BAB 3
ANALISIS DAN PERANCANGAN SISTEM

3.1 Analisis Sistem

Pada bagian ini akan dibahas mengenai analisa-analisa masalah kegunaan dan sistem *game* yang akan dibangun dan analisa pengguna (user).

3.1.1 Analisis Masalah

Kebudayaan adalah sesuatu yang akan mempengaruhi tingkat pengetahuan dan meliputi sistem ide atau gagasan yang terdapat dalam pikiran manusia, sehingga dalam kehidupan sehari-hari, kebudayaan itu bersifat abstrak. Sedangkan perwujudan

kebudayaan adalah benda-benda yang diciptakan oleh manusia sebagai makhluk yang berbudaya, berupa perilaku dan benda-benda yang bersifat nyata.

Kemajuan teknologi informasi dan komunikasi saat ini telah menimbulkan pengaruh terhadap perkembangan budaya bangsa Indonesia . Derasnya arus informasi dan telekomunikasi ternyata menimbulkan sebuah kecenderungan yang mengarah terhadap memudarnya nilai-nilai pelestarian budaya. Selain itu ketertarikan bahkan pengetahuan generasi muda terhadap kebudayaan sendiri amatlah minim. Hal ini disebabkan semakin gencarnya media elektronik, khususnya TV yang selalu menayangkan kebudayaan luar. Hal ini dengan mudahnya merusak pola pikir masyarakat khususnya para generasi muda. Mereka cenderung melupakan kebudayaan sendiri dan beralih ke budaya luar.

Jika hal ini terus dibiarkan, kekayaan budaya yang dimiliki oleh Indonesia akan semakin memudar. Contohnya adalah Reog Ponorogo dari Jawa Timur dan Tari Pendet dari Bali, diklaim sebagai warisan budaya oleh Negara tetangga. Ini disebabkan karena kurangnya rasa tanggung jawab dan kepedulian dari bangsa Indonesia [13].

3.1.2 Analisis *Game* Sejenis

Dalam pembahasan, penulis akan membahas *game* dengan jenis *turn based strategy*. *Game* yang akan dibahas dan menjadi acuan disini ialah *game* Battle for Wesnoth dan *game* Pandava Series.

3.1.2.1 Deskripsi *Game* Battle for Wesnoth

Battle for Wesnoth merupakan sebuah *game* dengan genre *turn based strategy* dengan cerita khayalan. *Game* ini pertama kali dikembangkan oleh David White pada july 2003 yang dibangun dengan bahasa C++ [14].

Dalam Battle for Wesnoth seorang pemain dapat memiliih karakter utma hero yang khas dengan *storyline* yang berbeda-beda. Untuk menyelesaikan permainan, pemain harus menyelesaikan setiap misi yang diberikan dengan cara menghancurkan setiap musuh yang ada dalam map ataupun mempertahankan daerah kekuasannya.

3.1.2.2.1 Kelebihan *Game* Battle for Wesnoth

Berikut ini adalah kelebihan-kelebihan yang ada pada *game* Battle for Wesnoth :

1. Memiliki skenario petualangan yang banyak
2. Memiliki beragam jenis pasukan dengan kemampuan yang berbeda-beda
3. Permainan dapat dilakukan *single player* atau *multiplayer* secara online
4. Adanya perubahan kemampuan pada setiap unit ketika berada pada petak-petak terrain tertentu dan pergantian hari.

3.1.2.2.2 Kekurangan *Game* Battle for Wesnoth

Berikut ini adalah kekurangan-kekurangan yang ada pada *game* Battle for Wesnoth :

1. Cerita yang diangkat berupa fantasi atau khayalan.
2. Sisi edukasinya masih kurang dan lebih menekankan pada permainan *strategy*

3.1.2.2.3 Tampilan *Game* Battle for Wesnoth

Pada Gambar 3.1 di bawah merupakan tampilan permainan pada *game* Battle for Wesnoth.



Gambar 3.1 Tampilan *Game* Battle for Wesnoth

3.1.2.2 Deskripsi *Game* Pandava Series

Pandava Series adalah rangkaian *game* yang diciptakan oleh Creacle *Game* Studio yang terdiri dari 4 *game* dengan judul dari masing-masing *game* adalah: Arjuna the Archer, Bima the Conqueror, Nakula Sadewa in The Labyrinth, dan The Legend of Gatotkaca for Mobile. Keempat *game* tersebut sama-sama kental dengan unsur budaya Indonesia dan mengangkat empat tokoh pahlawan legenda di Indonesia [15].

3.1.2.2.1 Fitur *Game* Pandava Series

Berikut ini adalah fitur-fitur yang ada pada *game* Pandava Series:

1. Mengangkat tema budaya indonesia yaitu tokoh wayang pandawa
2. Setiap cerita dan tokoh wayang pandawa dibangun pada *game* yang berbeda
3. Adanya cerita yang berbeda pada setiap level
4. Berbasis *mobile*
5. Jenis *game* berjenis *adventure*

3.1.2.2.2 Tampilan *Game* Pandava Series

Pada Gambar 3.2 di bawah merupakan tampilan permainan pada *game* Arjuna *the Archer* dan pada gambar 3.3 merupakan tampilan permainan pada *game* Bima *the Conqueror*.



Gambar 3.2 Tampilan *Game* Arjuna *the Archer*



Gambar 3.3 Tampilan Permainan *Game* Bima *the Conqueror*

3.1.3 Analisis *Game* yang Akan Dibangun (War of Baratayuda)

3.1.3.1 Pengenalan *Game* War of Baratayuda

Game yang akan dibuat berjudul War of Baratayuda yang bergenre Turn Based Strategy. *Game* ini dibangun dengan mengaplikasikan teknologi sebagai jembatan untuk menyampaikan kebudayaan Bangsa Indonesia. *Game* ini dibangun dengan grafis 2D (2 Dimensi).

Game ini menerapkan sistem single player dengan data-data yang dibagi ke dalam objek-objek karena menggunakan pemrograman berorientasi objek.

Berikut ini adalah fitur-fitur yang ada pada *game* ini :

1. Sistem single player.
2. Berbasis *desktop.*
3. Mengangkat tema budaya, khususnya budaya jawa, umumnya budaya Indonesia.
4. Jenis *game* bertipe *turn based strategy* yang disajikan secara interaktif.
5. Permainan bersifat edukatif sosial budaya.

3.1.3.2 *Storyline* Awal *Game* War of Baratayuda

Diceritakan pada sebuah cerita Mahabrata ada sebuah perang yang sangat besar dan merupakan klimaks dari cerita Mahabrata. perang tersebut dikenal dengan nama Baratayuda atau Perang di Kurukshetra. Perang ini merupakan puncak perselisihan antara keluarga pandawa (Yudistira, Bima, Arjuna, Nakula dan Sadewa) melawan sepupu mereka yaitu kurawa yang dipimpin oleh Duryudana.

Perebutan kekuasaan yang merupakan penyebab perang ini, terjadi karena para putra Dretarastra (Kurawa) tidak mau menyerahkan tahta kerajaan Kuru kepada saudara mereka yang lebih tua, yaitu Yudistira. Para Korawa, khususnya Duryudana, berambisi menguasai takhta kerajaan Kuru. Namun ambisi tersebut terhalangi sebab Yudistira dipandang lebih layak menjadi Raja Kuru daripada Duryudana. Untuk mewujudkan ambisinya, Duryudana berusaha menyingkirkan Yudistira dan para Pandawa dengan berbagai upaya, termasuk melakukan usaha pembunuhan. Namun kelima putra Pandu tersebut selalu selamat dari kematian.

Setelah gagal dalam usaha pembunuhan, kemudian Kurawa memutuskan untuk menipu para Pandawa dengan cara mengajak mereka bermain dadu, dengan syarat yang kalah harus meninggalkan istana selama tiga belas tahun. Permainan dadu yang sudah disetel dengan licik mengakibatkan Pandawa kalah, sehingga mereka harus meninggalkan kerajaan selama tiga belas tahun dan terpaksa mengasingkan diri ke hutan. Sebelum Pandawa dibuang, Dretarastra (Ayah dari Kurawa) berjanji akan menyerahkan takhta kerajaan Kuru kepada Yudistira sebab ia merupakan putra mahkota Dinasti Kuru yang sulung.

Setelah masa pengasingan selama tiga belas tahun berakhir, sesuai dengan perjanjian yang sah, Pandawa berhak meminta kembali kerajaannya. Namun Duryudana menolak mentah-mentah untuk menyerahkan kembali kerajaannya. Yudistira dan adik-adiknya masih mampu bersabar, sebagai seorang pangeran, mereka merasa wajib dan berhak turut serta dalam administrasi pe-merintahan, maka mereka meminta lima buah desa saja. Tetapi Duryudana sombong dan berkata bahwa ia tidak bersedia memberikan tanah kepada para Pandawa, bahkan yang seluas ujung jarum pun. Jawaban itu membuat para Pandawa tidak bisa

bersabar lagi dan perang tak bisa dihindari. Di pihak lain, Duryudana pun sudah mengharapkan peperangan.

Pertempuran berlangsung selama 18 hari kedua belah pihak bertarung di dataran kurukshetra dan setiap hari terjadi pertempuran yang berlangsung sengit dan mengesankan. Bima sebagai tokoh pandawa sangat mahir dalam memainkan gada memiliki sifat yang gagah, teguh, bersifat selalu kasar dan menakutkan bagi musuh walaupun sebenarnya hatinya lembut. Ia berperan sebagai komandan tentara Pandawa.

Pada hari terakhir Baratayuda, Bima berusaha mengejar Duryudanaa dan kemudian bertarung melawannya untuk mengakhiri perang Baratayuda dan merebut kembali hak kekuasaan yang seharusnya milik Pandawa.

3.1.3.3 *Storyline* Akhir *Game* War of Baratayuda

Akhirnya perang Baratayuda telah berakhir. Ketamakan dari pihak Kurawa (Duryudana) akhirnya dikalahkan oleh kebenaran dari pihak Pandawa (Bima). Karena Sebenarnya ketamakan, kejahatan, kelicikan, dan hal-hal buruk lainnya tidak akan membawa kedamaian dalam kehidupan namun akan membawa kehancuran.

3.1.3.4 *Gameplay Game* War of Baratayuda

Tugas utama pemain (*user*) dalam memainkan Aplikasi *game* War of Baratayuda ini yaitu melalui level-level yang terdiri dari empat level. Ketika pertama kali memainkan user hanya dapat memainkan permainan pada level 1 sedangkan untuk level lainnya masih terkunci. Untuk membuka level, user diharuskan menyelesaikan misi pada level sebelumnya.

Pada setiap level disediakan suatu *terrain* yang terdiri dari unit-unit *enemy* (*Non Playable Character*) dan unit dari pemain. Dari setiap levelnya disediakan *terrain* yang berbeda-beda dan tingkatan kesulitan yang berbeda, pemain harus mampu melewati setiap level dengan menghancurkan unit-unit musuh untuk melanjutkan ke level berikutnya. Pada level terakhir akan ada Boss (Duryudana) beserta pasukannya yang menanti untuk dihancurkan.

Pemain (*user*) dapat memulai permainan dengan memilih pilihan Mulai, kemudian menu pemilihan level. Pada level 1 akan ada jalan cerita pendek mengenai sebab terjadinya perang Baratayuda. Setelah cerita berakhir pemain dapat langsung memulai pemainannya. Selain pilihan Mulai, pada tampilan awal juga disediakan pilihan Instruksi yaitu pilihan untuk mengetahui kontrol-kontrol dan peraturan yang diperlukan untuk memainkan aplikasi ini. Pilihan lainnya adalah pilihan Pembuat yaitu keterangan yang menyediakan informasi mengenai pembuat *game* ini. Kemudian menu Pengaturan untuk mengatur tampilan layar dan suara. Pilihan terakhir yaitu Keluar pilihan untuk keluar dari aplikasi *game*.

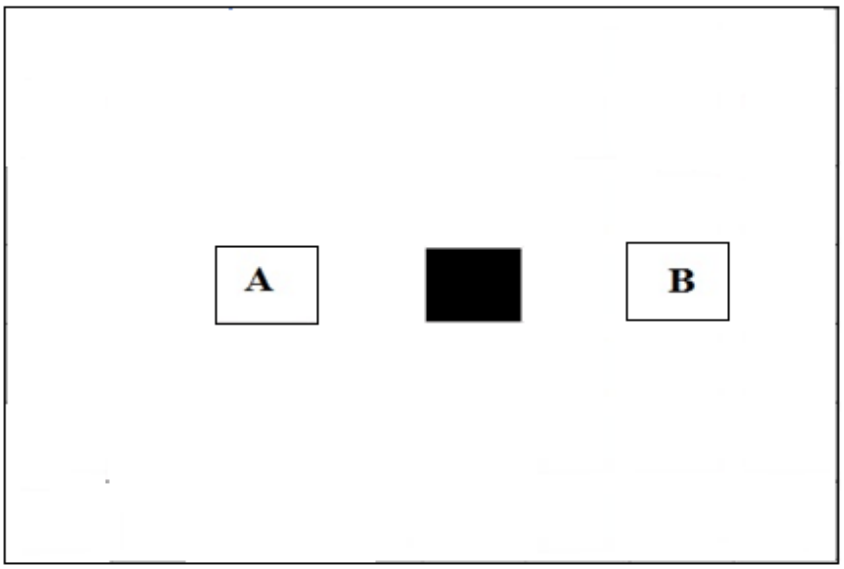
Ada beberapa aturan bermain yang diterapkan dalam pembangunan aplikasi *game* ini yaitu:

1. *User* hanya dapat melakukan aksinya yaitu bergerak atau menyerang ketika mendapat giliran. Ketika giliran berada pada *enemy, user* hanya dapat menunggu sampai giliran *enemy* telah selesai kemudian *user* dapat melakukan aksinya kembali.
2. Ada beberapa jenis karakter pada unit *user* maupun *enemy* yaitu *hero*, prajurit (*soldier*) dan pemanah (*archer*). Setiap unit karakter tersebut memiliki *damage*, jarak bergerak maupun jarak serang yang berbeda.
3. Setiap unit tidak dapat bergerak pada wilayah *unwalkable* (pohon, gunung dan air)
4. *User* dapat menyerang unit *enemy* ketika unit tersebut berada dalam jarak serang unit *user*.
5. *User* dapat menggerakkan unitnya hanya dalam jarak gerak unit tersebut.
6. Jika *user* telah memainkan semua unitnya maka otomatis giliran akan berpindah pada *enemy* atau dapat menekan tombol Selesai.

3.1.3.5 Analisis Algoritma A* Pada *Game* War of Baratayuda

Pada gambar 3.4, kotak A adalah simpul (*node*) awal yang merupakan letak unit *enemy* (*Non Playable Caracter*) berada dimana algoritma A* diterapkan. Kotak berwarna hitam adalah penghalang yang tidak bisa dilewati

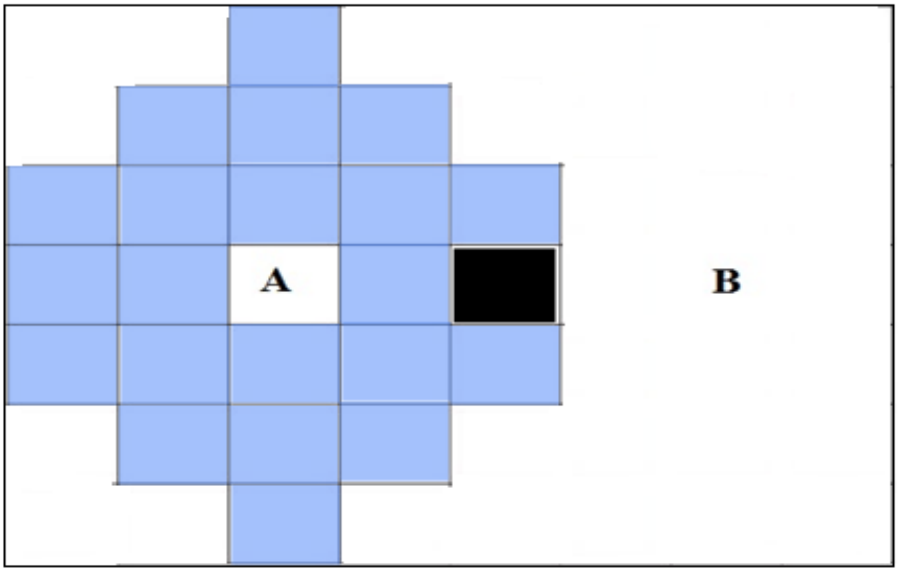
yang merupakan *tile* water atau pohon pada aplikasi *game* yang akan dibangun, dan B adalah simpul tujuan yang merupakan letak dari kunit *user* berada. Gerakan atau langkah yang dapat diambil oleh NPC dan *Player* hanya gerakan horisontal dan vertikal. Unit musuh A (NPC) memiliki batasan gerak seperti pada gambar 3.5 dibawah, unit ini memiliki jarak gerak hanya pada kotak berwarna biru yaitu sebanyak tiga langkah.



Gambar 3.4. Contoh Area A*

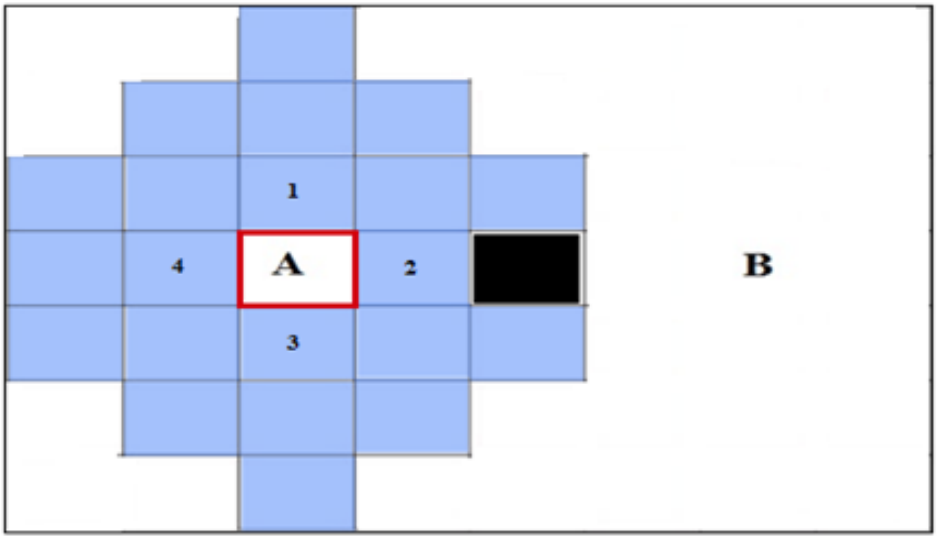
Langkah-langkah algoritma A* pada *enemy* pada *game* yang akan dibangun sebagai berikut:

1. Masukkan *starting node* (simpul A) ke *Open list* karena pada *Open list* hanya terdapat satu simpul (simpul A), masukkan simpul A ke *Closed list* dan jadikan *current node.*
2. Bangkitkan seluruh wilayah gerak simpul A.



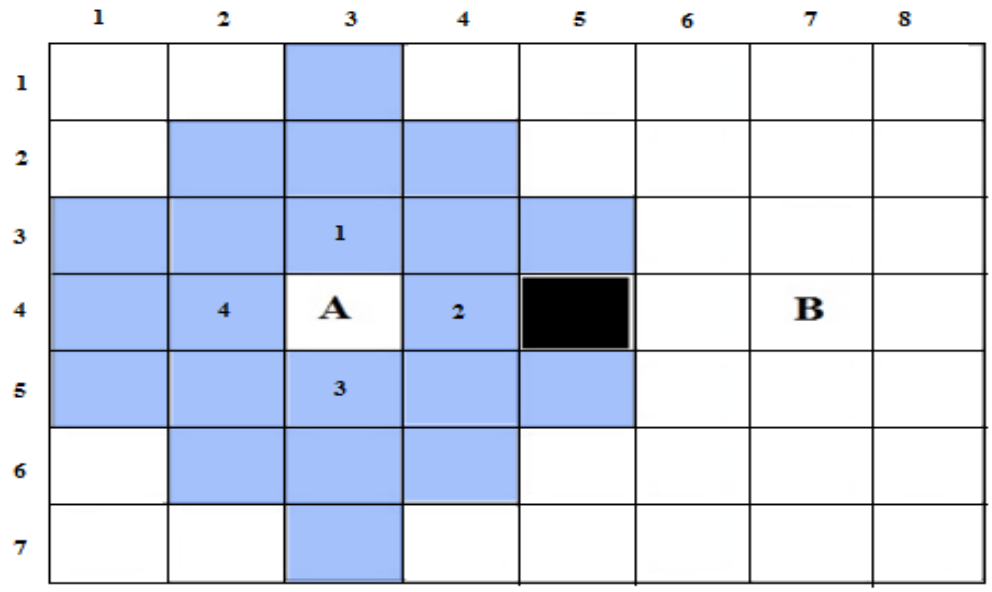
Gambar 3.5 *Move range* Simpul / Unit A

Kemudian bangkitkan suksesor dari *current node* yaitu simpul A kemudian masukkan simpul suksesor tersebut ke *Open list.* Suksesor dari simpul A adalah simpul nomor 1, 2, 3 dan 4 seperti gambar dibawah.



Gambar 3.6 Simpul Suksesor Simpul A

3. Kemudian hitung nilai F dari setiap simpul suksesor dan cari suksesor dengan nilai F terkecil. Untuk memudahkan dalam perhitungan, map di atas dijadikan *grid* dan diberi penomoran seperti gambar di bawah ini:



Gambar 3.7 Area Map Setelah diberi *Grid* dan Penomoran

Hitung nilai F dengan menambahkan nilai G dan H. Nilai G bernilai 1 ketika arah horisontal dan vertikal, karena dalam aplikasi *game* yang akan dibangun unit hanya dapat bergerak horisontal dan vertikal tidak ada nilai untuk diagonal. Nilai H didapat dari rumus :

h(n) = D * (abs(n.x-goal.x) + abs(n.y-goal.y))

Simpul 1 :

F(1) = G + H

= 1 + D * (abs(n.x-goal.x) + abs(n.y-goal.y))

= 1 + 1 * (abs(3-7) + abs(3-4))

= 1 + 5

= 6

Simpul 2 :

F(2) = G + H

= 1 + D * (abs(n.x-goal.x) + abs(n.y-goal.y))

= 1 + 1 * (abs(4-7) + abs(4-4))

= 1 + 3

= 4

Simpul 3 :

F(3) = G + H

= 1 + D * (abs(n.x-goal.x) + abs(n.y-goal.y))

= 1 + 1 * (abs(3-7) + abs(5-4))

= 1 + 5

= 6

Simpul 4 :

F(4) = G + H

= 1 + D * (abs(n.x-goal.x) + abs(n.y-goal.y))

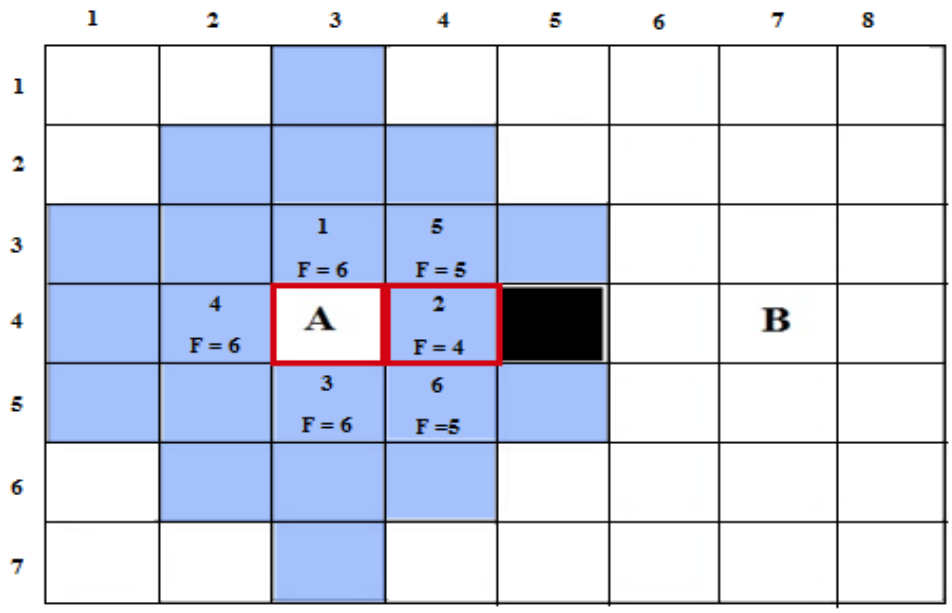
= 1 + 1 * (abs(2-7) + abs(4-4))

= 1 + 5

= 6

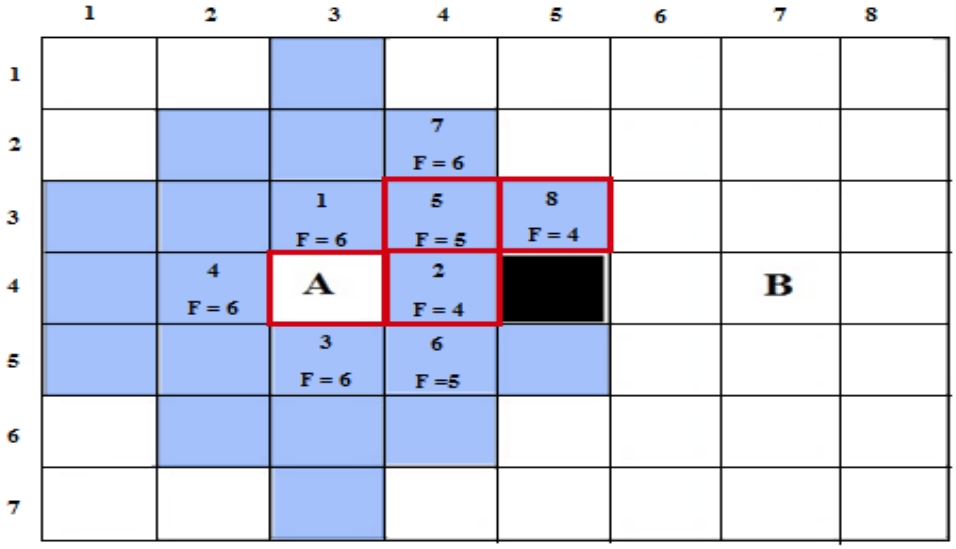
4. Didapatkan nilai F terkecil yaitu 4 pada simpul 2, pindahkan ke *Closed list* dan jadikan simpul 2 menjadi *current node.*
5. Kemudian bangkitkan suksesor dari *current node* atau simpul 2, dapat dilihat pada gambar 3.6 yaitu simpul 5 dan 6, dengan aturan sebagai berikut:
   - Jika sudah terdapat dalam *Closed list* dan merupakan simpul *unwalkable,* abaikan. Dalam kasus ini simpul simpul 2 dan simpul A berada dalam *Closed list* dan simpul berwarna hitam merupakan halangan (simpul *unwalkable*). Maka simpul-simpul tersebut diabaikan.
   - Jika belum berada di *Open list,* masukkan ke *Open list*. Simpan *current node* sebagai "*parent*" dari node-node ini. Simpan biaya masing-masing simpul.

     Dalam kasus ini simpul 5 dan 6 belum berada di *Open list,* masukkan ke *Open list*. Simpan simpul 2 sebagai *parent.* Simpan nilai masing-masing simpul.
   - Jika sudah ada dalam *Open list,* periksa jika simpul ini mempunyai nilai lebih kecil dibanding simpul sebelumnya. Jika lebih kecil, jadikan sebagai *current node* dan ganti *parent* simpul ini. Dalam kasus ini, semua suksesor simpul (3,3) yaitu simpul 5 dan 6 belum berada dalam *Open list.*



Gambar 3.8 Simpul Suksesor pada Simpul 2

6. Kembali ke langkah 2 sampai simpul tujuan ditemukan (berada di *Open list*).
7. Setelah melakukan beberapa kali langkah perhitungan, didapat hasil *path* dengan garis kotak berwarna merah seperti gambar 3.7 di bawah ini.



Gambar 3.9 Jalur Pencarian pada A*

3.2 Analisis Kebutuhan Non-Fungsional

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

3.2.1 Analisis Kebutuhan Perangkat Lunak

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

1. Sistem Operasi Windows 7
2. Game Maker 8 pro

Perangkat lunak yang dibutuhkan untuk menjalankan aplikasi *game* War of Baratayuda adalah sebagai berikut:

1. Sistem Operasi Windows 7

3.2.2 Analisis Kebutuhan Perangkat Keras

Agar aplikasi dapat berjalan dengan baik, maka dibutuhkan perangkat keras yang sesuai dengan kebutuhan aplikasi. Spesifikasi perangkat keras yang digunakan untuk membangun aplikasi *game* War of Baratayuda ini adalah sebagai berikut:

1. Prosesor dengan kecepatan 2.30GHz
2. RAM : 4GB
3. VGA 2GB
4. Harddisk 600GB

5. Monitor
6. Mouse dan Keyboard

Spesifikasi minimum perangkat keras agar dapat menjalankan *game* War of Baratayuda ini adalah sebagai berikut:

1. Prosesor dengan kecepatan 1.8 GHz
2. RAM 526 MB
3. VGA 256 MB
4. Harddisk 30 GB
5. Monitor
6. Mouse dan Keyboard

3.2.3 Analisis Pengguna

Game yang akan dibuat ditujukan untuk user anak-anak dari usia 15 tahun sampai user dewasa, yang dapat mengerti dan memahami komputer sehingga dapat menggunakan aplikasi yang akan dibangun.

3.3 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.

3.3.1 Perancangan Sistem

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

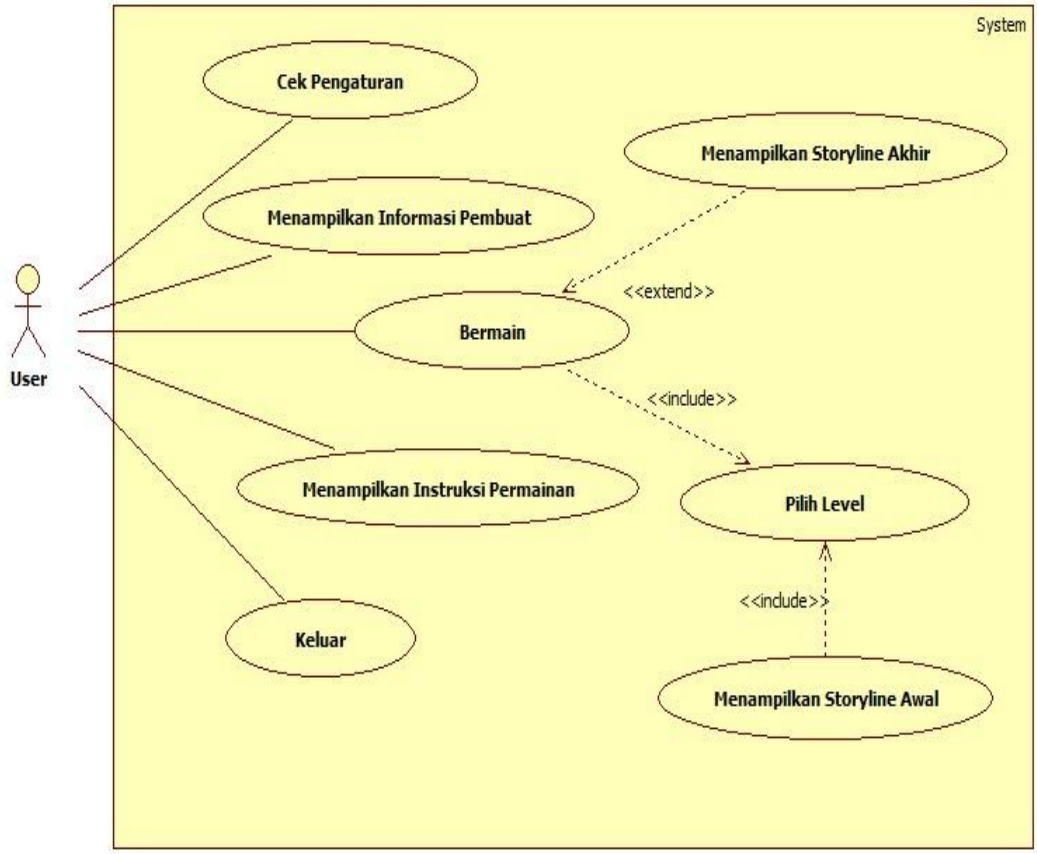
Berdasarkan uraian diatas perancangan sistem merupakan tahapan dari siklus pengembangan sistem yang didefinisikan dari kebutuhan-kebutuhan fungsional dan persiapan untuk rancang bangun implementasi yang menggambarkan bagaimana suatu sistem dibentuk, yang dapat berupa penggambaran, perancangan, dan pembuatan 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.

Perancangan sistem dimodelkan dengan menggunakan *UML (Unified Modeling Language).* Dimana tahap-tahap perancangan yang dilakukan dalam membangun *game* War of Baratayuda antara lain *Use Case Diagram, Class Diagram, Sequence Diagram, Activity Diagram.*

1 *Use Case Diagram*

Use case atau diagram *use case* merupakan pemodelan untuk melakukan (*behavior*) sistem yang akan dibuat.



Gambar 3.10 *Use Case Diagram Game* War of Baratayuda

1.1 Definisi *Actor*

Definisi *Actor* berfungsi untuk menjelaskan *Actor* yang berperan dalam sistem pada *Use Case Diagram.* Definisi *Actor* diterapkan pada Tabel 3.1

Tabel 3.1 Definisi *Actor*

No	*Actor*	Deskripsi
1.	*User*	Orang yang akan memainkan *game*

1.2 Definisi *Use Case*

Definisi *Use Case* berfungsi untuk menjelaskan proses yang terdapat pada setiap *Use Case*.

Tabel 3.2 Defini *Use Case*

No	Use Case	Deskripsi
1.	Menampilkan Informasi Pembuat	Proses untuk melihat informasi pembuat aplikasi
2.	Menampilkan Instruksi Permainan	Proses untuk menampilkan panduan bermain
3.	Cek Pengaturan	Proses untuk mengatur pengaturan *game*
4.	Menampilkan *Storyline* Awal	Proses untuk menampilkan jalan cerita awal *game*
5.	Bermain	Proses melakukan permainan *game* yang akan dibangun
6.	Pilih Level	Proses untuk memilih level *game*
7.	Menampilkan *Storyline* Akhir	Proses untuk menampilkan jalan cerita akhir *game*
8.	Keluar	Proses untuk keluar dari aplikasi

1.3 Skenario *Use Case*

Skenario *Use Case* mnggambarkan alur penggunaan sistem dimana setiap skenario digambarkan dari sudut pandang aktor, seseorang atau piranti yang berinteraksi dengan perangkat lunak dalam berbagai cara.

1.3.1 Skenario *Use Case* Menampilkan Informasi Pembuat

Tabel 3.3 Skenario *Use Case* Menampilkan Informasi Pembuat

Identifikasi	
Nomor	2
Nama	Menampilkan Informasi Pembuat
Tujuan	Menampilkan informasi pembuat aplikasi
Deskripsi	Proses untuk menampilkan informasi pembuat aplikasi
Aktor	*User*
Skenario Utama	
Kondisi Awal	*User* berada di menu utama
Aksi Aktor	**Reaksi Sistem**
1. Memilih menu Pembuat	
	2. Menampilkan menu informasi pembuat *game*
Kondisi Akhir	Aplikasi *game* menampilkan menu informasi pembuat *game*

1.3.2 Skenario *Use Case* Menampilkan Instruksi Permainan

Tabel 3.4 Skenario *Use Case* Menampilkan Instruksi Permainan

Identifikasi	
Nomor	3
Nama	Menampilkan Instruksi Permainan
Tujuan	Menampilkan panduan bermain
Deskripsi	Proses untuk menampilkan informasi panduan dalam bermain

Aktor	*User*
Skenario Utama	
Kondisi Awal	*User* berada di menu utama
Aksi Aktor	**Reaksi Sistem**
1. Memilih menu Instruksi	
	2. Menampilkan menu yang ada di menu Instruksi Permainan (informasi tentang panduan dalam bermain)
Kondisi Akhir	Aplikasi *game* menampilkan petunjuk dalam bermain

1.3.3 Skenario *Use Case* Cek Pengaturan

Tabel 3.5 Skenario *Use Case* Cek Pengaturan

Identifikasi	
Nomor	4
Nama	*Cek* Pengaturan
Tujuan	Mengatur pengaturan *game*
Deskripsi	Proses untuk mengatur pengaturan pada *game*
Aktor	*User*
Skenario Utama	
Kondisi Awal	*User* berada di menu utama
Aksi Aktor	**Reaksi Sistem**
1. Memilih menu Pengaturan	
	2. Menampilkan menu yang ada di menu Pengaturan
3. Mengatur pengaturan suara	
	4. Jika memilih menu ”On”, musik diaktifkan
	5. Jika memilih menu “Off”, musik dimatikan
6. Mengatur pengaturan	

layar	
	7. Jika memilih menu "√", layar menjadi *fullscreen*
	8. Jika memilih menu "X", *window mode* diaktifkan
Kondisi Akhir	Aplikasi *game* menampilkan pengaturan *game*

1.3.4 Skenario *Use Case* Menampilkan *Storyline* Awal

Tabel 3.6 Skenario *Use Case* Menampilkan *Storyline*

Identifikasi	
Nomor	5
Nama	Menampilkan *Storyline* Awal
Tujuan	Menampilkan jalan cerita awal permainan yaitu cerita pendek mengenai sebab terjadinya peperangan.
Deskripsi	Proses untuk menampilkan jalan cerita awal *game*
Aktor	*User*
Skenario Utama	
Kondisi Awal	*User* memilih level 1 pada menu pilih level
Aksi Aktor	**Reaksi Sistem**
	1. Menampilkan jalan cerita awal *game*
2. Menekan tombol Lanjutkan	
	3. Meneruskan jalan cerita
4. Menekan tombol Lewati	
	5. Melewati jalan cerita
Kondisi Akhir	Aplikasi *game* menampilkan jalan cerita awal *game*

1.3.5 Skenario *Use Case* Bermain

Tabel 3.7 Skenario *Use Case* Bermain

Identifikasi	
Nomor	6
Nama	Bermain
Tujuan	Melakukan permainan berupa *game* denga tipe *strategy*
Deskripsi	Proses melakukan permainan berupa *game* dengan tipe *strategy*
Aktor	*User*
Skenario Utama	
Kondisi Awal	*User* memilih salah satu level pada menu pilih level (2,3,4) atau setelah selesai jalan cerita ketika memilih level 1
Aksi Aktor	**Reaksi Sistem**
	1. Menampilkan halaman permainan
2. *User* memilih karakter dengan klik kiri pada mouse dan mengarahkan mouse pada area map kemudian klik kiri mouse	
	3. Karakter yang dipilih oleh *User* akan bergerak sesuai arahan *User*
4. *User* memilih karakter dengan klik kiri pada mouse dan klik kiri pada mouse pilih menu Serang	
	5. Karakter yang dipilih oleh *User* akan menyerang karakter lawan sesuai arahan *User*
6. User memilih menu	

Selesai	
	7. Karakter lawan bergerak atau menyerang
	8. Cek jumlah karakter
	9. Jika jumlah karakter user = 0 menampilkan pesan kalah
	10. Cek jumlah karakter dan cek keberadaan level
	11. Jika jumlah karakter musuh = 0, . Jika tidak berada pada level terakhir atau bukan level 4, melanjutkan ke level selanjutnya
	12. Jika jumlah karakter musuh = 0, Jika berada pada level terakhir atau level 4, menampilkan jalan cerita terakhir
Kondisi Akhir	*User* melakukan permainan

1.3.6 Skenario *Use Case* Pilih Level

Tabel 3.8 Skenario *Use Case* Pilih Level

Identifikasi	
Nomor	7
Nama	Pilih Level
Tujuan	Memilih level *game*
Deskripsi	Proses untuk memilih level *game* (level 1, level 2, level 3, dan level 4)
Aktor	*User*
Skenario Utama	
Kondisi Awal	User berada di menu pilih level
Aksi Aktor	**Reaksi Sistem**
1. *User* memilih level 1	
	2. Menampilkan *storyline game*
	3. Menampilkan permainan level 1
4. *User* memilih level 2	

	5. Menampilkan permainan level 2
6. *User* memilih level 3	
	7. Menampilkan permainan level 3
8. *User* memilih level 4	
	9. Menampilkan permainan level 4
Kondisi Akhir	*User* melakukan permainan level yang dipilih

1.3.7 *Skenario Use Case* Menampilkan *Storyline* Akhir

Tabel 3.9 Skenario *Use Case* Menampilkan *Storyline* Akhir

Identifikasi	
Nomor	8
Nama	Menampilkan *Storyline* Akhir
Tujuan	Menampilkan jalan cerita akhir permainan yang menandakan bahwa user telah berhasil memenangkan *game*
Deskripsi	Proses untuk menampilkan jalan cerita akhir *game*
Aktor	*User*
Skenario Utama	
Kondisi Awal	*User* berada pada halaman permainan level 4
Aksi Aktor	**Reaksi Sistem**
1. Memenangkan permainan level 4	
	2. Menampilkan pesan memenangkan *game*
3. Memilih tombol “Ok”	
	4. Menampilkan jalan cerita akhir *game*
Kondisi Akhir	Aplikasi *game* menampilkan jalan cerita akhir *game*

1.3.8 *Skenario Use Case* Keluar

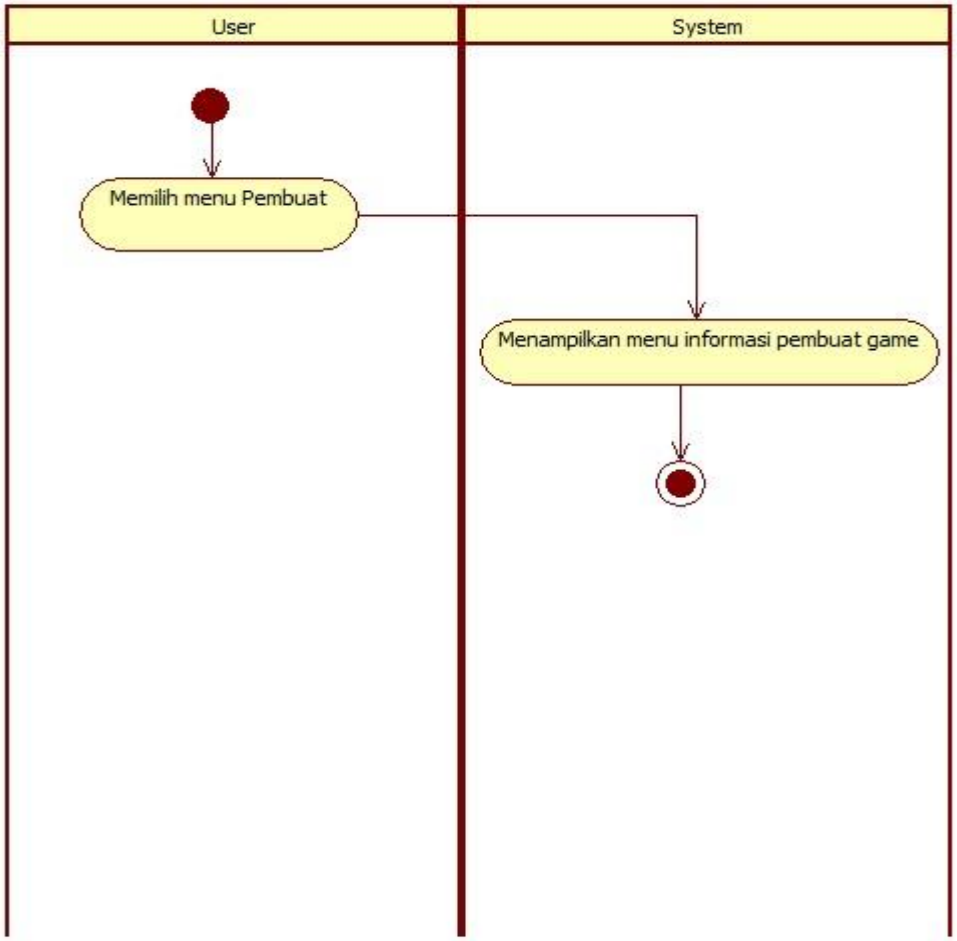
Tabel 3.10 Skenario *Use Case* Keluar

Identifikasi	
Nomor	9
Nama	Keluar
Tujuan	Kelur dari aplikasi *game*
Deskripsi	Proses untuk keluar dari aplikasi *game*
Aktor	*User*
Skenario Utama	
Kondisi Awal	*User* berada pada menu utama
Aksi Aktor	**Reaksi Sistem**
1. Memilih menu Keluar	
	2. Menampilkan pesan verifikasi keluar dari aplikasi *game*
	3. Jika memilih “Ya” keluar dari aplikasi *game*
	4. Jika memilih “Tidak” kembali ke Menu Utama
Kondisi Akhir	Keluar dari aplikasi *game* atau kembali ke Menu Utama

2 *Activity Diagram*

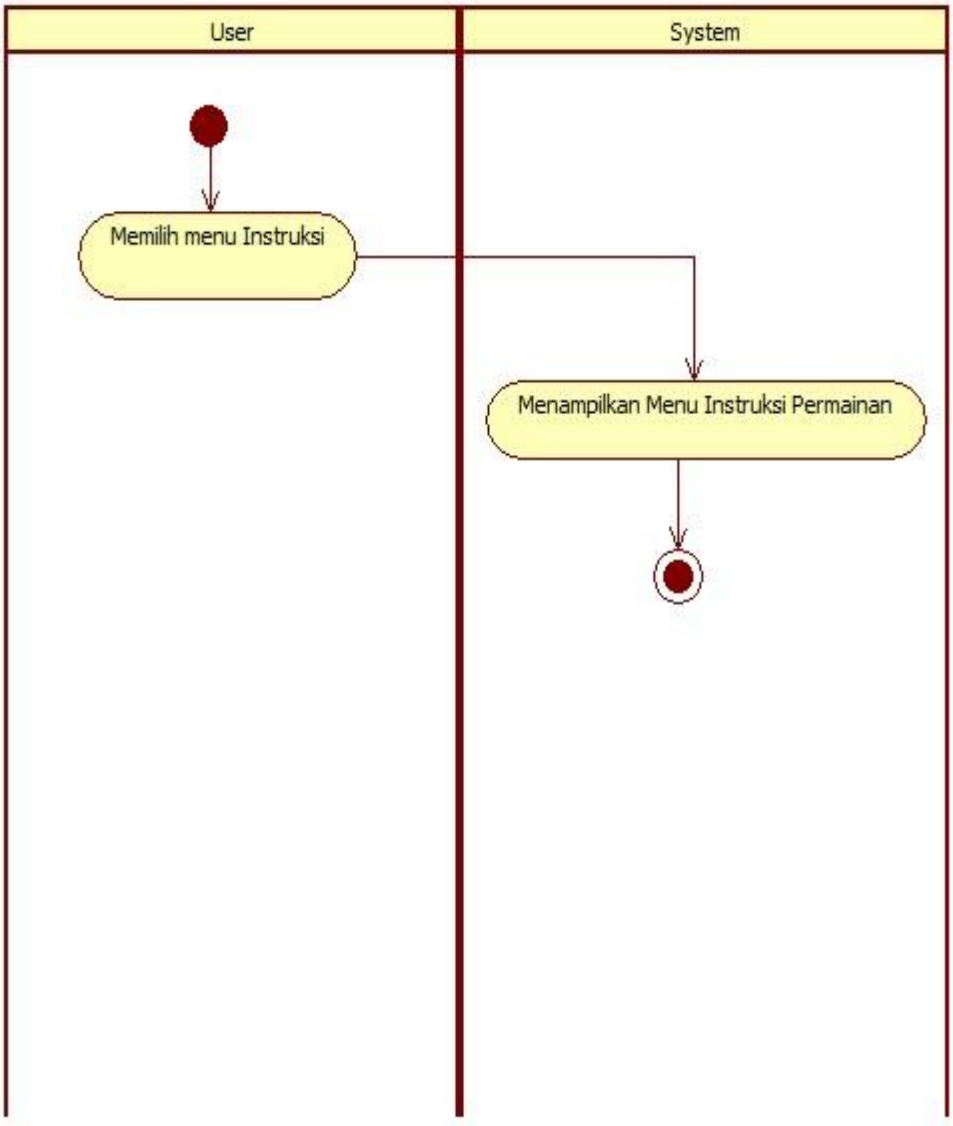
Activity Diagram atau diagram aktivitas lebih memfokuskan diri pada eksekusi dan alur sistem dari pada bagaimana sistem itu dirakit. *Activity diagram* menunjukan aktivitas sistem dalam bentuk kumpulan aksi-aksi [10].

2.1 *Activity Diagram* Menampilkan Informasi Pembuat



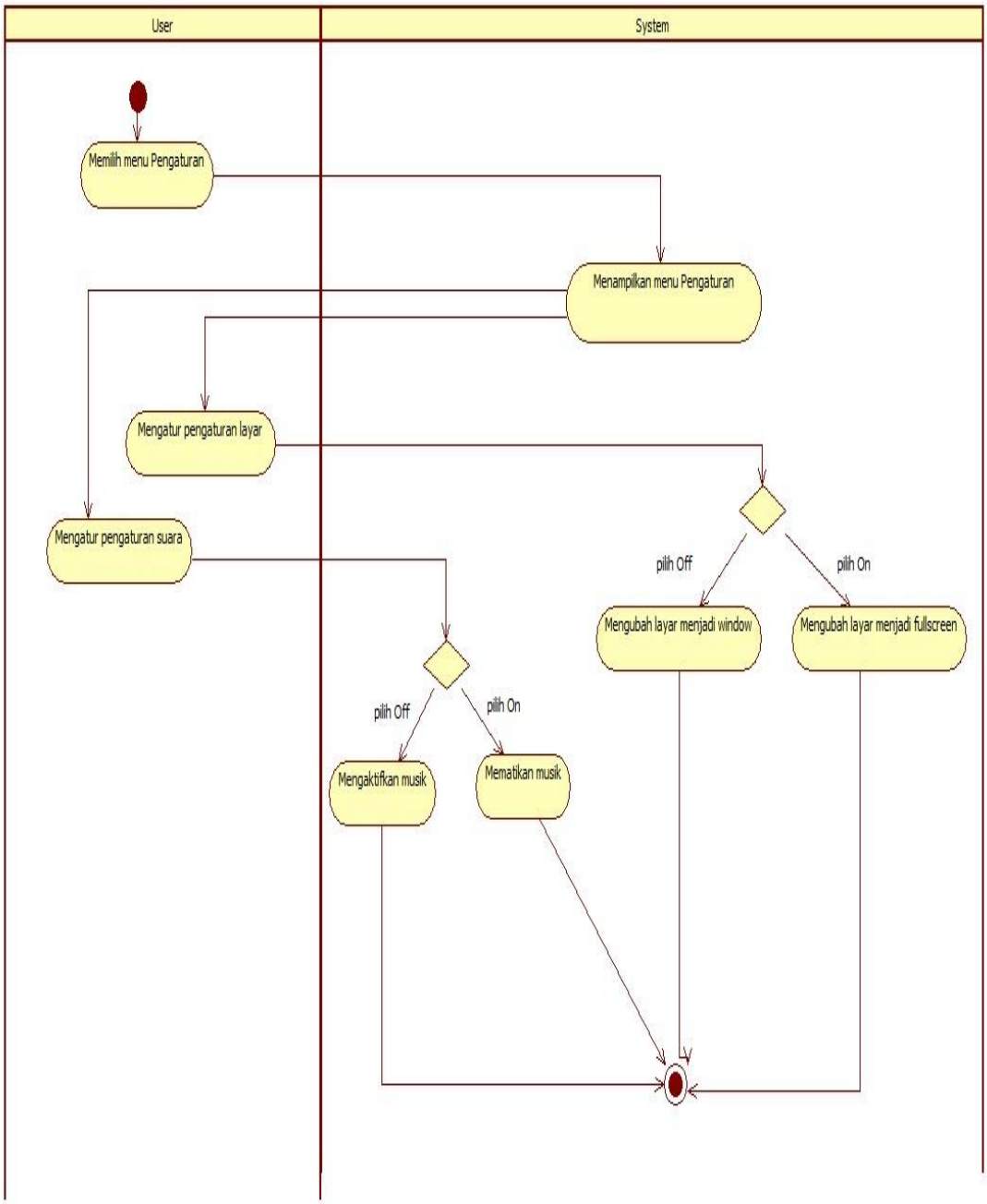
Gambar 3.11 *Activity Diagram* Menampilkan Informasi Pembuat

2.2 *Activity Diagram* Menampilkan Instruksi Permainan



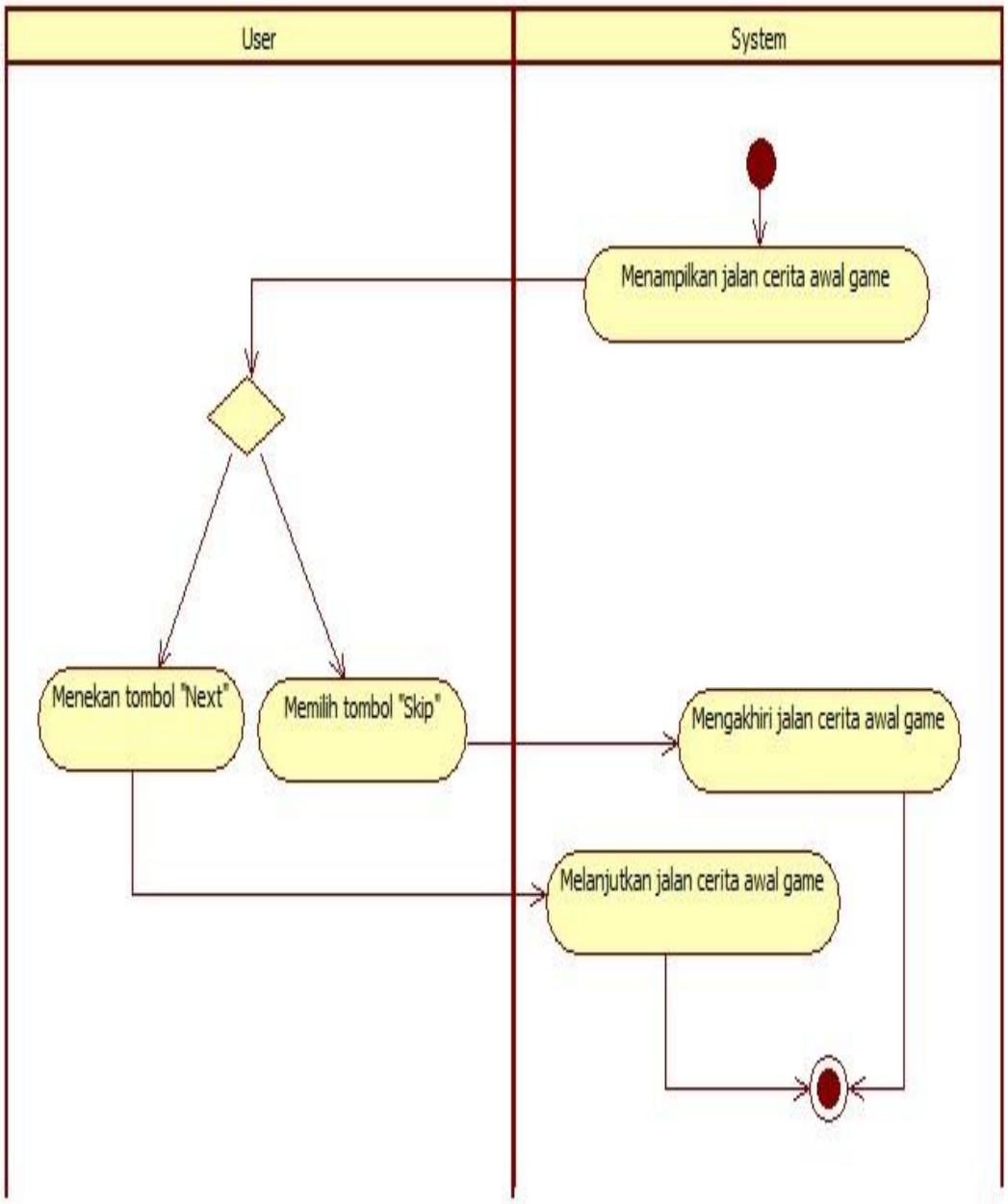
Gambar 3.12 *Activity Diagram* Menampilkan Instruksi Permainan

2.3 *Activity Diagram* Cek Pengaturan



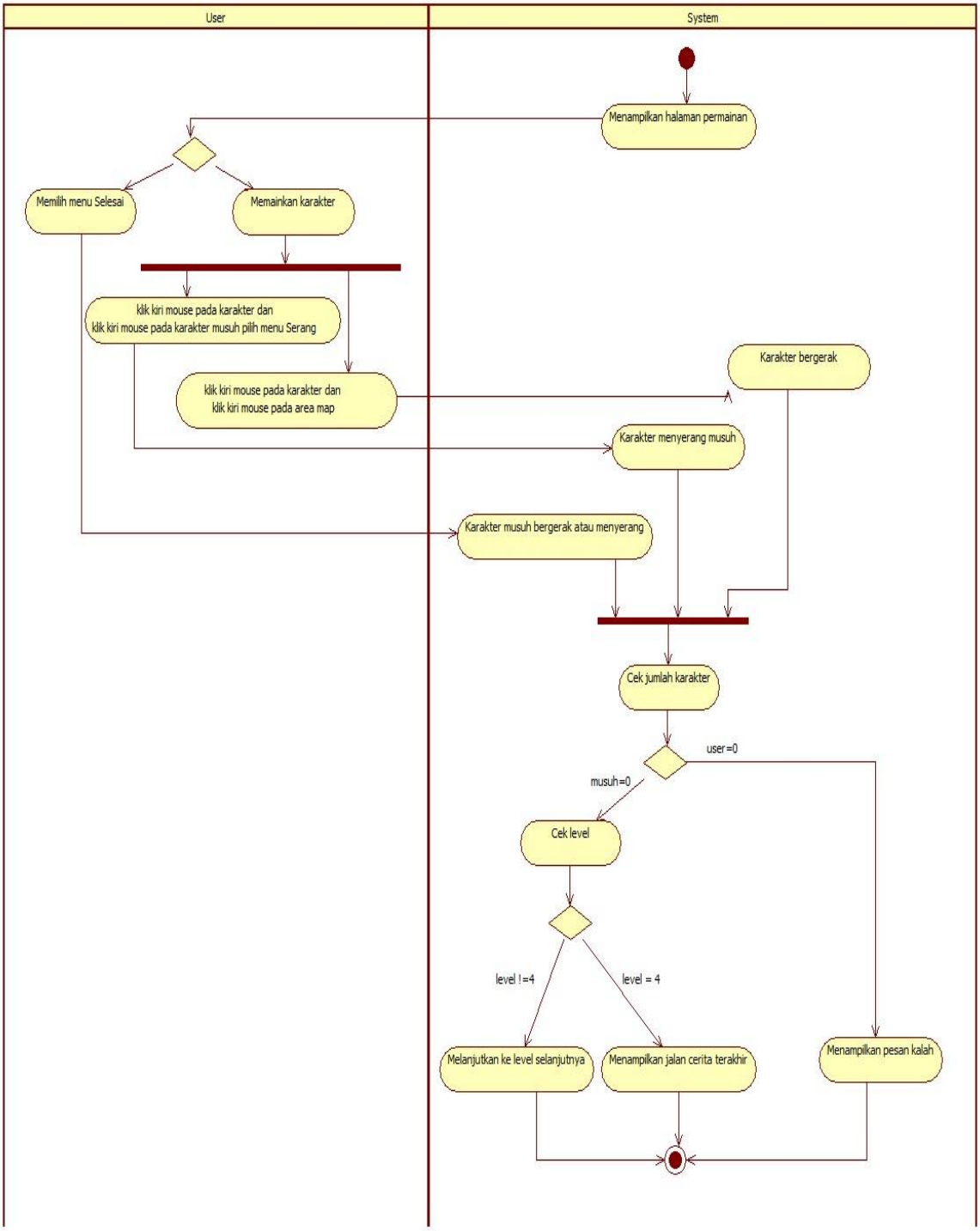
Gambar 3.13 *Activity Diagram* Menampilkan Cek Pengaturan

2.4 *Activity Diagram* Menampilkan *Storyline* Awal



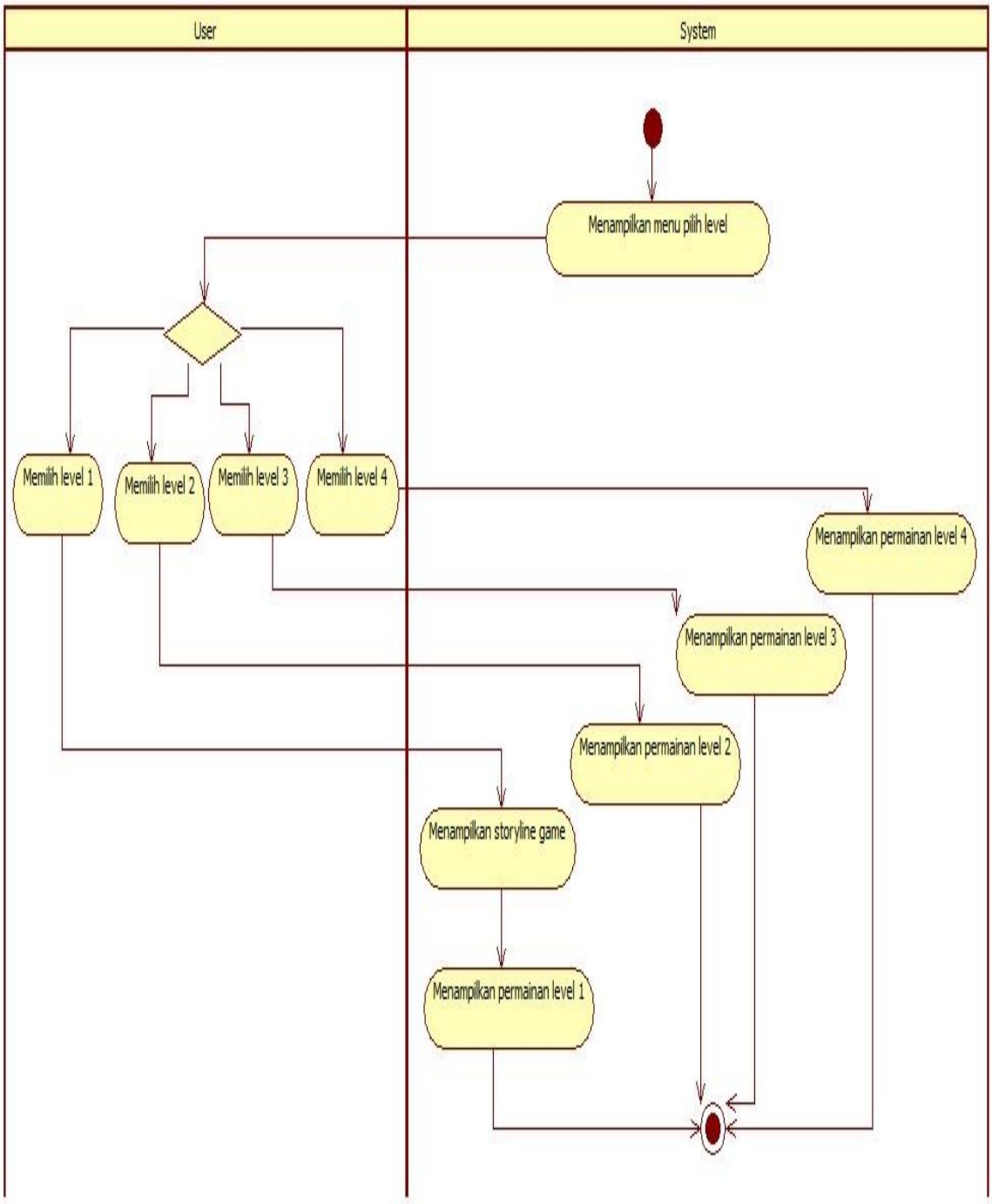
Gambar 3.14 *Activity Diagram* Menampilkan *Storyline*

2.5 *Activity Diagram* Bermain



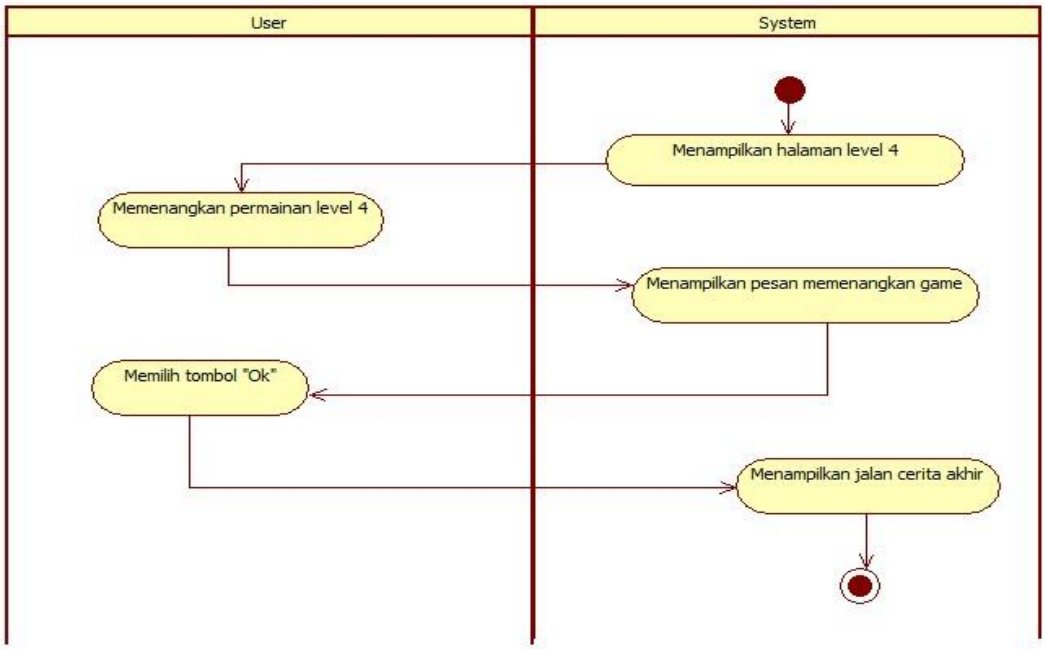
Gambar 3.15 *Activity Diagram* Bermain

2.6 *Activity Diagram* Pilih Level



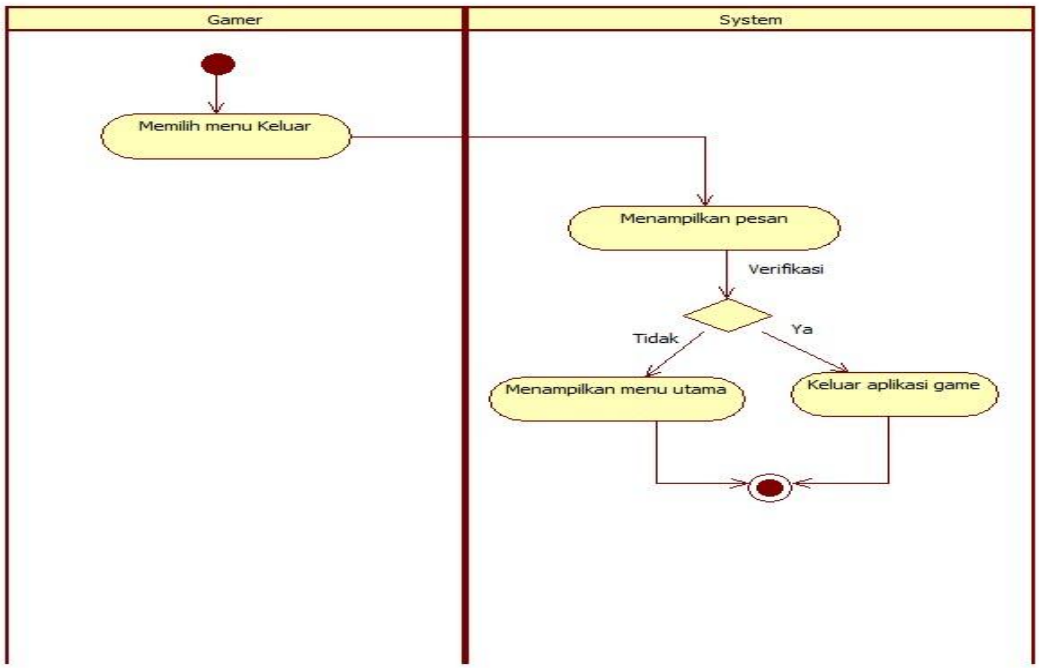
Gambar 3.16 *Activity Diagram* Pilih Level

2.7 *Activity Diagram* Menampilkan *Storyline* Akhir



Gambar 3.17 *Activity Diagram* Menampilkan *Storyline* Akhir

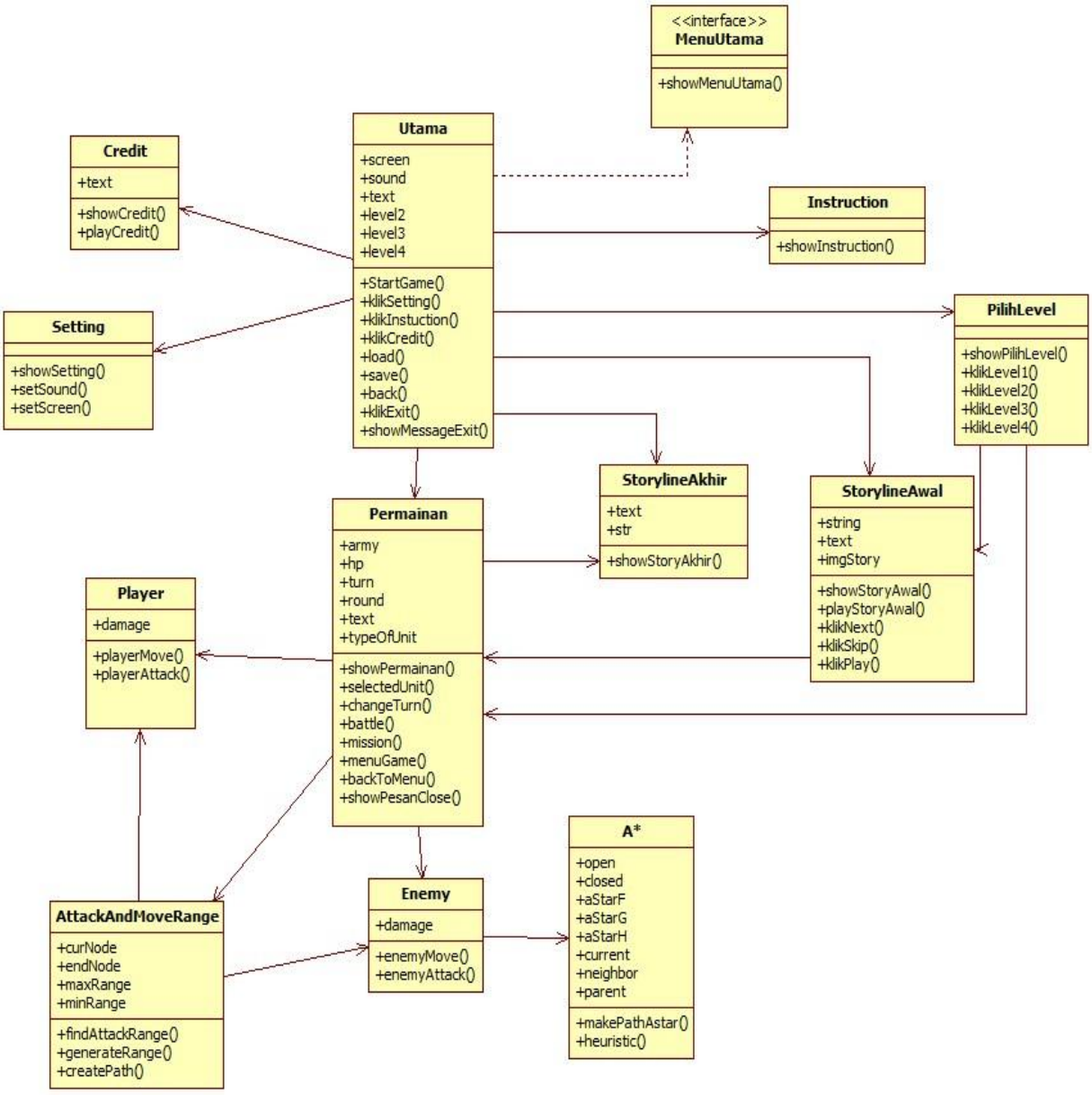
2.8 *Activity Diagram* Keluar



Gambar 3.18 *Activity Diagram* Keluar

3 *Class Diagram*

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

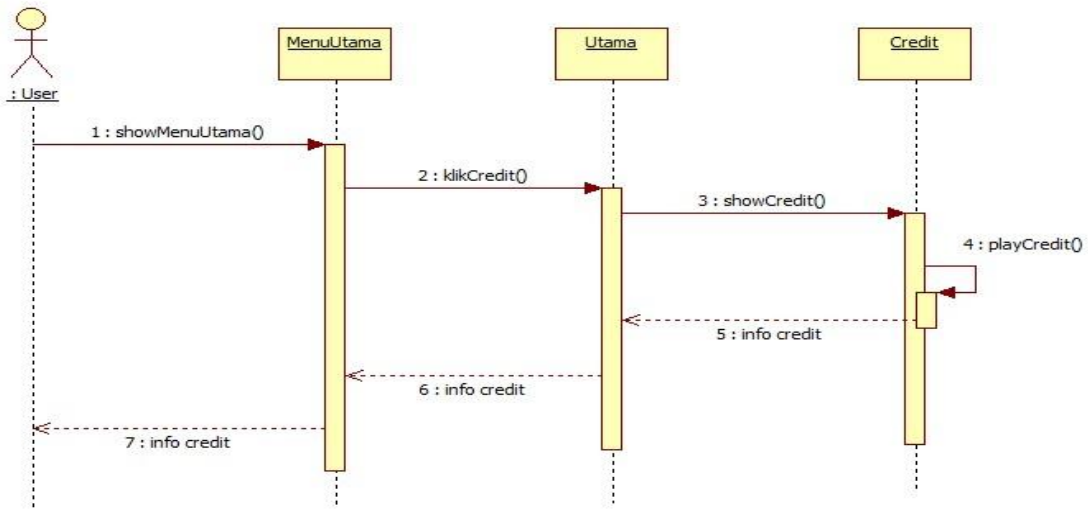


Gambar 3.19 *Class Diagram Game* War of Baratayuda

4 *Sequence Diagram*

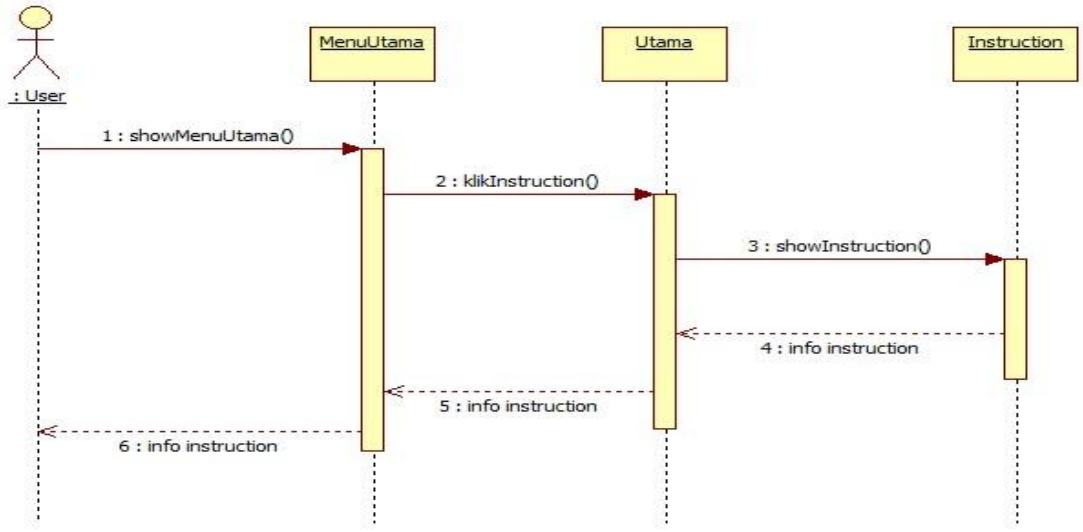
Sequence Diagram atau Diagram sekuen menggambarkan kelakuan objek pada use case dengan mendeskripsikan waktu hidup objek dan message yang dikirimkan dan diterima antarobjek.

4.1 *Sequence Diagram* Menampilkan Informasi Pembuat



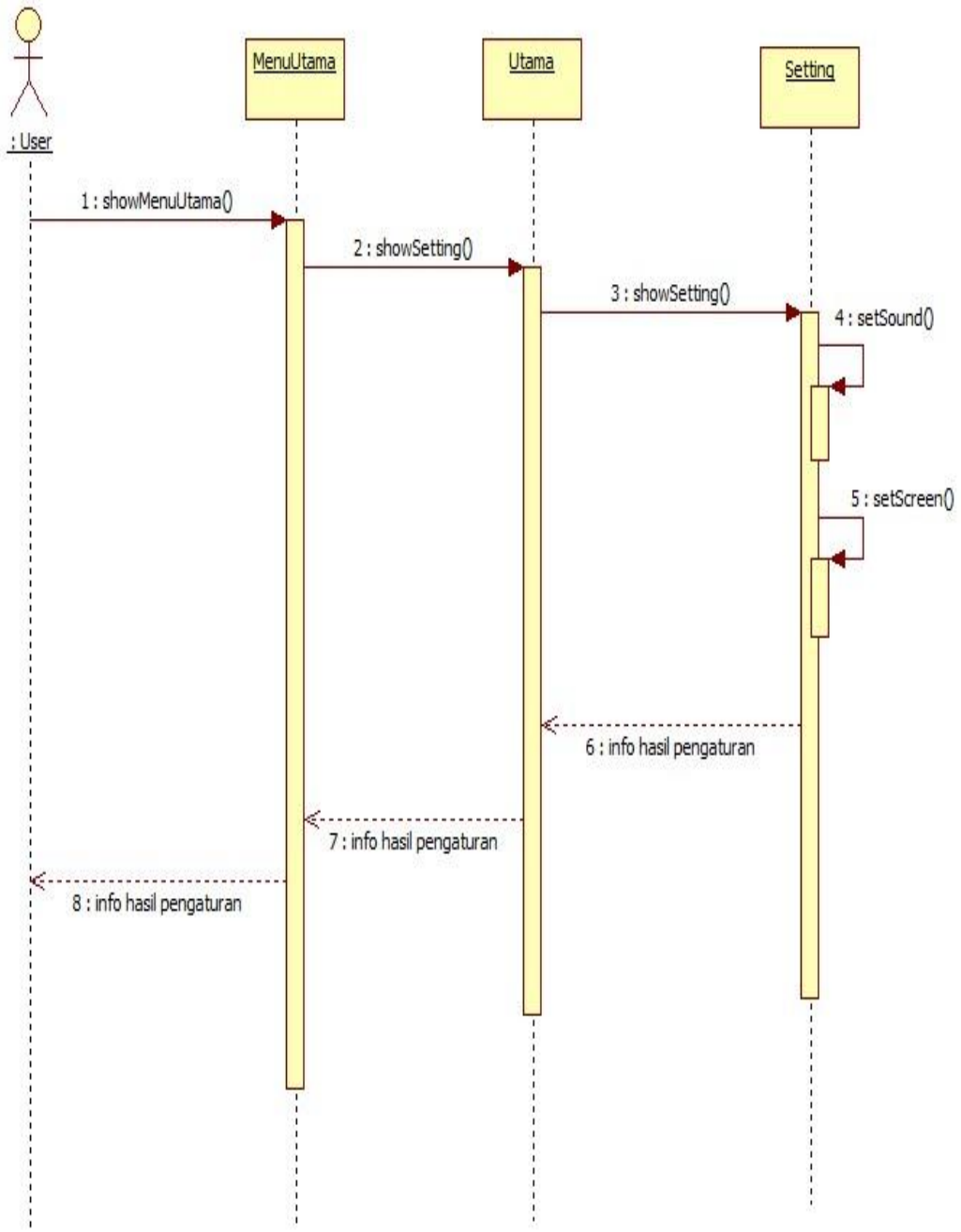
Gambar 3.20 *Sequence Diagram* Menampilkan Informasi Pembuat

4.2 *Sequence Diagram* Menampilkan Instruksi Permainan



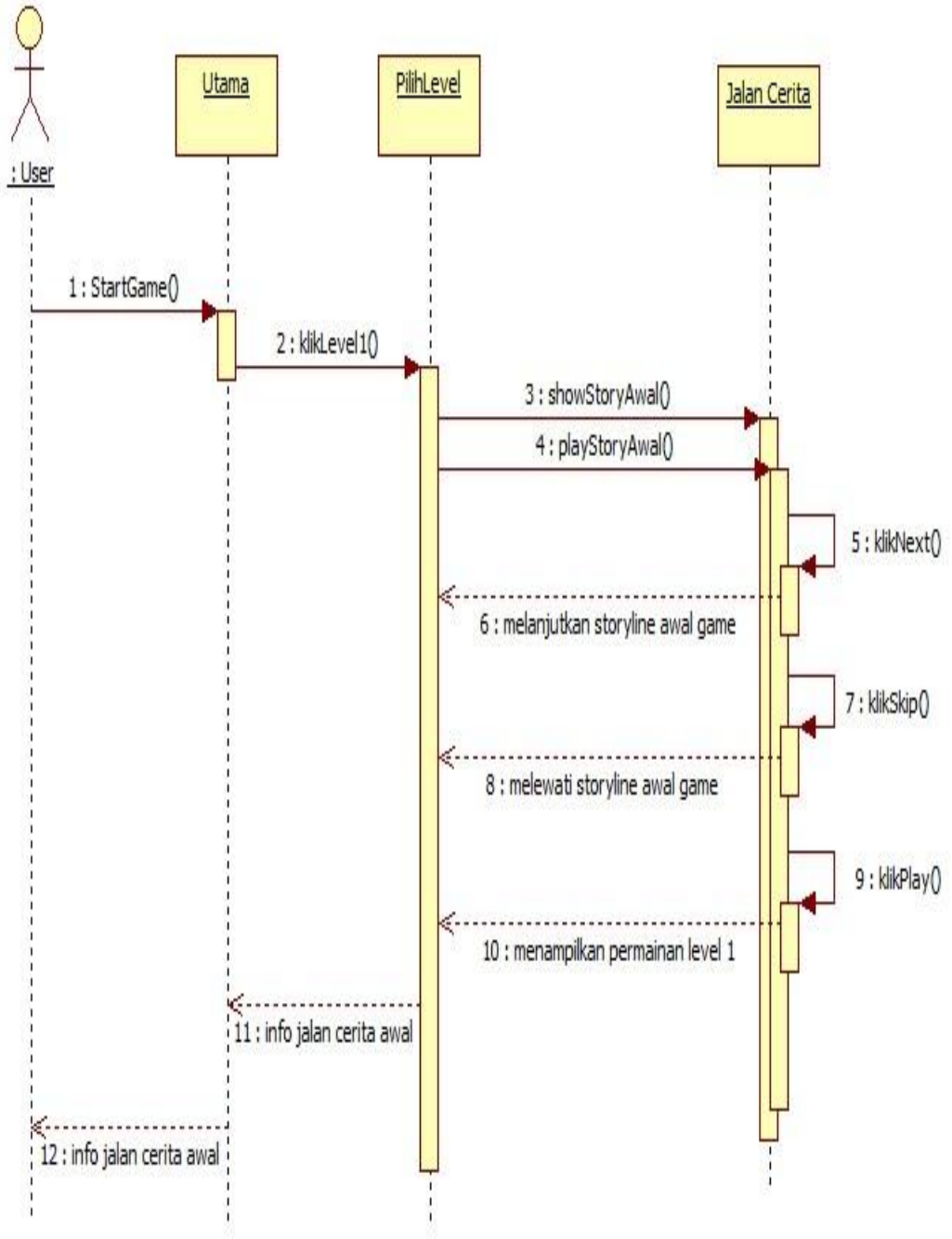
Gambar 3.21 *Sequence Diagram* Menampilkan Instruksi Permainan

4.3 *Sequence Diagram* Cek Pengaturan



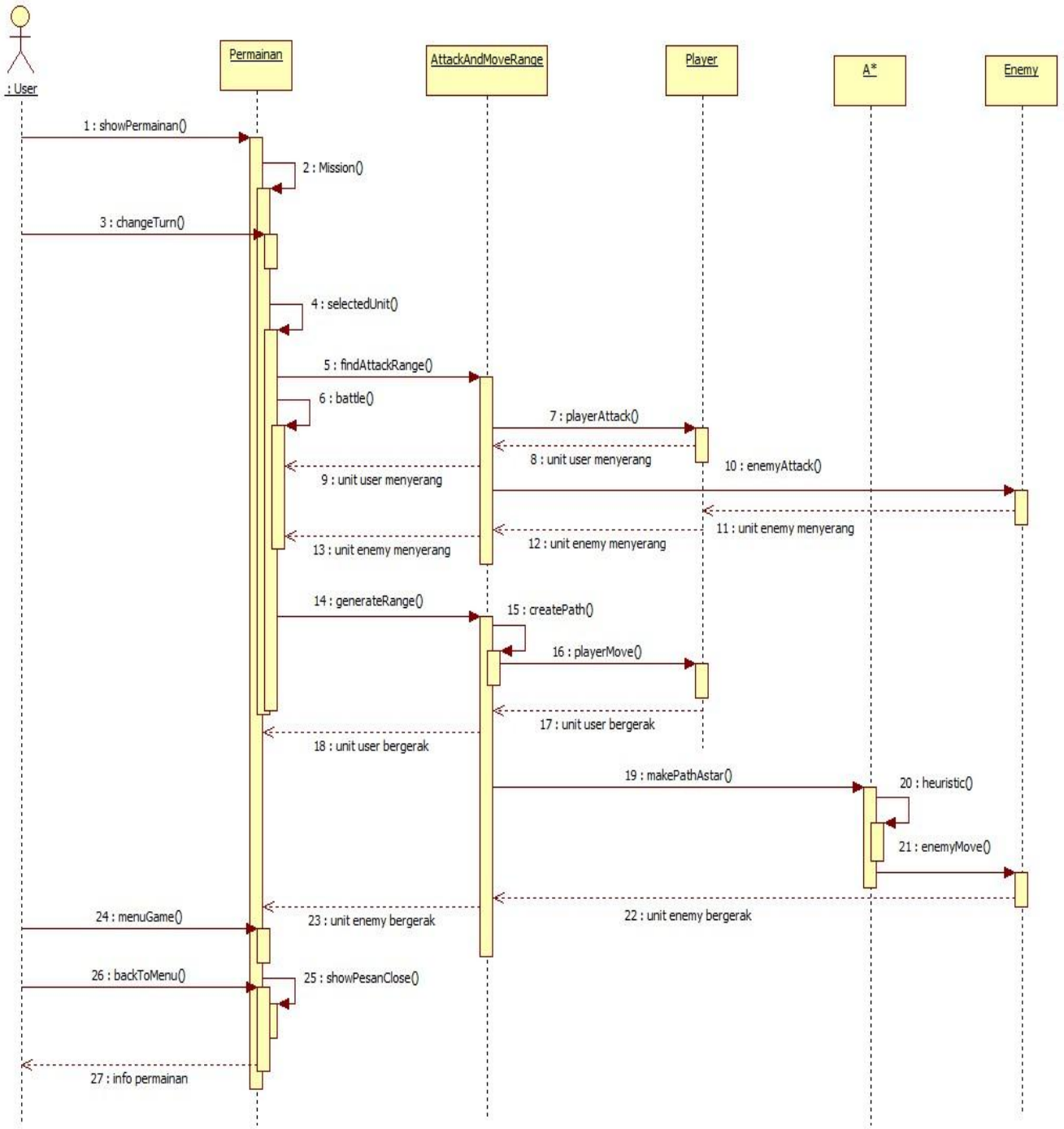
Gambar 3.22 *Sequence Diagram* Cek Pengaturan

4.4 *Sequence Diagram* Menampilkan *Storyline* Awal



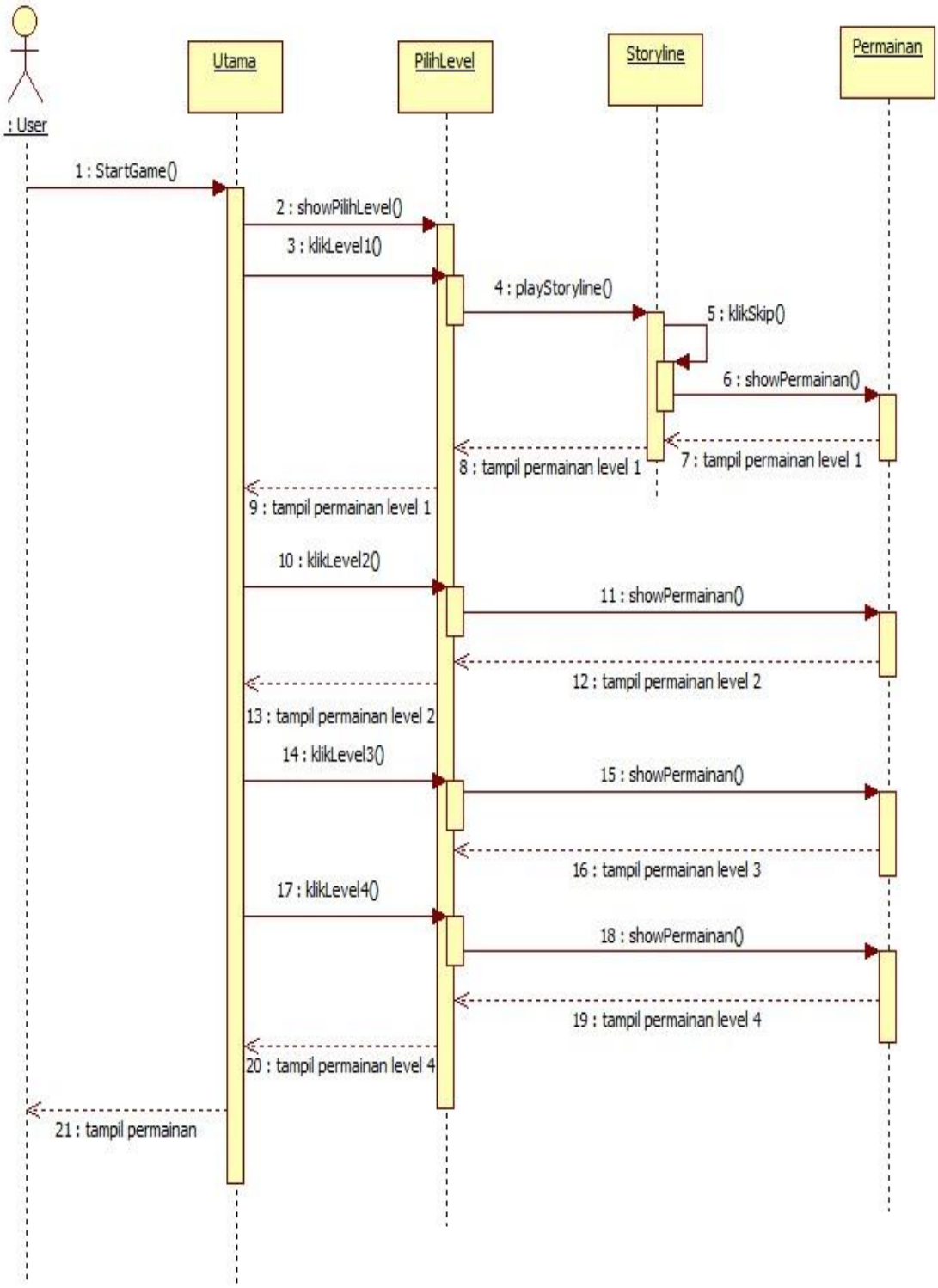
Gambar 3.23 *Sequence Diagram* Menampilkan *Storyline* Awal

4.5 *Sequence Diagram* Bermain



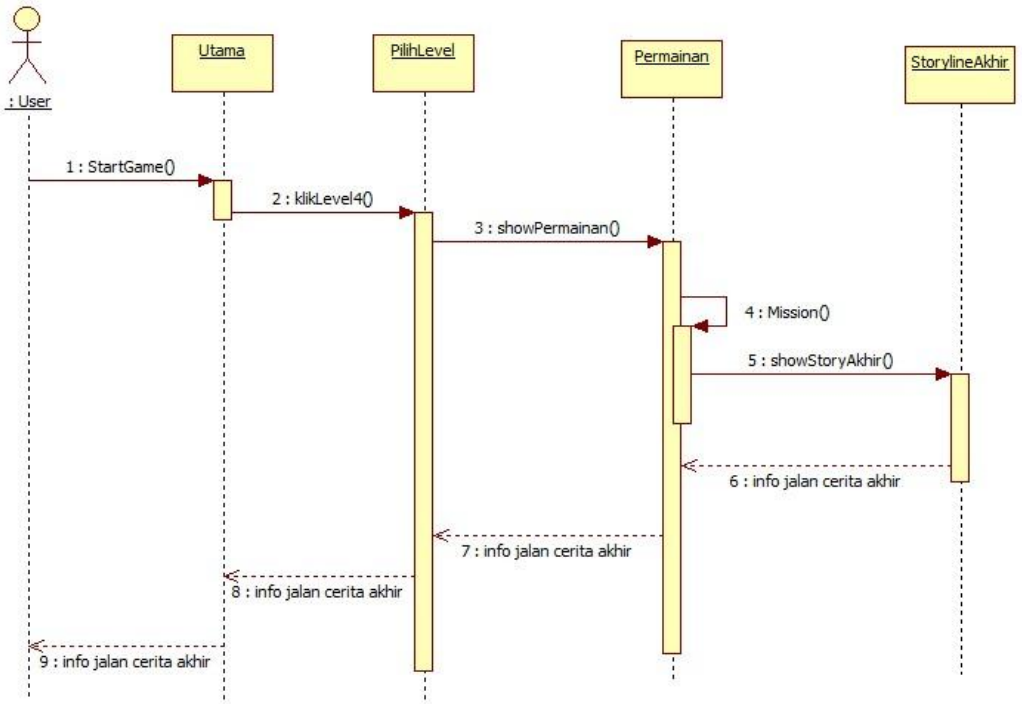
Gambar 3.24 *Sequence Diagram* Bermain

4.6 *Sequence Diagram* Pilih Level



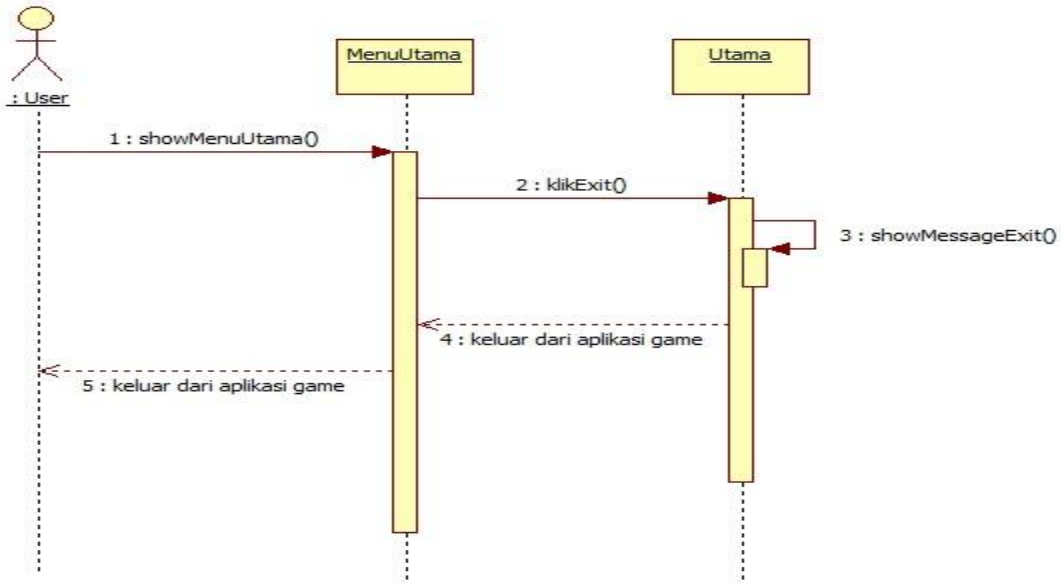
Gambar 3.25 Sequence Diagram Pilih Level

4.7 *Sequence Diagram Storyline* Akhir



Gambar 3.26 Sequence Diagram *Storyline* Akhir

4.8 *Sequence Diagram* Keluar



Gambar 3.27 *Sequence Diagram* Keluar

3.4 Perancangan

3.4.1 Pengenalan Karakter dan *Storyboard*

Storyboard adalah sketsa gambar yang disusun berurutan sesuai dengan naskah, dengan storyboard kita dapat menyampaikan ide cerita kita kepada orang lain dengan lebih mudah, karena kita dapat menggiring khayalan seseorang mengikuti gambar-gambar yang tersaji, sehingga menghasilkan persepsi yang sama pada ide cerita kita. Berikut merupakan storyboard dari Aplikasi *game* War of Baratayuda:

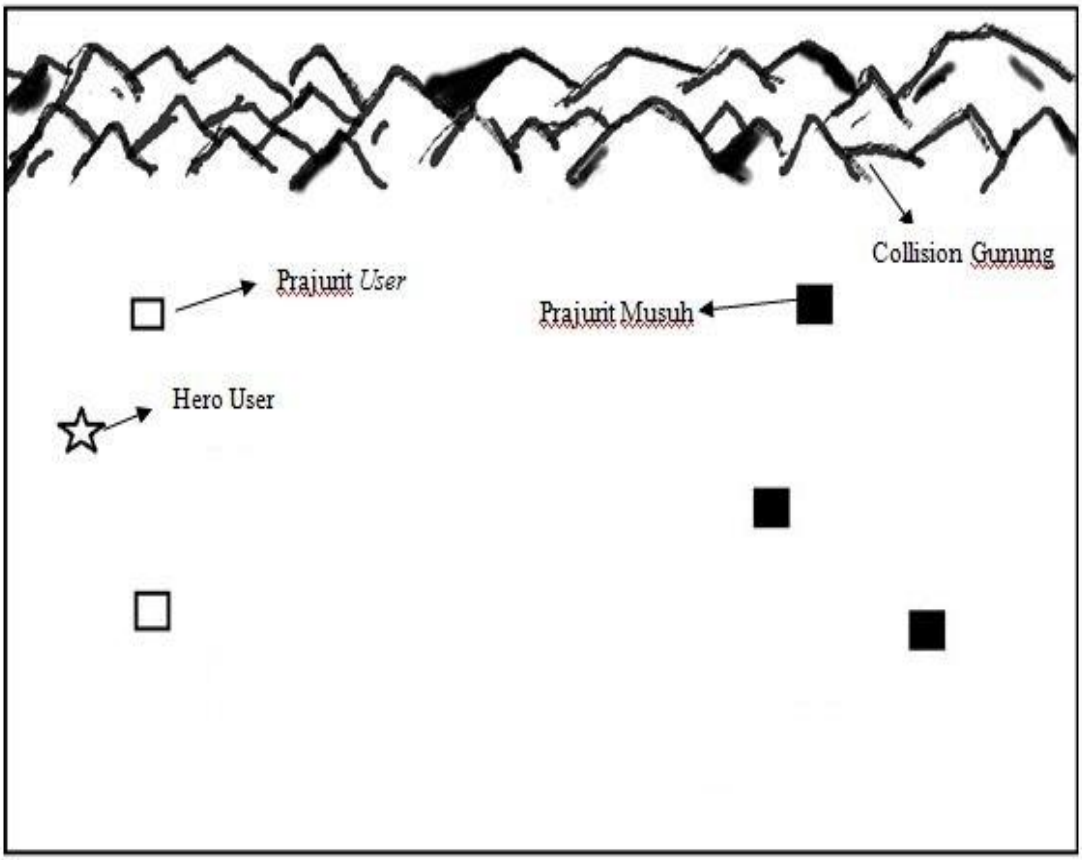
Tabel 3.11 Spesifikasi Karakter War of Baratayuda

No.	Simbol	Karakter	Deskripsi
1	□		Prajurit *user* dengan tipe serangan jarak dekat, memiliki area gerak sebanyak tiga langkah dan *hit damage* sebesar 20
2	■		Prajurit *enemy* (*Non Playable Character*) dengan tipe serangan jarak dekat, memiliki area gerak sebanyak tiga langkah dan *hit damage* sebesar 20
3	○		Pemanah *user*, memiliki serangan yang cukup kuat namun area gerak hanya satu langkah. *Hit damage* sebesar 30
4	●		Pemanah *enemy* (*Non Playable Character*), memiliki serangan yang cukup kuat namun area gerak hanya satu langkah. *Hit damage* sebesar 30
5	☆		Hero (Bima) merupakan karakter terkuat yang dimiliki *user*, memiliki area gerak sebanyak empat langkah dan *hit damage* sebesar 50

No.	Simbol	Karakter	Deskripsi
6			Hero (Duryudana) terkuat yang dimiliki *enemy* (*Non Playable Character*), memiliki area gerak sebanyak empat langkah dan *hit damage* sebesar 50

1. *Storyboard Level* 1

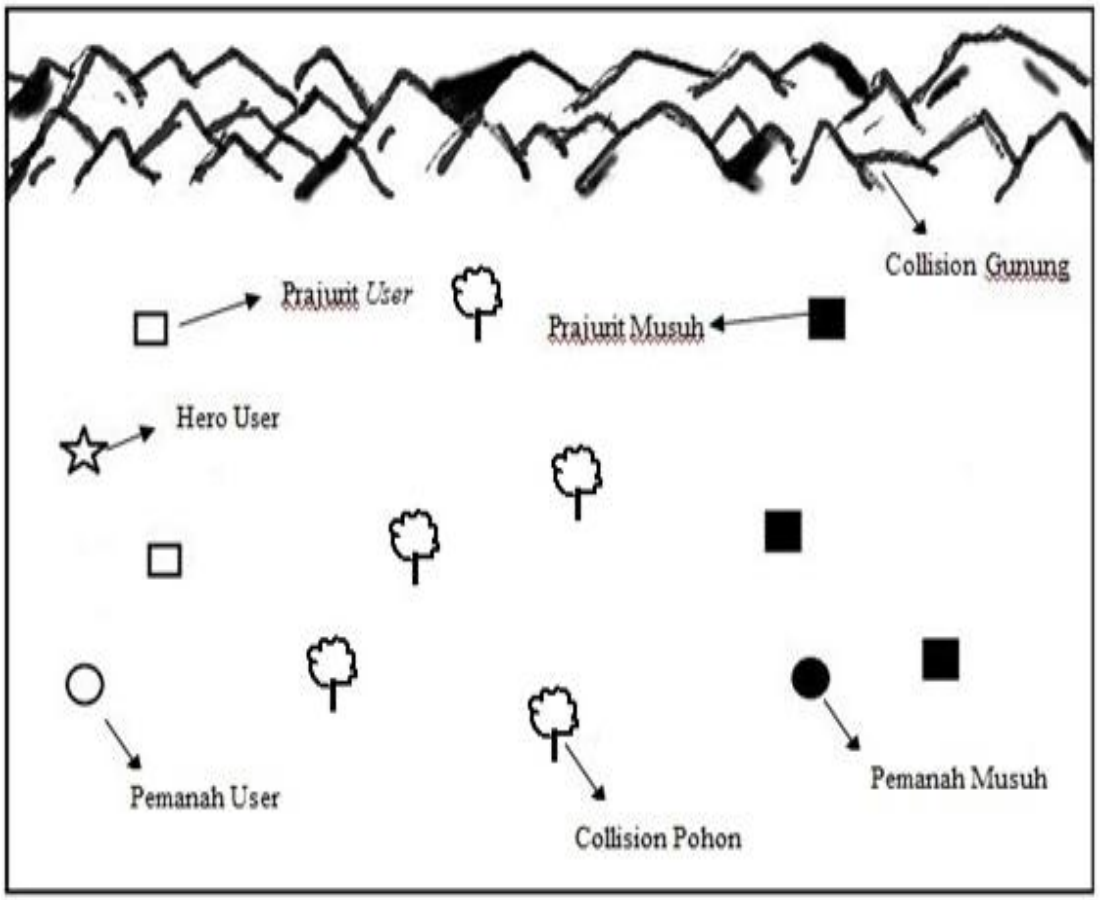
Pada Stage level 1, untuk *user* disediakan 2 unit prajurit dan 1 unit hero (berwarna putih). Sedangkan untuk *enemy (Non Playable Character)* disediakan 3 unit prajurit (berwarna hitam). *Collision* Gunung merupakan daerah yang tidak bisa ditempati oleh setiap pemain.



Gambar 3.28 *Storyboard* Level 1

2. *Storyboard Level* 2

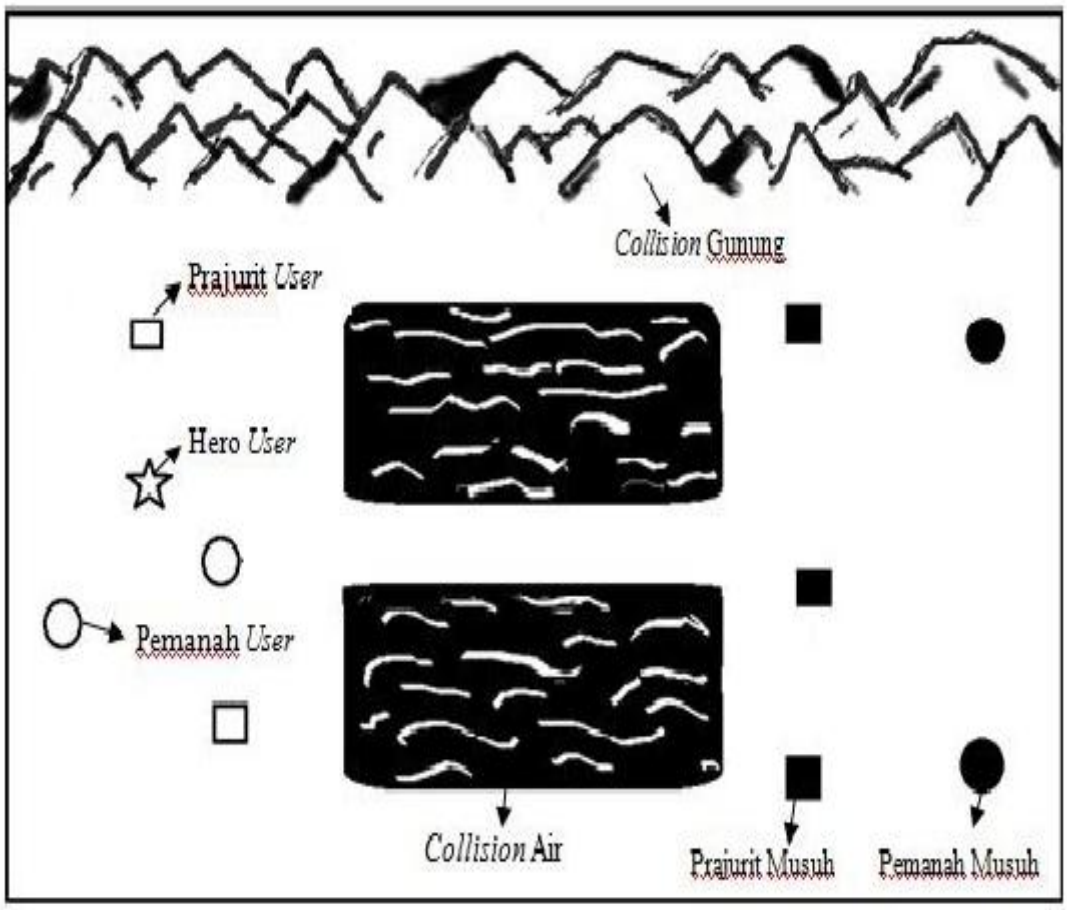
Pada Stage level 2, untuk *user* disediakan 2 unit prajurit, 1 unit pemanah dan 1 unit hero (berwarna putih). Sedangkan untuk *enemy (Non Playable Character)* disediakan 3 unit prajurit dan 2 unit pemanah (berwarna hitam). Di tengah-tengah permainan terdapat *unwalkable tile* (tempat yang tidak bisa ditempati oleh setiap pemain) berupa air (*Collision* Air) dan gunung (*Collision* Gunung).



Gambar 3.29 *Storyboard* Level 2

3. *Storyboard Level* 3

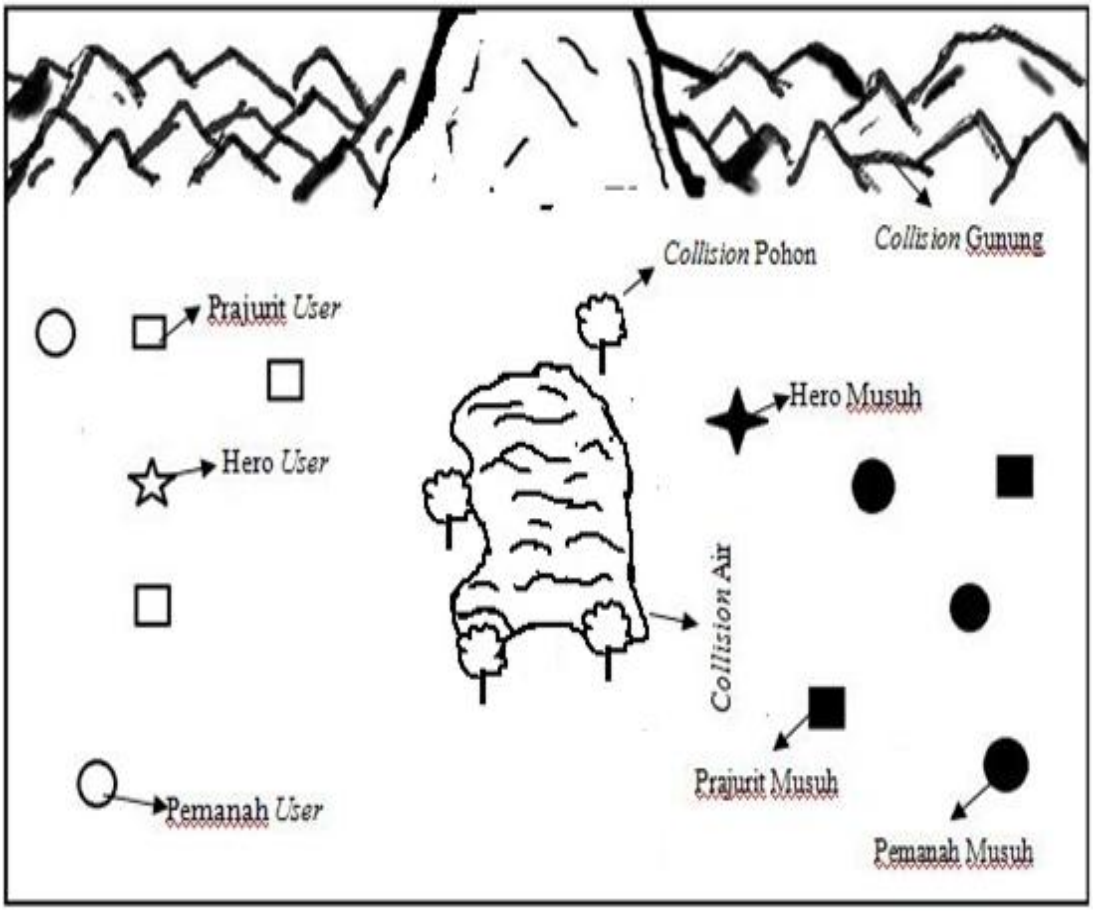
Pada Stage level 3, untuk *user* disediakan 2 unit prajurit, 2 unit pemanah dan 1 unit hero (berwarna putih). Sedangkan untuk *enemy (Non Playable Character)* disediakan 3 unit prajurit dan 2 unit pemanah (berwarna hitam). Di tengah-tengah permainan terdapat *unwalkable tile* (tempat yang tidak bisa ditempati oleh setiap pemain) berupa air (*Collision* Air) dan gunung (*Collision* Gunung).



Gambar 3.30 *Storyboard* Level 3

4. *Storyboard Level* 4

Pada Stage level 4, untuk *user* disediakan 3 unit prajurit, 2 unit pemanah dan 1 unit hero (berwarna putih). Sedangkan untuk *enemy (Non Playable Character)* disediakan 1 hero (berwarna hitam), 2 unit prajurit, dan 3 unit pemanah (berwarna hitam). Di tengah-tengah permainan terdapat *unwalkable tile* (tempat yang tidak bisa ditempati oleh setiap pemain) berupa air (*Collision* Air), pohon (*Collision* Pohon) dan gunung (*Collision* Gunung).

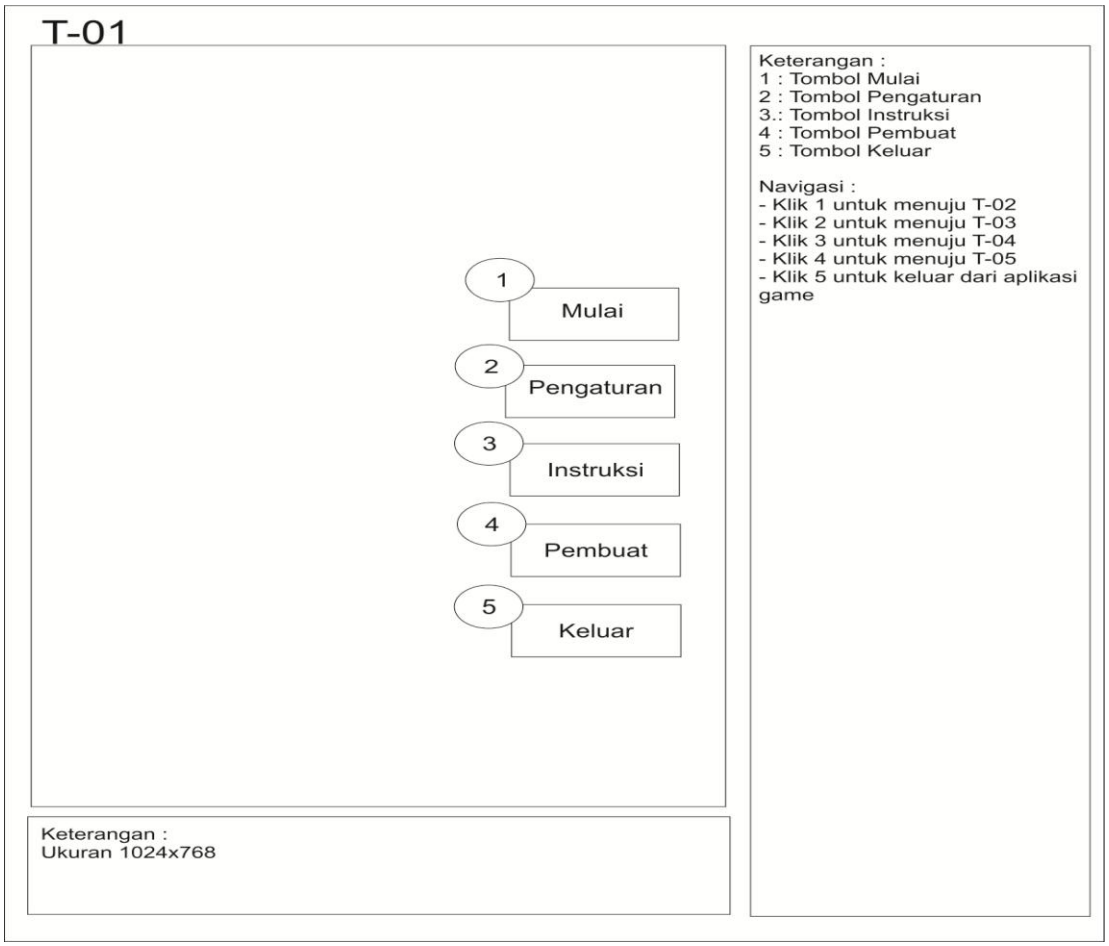


Gambar 3.31 *Storyboard* Level 4

3.4.2 Perancangan Antarmuka

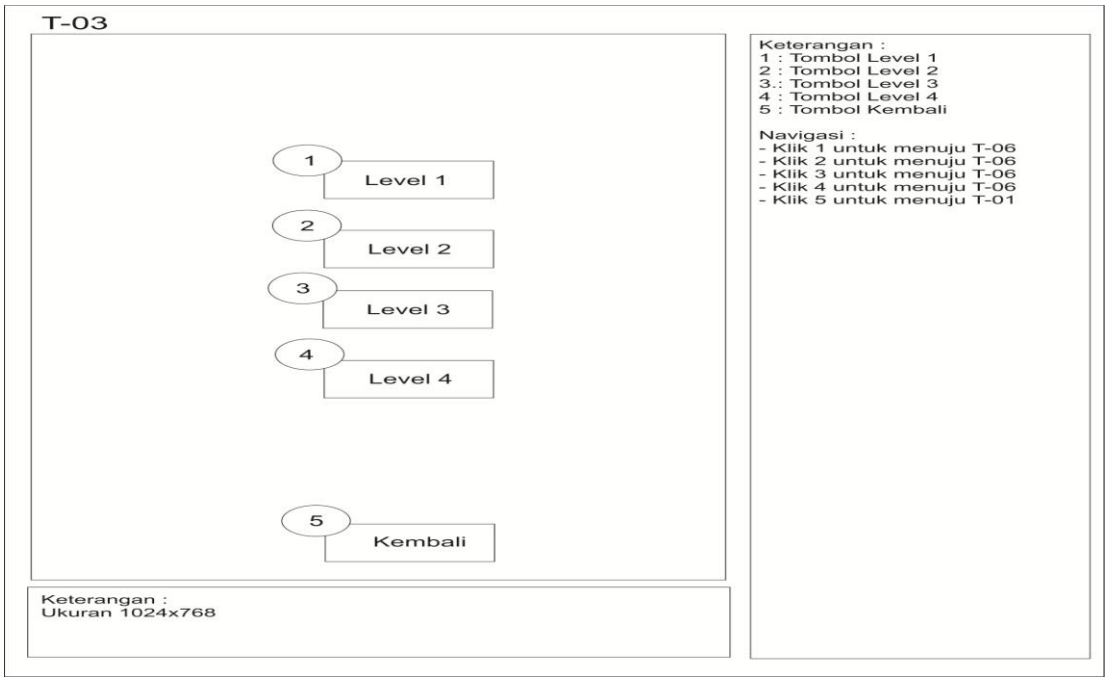
Perancangan antarmuka bertujuan untuk memberikan gambaran tentang aplikasi yang akan dibangun, sehingga akan mempermudah dalam mengimplementasikan aplikasi serta akan memudahkan pembuatan aplikasi.

3.4.2.1 Perancangan Antarmuka Menu Utama



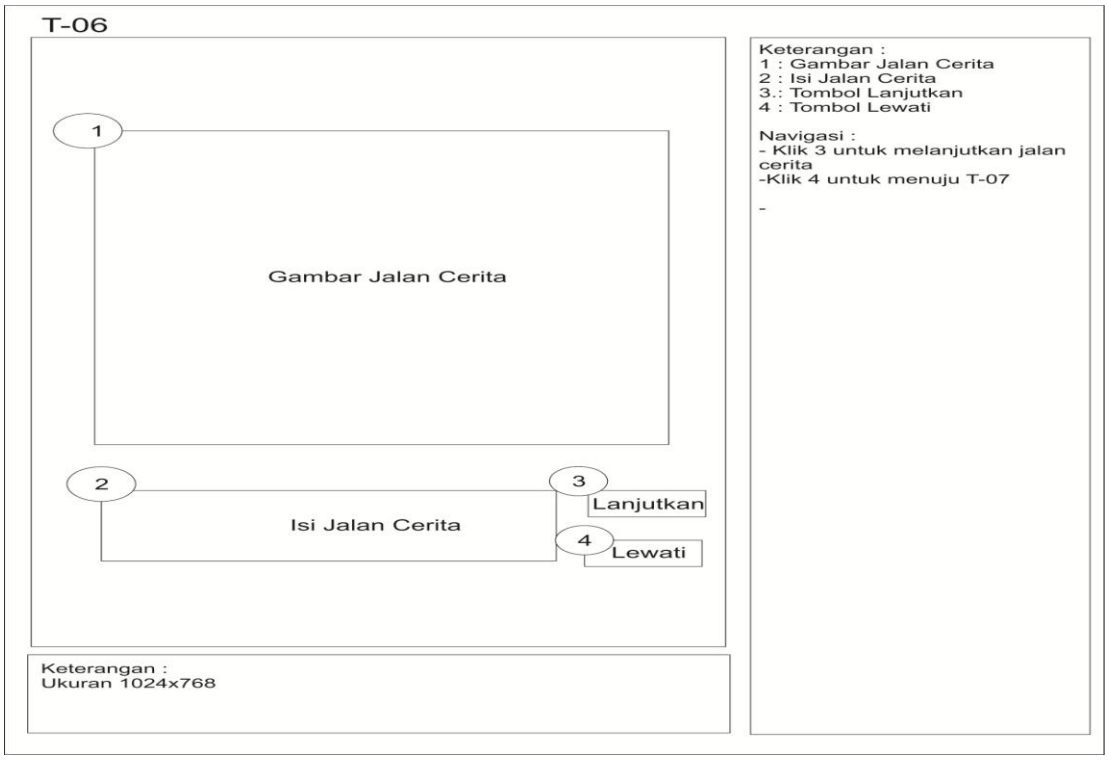
Gambar 3.32 Perancangan Antarmuka Menu Utama

3.4.2.2 Perancangan Antarmuka Pilih Level



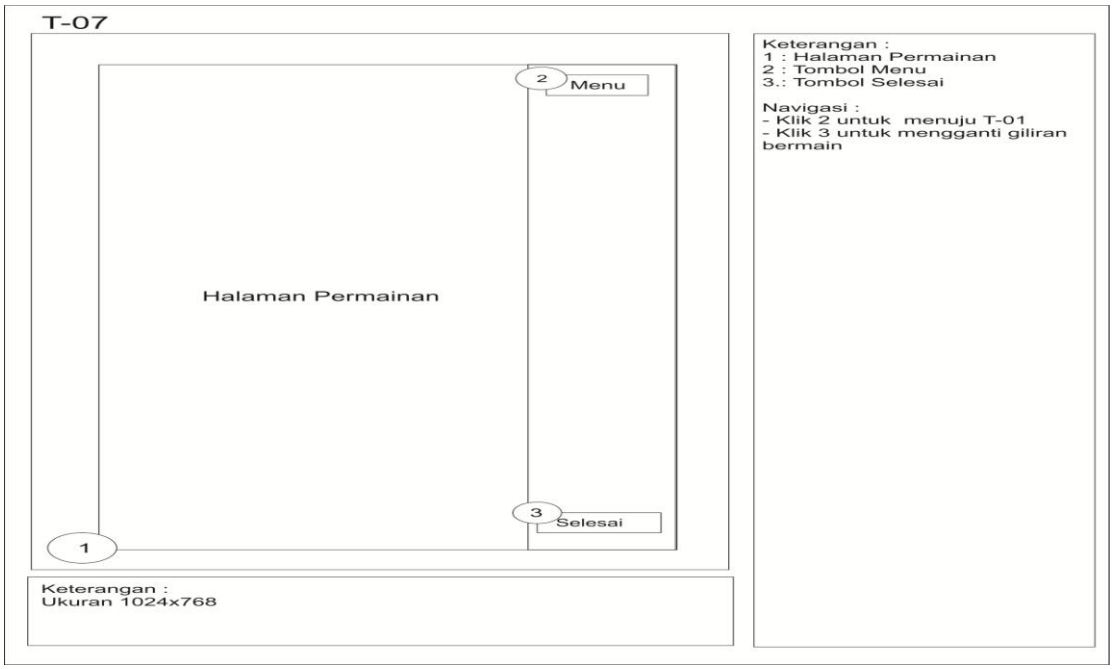
Gambar 3.33 Perancangan Antarmuka Pilih Level

3.4.2.3 Perancangan Antarmuka Jalan Cerita Awal



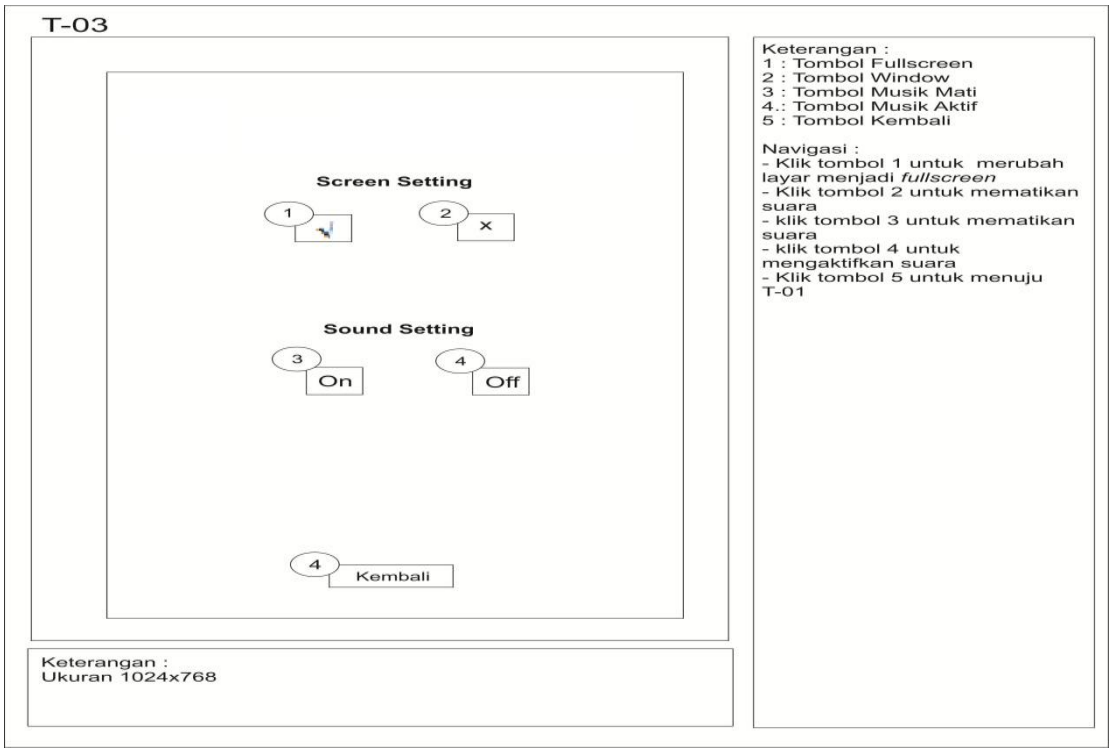
Gambar 3.34 Perancangan Antarmuka Jalan Cerita Awal

3.4.2.4 Perancangan Antarmuka Permainan



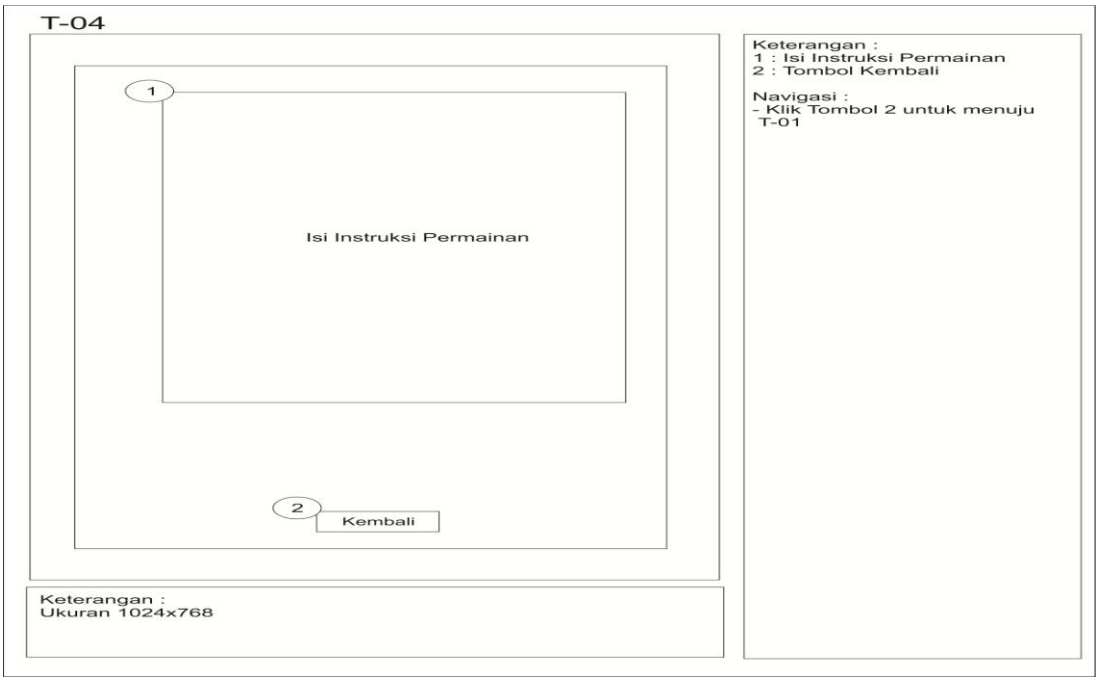
Gambar 3.35 Perancangan Antarmuka Permainan

3.4.2.5 Perancangan Antarmuka Pengaturan



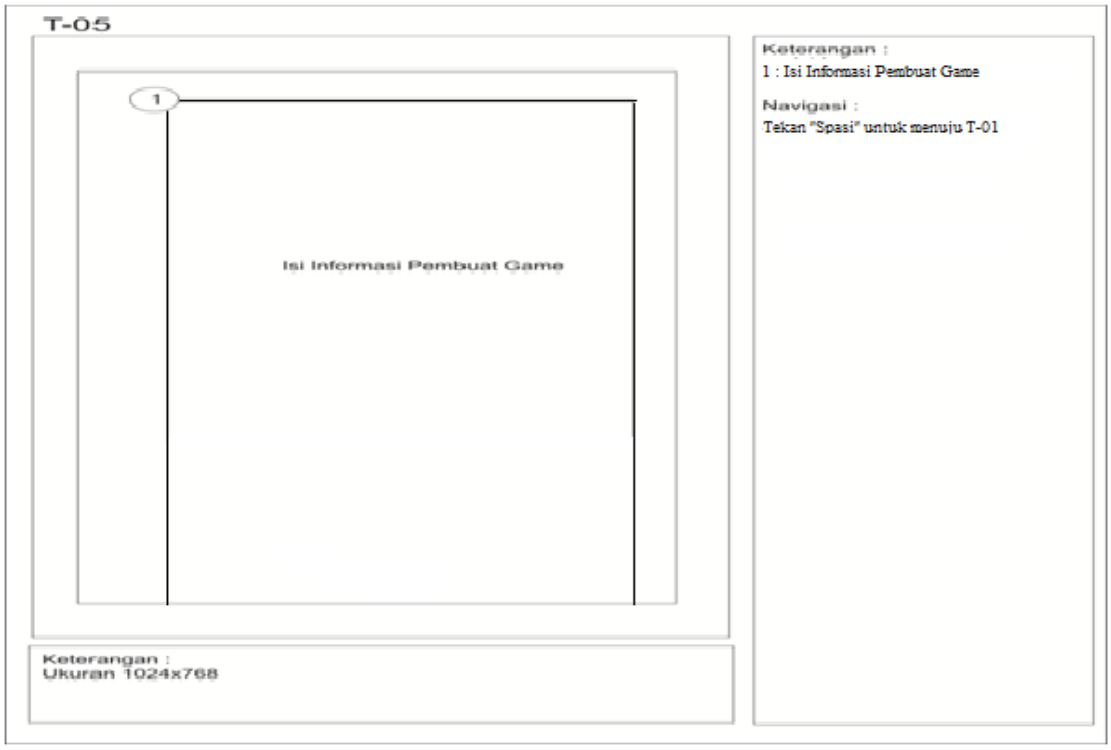
Gambar 3.36 Perancangan Antarmuka Pengaturan

3.4.2.6 Perancangan Antarmuka Instruksi



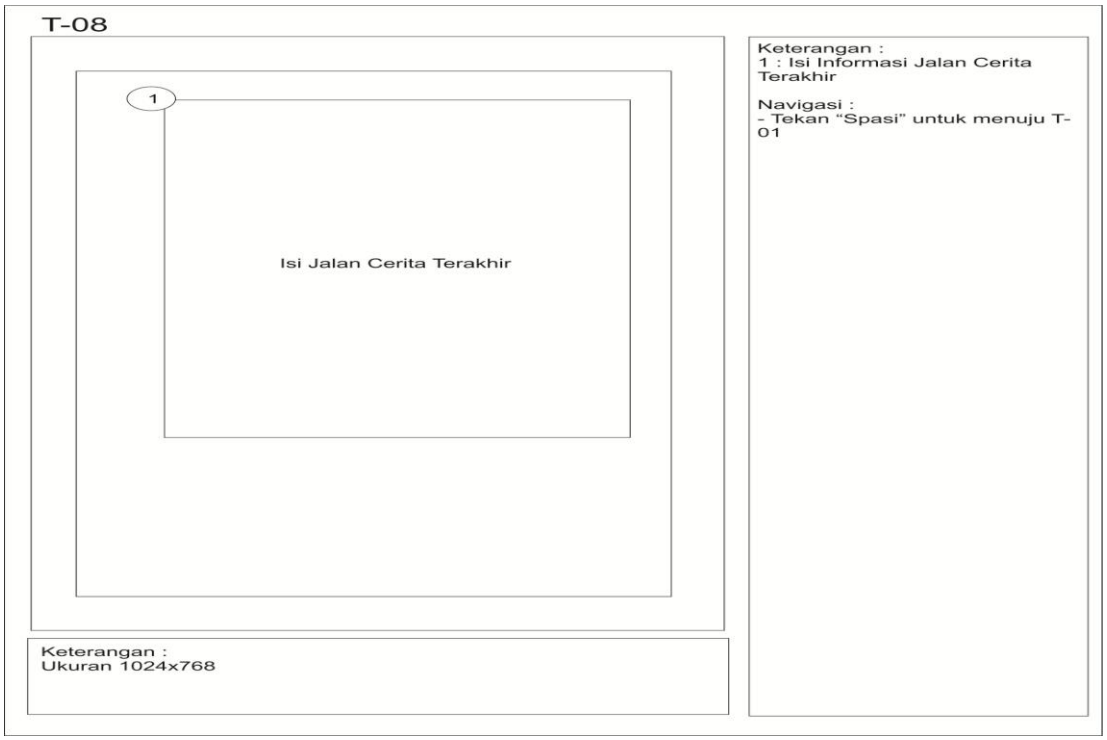
Gambar 3.37 Perancangan Antarmuka Instruksi

3.4.2.7 Perancangan Antarmuka Pembuat



Gambar 3.38 Perancangan Antarmuka Pembuat

3.4.2.8 Perancangan Antarmuka Jalan Cerita Terakhir

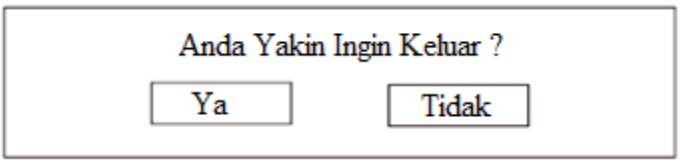


Gambar 3.39 Perancangan Antarmuka Jalan Cerita Terakhir

3.4.3 Perancangan Pesan

3.4.3.1 Perancangan Pesan Keluar dari Aplikasi

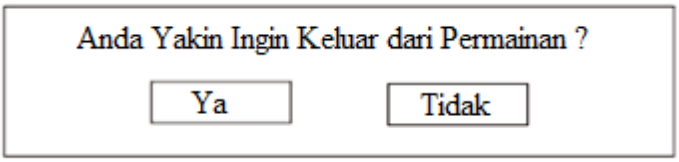
Pesan ini akan tampil ketika user memilih tombol "Keluar" pada menu utama untuk keluar dari aplikasi permainan.



Gambar 3.40 Perancangan Pesan Keluar dari Aplikasi

3.4.3.2 Perancangan Pesan Keluar dari Permainan

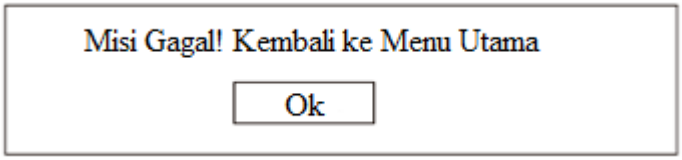
Pesan ini akan tampil ketika user memilih tombol Menu pada halaman permainan menuju halaman menu utama.



Gambar 3.41 Perancangan Pesan Keluar dari Permainan

3.4.3.3 Perancangan Pesan Kalah

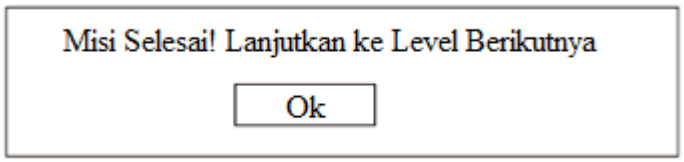
Pesan ini akan tampil ketika *user* kalah atau kehilangan semua *unit*-nya dalam permainan.



Gambar 3.42 Perancangan Pesan Kalah

3.4.3.4 Perancangan Pesan Misi Sukses

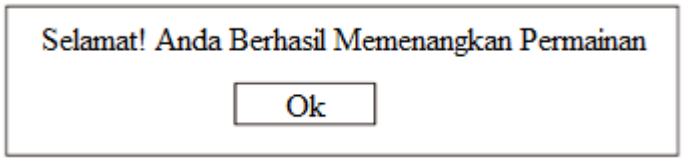
Pesan ini akan tampil ketika *user* menang atau menghancurkan semua *unit* musuh dalam permainan.



Gambar 3.43 Perancangan Pesan Misi Sukses

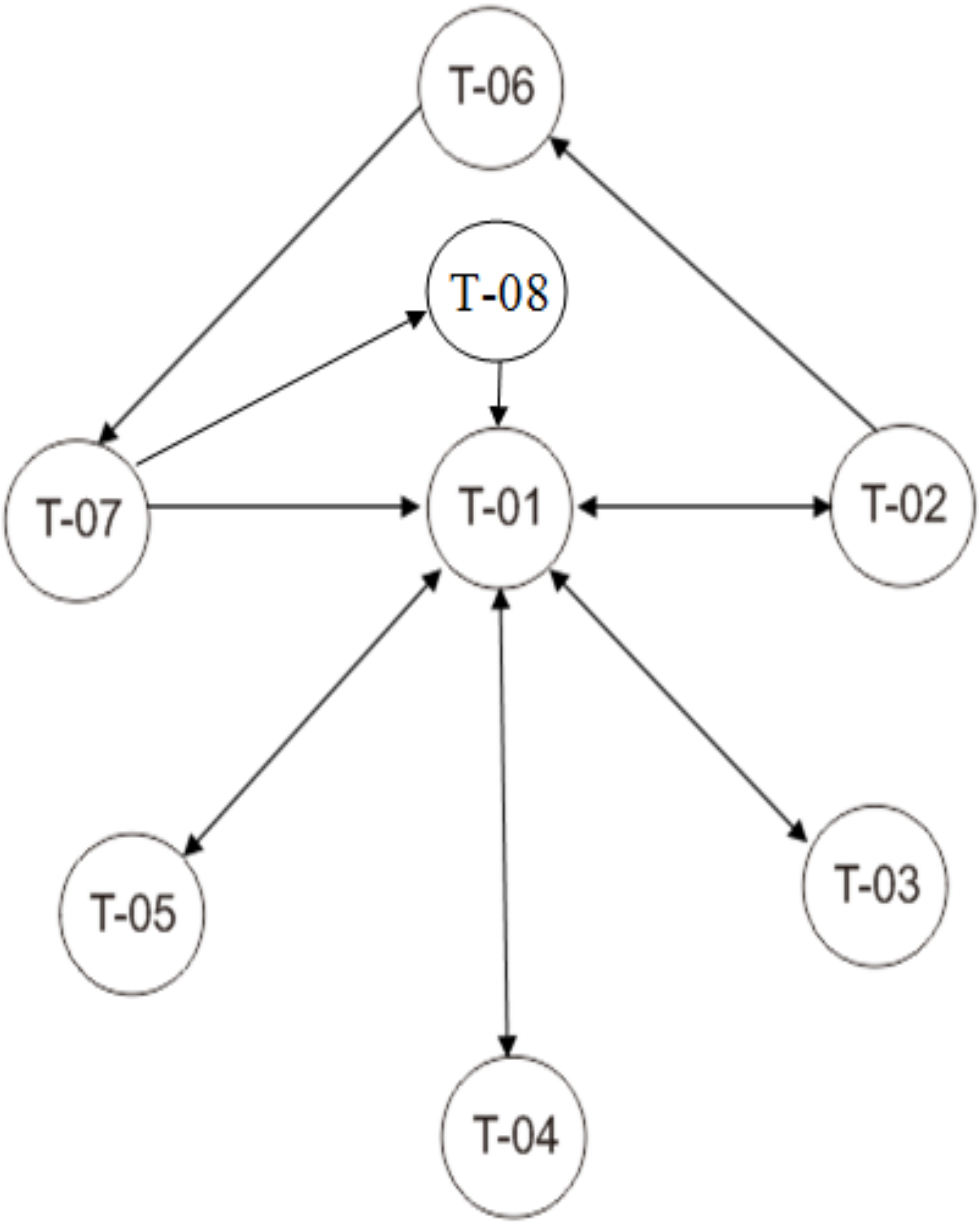
3.4.3.5 Perancangan Pesan Memenangkan Permainan

Pesan ini akan tampil ketika *user* memenangkan permainan pada level 4.



Gambar 3.44 Perancangan Pesan Memenangkan Permainan

3.4.4 Jaringan Semantik



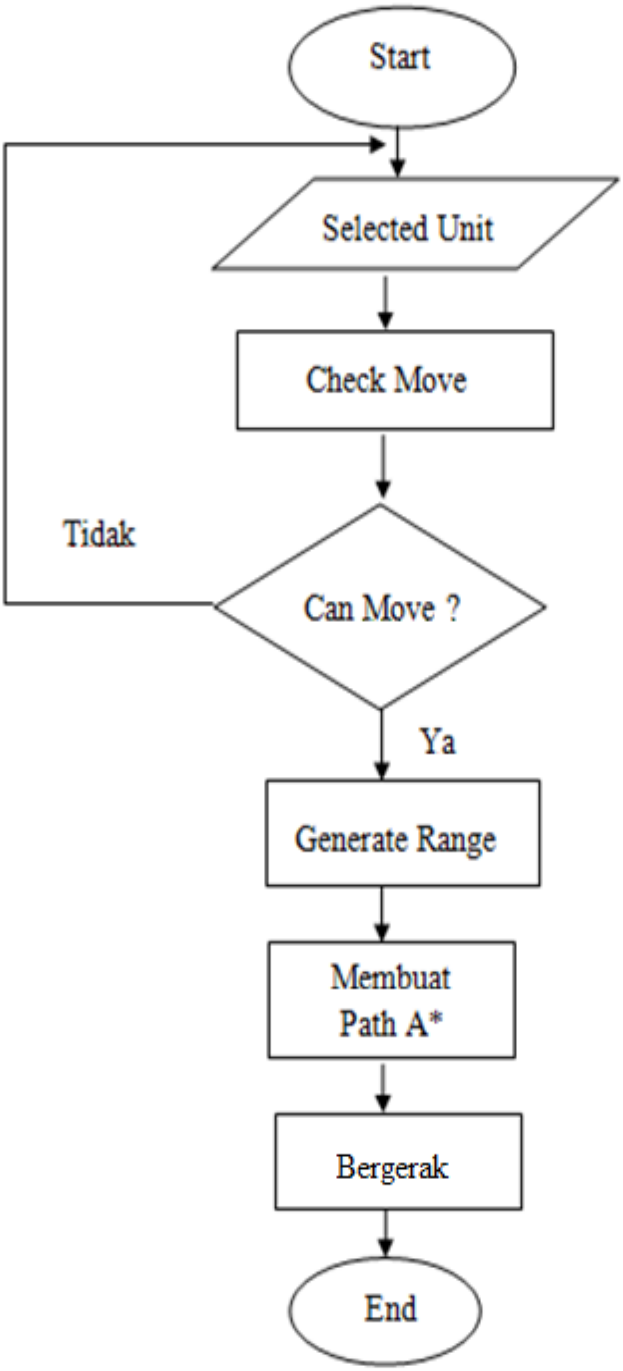
Gambar 3.45 Jaringan Semantik *Game* War of Baratayuda

3.4.5 Perancangan *Method*

3.4.5.1 Perancangan *Method enemyMove()*

Method ini digunakan disaat *user* memilih menu Selesai kemudian *enemy* akan melakukan aksi yaitu bergerak. Rincian prosesnya seperti berikut :

1. *Start*
2. *Selected Unit,* Unit yang akan bergerak
3. *Check* Move, lakukan pengecekan apakah unit dapat melakukan aksi bergerak
4. Jika *Can Move* = Ya, lanjutkan ke poin 5 jika tidak kembali ke *point* 2
5. *Generate Range,* Membuat jangkauan jarak bergerak (*move*) pada *Selected Unit*
6. Membuat *path* A*, Membuat jalan pada *Selected Unit* untuk bergerak
7. Bergerak, *Selected Unit* bergerak mengikut *path* A*
8. *End*

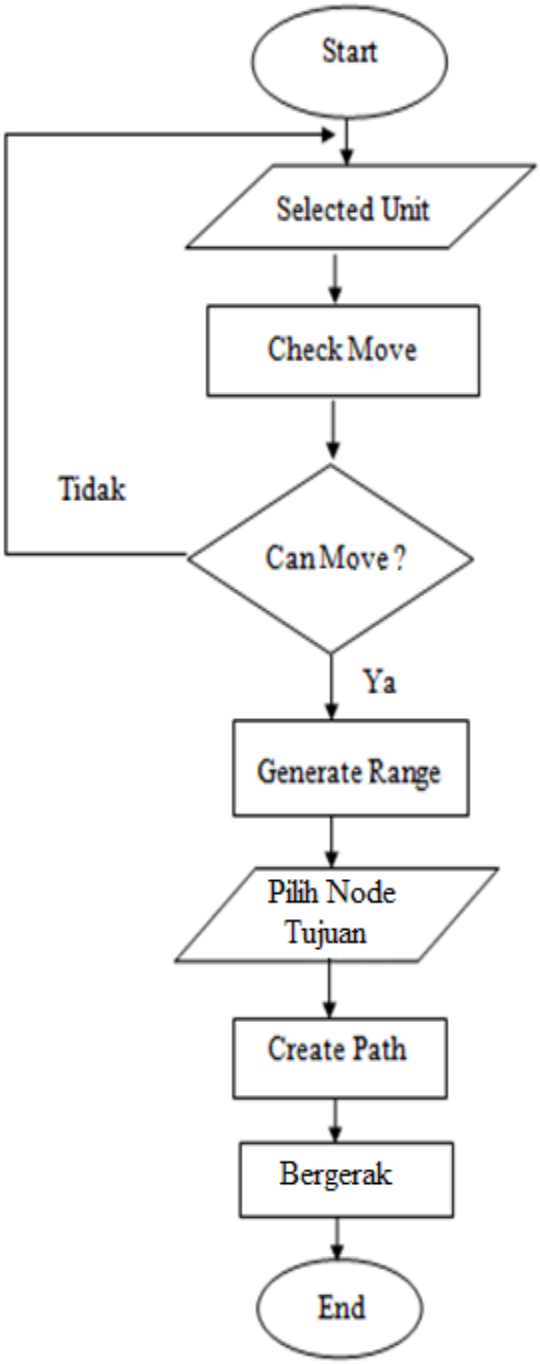


Gambar 3.46 Perancangan *Method enemyMove()*

3.4.5.2 Perancangan *Method playerMove()*

Method ini digunakan disaat pemain (*user*) akan melakukan aksi yaitu bergerak. Rincian prosesnya seperti berikut :

1. *Start*
2. *Selected Unit*, Unit yang akan bergerak
3. *Check* Move, lakukan pengecekan apakah unit dapat melakukan aksi bergerak
4. Jika *Can Move* = Ya, lanjutkan ke *point* 5 jika tidak kembali ke *point* 2
5. *Generate Range*, Membuat jangkauan jarak bergerak (*move*) pada *Selected Unit*
6. Pilih *Node* tujuan, memilih posisi yang diinginkan untuk berpindah
7. *Create Path*, membuat jalan dari posisi awal *Selected Unit* ke posisi tujuan
8. Bergerak, *Selected Unit* bergerak mengikuti *create path*
9. *End*

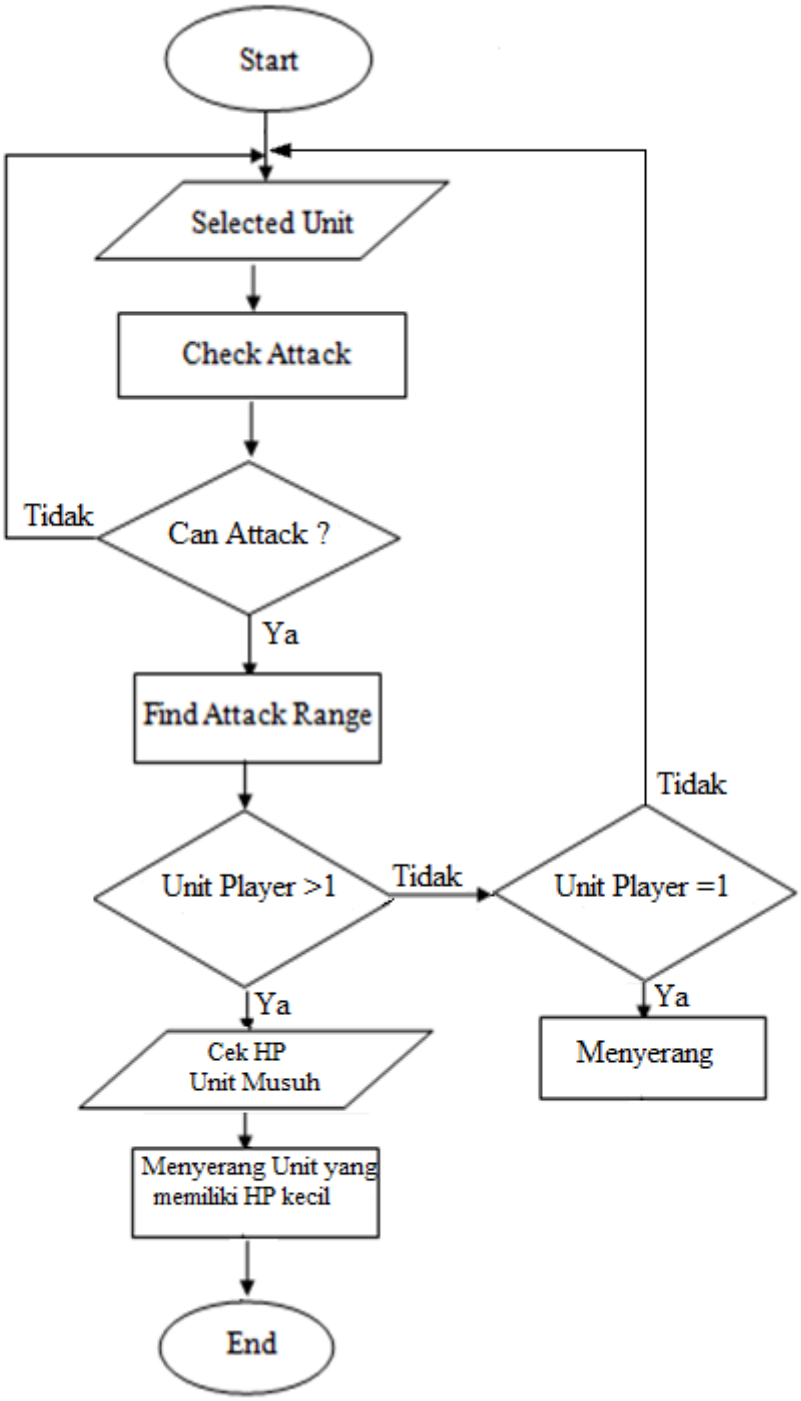


Gambar 3.47 Perancangan *Method playerMove()*

3.4.5.3 Perancangan *Method enemyAttack()*

Method ini digunakan disaat unit *enemy* akan melakukan aksi yaitu menyerang. Rincian prosesnya seperti berikut :

1. *Start*
2. *Selected Unit,* unit yang akan menyerang
3. *Check Attack,* Cek apakah unit dapat menyerang.
4. Jika *Can Attack* = ya lanjutkan ke point 5, jika tidak kembali ke *point* 2
5. *Find Attack Range*, Membuat jangkauan jarak serang pada unit yang akan menyerang
6. *Check* apakah jumlah unit *player* (unit yang akan diserang) pada jarak serang lebih dari 1. Jika Ya lanjutkan ke *poin*t 8, jika tidak lanjutkan ke *point* 7.
7. Lakukan pengecekan kembali jika unit *player* =1, *selected unit* menyerang. Jika tidak kembali ke *poin*t 2
8. Membaca dan melakukan pengecekan HP setiap unit *player* yang berada pada jarak serang
9. Menyerang unit *player* yang memiliki HP lebih kecil dari unit player lain. Jika unit *player* memiliki HP yang sama, *selected unit* menyerang unit *player* terdekat
10. *End*

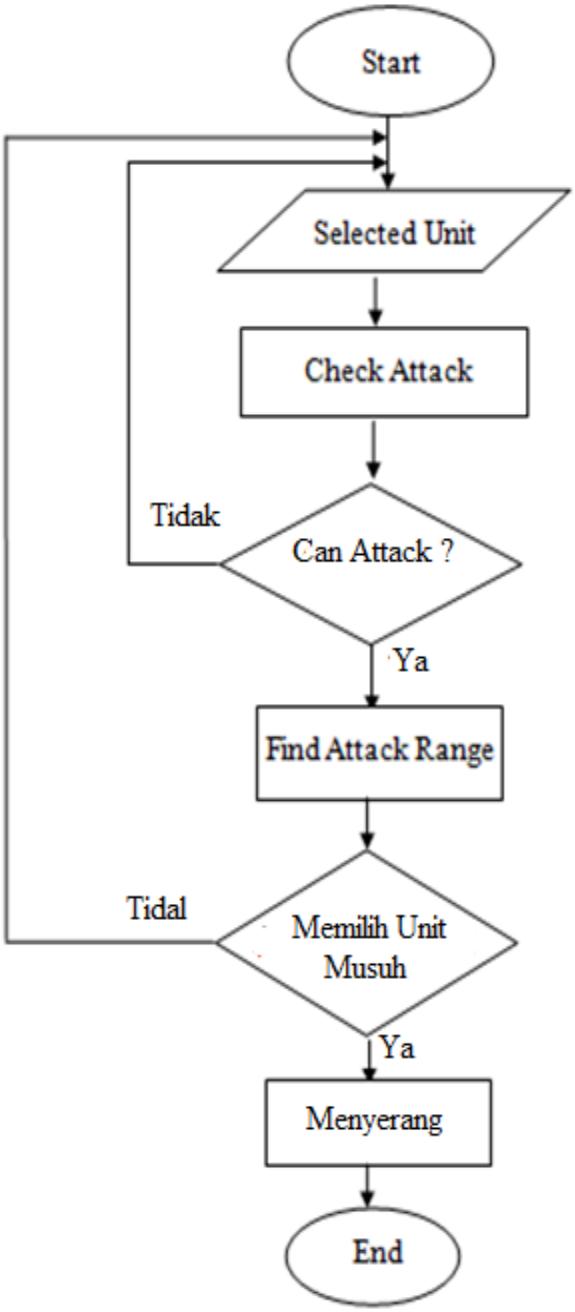


Gambar 3.48 Perancangan *Method enemyAttack()*

3.4.5.4 Perancangan *Method playerAttack()*

Method ini digunakan disaat unit *user* akan melakukan aksi yaitu menyerang. Rincian prosesnya seperti berikut :

1. *Start*
2. *Selected Unit*, unit yang akan menyerang
3. *Check Attack*, Cek apakah unit yang dipilih dapat menyerang.
4. Jika *Can Attack* = ya lanjutkan ke point 5, jika tidak kembali ke point 2
5. *Find Attack Range*, Membuat jangkauan jarak serang pada unit yang dipilih dan melakukan pengecekan
6. *Check* apakah *user* memilih unit yang akan diserang,
7. Jika ya maka selected unit akan menyerang
8. Jika tidak kembali ke point 2
9. *End*

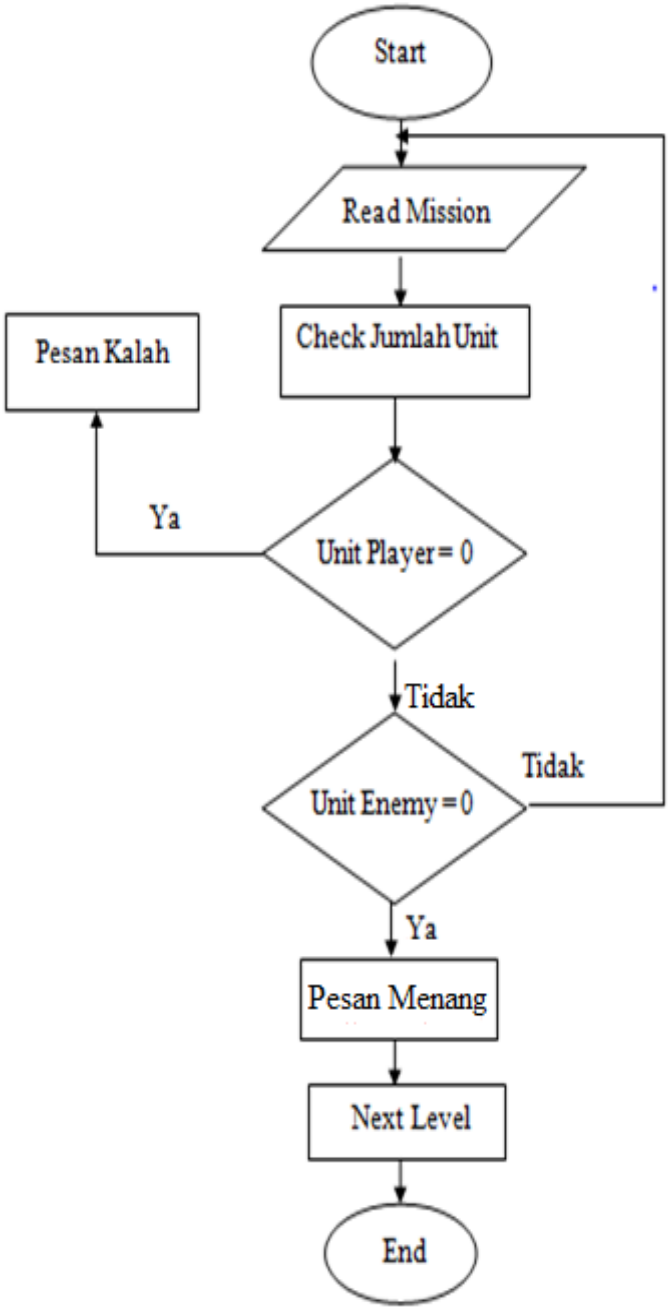


Gambar 3.49 Perancangan *Method playerAttack()*

3.4.5.5 Perancangan *Method mission()*

Method ini digunakan disaat unit *user* atau enemy akan melakukan aksi yaitu menyerang. Rincian prosesnya seperti berikut :

1. *Start*
2. *Read Mission*
3. *Check* Jumlah Unit, melakukan pengecekan terhadap unit *user* dan *enemy*
4. Jika *Unit Player* (*User*)=0 tampilkan pesan kalah, jika tidak lanjutkan ke poin 5
5. Jika Unit *Enemy*=0 lanjutkan ke poin 6, jika tidak kembali ke poin 2
6. Pesan Menang, menampilkan pesan bahwa misi telah berhasil dilakukan
7. *Next Level*, lanjutkan ke level berikutnya
8. *Check* apakah ada unit musuh, Melakukan pengecekan apakah terdapat unit lain (musuh) pada jarak serang.
9. *End*

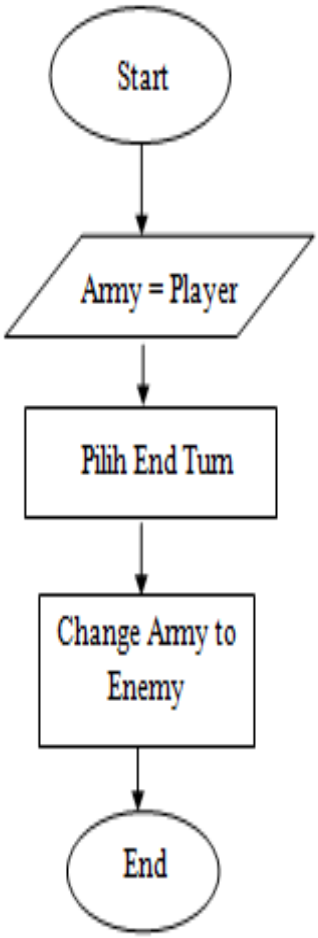


Gambar 3.50 Perancangan *Method mission()*

3.4.5.6 Perancangan *Method changeTurn()*

Method ini digunakan disaat *user* telah selesai melakukan aksinya. Rincian prosesnya seperti berikut :

1. *Start*
2. *Army = Player*, giliran berada pada *user*
3. Pilih *End Turn*, *user* memilih menu “End Turn”
4. *Change Army to Enemy*, mengganti giliran kepada *enemy*
5. *End*

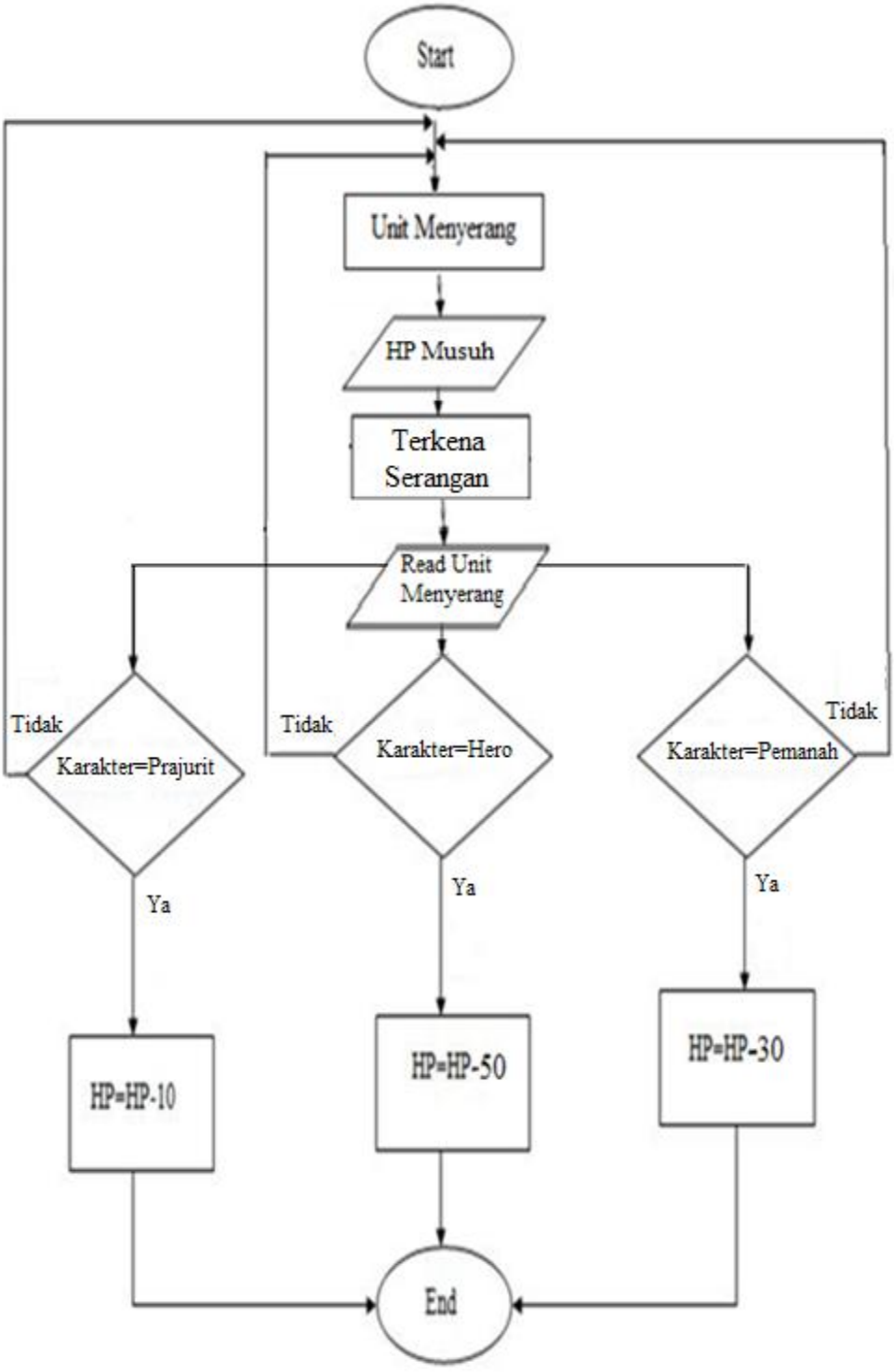


Gambar 3.51 Perancangan *Method changeTurn()*

3.4.5.7 Perancangan *Method Battle()*

Method ini digunakan ketika salah satu unit (*user* / *enemy*) melakukan aksi menyerang, kemudian akan terjadi pengurangan health bar terhadap unit yang diserang. Rincian prosesnya seperti berikut :

1. *Start*
2. Unit menyerang, Unit yang dimiliki *user* atau *enemy* melakukan aksi menyerang
3. *Read* HP, membaca HP (*Healt Bar)* unit yang akan diserang
4. Unit *enemy* atau *user* terkena serangan
5. *Read* unit menyerang, membaca tipe karakter unit yang akan menyerang
6. Jika Karakter yang menyerang adalah Hero maka HP karakter yang terkena serangan adalah HP=HP-30 (HP karakter dikurang 30), jika tidak kembali ke poin 2
7. Jika Karakter yang menyerang adalah Prajurit maka HP karakter yang terkena serangan adalah HP=HP-20 (HP karakter dikurang 10), jika tidak kembali ke poin 2
8. Jika Karakter yang menyerang adalah Pemanah maka HP karakter yang terkena serangan adalah HP=HP-20 (HP karakter dikurang 20), jika tidak kembali ke poin 2
9. *End*

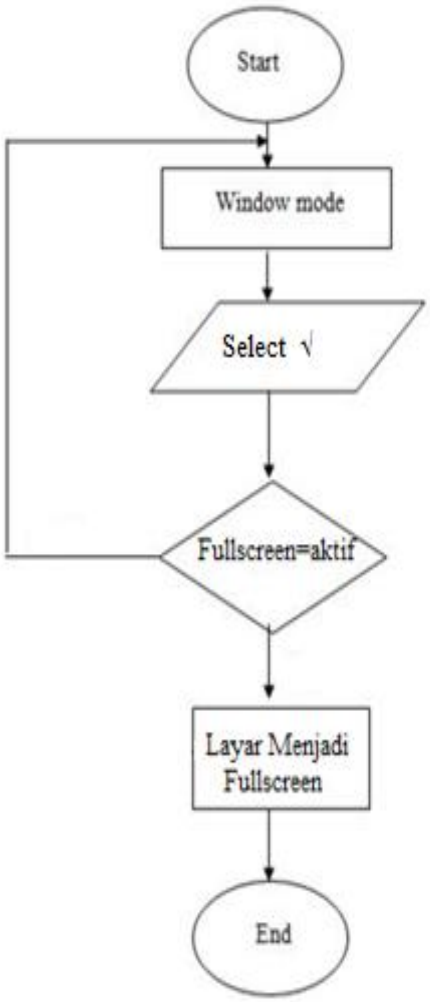


Gambar 3.52 Perancangan *Method Battle()*

3.4.5.8 Perancangan *Method setScreen()*

Method ini digunakan disaat *user* memilih menu Pengaturan dan mengatur pengaturan layar. Rincian prosesnya seperti berikut :

1. *Start*
2. *Window Mode* (layar tidak penuh / tidak *fullscreen)*
3. Pilih menu √ untuk mengubah layar menjadi *fullscreen*
4. Jika fullscreen = aktif, layar menjadi penuh *(fullscreen)*
5. Jika tidak kembali ke point 2
6. *End*

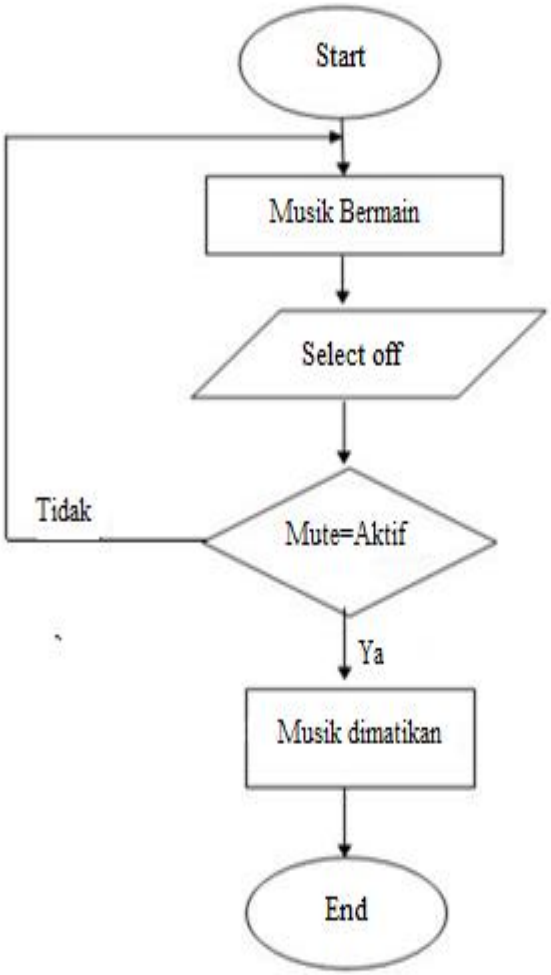


Gambar 3.53 Perancangan *Method setScreen()*

3.4.5.9 Perancangan *Method setSound()*

Method ini digunakan disaat *user* memilih menu Pengaturan dan mengatur pengaturan suara. Rincian prosesnya seperti berikut :

1. *Start*
2. Musik bermain, terdengar suara musik pada aplikasi
3. Pilih menu *Off*
4. Jika *Off* = aktif, musik dimatikan
5. Jika tidak kembali ke point 2
6. *End*



Gambar 3.54 Perancangan *Method setSound()*