks ke panjang array **distances** dikurangi 1 kedalam **minDistanceGraph**	minDistanceGraph ← distances[panjang array distances - 1]

Hasil dari procedure Reconstructor Predecessor, sebagai berikut :

Tabel 3.6 Hasil ***Procedure Reconstructor Predecessor***

i	currentVertex	predecessors[currentVertex]	solutionPath
0	5	2	[5]
1	2	1	[5,2]
2	1	0	[5,2,1]
-	0	-	[5,2,1,0]
minDistanceGraph		8	

Dari tabel diatas dapat didapatkan hasil bahwa jarak minimum dari *graph* adalah 8 yang menjadi nilai Y, dengan jalur yang ditempuh sebagai berikut :



Gambar 3.13 Graph Hasil dari *Procedure Dijkstra* (Algoritma *Dijkstra*)

3.1.1.2 Algoritma Fisher Yates

Algoritma *Fisher Yates* merupkan algoritma untuk pengacakan suatu urutan. Dalam pembangunan *game* Kabayan Berhitung, algoritma ini akan digunakan untuk memenuhi pengacakan jawaban yang bertempat pada *level* 1 atau pelatihan 1. Jawaban akan berupa *array* dengan inisialisasi nilai yaitu : {0,1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19}.

Berikut adalah *function Fisher Yates* :

```
function fisherYates(input array : array[0..JumlahElemen] of integer) →
array[0..JumlahElemen] of integer
{ menghasilkan array yang sudah teracak }

DEKLARASI
    m : integer
    i : integer
    t : integer

ALGORITMA
    m ← JumlahElemen
    i ← 0
    t ← 0
    while ( m > 0 ) do
        i ← random(m-1)
        t ← array[m]
        array[m] ← array[i]
        array[i] ← t
    endwhile
    return array
```

Berikut penjelasan dari *function Fisher Yates* :

Tabel 3.7 Graph Hasil dari *Procedure Dijkstra* (Algoritma *Dijkstra*)

No	Algoritma	*Pseudocode*
1	Inisialisasi **m** dengan nilai jumlah elemen di variabel **array**	m ← jumlah elemen di variabel array
2	Inisialisasi **i** dengan nilai 0	i ← 0
3	Insialisasi **t** dengan nilai 0	t ← 0
4	Ulangi ketika **m** lebih dari 0 lakukan proses selanjutnya	while (m > 0) do
5	Simpan nilai random dengan indeks **m**-1	i ← random(m-1)
6	Simpan nilai yang berada dalam **array** dengan indeks ke **m** ke dalam **t**	t ← array[m]
7	Tukar nilai di **array** indeks ke **m** dengan nilai di **array** indeks ke **i**	array[m] ← array[i]
8	Ganti nilai di **array** indeks ke **i** dengan **t**	array[i] ← t
9	Akhir dari **while**	endwhile
10	Kirimkan **array** sebagai *output*.	return array

Hasil dari pengacakan *function Fisher Yates* dalam pengcakan jawaban sebagai berikut :

Tabel 3.8 Hasil *Function* Fisher Yates

m	t	array[m]	array[t]	Array yang sudah fix	Array yang telah di swap
19	13	19	13	13	0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 19, 14, 15, 16, 17, 18, 13
18	7	18	7	7, 13	0, 1, 2, 3, 4, 5, 6, 18, 8, 9, 10, 11, 12, 19, 14, 15, 16, 17, 7, 13
17	15	17	15	15, 7, 13	0, 1, 2, 3, 4, 5, 6, 18, 8, 9, 10, 11, 12, 19, 14, 17, 16, 15, 7, 13
16	11	16	11	11, 15, 7, 13	0, 1, 2, 3, 4, 5, 6, 18, 8, 9, 10, 16, 12, 19, 14, 17, 11, 15, 7, 13
15	13	17	19	19, 11, 15, 7, 13	0, 1, 2, 3, 4, 5, 6, 18, 8, 9, 10, 16, 12, 17, 14, 19, 11, 15, 7, 13
14	6	14	6	6, 19, 11, 15, 7, 13	0, 1, 2, 3, 4, 5, 14, 18, 8, 9, 10, 16, 12, 17, 6, 19, 11, 15, 7, 13
13	4	17	4	4, 6, 19, 11, 15, 7, 13	0, 1, 2, 3, 17, 5, 14, 18, 8, 9, 10, 16, 12, 4, 6, 19, 11, 15, 7, 13
12	10	12	10	10, 4, 6, 19, 11, 15, 7, 13	0, 1, 2, 3, 17, 5, 14, 18, 8, 9, 12, 16, 10, 4, 6, 19, 11, 15, 7, 13
11	3	16	3	3, 10, 4, 6, 19, 11, 15, 7, 13	0, 1, 2, 16, 17, 5, 14, 18, 8, 9, 12, 3, 10, 4, 6, 19, 11, 15, 7,

					13
10	4	12	17	17, 3, 10, 4, 6, 19, 11, 15, 7, 13	0, 1, 2, 16, 12, 5, 14, 18, 8, 9, 17, 3, 10, 4, 6, 19, 11, 15, 7, 13
9	5	9	5	5, 17, 3, 10, 4, 6, 19, 11, 15, 7, 13	0, 1, 2, 16, 12, 9, 14, 18, 8, 5, 17, 3, 10, 4, 6, 19, 11, 15, 7, 13
8	3	8	16	16, 5, 17, 3, 10, 4, 6, 19, 11, 15, 7, 13	0, 1, 2, 8, 12, 9, 14, 18, 16, 5, 17, 3, 10, 4, 6, 19, 11, 15, 7, 13
7	6	18	14	14, 16, 5, 17, 3, 10, 4, 6, 19, 11, 15, 7, 13	0, 1, 2, 8, 12, 9, 18, 14, 16, 5, 17, 3, 10, 4, 6, 19, 11, 15, 7, 13
6	4	18	12	12, 14, 16, 5, 17, 3, 10, 4, 6, 19, 11, 15, 7, 13	0, 1, 2, 8, 18, 9, 12, 14, 16, 5, 17, 3, 10, 4, 6, 19, 11, 15, 7, 13
5	3	9	8	8, 12, 14, 16, 5, 17, 3, 10, 4, 6, 19, 11, 15, 7, 13	0, 1, 2, 9, 18, 8, 12, 14, 16, 5, 17, 3, 10, 4, 6, 19, 11, 15, 7, 13
4	2	18	2	2, 8, 12, 14, 16, 5, 17, 3, 10, 4, 6, 19, 11, 15, 7, 13	0, 1, 18, 9, 2, 8, 12, 14, 16, 5, 17, 3, 10, 4, 6, 19, 11, 15, 7, 13
3	0	9	0	0, 2, 8, 12, 14, 16, 5, 17, 3, 10, 4, 6, 19, 11, 15, 7, 13	9, 1, 18, 0, 2, 8, 12, 14, 16, 5, 17, 3, 10, 4, 6, 19, 11, 15, 7, 13
2	0	18	9	9, 0, 2, 8, 12, 14, 16, 5, 17, 3, 10, 4, 6, 19, 11, 15, 7, 13	18, 1, 9, 0, 2, 8, 12, 14, 16, 5, 17, 3, 10, 4, 6, 19, 11, 15, 7, 13
1	0	1	18	1, 18, 9, 0, 2, 8, 12, 14, 16, 5, 17, 3, 10, 4, 6, 19, 11, 15, 7, 13	1, 18, 9, 0, 2, 8, 12, 14, 16, 5, 17, 3, 10, 4, 6, 19, 11, 15, 7, 13

Dari hasil analisis algoritma *Fisher Yates* diapatkan hasil bahwa, algoritma ini sangat cocok untuk pengacakan yang nantinya akan digunakan untuk mengacak jawaban.

3.1.5 Spesifikasi Kebutuhan Perangkat Lunak

Spesifikasi kebutuhan perangkat lunak merupakan deskripsi dari apa yang harus sistem miliki untuk memenuhi kebutuhan dari pengguna.

Kode Kebutuhan	Deskripsi
SKPL-F-01	Sistem dapat menyediakan untuk memulai permainan
SKPL-F-02	Sistem dapat menyediakan untuk mencangkul ladang
SKPL-F-03	Sistem dapat menyediakan untuk mencari jalur pada graph
SKPL-F-04	Sistem dapat menyediakan untuk menjawab pertanyaan
SKPL-F-05	Sistem dapat menyediakan untuk memasukan sandi
SKPL-NF-01	Sistem dapat berjalan pada perangkat *mobile* Android
SKPL-NF-02	Sistem dapat berjalan pada layar WVGA 480 x 800 pixel
SKPL-NF-03	Sistem dapat berjalan pada sistem operasi Android minimal versi 2.2 (froyo)

SKPL-NF-04	Sistem dibangun oleh perangkat lunak Android Developer Tools dan Oracle Java Development Kit
SKPL-NF-05	Sistem dibangun oleh bahasa pemrograman Android
SKPL-NF-06	Pembangunan grafis dibangun dengan menggunakan perangkat lunak Corel Draw
SKPL-NF-07	Pembangunan music dibangun dengan menggunakan perangkat lunak Cubase dan Reason
SKPL-NF-08	Sistem dapat digunakan pemain dengan *knowlage and experience*, sebagai berikut : *MobileLiteracy* : Rendah, minimal bisa menggunakan *smartphone* android. Pengalaman Sistem : Sedang, minimal biasa menggunakan sistem operasi Android 2.2 (froyo) Pengalaman Aplikasi : Sedang, minimal pernah memainkan game edukasi dan petualangan sejeneis lainnya. Penggunaan Sistem Lain : Jarang Edukasi : Sekolah Dasar Kemampuan Membaca : Bisa membaca dengan lancar. Setara dengan kemampuan membaca anak kelas 2 SD. Bahasa / Kultur : Bahasa Indonesia
SKPL-NF-09	Sistem mempengaruhi pemain dalam *Job/Task/Need*, sebagai berikut : Frekuensi Penggunaan : *Occasional* Tingkat Kepentingan *Task* : Lemah Interaksi Sosial : tidak diperlukan
SKPL-NF-10	Sistem mempengaruhi pemain dalam *Psychological Characteristics*, sebagai berikut: *Attitude* : Positif Motivasi : Tinggi *Stress Level* : *No Stress*
SKPL-NF-11	Sistem dapa digunakan pemain dengan *Physical Characteristics*, sebagai berikut : Umur : 8 – 11 tahun Jenis Kelamin : Laki-laki dan perempuan *Handedness* : Kanan, Kidal, dan Keduanya *Disabilities* : Tidak ada masalah dengan penglihatan

3.1.6 Analisis Kebutuhan Non Fungsional

Analisis dan kebutuhan non-fungsional meliputi analisis dan kebutuhan perangkat lunak, analisis dan kebutuhan perangkat keras, serta analisis dan kebutuhan user.

3.1.6.1 Analisis Kebutuhan Perangkat Keras

Perangkat keras merupakan suatu alat untuk menjalankan sekumpulan instruksi dari perangkat lunak.

Pada apikasi *game* Kabayan Berhitung ini, perangkat keras yang dibutuhkan untuk menjalankan aplikasi adalah :

1. *Mobile* Android dengan sistem operasi minimal Android 2.2 (froyo)
2. Layar WVGA 480 x 800 pixel

3.1.6.2 Analisis Kebutuhan Perangkat Lunak

Perangkat lunak digunakan dalam sebuah sistem merupakan perintah-perintah yang diberikan kepada perangkat keras agar bisa dapat berinteraksi diantara keduanya.

Perangkat lunak yang dibutuhkan dalam penerapan aplikasi *game* edukasi ini adalah Sistem Operasi Android minimal versi 2.2 (froyo).

Perangkat lunak yang digunakan dalam pembangunan *game* edukasi ini yaitu :

1. Pembangun *game* :
   a. Android Developer Tools (include : Eclipse, Android SDK, ADT Plugins for Eclipse)
   b. Oracle Java Development Kit
2. Bahasa pemograman : Android
3. Grafis : Corel Draw
4. Music : Reason dan Cubase

3.1.6.3 Analisis Pengguna

Analisis pengguna merupakan analisis terhadap pengguna yang akan menggunakan aplikasi yang telah dibangun. Perangkat keras dan perangkat lunak yang ada tidak dapat berguna apabila tidak ada pengguna yang mengoperasikannya. Oleh karena itu dibuatlah analisis pengguna yang berdasarkan pada *user* profilenya, yakni berdasarkan *user knowledge & experience, user job, task & needs, user physicological characteristic, dan user physical characteristic.*

Tabel 3.9 Analisis Pengguna

KNOWLEDGE AND EXPERIENCE	
MobileLiteracy	Rendah, minimal bisa menggunakan smartphone android.
Pengalaman Sistem	Sedang, minimal biasa menggunakan sistem operasi Android 2.2 (froyo)

Pengalaman Aplikasi	Sedang, minimal pernah memainkan game edukasi dan petualangan sejeneis lainnya.
Penggunaan Sistem Lain	Jarang
Edukasi	Sekolah Dasar
Kemampuan Membaca	Bisa membaca dengan lancar. Setara dengan kemampuan membaca anak kelas 2 SD.
Kemampuan Mengetik	-
Bahasa / Kultur	Bahasa Indonesia
JOB / TASK / NEED	
Frekuensi Penggunaan	Occasional
Tingkat Kepentingan Task	Lemah
Interaksi Sosial	Tidak diperlukan
PSYCHOLOGICAL CHARACTERISTICS	
Attitude	Positif
Motivasi	Tinggi
Stress Level	No Stress
PHYSICAL CHARACTERISTICS	
Umur	8 - 11 tahun
Jenis Kelamin	Laki – laki dan Perempuan
Handedness	Kanan, Kidal, dan Keduanya
Disabilities	Tidak ada masalah dengan penglihatan

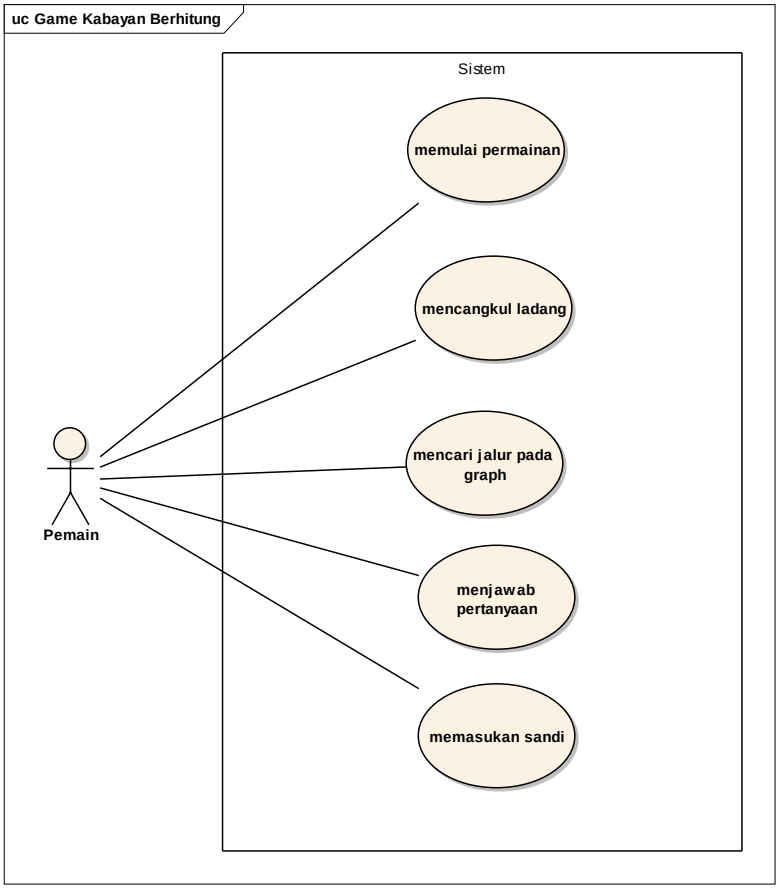
3.1.7 Analisis Kebutuhan Fungsional

Analisis kebutuhan fungsional menggambarkan proses kegiatan yang akan diterapkan dalam sebuah sistem dan menjelaskan kebutuhan yang diperlukan sistem agar sistem dapat berjalan dengan baik sesuai kebutuhan. Pada bagian ini akan dibahas tentang *use case diagram, activity diagram, class diagram, state chart diagram* dan *sequence diagram.*

3.1.7.1 Use Case Diagram

Use Case Diagram adalah diagram yang menggambarkan sebuah kegiatan yang dilakukan oleh *actor* terhadap sistem yang akan dirancang sehingga terjadi

interaksi. Pada bagaian ini akan dijelaskan *Use Case* yang akan dirancang beserta definisinya.



Gambar 3.14 *Use Case Diagram* Kabayan Berhitung

1. *Actor Definition*

Actor Definition berfungsi untuk menjelaskan *Actor* yang terdapat pada *Use Case Diagram*.

Tabel 3.10 *Actor Definition*

No	*Actor*	Deskripsi
1	Pemain	Pemain merupakan orang yang memainkan *game* dan berinteraksi langsung dengan antar muka *game*.

2. *Use Case Definition*

Use Case Definition berfungsi untuk menjelaskan fungsi *Use Case* yang terdapat pada *Use Case Diagram*.

Tabel 3.11 ***Use Case Definition***

No.	*Use case*	Deskripsi
1	Memulai Permainan	**Kode *Use case* :** UC-01 **Deskripsi Singkat:** Proses ini dilakukan untuk memulai permainan **Aktor :** pemain **Trigger :** pemain menyentuh tombol mulai **Prekondisi :** *game* sudah *terinstall*, siap untuk dimainkan, dan berada pada menu utama. **Pascakondisi :** menampilkan cerita kabayan
2	Mencangkul Ladang	**Kode *Use case* :** UC-02 **Deskripsi Singkat:** Proses ini dilakukan ketika pemain akan mencari jawaban di ladang. **Aktor :** pemain **Trigger :** pemain memilih menyentuh ladang **Prekondisi :** permainan berada pada pelatihan 1 **Pascakondisi :** sistem menampilkan angka pada ladang
3	Mencari Jalur Pada Graph	**Kode *Use case* :** UC-03 **Deskripsi Singkat:** Proses ini dilakukan untuk mencari jalur pada graph yang tertampil pada pelatihan 2 **Aktor :** pemain **Trigger :** pemain memilih titik tujuan **Prekondisi :** permainan berada pada pelatihan 2 **Pascakondisi :** sistem menampilkan pesan
4	Menajawab Pertanyaan	**Kode *Use case* :** UC-04 **Deskripsi Singkat:** Proses ini dilakukan untuk menjawab pertanyaan pada pelatihan 2 dan pelatihan 3 **Aktor :** pemain **Trigger :** Pemain menyentuh angka-angka untuk memasukan jawaban **Prekondisi :** permainan berada pada pelatihan 2 atau pelatihan 3 **Pascakondisi :** sistem menampilkan pesan
5	Memasukan Sandi	**Kode *Use case* :** UC-05 **Deskripsi Singkat:** Proses ini dilakukan ketika pemain sudah menjawab pertanyaan pada pelatihan 3 untuk memasukan sandi dari pertanyaan. **Aktor :** pemain **Trigger :** Pemain menyentuh kolom puluhan **Prekondisi :** permainan berada pada pelatihan 3 dan sudah menjawab pertanyaan pelatihan 3 dengan benar **Pascakondisi :** sistem menampilkan pesan

3. *Use Case Scenario*

Use Case Scenario akan menjelaskan alur dari setiap *Use Case* dalam bentuk *text*.

a. *Use Case Scenario* Memulai Permainan

Tabel 3.12 *Use Case Scenario* Memulai Permainan

Kode *Use Case* : UC-01
Scenario normal : 1. Pemain menyentuh tombol mulai 2. Sistem menampilkan cerita kabayan
Scenario Alternatif : Tidak ada

b. *Use Case Scenario* Mencangkul Ladang

Tabel 3.13 *Use Case Scenario* Mencangkul Ladang

Kode *Use Case* : UC-02
Scenario normal : 1. Pemain memilih menyentuh ladang 2. Sistem menampilkan angka pada ladang yang disentuh dan kabayan mencangkul
Scenario Alternatif : Tidak ada

c. *Use Case Scenario* Mencari Jalur Pada Graph

Tabel 3.14 *Use Case Scenario* Mencari Jalur Pada Graph

Kode *Use Case* : UC-03
Scenario normal : 1. Pemain memilih titik tujuan 2. Sistem menambah atau mengurangi jarak yang sudah di tempuh pemain dan menapilkan kabayan bergerak ke titik yang dipilih 3. Sistem memeriksa jarak pemain 4. Sistem menampilkan pesan
Scenario Alternatif : 3a. Posisi pemain belum sapai akhir Pemain mengulang step 1-2 sampai menemukan titik akhir

d. *Use Case Scenario* Menjawab Pertanyaan

Tabel 3.15 *Use Case Scenario* Menjawab Pertanyaan

Kode *Use Case* : UC-04
Scenario normal : 1. Pemain menyentuh angka-angka untuk memasukan jawaban 2. Sistem menampilkan angka yang disentuh pemain di kolom jawaban Pemain mengulang sampai step 1-2 sampai jawaban merasa benar 3. Pemain menyentuh tombol sudah

4. Sistem memeriksa jawaban 5. Sistem menampilkan pesan
Scenario Alternatif : 3a. Pemain menyentuh tombol hapus 3b. Sistem menghampus angka yang sudah ditampilkan Pemain mengulang step 1-2

e. Use Case Scenario Memasukan Sandi

Tabel 3.16 Use Case Scenario Memasukan Sandi

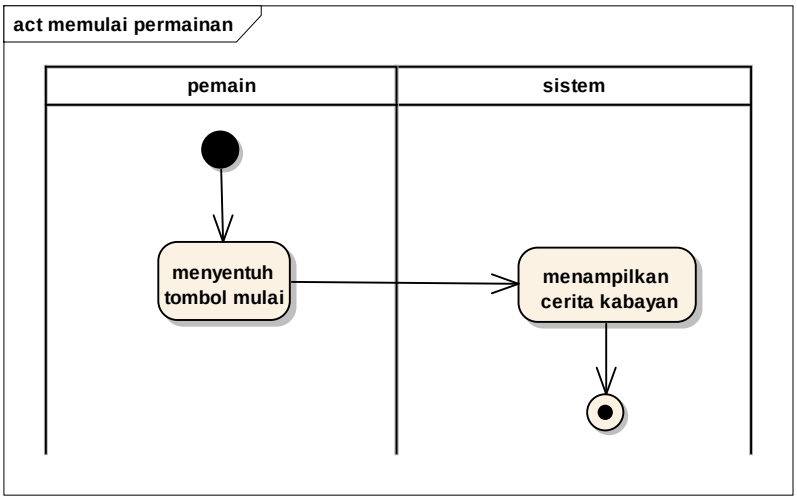
Kode *Use Case* : UC-05
Scenario normal : 1. Pemain menyentuh kolom puluhan 2. Sistem menampilkan kolom puluhan tersorot 3. Pemain menyentuh angka-angka 4. Sistem menampilkan angka yang disentuh pada kolom puluhan 5. Pemain menyentuh kolom satuan 6. Sistem menampilkan kolom satuan tersosrot 7. Pemain menyentuh angka-angka 8. Sistem menampilkan angka yang disentuh pada kolom satuan 9. Pemain menyentuh tombol sudah 10. Sistem memeriksa sandi 11. Sistem menampilkan pesan
Scenario Alternatif : 9a. Pemain menyentuh tombol hapus 1. Kolom puluhan sedang tersorot : a. Sistem menghapus angka pada kolom puluhan 2. Kolom satuan sedang tersorot : a. Sistem menghapus angka pada kolom satuan

3.1.7.2 Activity Diagram

Diagram aktivitas atau *Activity Diagram* menggambarkan *work flow* (aliran kerja) atau aktifitas dari sebuah sistem atau proses bisnis. *Activity diagram* digunakan untuk memperjelas atau mempresentasikan sekenario dari *use case*.

1. *Activity Diagram* Memulai Permainan

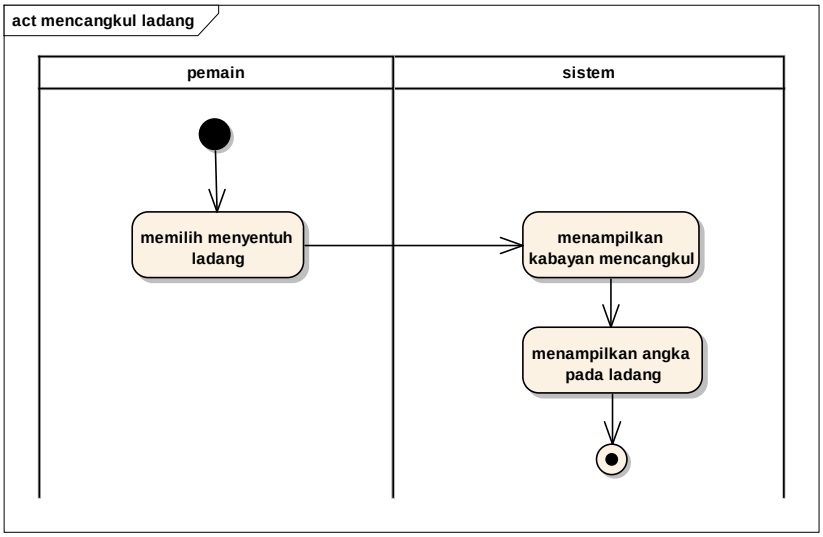
Berikut adalah *activity diagram* memulai permainan.



Gambar 3.15 *Activity Diagram* Memulai Permainan

2. *Activity Diagram* Mencangkul Ladang

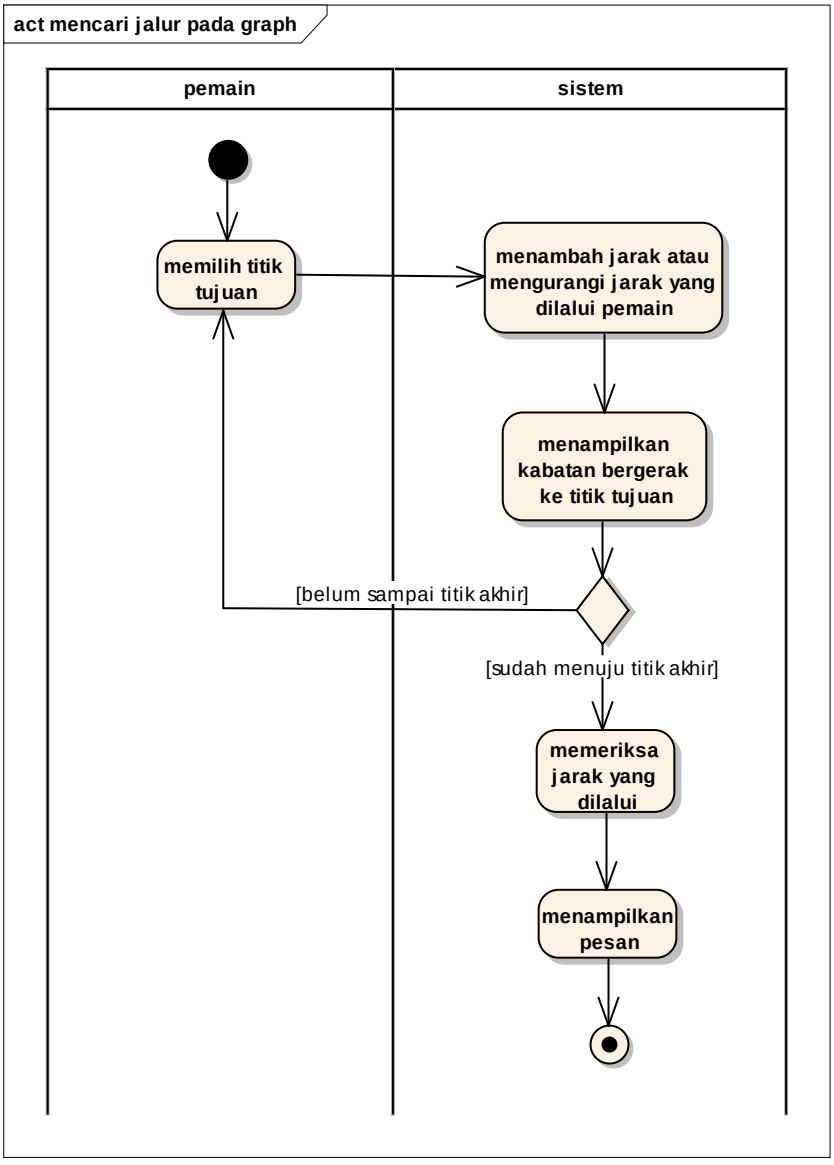
Berikut adalah *activity diagram* mencangkul ladang.



Gambar 3.16 *Activity Diagram* Mencangkul Ladang

3. *Activity Diagram* Mencari Jalur Pada Graph

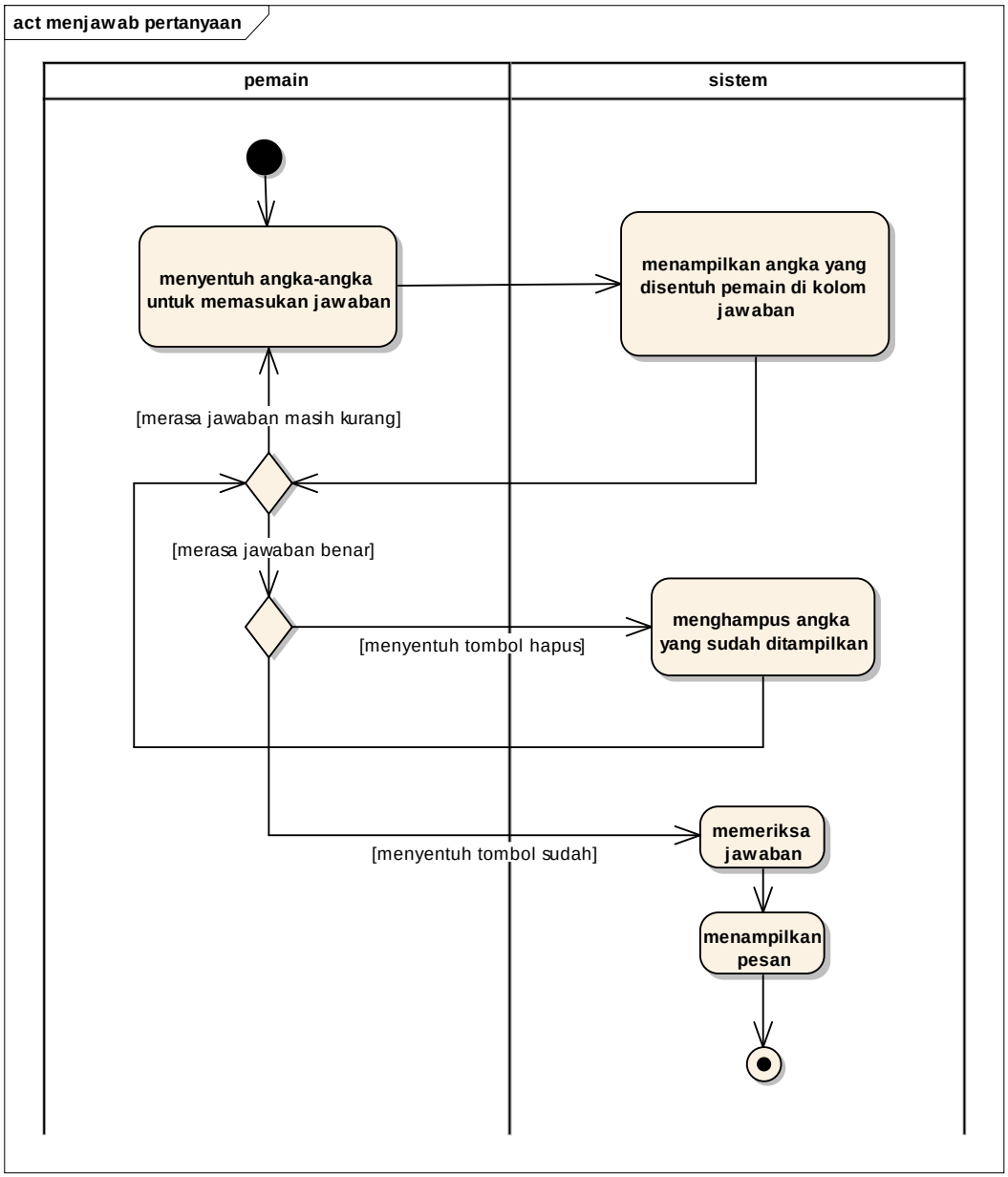
Berikut adalah *activity diagram* mencari jalur pada graph.



Gambar 3.17 *Activity Diagram* Mencari Jalur Pada Graph

4. *Activity Diagram* Menjawab Pertanyaan

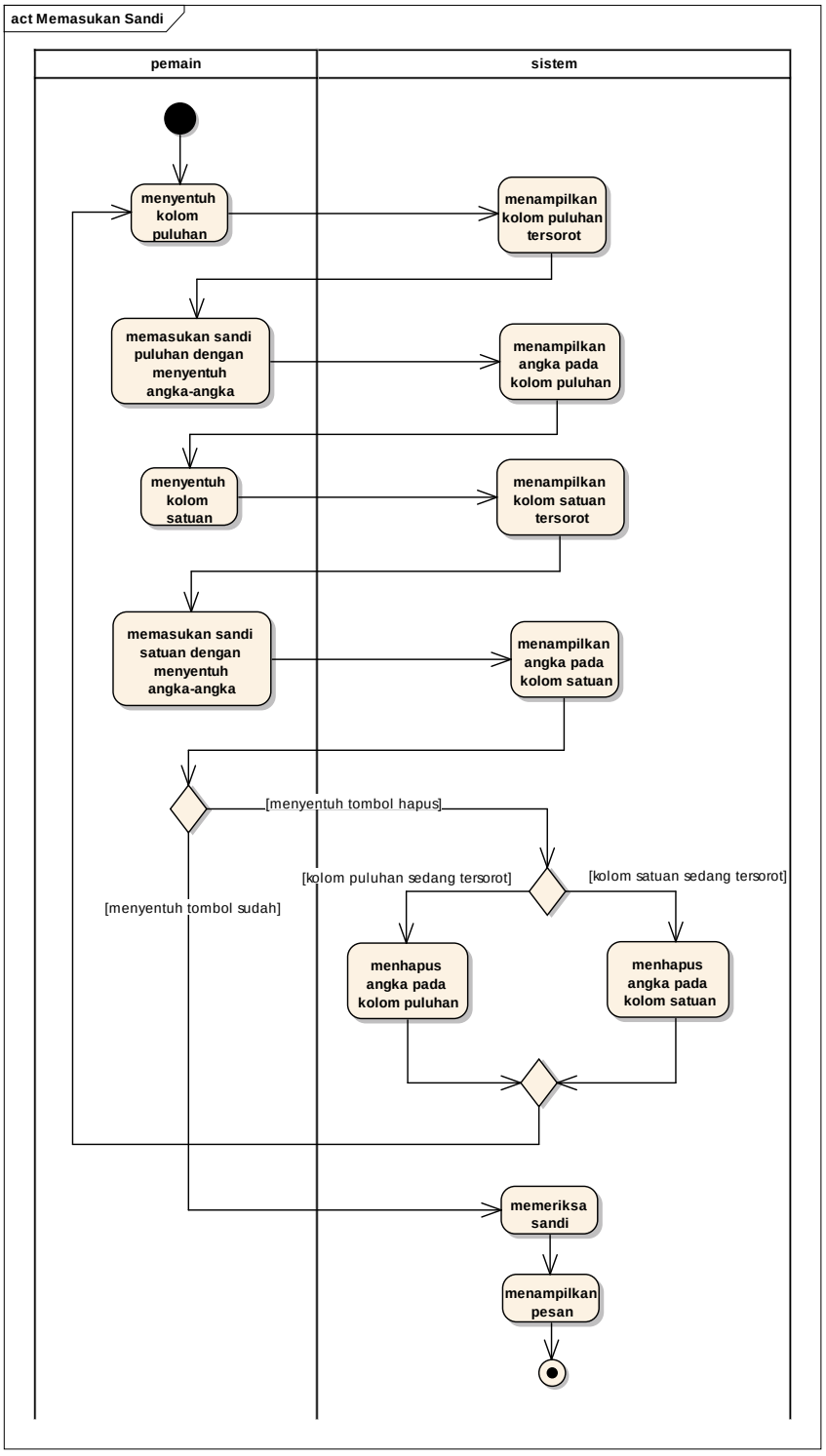
Berikut adalah *activity diagram* menjawab pertanyaan.



Gambar 3.18 *Activity Diagram* Menjawab Pertanyaan

5. *Activity Diagram* Memasukan Sandi

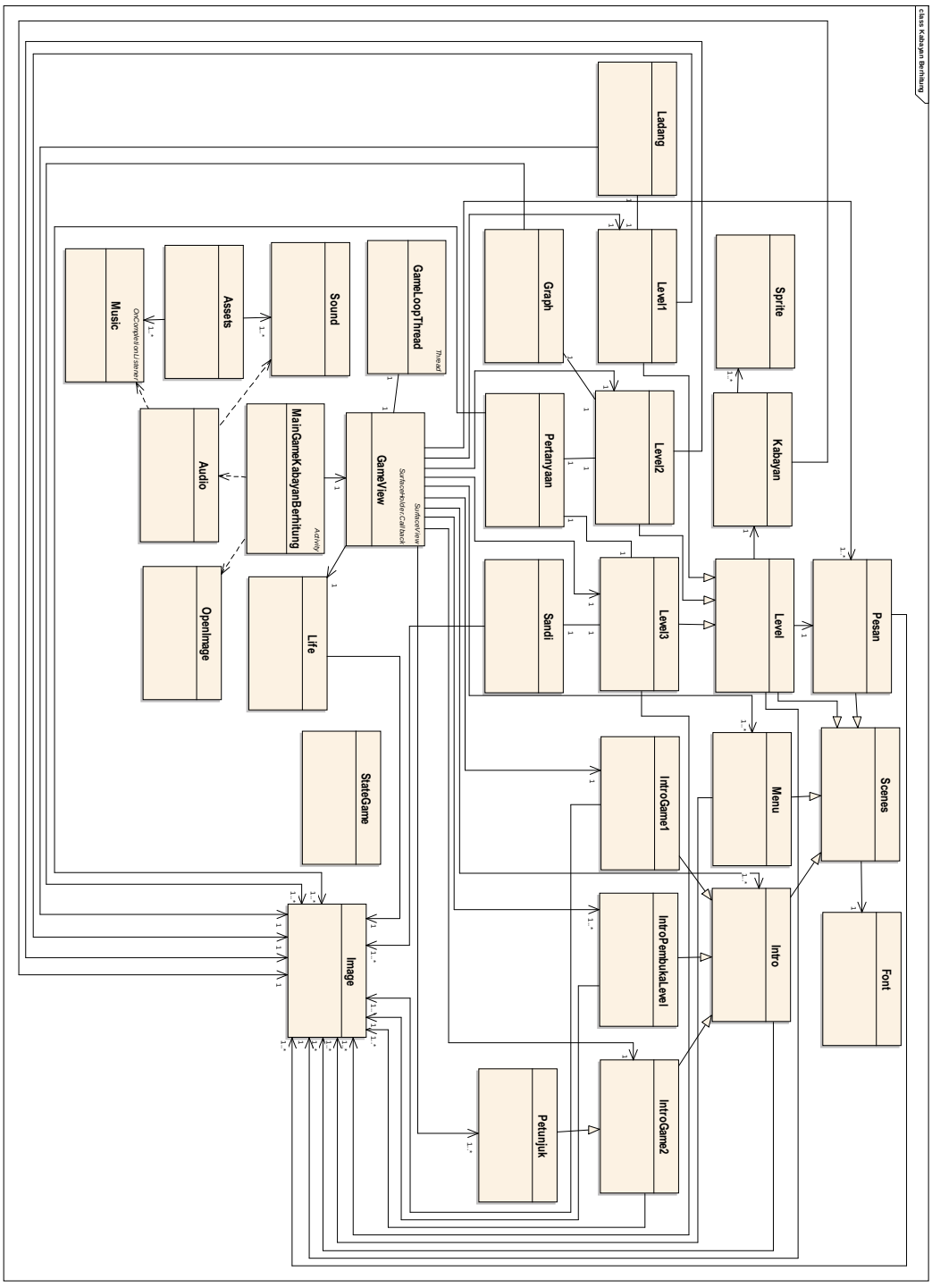
Berikut adalah *activity diagram* memasukan sandi.



Gambar 3.19 *Activity Diagram* Memasukan Sandi

3.1.7.3 Class Diagram

Diagram kelas atau *class diagram* menggambarkan struktur sistem dari segi pendefinisian kelas-kelas yang akan di buat untuk membangun sistem.



Gambar 3.20 *Class Diagram Game* Kabayan Berhitung

Keterangan kelas-kelas pada *class diagram* akan dijelaskan sebagai berikut :

1. Kelas MainGameKabayanBerhitung

Nama Kelas : MainGameKabayanBerhitung

Tabel 3.17 Kelas MainGameKabayanBerhitung

Nama Operasi	Visibility	Keterangan
onCreate(savedInstanceState:Bundle):void	protected	Override dari android.app.Activity
onResume():void	protected	Override dari android.app.Activity
onPause():void	protected	Override dari android.app.Activity
onKeyDown(keyCode:int, event:KeyEvent):void	public	Override dari android.app.Activity
finish():void	public	Override dari android.app.Activity
intialAssets(mainGameKabayanBerhitung :MainGameKabayanBerhitung):void	private	Untuk menginisialisasi Assets
Nama Atribut	**Visibility**	**Tipe**
gameView	public	GameView

2. Kelas GameView

Nama Kelas : GameView

Tabel 3.18 Kelas Game View

Nama Operasi	Visibility	Keterangan
GameView(context:Context)	public	Konstruktor
resume():void	public	Digunakan ketika sistem dalam keadaan resume()
pause():void	public	Digunakan ketika sistem dalam keadaan pause()
CreateGameLoopThread():GameLoopThread	public	Untuk membuat GameLoopThread
surfaceChanged(holder: SurfaceHolder, format:int, width:int, height:int):void	public	Override dari android.view.SurfaceView
surfaceCreated(holder:SurfaceHolder):void	public	Override dari android.view.SurfaceView
surfaceDestroyed(holder:SurfaceHolder):void	public	Override dari android.view.SurfaceView
getGameLoopThread():GameLoopThread	public	Untuk mengambil gameLoopThread
onWindowFocusChanged(hasWindowFocus: boolean):void	public	Override dari android.view.SurfaceView
goAwal():void	public	Untuk menginisialisasi game ke kondisi awal
getStateGame():int	public	Untuk mengambil stateGame yang berjalan

setStateGame(stateGame:int):void	public	Untuk menset stateGame
render(canvas:Canvas):void	public	Untuk menampilkan game dan mengupdate pada layar perangkat
onTouchEvent(event:MotionEvent): boolean	public	Override dari android.view.SurfaceView
cekSound():void	public	Untuk mengecek keadaan musik yang harus dimainakan dalam keadaan state game tertentu
stopAllSound():void	public	Untuk memberhentikan seluruh musik
Nama Atribut	**Visibility**	**Tipe**
gameLoopThread	public	GameLoopThread
holder	private	SurfaceHolder
GO_PREVIOUS_STATE_GAME	public	final static int
GO_EXIT	public	final static int
stateGame	private	static int
previousStateGame	public	static int
kesempatan	public	static int
nilai	public	static int
life	public	static Life
introAdzie	public	static Intro
introGame1	public	static IntroGame1
introGame2	public	static IntroGame2
introPembukaLevel1	public	static IntroPembukaLevel
introPembukaLevel2	public	static IntroPembukaLevel
introPembukaLevel3	public	static IntroPembukaLevel
introLevel1	public	static IntroLevel
introLevel2	public	static IntroLevel
introLevel3	public	static IntroLevel
introGameKabayanBerhitung	public	static Intro
menuUtama	public	static Menu
menuKeluar	public	static Menu
menuPause	public	static Menu
level1	public	static Level1
level2	public	static Level2
level3	public	static Level3
pesanGameEnd	public	static Pesan
pesanGameOver	public	static Pesan

3. Kelas GameLoopThread

Nama Kelas : GameLoopThread

Tabel 3.19 Kelas GameLoopThread

Nama Operasi	Visibility	Keterangan
GameLoopThread(gameView:GameView, holder:SurfaceHolder)	public	Konstruktor
setRunning(boolean run):void	public	Untuk membuat thread jalan
run():void	private	Override dari android.java.lang.Thread
pause():void	public	Digunakan pada saat sistem pause()
Nama Atribut	**Visibility**	**Tipe**
FPS	public	static final long
running	public	Boolean
view	public	GameView
surfaceHolder	private	SurfaceHolder

4. Kelas Scenes

Nama Kelas : Scenes

Tabel 3.20 Kelas Scanes

Nama Operasi	Visibility	Keterangan
Scenes(stateStart:int)	public	Konstruktor
startScanes():void	public	Untuk memulai Scanes
updateScenes():void	public	Untuk mengupadate keadaan di scanes
onTouchScenes(event:MotionEvent):boolean	public	Untuk menerima inputan terhadap scanes
getStateScenes():int	public	Untuk mengambil state pada scanes
setStateScenes(stateScenes:int):void	public	Untuk mengeset state pada scanes
drawScenes(canvas:Canvas):void	public	Untuk menampilkan scanes
drawScenes(canvas:Canvas, pesan:String, nilai:int):void	public	Untuk menampilkan scanes
drawScenes(canvas:Canvas, kesempatan:int, nilai:int):void	public	Untuk menampilkan scanes
Nama Atribut	**Visibility**	**Tipe**
stateScenes	private	int
stateStart	public	int
font	public	Font

5. Kelas Image

Nama Kelas : Image

Tabel 3.21 Kelas Image

Nama Operasi	Visibility	Keterangan
Image(bitmap:Bitmap, x:int, y:int)	public	Konstruktor
getBitmap():Bitmap	public	Untuk mengambil Bitmap
setBitmap(bitmap:Bitmap):void	public	Untuk mengeset Bitmap
getX():int	public	Untuk mengambil nilai x
setX(x:int):int	public	Untuk mengeset nilai x
getY():int	public	Untuk mengambil nilai y
setY(int y):int	public	Untuk mengeset nilai y
draw(canvas:Canvas):void	public	Untuk menampilkan gambar
isCollition(eventX:float, eventY:float):boolean	public	Untuk mengecek sentuhan terhadap gambar
Nama Atribut	**Visibility**	**Tipe**
bitmap	private	Bitmap
x	private	int
y	private	int

6. Kelas Sprite

Nama Kelas : Sprite

Tabel 3.22 Kelas Sprite

Nama Operasi	Visibility	Keterangan
Sprite(bitmap:Bitmap, x:int, y:int, width:int, height:int, fps:int, frameCount: int)	public	Konstruktor
getBitmap():Bitmap	public	Untuk mengambil Bitmap
setBitmap(bitmap:Bitmap):void	public	Untuk mengeset Bitmap
getSourceRect():Rect	public	Untuk mengambil Rect
setSourceRect(sourceRect:Rect):void	public	Untuk mengeset Rect
getFrameNr():int	public	Untuk mengambil frameNr
setFrameNr(frameNr:int):void	public	Untuk mengeset frameNr
getCurrentFrame():int	public	Untuk mengambil currentFrame
setCurrentFrame(currentFrame:int):void	public	Untuk mengeset currentFrame
getFramePeriod():int	public	Untuk mengambil FramePeriod
setFramePeriod(framePeriod:int):void	public	Untuk mengeset FramePeriod
getSpriteWidth():int	public	Untuk mengambil SpriteWidth
setSpriteWidth(spriteWidth:int):void	public	Untuk mengeset SpriteWidth
getSpriteHeight():int	public	Untuk mengambil spriteHeight
setSpriteHeight(spriteHeight:int):void	public	Untuk mengeset spriteHeight
getX():int	public	Untuk mengambil nilai x
setX(x:int):int	public	Untuk mengeset nilai x
getY():int	public	Untuk mengambil nilai y

setY(int y):int	public	Untuk mengeset nilai y
update(gameTime:long):void	public	Untuk mengupdate gerakan
draw(canvas:Canvas):void	public	Untuk menampilkan gambar
Nama Atribut	**Visibility**	**Tipe**
bitmap	private	Bitmap
x	private	Int
y	private	int
frameNr	private	int
currentFrame	private	int
frameTicker	private	long
framePeriod	private	int
spriteWidth	private	int
spriteHeight	private	int

7. Kelas OpenImage

Nama Kelas : OpenImage

Tabel 3.23 Kelas OpenImage

Nama Operasi	Visibility	Keterangan
OpenImage(context:Context)	public	Konstruktor
open(pathname:String):Bitmap	public	Untuk membuka file image (.png)
Nama Atribut	**Visibility**	**Tipe**
manager	public	AssetManager

8. Kelas Audio

Nama Kelas : Audio

Tabel 3.24 Kelas Audio

Nama Operasi	Visibility	Keterangan
Audio(activity:Activity)	public	Konstruktor
newMusic(filename:String):Music	public	Untuk membuat Music
newSound(filename:String):Sound	public	Untuk membuat Sound
Nama Atribut	**Visibility**	**Tipe**
assets	public	AssetManager
soundPool	public	SoundPool

9. Kelas Music

Nama Kelas : Music

Tabel 3.25 Kelas Music

Nama Operasi	Visibility	Keterangan
Music(assetDescriptor:AssetFileDescriptor)	public	Konstruktor
dispose():void	public	Untuk mengalihkan

isLooping():boolean	public	Untuk memeriksa musik sedang berulang
isPlaying():boolean	public	Untuk memeriksa musik sedang bermain
isStopped():boolean	public	Untuk memeriksa musik tidak bermain
pause():void	public	Untuk pause musik
play():void	public	Untuk memainkan musik
setLooping(isLooping:boolean):void	public	Untuk mengeset perulangan
setVolume(volume:float):void	public	Untuk mengeset volume
stop():void	public	Untuk memberhentikan musik
onCompletion(player:MediaPlayer): void	public	Override dari android.media.MediaPlayer.OnCompletionListener
Nama Atribut	**Visibility**	**Tipe**
mediaPlayer	public	MediaPlayer
isPrepared	public	boolean

10. Kelas Sound

Nama Kelas : Sound

Tabel 3.26 Kelas Sound

Nama Operasi	Visibility	Keterangan
Sound(soundPool:SoundPool, soundId:int)	public	Konstruktor
dispose():void	public	Untuk mengalihkan
Nama Atribut	**Visibility**	**Tipe**
soundId	public	int
soundPool	public	SoundPool

11. Kelas Menu

Nama Kelas : Menu

Tabel 3.27 Kelas Menu

Nama Operasi	Visibility	Keterangan
Menu(stateStart:int, stateFinish:int, stateFinish2:int, background:Bitmap, button1:Bitmap, xButton1:int, yButton1:int, button2:Bitmap, xButton2:int, yButton2:int)	public	Konstruktor
startScanes():void	public	Override dari Kelas Scenes
drawScenes(canvas:Canvas):void	public	Override dari Kelas Scenes
onTouchScenes(event:MotionEvent):boolean	public	Override dari Kelas Scenes
Nama Atribut	**Visibility**	**Tipe**
stateFinish	public	int

background	public	Image
button1	public	Image
button2	public	Image
stateFinish2	public	int

12. Kelas Pesan

Nama Kelas : Pesan

Tabel 3.28 Kelas Pesan

Nama Operasi	Visibility	Keterangan
Pesan(stateStart:int, stateFinish:int, background:Bitmap, button:Bitmap)	public	Konstruktor
startPesan(pesan:String, nilai:int):void	public	Untuk memulai menampilkan pesan
hidePesan():void	public	Untuk menyembunyikan pesan
drawScenes(canvas:Canvas):void	public	Override dari Kelas Scenes
onTouchScenes(event:MotionEvent):boolean	public	Override dari Kelas Scenes
Nama Atribut	**Visibility**	**Tipe**
stateFinish	public	int
background	public	Image
button	public	Image
bBackground	public	Bitmap
bButton	public	Bitmap
pesan	public	String
nilai	public	String
xNilai	public	int
yNilai	public	int

13. Kelas Intro

Nama Kelas : Intro

Tabel 3.29 Kelas Intro

Nama Operasi	Visibility	Keterangan
Intro(stateStart:int, stateFinish:int, background:Bitmap, button:Bitmap, endTime:int)	public	Konstruktor
startScanes():void	public	Override dari Kelas Scenes
drawScenes(canvas:Canvas):void	public	Override dari Kelas Scenes
updateScenes():void	public	Override dari Kelas Scenes
onTouchScenes(event:MotionEvent):boolean	public	Override dari Kelas Scenes
Nama Atribut	**Visibility**	**Tipe**
stateFinish	public	int
background	public	Image
button	public	Image
time	public	int

endTime	public	int
runningTime	public	boolean
activeTouch	public	boolean

14. Kelas Level

Nama Kelas : Level

Tabel 3.30 Kelas Level

Nama Operasi	Visibility	Keterangan
Level(stateStart:int)	public	Konstruktor
drawScenes(canvas:Canvas):void	public	Override dari Kelas Scenes
updateScenes():void	public	Override dari Kelas Scenes
onTouchScenes(event:MotionEvent):boolean	public	Override dari Kelas Scenes
getInternalState():int	public	Untuk mengambil internalState
setInternalState(internalState:int):void	public	Untuk mengeset internalState
Nama Atribut	**Visibility**	**Tipe**
STATE_LEVEL	public	static final int
STATE_PESAN	public	static final int
STATE_SUB1_LEVEL	public	static final int
STATE_SUB2_LEVEL	public	static final int
internalState	private	int
background	public	Image
kesempatan	public	int
nilai	public	int
kabayan	public	Kabayan
activeTouch	public	boolean
pesan	public	Pesan
stringPesan	public	String

15. Kelas Kabayan

Nama Kelas : Kabayan

Tabel 3.31 Kelas Kabayan

Nama Operasi	Visibility	Keterangan
Kabayan()	public	Konstruktor
startKabayan():void	public	Untuk memulai kabayan
setKabayan(x:int, y:int) :void	public	Untuk mengeset posisi kabayan
hideKabayan():void	public	Untuk menyembunyikan kabayan
drawKabayan(canvas:Canvas):void	public	Untuk menampilkan kabayan
startKabayanCangkul(x:int, y:int):void	public	Untuk memulai kabayan mencangkul

hideKabayanCangkul():void	public	Untuk menyembunyikan kabayan mencangkul
updateKabayanCangkul():void	public	Untuk mengupdate gerkan kabayan mencangkul
drawKabayanCangkul(canvas:Canvas):void	public	Untuk menampilkan kabayan mencangkul
startKabayanJalan(bitmap:Bitmap):void	public	Untuk memulai kabayan cangkul
hideKabayanJalan():void	public	Untuk menyembunyikan kabayan jalan
setKabayanJalan(x:int, y:int):void	public	Untuk mengeset posisi kabayan jalan
drawKabayanJalan(canvas:Canvas):void	public	Untuk menampilkan kabayan jalan
Nama Atribut	**Visibility**	**Tipe**
kabayan	public	Image
kabayanCangkul	public	Sprite
kabayanJalan	public	Sprite
activeKabayanJalan	public	boolean
kabayanTime	public	int
activeKabayanCangkul	public	boolean

16. Kelas IntroGame1

Nama Kelas : IntroGame1

Tabel 3.32 Kelas IntroGame1

Nama Operasi	Visibility	Keterangan
IntroGame1(stateStart:int, stateFinish:int, background:Bitmap, button:Bitmap, endTime:int)	public	Konstruktor
createIntroGame1():void	public	Untuk membuat kembali intro game 1
updateScenes():void	public	Override dari Kelas Intro
drawScenes(canvas:Canvas):void	public	Override dari Kelas Intro
onTouchScenes(event:MotionEvent):boolean	public	Override dari Kelas Intro
Nama Atribut	**Visibility**	**Tipe**
intro1	public	Image
intro2	public	Image
intro3	public	Image
intro4	public	Image
countTampilIntro	public	int

17. Kelas IntroGame2

Nama Kelas : IntroGame2

Tabel 3.33 Kelas IntroGame2

Nama Operasi	Visibility	Keterangan
IntroGame2(stateStart:int, stateFinish:int, background:Bitmap, button:Bitmap, endTime:int)	public	Konstruktor
createIntroGame2():void	public	Untuk membuat kembali intro game 2
updateScenes():void	public	Override dari Kelas Intro
drawScenes(canvas:Canvas):void	public	Override dari Kelas Intro
onTouchScenes(event:MotionEvent):boolean	public	Override dari Kelas Intro
Nama Atribut	**Visibility**	**Tipe**
intro1	public	Image
intro2	public	Image
intro3	public	Image
countTampilIntro	public	int

18. Kelas IntroLevel

Nama Kelas : IntroLevel

Tabel 3.34 Kelas IntroLevel

Nama Operasi	Visibility	Keterangan
IntroLevel(stateStart:int, stateFinish:int, background:Bitmap, button:Bitmap, endTime:int, level:int)	public	Konstruktor
createIntroGame2():void	public	Override dari Kelas IntroGame2
updateScenes():void	public	Override dari Kelas IntroGame2
drawScenes(canvas:Canvas):void	public	Override dari Kelas IntroGame2
onTouchScenes(event:MotionEvent):boolean	public	Override dari Kelas IntroGame2
Nama Atribut	**Visibility**	**Tipe**
level	public	int
countClick	public	int

19. Kelas IntroPembukaLevel

Nama Kelas : IntroPembukaLevel

Tabel 3.35 Kelas IntroPembukaLevel

Nama Operasi	Visibility	Keterangan
IntroPembukaLevel (stateStart:int, stateFinish:int, background:Bitmap, button:Bitmap, endTime:int, text:Bitmap)	public	Konstruktor
drawScenes(canvas:Canvas):void	public	Override dari Kelas Intro

Nama Atribut	Visibility	Tipe
text	public	Image

20. Kelas Level1

Nama Kelas : Level1

Tabel 3.36 Kelas Level1

Nama Operasi	Visibility	Keterangan
Level1(stateStart:int)	public	Konstruktor
createLevel1():void	public	Untuk membuat kembali level 1
startLevel1():void	public	Untuk memulai level 1
fisherYates(array:int[]):int[]	public	Untuk mengacak jawaban
createSoal():void	public	Untuk membuat soal
randomSoal():void	public	Untuk membuat soal acak
startSoal():void	public	Untuk menampilkan soal
drawScenes(canvas:Canvas):void	public	Override dari Kelas Level
updateScenes():void	public	Override dari Kelas Level
onTouchScenes(event:MotionEvent):boolean	public	Override dari Kelas Level
Nama Atribut	**Visibility**	**Tipe**
countPertanyaan	public	int
soalA	public	int
soalB	public	Int
soalTemp	public	int
soalHasil	public	int
soalOperator	public	char[]
op	public	int
soal	public	String
jawab	public	int
jawaban	public	int[]
activeTime	public	boolean
time	public	int
timeMenjawab	public	int
buttonSudah	public	Image
activeButtonSudah	public	boolean
ladang	public	Ladang

21. Kelas Level2

Nama Kelas : Level2

Tabel 3.37 Kelas Level2

Nama Operasi	Visibility	Keterangan
Level2(stateStart:int)	public	Konstruktor
createLevel2():void	public	Untuk membuat kembali level 2
startLevel2():void	public	Untuk memulai level 2
createPenghubung():void	public	Untuk membuat penghubung

startPenghubung():void	public	Untuk memulai menampilkan penghubung
hidePenghubung():void	public	Untuk menutup penghubung
drawPenghubung(canvas:Canvas):void	public	Untuk menampilkan penghubung dilayar
gerak(x_a:int, y_a:int):void	public	Untuk menggerakan karakter kabayan
cekJawaban():void	public	Untuk mengecek jawaban
getStateGraph():int	public	Untuk mengambil stateGraph
setStateGraph(stateGraph:int):void	public	Untuk mengeset stateGraph
drawScenes(canvas:Canvas):void	public	Override dari Kelas Level
updateScenes():void	public	Override dari Kelas Level
onTouchScenes(event:MotionEvent):boolean	public	Override dari Kelas Level
Nama Atribut	**Visibility**	**Tipe**
STATE_GRAPH1	public	final static int
STATE_GRAPH2	public	final static int
STATE_GRAPH3	public	final static int
STATE_FINISH	public	final static int
STATE_PENGHUBUNG	public	final static int
stateGraph	private	int
keterangan1	public	String
keterangan2	public	String
jarakYangSudahDitempuh	public	int
x0	public	int
x1	public	int
y0	public	int
y1	public	int
dx	public	int
dy	public	int
x_tambah	public	int
y_tambah	public	int
x_tambah_tmp	public	float
y_tambah_tmp	public	float
step	public	float
step_tmp	public	float
speed	public	float
k	public	int
pertanyaan	public	Pertanyaan
penghubung	public	Image

22. Kelas Level3

Nama Kelas : Level3

Tabel 3.38 Kelas Level3

Nama Operasi	Visibility	Keterangan
Level3(stateStart:int)	public	Konstruktor
createLevel3():void	public	Untuk membuat kembali level 3
startLevel3():void	public	Untuk memulai level 3
createPintu1():void	public	Untuk membuat pintu 1
startPintu1():void	public	Untuk memulai pintu 1
hidePintu1():void	public	Untuk myembunyikan pintu 1
createPintu2():void	public	Untuk membuat pintu 2
startPintu2():void	public	Untuk memulai pintu 2
hidePintu2():void	public	Untuk myembunyikan pintu 2
createPintu3():void	public	Untuk membuat pintu 3
startPintu3():void	public	Untuk memulai pintu 3
hidePintu3():void	public	Untuk myembunyikan pintu 3
showPintu():void	public	Untuk menampilkan pintu
cekJawaban():void	public	Untuk mengecek jawaban
cekSandi():void	public	Untuk mngecek sandi
getStatePintu():int	public	Untuk mengambil statePintu
setStatePintu(int statePintu):void	public	Untuk mengeset statePintu
drawScenes(canvas:Canvas):void	public	Override dari Kelas Scenes
updateScenes():void	public	Override dari Kelas Scenes
onTouchScenes(event:MotionEvent):boolean	public	Override dari Kelas Scenes
Nama Atribut	**Visibility**	**Tipe**
STATE_GRAPH1	public	final static int
STATE_GRAPH2	public	final static int
STATE_GRAPH3	public	final static int
STATE_FINISH	public	final static int
statePintu	private	int
pintu1	public	Image
pintu2	public	Image
pintu3	public	Image
pagar1	public	Image
pagar2	public	Image
sandi	public	Sandi
pertanyaan	public	Pertanyaan

23. Kelas Ladang

Nama Kelas : Ladang

Tabel 3.39 Kelas Ladang

Nama Operasi	Visibility	Keterangan
Ladang(level1:Level1)	public	Konstruktor
startLadang():void	public	Untuk memulai menampilkan ladang
draw(canvas:Canvas):void	public	Untuk menampilkan ladang
touch(event:MotionEvent):void	public	Untuk menyentuh ladang
touchPetakLadang(index:int, xKabayanCangkul:int, yKabayanCangkul:int):void	public	Untuk menyentuh setiap petak ladang
Nama Atribut	**Visibility**	**Tipe**
level1	public	Level1
petak	public	Image[]
textPetak	public	String[]
currentPetak	public	int

24. Kelas Graph

Nama Kelas : Graph

Tabel 3.40 Kelas Graph

Nama Operasi	Visibility	Keterangan
Graph(level2:Level2)	public	Konstruktor
inisialisasiIntegerArray(integers: int[]):void	public	Untuk menginisialisasi array integer
inisialisasiStringWeight(array:String[]):void	public	Untuk menginisialisasi array String
beriNilaiStringWeight(array:String[], integer:int[]):void	public	Untuk memberi nilai Weigth dalam bentuk string
startGraph(image:Image[]):void	public	Untuk memulai menampilkan graph
hideGraph(images:Image[]):void	public	Untuk menyembunyikan graph
drawGraph(canvas:Canvas, images:Image[]):void	public	Untuk menampilkan graph
startGraph1():void	public	Untuk memulai menampilkan graph 1
hideGraph1():void	public	Untuk menyembunyikan graph 1
drawGraph1(canvas:Canvas):void	public	Untuk menampilkan graph 1
displaySolutionGraph1():void	public	Untuk menampilkan solusi graph 1
startGraph2():void	public	Untuk memulai menampilkan graph 2

hideGraph2():void	public	Untuk menyembunyikan graph 2
drawGraph2(canvas:Canvas):void	public	Untuk menampilkan graph 2
displaySolutionGraph2():void	public	Untuk menampilkan solusi graph 2
startGraph3():void	public	Untuk memulai menampilkan graph 3
hideGraph3():void	public	Untuk menyembunyikan graph 3
drawGraph3(canvas:Canvas):void	public	Untuk menampilkan graph 3
displaySolutionGraph3():void	public	Untuk menampilkan solusi graph 3
dijkstra(d:int[][], from:int):void	public	Untuk mencari jarak minimum di setiap graph
reconstructorPredecessor(sumVertex:int, s:int):void	public	Untuk merubah pembacaan dari hasil dijkstra
touchGraph(event:MotionEvent):void	public	Untuk menerima inputan
Nama Atribut	**Visibility**	**Tipe**
level2	public	Level2
Infinity	public	static int
currentVertex	public	int
fromVertex	public	int
graph1	public	Image
graph1Vertex	public	Image[]
graph1Weight	public	int[]
graph1WeightString	public	String[]
graph1Route	public	int[][]
graph1SolutionRoute	public	int[]
graph1MinDistance	public	int
graph1HasilRute	public	int[]
graph2	public	Image
graph2Vertex	public	Image[]
graph2Weight	public	int[]
graph2WeightString	public	String[]
graph2Route	public	int[][]
graph2SolutionRoute	public	int[]
graph2MinDistance	public	int
graph2HasilRute	public	int[]
graph3	public	Image
graph3Vertex	public	Image[]
graph3Weight	public	int[]
graph3WeightString	public	String[]
graph3Route	public	int[][]
graph3SolutionRoute	public	int[]
graph3MinDistance	public	int
graph3HasilRute	public	int[]
distances	public	int[]

predecessors	public	int[]
solution	public	int[]
minDistance	public	int

25. Kelas Pertanyaan

Nama Kelas : Pertanyaan

Tabel 3.41 Kelas Pertanyaan

Nama Operasi	Visibility	Keterangan
Pertanyaan(level2:Level2, level3:Level3)	public	Konstruktor
createSoal():void	public	Untuk membuat kembali soal
startSoal():void	public	Untuk memulai menampilkan soal
createDisplayPertanyaan():void	public	Untuk membuat displat pertanyaan
startDisplayPertanyaan():void	public	Untuk menampilkan pertanyaan
hideDisplayPertanyaan():void	public	Untuk menyembunyikan pertanyaan
hapusAngkaJawaban():void	public	Untuk menghapus jawaban yang tertera pada kolom jawaban
tambahAngkaJawaban(i:int):void	public	Untuk menambah ankga pada kolom jawaban
drawDisplayPertanyaan(canvas:Canvas):void	public	Untuk menampilkan pertanyaan
touchPertanyaan(event:MotionEvent):void	public	Untuk menerima inputan
Nama Atribut	**Visibility**	**Tipe**
level2	public	Level2
level3	public	Level3
dialogAbah	public	Image
editText	public	Image
number	public	Image[]
numberBackground	public	Image
buttonHapus	public	Image
buttonSudah	public	Image
soalA	public	int
soalB	public	int
soalC	public	int
soalTemp	public	int

soalHasil	public	int
soalHasilKaliBagi	public	int
soalHasilTamabahKurang	public	int
soalHasilPuluhan	public	int
soalHasilSatuan	public	int
soalOperator1	public	char[]
op1	public	int
soalOperator2	public	char[]
op2	public	int
soal	public	String
jawab	public	String

26. Kelas Sandi

Nama Kelas : Sandi

Tabel 3.42 Kelas Sandi

Nama Operasi	Visibility	Keterangan
Sandi(level3:Level3)	public	Konstruktor
createSandi():void	public	Untuk membuat kembali sandi
startDisplaySandi():void	public	Untuk memulai menampilkan sandi
hideDisplaySandi():void	public	Untuk menyembunyikan sandi
drawDisplaySandi(canvas:Canvas):void	public	Untuk menampilkan sandi
touchSandi(event:MotionEvent):void	public	Untuk menerima inputan
hapusAngkaPuluhan():void	public	Untuk menghapus angka dikolom puluhan
hapusAngkaSatuan():void	public	Untuk menghapus angka dikolom satuan
hitPuluhan():void	public	Untuk menghit kolom puluhan
hitSatuan():void	public	Untuk menghit kolom satuan
tampilAngkaPuluhan(i:int):void	public	Untuk menampilkan angka pada kolom puluhan
tampilAngkaSatuan(i:int):void	public	Untuk menampilkan angka pada kolom satuan
Nama Atribut	**Visibility**	**Tipe**
level3	public	Level3
subBackGround	public	Image
subeditTextPuluhan	public	Image
subeditTextSatuan	public	Image
subnumber	public	Image[]
subnumberBackground	public	Image
subbuttonHapus	public	Image
subbuttonSudah	public	Image
activeEditTextPuluhan	public	boolean

activeEditTextSatuan	public	boolean
kunci	public	String
jawabPuluhan	public	String
jawabSatuan	public	String
soalHasil	public	int

27. Kelas Life

Nama Kelas : Life

Tabel 3.43 Kelas Life

Nama Operasi	Visibility	Keterangan
Life()	public	Konstruktor
setKesempatan(life1:Bitmap, life2:Bitmap, life3:Bitmap, life4:Bitmap):void	public	Untuk mengeset kesempatan yang tampil
drawLife(canvas:Canvas, kesempatan:int):void	public	Untuk menampilkan kesempatan
Nama Atribut	**Visibility**	**Tipe**
kesempatan	public	Image
iconKesempatan1	public	Image
iconKesempatan2	public	Image
iconKesempatan3	public	Image
iconKesempatan4	public	Image

28. Kelas Font

Nama Kelas : Font

Tabel 3.44 Kelas Font

Nama Operasi	Visibility	Keterangan
Font(font1:Typeface, font2:Typeface)	public	Konstruktor
fontPesan():void	public	Untuk mengeset font pesan
fontLevel1Soal():void	public	Untuk mengeset font Level1Soal
fontLevel1Time():void	public	Untuk mengeset font Level1Time
fontLevel1Sawah():void	public	Untuk mengeset font Level1Sawah
getFont():TextPaint	public	Untuk mendapatkan font
fontLevel2Wieight():void	public	Untuk mengeset font Level2Wieight
fontLevel2Keterangan():void	public	Untuk mengeset font Level2Keterangan
fontLevel2Soal():void	public	Untuk mengeset font Level2Soal
fontLevel3Soal():void	public	Untuk mengeset font Level3Soal
Nama Atribut	**Visibility**	**Tipe**
font	public	TextPaint
font1	public	Typeface
font2	public	Typeface

29. Kelas Assets

Nama Kelas : Assets

Tabel 3.45 Kelas Assets

Nama Operasi	Visibility	Keterangan
-	-	-
Nama Atribut	**Visibility**	**Tipe**
kesempatan	public	static Bitmap
iconKesempatan	public	static Bitmap
kosong	public	static Bitmap
level1Background	public	static Bitmap
level1PetakSawahBelumDicangkul	public	static Bitmap
level1PetakSawahSudahDicangkul	public	static Bitmap
level2Background	public	static Bitmap
level2Graph1	public	static Bitmap
level2Graph2	public	static Bitmap
level2Graph3	public	static Bitmap
level2VertexBiru	public	static Bitmap
level2VertexCokelat	public	static Bitmap
level2VertexKuning	public	static Bitmap
level2VertexRumah	public	static Bitmap
level3Background	public	static Bitmap
level3Pagar1	public	static Bitmap
level3Pagar2	public	static Bitmap
level3Pintu1	public	static Bitmap
level3Pintu2	public	static Bitmap
level3Pintu3	public	static Bitmap
level3SubBackground	public	static Bitmap
buttonCek	public	static Bitmap
buttonKeluar	public	static Bitmap
buttonKeluar2	public	static Bitmap
buttonLanjutkan	public	static Bitmap
buttonMulai	public	static Bitmap
buttonNext	public	static Bitmap
buttonSudah	public	static Bitmap
buttonTidak	public	static Bitmap
buttonYa	public	static Bitmap
kabayan	public	static Bitmap
kabayanCangkul	public	static Bitmap
kabayanJalanKanan	public	static Bitmap
kabayanJalanKiri	public	static Bitmap
introAdzie	public	static Bitmap
introKabayanBerhitung	public	static Bitmap
introBackground	public	static Bitmap
intro1_1	public	static Bitmap
intro1_2	public	static Bitmap
intro1_3	public	static Bitmap
intro1_4	public	static Bitmap

intro2_1	public	static Bitmap
intro2_2	public	static Bitmap
intro2_3	public	static Bitmap
introLevelHeading	public	static Bitmap
introLevel1_1	public	static Bitmap
introLevel1_2	public	static Bitmap
introLevel2_1	public	static Bitmap
introLevel2_2	public	static Bitmap
introLevel3_1	public	static Bitmap
introLevel3_2	public	static Bitmap
pembukaLevelBackground	public	static Bitmap
pembukaText1	public	static Bitmap
pembukaText2	public	static Bitmap
pembukaText3	public	static Bitmap
menuKeluar	public	static Bitmap
menuPause	public	static Bitmap
menuUtama	public	static Bitmap
pesanGagal	public	static Bitmap
pesanTamat	public	static Bitmap
number0	public	static Bitmap
number1	public	static Bitmap
number2	public	static Bitmap
number3	public	static Bitmap
number4	public	static Bitmap
number5	public	static Bitmap
number6	public	static Bitmap
number7	public	static Bitmap
number8	public	static Bitmap
number9	public	static Bitmap
numberBackground	public	static Bitmap
numberHapus	public	static Bitmap
editTextBlank	public	static Bitmap
editTextHit	public	static Bitmap
dialogAbah	public	static Bitmap
comic	public	Typeface
cour	public	Typeface
ceritaKabayan	public	Music
introSongs	public	Music
musicPelatihan1	public	Music
musicPelatihan2	public	Music
musicPelatihan3	public	Music
musicPembukaPelatihan	public	Music
soundButton	public	Sound
soundCangkul	public	Sound
soundJawabanBenar	public	Sound
soundJawabanSalah	public	Sound

30. Kelas StateGame

Nama Kelas : StateGame

Tabel 3.46 Kelas StateGame

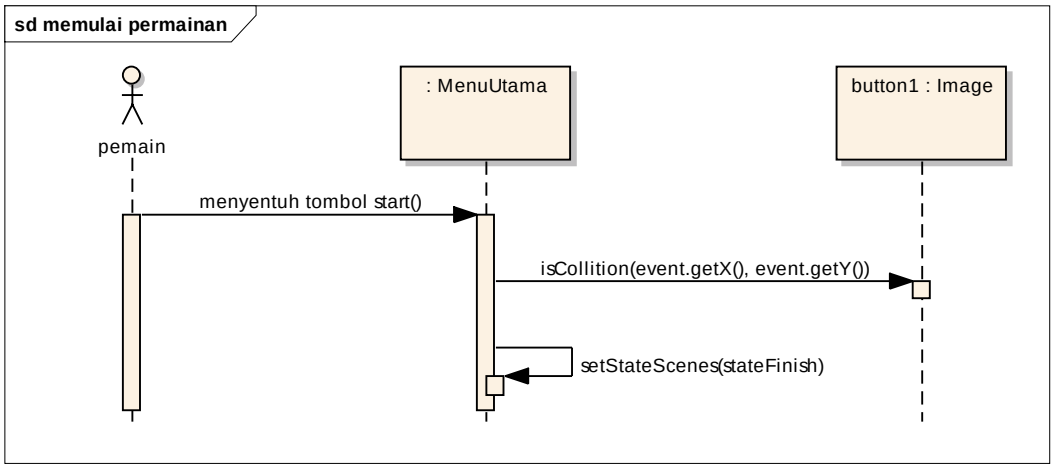
Nama Operasi	Visibility	Keterangan
-	-	-
Nama Atribut	**Visibility**	**Tipe**
STATE_MENU_KELUAR	public	static final int
STATE_MENU_PAUSE	public	static final int
STATE_MENU_UTAMA	public	static final int
STATE_LEVEL_1	public	static final int
STATE_LEVEL_2	public	static final int
STATE_LEVEL_3	public	static final int
STATE_GAME_END	public	static final int
STATE_GAME_OVER	public	static final int
STATE_INTRO_ADZIE	public	static final int
STATE_INTRO_KABAYAN_BERHITUNG	public	static final int
STATE_INTRO_GAME_1	public	static final int
STATE_INTRO_GAME_2	public	static final int
STATE_INTRO_PEMBUKA_LEVEL_1	public	static final int
STATE_INTRO_PEMBUKA_LEVEL_2	public	static final int
STATE_INTRO_PEMBUKA_LEVEL_3	public	static final int
STATE_INTRO_LEVEL_1	public	static final int
STATE_INTRO_LEVEL_2	public	static final int
STATE_INTRO_LEVEL_3	public	static final int

3.1.7.4 Sequence Diagram

Diagram sekuen atau *Sequence Diagram* menggambarkan kelakuan objek atau *class* pada *use case* dengan mendeskripsikan waktu hidup dan *message* yang dikirimkan dan diterima antara *object* atau *class*.

1. *Sequence Diagram* Memulai Permainan

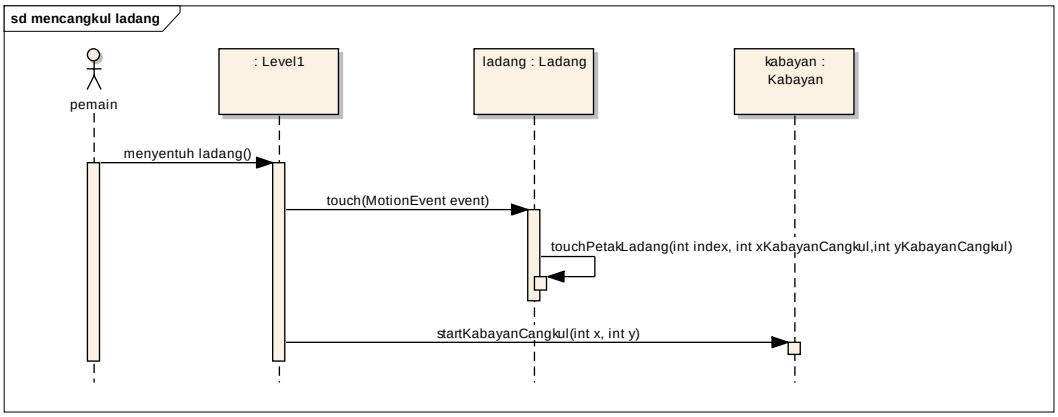
Berikut adalah *sequence diagram* memulai permainan.



Gambar 3.21 *Sequence Diagram* Memulai Permainan

2. *Sequence Diagram* Mencangkul Ladang

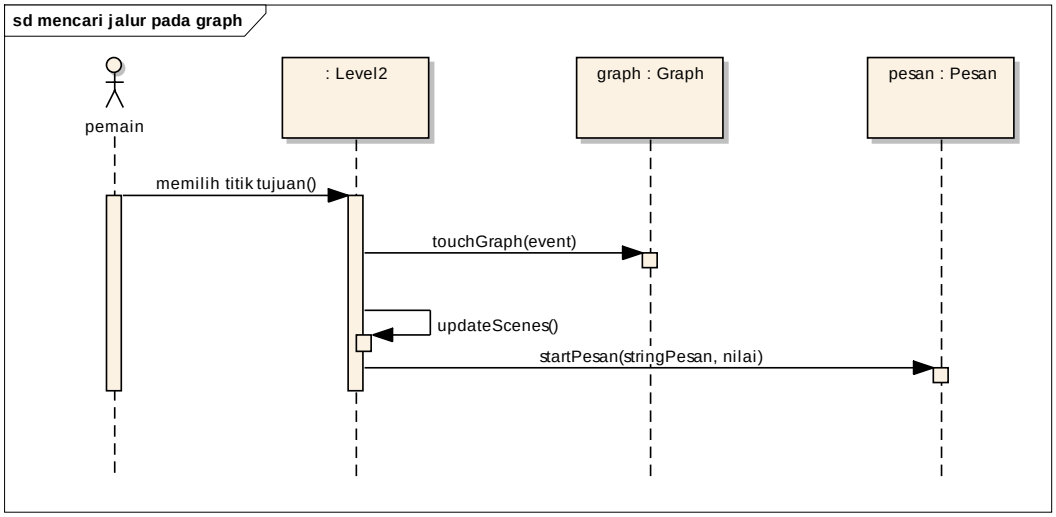
Berikut adalah *sequence diagram* mencangkul ladang.



Gambar 3.22 *Sequence Diagram* Mencangkul Ladang

3. *Sequence Diagram* Mencari Jalur Pada Graph

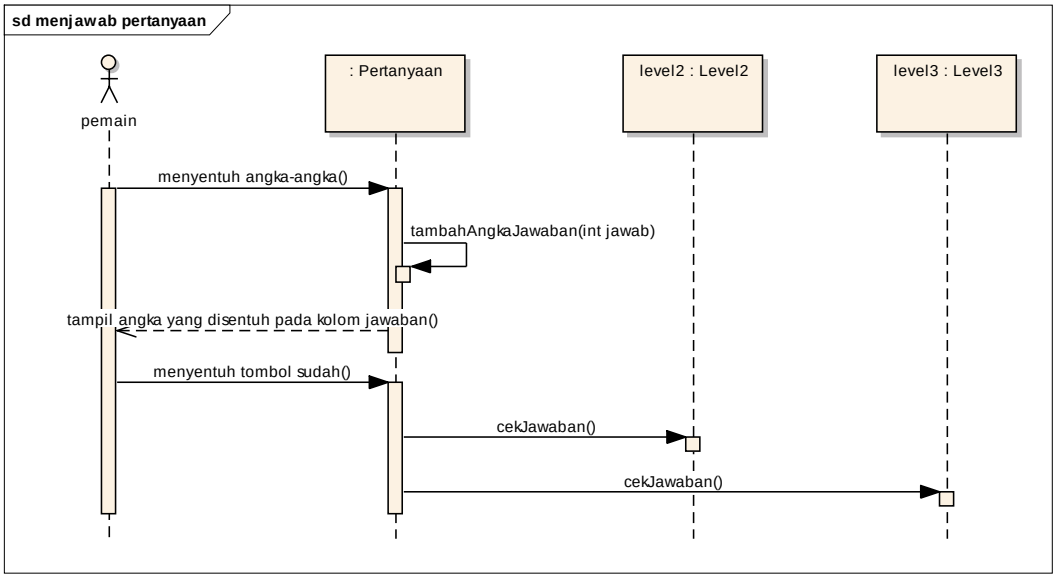
Berikut adalah *sequence diagram* mencari jalur pada graph.



Gambar 3.23 *Sequence Diagram* Mencari Jalur Pada Graph

4. *Sequence Diagram* Menjawab Pertanyaan

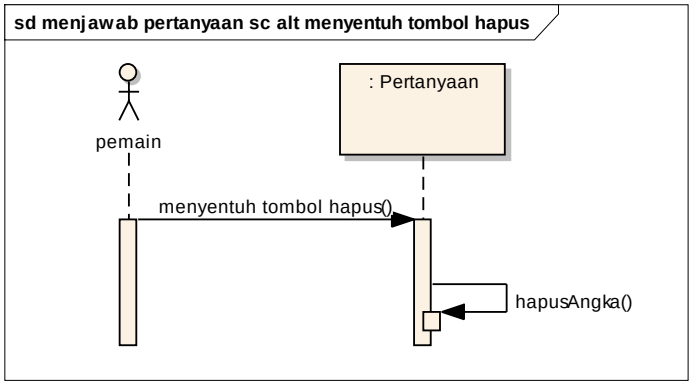
Berikut adalah *sequence diagram* menjawab pertanyaan.



Gambar 3.24 *Sequence Diagram* Menjawab Pertanyaan

1. *Scenario* Alternatif Menyentuh Tombol Hapus

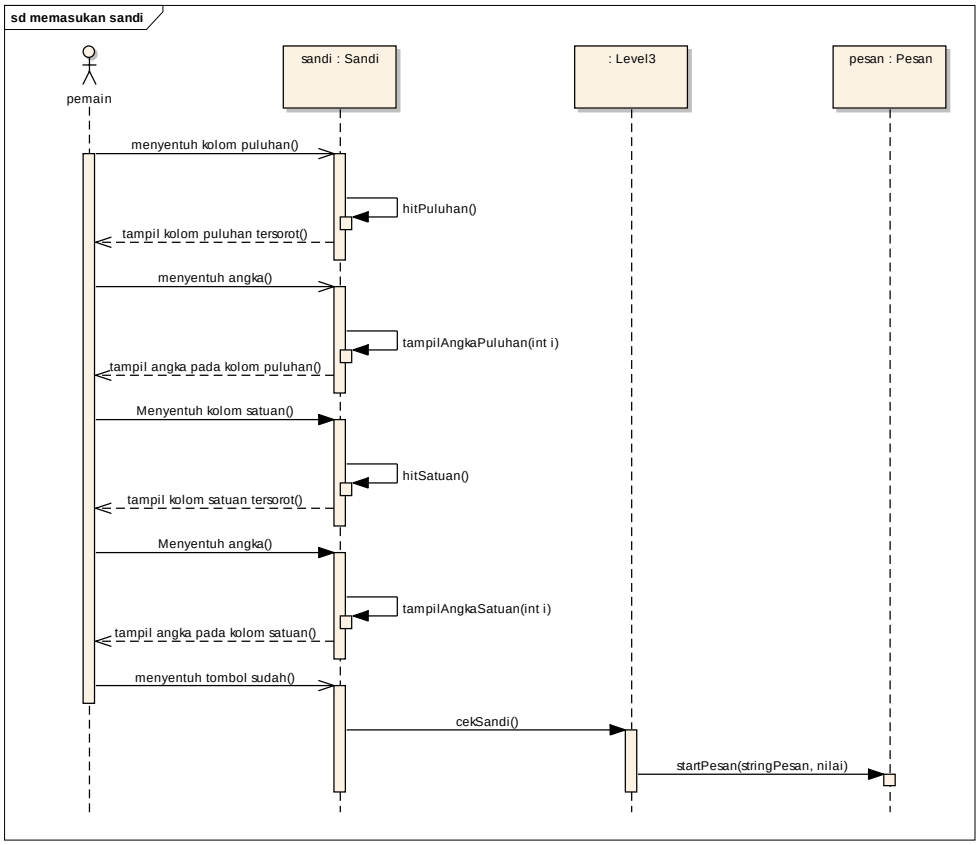
Berikut adalah *sequence diagram* dari *scenario* alternatif dari menjawab pertanyaan yaitu menyentuh tombol hapus .



Gambar 3.25 *Scenario* Alternatif Menyentuh Tombol Hapus dari Menjawab Pertanyaan

5. *Sequence Diagram* Memasukan Sandi

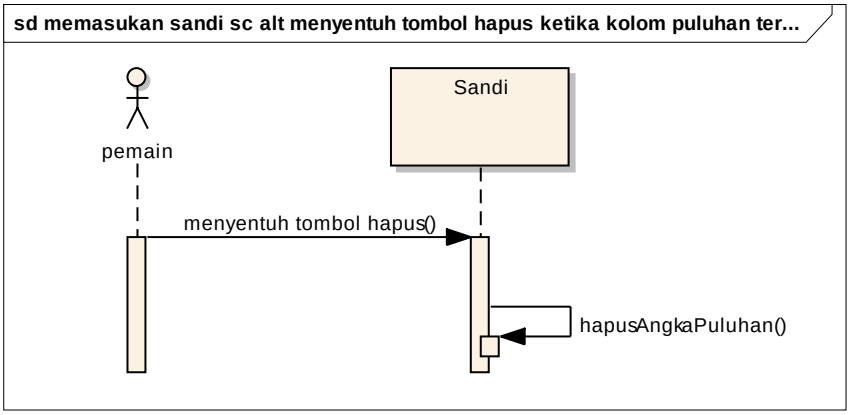
Berikut adalah *sequence diagram* memasukan sandi.



Gambar 3.26 *Sequence Diagram* Memasukan Sandi

1. **S*cenario* Alternatif Menyentuh Tombol Hapus Ketika Kolom Puluhan Tersorot**

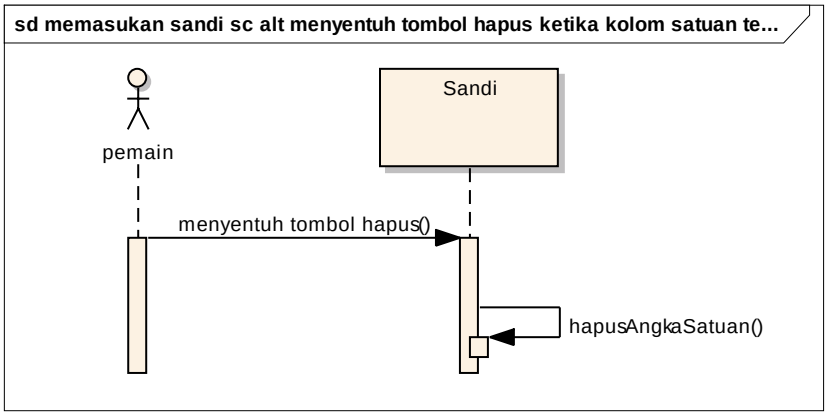
Berikut adalah *sequence diagram* dari *scenario* alternatif dari memasukan sandi yaitu menyentuh tombol hapus ketika kolom puluhan tersorot.



Gambar 3.27 *Scenario* Alternatif Menyentuh Tombol Hapus Ketika Kolom Puluhan Tersorot dari Memasukan Sandi

2. ***Scenario* Alternatif Menyentuh Tombol Hapus Ketika Kolom SatuanTersorot**

Berikut adalah *sequence diagram* dari *scenario* alternatif dari memasukan sandi yaitu menyentuh tombol hapus ketika kolom satuan tersorot.



Gambar 3.28 *Scenario* Alternatif Menyentuh Tombol Hapus Ketika Kolom Satuan Tersorot dari Memasukan Sandi

3.2 Perancangan Sistem

Pada tahap perancangan sistem merupakan tahap untuk merancang sistem yang akan dibangun.

3.2.1 Perancangan Komponen

Perancangan komponen merupakan tahap untuk merencang komponen-komponen yang akan ada pada pembangunan game. Didalam bagian ini akan dijelaskan tentang karakter-karakter yang digunakan dan storyboard dari Game Kabayan Berhitung.

3.2.1.1 Karakter

Di dalam bagian ini akan dijelaskan karakter-karakter yang ada pada *game* ini, sebagai berikut :

Tabel 3.47 Karakter Game Kabayan Berhitung

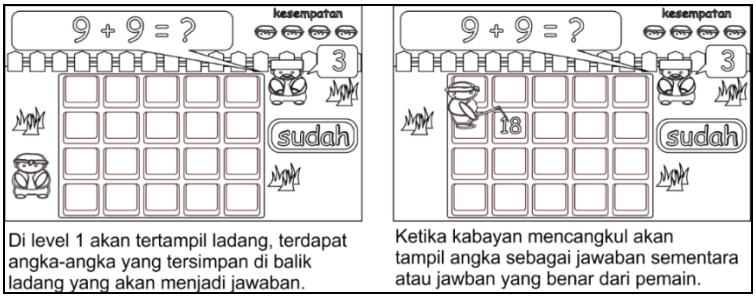
No	Nama	Gambar	Keterangan
1	Kabayan		Karakter utama yang dimainkan oleh pemain
2	Nyi Iteung		Karakter pembantu ditemui di awal dan akhir game
3	Abah		Karakter pembantu ditemui dari awal sampai akhir game

3.2.1.2 Storyboard

Storyboard adalah sketsa gambar yang disusun berurutan sesuai dengan naskah, dengan *storyboard* ide cerita dapat disampaikan kepada orang lain dengan lebih mudah, karena dapat menggiring khayalan seseorang mengikuti gambar-gambar yang tersaji sehingga menghasilkan persepsi yang sama.

a. *Storyboard Level* 1(Ladang)

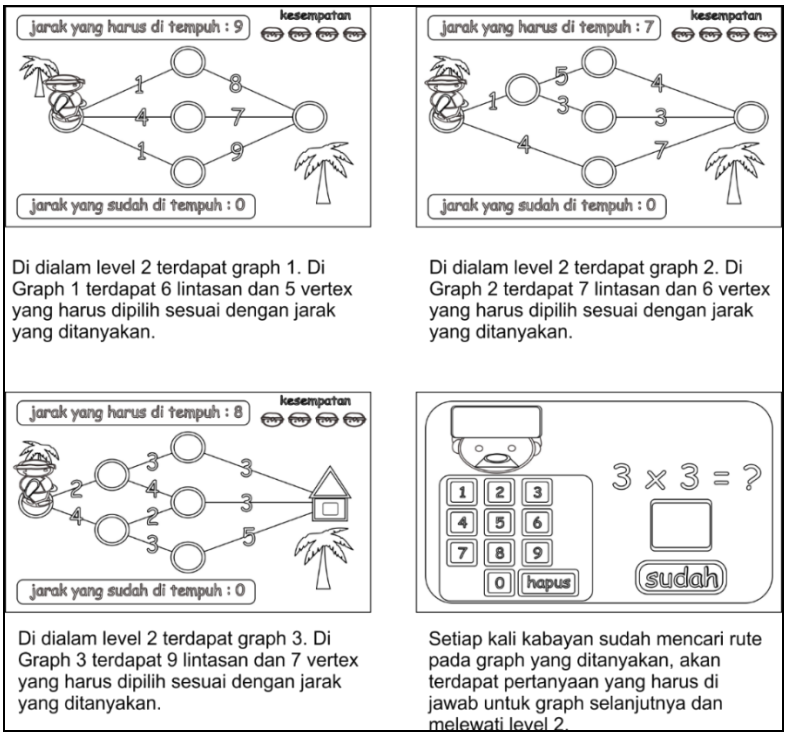
Berikut adalah *storyboard level* 1.



Gambar 3.29 Storyboard Level 1

b. *Storyboard Level* 2(Jalan menuju rumah)

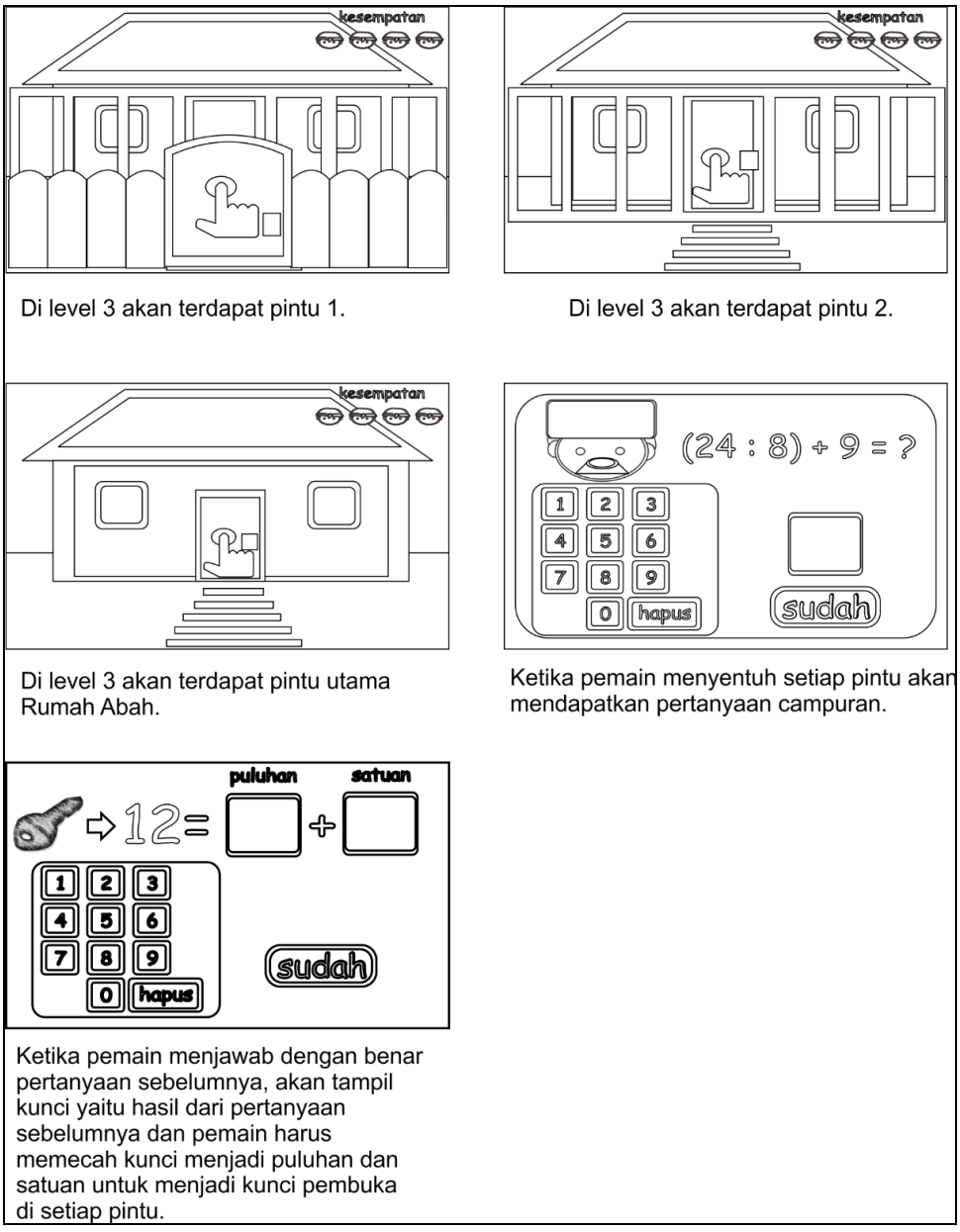
Berikut adalah *storyboard level* 2.



Gambar 3.30 Storyboard Level 2

c. *Storyboard Level* 3(Rumah Abah)

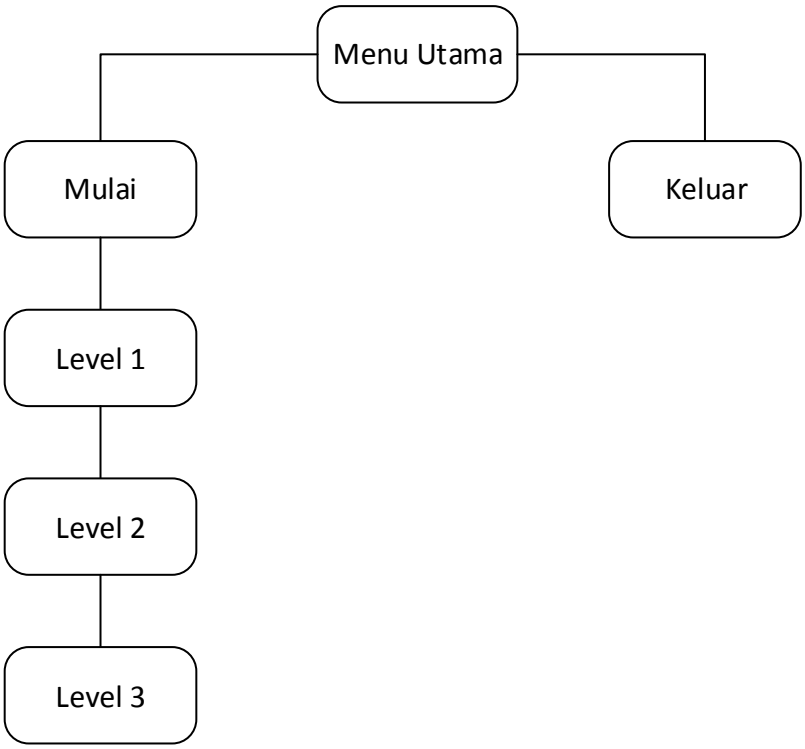
Berikut adalah *stroyboard level* 3.



Gambar 3.31 Storyboard Level 3

3.2.2 Perancangan Struktur Menu

Perancangan struktur menu pada *game* Kabayan Berhitung sebagai berikut.



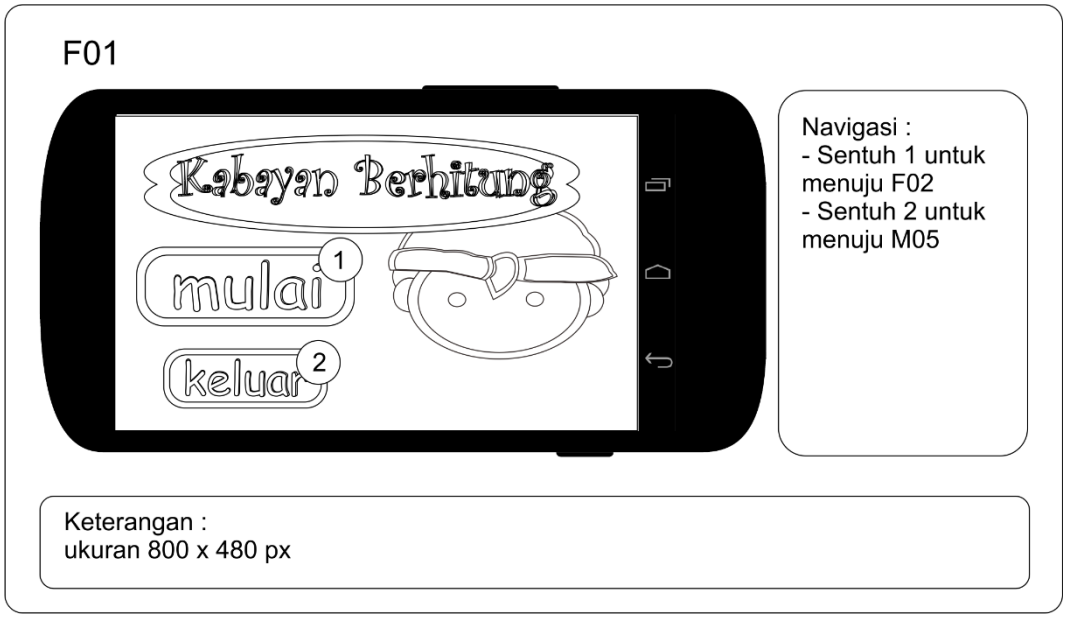
Gambar 3.32 Struktur Menu Game Kabayan Berhitung

3.2.3 Perancangan Antarmuka

Perancangan antarmuka bertujuan untuk memberikan gambaran tentang aplikasi yang akan dibangun, sehingga akan mempermudah dalam mengimplementasikan aplikasi serta akan memudahkan pembuatan aplikasi.

1. Perancangan Antarmuka Menu Utama

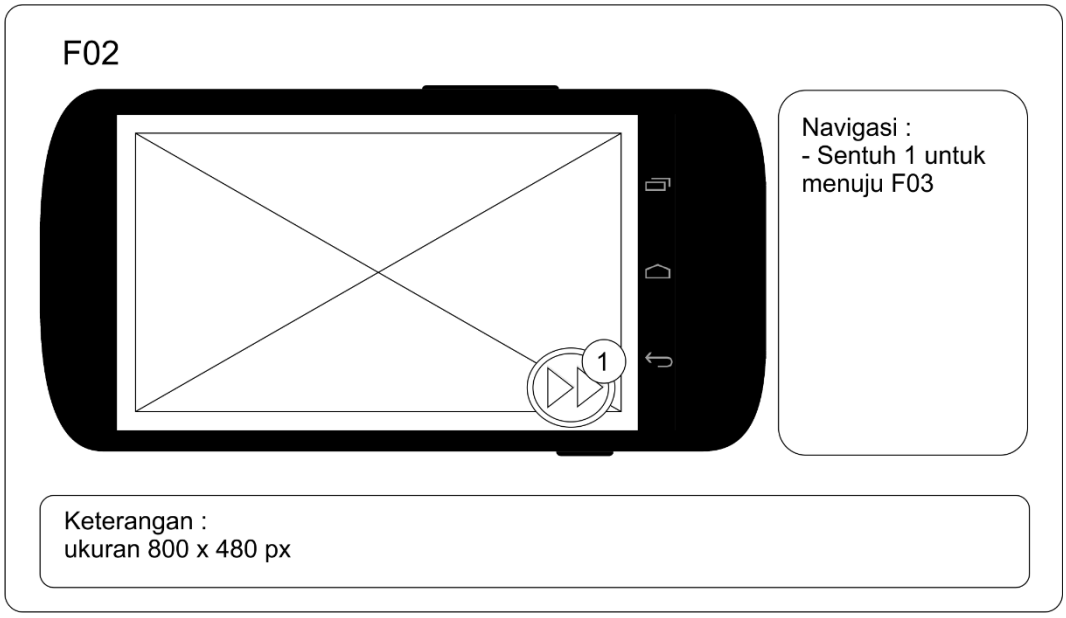
Berikut adalah perancangan antar muka menu utama.



Gambar 3.33 Perancangan Antarmuka Menu Utama

2. Perancangan Antarmuka Intro Game

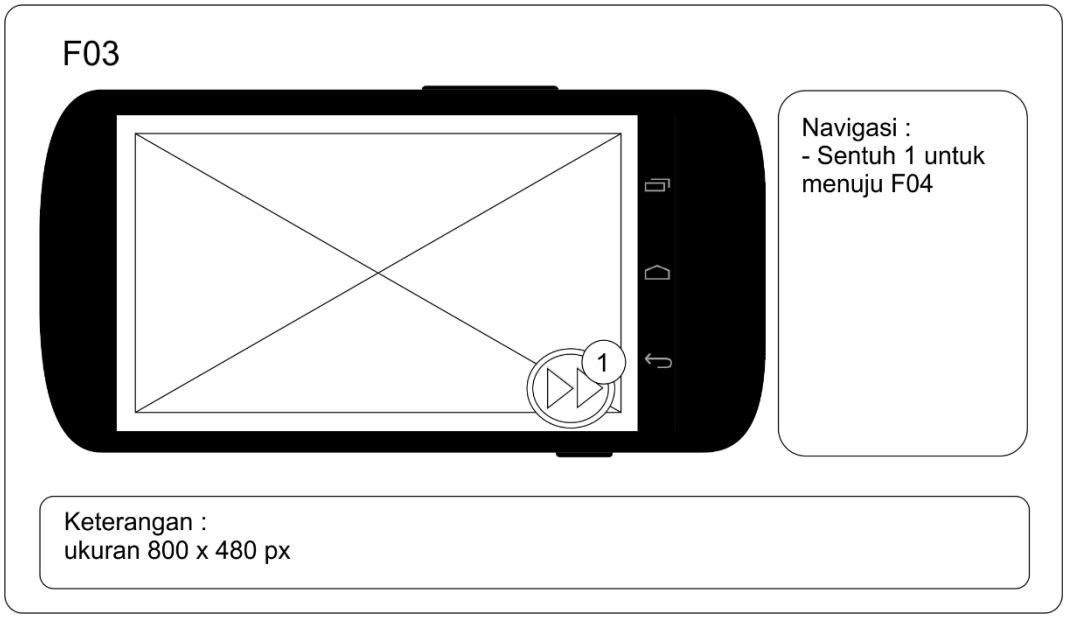
Berikut adalah perancangan antarmuka intro *game*.



Gambar 3.34 Perancangan Antarmuka Intro Game

3. Perancangan Antarmuka Petunjuk Level 1

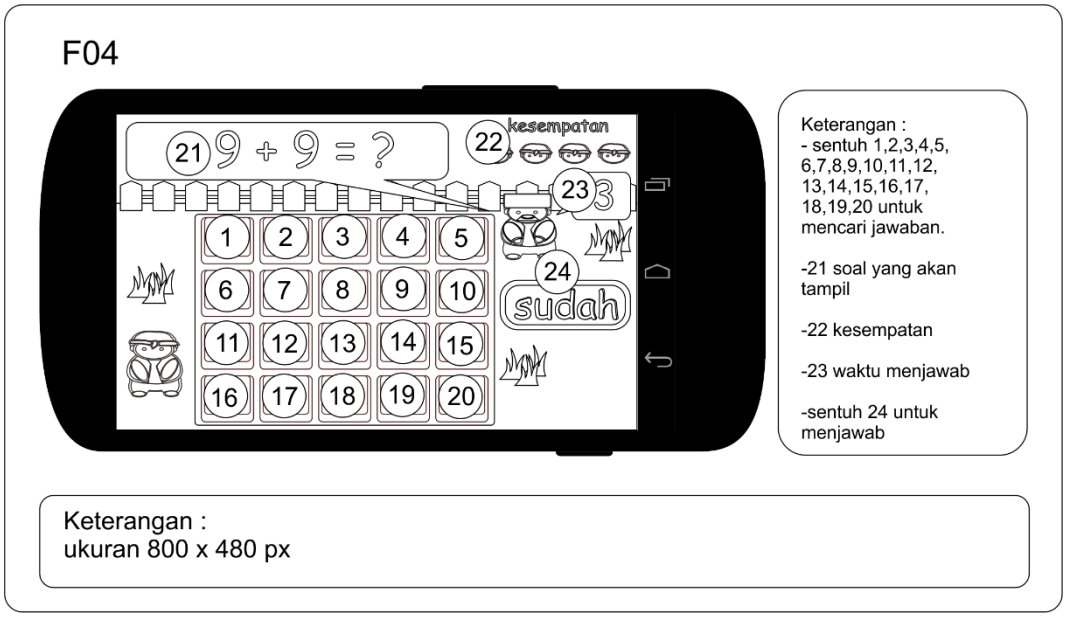
Berikut adalah perancangan antarmuka petunjuk *level* 1.



Gambar 3.35 Perancangan Antarmuka Petunjuk Level 1

4. Perancangan Antarmuka Level 1

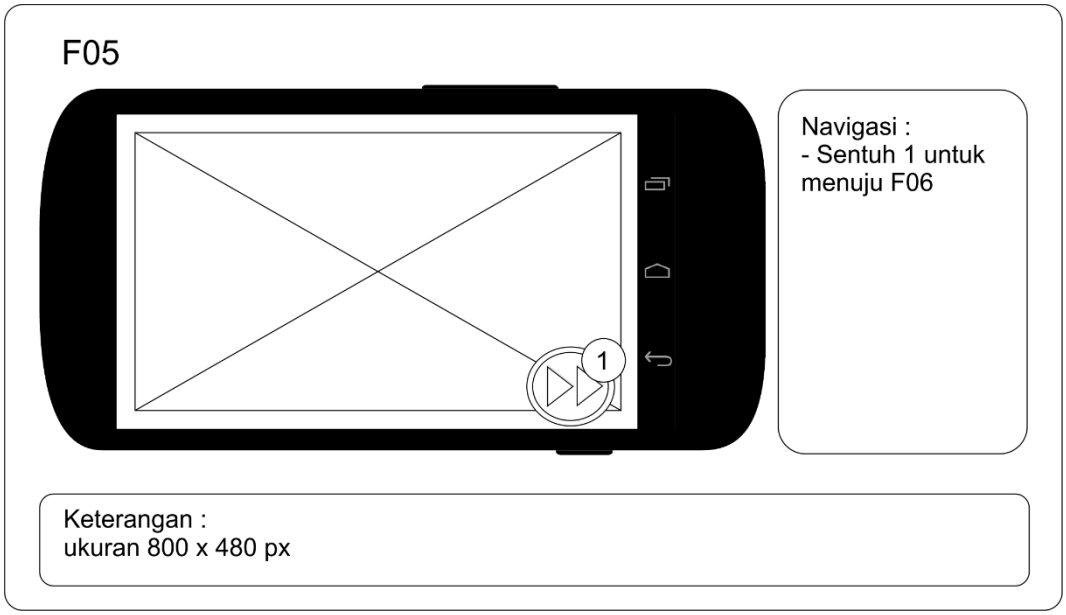
Berikut adalah perancangan antarmuka *level* 1.



Gambar 3.36 Perancangan Antarmuka Level 1

5. Perancangan Antarmuka Petunjuk Level 2

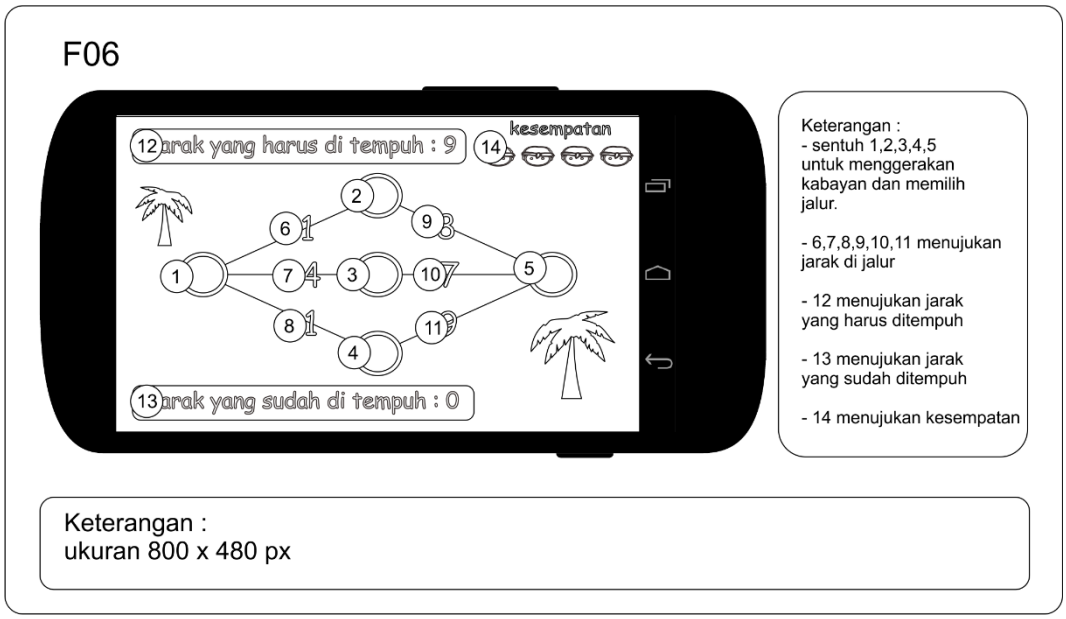
Berikut adalah perancangan antarmuka petunjuk *level* 2.



Gambar 3.37 Perancangan Antarmuka Petunjuk Level 2

6. Perancangan Antarmuka Level 2 Graph 1

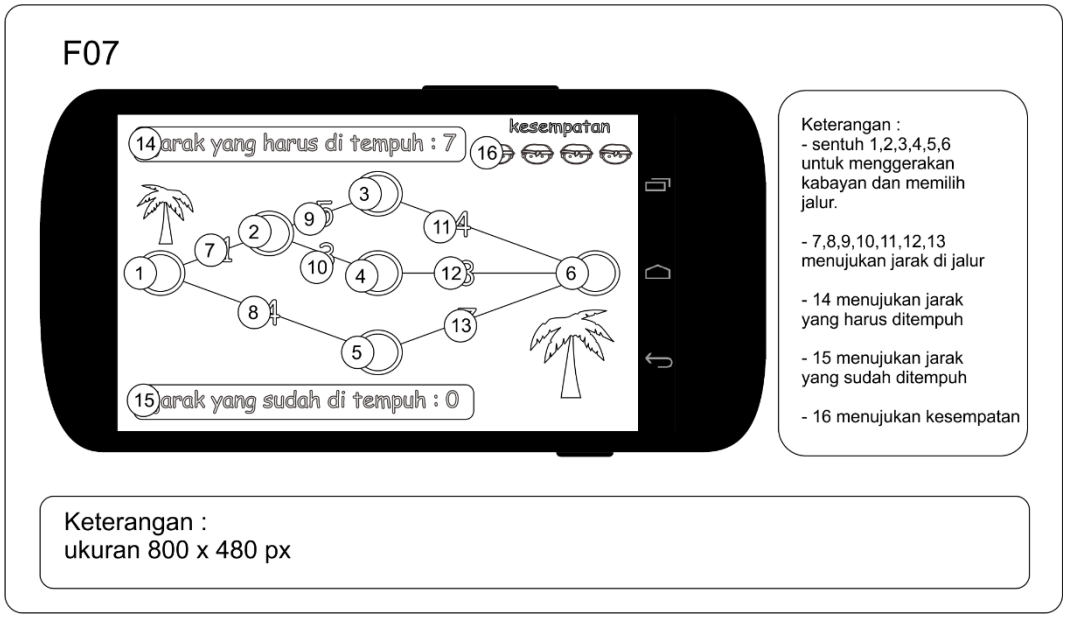
Berikut adalah perancangan antarmuka *level* 2 *graph* 1.



Gambar 3.38 Perancangan Antarmuka Level 2 Graph 1

7. Perancangan Antarmuka Level 2 Graph 2

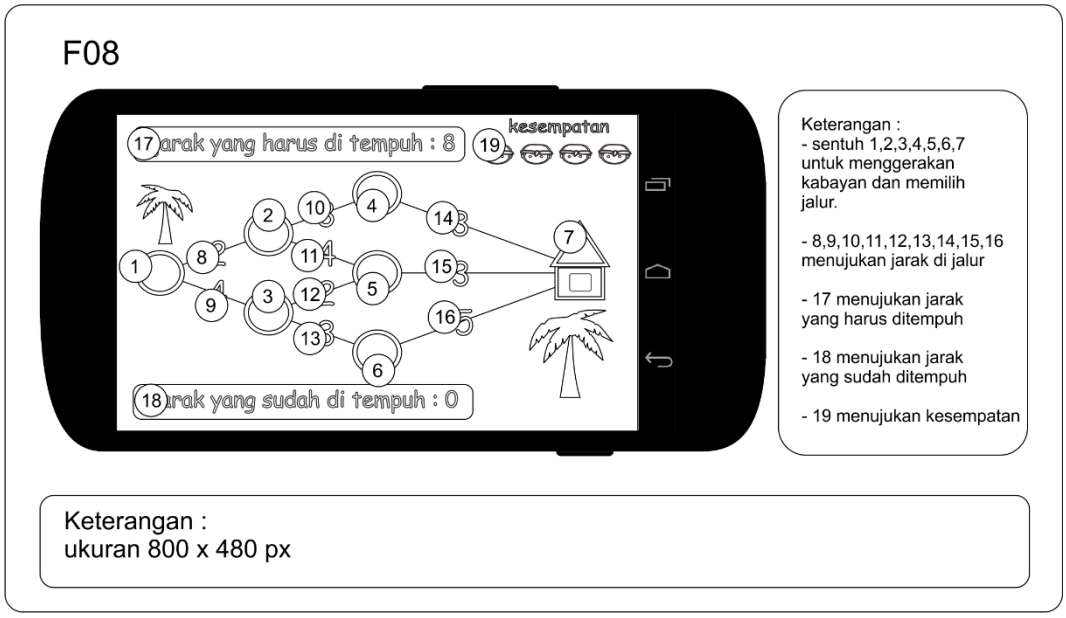
Berikut adalah perancangan antarmuka *level* 2 *graph* 2.



Gambar 3.39 Perancangan Antarmuka Level 2 Graph 2

8. Perancangan Antarmuka Level 2 Graph 3

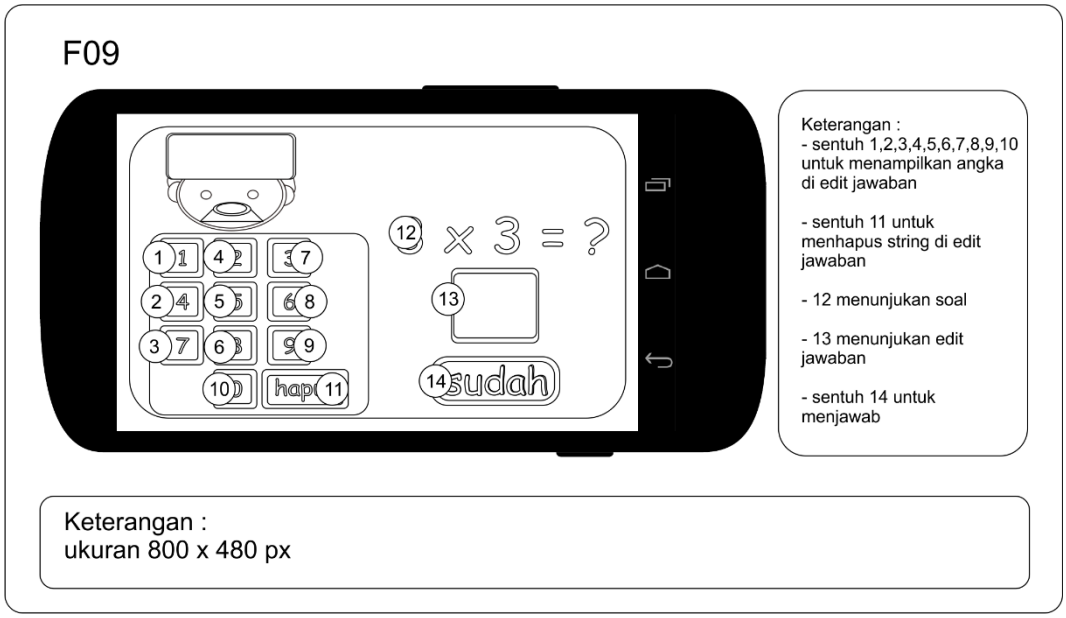
Berikut adalah perancangan antarmuka *level* 2 *graph* 3.



Gambar 3.40 Perancangan Antarmuka Level 2 Graph 3

9. Perancangan Antarmuka Level 2 Pertanyaan

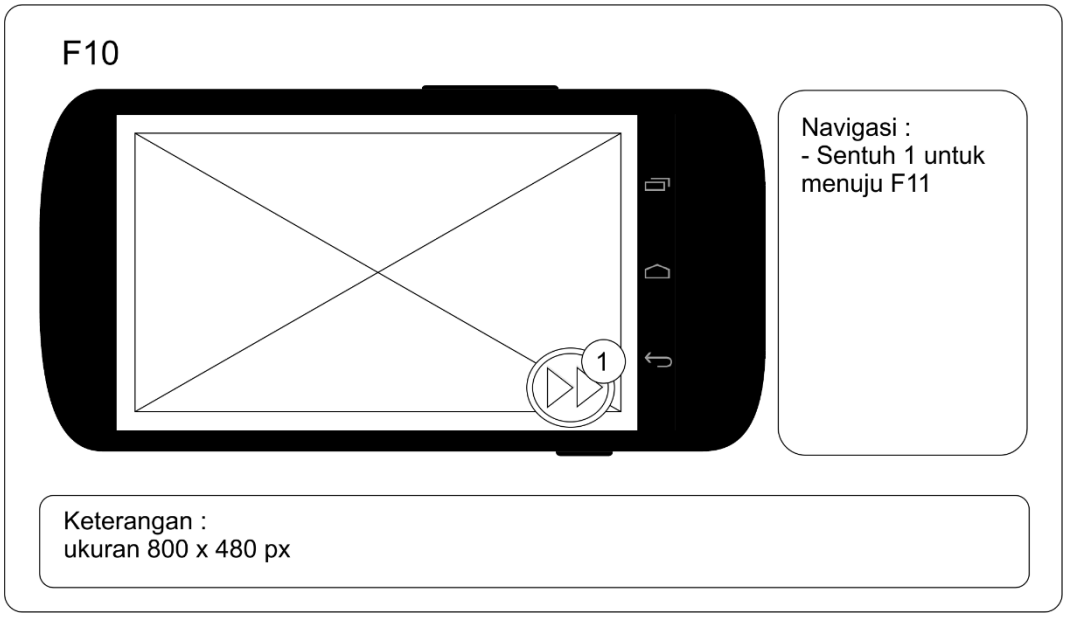
Berikut adalah perancangan antarmuka *level* 2 pertanyaan.



Gambar 3.41 Perancangan Antarmuka Level 2 Pertanyaan

10. Perancangan Antarmuka Petunjuk Level 3

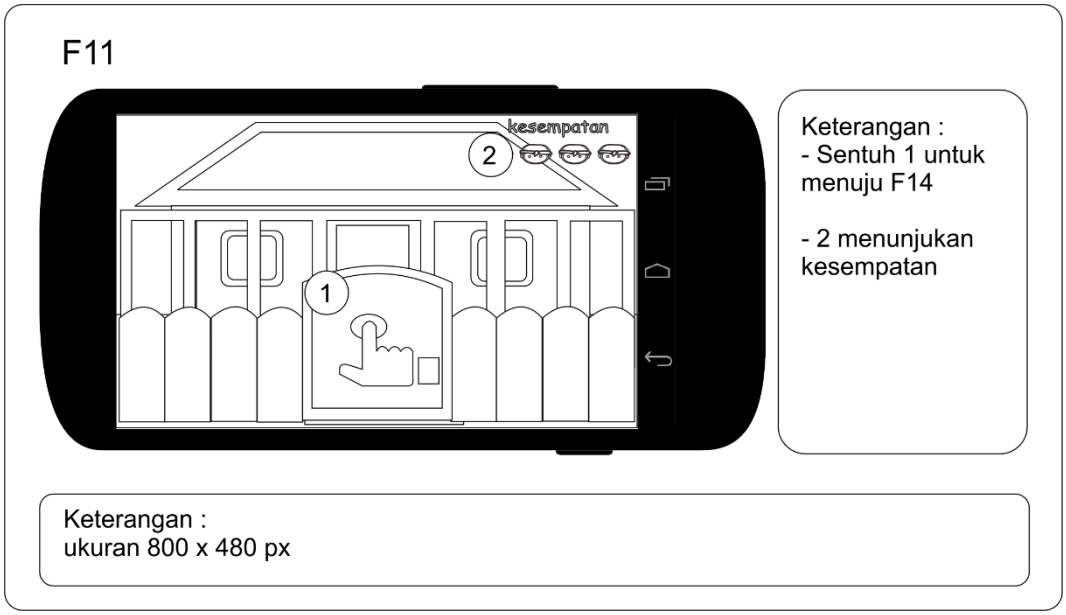
Berikut adalah perancangan antarmuka petunjuk *level* 3.



Gambar 3.42 Perancangan Antarmuka Petunjuk Level 3

11. Perancangan Antarmuka Level 3 Pintu 1

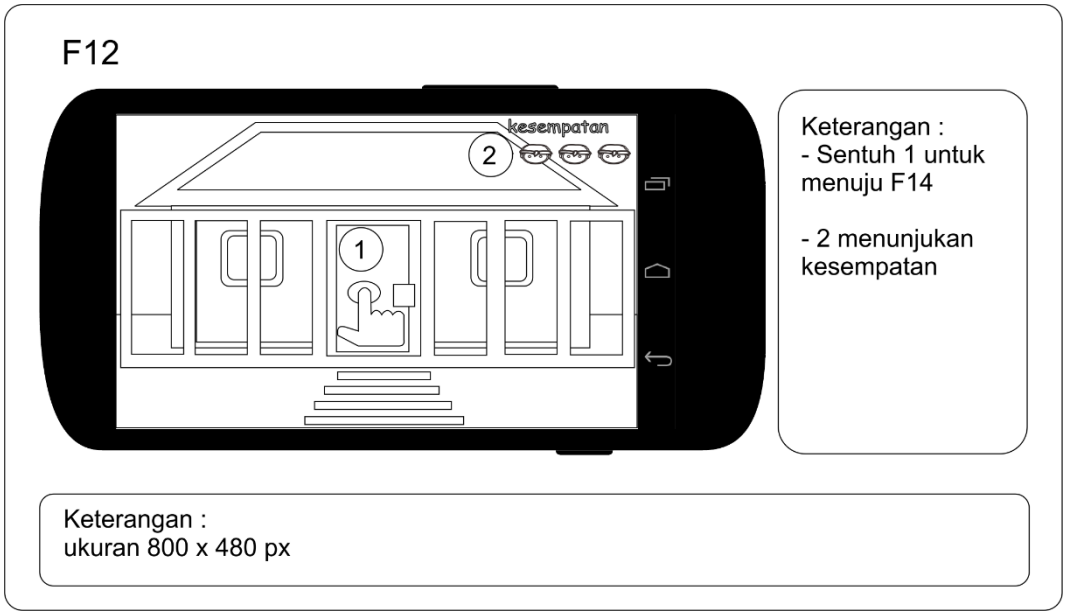
Berikut adalah perancangan antarmuka *level* 3 pintu 1.



Gambar 3.43 Perancangan Antarmuka Level 3 Pintu 1

12. Perancangan Antarmuka Level 3 Pintu 2

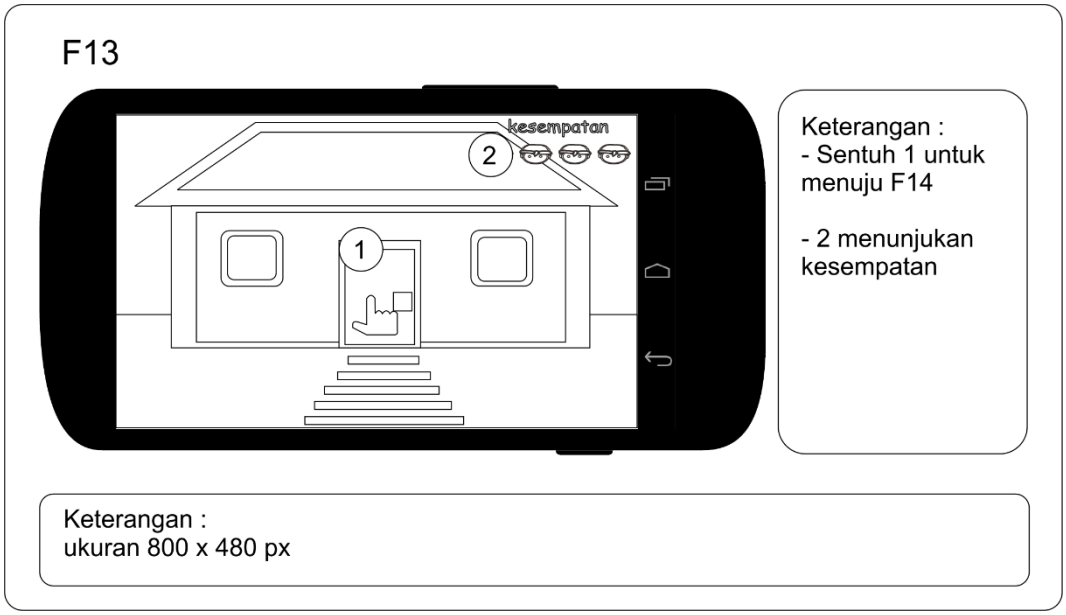
Berikut adalah perancangan antarmuka *level* 3 pintu 2.



Gambar 3.44 Perancangan Antarmuka Level 3 Pintu 2

13. Perancangan Antarmuka Level 3 Pintu 3

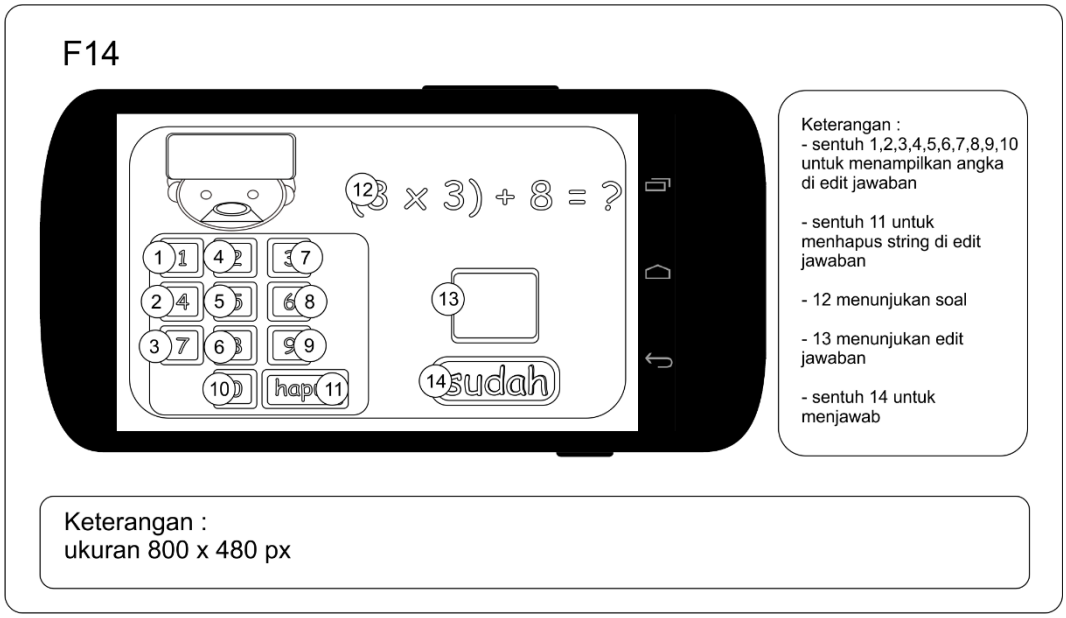
Berikut adalah perancangan antarmuka *level* 3 pintu 3.



Gambar 3.45 Perancangan Antarmuka Level 3 Pintu 3

14. Perancangan Antarmuka Level 3 Pertanyaan

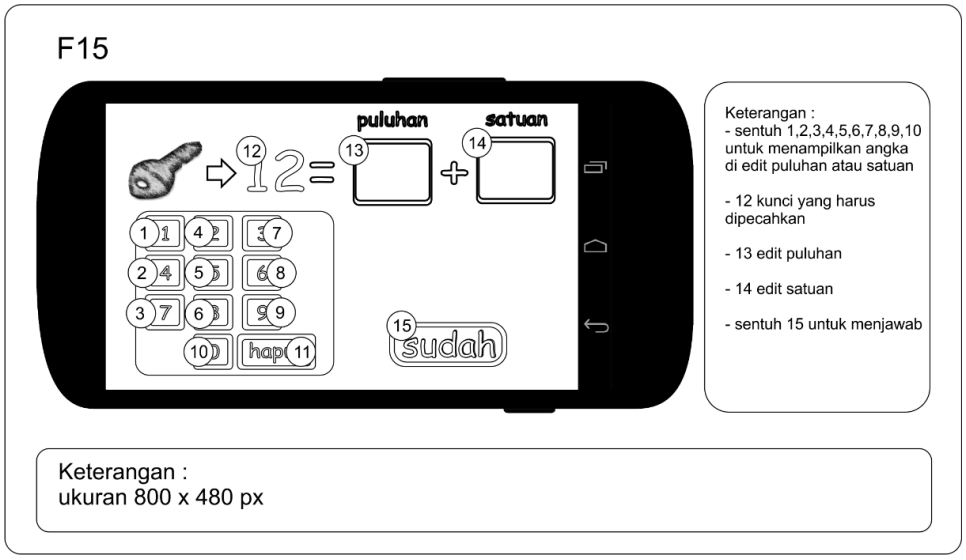
Berikut adalah perancangan antarmuka *level* 3 pertanyaan.



Gambar 3.46 Perancangan Antarmuka Level 3 Pertanyaan

15. Perancangan Antarmuka Level 3 Memasukan Sandi

Berikut adalah perancangan antarmuka *level* 3 memasukan sandi.



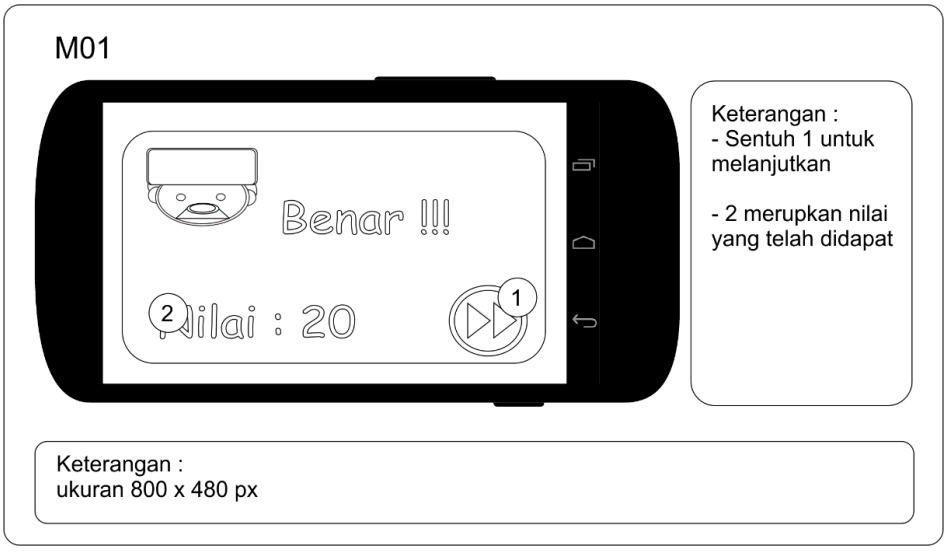
Gambar 3.47 Perancangan Antarmuka Level 3 Memasukan Sandi

3.2.4 Perancangan Pesan

Pada bagian ini akan dijelaskan perancangan pesan yang terdapat pada game edukasi *game* Kabayan Berhitung.

1. Perancangan Pesan Jawaban Benar

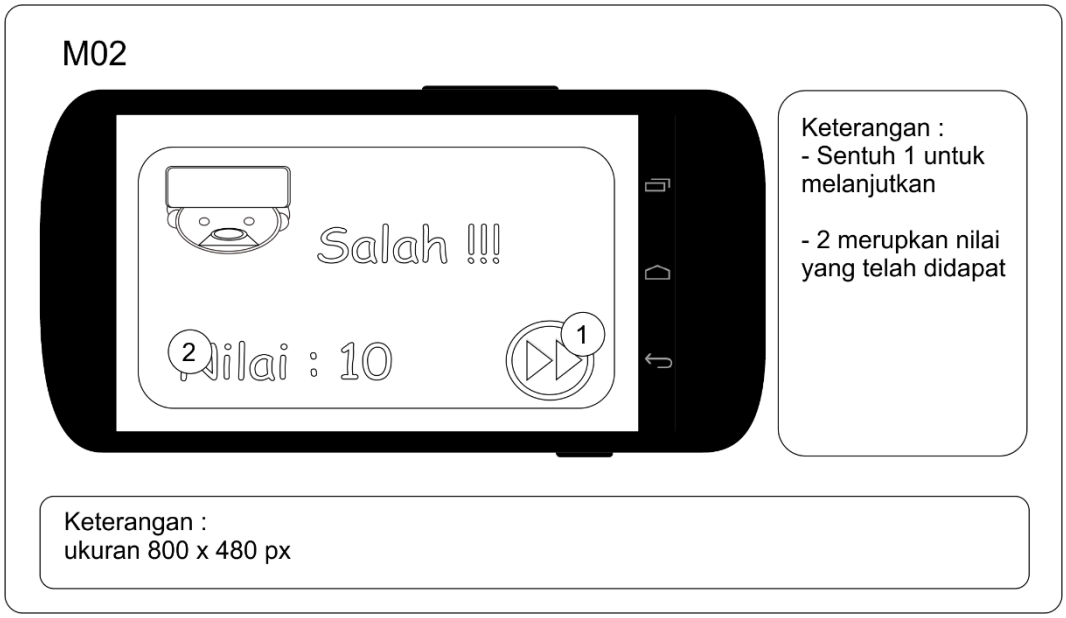
Berikut adalah perancangan pesan jawaban benar.



Gambar 3.48 Perancangan Pesan Jawaban Benar

2. Perancangan Pesan Jawaban Salah

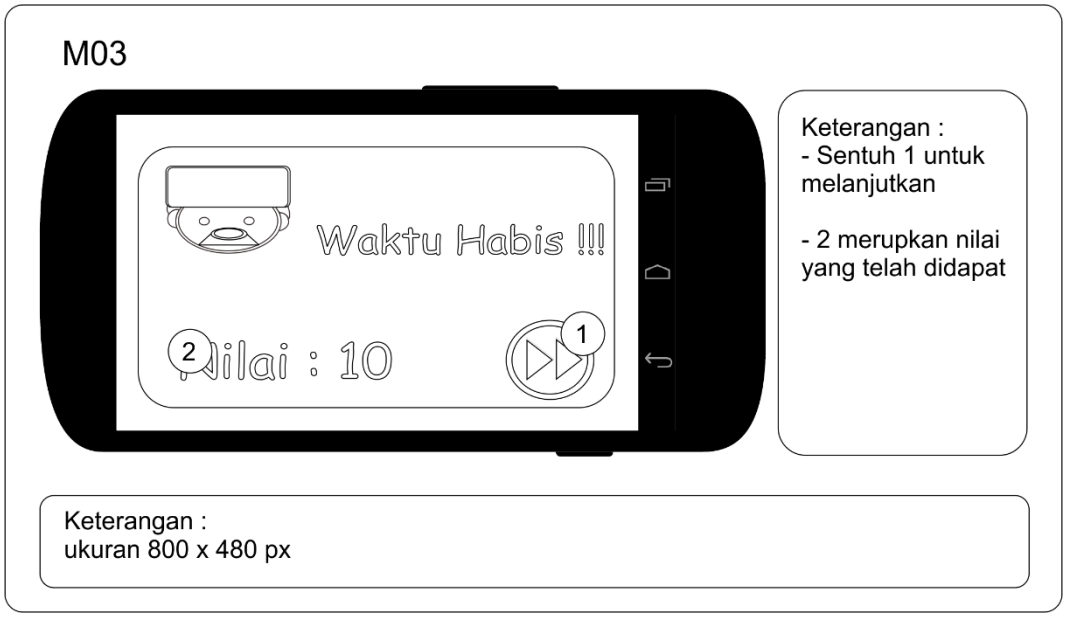
Berikut adalah perancangan pesan jawaban salah.



Gambar 3.49 Perancangan Pesan Jawaban Salah

3. Perancangan Pesan Waktu Habis

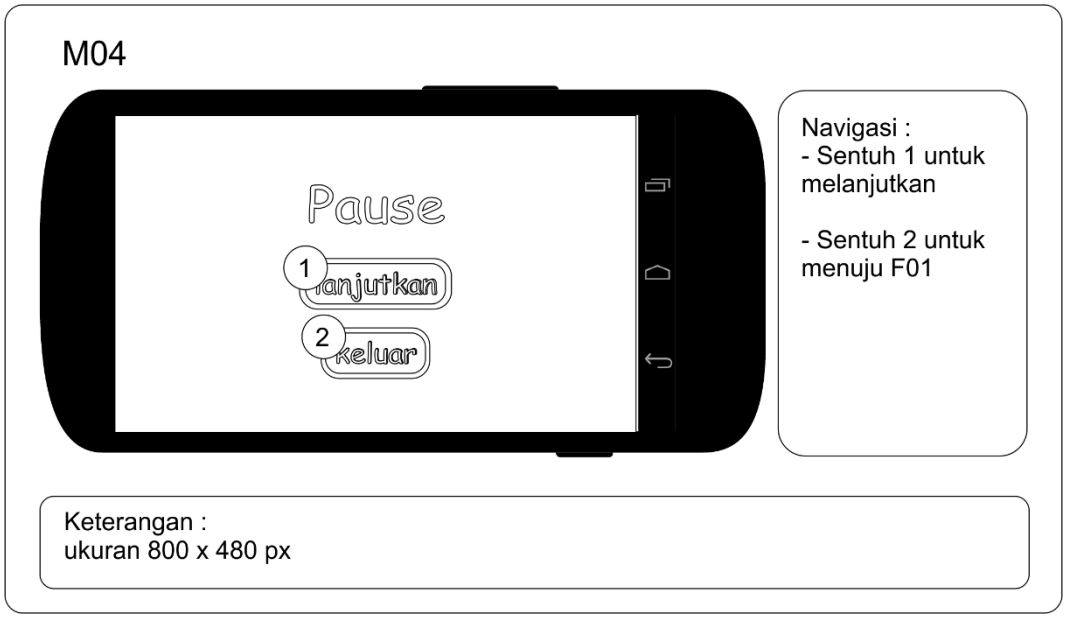
Berikut adalah perancangan pesan waktu habis.



Gambar 3.50 Perancangan Pesan Waktu Habis

4. Perancangan Pesan Menu Pause

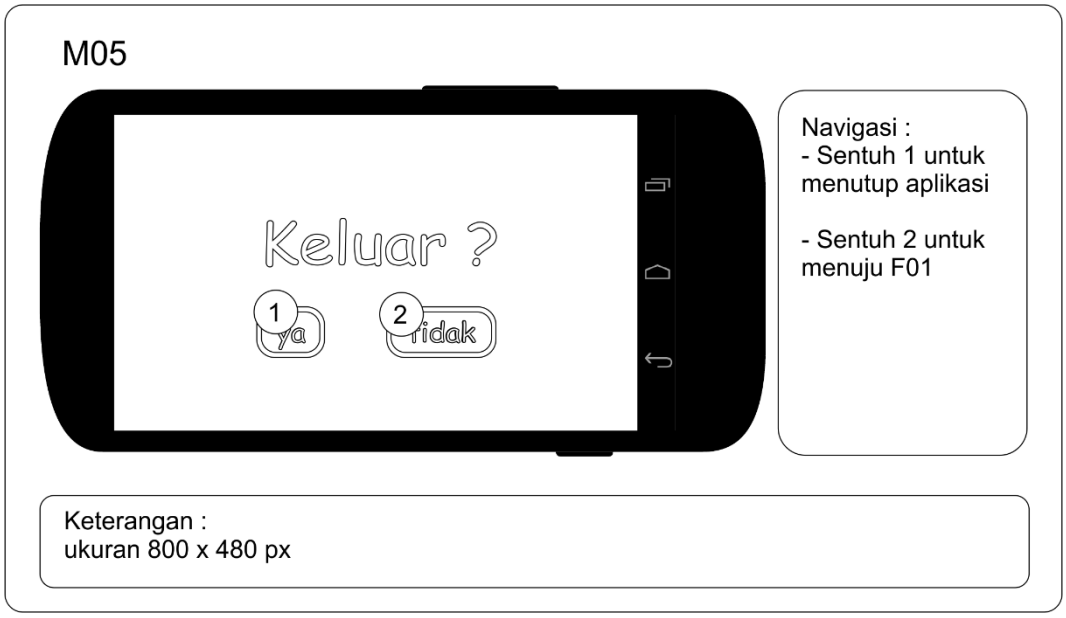
Berikut adalah perancangan pesan menu pause.



Gambar 3.51 Perancangan Pesan Menu Pause

5. Perancangan Pesan Menu Keluar

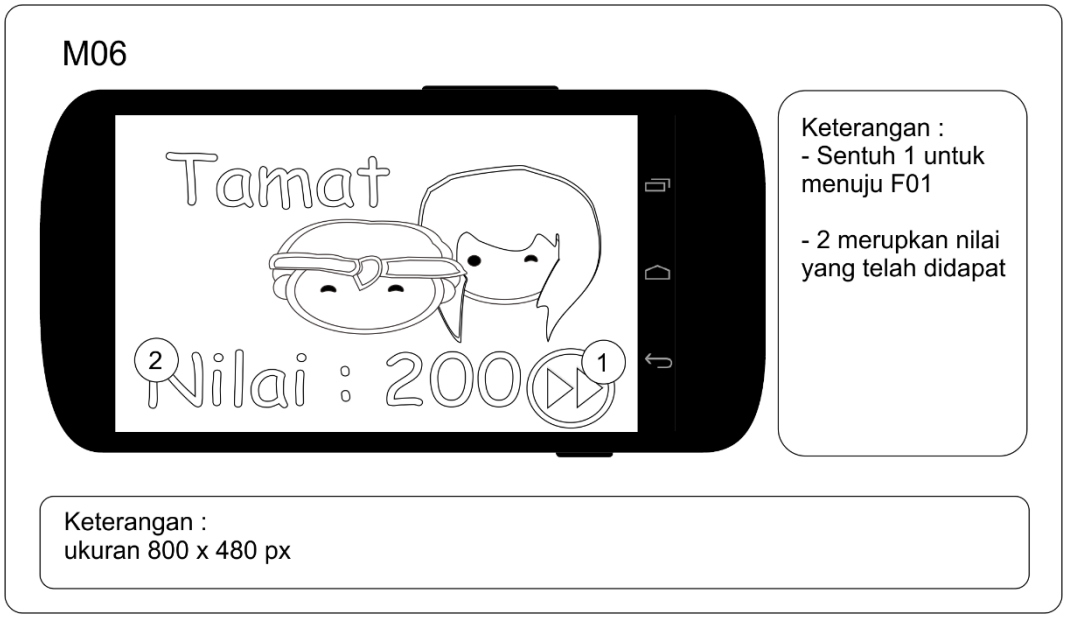
Berikut adalah perancangan pesan menu keluar.



Gambar 3.52 Perancangan Pesan Menu Keluar

6. Perancangan Pesan Tamat

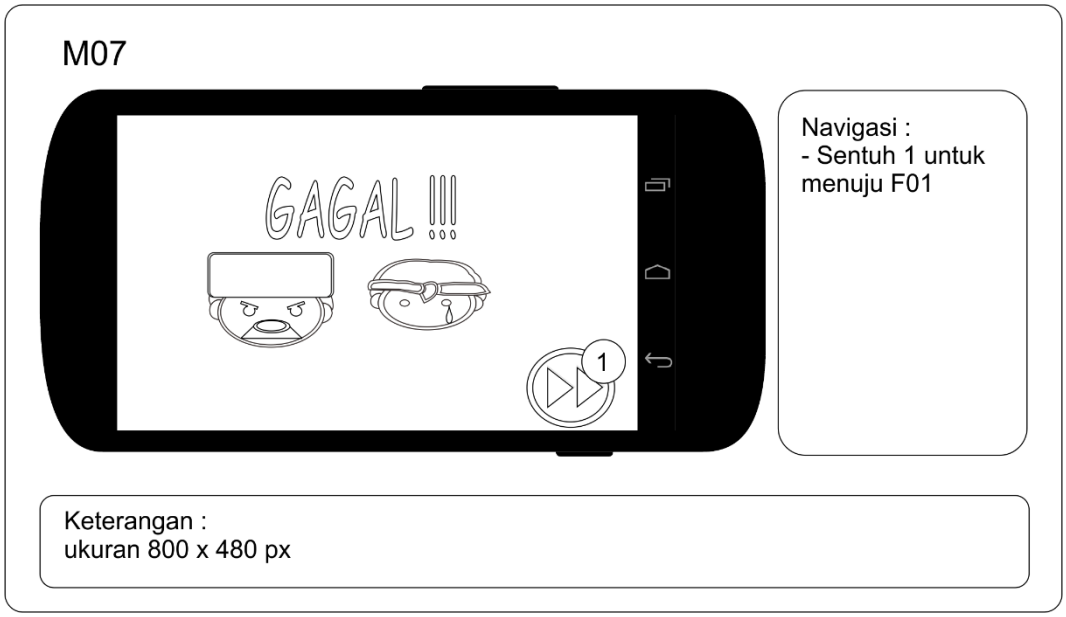
Berikut adalah perancangan pesan tamat.



Gambar 3.53 Perancangan Pesan Tamat

7. Perancangan Pesan Gagal

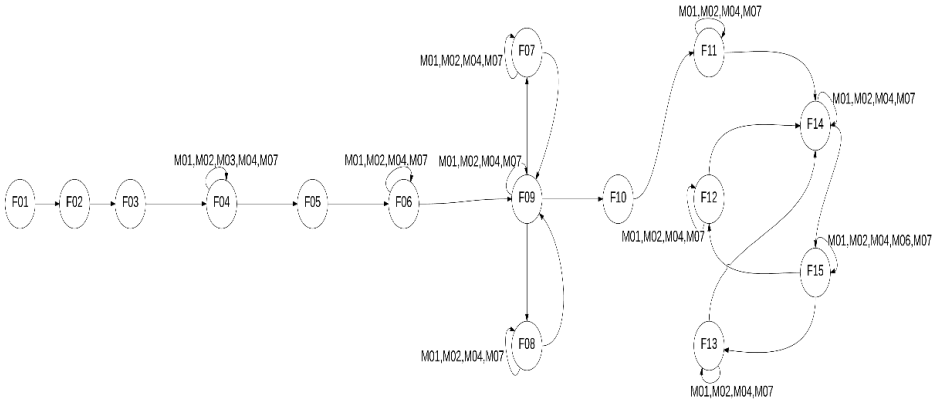
Berikut adalah perancangan pesan gagal.



Gambar 3.54 Perancangan Pesan Gagal

3.2.5 Jaringan Semantik

Untuk mengetahui keterhubungan dari setiap tampilan yang dibuat dapat menggunakan jaringan semantik.



Gambar 3.55 Jaringan Semantik Game Kabayan Berhitung

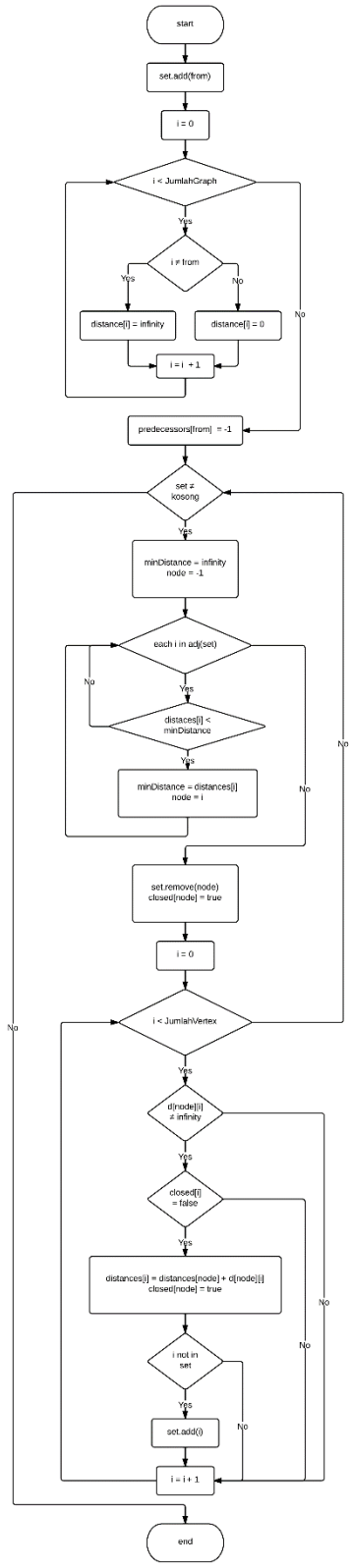
3.2.6 Perancangan Metode

Perancangan metode merupakan perancangan yang dilakukan untuk menetapkan detail algoritma yang akan dinyatakan ke dalam suatu program. Perancangan metode menjelaskan tentang bagaimana program aplikasi berjalan.

Pada *game* ini banyak sekali metode yang digunakan. Namun, pada perancangan ini akan menjelaskan beberapa metode yang penting, sebagai berikut.

1. Perancangan Metode Dijkstra

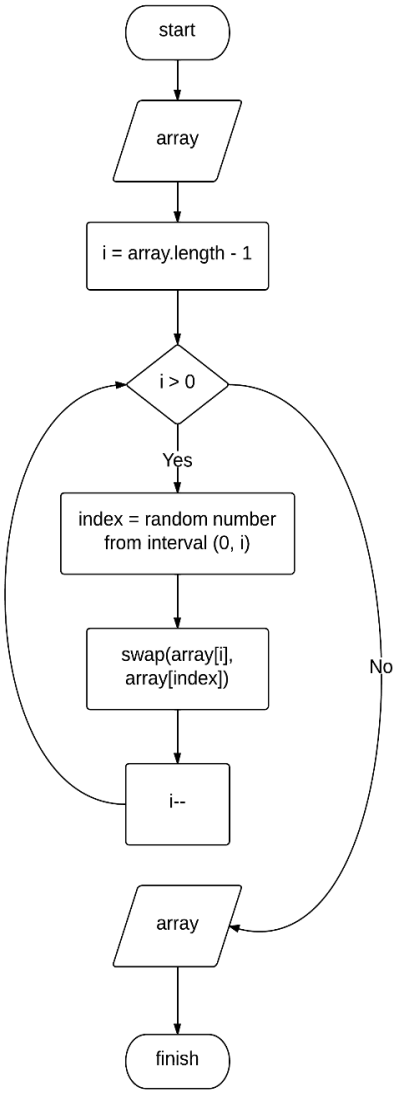
Metode ini digunakan untuk mencari rute terpendek sebagai solusi dan jarak terpendek sebagai perbandingan jarak yang harus ditempuh. Metode ini diterapkan di Class Graph. Berikut adalah flow chart dari dijkstra :



Gambar 3.56 Flow Chart Metode Dijkstra

2. Perangcangan Metode Fisher Yates

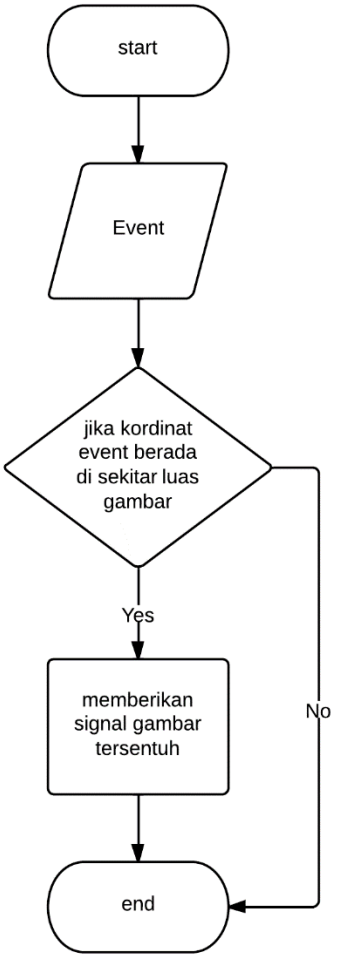
Metode ini digunakan untuk pengacakan jawaban pada level pertama game ini. Metode diterapkan di class Level1.



Gambar 3.57 Flow Chart Metode Fisher Yates

3. Perancangan Metode isCollition

Metode ini diunakan untuk pengecekan setiap gambar keteika di sentuh. Metode ini diterapkan pada *Class* Image.



Gambar 3.58 Flow Chart Metode isCollition