BAB III
ANALISIS DAN PERANCANGAN

III.1 Analisis Sistem

Analisis sistem dilakukan untuk memperoleh definisi permasalahan dan gambaran yang tepat dari apa yang akan dilakukan oleh sistem. Analisis sistem bertujuan untuk mengetahui bagaimana seluk beluk sistem yang akan diteliti dan dibangun.

Pada bagian analisis sistem ini akan dibahas mengenai analisa-analisa masalah yang akan dilakukan terhadap *game*, analisa pengguna dan kegunaan dari *game* yang akan dibangun.

III.2 Analisis Masalah

Laut Indonesia kini dihadapkan dengan berbagai masalah diantaranya tingkat pencemaran yang masih tinggi, ditandai dengan terjadinya peningkatan jumlah nutrisi yang disebabkan oleh polutan. Pencemaran laut saat ini banyak berasal dari partikel kimia, limbah industri, limbah pertanian dan perumahan yang bisa merusak lingkungan laut. Kurang lebih 80% pencemaran yang terjadi di laut adalah pencemaran sampah plastik. Sampah-sampah plastik yang terbawa ke laut sebagian terampung dan beberapa waktu kemudian sebagian akan terendap di lautan. Sampah-sampah berupa plastik ini masuk ke laut melalui aliran sungai atau selokan yang pada umumnya langsung menuju laut. Belum lagi masyarakat yang hidup di sekitar pesisir laut dan membuang sampah mereka setiap hari ke laut. Akibatnya tidak sedikit hewan laut seperti penyu, burung dan kura-kura mati akibat menelan sampah plastik [1].

Saat ini *game* memiliki potensi untuk meningkatkan proses pembelajaran menjadi lebih efektif. Karena manusia memiliki sifat dasar lebih cepat mempelajari segala sesuatu secara visual-verbal, sehingga *game* juga baik jika dilibatkan dalam proses pendidikan (*game* edukatif). *Game* selalu dimainkan secara berulang-ulang dan terus-menerus sampai para pemain merasa puas. Ketika

orang bermai *game*, segenap potensinya dikerahkan untuk menyelesaikan setiap *level* dalam *game*. Pikiran fisik, bahkan emosi terlibat. Pada saat seperti ini, bisaanya materi-materi yang disampaikan akan mudah dicerna dan dimengerti oleh pemain *game* [2].

Salah satu *game* yang akan dibahas dan menjadi acuan disini ialah *game Feeding Frenzy. Game* yang akan dibangun akan mengadaptasi *Gameplay Game Feeding Frenzy 2* sebagai lingkungan permainan yang serupa. Oleh karena itu akan dikembangkan tipe *gameplay Game Feeding Frenzy 2* dengan menambahkan unsur edukasi lindungi kebersihan lingkungan laut serta pembelajaran mengenai jenis-jenis sampah.

III.2.1 Analisis Game Sejenis

Pada tahap ini dilakukan analisis terhadap *game* sejenis yang telah ada. Tujuan dari analisis ini yaitu untuk membandingkan dan mendapatkan konsep permainan dari *game* yang sudah ada untuk kemudian dikembangkan. *Game* yang dibandingkan dan akan menjadi acuan adalah *game Feeding Frenzy 2, Fish Predator* dan *Hungry Shark Evolution* .

1. ***Game Feeding Frenzy 2***



Gambar III.1 *Game Feeding Frenzy 2*

Feeding Frenzy 2 adalah sebuah *game* air *arcade-style* ditulis oleh Sprout *Game*s dan diterbitkan oleh PopCap *Game*s dengan debut awal tahun 2004**.** *Game* ini menghadirkan sebuah simulasi yang seolah-olah menjadikan pemain *game* (*player*) untuk masuk kedalam dunia bawah laut dan menjelma menjadi seekor ikan. **Feeding Frenzy 2** mencakup 60 tingkatan yang menampilkan pemandangan

seperti dunia bawah laut baru, terumbu karang, gua laut dalam, dan kapal tenggelam, serta tantangan di atas air.

a. Gameplay

Pemain harus mengendalikan beberapa cepat tumbuh predator laut yang keluar untuk mengungkap misteri yang mengintai di lautan. Misi dari permainan ini adalah untuk menghindari predator dan kendala sewaktu makan ikan kecil dan makhluk lain, akhirnya mencapai puncak rantai makanan.

b. Kartakter

Karakter yang ada pada *game Game Feeding Frenzy2* antara lain:

Tabel III.1 Karakter *Game Feeding Frenzy2*

No	Nama Karakter	Deskripsi
1.	Boris – Boris	Ikan kupu-kupu adalah karakter pertama dimainkan dalam permainan. Dia tinggal di perairan hangat dari Shoal Sandy, sebuah daerah yang indah dari Pantai Frenzy.
2.	Layla – Layla	triggerfish adalah karakter dimainkan kedua dalam permainan. Dia tinggal di air hangat surga Reef.
3.	Edie – Edie	anglerfish adalah karakter yang dapat dimainkan ketiga. tingkat Edie sulit karena tingkat-nya adalah "Midnight Zona". Perhatikan bahwa gender ikan telah diubah menjadi perempuan karena fakta bahwa hanya memancarkan cahaya pemancing ikan betina, ini adalah kesalahan dalam permainan asli.
4.	Peter – Peter	ikan kuwe adalah karakter yang dapat dimainkan keempat dalam permainan. Dia tinggal di Buccaneer Bay, di mana pemain pertama akan menemukan sotong yang menembak tinta.
5.	Harry	Harry ikan humphead adalah karakter yang dapat dimainkan kelima dalam permainan, para Balon yang kembali setelah beberapa saat dan Anda dapat kembali melompat untuk bonus keluar dari air.
6.	Goliath	Goliat, hiu putih besar adalah karakter yang dapat dimainkan keenam dalam permainan. Dia sangat tinggi dalam rantai makanan, dan adalah yang pertama untuk memerangi "The Intruder".
7.	The Intruder	The Intruder merupakan ikan asing mengganggu rantai makanan. Pemain diperlukan untuk melawan setiap 3 *level* setelah Goliat kekalahan untuk pertama kalinya. Pemain kemudian bergerak rendah ke bawah rantai makanan,

No	Nama Karakter	Deskripsi
		bermain sebagai karakter sebelumnya sampai The Intruder dikalahkan untuk terakhir kalinya, setelah Boris memakannya. Ini melengkapi permainan. The Intruder kemudian membuka sebagai karakter yang dapat dimainkan, dan pemain kemudian dapat bermain melalui seluruh permainan lagi sebagai The Intruder.

2. Game Hungry Shark Evolution



Gambar III.2 *Game Hungry Shark Evolution*

*Game Hungry Shark Evolution m*erupakan *game* buatan Future *Game*s of London dirilis bulan September 2012. Hungry Shark Evolution, sebuah *game* android dengan permainan cukup mudah.

a. Gameplay

Gameplay-nya sebenarnya tidak terlalu berbeda dengan pendahulunya dimana pemain akan memainkan sebuah hiu yang selalu lapar dan harus memakan apapun yang bisa ia makan untuk bertahan hidup. Sedikit mirip dengan *Feeding Frenzy*. Pemain bisa mengendalikan sang hiu dengan tilt atau D-pad untuk berenang kesana-kemari. Ketika menemukan mangsa maka pemain cukup mengarahkan sang hiu ke dekatnya dan otomatis hiu akan memakannya.

Pemain juga bisa tap dimana saja pada layar untuk membuat hiu melesat lebih kencang, namun ini hanya bisa digunakan secara terbatas. Setelah boost habis maka pemain harus menunggu sampai bisa digunakan kembali. Boost ini berfungsi untuk mengejar beberapa mangsa yang sangat cepat seperti barracuda

dan juga bisa digunakan untuk meloncat tinggi dari air untuk menangkap pelican yang sedang terbang.

Pemain akan memiliki sebuah HP bar yang berada di kanan atas layar. HP bar ini akan terus menerus turun seiring waktu dan satu-satunya cara menambah HP ini adalah dengan memakan ikan atau binatang lain, jadi game akan berakhir jika HP pemain habis akibat tidak makan.

Beda tipe ikan beda juga tingkat HP yang diberikan, semakin besar mangsa maka HP yang akan pemain dapatkan semakin besar juga, yang unik di game ini adalah pemain juga dapat menemukan manusia yang sedang berenang atau diving, dan ya pemain juga bisa memakannya. Tapi beberapa manusia khususnya *diver* akan menyerang pemain jadi sebaiknya hati-hati. Selain manusia ada berbagai objet seperti limbah, bom atau ubur-ubur yang dapat mengurangi HP pemain.

Setiap pemain bermain maka pemain akan selalu bermain ditempat yang sama, tapi serunya tempat yang menjadi habitat pemain ini cukup luas dan pemain bisa menjelajah dengan cukup nyaman tanpa merasa bosan terlalu cepat. Seiring penjelajahan pemain juga dapat menemukan keong yang tergeletak ditempat-tempat tertentu, keong ini jika pemain ambil akan memberikan sebuah misi tambahan seperti memangsa 3 manusia atau berenang sejauh 500m.

Bagian yang tidak kalah serunya dari *game* ini adalah *upgrad*e. Pemain bisa *upgrade skill* (lebih cepat memakan atau lebih cepat bergerak) dan tipe hiu serta mendandaninya lengkap dengan topi atau sirip baru yang sangat *eye catching*. Hiu awal yang pemain kendalikan adalah hiu yang paling sederhana dan kecil, hiu ini mempunyai batas menyelam yang cukup rendah dan jika pemain mencoba untuk menyelam lebih jauh maka pemain akan terluka. Sehingga jika ingin lanjut maka pemain harus *upgrade* dan menjadi hiu tipe baru yang lebih hebat lagi.

b. Persentasi

Hungry Shark Evolution mempunyai grafis 3D yang sangat baik, ketika pemain menyelam maka pemain bisa melihat bahwa lautan ini mempunyai dimensi kedalaman yang cukup nyata. Tipe ikan yang ada juga sangat beragam apa lagi ketika pemain mulai bisa menyelam di kedalaman yang lebih dalam maka pemain mulai akan menemukan ikan dengan bentuk yang aneh, yang cukup mengagumkan dari Hungry Shark Evolution adalah ketika pemain menyerang manusia atau burung dari bawah air maka gerakan pemain akan menjadi lambat dan layar menjadi *zoom*.

c. Karakter

Karakter yang ada pada *game Hungry Shark Evolution* antara lain:

Tabel III.2 Karakter *Hungry Shark Evolution*

No	Nama Karakter	Deskripsi
1.	Mako Shark	Mako Shark adalah jenis hiu yang dapat dibeli dengan harga 1400 koin. Mako Shark lebih cepat dan lebih kuat daripada Reef Shark, dan dapat menyelam jauh lebih dalam. Mako Shark mempunyai kemampuan berenang dengan kedalaman 150 meter, ukuran maximal 5 meter. Santapan: Semuanya Reef Shark bisa makan, serta mangsa yang lebih agresif .
2.	Hammerhead Shark	Hammerhead Shark adalah salah satu hiu yang paling karismatik dan mampu makan lebih banyak makhluk dari Reef dan Mako hiu. Hammerhead Shark dapat dibeli dengan harga 6.000 Koin. Hammerhead Shark mempunyai kemampuan berenang dengan kedalaman 200 meter, ukuran maximal 5 meter .
3.	Tiger Shark	Tiger Shark adalah hiu sangat kuat dan akan bertahan sebagian besar serangan - mereka juga bisa makan orang sial Puffer Fish. Tiger Shark dapat dibeli dengan harga 15.000 Koin. Tiger Shark mempunyai kemampuan berenang dengan kedalaman 275 meter, ukuran maximal 7 meter .
4.	Great White Shark	Great White Shark adalah Great White Sharks adalah makhluk yang paling kuat di laut dan pada dasarnya tujuan akhir di Hungry Shark Evolution. Great White Shark dapat dibeli dengan harga 35.000 Koin. Tiger Shark mempunyai kemampuan berenang dengan kedalaman yang tak terbatas, ukuran maximal 10 meter .

No	Nama Karakter	Deskripsi
5.	Megalodon	Megalodon adalah nenek moyang legendaris hiu dan makhluk akhir yang tersedia di Hungry Shark Evolution. Ini adalah hiu yang paling kuat dan setelah Anda membelinya maka secara otomatis memiliki gigitan maksimum, kecepatan dan meningkatkan atribut. Sebagian besar misi Megalodon yang berlangsung terhadap permukaan air.

3. *Game Hungry Piranha*



Gambar III.3 *Game Hungry Piranha*

Permainan ini dilakukan pada prinsip evolusi. *Gameplay game* ini hampir sama dengan *game Feeding Frenzy*.

a. Gameplay

Pemain akan menjadi seekor piranha kecil, dimana piranha harus makan ikan lebih kecil dari piranha agar bisa tumbuh. Piranha harus makan terus-menerus untuk menjadi lebih besar dan dapat mempertahankan kekuasaan. Piranha harus selalu menghindari ikan yang lebih besar sehingga piranha tidak akan menjadi mangsa mereka.

b. Karakter

Karakter yang ada pada *game Hungry Piranha* antara lain:

Tabel III.3 Karakter Hungry Piranha

No	Nama Karakter	Deskripsi
1.	Piranha	Adalah karakter utama seekor ikan piranha yang harus makan terus menerus agar bisa tumbuh dan mempertahankan kekuasaan

Perbandingan *game* yang menjadi acuan dapat dilihat pada Tabel III.4 berikut ini:

Tabel III.4 Perbandingan Game Sejenis

	Game Feeding Frenzy2	***Game Hungry Piranha***	***Game Hungry Shark Evolution***	***Game yang akan dibangun***
Platform	PC	PC	Android	PC
Gendre	*Action*	*Action*	*Action*	*Action*
User	Single user	Single user	Single user	Single user
Story	Tidak memiliki cerita, hanya ikan yang memakan ikan lain untuk terus berubah menjadi varian / jenis ikan lain	Permainan ini mengunakan prinsip evolusi, dimana ikan piranha akan terus tumbuh besar jika memakan ikan lain yang lebih kecil	Permainan ini menceritakan hiu yang selalu lapar dan harus memakan apapun yang bisa ia makan untuk bertahan hidup. Termasuk bisa memakan manusia.	Pertualangan seorang anak yang bertekad untuk menjaga kebersihan laut dengan cara mengumpulkan semua sampah yang ada dilaut dan mempelajari jenis sampah sehingga ia tahu jenis sampah dan tempat penyimpananya.
Pesan yang Disampaikan	Tidak Ada	Tidak Ada	Hiu adalah hewan laut yang ganas dan bisa memakan manusia	Pentingnya menjaga kebersihan, terutama mengenai membuang sampah. Karena sampah dapat membuat dampak buruk bagi kehidupan laut. Contoh dari dampak sampah yang ada dilaut adalah banyak kura-kura yang mati akibat menelan sampah.
Konten Edukasi	Tidak ada	Tidak ada	Tidak ada	Ada

Dilihat dari konsep permainan, baik *Feeding Frenzy2, Hungry Piranha* ataupun *Hungry Shark Evolution* memiliki konsep permainan yang hampir sama. Ketiga *game* yang dibahas lebih menekankan pada hiburan. Oleh sebab itu, ketiga *game* tersebut dapat dikembangkan dengan menambahkan unsur edukasi didalamnya. Dengan penambahan unsur edukasi ini, diharapkan *game* yang akan dikembangkan dapat dijadikan sarana pembelajaran yang menyenangkan.

III.2.2 Analisis *Game* yang Akan Dikembangkan

Game edukasi Lindungi Kebersihan Laut yang akan dibangun yaitu *game* yang mempunyai grafis 2D yang ber-*genre action. Game* ini mempunyai *gameplay* yang serupa dengan *game* sejenis seperti *Feeding Frenzy2*.

Game dibangun menggunakan lingkungan laut sebagai lingkungan permainannya dengan tujuan agar mengenalkan keindahan bawah laut yang penuh dengan biota laut didalamnya, dengan *gendre game action* maka dibutuhkan kecepatan reflex, akurasi, dan ketepatan waktu untuk menghadapi sebuah rintangan. Sehingga *game* ini baik digunakan untuk melatih tingkat kesabaran, reflex, akurasi dan ketepatan waktu pada anak agar anak bisa lebih aktif.

Beberapa fitur yang ditawarkan di dalam *game* ini:

1. *Single player*
2. Lingkungan laut yang berbeda di tiap *level*
3. Menyertakan pesan edukasi
4. Terdapat *minigame* di tiap *level*, dan
5. *Game* dibuat dengan grafis 2D

III.2.2.1 Storyline

Game edukasi lindungi kehidupan laut ini merupakan *game* edukasi ber-*genre action*, yang dimainkan oleh satu pemain. *Game* ini menceritakan tentang seorang anak yang bernama Toni yang berusaha untuk menyelamatkan kehidupan laut dari sampah dengan cara mengumpulkan sampah-sampah tersebut dari dalam laut.

Cerita dimulai saat Toni yang pada saat itu melihat laut dipenuhi dengan sampah, sampah-sampah tersebut termakan oleh hewan laut seperti kura-kura yang akhirnya mati karena menelan sampah tersebut. Toni yang merupakan salah satu orang yang sangat perduli terhadap kebersihan dan kehidupan laut, tergerak untuk menyelamatkan kehidupan laut dari sampah-sampah yang ada di laut. Toni harus mengumpulkan sampah-sampah yang ada di tiga laut, yaitu:

1. Laut *level* 1
2. Laut *level* 2
3. Laut *level* 3

Disetiap *level* pemain harus mengumpulkan berbagai macam sampah. Sampah-sampah yang harus dikumpulkan antara lain adalah sampah organik dan sampah anorganik seperti botol, kaleng, daun kering, kantong plastik dan sebagainya. Pada akhir *level* pertama pemain harus menyimpan sampah-sampah yang telah dikumpulkan ke tempat sampah sesuai dengan jenis sampah tersebut.

Pada *level* kedua, pemain harus mengumpulkan sampah organik dan anorganik sebanyak-banyaknya. Namun pada *level* ini, pemain akan dihadapkan dengan adanya gangguan ubur-ubur yang dapat menyengat pemain. Di *level* ini juga pemain akan dikejutkan dengan kedatangan hiu yang dapat memangsa pemain dan mengurangi nyawa pemain. Diakhir *level*, pemain akan dihadapkan pada *minigame* kalahkan AI , pada *game* ini pemain harus membuat garis horizontal, vertikal, maupun diagonal,.

Pada *level* ketiga, pemain harus mengumpulkan sampah organik dan anorganik sebanyak-banyaknya. Sama seperti pada *level* sebelumnya, pada *level* ini pemain akan dihadapkan dengan adanya gangguan ubur-ubur yang dapat menyengat pemain. Di *level* ini juga pemain akan dikejutkan dengan kedatangan hiu yang dapat memangsa pemain dan mengurangi nyawa pemain. Diakhir *level*, pemain akan dihadapkan pada *minigame pilih jenis sampah*, pada *game* ini pemain harus men-*drag & drop* gambar sampah dan meletakkan sampah-sampah tersebut pada tempat yang telah disediakan sesuai jenisnya. Jika pemain telah menyelesaikan *minigame* ini maka permainan akan selesai.

III.2.2.2 Analisis Tingkat Kesulitan

Pada *game* edukasi yang akan dibangun ini, terdiri dari 3 *level* kesulitan. Berikut ini rincian penjelasan tingkat kesulitan pada setiap *level* nya yang dapat dilihat pada tabel III.5.

Tabel III.5 Deskripsi Tingkat Kesulitan

Level	**Deskripsi *Level***	**Jumlah Musuh**
1	Jumlah sampah yang harus dikumpulkan sedikit dan waktu yang dibutuhkan untuk mengumpulkan sampah-sampah tersebut cukup lama.	0
2	Jumlah sampah yang harus dikumpulakan semakin bertambah dan waktu pengambilan sampahnya terbatas, ditambah lagi adanya kedatangan ubur-ubur yang dapat menyengat pemain dan hiu yang dapat memakan pemain.	10
3	Jumlah sampah yang harus dikumpulakan semakin bertambah dan waktu pengambilan sampahnya sangat singkat, ditambah lagi adanya kedatangan ubur-ubur yang dapat menyengat pemain dan hiu yang dapat memakan pemain.	15

III.2.2.3 Gameplay

Game yang akan dibangun bersifat *single player* (satu pemain). *Game* ini ditujukan untuk memberikan pembelajaran dalam menjaga kebersihan. Pada tampilan awal permainan terdapat *intro game* berupa cerita dibalik *game* yang akan dimainkan Pada *menu* utama terdapat beberapa pilihan yaitu main, nilai tertinggi, bantuan dan keluar. Rincian penjelasan mengenai *menu* utama tersebut adalah sebagai berikut:

1. Tombol mulai bermain berfungsi untuk menuju tampilan pilihan main.
2. Tombol nilai tertinggi berfungsi untuk informasi nilai / *point* yang didapatkan pemain.
3. Tombol bantuan berfungsi untuk memberikan informasi tentang aturan-aturan dan petunjuk permainan dalam memainkan *game* ini.
4. Tombol keluar berfungsi untuk menutup aplikasi.

Tugas pemain dalam memainkan *game* ini adalah mengumpulkan sampah-sampah organik dan anorganik, serta menyelamatkan kura-kura yang bisa memakan sampah-sampah tersebut yang dapat mengakibatkan kematian. Pemain

diberikan 3 nyawa dalam permainan ini. Pemain diharuskan mengumpulkan sampah-sampah organik dan anorganik sehingga pemain akan mendapatkan *point.* Pemain juga dituntut untuk menghindari sengatan ubur-ubur dan ikan hiu yang dapat mengurangi *point* bahkan dapat mengurangi nyawa.

III.2.2.4 Scoring

Pada setiap *level*, setiap kali mengumpulkan sampah baik organik atau sampah anorganik, pemain akan mendapatkan *point*, dan *point* akan diakumulasikan sesuai jenis sampah yang diperoleh oleh pemain. Berikut ini rincian penjelasan *point* yang diperoleh pada setiap *level*-nya yang dapat dilihat pada tabel III.6.

Tabel III.6 Analisis Scoring Terhadap Sampah

Level	Jenis Sampah	Jumlah	*Point*
1	**Organik**	**40**	**5**
	Anorganik		**10**
2	**Organik**	**60**	**10**
	Anorganik		**20**
3	**Organik**	**90**	**15**
	Anorganik		**30**

Tabel III.7 Analisis Scoring Terhadap Musuh

Jenis	*Minus Point*	*Minus Nyawa*
Sampah anorganik dimakan kura-kura	**10**	**-**
Tabrakan dengan ubur-ubur	**10**	**-**
Pemain dimakan hiu	**-**	**1**

III.3 Analisis Algoritma Fisher-Yates Shuffle

Algoritma *Fisher-Yates Shuffle* akan diimplementasikan pada *game* lindungi kebersihan laut untuk mengubah urutan masukan objek sampah yang diberikan secara acak**.** Permutasi yang dihasilkan oleh algoritma ini muncul dengan probabilitas yang sama. Metode dasar yang diberikan untuk menghasilkan permutasi acak dari angka 1 - *N* berjalan sebagai berikut:

1. Tuliskan angka dari 1 sampai n
2. Isi nilai k dengan bilangan acak antara 0 hingga i+1 bulatkan kebawah

3. Hitung dari low end, gantikan nilai k dan tuliskan di tempat lain
4. Ulangi dari langkah 2 sampai semua nomor digantikan
5. Urutan angka yang tertulis di langkah 3 sekarang permutasi acak dari nomor asli.

Jika diimplementasikan pada game yang dibangun, maka berikut merupakan analisisnya:

Psudocode dengan menggunakan bahasa algoritmik:

```
Deklarasi :
    n     :  integer {jumlah objek sampah yang akan diacak}
    i     :  integer {pencacah banyaknya pengulangan}
    k     :  integer {tempat penyimpanan sementara nilai i+1 yang telah
             dirandom dan dibulatkan ke bawah}
    temp  :  integer {tempat penyimpanan sementara nilai arrayshuffle[k]}
Algoritma :
Read (n)
For i ← n-1 downto 0 do
      ArrayShuffle[i] ← i
endfor
While i ≥ 0 do
      k ← floor(random(i+1))
      Temp ←ArrayShuffle[k]
      ArrayShuffle[k] ←ArrayShuffle[i]
      ArrayShuffle[i] ← temp
      i ← i + 1
endwhile
```

Gambar III.4 *Psudocode* Algoritma Fisher Yates

Dengan algoritma diatas maka akan muncul urutan objek sampah yang diberikan secara acak**.** Hasil dari perhitungan didapatkan kemunculan urutan objek sampah dengan jumlah i=40 sebagai berikut:

Tabel III.8 Kemunculan Objek Dengan Menggunakan Fisher Yates

i	Objek Sampah Yang Muncul	
6.		Sampah 1 (Sampah Anorganik)
2.		Sampah 6 (Sampah Anorganik)
3.		Sampah 2 (Sampah Anorganik)
4.		Sampah 7 (Sampah Anorganik)
5.		Sampah 9 (Sampah Organik)
6.		Sampah 5 (Sampah Organik)
7.		Sampah 1 (Sampah Anorganik)
8.		Sampah 10 (Sampah Organik)
9.		Sampah 8 (Sampah Organik)
10.		Sampah 3 (Sampah Organik)
11.		Sampah 4 (Sampah Anorganik)
12.		Sampah 6 (Sampah Anorganik)
13.		Sampah 2 (Sampah Anorganik)
14.		Sampah 7 (Sampah Anorganik)
15.		Sampah 9 (Sampah Organik)

i	Objek Sampah Yang Muncul	
16.		Sampah 5 (Sampah Organik)
17.		Sampah 1 (Sampah Anorganik)
18.		Sampah 9 (Sampah Organik)
19.		Sampah 8 (Sampah Organik)
20.		Sampah 3 (Sampah Organik)
21.		Sampah 4 (Sampah Anorganik)
22.		Sampah 6 (Sampah Anorganik)
23.		Sampah 2 (Sampah Anorganik)
24.		Sampah 9 (Sampah Organik)
25.		Sampah 5 (Sampah Organik)
26.		Sampah 1 (Sampah Anorganik)
27.		Sampah 10 (Sampah Organik)
28.		Sampah 8 (Sampah Organik)
29.		Sampah 3 (Sampah Organik)
30.		Sampah 4 (Sampah Anorganik)
31.		Sampah 6 (Sampah Anorganik)
32.		Sampah 2 (Sampah Anorganik)

i	Objek Sampah Yang Muncul	
33.		Sampah 7 (Sampah Anorganik)
34.		Sampah 9 (Sampah Organik)
35.		Sampah 5 (Sampah Organik)
36.		Sampah 1 (Sampah Anorganik)
37.		Sampah 10 (Sampah Organik)
38.		Sampah 8 (Sampah Organik)
39.		Sampah 3 (Sampah Organik)
40.		Sampah 4 (Sampah Anorganik)

Dari hasil kemunculan objek sampah tersebut, maka berikut jumlah kemunculan terhadap masing-maing jenis sampah:

Tabel III.9 Jumlah Kemunculan Terhadap Masing-Masing Jenis Sampah

i	Objek Sampah Yang Muncul		Jumlah Kemunculan
1.		Sampah 1 (Sampah Anorganik)	5
2.		Sampah 2 (Sampah Anorganik)	4
3.		Sampah 3 (Sampah Organik)	4
4.		Sampah 4 (Sampah Anorganik)	4
5.		Sampah 5 (Sampah Organik)	4

i	Objek Sampah Yang Muncul		Jumlah Kemunculan
6.		Sampah 6 (Sampah Anorganik)	4
7.		Sampah 7 (Sampah Anorganik)	3
8.		Sampah 8 (Sampah Organik)	4
9.		Sampah 9 (Sampah Organik)	5
10.		Sampah 10 (Sampah Organik)	3

Berdasarkan analisis diatas dipastikan dengan menggunakan algoritma fisher Yates semua objek sampah muncul tanpa ada objek sampah yang tak muncul.

III.4 Analisis Algoritma Minimax Pada Minigame Kalahkan AI

Permainan *minigame* kalahkan AI adalah sebuah permainan yang menggunakan papan berukuran n baris dan m kolom sehingga ukuran papan menjadi n x m. "*Minigame* Kalahkan AI" ini disajikan permainan pada papan berukuran 3 x 3. Pemain harus mengisi bagian-bagian tersebut sedemikian rupa sehingga karakter yang dimasukkan pemain tersebut dapat membentuk suatu garis *horizontal, vertikal*, ataupun *diagonal*. Permainan ini dimainkan oleh 2 orang pemain, tapi pada versi permainan komputer, pemain lawan dapat digantikan oleh komputer. Dalam permainan ini hasilnya dapat berupa menang,kalah, ataupun seri.

Pada permainan ini dengan adanya *AI* yang mampu meminimalisir kemungkinan untuk pemain menang, permainan ini akan menjadi sangat sulit untuk dimenangkan oleh pemain. Bahkan kemungkinan terbaik untuk pemain hanyalah seri. Dengan kata lain dengan menggunakan algoritma *minimax* ini, komputer tidak akah pernah kalah.

Pada algoritma minimax, pengecekan dilakukan pada seluruh kemungkinan yang ada sampai akhir permainan. Pengecekan tersebut akan menghasilkan pohon permainan yang berisi semua kemungkinan tersebut. Algoritma minimax ini bekerja secara rekursif dengan mencari langkah yang akan membuat lawan mengalami kerugian minimum. Semua strategi lawan akan dihitung dengan algoritma yang sama dan seterusnya. Ini berarti, pada langkah pertama komputer akan menganalisis seluruh pohon permainan. Dan untuk setiap langkahnya, komputer akan memilih langkah yang paling membuat lawan mendapatkan keuntungan minimum, dan yang paling membuat komputer itu sendiri mendapatkan keuntungan maksimum.

Dalam penentuan keputusan tersebut dibutuhkan suatu nilai yang merepresentasikan kerugian atau keuntungan yang akan diperoleh jika langkah tersebut dipilih. Untuk itulah disini digunakan sebuah fungsi *heurisitic* untuk mengevaluasi nilai sebagai nilai yang merepresentasikan hasil permainan yang akan terjadi jika langkah tersebut dipilih.

Pada permainan ini akan digunakan nilai 1,0,-1 untuk mewakilkan hasil akhir permainan berupa menang, seri, dan kalah. Dari nilai-nilai *heuristic* inilah komputer akan menentukan simpul mana dari pohon permainan yang akan dipilih, tentunya simpul yang akan dipilih tersebut adalah simpul dengan nilai *heuristic* yang akan menuntun permainan ke hasil akhir yang menguntungkan bagi komputer.

Berikut ini merupakan *Psudocode* algoritma minimax:

Deklarasi :

node	:	{simpul}
depth	:	{kedalaman pencarian}
maximizingPlayer	:	{untuk memaksimalisasi peluang menang}
bestValue	:	{merepresentasikan sebagaimana bagusnya suatu langkah dalam permainan}
child	:	{mempresentasikan anak dari *node*}

Algoritma:

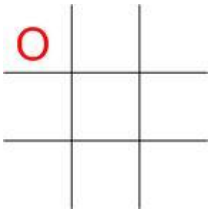
```
function minimax(node, depth, maximizingPlayer)
   if kedalaman = 0 atau node tidak mempunyai anak
      return nilai node heuristik
  if maximizingPlayer
      bestValue ← -1
      untuk setiap anak dari node
         val ← minimax(child, depth - 1, FALSE))
         bestValue ← max(bestValue, val);
      return bestValue
  else
      bestValue ← +1
      untuk setiap anak dari node
         val ← minimax(child, depth - 1, TRUE))
         bestValue ← min(bestValue, val);
      return bestValue

(* Initial call for maximizing player *)
minimax(origin, depth, TRUE)
```

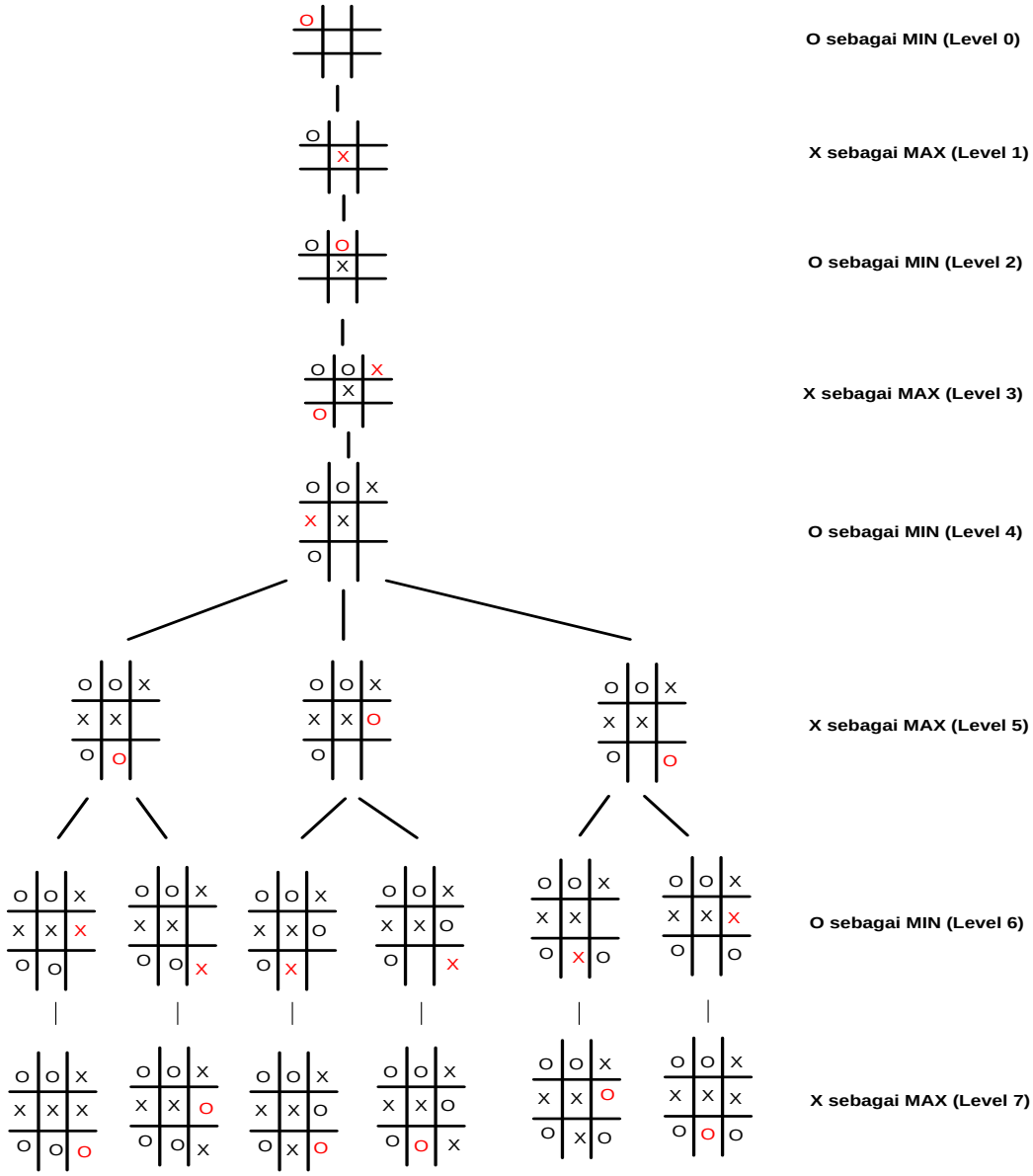
Gambar III.5 *Psudocode* Algoritma Minimax

Agar lebih memudahkan analisis diberikan gambar O untuk pemain dan X untuk komputer. Pemain (O) diberikan giliran pertama untuk memulai permainan, kemudian komputer (X) akan melakukan giliran. Dalam analisis ini diberikan tanda Merah (■) untuk menggambarkan siapa yang sedang bermain.

Misalnya pemain memilih langkah awal untuk meletakkan simbol O seperti dibawah ini:



Langkah apa yang sebaiknya dilakukan oleh komputer? Berdasarkan keadaan ini, komputer bisa membuat pohon pencarian yang berisi seluruh kemungkinan langkah yang bisa diambil oleh O dan X.



Gambar III.6 Pohon Pencarian Komputer

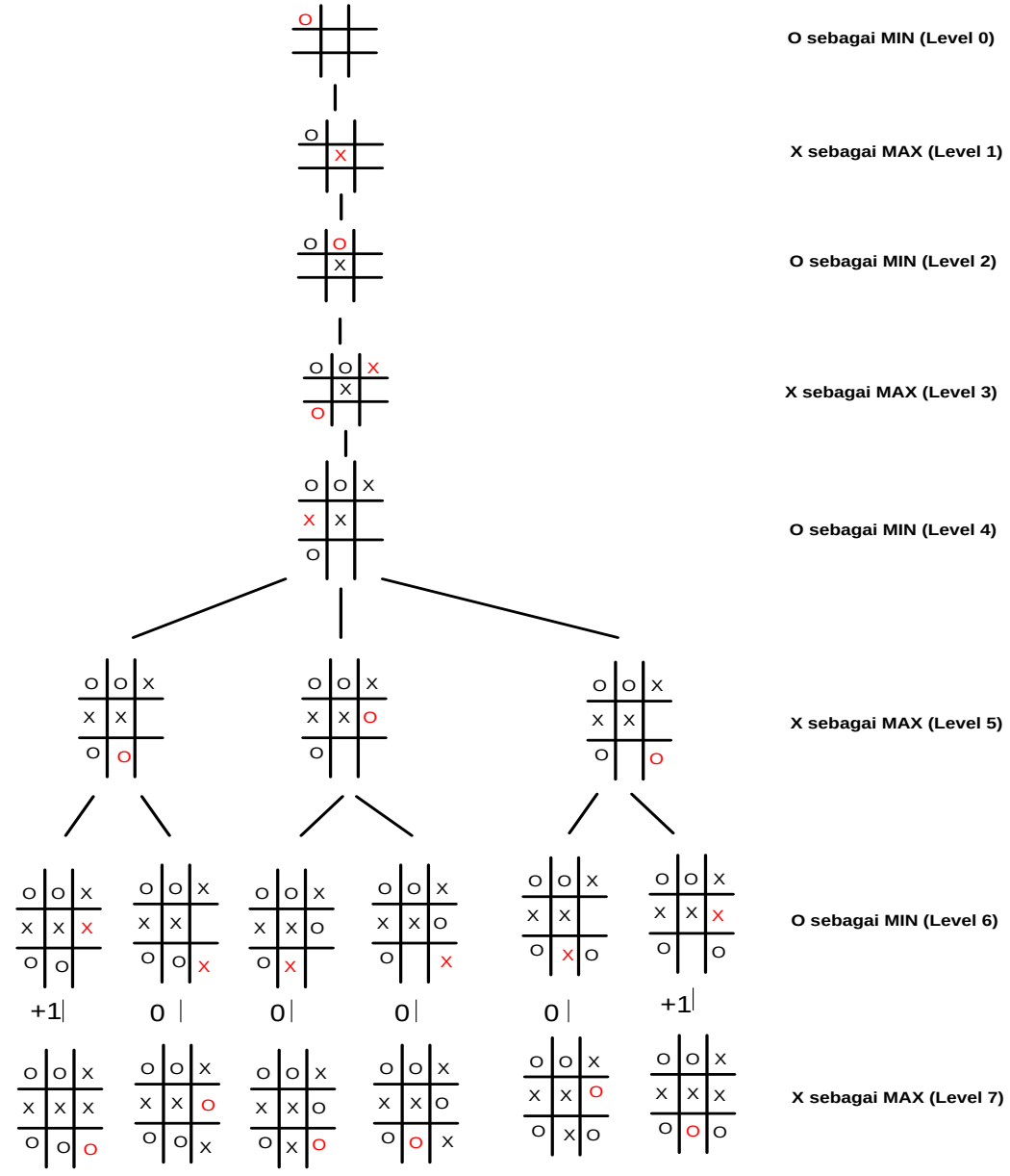
Kemudian, komputer menganalisis keadaan tersebut berdasarkan prosedur minimax. *Node* akar (paling atas) menunjukkan kedaan papan sekarang, sedangkan *node-node* anak menunjukkan kemungkinan langkah-langkah yang dapat dijalankan. Fungsi evaluasi yang digunakan adalah:

f(n) = 1 bila X (komputer) menang atau O (pemain) kalah

f(n) = -1 bila O(pemain) menang atau X (komputer) kalah

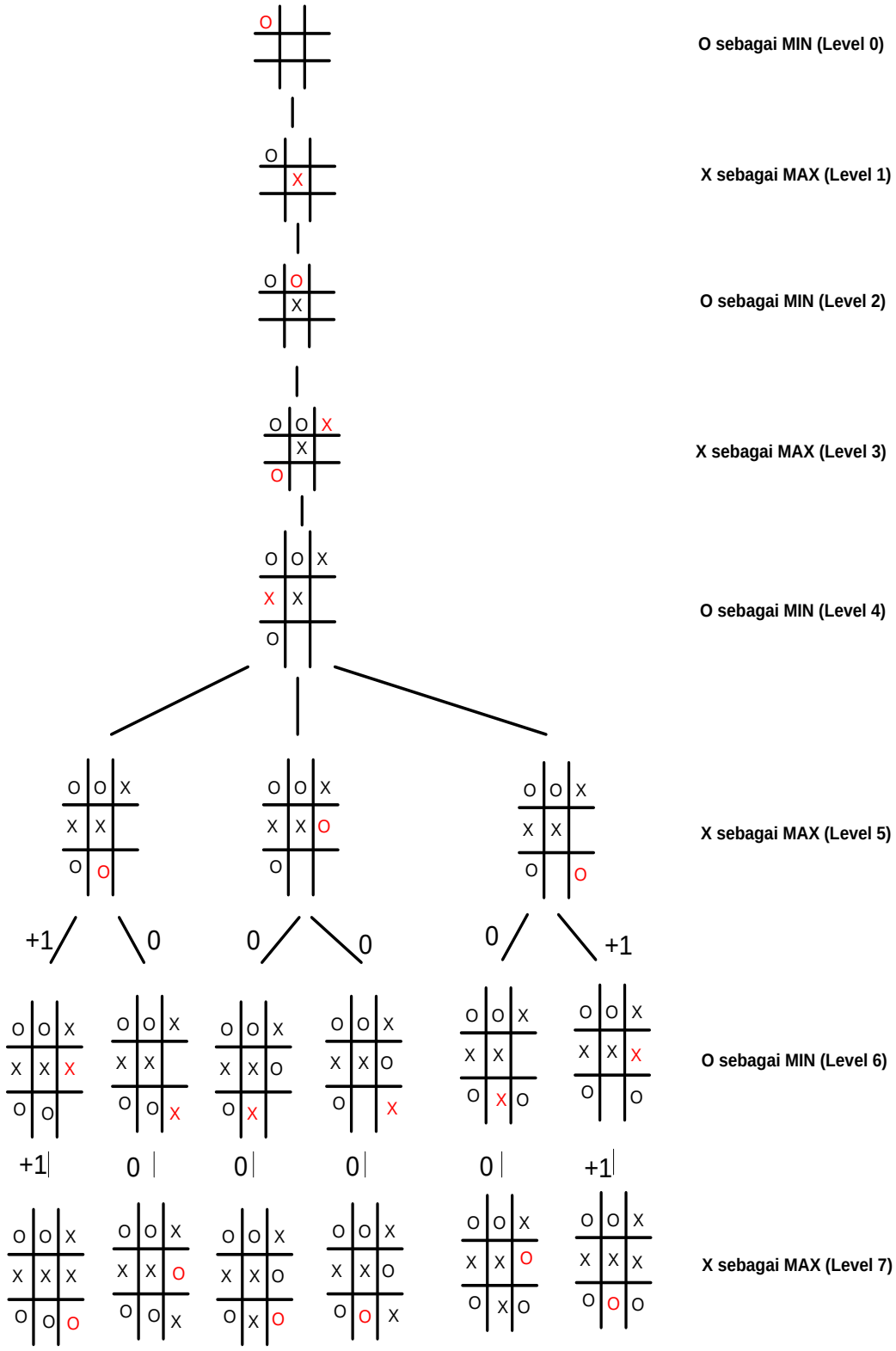
f(n) = 0 bila keduannya bermain seri

Komputer akan melihat *node-node* paling bawah (*level* 7) terlebih dahulu dan sekaligus memberikan skor sebagai berikut:



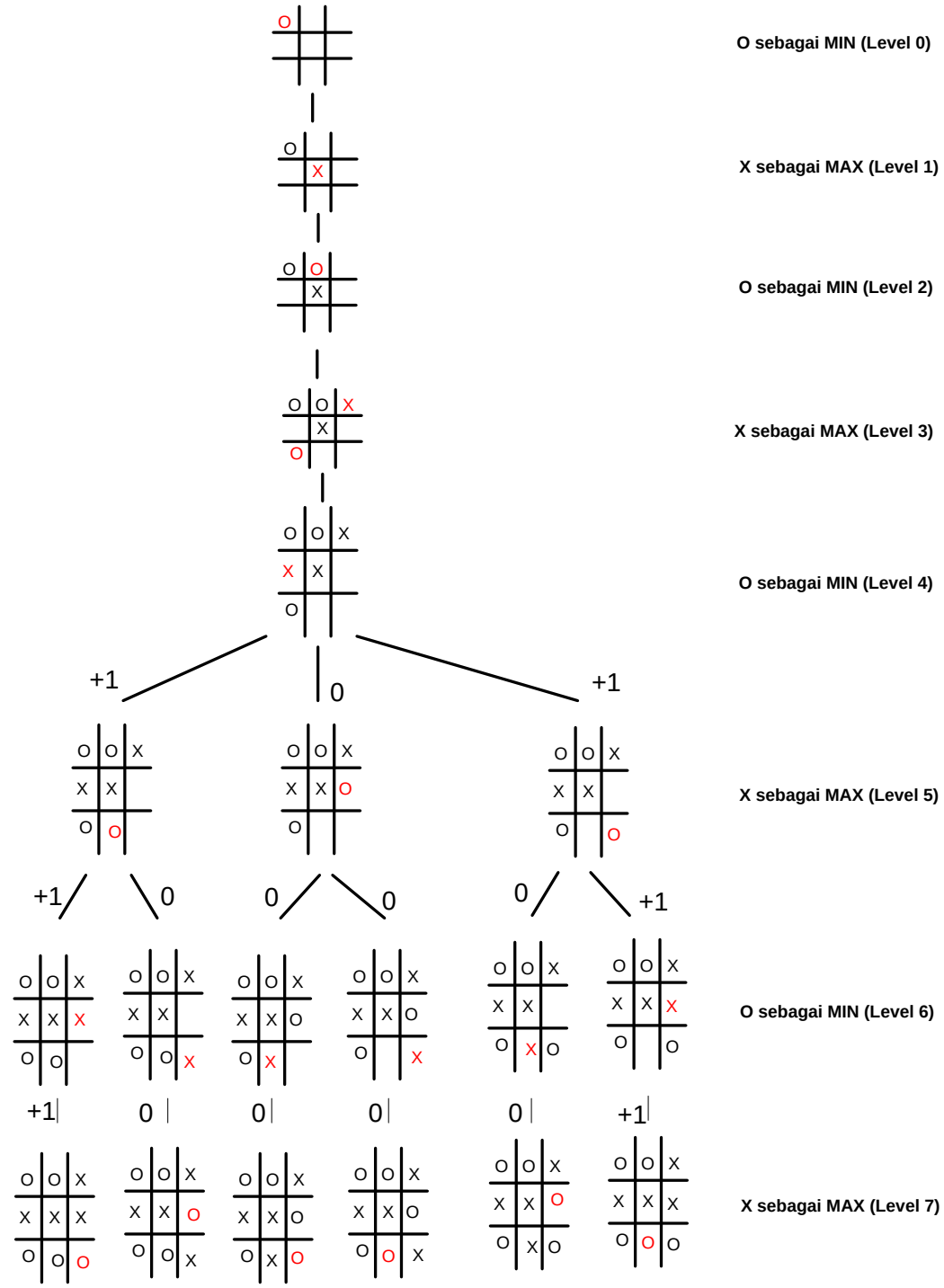
Gambar III.7 Pohon Pencarian Komputer *Level* 7

Kemudian komputer akan melihat *node-node* diatasnya di (*level* 6) dan memberikan skor berdasarkan *level* 7 sebagai berikut:



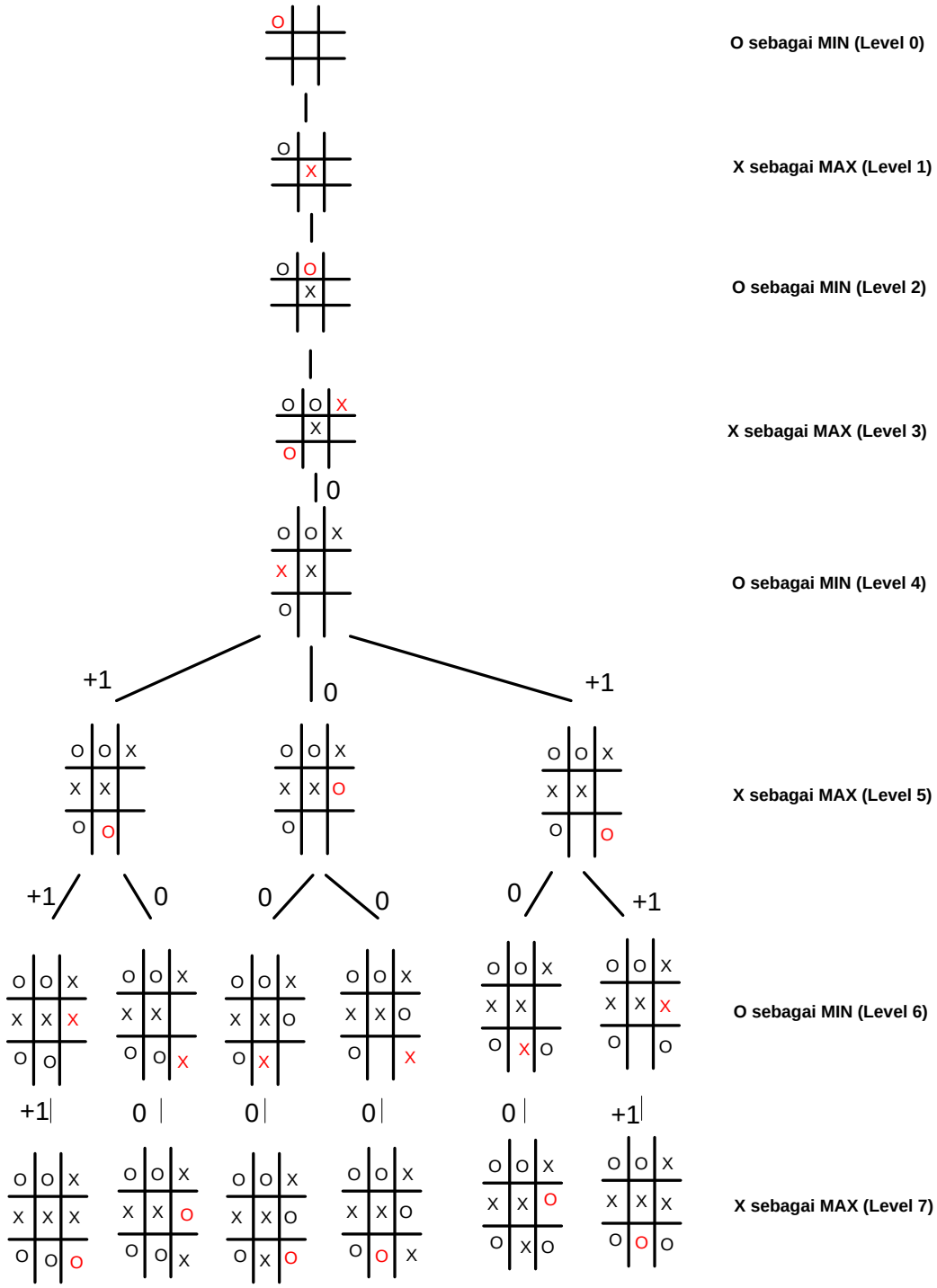
Gambar III.8 Pohon Pencarian Komputer *Level* 6

Kemudian komputer akan melihat *node-node* diatasnya (*level* 5) dan memberikannya skor berdasarkan *level* 6 sebagai berikut:



Gambar III.9 Pohon Pencarian Komputer *Level* 5

Kemudian komputer akan melihat *node-node* diatasnya (*level* 4) dan memberinya skor berdasarkan *level* 5 sebagai berikut:



Gambar III.10 Pohon Pencarian Komputer *Level* 4

Berdasarkan hasil analisis prosedur minimax ini agar komputer menang pada *level* 5 komputer akan mengambil langkah node paling kiri (+1) atau kanan (+1). Sehingga dapat disimpulkan jika dengan langkah-langkah seperti di atas, pemain (O) tidak akan menang. Paling tidak pemain (O) akan bermain seri, karena komputer(X) pasti menang jika menggunakan langkah-langkah diatas.

III.5 Analisis Kebutuhan Non Fungsional

Analisis non-fungsional merupakan analisis yang dibutuhkan untuk menentukan spesifikasi kebutuhan *system*. Spesifikasi ini juga meliputi elemen atau komponen-komponen apa saja yang dibutuhkan untuk *system* yang akan dibangun sampai dengan *system* tersebut diimplementasikan. Analisis kebutuhan ini juga menentukan spesifikasi masukan yang diperlukan *system*, keluaran yang akan dihasilkan *system* dan proses yang dibutuhkan untuk mengolah masukan sehingga menghasilkan suatu keluaran yang diinginkan.

Pada analisis kebutuhan non-fungsional ini dijelaskan analisis kebutuhan perangkat lunak, analisis kebutuhan perangkat keras, dan analisis pengguna.

III.5.1 Analisis Kebutuhan Perangkat Keras

Perangkat keras atau *hardware* merupakan salah satu hal yang penting karena tanpa *hardware* yang memenuhi syarat, program yang akan dibuat tidak akan dapat berjalan. Agar aplikasi dapat berjalan dengan baik, maka dibutuhkan perangkat keras yang sesuai dengan kebutuhan aplikasi.

Perangkat keras yang dibutuhkan oleh *developer* dalam membangun *game* ini antara lain, dapat dilihat pada tabel III.10 berikut:

Tabel III.10 Spesifikasi Perangkat Keras (*Developer*)

Nama Perangkat	Spesifikasi
Processor	Dual Core T4400, 220GHz
Memory	2 GB
HDD	320 GB
VGA	512 MB
Mouse	1 Unit
Keyboard	1 Unit

Perangkat keras yang dibutuhkan oleh *user* untuk menggunakan atau menjalankan *game* ini antara lain, dapat dilihat pada tabel III.11 berikut:

Tabel III.11 Spesifikasi Perangkat Keras (*User)*

Nama Perangkat	Spesifikasi
Processor	1 Ghz
Memory	512 MB
HDD	320 GB
VGA	64 MB
Mouse	1 Unit
Keyboard	1 Unit

III.5.2 Analisis Kebutuhan Perangkat Lunak

Perangkat lunak atau *software* merupakan hal yang terpenting dalam mendukung kinerja sebuah sistem. Perangkat lunak digunakan dalam sebuah sistem merupakan perintah-perintah yang diberikan kepada perangkat keras agar dapat saling berinteraksi diantara keduanya. Perangkat lunak yang dibutuhkan untuk membangun *game* edukasi lindungi kebersihan laut ini adalah sebagai berikut:

Perangkat lunak yang dibutuhkan untuk *developer* dalam membangun *game* ini dapat dilihat pada tabel III.12.

Tabel III.12 Spesifikasi Perangkat Lunak (*Developer)*

Nama Perangkat Lunak	Jenis
System Operasi	Windows 7 Premium
Tools Pembangun	Construct2-r140
Web Browser	Google Chrome
Plugin	*Node*-Webkit

Perangkat lunak yang dibutuhkan untuk *user* untuk menjalankan atau menggunakan *game* ini dapat dilihat pada tabel III.13**.**

Tabel III.13 Spesifikasi Perangkat Lunak (*User*)

Nama Perangkat Lunak	Jenis
System Operasi	Windows XP Service Pack 3 or newer,Windows 7
Web browser	Firefox, Google Chrome
Plugin	*Node*-Webkit

III.5.3 Analisis Pengguna

Game yang akan dibangun ini digunakan khusus untuk anak usia 12-14 tahun, tetapi tidak terlepas kemungkinan dapat digunakan oleh kalangan siapapun. Adapun spesifikasi secara umum penggunanya yaitu mengerti dan memahami dalam mengoperasikan komputer serta dapat menggunakan *mouse*. Pada analisis pengguna ini juga akan mencakup analisis beberapa parameter terhadap calon pengguna dari aplikasi, diantaranya adalah:

1. *User Knowledge and Experience* (Pengetahuan dan Pengalaman Pengguna)

Adapun parameter pertama dalam analisis pengguna ini, yaitu pengetahuan dan pengalaman pengguna. Pengetahuan dan pengalaman merupakan salah satu faktor penting yang dapat dijadikan acuan pengguna dalam penggunaan aplikasi yang akan dibangun. Berikut ini klasifikasi *knowledge and experience* dari pengguna aplikasi yang dapat dilihat pada tabel III.14.

Tabel III.14 Analisis Klasifikasi *Knowledge and Experience*

Educational level	***Reading Level***	***Typing Skills***
Game yang akan dibangun ini ditujukan khusus untuk anak dengan usia 12-14 tahun.	*Game* yang akan dibangun ini ditujukan untuk pengguna yang sudah bisa membaca dengan baik.	*Game* yang akan dibangun ini tidak memerlukan kemampuan mengetik typing skills yang tinggi.
Computer Literacy	***Task Experience***	***System Experience***
Game yang akan dibangun ini ditujukan untuk pengguna yang sudah memiliki kemampuan dalam penggunaan komputer yang cukup memadai.	*Game* yang akan dibangun ini ditujukan untuk pengguna yang sudah pernah atau paham dan terbisaa dalam penggunaan komputer.	*Game* yang akan dibangun ini ditujukan untuk pengguna yang memiliki pengalaman dalam mengoperasikan sistem komputer.

Application Experience	***Native Language***	***Use Of Other System***
Game yang akan dibangun ini ditujukan untuk pengguna yang memiliki pengalaman dalam penggunaan aplikasi komputer.	*Game* yang akan dibangun ini menggunakan bahasa Indonesia.	Pengguna *game* yang akan dibangun ini tidak harus paham cara penggunaan aplikasi lain.

2. *User Physical Characteristic* (Karakter Fisik Pengguna)

Keadaan fisik seseorang mungkin akan berpengaruh pada penggunaan aplikasi *game* ini. Ada hal-hal yang harus diperhatikan juga terhadap pengguna dari karakteristik fisiknya untuk dapat menggunakan aplikasi ini, yaitu seperti yang ditunjukkan pada tabel III.15**.**

Tabel III.15 Analisis *Users Physical Characteristic*

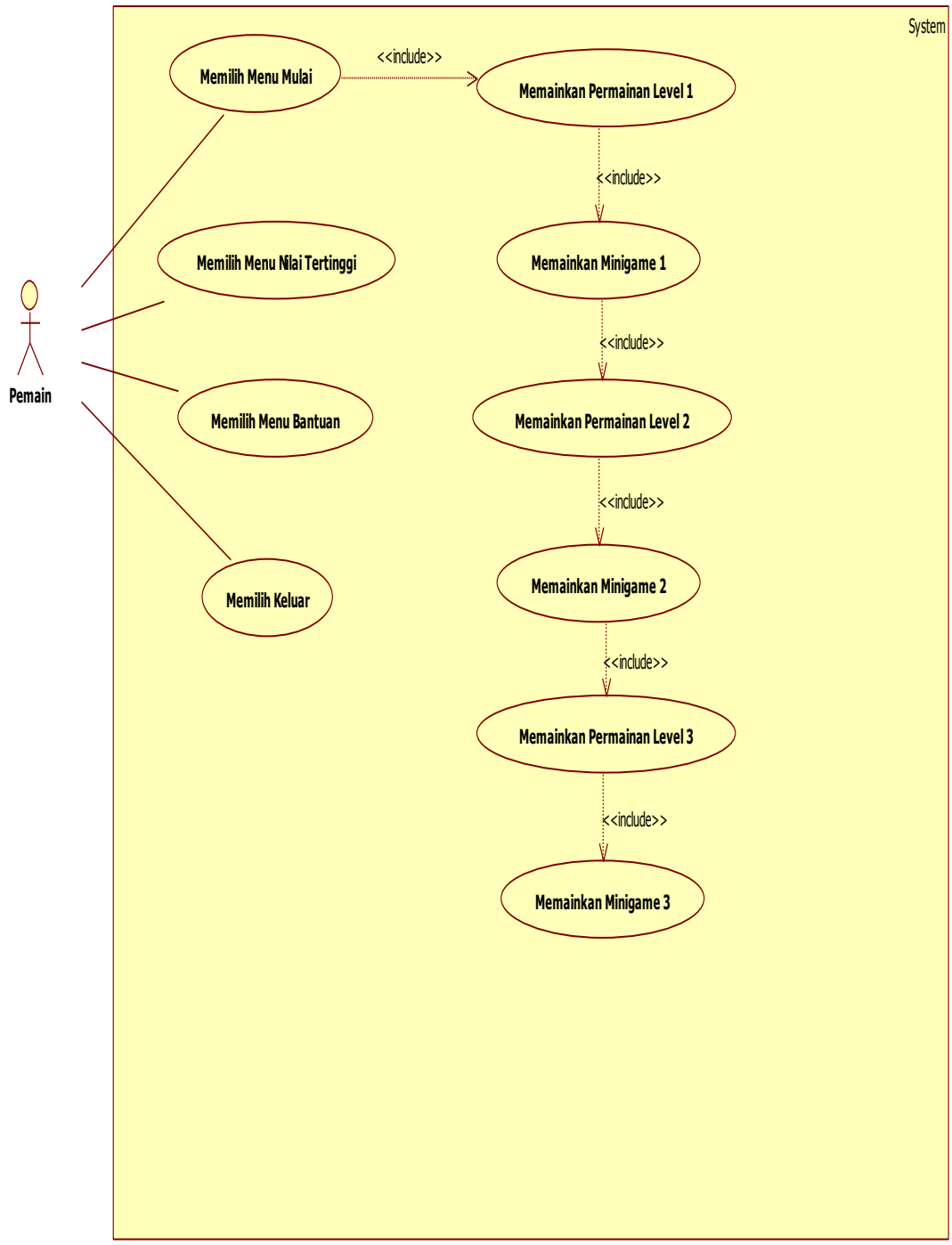
Age	12-14 tahun
Gender	Tidak ada batasan gender
Handedness	Tangan kanan dan kiri
Color Blind	*User* yang tidak bisa membedakan warna yang satu dengan yang lainnya (buta warna) masih mampu menggunakan aplikasi ini, karena tidak ada indicator warna-warna khusus yang membedakan antara fungsional yang satu dengan fungsional yang lainnya. Akan tetapi penggunaannya tidak akan optimal karena dalam *game* ini terdapat banyak sekali perbedaan warna yang menunjang interaksi dan ketertarikan dalam permainan.

III.6 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. Pendetakan pemodelan analisis yang digunakan untuk membangun aplikasi *game* edukasi ini adalah pemodelan berorientasi objek. Perangkat lunak ini dimodelkan menggunakan UML (*Unified Modeling Language). Tools* yang digunakan adalah *use case diagram, activity diagram, sequence diagram,* dan *class diagram*.

III.6.1 Use Case Diagram

Use case diagram game edukasi lindungi kebersihan laut memiliki 1 aktor dan 10 *use case* dimana keempat *use case* ini mewakili setiap fungsionalitas utama pada *game* edukasi lindungi kebersihan laut. *Use case diagram* untuk *game* edukasi lindungi kebersihan laut ini dapat dilihat pada gambar III.11**.**



Gambar III.11 ***Use Case Diagram***

III.6.1.1 Definisi Aktor

Berikut ini deskripsi pendefinisian aktor pada *game* edukasi lindungi kebersihan laut:

Tabel III.16 Definisi Aktor

Aktor	Deskripsi
Pemain	Orang yang memerankan tokoh tertentu.

III.6.1.2 Definisi Use case

Berikut ini deskripsi pendefinisian aktor pada *game* edukasi lindungi kebersihan laut:

Tabel III.17 Definisi *Use Case*

No	Nama Use case	Deskripsi
1.	Memilih Menu Mulai	merupakan fungsionalitas untuk memulai permainan.
2.	Memilih Menu Nilai Tertinggi	merupakan fungsionalitas untuk menampilkan informasi nilai / *point* yang didapatkan pemain
3.	Memainkan Permaian *Level* 1	Merupakan fungsionalitas melakukan permainan *level* 1
4.	Memainkan *Minigame* 1	Merupakan fungsionalitas melakukan permainan *minigame* 1
5.	Memainkan Permaian *Level* 2	Merupakan fungsionalitas melakukan permainan *level* 2
6.	Memainkan *Minigame* 2	Merupakan fungsionalitas melakukan permainan *minigame* 2
7	Memainkan Permaian *Level* 3	Merupakan fungsionalitas melakukan permainan *level* 3
8.	Memainkan *Minigame* 3	Merupakan fungsionalitas melakukan permainan *minigame*3
9.	Memilih Menu Bantuan	merupakan fungsionalitas untuk memberikan informasi tentang aturan-aturan dan cara bermain
10.	Memilih Menu Keluar	merupakan fungsionalitas untuk keluar dari aplikasi

III.6.1.3 Skenario Use case

Skenario *use case* berfungsi untuk menjelaskan lebih detail mengenai modul-modul yang terdapat dalam perangkat lunak. Berikut ini adalah skenario *use case* yang terdapat pada *game* edukasi lindungi kebersihan laut:

Tabel III.18 Skenario *Use Case* Memilih Menu Mulai

Nama Use case	:	Memilih Menu Mulai
Prekondisi	:	Pemain telah melihat perkenalan permainan dan berada di menu utama
Skenario		
Aksi Aktor		**Aksi *System***
1. Pemain memilih menu mulai		
		2. Menampilkan materi
		3. Menampilkan permainan *level* 1
Alternatif		
Jika pemain tidak ingin melihat materi yang ada di permainan maka, pemain dapat menekan tombol lewat untuk masuk ke permainan *level* 1.		

Tabel III.19 Skenario *Use Case* Memainkan Permainan *Level* 1

Nama Use case	:	Memainkan Permainan *Level* 1
Prekondisi	:	Pemain telah memilih menu mulai dan melihat materi permainan
Skenario		
Aksi Aktor		**Aksi *System***
		1. Menampilkan permainan *level* 1
		2. Memanggil algoritma fisher yates
		3. Menampilkan sampah
4. Menggerakkan *mouse*		
		5. Menggerakkan *player*
6. Mengumpulkan sampah		
		7. Memberikan *point*
		8. Menampilkan nyawa
		9. Menampilkan nilai
10.Menyelesaikan permainan *level* 1		
		11.Menampilkan *minigame* 1

Tabel III.20 Skenario *Use Case* Memainkan *Minigame* 1

Nama Use case	:	Memainkan *Minigame* 1
Prekondisi	:	Pemain telah menyelesaikan permainan *level* 1
Skenario		
Aksi Aktor		**Aksi *System***
		1. Menampilkan *minigame* 1
2. Menggerakkan *mouse*		
		3. Menggerakkan *player*
4. Mengambil sampah		
5. Menyimpan sampah pada tempatnya		
		6. Memberikan *point*
		7. Menampilkan nyawa
		8. Menampilkan nilai
9. Menyelesaikan permainan *minigame1*		
		10.Menampilkan permainan *level* 2

Tabel III.21 Skenario *Use Case* Memainkan Permainan *Level* 2

Nama Use case	:	Memainkan Permainan *Level* 2
Prekondisi	:	Pemain telah menyelesaikan *level* 1 dan *minigame* 1
Skenario		
Aksi Aktor		**Aksi *System***
		1. Menampilkan permainan *level* 1
		2. Memanggil algoritma fisher yates
		3. Menampilkan sampah
4. Menggerakkan *mouse*		
		5. Menggerakkan *player*
6. Mengumpulkan sampah		
		7. Memberikan *point*
		8. Menampilkan nyawa
		9. Menampilkan nilai
10. Menyelesaikan permainan *level* 2		
		11. Menampilkan *minigame* 2
		12. Menampilkan permainan *level 2*
Alternatif		
a. Jika *player* terkena ubur-ubur maka nilai akan dikurangi 10 *point.* b. Jika *player* terkena hiu maka nyawa *player* akan berkurang 1. c. Jika pemain kalah (nyawa *player* = 0) *system* akan menampilkan halaman *input* nama pemain untuk menyimpan *score* di permainan dan menampilkan nilai tertinggi.		

Tabel III.22 Skenario *Use Case* Memainkan *Minigame* 2

Nama Use case	:	Memainkan *Minigame* 2
Prekondisi	:	Pemain telah menyelesaikan permainan *level* 2
Skenario		
Aksi Aktor		**Aksi *System***
		1. Menampilkan *minigame* 2
2. Membuat garis *vertical, horizontal,* atau *diagonal*		
		3. Memanggil BHT_TicTacToe_AI_plugin.js
		4. Menampilkan garis yang dibuat
		5. Membuat garis vertical, horizontal, atau diagonal
		6. Menampilkan garis yang dibuat
		7. Memberikan *point*
		8. Menampilkan nyawa
		9. Menampilkan nilai
10. Menyelesaikan permainan *minigame*2		
		11. Menampilkan permainan *level* 3
Alternatif		
Jika pemain tidak behasil membuat garis *vertical, horizontal,* ataupun *diagonal* maka: a. permainan dikatakan seri jika, lawan (AI) juga tidak bisa membuat garis *vertical,*		

horizontal, atau *diagonal* namun hanya mendapatkan 50 *point* dan bisa masuk ke *level 3.* b. permainan dikatakan kalah jika, lawan (AI) bisa membuat garis *vertical, horizontal,* atau *diagonal* namun hanya mendapatkan 10 *point* dan bisa masuk ke *level 3* .

Tabel III.23 Skenario *Use Case* Memainkan Permainan *Level* 3

Nama Use case	:	Memainkan Permainan *Level* 3
Prekondisi	:	Pemain telah menyelesaikan *level* 2 dan *minigame* 2

Skenario	
Aksi Aktor	**Aksi *System***
	1. Menampilkan permainan *level* 3
	2. Memanggil algoritma fisher yates
	3. Menampilkan sampah
4. Menggerakkan *mouse*	
	5. Menggerakkan *player*
6. Mengumpulkan sampah	
	7. Memberikan *point*
	8. Menampilkan nyawa
	9. Menampilkan nilai
10. Menyelesaikan permainan *level* 3	
	11. Menampilkan *minigame* 3
	12. Menampilkan halaman input pemain

Alternatif
c. Jika *player* terkena ubur-ubur maka nilai akan dikurangi 10 *point.* d. Jika *player* terkena hiu maka nyawa *player* akan berkurang 1. e. Jika pemain kalah (nyawa *player* = 0) *system* akan menampilkan halaman *input* nama pemain untuk menyimpan *score* di permainan dan akan menampilkan nilai tertinggi.

Tabel III.24 Skenario *Use Case* Memainkan *Minigame* 3

Nama Use case	:	Memainkan *Minigame* 3
Prekondisi	:	Pemain telah menyelesaikan permainan *level* 3

Skenario	
Aksi Aktor	**Aksi *System***
	1. Menampilkan *minigame* 1
2. Menggerakkan *mouse*	
	3. Menggerakkan *player*
4. Mengambil sampah	
5. Menyimpan sampah sesuai jenisnya	
	6. Memberikan *point*
7. Menyelesaikan permainan *minigame* 3	
	7. Menampilkan nyawa
	8. Menampilkan nilai
	9. Menampilkan halaman *input* nama pemain

10.Memasukan nama pemain	
	11.Menampilkan nama pemain dan nilai

Tabel III.25 Skenario *Use case* Memilih Menu Nilai Tertinggi

Nama Use case	:	Memilih Menu Nilai Tertinggi
Prekondisi	:	Aplikasi sudah pernah dimainkan
Skenario		
Aksi Aktor		**Aksi *System***
1. Memilih menu nilai tertinggi		
		1. Membaca data nilai tertinggi.
		2. Menampilkan informasi nilai tertinggi
Alternatif		
Jika setelah melihat informasi nilai tertinggi pemain ingin menuju ke menu utama maka, pemain dapat menekan tombol kembali.		

Tabel III.26 Skenario *Use case* Memilih Menu Bantuan

Nama Use case	:	Memilih Menu Bantuan
Prekondisi	:	Pemain ada di menu utama dan akan memulai permainan.
Skenario		
Aksi Aktor		**Aksi *System***
1. Memilih menu bantuan		
		2. Menampilkan menu bantuan
Alternatif		
Jika setelah melihat informasi bantuan pemain ingin menuju ke menu utama maka, pemain dapat menekan tombol kembali.		

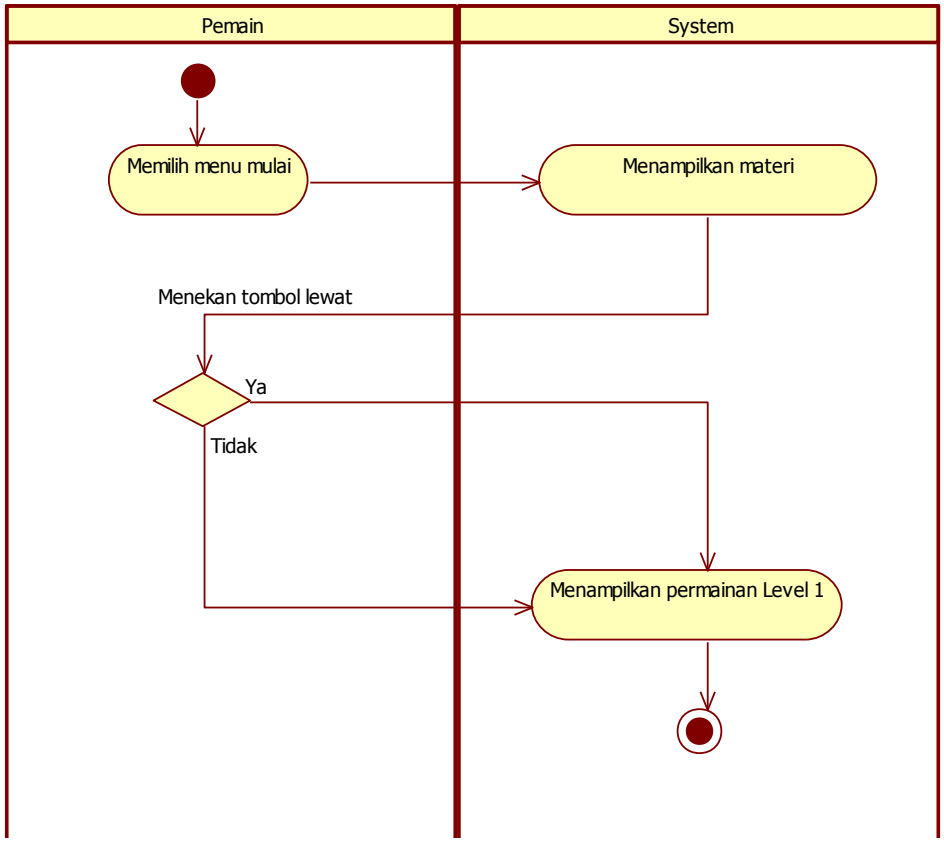
Tabel III.27 Skenario *Use case* Memilih Menu Keluar

Nama Use case	:	Keluar
Prekondisi	:	Pemain telah melakukan permainan atau berada di menu utama
Skenario		
Aksi Aktor		**Aksi *System***
1. Memilih menu keluar		
		2. Menampilkan pesan keluar
3. Menekan salah satu pesan keluar		
4.		

Alternatif
a. Jika pemain memilih pesan keluar Tidak maka, pemain akan kembali ke menu utama. b. Jika pemain memilih pesan keluar Ya maka, pemain akan keluar dari aplikasi,

III.6.2 Activity Diagram

Activity diagram menggambarkan berbagai alir aktivitas dalam perangkat lunak yang sedang dirancang, bagaimana masing-masing alir berawal, *decision* yang mungkin terjadi dan bagaimana mereka berakhir. *Activity diagram* juga dapat menggambarkan proses paralel yang mungkin terjadi pada beberapa eksekusi. Berikut *activity diagram* yang ada pada *game* edukasi lindungi kebersihan laut ini:

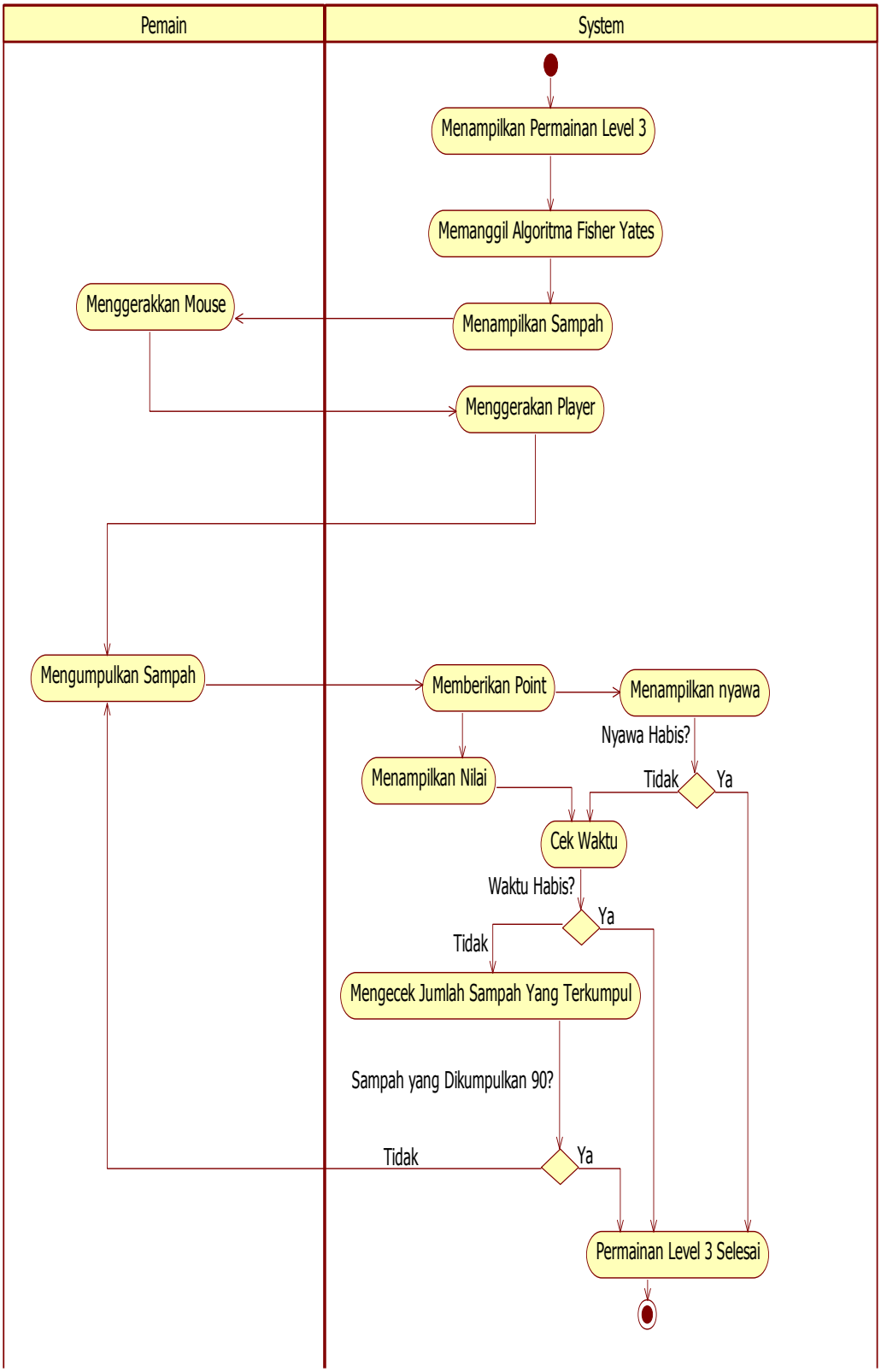


Gambar III.12 *Activity Diagram* Mulai

Keterangan :

1. Pemain memilih menu mulai untuk memulai permainan.
2. *System* menampilkan materi.

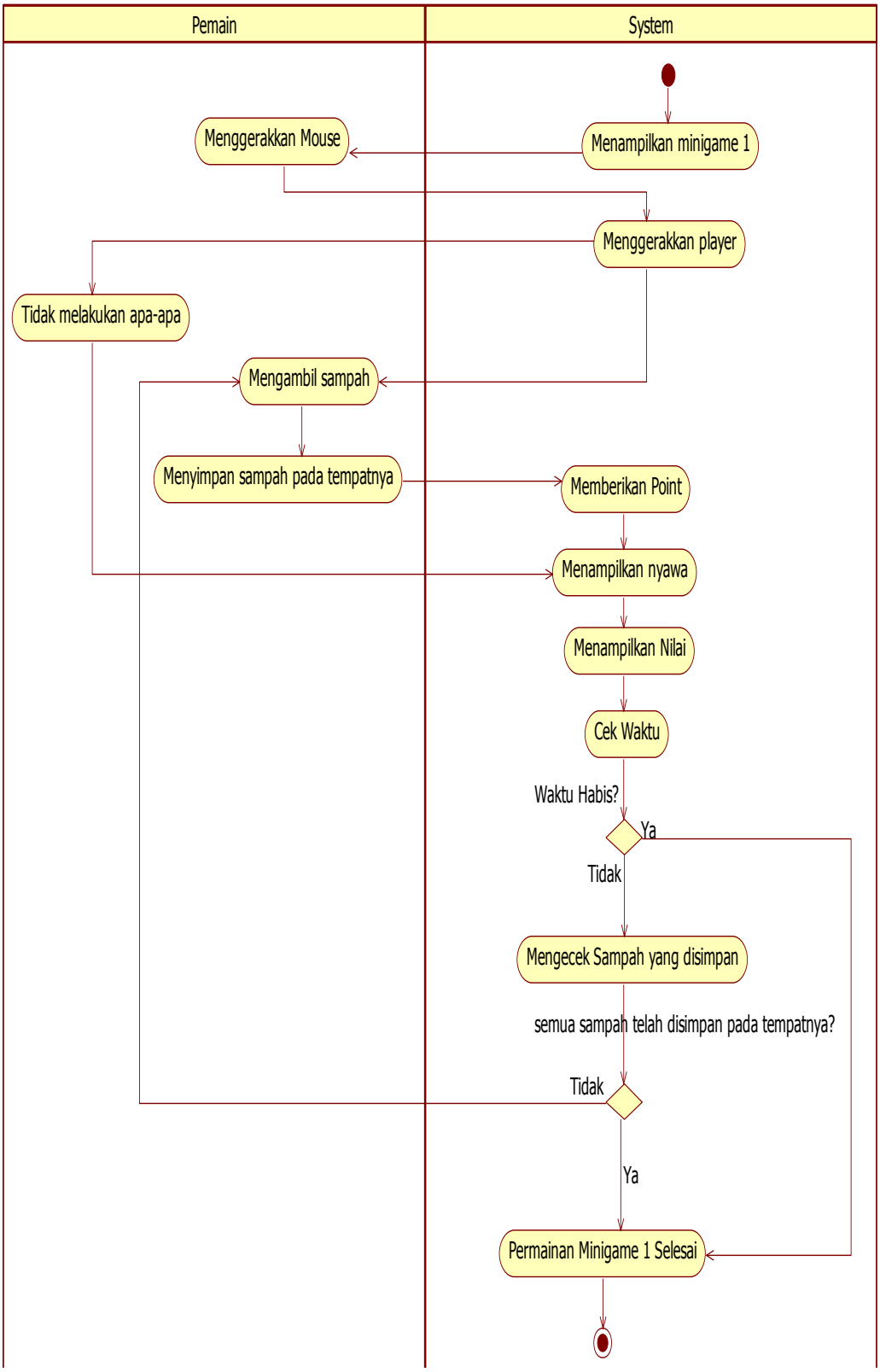
3. Jika pemain:
    a. Menekan tombol lewat pada saat menampilkan materi maka pemain tidak perlu melihat semua materi hingga selesai dan *system* akan menampilkan permainan *level* 1
    b. Tidak menekan tombol lewat pada saat menampilkan materi maka pemain harus melihat semua materi yang ditampilkan dan setelah beberapa waktu *system* akan menampilkan permainan *level* 1



Gambar III.13 ***Activity Diagram Level*** **1**

Keterangan:

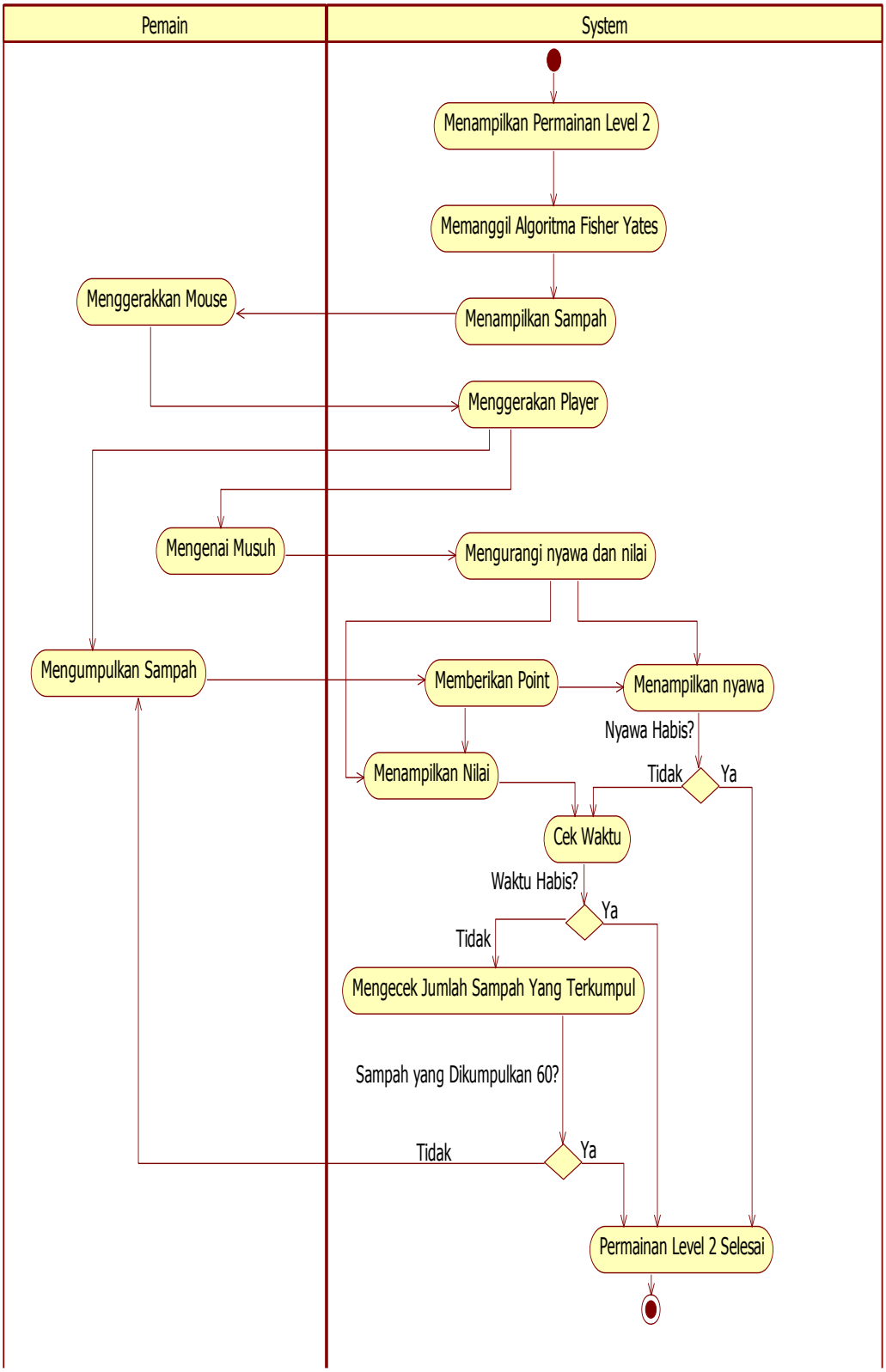
1. *System* menampilkan permainan *level* 1
2. *System* memanggil algoritma fisher Yates.
3. *System* akan menampilkan sampah.
4. Pemain menggerakkan *mouse*
5. *System* akan menggerakkan *player*
6. Pemain akan mengumpulkan sampah dan *system* akan memberikan point.
7. *System* akan menampilkan nilai dan menampilkan nyawa.
8. *System* akan mengecek apakah nyawa habis atau tidak, jika nyawa habis maka permainan selesai, jika tidak *system* akan mengecek waktu.
9. *System* mengecek apakah waktu habis atau tidak, Jika waktu permainan habis maka permainan *level* 1 selesai, jika waktu belum habis maka pemain akan mengumpulkan sampah kembali.
10. *System* akan mengecek jumlah sampah yang terkumpul, jika sampah yang dikumpulkan telah 40 buah sampah maka permainan *level* 1 selesai, jika tidak pemain harus mengumpulkan sampah kembali.



Gambar III.14 *Activity Diagram Minigame* 1

Keterangan:

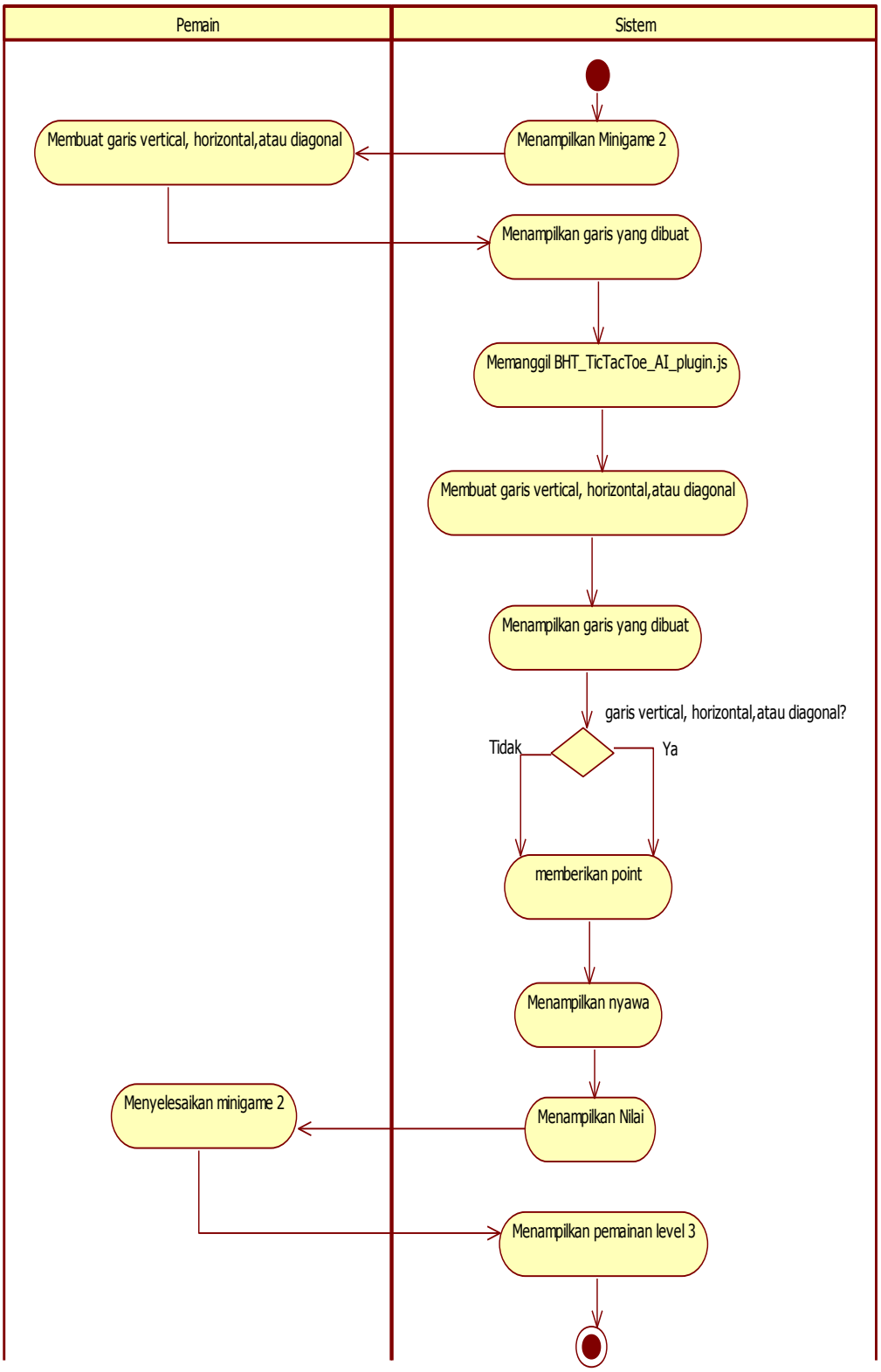
1. *System* menampilkan *minigame* 1
2. Pemain akan menggerakkan *mouse*
3. *System* menggerakkan *player*
4. Pemain akan mengambil sampah dan menyimpan sampah pada tempatnya
5. *System* akan memberikan *point*
6. *System* akan menampilkan nyawa
7. *System* menampilkan nilai
8. *System* akan mengecek waktu, apakah waktu habis atau tidak. Jika waktu habis maka permainan *minigame* 1 selesai, jika waktu belum habis maka system akan mengecek sampah yang telah disimpan.
9. *System* akan mengecek apakah semua sampah telah berada pada tempatnya, jika ya maka pemain menyelesaikan permainan *minigame1,* jika tidak maka pemain akan mengambil sampah dan menyimpan sampah pada tempatnya.
10. Jika pemain tidak melakukan apa-apa maka *system* akan mengecek apakah waktu habis jika habis maka *minigame* 1 selesai, jika tidak *system* akan menampilkan nyawa dan nilai.
11. *System* akan menampilkan permainan *level* 2



Gambar III.15 *Activity Diagram Level* 2

Keterangan:

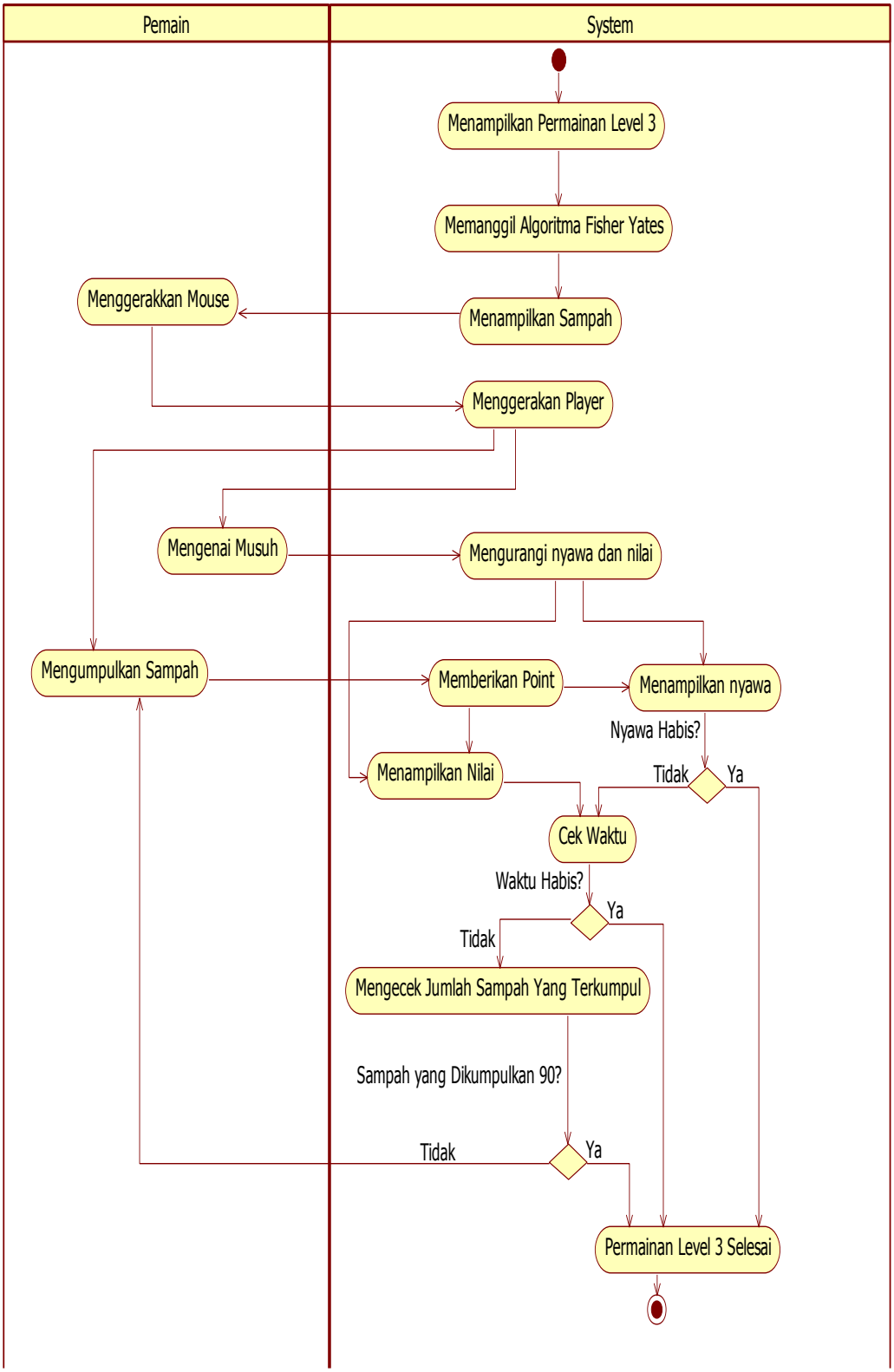
1. *System* menampilkan permainan *level* 2.
2. *System* memanggil algoritma fisher Yates.
3. *System* akan menampilkan sampah.
4. Pemain menggerakkan *mouse*
5. *System* akan menggerakkan *player*
6. Pemain akan mengumpulkan sampah dan *system* akan memberikan point.
7. *System* akan menampilkan nilai dan menampilkan nyawa.
8. *System* akan mengecek apakah nyawa habis atau tidak, jika nyawa habis maka permainan selesai, jika tidak *system* akan mengecek waktu.
9. *System* mengecek apakah waktu habis atau tidak, Jika waktu permainan habis maka permainan *level* 2 selesai, jika waktu belum habis maka pemain akan mengumpulkan sampah kembali.
10. *System* akan mengecek jumlah sampah yang terkumpul, jika sampah yang dikumpulkan telah 60 buah sampah maka permainan *level* 2 selesai, jika tidak pemain harus mengumpulkan sampah kembali.
11. Jika pemain mengenai musuh maka nilai dan nyawa pemain akan berkurang.



Gambar III.16 ***Activity Diagram Minigame 2***

Keterangan:

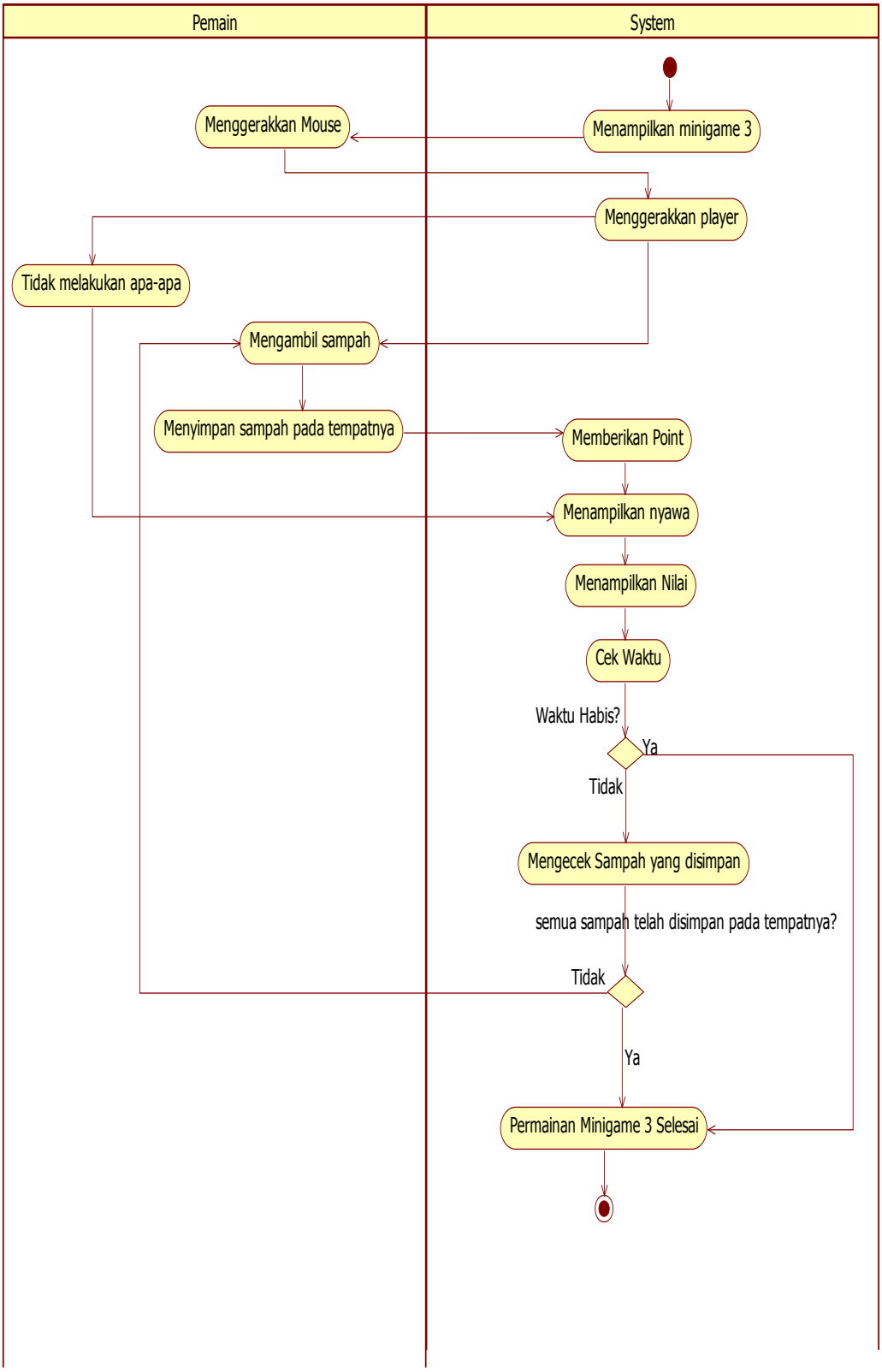
1. *System* menampilkan *minigame* 2
2. Pemain membuat garis *vertical, horizontal,* atau *diagonal*
3. *System* akan menampilkan garis yang dibuat
4. *System* akan membuat garis *vertical, horizontal,* atau *diagonal* dengan memanggil BHT_TicTacToe_AI_plugin.js.
5. *System* akan menampilkan garis yang dibuat
6. *System* memberikan *point*
7. *System* menampilkan nyawa
8. *System* menampilkan nilai
9. Pemain menyelesaikan permainan *minigame*2
10. *System* menampilkan permainan *level* 3



Gambar III.17 ***Activity Diagram Level 3***

Keterangan:

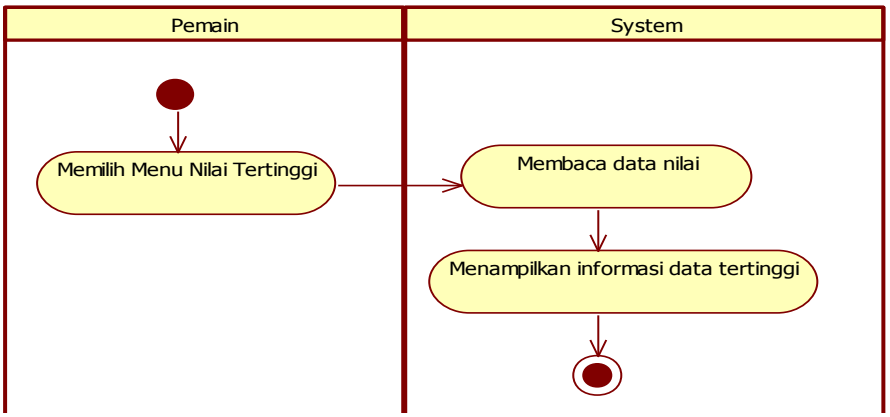
1. *System* menampilkan permainan *level* 3.
2. *System* memanggil algoritma fisher Yates.
3. *System* akan menampilkan sampah.
4. Pemain menggerakkan *mouse*
5. *System* akan menggerakkan *player*
6. Pemain akan mengumpulkan sampah dan *system* akan memberikan point.
7. *System* akan menampilkan nilai dan menampilkan nyawa.
8. *System* akan mengecek apakah nyawa habis atau tidak, jika nyawa habis maka permainan selesai, jika tidak *system* akan mengecek waktu.
9. *System* mengecek apakah waktu habis atau tidak, Jika waktu permainan habis maka permainan *level* 3 selesai, jika waktu belum habis maka pemain akan mengumpulkan sampah kembali.
10. *System* akan mengecek jumlah sampah yang terkumpul, jika sampah yang dikumpulkan telah 90 buah sampah maka permainan *level* 3 selesai, jika tidak pemain harus mengumpulkan sampah kembali.
11. Jika pemain mengenai musuh maka nilai dan nyawa pemain akan berkurang.



Gambar III.18 ***Activity Diagram Minigame 3***

Keterangan:

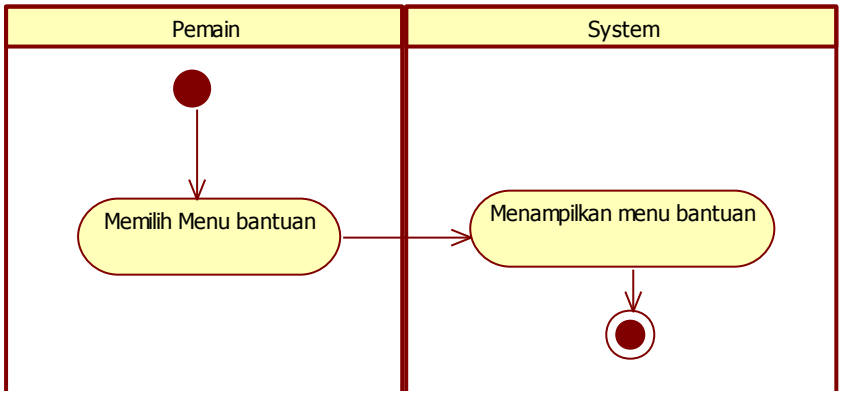
1. *System* menampilkan *minigame* 3.
2. Pemain akan menggerakkan *mouse*
3. *System* menggerakkan *player*
4. Pemain akan mengambil sampah dan menyimpan sampah pada tempatnya
5. *System* akan memberikan *point*
6. *System* akan menampilkan nyawa
7. *System* menampilkan nilai
8. *System* akan mengecek waktu, apakah waktu habis atau tidak. Jika waktu habis maka permainan *minigame* 3 selesai, jika waktu belum habis maka system akan mengecek sampah yang telah disimpan.
9. *System* akan mengecek apakah semua sampah telah berada pada tempatnya, jika ya maka pemain menyelesaikan permainan *minigame3,* jika tidak maka pemain akan mengambil sampah dan menyimpan sampah pada tempatnya.
10. Jika pemain tidak melakukan apa-apa maka *system* akan mengecek apakah waktu habis jika habis maka *minigame* 3 selesai, jika tidak *system* akan menampilkan nyawa dan nilai.



Gambar III.19 *Activity Diagram* Nilai Tertinggi

Keterangan :

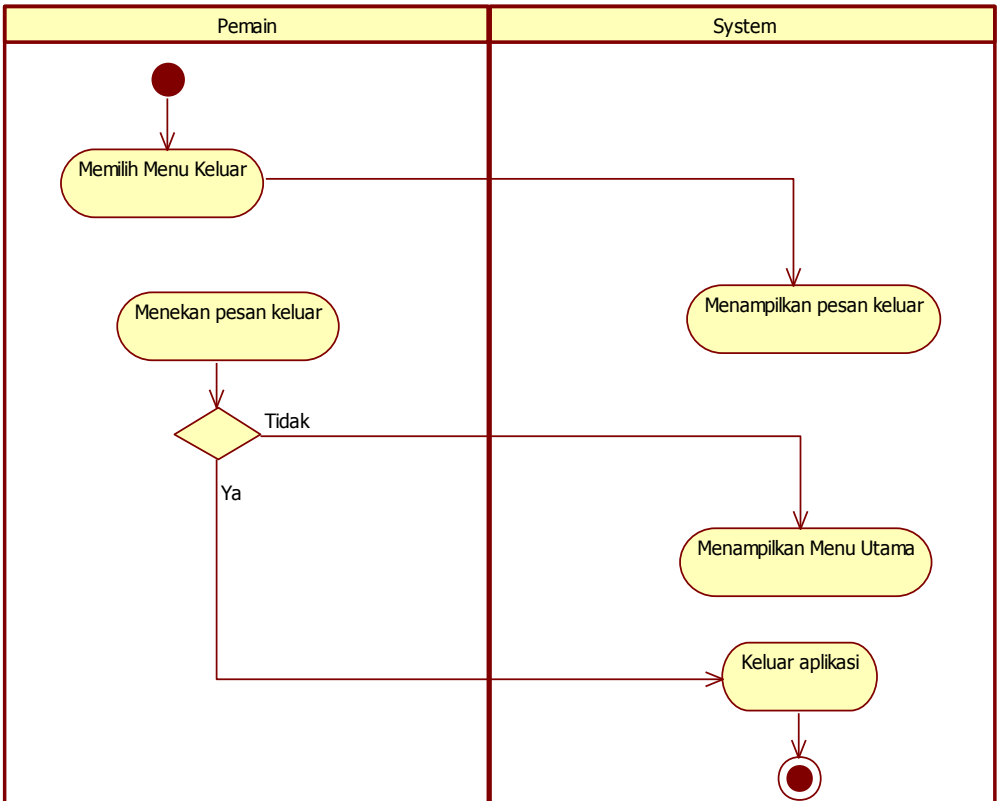
1. Pemain memilih menu nilai tertinggi untuk melihat informasi nilai.
2. *System* menerima *request* nilai tertinggi dan membaca data nilai.
3. *System* menampilkan data nilai.



Gambar III.20 *Activity Diagram* Bantuan

Keterangan :

1. Pemain memilih menu bantuan untuk melihat informasi aturan dan cara bermain.
2. *System* menampilkan menu bantuan.



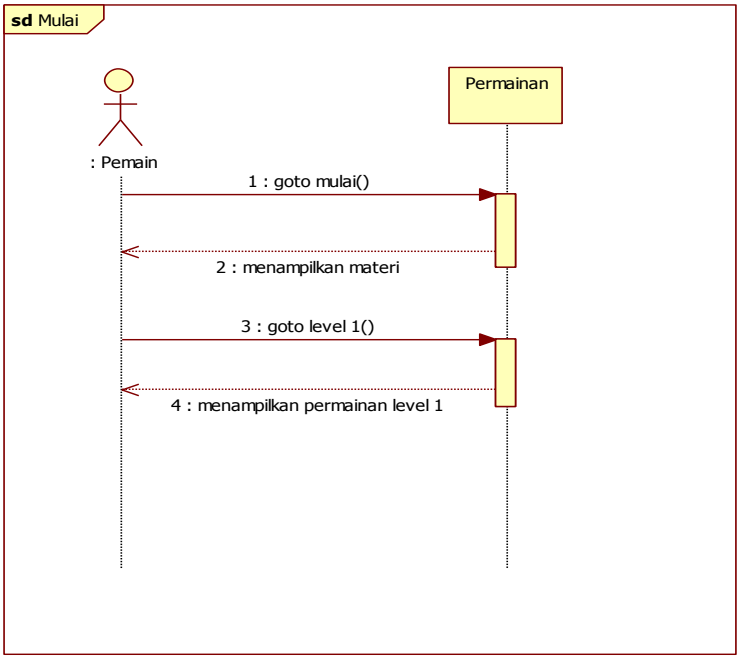
Gambar III.21 *Activity Diagram* Keluar

Keterangan :

1. Memilih keluar
2. Menampilkan pesan keluar
3. Jika pemain memilih pesan keluar
    a. Jika pesan keluar yang dipilih adalah Tidak maka, pemain akan kembali ke menu utama.
    b. Jika pesan keluar yang dipilih adalah Ya maka, pemain akan keluar dari aplikasi.

III.6.3 Sequence Diagram

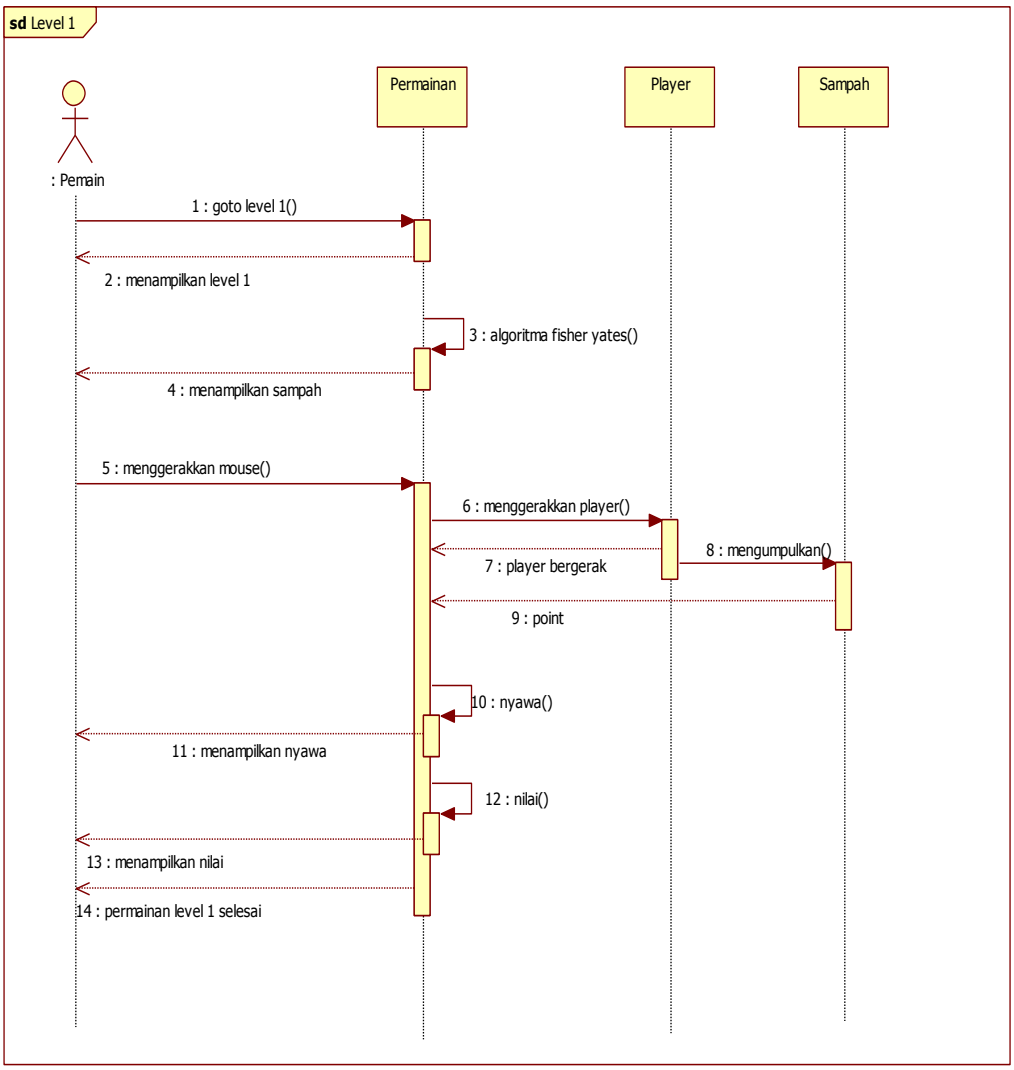
Sequence diagram menggambarkan interaksi antar objek di dalam dan di sekitar perangkat lunak (termasuk pengguna, display, dan sebagainya) berupa *message* yang digambarkan terhadap waktu. *Sequence diagram* terdiri atas dimensi vertikal (waktu) dan dimensi horizontal (objek-objek yang terkait). *Sequence diagram* biasa digunakan untuk menggambarkan skenario atau rangkaian langkah-langkah yang dilakukan sebagai respons dari sebuah event untuk menghasilkan output tertentu. Diawali dari apa yang men-*trigger* aktivitas tersebut, proses dan perubahan apa saja yang terjadi secara internal dan output apa yang dihasilkan. *Sequence diagram* pada *game* edukasi lindungi kebersihan laut ini terbagi menjadi beberapa bagian, yaitu:



Gambar III.22 *Sequence Diagram* Mulai

Keterangan :

1. Pemain memilih menekan tombol mulai dan akan memanggil *method* goto mulai() untuk masuk ke permainan.
2. Selanjutnya akan menampilkan materi
3. Pemain akan memanggil *method* goto *level* 1() dan akan menampilkan permainan *level* 1.



Gambar III.23 *Sequence Diagram Level* 1

Keterangan :

1 Pemain menuju ke *level* 1 dengan memanggil *method* goto *level* 1() dan akan menampilkan permainan *level* 1.

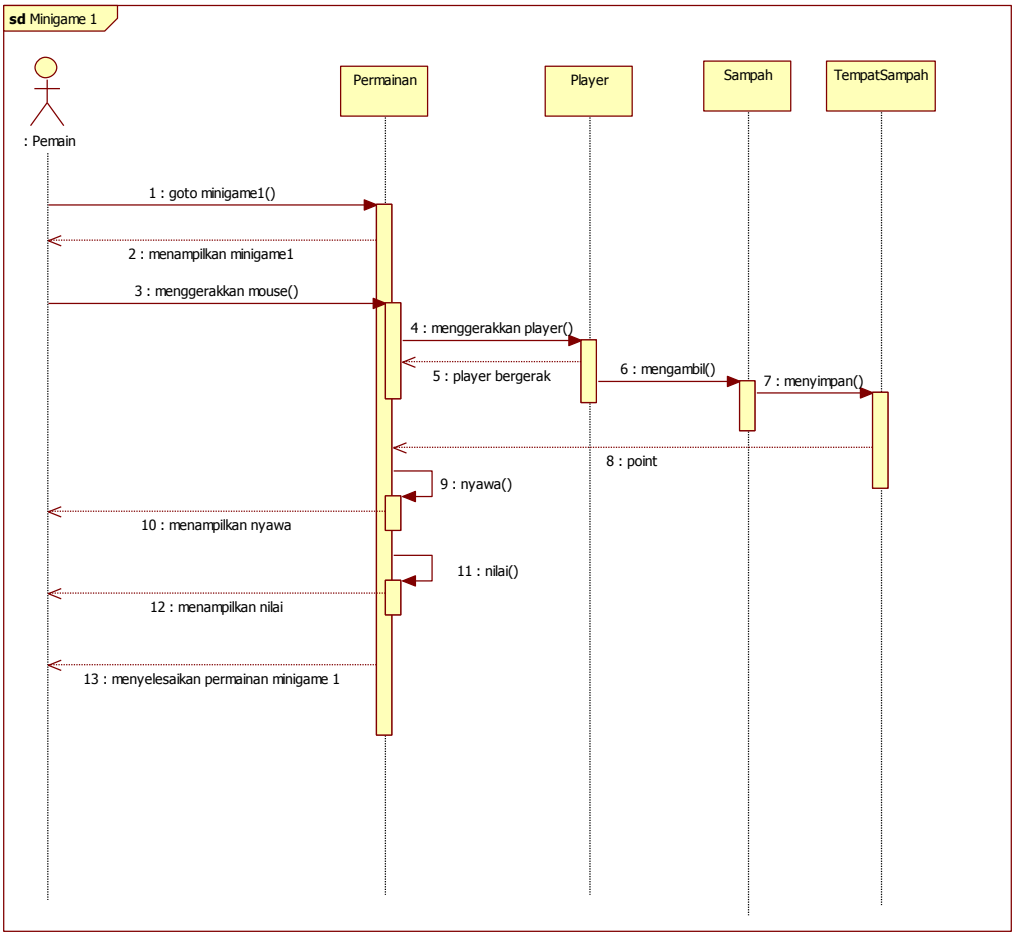
2 Permainan akan memanggil algoritma fisher Yates dan akan menampilkan sampah.

3 Pemain menggerakkan mouse sehingga di permainan akan menggerakkan *player* dengan memanggil *method* menggerakkan *player*() dan akan menampilkan *player* bergerak.

4 Pemain akan mengumpulkan sampah dengan memanggil *method* mengumpulkan() untuk mendapatkan *point*.

5 Di permainan akan menampilkan nyawa dengan memanggil *method* nyawa() dan menampilkan nilai dengan memanggil *method* nilai().

6 Pemain menyelesaikan permainan *level* 1.



Gambar III.24 *Sequence Diagram Minigame* 1

Keterangan:

1 Pemain menuju ke *minigame* 1 dengan memanggil *method* goto *minigame* 1() dan akan menampilkan *minigame* 1.

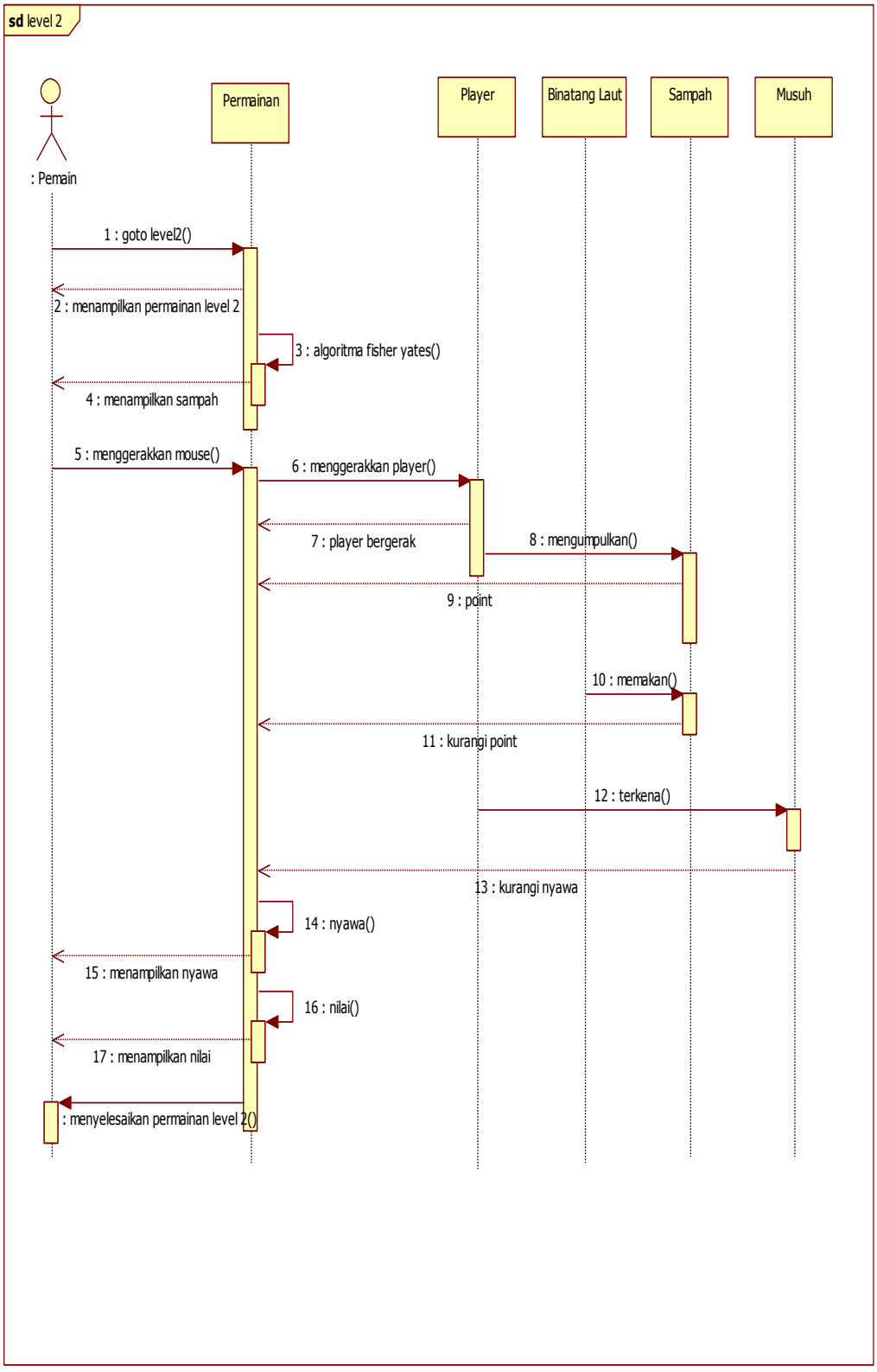
2 Pemain menggerakkan *mouse* dengan memanggil *method* menggerakkan *mouse*().

3 Permainan akan menggerakkan *player* dengan memanggil *method* menggerakkan *player*() dan akan menampilkan pergerakkan *player*.

4 Pemain akan mengambil sampah dengan memanggil *method* mengambil() dan menyimpan sampah tersebut dengan memanggil *method* menyimpan() dan mendapatkan *point*.

5 Di permainan akan menampilkan nyawa dengan memanggil *method* nyawa() dan menampilkan nilai dengan *method* nilai().

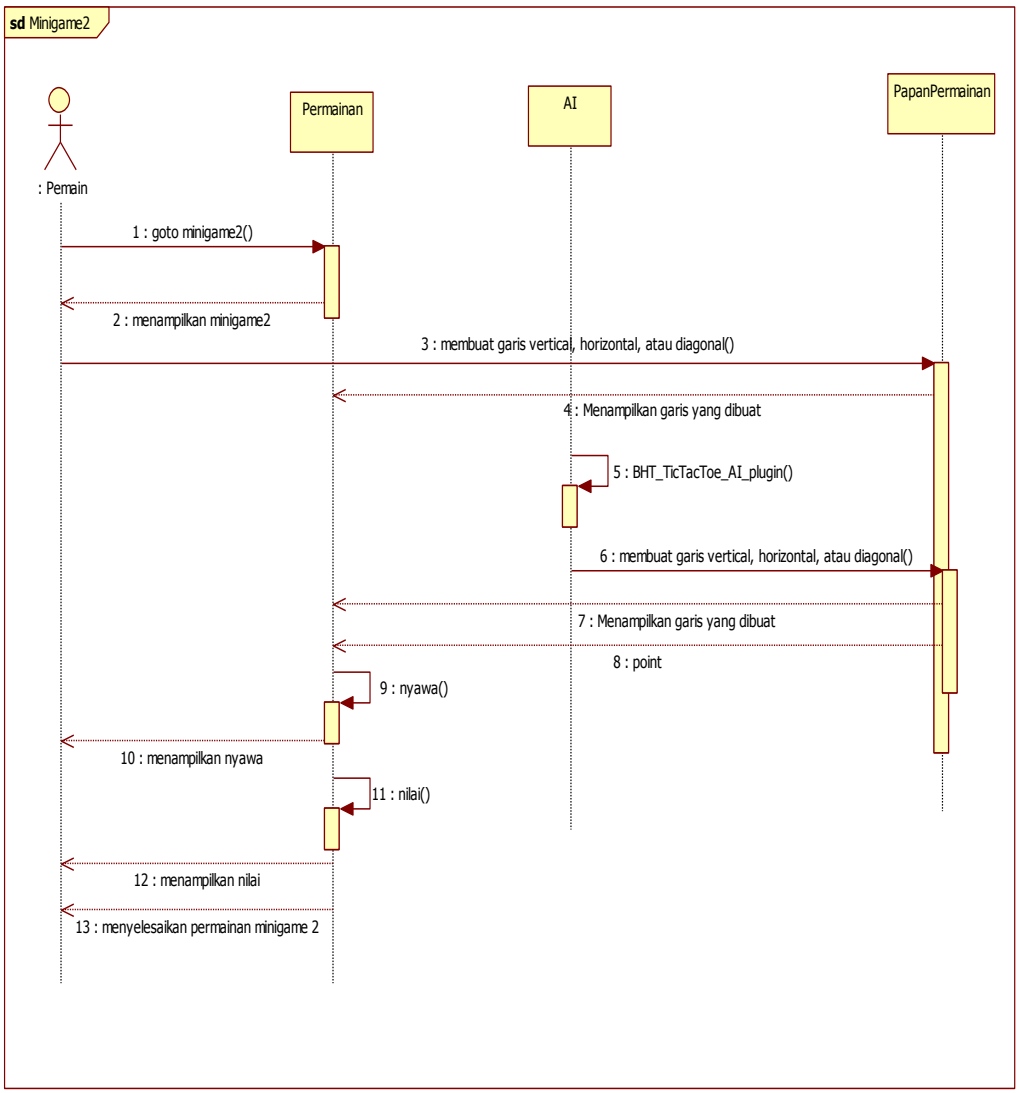
6 Pemain menyelesaikan permainan *minigame*1.



Gambar III.25 *Sequence Diagram Level* 2

Keterangan :

1 Pemain menuju ke *level* 2 dengan memanggil *method* goto *level* 2() dan akan menampilkan *level* 2.
2 Pemain menggerakkan *mouse* dengan *method* menggerakkan *mouse*() dan menggerakkan *player* dengan memanggil *method* menggerakkan *player*().
3 Pemain akan mengumpulkan sampah dengan memanggil *method* mengumpulkan() untuk mendapatkan *point*.
4 Binatang laut akan memakan sampah dengan memanggil *method* memakan() dan dapat mengurangi *point* pemain.
5 Musuh akan terkena dengan *player* dengan memanggil *method* terkena() dan akan mengurangi nyawa pemain.
6 Di permainan akan menampilkan nyawa dengan *method* nyawa() dan menampilkan nilai dengan *method* nilai().
7 Pemain menyelesaikan permainan *level* 2.



Gambar III.26 *Sequence Diagram Minigame* 2

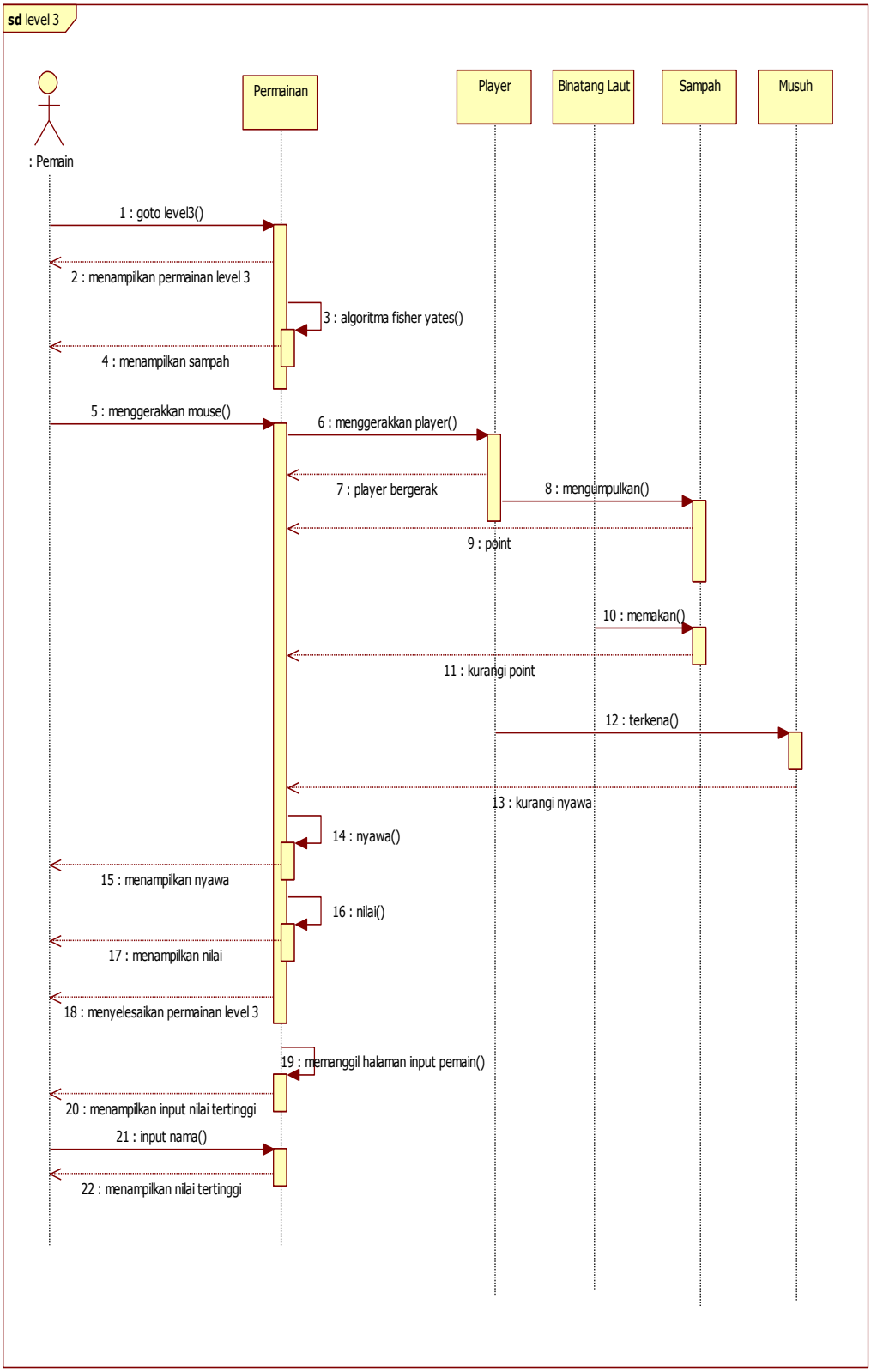
Keterangan:

1 Pemain menuju ke *minigame* 2 dengan memanggil *method* goto *minigame* 2() dan akan menampilkan *minigame* 2.

2 Pemain membuat garis *vertical*, *horizontal*, atau *diagonal* pada papan permainan dengan memanggil *method* membuat garis *vertical*, *horizontal*, atau *diagonal* () dan akan mendapatkan *point*.

3 AI akan memanggil BHT_TicTacToe_AI_plugin.

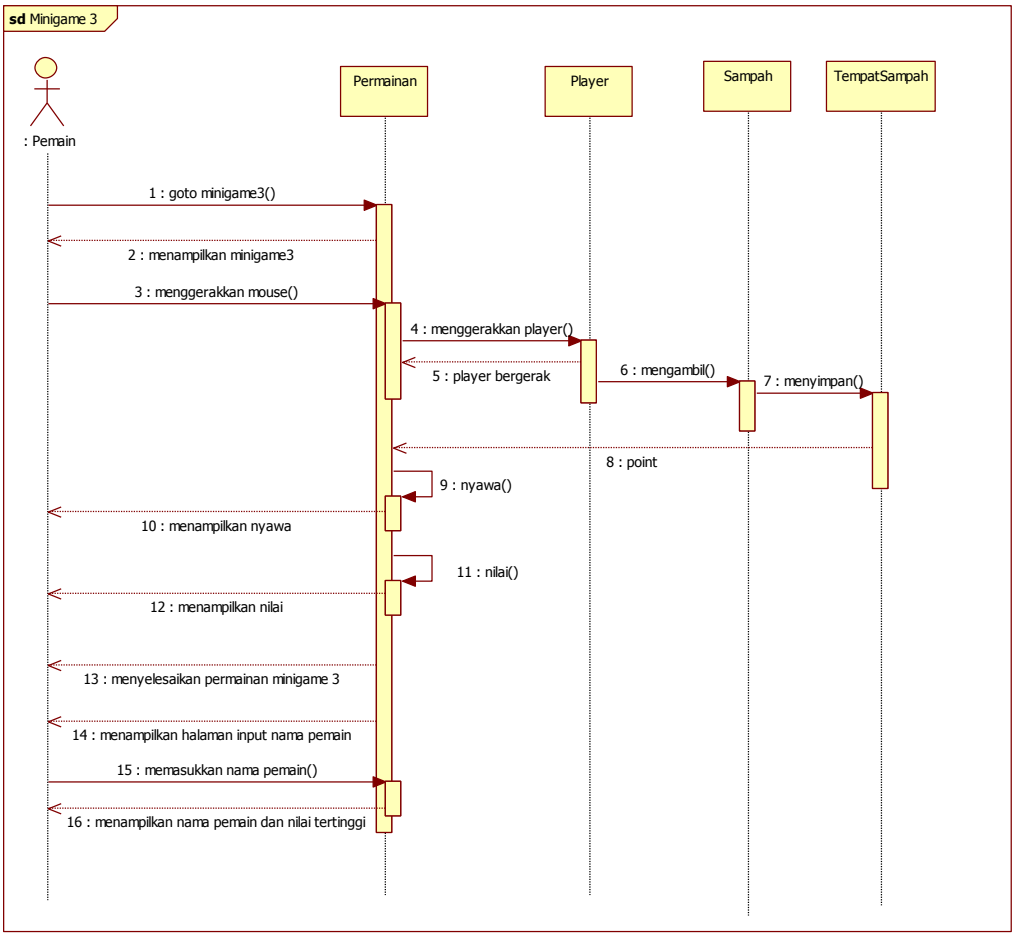
4 Kemudian giliran AI untuk membuat garis *vertical*, *horizontal*, atau *diagonal* dengan menggunakan *method* membuat garis *vertical*, *horizontal*, atau *diagonal()* dan menampilkan garis yang dibuat.
5 Menampilkan point.
6 Permainan akan menampilkan nyawa dan nilai dengan memanggil method nyawa() dan nilai().
7 Pemain menyelesaikan permainan *minigame* 2.



Gambar III.27 *Sequence Diagram Level* 3

Keterangan :

1 Pemain menuju ke *level* 3 dengan memanggil *method* goto *level* 3() dan akan menampilkan *level* 3.
2 Pemain menggerakkan *mouse* dengan *method* menggerakkan *mouse*() dan menggerakkan *player* dengan memanggil *method* menggerakkan *player*().
3 Pemain akan mengumpulkan sampah dengan memanggil *method* mengumpulkan() untuk mendapatkan *point*.
4 Binatang laut akan memakan sampah dengan memanggil *method* memakan() dan dapat mengurangi *point* pemain.
5 Musuh akan terkena dengan *player* dengan memanggil *method* terkena() dan akan mengurangi nyawa pemain.
6 Di permainan akan menampilkan nyawa dengan *method* nyawa() dan menampilkan nilai dengan *method* nilai().
7 Pemain menyelesaikan permainan *level* 3.
8 Permainan akan memanggil halaman input nilai tertinggi dan menampilkan halaman nilai tertinggi.
9 Pemain memasukkan nama pemain dan permainan akan menampilkan nilai tertinggi



Gambar III.28 *Sequence Diagram Minigame* 3

Keterangan:

1 Pemain menuju ke *minigame* 3 dengan memanggil *method* goto *minigame* 3() dan akan menampilkan *minigame* 3.

2 Pemain menggerakkan *mouse* dengan memanggil *method* menggerakkan *mouse*().

3 Permainan akan menggerakkan *player* dengan memanggil *method* menggerakkan *player*().

4 Pemain akan mengambil sampah dengan memanggil *method* mengambil() dan menyimpan sampah tersebut dengan memanggil *method* menyimpan() dan mendapatkan *point*.

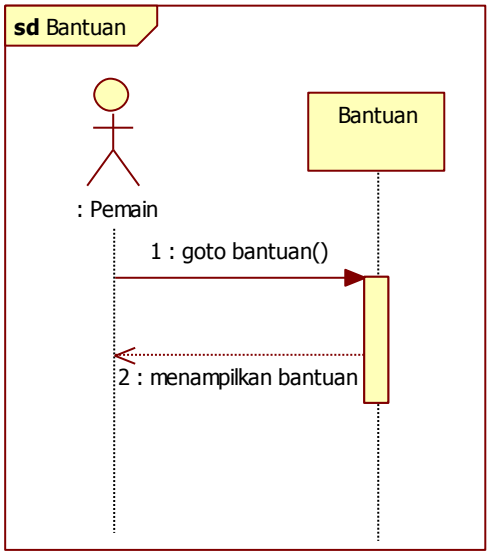
5 Di permainan akan menampilkan nyawa dengan memanggil *method* nyawa() dan menampilkan nilai dengan *method* nilai().

6 Pemain menyelesaikan permainan *minigame* 3.

7 Permainan akan menampilkan halaman input nama pemain.

8 Pemain memasukkan nama pemain.

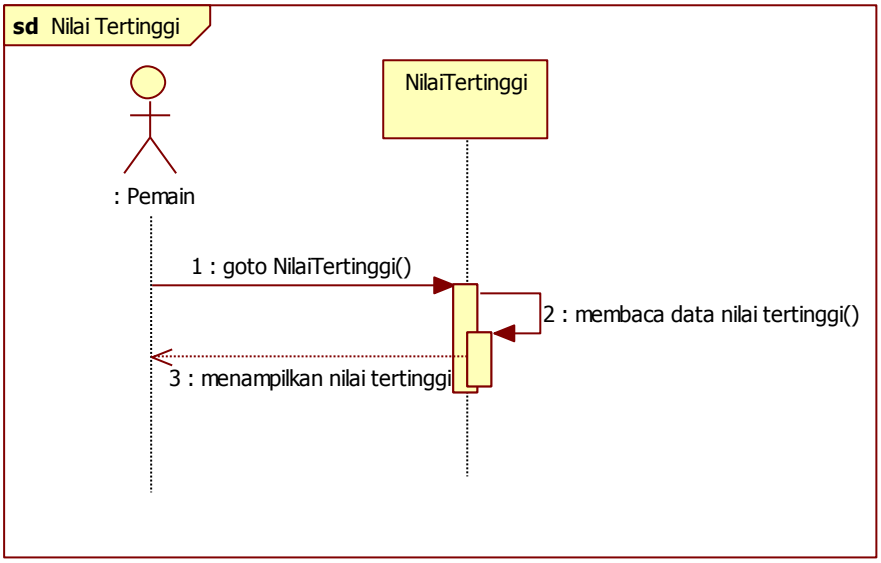
9 Permainan menampilkan nama pemain dan nilai tertinggi.



Gambar III.29 *Sequence Diagram* Bantuan

Keterangan :

1. Pemain memanggil *method* bantuan untuk menampilkan bantuan.



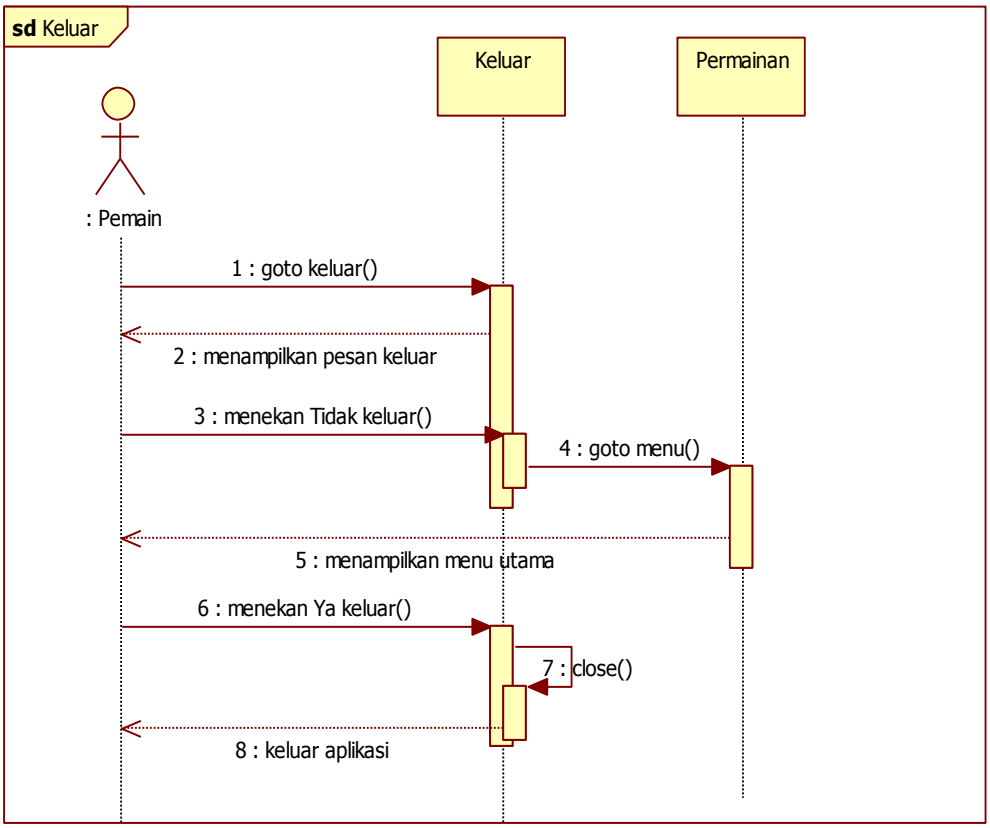
Gambar III.30 *Sequence Diagram* Nilai Tertinggi

Keterangan:

1 Pemain menuju ke nilai tertinggi dengan memanggil *method* goto nilai tertinggi().

2 Akan membaca data nilai dengan memanggil *method* membaca data nilai tertinggi().

3 Menampilkan nilai tertinggi.



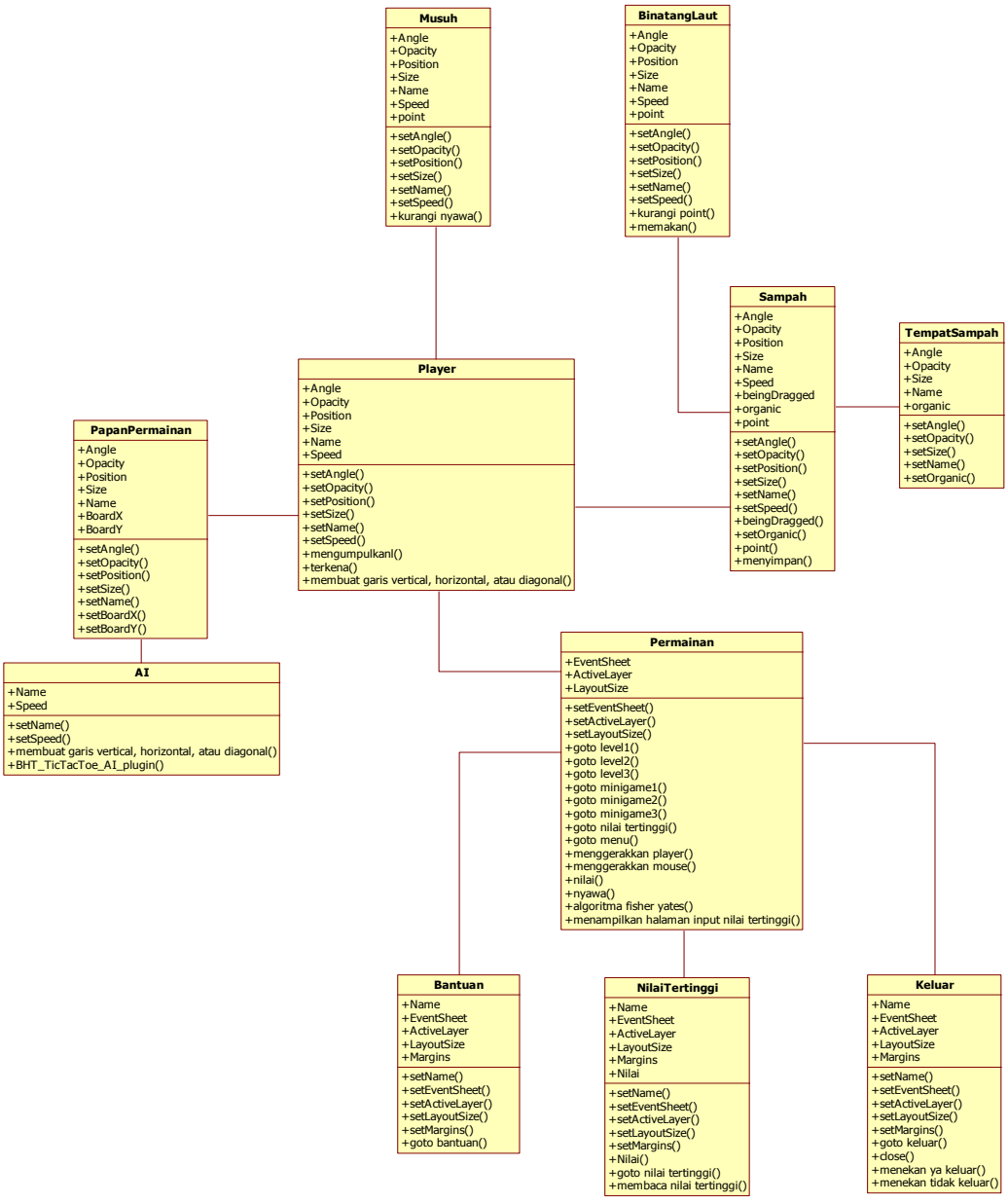
Gambar III.31 *Sequence Diagram* Keluar

Keterangan:

1. Pemain memilih keluar dan menampilkan pesan keluar.
2. Jika pemain menekan tombol tidak pada pesan keluar maka akan kembali ke menu utama dengan memanggil *method* goto menu() dan menampilkan menu utama.
3. Jika pemain menekan tombol ya pada pesan keluar maka akan keluar aplikasi dengan memanggil *method* close().

III.6.4 Class Diagram

Class adalah sebuah spesifikasi yang jika diinstansiasi akan menghasilkan sebuah objek dan merupakan inti dari pengembangan dan desain berorientasi objek. *Class* menggambarkan keadaan (atribut/properti) suatu perangkat lunak, sekaligus menawarkan layanan untuk memanipulasi keadaan tersebut (metoda/fungsi). Berikut *class diagram* yang terbentuk pada *game* edukasi lindungi kebersihan laut:



Gambar III.32 *Class Diagram*

Keterangan:

1. *Class* Permainan untuk menampilkan aplikasi *game.*
2. *Class* Sampah untuk merepresentasikan sampah.
3. *Class* BinatangLaut untuk merepresentasikan ikan dan kura-kura.
4. *Class Player* untuk merepresentasikan *player*.
5. *Class* Musuh untuk merepresentasikan musuh (hiu dan ubur-ubur).
6. *Class* Tempat sampah untuk merepresentasikan tempat penyimpanan sampah.
7. *Class* Papan permainan untuk mempresentasikan area permainan *minigame* 2.
8. *Class* AI untuk merepresentasikan komputer dalam permainan *minigame*2.
9. *Class* NilaiTertinggi untuk menampilkan informasi nilai.
10. *Class* Bantuan untuk menampilkan informasi cara main dan aturan.
11. *Class* Keluar untuk keluar dari aplikasi.

III.7 Perancangan Sistem

Perancangan adalah suatu bagian dari metodologi pengembangan suatu perangkat lunak yang dilakukan setelah tahapan untuk memberikan gambaran secara terperinci.

Berdasarkan uraian diatas perancangan sistem merupakan tahapan dari siklus pengembangan sistem yang didefinisikan dari kebutuhan-kebutuhan fungsional dan persiapan untuk rancang bangun implementasi yang menggambarkan bagaimana suatu sistem dibentuk, yang dapat berupa penggambaran, perancangan, dan pembuatan sketsa atau pengaturan dari beberapa elemen yang terpisah kedalam satu kesatuan yang utuh dan berfungsi, juga menyangkut konfigurasi dari komponen-komponen perangkat keras dan perangkat lunak dari suatu sistem.

III.7.1 Perancangan Komponen Permainan

Perancangan komponen permainan meliputi deskripsi karakter deskripsi *item,* dan *storyboard* yang didalamnya menjelaskan karakter beserta *item-item* apa yang dapat diperoleh pemain. Sedangkan *storyboard* sebagai gambar perancangan pembuatan *game* edukasi lindungi kebersihan laut.

III.7.1.1 Deskripsi Karakter

Deskripsi karakter *game* merupakan penjelasan dari setiap karakter yang terdapat pada *game*. Berikut beberapa gambaran karakter yang terdapat pada *game* yang akan dibangun yaitu:

1. Karakter Utama

   Karakter utama dalam *game* ini bernama Toni. Karakter ini merupakan seorang anak umur 10 tahun.

2. Karakter Pendukung

   Karakter pendukung dalam *game* ini adalah kura-kura, ikan, lumba-lumba dan kuda laut. Kura-kura dalam *game* ini digambarkan dengan kura-kura yang berwarna hijau, ikan yang berwarna biru, lumba-lumba yang berwarna abu-abu serta kuda laut yang berwarna coklat.

3. Karakter Musuh

   Karakter musuh dalam *game* ini adalah ubur-ubur dan hiu. Ubur-ubur memiliki tubuh yang berwarna merah muda serta hiu yang berwarna hitam.

Deskripsi karakter dapat dilihat pada table III.28**.**

Tabel III.28 Deskripsi Karakter

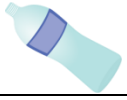
No	Nama Karakter	Deskripsi
1.	 **Toni**	Toni adalah anak berumur 10 tahun dan merupakan karakter utama pada *game* ini. Toni memiliki kemampuan bergerak ke kiri, ke kanan, ke atas dan ke bawah.
2.	 **Ikan**	Ikan merupakan karakter pendukung yang bisa bergerak ke kiri dan ke kanan.
3.	 **Kura-kura**	Kura-kura merupakan karakter yang harus diselamatkan oleh Toni. Kura-kura ini akan bergerak mendekati sampah-sampah yang ada di dalam laut dan memakan sampah-sampah tersebut.
4.	 **Ubur-ubur (musuh)**	Ubur ubur merupakan musuh dari Toni. Ubur-ubur bergerak ke kiri dan ke kanan. Ubur- ubur ini akan menyengat Toni, jika Toni bersentuhan dengan kulit ubur-ubur pergerakan Toni akan berkurang.

No	Nama Karakter	Deskripsi
5.	 **Hiu (musuh)**	Hiu merupakan musuh dari Toni. Hiu ini dapat mengancam keselamatan Toni dengan cara memakan Toni yang dapat mengurangi nyawa Toni.

III.7.1.2 Deskripsi Item

Deskripsi *item game* merupakan penjelasan dari setiap *item* yang harus dikumpulkan pada setiap *game*. Berikut deskripsi *item* yang terdapat pada *game* yang disajikan pada tabel III.29.

Tabel III.29 Deskripsi Item

No	Item	Deskripsi
1.		Sampah Organik
2.		Sampah Organik
3.		Sampah Organik
4.		Sampah Organik
5.		Sampah Organik
6.		Sampah Anorganik
7.		Sampah Anorganik
8.		Sampah Anorganik
9.		Sampah Anorganik
10.		Sampah Anorganik

III.7.1.3 Storyboard

Storyboard merupakan visualisasi ide dari aplikasi yang akan dibangun, sehingga dapat memberikan gambaran dari aplikasi yang akan dihasilkan. *Storyboard* dapat dikatakan juga *visual script* yang akan dijadikan *outline* dari sebuah proyek, ditampilkan *shot by shot* yang biasa disebut dengan istilah *scene*. *Storyboard* dapat menyampaikan ide cerita kepada orang lain dengan lebih mudah karena dengan *storyboard,* pengguna dapat mengimajinasikan khayalan yang sesuai dengan gambar-gambar yang ada sehingga dapat menghasilkan persepsi yang sama pada ide cerita yang dibuat.

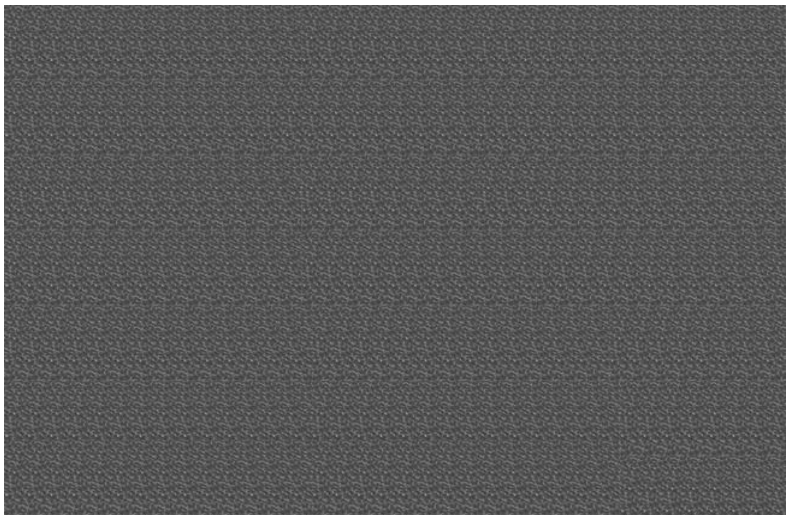
1. ***Storyboard Perkenalan Game***



Gambar III.33 *Storyboard* Perkenalan *Game*

Gambar III.33 merupakan bagian perkenalan dari *game*. Perkenalan ini menceritakan alasan mengapa Toni bertekad untuk menyelamatkan kehidupan laut dengan cara mengumpulkan sampah-sampah yang ada di laut.

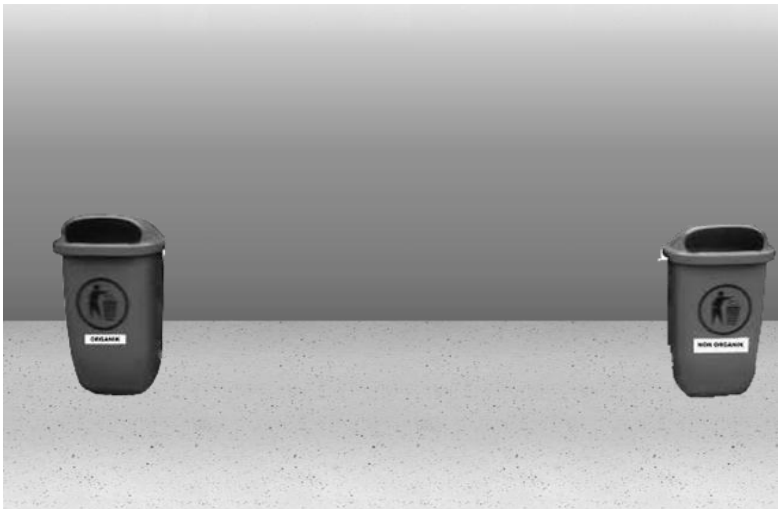
2. *Storyboard Level 1*



Gambar III.34 *Storyboard Level* 1

Gambar III.21 ini merupakan *storyboard level* 1, dimana pada *level* ini akan ada 40 sampah yang harus dikumpulkan. Jika berhasil menyelesaikan *level* ini akan masuk ke *minigame* menyimpan sampah pada tempatnya.

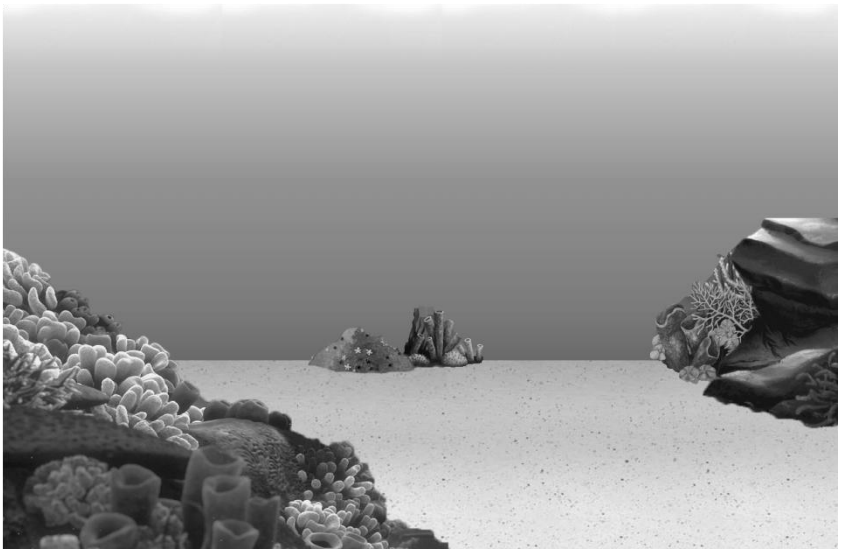
3. *Storyboard Minigame* Menyimpan Sampah Pada Tempatnya



Gambar III.35 *Storyboard Minigame* Menyimpan Sampah

Pada *minigame* ini akan muncul sampah-sampah yang telah dikumpulkan di *level* sebelumnya, tugas pemain adalah menyimpan sampah-sampah yang telah dikumpulkan tadi pada tempatnya sesuai jenis sampah.

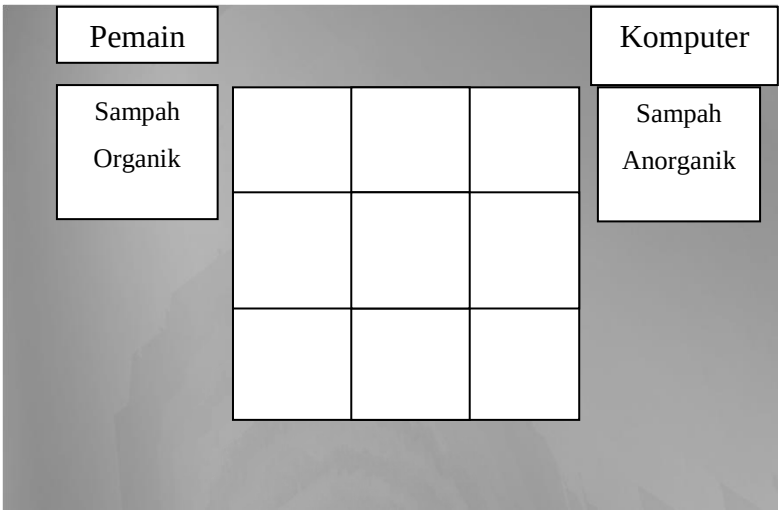
4. *Storyboard Level 2*



Gambar III.36 *Storyboard Level* 2

Gambar III.36 ini merupakan *storyboard level* 2, dimana pada *level* ini akan ada 60 sampah yang harus dikumpulkan. Jika berhasil menyelesaikan *level* ini akan masuk ke *minigame* cari pasangan gambar.

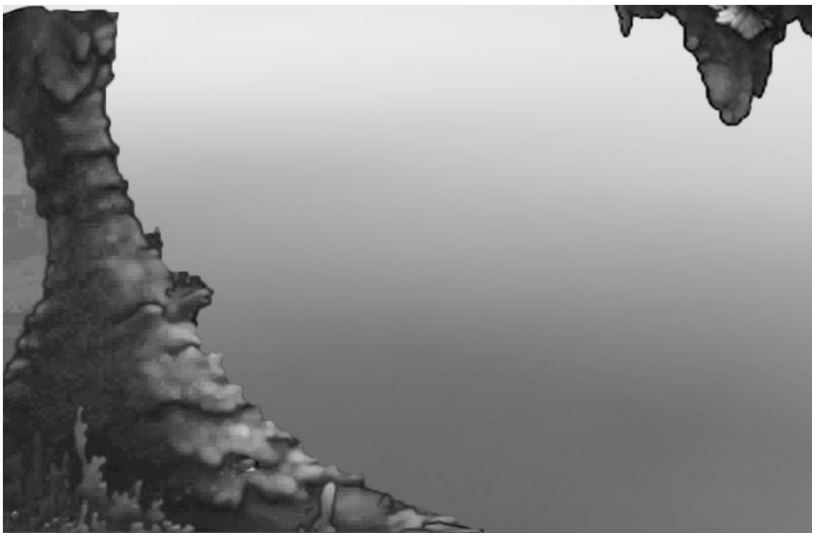
5. *Storyboard Minigame* Kalahkan AI



Gambar III.37 *Storyboard Minigame* Kalahkan AI

Pada *minigame* ini pemain diminta untuk meletakkan simbol yang dimainkannya dalam satu garis lurus baik secara vertikal, horisontal, maupun diagonal. Selain itu pemain juga dituntut agar mampu menghalangi komputer untuk melakukan hal yang serupa, yaitu meletakkan simbol yang dimainkan komputer dalam satu garis lurus. Pemain harus berhati-hati agar tidak lengah saat menyusun simbol yang dimainkannya, karena salain menghalangi jalan yang pemain mainkan, komputer juga tetap berusaha agar mampu meletakkan simbolnya dalam satu garis lurus baik secara vertikal, horisontal, maupun secara diagonal. Pemain akan disimbolkan dengan gambar organik dan komputer (musuh) disimbolkan dengan gambar anorgaink.

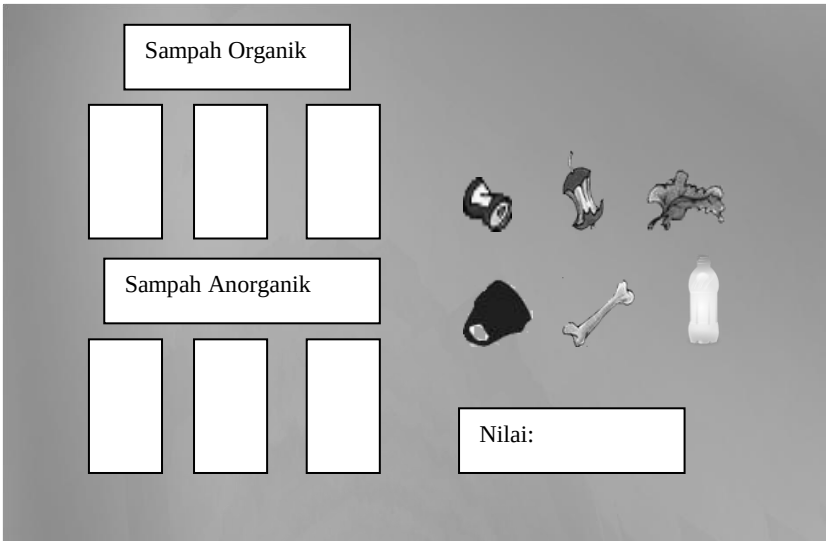
6. *Storyboard Level 3*



Gambar III.38 *Storyboard Level* 3

Gambar III.38 ini merupakan *storyboard level* 3, dimana pada *level* ini akan ada 90 sampah yang harus dikumpulkan. Jika berhasil menyelesaikan *level* ini akan masuk ke *minigame* pilih jenis sampah.

7. *Storyboard Minigame* Pilih Jenis Sampah

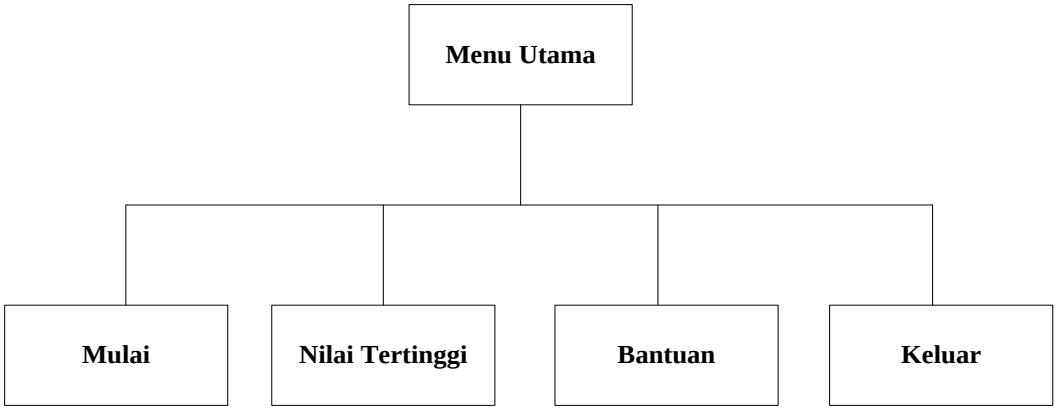


Gambar III.39 *Storyboard Minigame* Pilih Jenis Sampah

Pada *minigame* ini pemain harus men-*drag & drop* sampah-sampah sesuai jenisnya ke tempat yang telah ditentukan.

III.7.2 Perancangan Struktur Menu

Struktur menu adalah bentuk umum dari suatu rancangan program untuk memudahkan pemakai dalam menjalankan program komputer. Sehingga saat menjalankan program, pengguna tidak mengalami kesulitan dalam memilih menu-menu yang diinginkan. Berikut ini perancangan struktur menu pada *game* edukasi lindungi kebersihan laut yang di tunjukan pada gambar III.40.



Gambar III.40 Perancangan Struktur Menu Utama

III.7.3 Perancangan Antarmuka

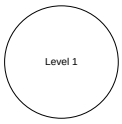
Antarmuka merupakan tampilan dari suatu perangkat lunak yang berperan sebagai media komunikasi yang digunakan sebagai sarana antara perangkat lunak dan pengguna. Perancangan ini merupakan sebuah penggambaran, perencanaan dan pengaturan dari beberapa elemen yang terpisah ke dalam satu kesatuan utuh. Perangkat lunak yang dibangun diharapkan menyediakan suatu tampilan antarmuka yang mudah dipahami dan digunakan oleh pengguna. Berikut ini adalah tampilan antarmuka yang digunakan pada *game* edukasi lindungi kebersihan laut:

T01 (Tampilan Perkenalan)	**Keterangan**
Tampilan Perkenalan Lewat	- Klik tombol “Lewat” atau menunggu selama 2 menit akan menuju ke halaman T02
Ukuran layar : *Width* 800 *Pixe*l (*Default*) Besar font beragam disesuaikan	

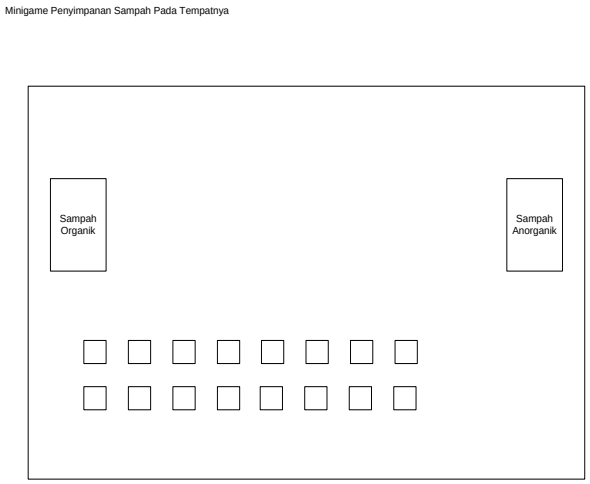
Gambar III.41 Tampilan Antarmuka Perkenalan

T02 (Menu Utama)	Keterangan
Game Edukasi **Lindungi Kebersihan Laut** **Mulai Bermain** **Nilai Tertinggi** **Bantuan** **Keluar**	- Klik “Mulai Bermain” akan menuju ke halaman T03 - Klik “Nilai Tertinggi” akan menuju ke halaman T07 - Klik “Bantuan” akan menuju ke halaman T08 - Klik “Keluar” akan menuju ke halaman T09
Ukuran layar : *Width* 800 *Pixe*l (*Default*) Besar font beragam disesuaikan	

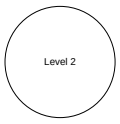
Gambar III.42 Tampilan Antarmuka Menu Utama

T03 (Level 1)	Keterangan
	- Setelah pemain menyelesaikan permainannya akan masuk ke T10
Ukuran layar : *Width* 800 *Pixe*l (*Default*) Besar font beragam disesuaikan	

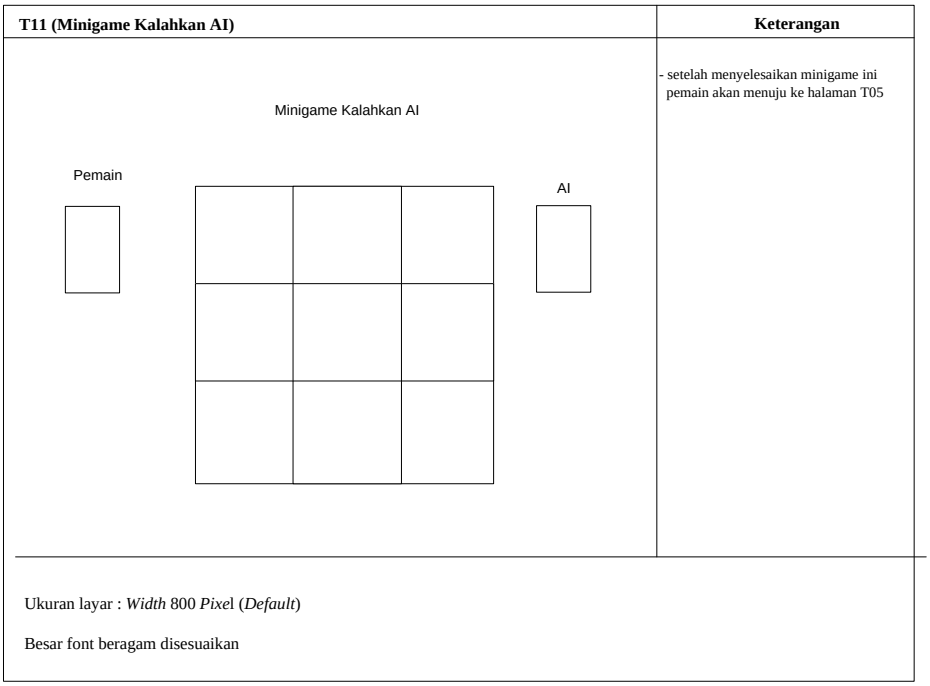
Gambar III.43 Tampilan Antarmuka *Level* 1

T10 (Minigame Penyimpanan Sampah Pada Tempatnya)	Keterangan
	- setelah menyelesaikan minigame ini pemain akan menuju ke halaman T04
Ukuran layar : *Width* 800 *Pixe*l (*Default*) Besar font beragam disesuaikan	

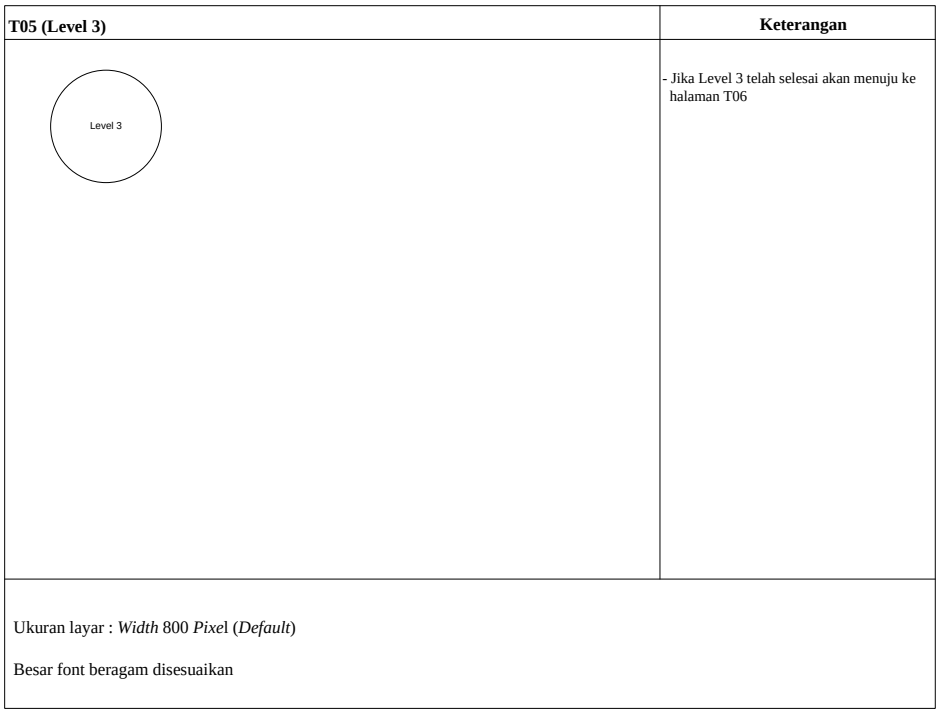
Gambar III.44 Tampilan Antarmuka *Minigame* Menyimpan Sampah

T04 (Level 2)	Keterangan
	- Setelah menyelesaikan level 2 akan masuk ke halaman T11
Ukuran layar : *Width* 800 *Pixe*l (*Default*) Besar font beragam disesuaikan	

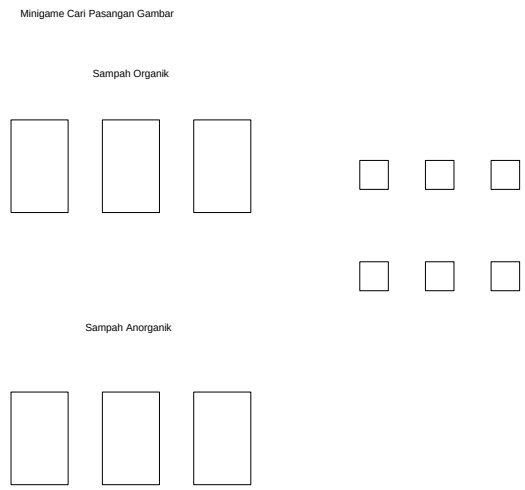
Gambar III.45 Tampilan Antarmuka *Level* 2



Gambar III.46 Tampilan Antarmuka *Minigame* Kalahkan AI



Gambar III.47 Tampilan Antarmuka *Level* 3

T12 (Minigame Pilih Jenis Sampah)	Keterangan
	- setelah menyelesaikan minigame ini berarti pemain telah menyelesaikan permainan ini dan akan menuju ke halaman T06.
Ukuran layar : *Width* 800 *Pixe*l (*Default*) Besar font beragam disesuaikan	

Gambar III.48 Tampilan Antarmuka *Minigame* Pilih Jenis Sampah

T06 (Permainan Selesai)	Keterangan
Selamat Anda Telah Menyelesaikan Permainan Score Anda …….. << Kembali Ke Menu Keluar >>	- Klik "Kembali Ke Menu" akan menuju ke halaman T02 - Klik "Keluar" akan menuju ke halaman T09
Ukuran layar : *Width* 800 *Pixe*l (*Default*) Besar font beragam disesuaikan	

Gambar III.49 Tampilan Antarmuka Permainan Selesai

T07 (Nilai Tertinggi)	Keterangan
Nilai Tertinggi 1. ……. 2. ……. 3. ……. 4. ……. 5. ……. …….. **Kembali**	- Klik "Kembali" akan menuju ke halaman T02
Ukuran layar : *Width* 800 *Pixe*l (*Default*) Besar font beragam disesuaikan	

Gambar III.50 Tampilan Antarmuka Nilai Tertinggi

T08 (Bantuan)	Keterangan
Bantuan 1. ……. 2. ……. 3. ……. 4. ……. 5. ……. …….. **Kembali**	- Klik "Kembali" akan menuju ke halaman T02
Ukuran layar : *Width* 800 *Pixe*l (*Default*) Besar font beragam disesuaikan	

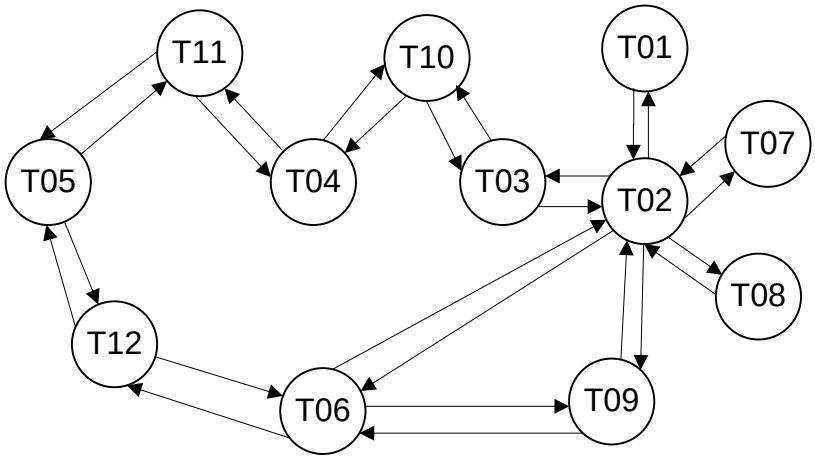
Gambar III.51 Tampilan Antarmuka Bantuan

T09 (Keluar)	Keterangan
Anda yakin ingin keluar? **Ya** **Tidak** **Kembali**	- Klik "Ya" akan menutup aplikasi - Klik Tidaki" akan menuju ke halaman T02
Ukuran layar : *Width* 800 *Pixe*l (*Default*) Besar font beragam disesuaikan	

Gambar III.52 Tampilan Antarmuka Keluar

III.7.4 Jaringan Semantik

Jaringan semantik merupakan jaringan data yang menunjukan hubungan antar berbagai objek. gambar III.53 menunjukkan jaringan semantik dari *game* yang dibangun.



Gambar III.53 Jaringan Semantik

Keterangan:

T01 : Perancangan antarmuka perkenalan *game.*

T02 : Perancangan antarmuka menu utama.

T03 : Perancangan antarmuka mulai bermain di *level* 1.

T04 : Perancangan antarmuka mulai bermain di *level* 2.

T05 : Perancangan antarmuka mulai bermain di *level* 3.

T06 : Perancangan antarmuka permainan selesai.

T07 : Perancangan antarmuka nilai tertinggi.

T08 : Perancangan antarmuka bantuan.

T09 : Perancangan antarmuka pesan keluar.

T10 : Perancangan antarmuka *minigame* penyimpanan sampah pada tempatnya.

T11 : Perancangan antarmuka *minigame* kalahkan AI.

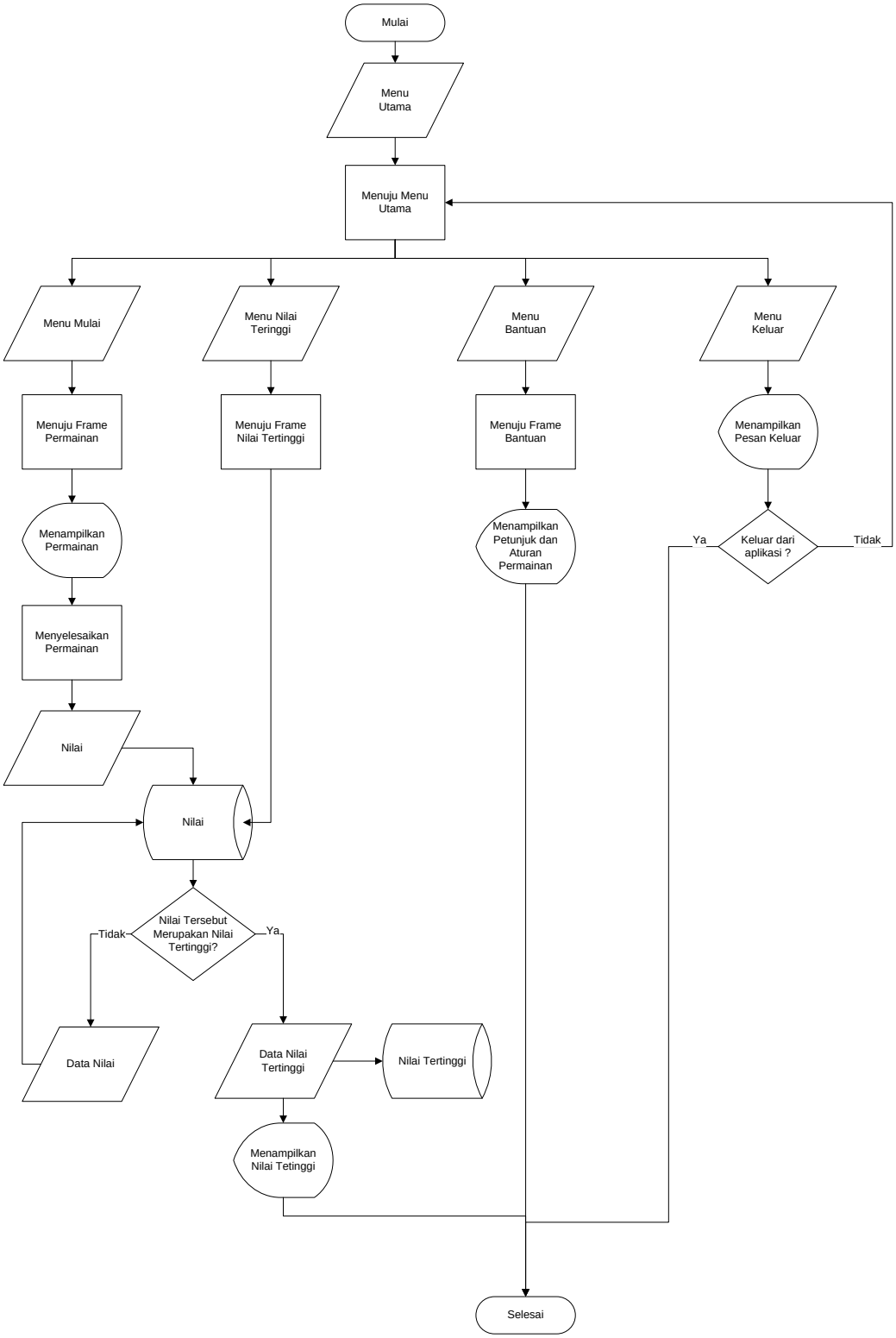
T12 : Perancangan antarmuka *minigame* pilih jenis sampah.

III.7.5 Perancangan Prosedural

Perancangan prosedural merupakan perancangan yang berfungsi untuk mendeskripsikan prosedural-prosedural yang berada di dalam aplikasi. Prosedural dapat dipanggil dengan menyertakan variabel, baik hanya satu variabel, banyak variabel atau bahkan tidak ada sama sekali. Adapun prosedural-prosedural yang terdapat dalam *game* edukasi lindungi kebersihan laut yang akan dibangun dijelaskan melalui *flowchart* sebagai berikut:

1. Perancangan Prosedur Menu Utama

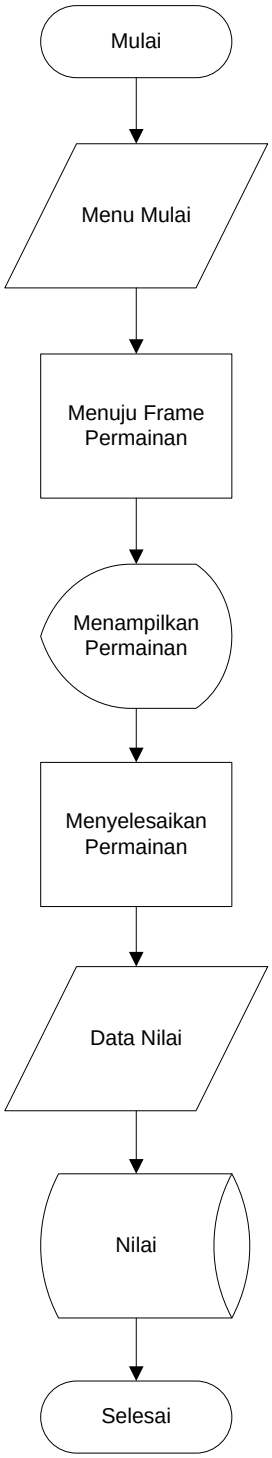
   Pada gambar III.54 merupakan prosedur menu utama pada *game* yang menjelaskan menu-menu apa saja yang ada pada *game*.



Gambar III.54 Prosedur Menu Utama

2. Perancangan Prosedur Menu Mulai

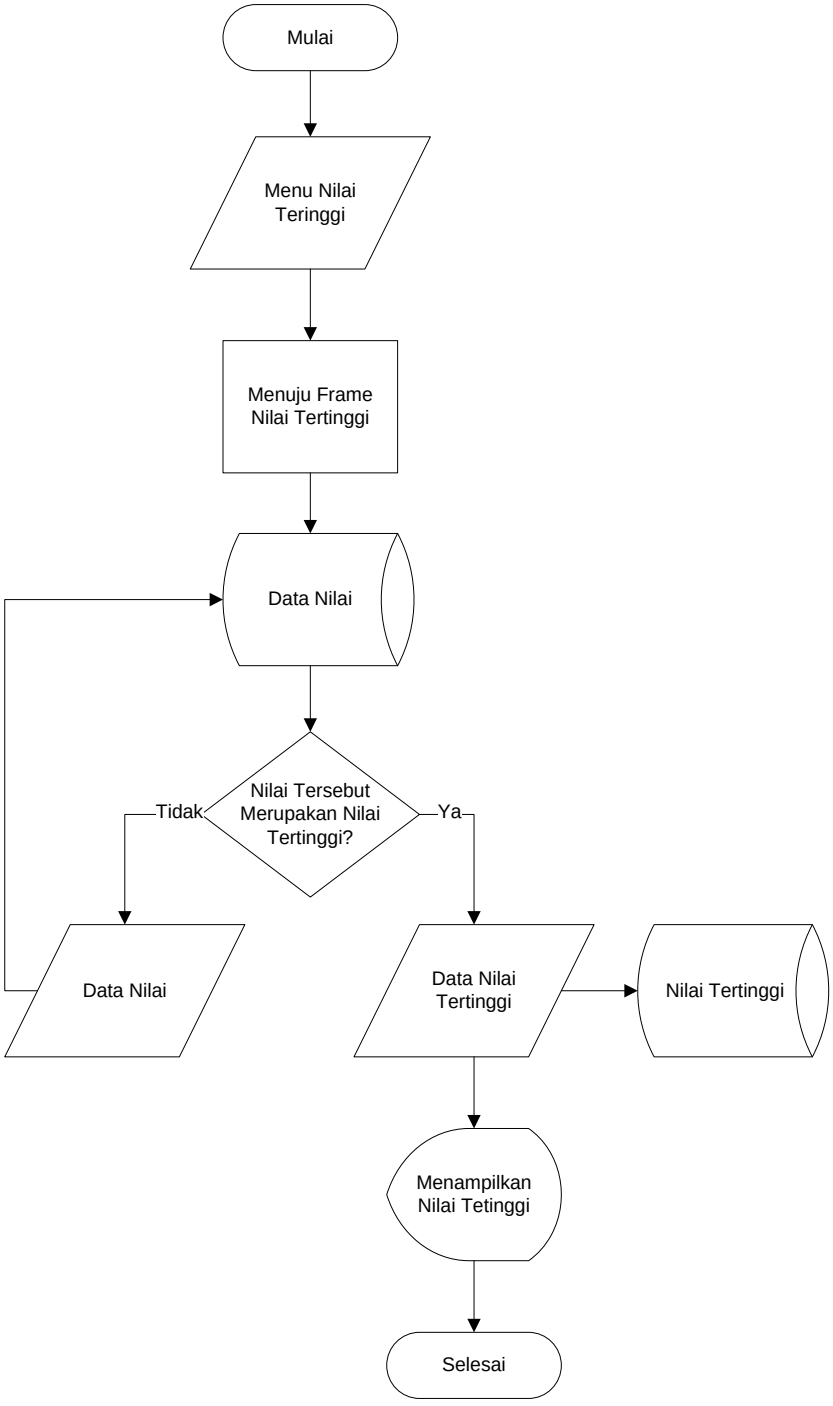
Pada gambar III.55 merupakan prosedur bermain pada *game* yang menjelaskan tentang proses awal dari permainan.



Gambar III.55 Prosedur Menu Mulai

3. Perancangan Prosedur Menu Nilai Tertinggi

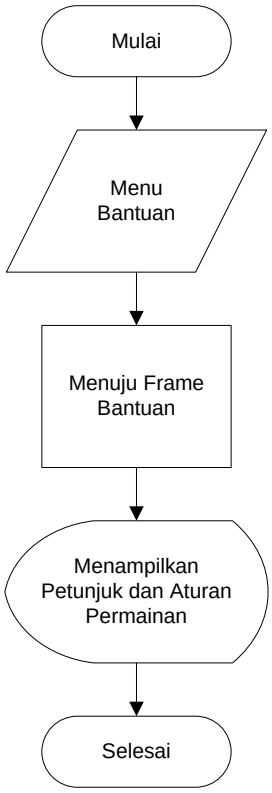
Pada gambar III.56 merupakan prosedur nilai tertinggi pada *game* yang menjelaskan tentang proses menampilkan menu nilai tertinggi.



Gambar III.56 Prosedur Menampilkan Menu Nilai Tertinggi

4. Perancangan Prosedur Menu Bantuan

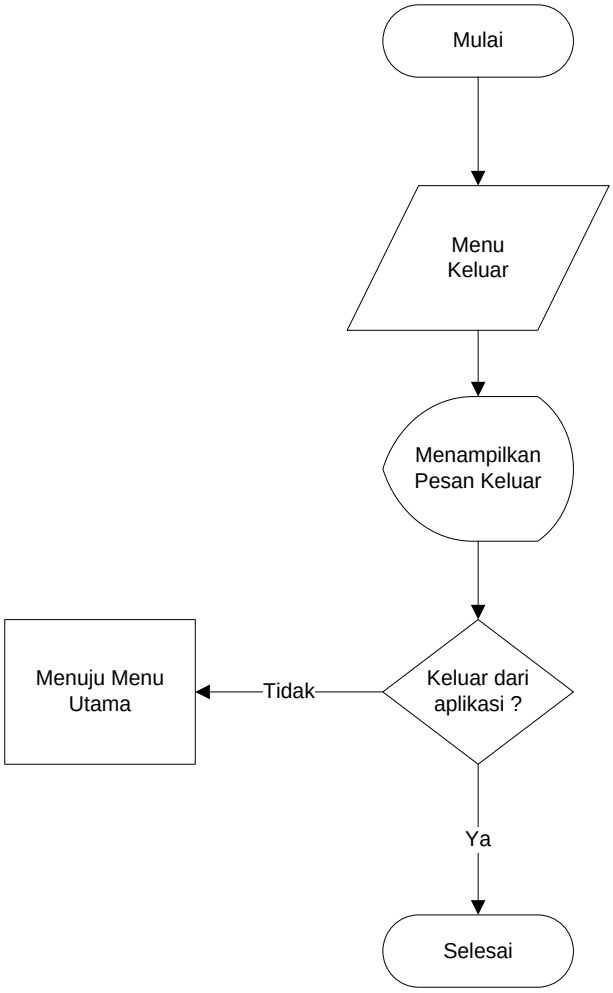
Pada gambar III.57 merupakan prosedur bantuan pada *game* yang menjelaskan tentang proses menampilkan menu bantuan.



Gambar III.57 Prosedur Menampilkan Menu Bantuan

5. Prosedur Prosedur Menu Keluar

Pada gambar III.58 merupakan prosedur keluar pada *game* yang menjelaskan tentang proses menampilkan menu keluar.



Gambar III.58 Prosedur Menampilkan Menu Keluar