# BAB III Analisa dan Perancangan

## 3.1 Analisa Masalah

Jarang sekali sebuah gimdengan *genre action adventure RPG* menerapkan sebuah algoritma *pathfinding.* Hal ini terjadi karena menerapkan algoritma *pathfinding* pada sebuah gimbukanlah hal yang mudah. Penerapan algoritma *pathfinding* tersebut harus disesuaikan dengan gim yang akan dibuat. Misalkan saja jika semua musuh yang ada di dalam gim menggunakan *pathfinding* sebelum melakukan pergerakan maka semua musuh akan bergerak melalui jalan terbaik menuju posisi aktor. Akibatnya adalah musuh akan mengepung aktor sehingga pemain akan kesulitan mengalahkan musuh karena mereka semua akan bergerak menuju posisi aktor dan menyerang aktor tanpa henti. Maka dari itu *pathfinding* pada gim harus disesuaikan dengan *gameplay* yang ada. Misalkan jika *NPC* musuh yang pertama berhasil dikalahkan oleh pemain makamusuh yang ke-2 baru akan melakukan *pathfinding* dan bergerak. Dengan kata lain penggunaan *pathfinding* pada gim akan membuat *gameplay* yang lebih baik dan lebih menantang, namun jika digunakan sembarangan penerapan *pathfinding* tersebut justru akan membuat gim tidak bisa dimainkan karena terlalu sulit.

Sebaliknya jika gim dibuat tanpa menggunakan *pathfinding* maka pergerakan *NPC* menjadi tidak efektif. Mereka tidak akan bisa menyesuaikan pergerakan mereka dengan pergerakan pemain yang lebih dinamis. Pemain seringkali memanfaatkan hal ini dengan membodohi *NPC* sehingga *NPC* menyangkut / *stuck*, berputar – putar di suatu tempat, atau mengambil jalan berputar untuk mencapai posisi aktor. Tidak peduli seberapa sulit atau tinggi *parameter* yang dimiliki musuh, jika mereka tidak dapat mencapai posisi pemain untuk melakukan serangan maka mereka dapat dengan mudah dikalahkan.

Untuk mengatasi permasalahan tersebut, algoritma *A star (A\*)* dapat diterapkan sebagai algoritma *pathfinding* untuk pergerakan *NPC.* Algoritma A\* dipilih karena algoritma ini dapat memperhitungkan seberapa jauh posisi antara *NPC* dengan pemain lalu memperhitungkan jalur mana yang memiliki jarak terpendek untuk mencapai posisi pemain. Jadi sebelum *NPC* melakukan pergerakan, mereka akan memperhitungkan terlebih dahulu jalur mana yang harus mereka ambil untuk mencapai posisi pemain. *NPC* tidak akan melakukan pergerakan secara

sia – sia apalagi *stuck* di suatu tempat. Dengan demikian, gimdengan *genre action adventure RPG* dapat menyajikan *gameplay* yang lebih baik dan menantang.

### 3.1.1 Identifikasi masalah

Identifikasi masalah yang ada pada penerapan algoritma A\* sebagai *pathfinding* untuk gim *action adventure RPG* membuktikan bahwa memadukan antara algoritma *pathfinding* dengan sebuah gimbukanlah hal yang mudah. Masalah :

* Sulitnya mengimplementasikan algoritma A\* sebagai *pathfinding* pada sebuah gim*.*
* Banyak pengembang yang tidak mengimplementasikan penggunaan algoritma *A\** sebagai *pathfinding* karena implementasi *pathfinding* pada gim sulit dilakukan dan sulit disesuaikan dengan *gameplay* yang ada.

Penyebab :

* Kurangnya pemahaman cara kerja algoritma A\*.
* Banyak pengembang yang menghiraukan permasalahan / *bug* yang terjadi pada gimyang mereka buat.

Dampak :

* *NPC* / musuh sering kali mengambil langkah yang sia – sia dan tidak efektif seperti bergerak menjauh dari pemain tanpa alasan, berputar – putar di suatu tempat, atau bahkan tidak dapat bergerak sama sekali / *stuck.*
* Kualitas gimmenjadi menurun dan membosankan karena musuh yang mengalami *stuck* tidak dapat berbuat apa – apa.

### 3.1.2 Usulan Pemecahan Masalah

Untuk memecahkan permasalahan yang sudah diuraikan di atas dapat dilakukan dengan mengimplementasikan algoritma *pathfinding* seperti algoritma A\* untuk pergerakan *NPC /* musuh. Dengan digunakannya algoritma ini, musuh dapat menemukan posisi aktor di mana pun posisi tersebut berada dengan catatan bahwa ada jalan atau alur yang dapat dilalui musuh untuk mencapai posisi aktor.

### 3.1.3 Kelebihan Usulan Pemecahan Masalah

Terdapat keuntungan terhadap pemecahan masalah yang ada di atas seperti :

*Gameplay* menjadi lebih baik dan menantang karena musuh dapat beradaptasi dengan pergerakan pemain sehingga mereka tidak akan *stuck* di suatu tempat.

*Map design* dapat dibuat dengan lebih kompleks seperti labirin karena tidak peduli seberapa rumit *map* dibuat, musuh pasti dapat menemukan posisi pemain selama ada jalan untuk mencapai posisi pemain tersebut.

*Enemy* menjadi lebih sulit untuk dikalahkan karena mereka bergerak dengan cepat dan efektif untuk mendekati posisi pemain dan melakukan serangan.

*Enemy type* dapat dibuat lebih banyak seperti jenis musuh yang dibedakan berdasarkan pergerakannya ataupun kecepatan geraknya.

## 3.2 Perancangan Gim

Pada tahap ini akan dijelaskan struktur perancangan gim meliputi konsep gim, *gameplay* pada gim, perancangan desain gim seperti UI, dan cerita utama dalam gim.

### 3.2.1 Gambaran Umum

Konsep pembuatan gim akan ditunjukan pada tabel berikut :

###### Tabel 3.1 Gambaran Umum Desain Gim

|  |  |  |
| --- | --- | --- |
| **No** | **Konsep** | **Keterangan** |
| 1 | Title | Forsena |
| 2 | Game Genre | *Action, Adventur, RPG* |
| 3 | Game Type | *Science-Fictional (Sci-Fi)* |
| 4 | Language | Bahasa Indonesia dengan beberapa nama barang / sihir menggunakan bahasa Inggris. |
| 5 | Goal of Game | Tujuan utama dari gim adalah mengalahkan musuh bos terakhir dari kisah gim yang ada. Di sepanjang perjalan pemain harus menyelesaikan misi – misi tertentu untuk melanjutkan alur cerita yang ada. |
| 6 | Winnable | Kalahkan musuh dengan menyerangnya menggunakan sihir, kemampuan, atau senjata sampai nyawa musuh habis. |
| 7 | Platform | Desktop dengan *OS windows.* Gim dapat dimainkan pada PC / laptop dengan *OS windows.* |
| 8 | Price | 3$ atau sekitar Rp 43.000 |

Gambaran umum dari gim diperuntukan agar pemain mengerti gim apa yang akan mereka mainkan sebelum membeli dan menginstalnya.

### 3.2.2 Design Elemen of Game/Content

Seperti yang sudah dijelaskan di atas, ada 11 elemen gim yang diperlukan untuk membuat gim yang baik. Ke 11 elemen tersebut adalah :

1. Format : Gim yang dibuat berbentuk action adventure RPG sehingga pemain dalam melakukan serangan secara langsung pada musuh yang ada pada *map*.

2. Rules : Jika HP (Hit point) musuh berkurang sampai 0 maka musuh akan kalah. Namun jika HP pemain berkurang sampai 0 maka pemain yang kalah sehingga terjadi game over.

3. Policy : Pemain dapat mengganti perlengkapan, senjata, ataupun sihir sesuai dengan keinginan pemain.

4. Scenario : Alur cerita pada gim mengisahkan seorang prajurit bayaran yang dijadikan kambing hitam oleh orang jahat.

5. Event : Event yang ada pada gim akan dibuat di mana tokoh utama harus menolong NPC yang mendapatkan masalah. Event ini ditujukan agar tokoh utama semakin kuat atau mendapatkan perlengkapan / sihir baru.

6. Roles : Peran dari gim sebagian besar dibagi menjadi 2 yaitu prajurit bayaran yang berperan sebagai tokoh utama dan raja yang jahat akan berperan sebagai musuh utama.

7. Decision : Terdapat beberapa pengambilan keputusan yang harus dilakukan pemain di dalam gim. Keputusan yang dibuat seperti ‘ya’ atau ‘tidak’. Keputusan yang dibuat akan mempengaruhi alur cerita,

8. Level : Semakin tinggi level musuh yang ada, semakin sulit musuh tersebut dikalahkan. Musuh tersebut akan memiliki daya serang yang tinggi dan HP yang banyak.

9. Score model : Score model akan menampilkan musuh apa saja yang berhasil dikalahkan oleh pemain, event apa saja yang sudah diselesaikan, dan berapa lama pemain memainkan gim tersebut.

10. Indicators : Indikator dibuat agar pemain terbiasa dengan mekanisme yang ada di dalam gim. Indikator akan dibuat supaya pemain mencoba semua senjata, perlengkapan / sihir yang ada di gim. Contoh indikator adalah kalahkan 100 musuh, tangkis 50 serangan musuh, gunakan bom 100 kali, dan lain sebagainya.

11. Symbols : Penataan simbol pada gim akan dibuat sederhana sehingga bagi pemain yang awam, mereka dapat langsung memahami makna dari simbol tersebut. Simbol dengan gambar bom menunjukkan bahwa simbol tersebut untuk menggunakan bom sedangkan simbol api untuk menembakkan api kepada musuh.

### 3.2.3 Teknologi

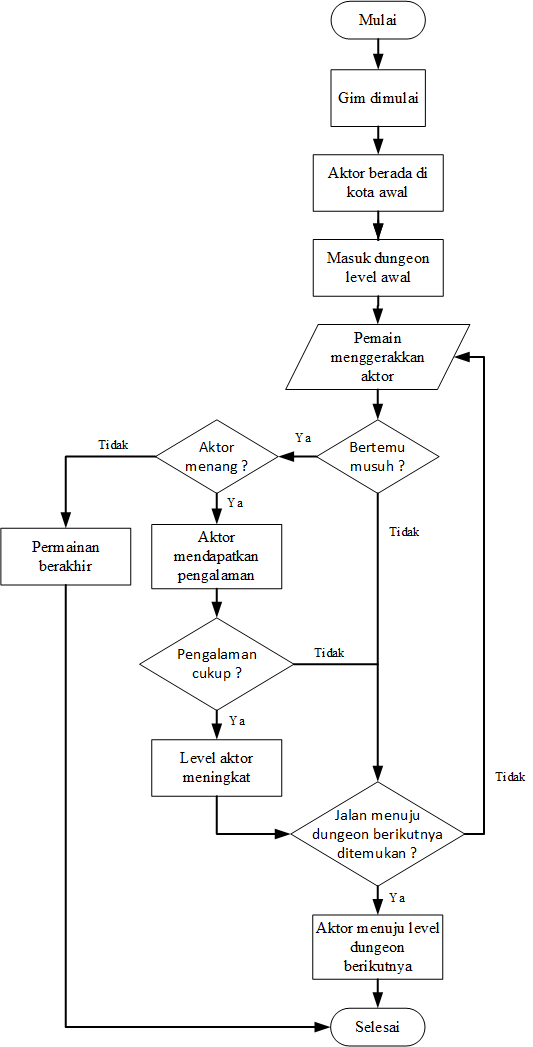
Berikut adalah teknologi yang dibutuhkan untuk membuat gim ini baik itu *hardware* ataupun *software* diperlihatkan pada tabel berikut :

###### Tabel 3.2 Tabel Teknologi

|  |  |  |  |
| --- | --- | --- | --- |
| No | Kebutuhan | Aplikasi | Fungsi |
| 1 | *Game Engine* | RPG Maker MV | Aplikasi ini berfungsi sebagai aplikasi utama yang digunakan untuk membuat gim. |
| 2 | *Graphic software* | Manga studio | Berfungsi sebagai aplikasi untuk membuat *sprite,* gambar 2D, *map* dan lain sebagainya. |
| 3 | *Code editor* | Sublime text | Berfungsi sebagai perangkat untuk melakukan *coding* menggunakan *javascript.* Hasil dari *coding* ini akan digunakan sebagai *plugin* pada gim engine. |
| 4 | *word-processing software* | Microsoft word | Digunakan sebagai aplikasi untuk membuat laporan dan proposal skripsi. |
| 5 | *Hardware* | Laptop  OS : Windows 10 64 bit  Processor : Intel I-7  Memory : 12 GB RAM  Render : NVIDIA GeForce GTX 950M | Perangkat untuk mendesain dan menguji gim yang dibuat. |
|  |  | Mouse | Digunakan ketika pembuatan gambar 2D menggunakan *graphic software.* |

### 3.2.4 Perancangan *level*

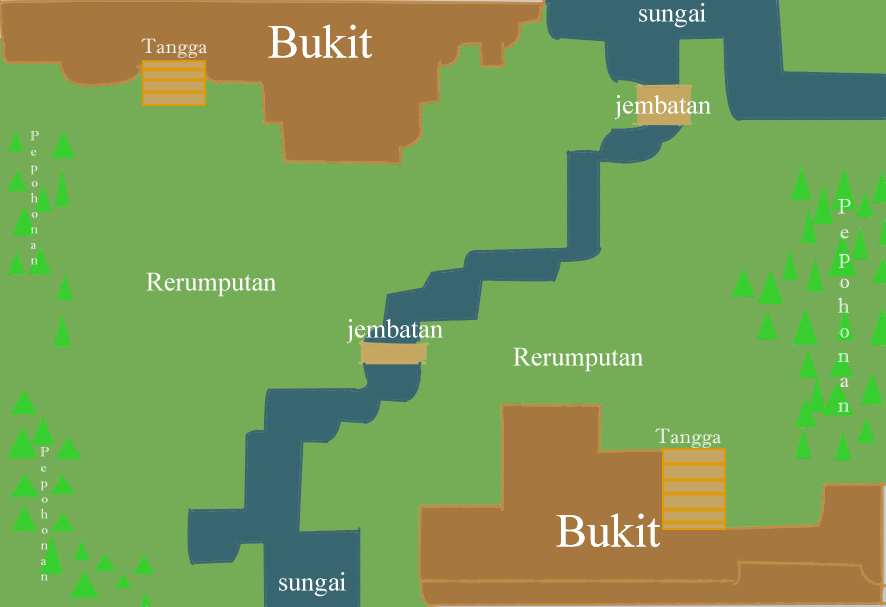
Perancangan kerangka *level dungeon* pada gim dapat dilihat pada *flowchart* berikut. *Level dungeon* akan dirancang dengan bertambahnya tingkat kesulitan setiap *levelnya.* Pada saat gim baru dimulai, pemain akan menggerakkan aktor di dalam kota tanpa adanya musuh. Pada kota ini, pemain dapat menyiapkan aktor untuk menuju *dungeon level* awal. Ketika pemain bertemu dengan musuh dalam *dungeon* tersebut, pemain dapat mencoba melawan dan mengalahkannya untuk mendapatkan pengalaman (*experience).* Jika jumlah pengalaman sudah cukup maka *level* aktor akan bertambah beserta dengan parameter milik aktor. Aktor ditugaskan untuk menemukan jalan menuju *dungeon* berikutnya untuk dapat menyelesaikan naratif pada gim.



##### Gambar 3.1 Flowchart perancangan level dungeon

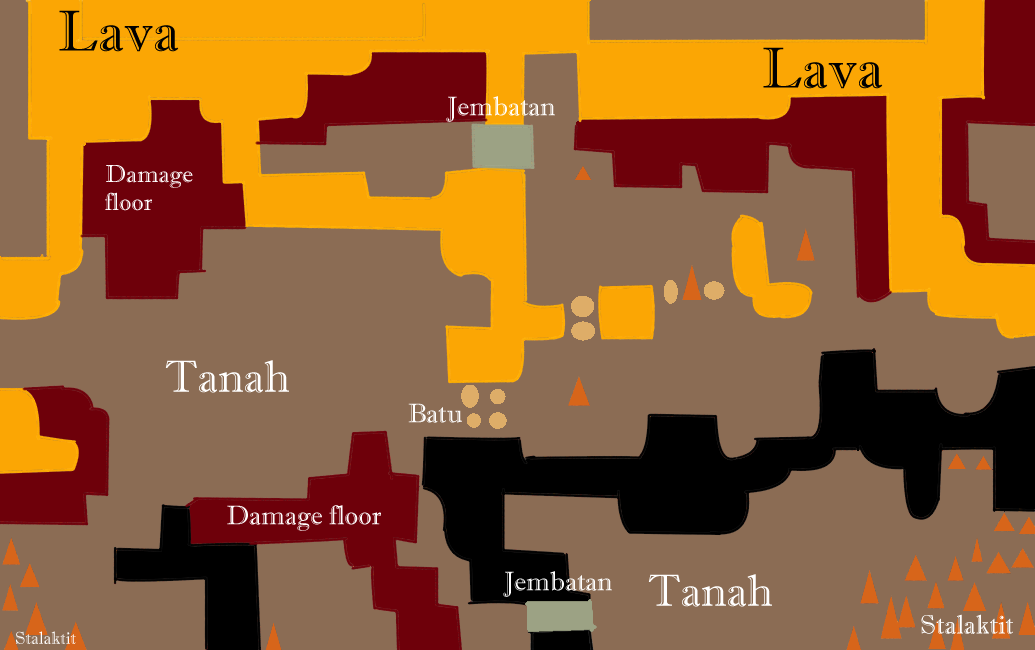
Pada gim ini, setiap *level dungeon* akan dibagi menjadi 3 kategori berdasarkan tingkat kesulitannya. Tingkat kesulitan yang dimaksud adalah parameter musuh, jumlah musuh setiap *map,* desain *map,* kecepatan gerak musuh, jebakan, dan penataan posisi musuh. Jadi ketiga kategori *level* tersebut adalah :

* *Level* awal : *Map* berisi sedikit musuh dengan pergerakan yang lambat dan sederhana. *Map* dirancang memiliki area yang luas sehingga pemain dapat mudah bergerak mencapai posisi musuh ataupun menghindari serangan musuh. Pemain hanya diberi sedikit pilihan dalam penggunaan sihir dan persenjataan agar pemain terbiasa menggunakan setiap sihir yang ada di gim. Pada *level* ini pemain akan dibuat terbiasa dahulu dengan mekanisme pertarungan yang disajikan oleh gim.



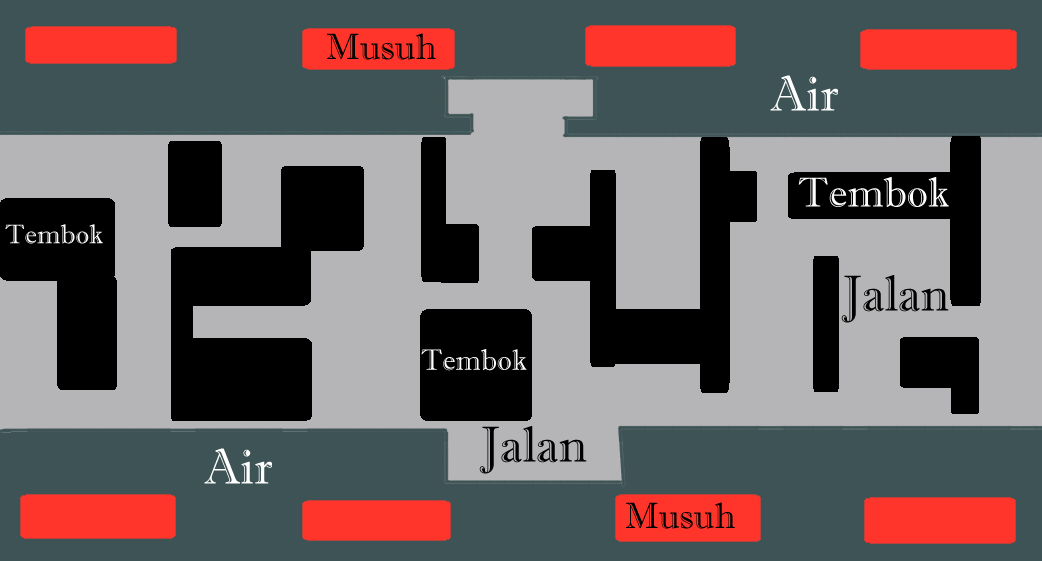
##### Gambar 3.2Perancangan *Map* Untuk *Level* Awal

* *Level* pertengahan : *Map* dibuat lebih kompleks seperti adanya jebakan (*damage floor)* pada lantai dan berliku – liku layaknya labirin. *Map* akan diisi dengan lebih banyak musuh dan lebih bervariasi dari pada *level* awal. Musuh mulai dapat menyerang pemain dari jarak jauh dan bergerak dengan cepat. Pemain akan diberikan perlengkapan dan persenjataan yang lebih kuat untuk mengatasi tantangan pada *level* ini. Pemain juga akan diberikan sihir yang dapat mengenai banyak musuh sekaligus. *Level* ini ditujukan untuk mengasah kemampuan dan kelincahan pemain akan mekanisme yang ada di dalam gim.



##### Gambar 3.3Perancangan *Map* Untuk *Level* Menengah

* *Level* akhir : *Map* akan dirancang sehingga musuh memiliki posisi yang menguntungkan dari pemain. Musuh tersebut juga akan memiliki parameter yang tinggi dan HP yang berlimpah sehingga mereka akan sulit dikalahkan oleh pemain. Pemain akan dibuat untuk menggunakan segala macam *item,* kemampuan*,* sihir dan perlengkapan yang ia miliki untuk menyelesaikan *level* akhir. *Level* ini dibuat sesulit mungkin dengan tujuan untuk menguji kemampuan pemain dalam bermain.



##### Gambar 3.4Perancangan *Map* Untuk *Level* Akhir

### 3.2.5 Sound

*Sound* yang akan digunakan dalam gim ini akan dibagi menjadi 2 bagian yaitu :

* *Background Music (BGM)* : BGM dalam gim akan menggunakan aset yang disediakan oleh engine ditambah dengan beberapa BGM gratis yang diperoleh dari internet. BGM akan diletakkan pada *title screen,* kota, *dungeon, game over screen,* pertarungan dengan bos dan lain sebagainya.
* *Sound Effect (SE)* : SE akan menggunakan SE yang disediakan oleh engine dengan beberapa SE tambahan yang didapat dari internet secara gratis. SE yang akan digunakan adalah suara ledakan saat pemain menggunakan bom, suara api saat pemain menembakkan sihir api dan lain sebagainya.

### 3.2.6 Graphic

Graphic dalam gim dibagi menjadi 2 yaitu *actor /* tokoh utama dan NPC. Tokoh utama adalah tokoh yang menjadi sorotan utama pada cerita dalam gim dan akan digerakan secara langsung oleh pemain. Sedangkan NPC adalah tokoh lain selain tokoh utama dalam cerita namun tidak dapat digerakan oleh pemain.

* **Actor / tokoh utama**

Kinen, Seorang prajurit bayaran / *mercenary* yang bekerja secara individu. Ia memiliki moral ambiguitas sehingga ia terkesan hanya mementingkan diri sendiri. Tidak peduli seberapa sulit misi atau tugas yang diberikan oleh kliennya, ia akan menyelesaikannya tanpa mempertanyakan apapun.

* **Non Player Characters:**

1. Raja Abigail, raja yang haus akan kekuasaan dan kekayaan. Ia bertekad membunuh siapa saja yang mengancam kedudukannya sebagai raja.
2. Kaptain Russell, kesatria sombong yang selalu meremehkan tokoh utama karena dia merupakan prajurit bayaran.
3. Penyihir Matilda, seorang penyihir misterius yang membantu kinen dalam perjalannya menghadapi Raja Abigail.
4. Aran, seorang pembunuh bayaran yang disewa Raja Abigail untuk menghabisi Kinen.

### *3.2.7 Game Story*

Alur cerita pada gim akan dibagi menjadi 3 bagian yaitu *start, middle,* dan *ending of game.* Setiap bagian cerita juga akan ditandai dengan perubahan *level* yang akan dimainkan oleh pemain.

* **Start of Game :**

Pada suatu hari, Kinen diberikan sebuah misi oleh utusan raja. Misi tersebut adalah untuk membawa suatu paket ke suatu tempat. Tanpa menanyakan apa isi dari kotak tersebut, kinen langsung pergi ke tempat yang dimaksud. Namun sayangnya disepanjang perjalan, ia mendapatkan banyak masalah. Selain berbagai *monster* yang harus dihadapi oleh kinen, ia juga harus bertarung dengan perampok dan pembunuh bayaran bernama Aran yang berusaha mencuri paket yang dibawa oleh kinen. Bingung akan pencuri yang mengetahui posisi dan paket yang dibawa kinen, kinen tetap memutuskan pergi ke tempat yang menjadi tujuan dari misinya.

* **Middle Game :**

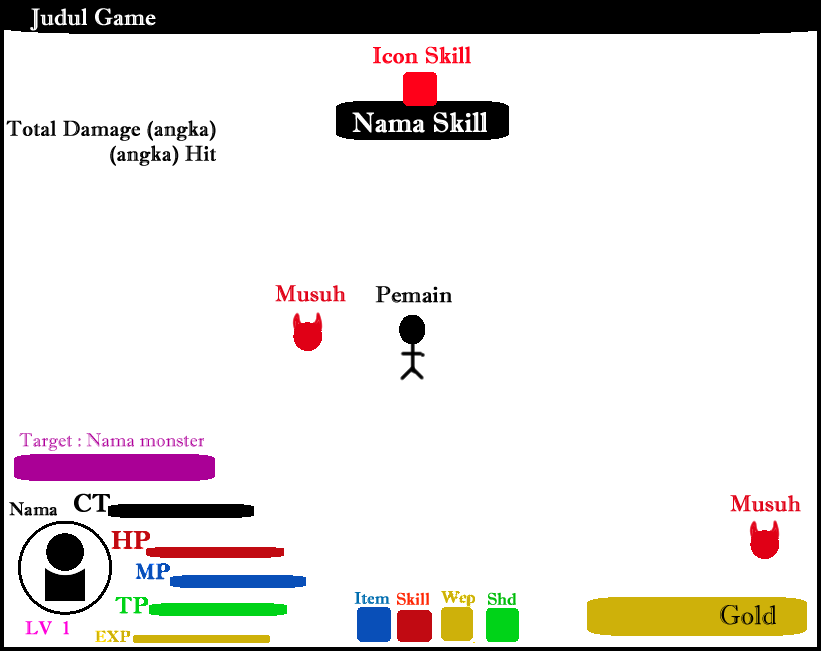
Ketika ia sampai di tempat tujuan, ia disergap oleh Kapten Russell dengan tuduhan telah memberontak terhadap raja. Bukti atas tuduhan oleh Kapten Russell adalah bahwa kinen telah membunuh putra mahkota raja. Paket yang diberikan oleh utusan raja ternyata berisi mahkota milik pangeran. Kinen sengaja dijadikan kambing hitam oleh tindak kejahatan orang lain. Tidak mau menyerah begitu saja, kinen berusaha memberikan perlawanan pada Kapten Russell. Meskipun dapat mengalahkan Kapten Russell, itu semua merupakan jebakan yang dirancang seseorang yang ditujukan untuk semakin memberatkan tuduhan pada Kinen. Kinen bukan hanya dituduh telah membunuh putra mahkota raja namun juga berusaha membunuh kesatria kerajaan.

* **Ending of Game :**

Kinen dan penyihir Matilda berusaha mengetahui siapa dalang di balik semua kejahatan yang dituduhkan kepadanya. Setelah berhasil menemui utusan raja yang memberinya paket misterius tersebut, ia sadar bahwa dalang di balik semua ini adalah Raja Abigail yang membunuh anaknya sendiri untuk menjaga takhta kerajaannya. Demi menutupi kejahatannya, ia sengaja mengirimkan utusan untuk memberikan paket berisi mahkota pangeran kepada prajurit bayaran yang bernama Kinen. Lalu raja mengirimkan Kapten Russell untuk menangkap kinen dengan tuduhan bahwa ia yang membunuh pangeran. Sedangkan paket yang dibawa kinen menjadi bukti palsu atas tuduhannya. Setelah mengetahui rencana jahat dari sang raja, ia memutuskan akan bertarung melawan raja Abigail dengan seluruh isi kerajaan dan bertekad untuk menghabisinya.

### 3.2.8 Gameplay

*Gameplay* yang ada pada gim ini berjenis *action adventure RPG. Map* yang ada pada gim ini dirancang seperti gim bergenre *adventure* dengan dungeon, labirin, dan beberapa teka – teki. Sedangkan gaya pertarungan yang ada pada gim ini adalah *action.* Pemain dapat bertarung dengan musuh secara langsung di *map*. Sedangkan perkembangan karakter akan membawa beberapa unsur dari gim *RPG* seperti adanya bermacam – macam senjata yang dapat digunakan ataupun beragam sihir yang dapat diperoleh dari menyelesaikan misi / *event* tertentu.

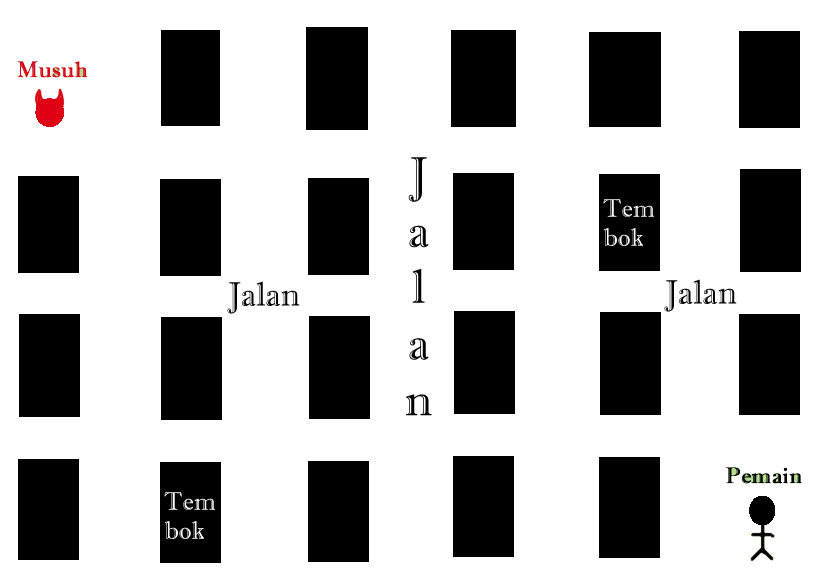


##### Gambar 3.5 Perancangan *Gameplay* Yang Ada Pada Gim Ini

### 3.2.9 Analisa algoritma A\* pada gim

Algoritma A\* akan diterapkan untuk *pathfinding* pada pergerakan NPC musuh agar musuh dapat menemukan jalur tercepat menuju posisi pemain. Algoritma ini harus diterapkan agar *NPC* dapat menemukan jalur menuju posisi pemain tidak peduli serumit apa *map* yang ada di dalam gim. Selama memang ada jalur yang menuju pemain maka algoritma ini pasti dapat menentukan jalur tersebut. Rumit atau tidaknya musuh dalam melakukan *pathfinding* dengan menggunakan algoritma A\* dapat dilihat dengan melakukan beberapa pengujian terhadap kemampuannya dalam beradaptasi dengan tindakan pemain dan *map* yang ada. Pengujian algoritma A\* sebagai *pathfinding* dapat dilakukan dengan menggunakan metode *sample test. Sample* yang akan digunakan adalah *sample map* pada gim berbentuk labirin dengan tingkat kesulitan yang berbeda – beda. Berikut adalah perancangan *sample map* yang digunakan untuk menguji kemampuan algoritma A\* dalam melakukan *pathfind :*

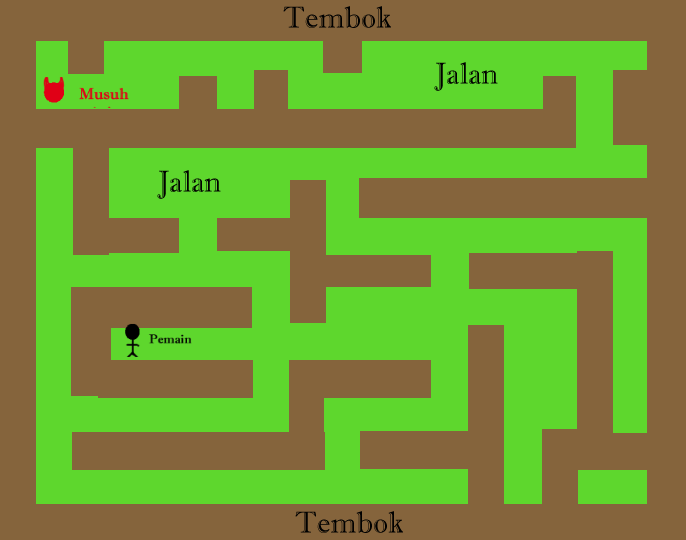
1. *Map* dibuat secara luas dengan adanya beberapa pagar sebagai penghalang



##### Gambar 3.6 Perancangan *Map* Untuk Tes Pertama *Pathfinding*

Pada tahap pertama ini *map* akan dibuat agar musuh (*slime* biru) pada posisi pojok kiri atas dapat menghampiri aktor dengan mudah. Aktor akan digerakan bebas oleh pemain. Pagar yang berupa penghalang disusun agar musuh tidak akan mengalami *stuck.* Pengujian akan berhasil jika musuh dapat terus mengikuti pergerakan aktor yang digerakan oleh pemain.

1. *Map* dibuat berbentuk labirin dengan banyak tikungan



##### Gambar 3.7 Perancangan *Map* Untuk Tes ke-2 *Pathfinding*

Pada *map* ini akan dibuat berbentuk seperti labirin dengan banyak jalan dan tikungan. *Map* ini sengaja dirancang agar jika algoritma A\* tidak bekerja dengan benar maka musuh akan mengalami *stuck* di suatu tempat. Pada tahap pengujian ini aktor akan dibuat diam di tempat yang sama. Pengujian ini akan berhasil jika musuh dapat menghampiri aktor yang tidak bergerak tanpa mengalami *stuck.*

1. *Map* akan dibuat sama persis dengan yang ada pada tahap ke-2

Pada tahap ke-3 *map* akan dibuat sama persis dengan *map* pada tahap ke-2. Hanya saja pada pengujian tahap ini, aktor akan digerakan oleh pemain. Pengujian ini akan berhasil jika musuh dapat bergerak mengikuti pergerakan aktor dalam labirin.

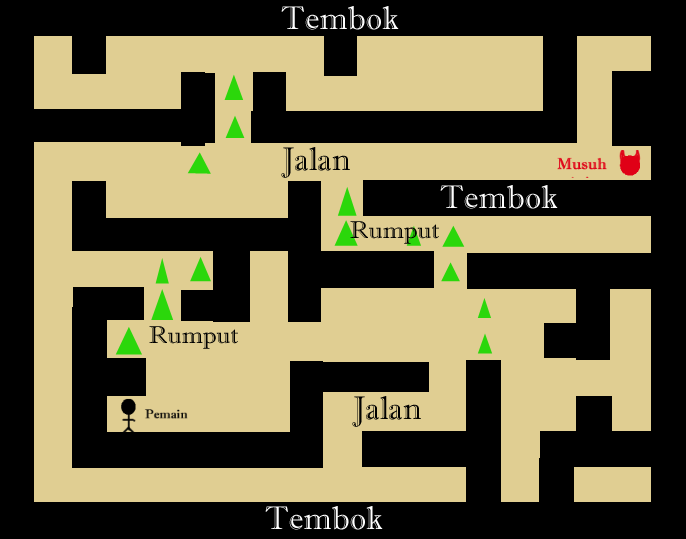
1. *Map* dibuat berbentuk labirin dengan lebih dari 1 musuh.



##### Gambar 3.8 Perancangan *Map* Untuk Tes ke-4 *Pathfinding*

Pada tahap ke-4 *map* akan dirancang lebih rumit lagi. *Map* ini memiliki 3 musuh yang diletakkan pada posisi yang berbeda-beda. Ke-3 musuh tersebut harus dapat menghampiri aktor melalui jalur yang berbeda. Tahap ini akan berhasil jika ke-3 musuh tersebut dapat memilih jalur yang paling efektif untuk bergerak menuju posisi aktor berdasarkan posisi awal mereka.

1. *Map* dibuat dengan objek yang dapat dihancurkan.

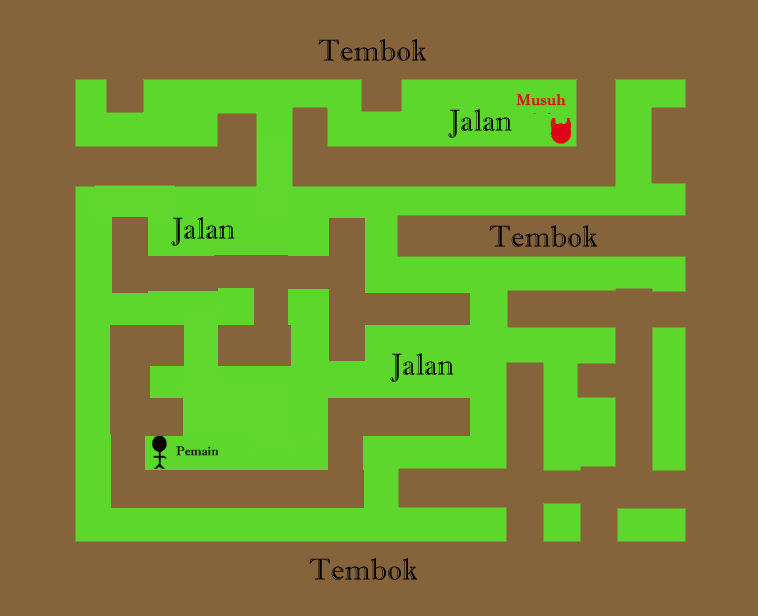


##### Gambar 3.9 Perancangan *Map* Untuk Tes ke-5 *Pathfinding*

Pada *map* ini terdapat objek rumput yang berperan sebagai penghalang sama seperti pagar. Rumput tidak dapat dilalui oleh musuh ataupun aktor namun aktor dapat menghancurkan rumput tersebut dengan senjatanya. Jadi ketika aktor memotong rumput – rumput tersebut maka akan terbuka jalur – jalur baru yang dapat dilalui oleh musuh ataupun aktor. Tahap ini berhasil jika algoritma A\* yang diterapkan dapat mempertimbang jalur baru yang dibuat oleh tindakan pemain.

1. Manipulasi algoritma A\* dalam membuat beragam jenis musuh

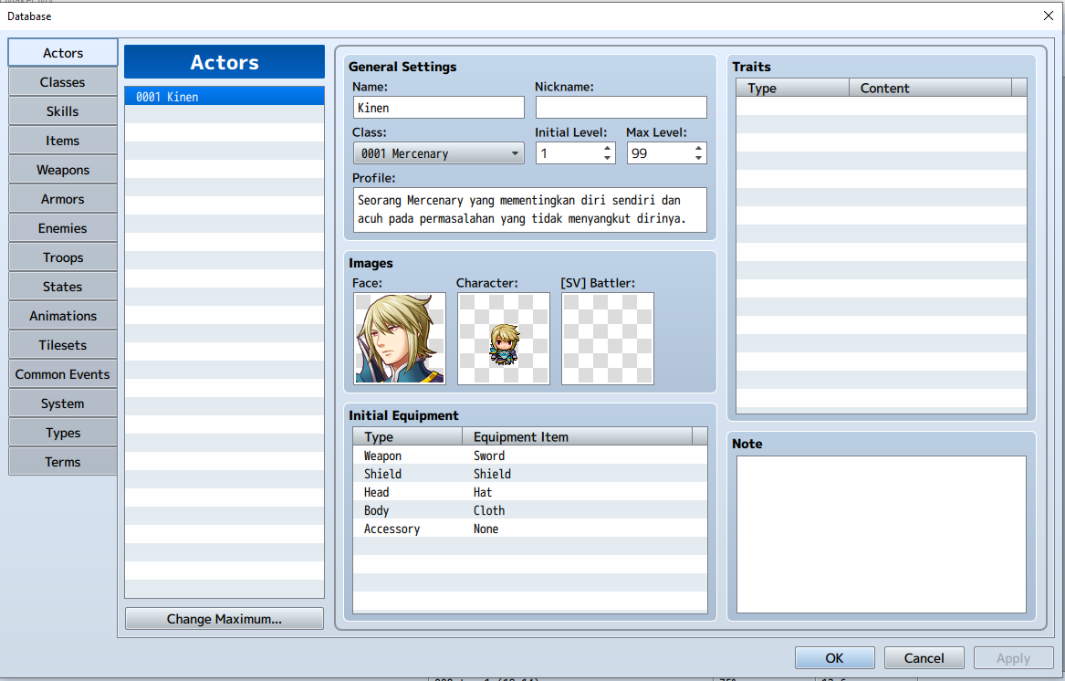
Manipulasi yang dimaksud pada tahap ini adalah bagaimana memanfaatkan algoritma A\* sebagai *pathfind* namun bukan hanya untuk menggerakan posisi musuh menuju posisi aktor. Sebagai contoh *pathfinding* dapat diterapkan pada serangan jarak jauh musuh seperti bola api. Dengan menerapkan *pathfind* pada bola api tersebut, bola api dapat bergerak mengikuti pergerakan pemain. Contoh lainnya adalah membuat musuh yang dapat bergerak menembus penghalang seperti pagar ataupun rumput.



##### Gambar 3.10 Manipulasi *Pathfind* Selain Untuk Pergerakan Musuh

### 3.3.10 Database pada gim

RPG Maker MV sudah menyediakan *database* yang dapat langsung digunakan untuk mengatur *database* untuk gim yang dibuat. *Database* ini dibagi menjadi *actors, classes*, *skills, items, weapon, armors, enemies, troops, states, animations, tilesets, common events, system, types, terms.*



##### Gambar 3.11 Database Pada RPG Maker MV

Keterangan *database :*

1. *Actors* : Digunakan untuk mengatur nama, kelas, *profile, bust character, sprite character,* dan perlengkapan aktor dan tokoh – tokoh lainnya.

*2. Classes* : Mengatur kelas yang ada pada gim. Kelas tersebut akan menentukan *parameter,* perlengkapan dan kemampuan yang dimiliki aktor.

3. S*kills* : Mengatur kemampuan dan sihir yang ada di dalam gim diantaranya nama kemampuan*,* ikon kemampuan*,* biaya MP/TP dari penggunaan kemampuandan formula kerusakan dari kemampuantersebut.

*4. Items :* Berisi barang atau peralatan yang dapat digunakan dalam gim. *Items* menentukan nama barang, ikon barang, harga dari barang tersebut, jenis barang dan efek dari barang tersebut.

*5. Weapon :* Berisi persenjataan yang dapat digunakan aktor dalam gim. *Weapons* mengatur nama senjata, ikon senjata, jenis senjata, harga senjata dan *parameter* senjata.

*6. Armors :* Berisi perlengkapan pelindung diri yang dapat dikenakan oleh aktor. *Armors* berisi nama pelindung, ikon pelindung, jenis pelindung, harga pelindung dan *parameter* pelindung.

*7. Enemies :* Mengatur musuh yang akan dihadapi oleh aktor. *Enemies* mengatur nama musuh, *parameter* musuh dan apa yang dijatuhkan musuh saat musuh mati seperti barang, uang dan juga pengalaman.

8. *Troops* : Data untuk pasukan musuh yang ada di dalam gim.

9. *States :* Berisikan data status efek yang dapat mempengaruhi aktor ataupun musuh seperti keracunan (*Poison*) ataupun tertidur (*Sleep).*

*10. Animations :* Berisikan animasi yang berfungsi sebagai visual efek. *Animations* digunakan pada saat aktor atau musuh melakukan serangan, menggunakan kemampuanatau *items.*

*11. Tilesets :* Berisikan *sprite* yang berfungsi untuk membuat *map* pada gim. *Tilesets* juga menentukan tingkah laku setiap *tile* yang ada pada gim seperti aksesibilitas dan *tile* yang dapat melukai aktor (*damage floor).*

*12. Common events :* Berisikan *events* yang dapat dipanggil berulang – ulang sehingga pengembang tidak perlu membuat *events* baru setiap kali ingin membuat *events* yang sama.

*13. System :* Pengaturan dasar untuk seluruh gim seperti *title screen* pada gim, BGM yang dimainkan pada *title screen,* dan SE pada saat pemain menyimpan data.

*14. Types :* Daftar dari beragam jenis senjata, pelindung dan kemampuanyang ada di dalam gim.

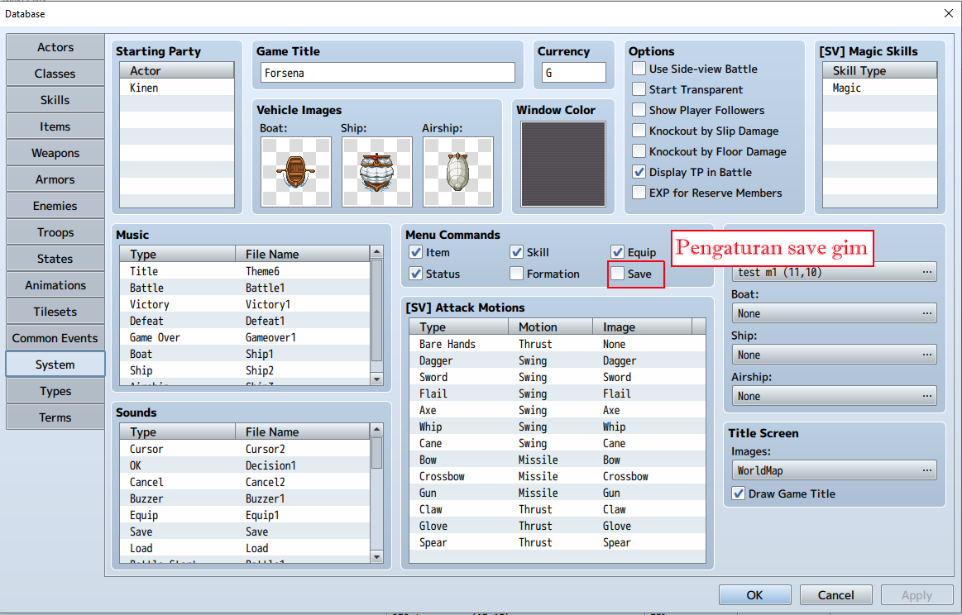
*15. Terms:* Data teks yang digunakan di dalam gim seperti nama perintah dan *parameter*.

### 3.3.11 Penyimpanan data pada gim

Setiap gim yang dibuat dengan RPG Maker MV pasti akan dilengkapi dengan fitur penyimpanan data gim (*save game)* dan melanjutkan gim (*continue*)*.* Penyimpanan data gim dapat diakses melalui *status screen* atau *event* yang ada di dalam gim. Pengembang gim dapat mengatur bagaimana pemain dapat menyimpan data gim mereka melalui *database system*.

#### 3.3.11.1 Akses penyimpanan data pada gim

Dalam gim ini, *save game* akan dirancang sehingga pemain hanya dapat melakukan *save game* melalui *event* berbentuk objek berupa kristal merah. Pengaturan akses *save game* melalui *status screen* dapat dihilangkan dengan menghapus centang pada *menu commands* yang terdapat di *database system.* Dengan demikian pemain tidak akan dapat melakukan *save game* melalui *status screen.*



##### Gambar 3.12 Pengaturan *Save Game* Pada *Database*

Hal ini dilakukan agar pemain tidak melakukan *save game* di sembarang *map* yang ada. Selain itu pembuatan *event* khusus untuk *save game* dapat digunakan sebagai peringatan bagi pemain jika ada pertarungan yang sulit di area atau *map* selanjutnya.

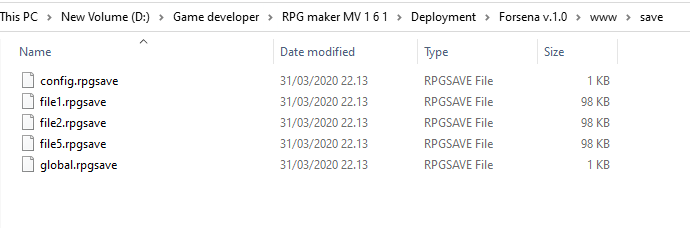
#### 3.3.11.2 Lokasi berkas *save game*

Ketika pemain melakukan *save game*, maka gim akan membuat berkas baru yang berisi data permainan. Berkas baru yang dibuat oleh gim secara otomatis akan diberi nama file(angka).rpgsave. Angka pada nama berkas akan ditentukan dari nomor *save* slot di mana pemain melakukan *save game.*



##### Gambar 3.13 Contoh Menu *Save Game* Pada Gim

Dari gambar di atas dapat dilihat bahwa ada 3 slot yang berisi data gim yaitu *file 1, file 2* dan *file 5.* Gim akan membuat 3 berkas yang masing – masing berisi data gim dari setiap *file* yang dibuat pemain*.* Berkas – berkas tersebut dapat ditemukan pada [Lokasi gim pada PC]\Forsena\www\save.



##### Gambar 3.14 Lokasi *Save Game* Pada Gim

Pada lokasi *save game* terdapat 5 berkas yaitu :

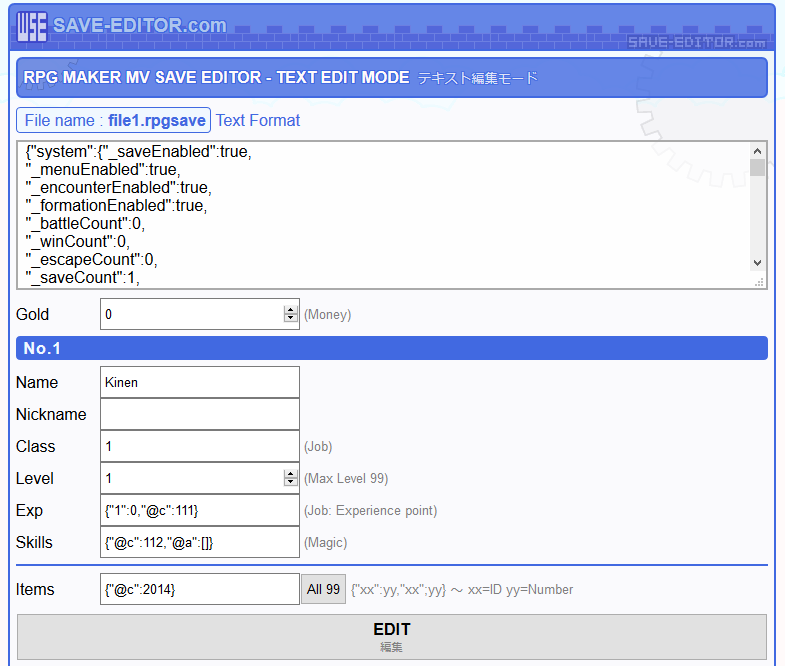
* + - 1. Config.rpgsave menyimpan data konfigurasi *menu options* seperti pengaturan volume BGM atau SE.
      2. File1.rpgsave menyimpan data permainan pada *save* slot nomor 1.
      3. File2.rpgsave menyimpan data permainan pada *save* slot nomor 2.
      4. File5.rpgsave menyimpan data permainan pada *save* slot nomor 5.
      5. Global.rpgsave dibuat setiap kali pemain melakukan *save game.* Berkas ini digunakan untuk memuat berkas – berkas hasil dari *save game* ketika pemain ingin melanjutkan permainan.

Jika pemain ingin menghapus data permainan pada *save* slot nomor 5 maka pemain tinggal menghapus berkas file5.rpgsave secara manual. Perlu diingat bahwa jika pemain menghapus berkas global.rpgsave maka pemain tidak akan dapat melanjutkan permainan.

#### 3.3.11.3 Format save game

Berkas yang menyimpan data dari *save game* pada gim ini memiliki format .rpgsave. Format ini merupakan bentuk enkripsi yang digunakan oleh RPG Maker MV. Engine ini secara otomatis akan melakukan enkripsi pada *save game* dari gim yang dibuat untuk menghindari terjadinya kecurangan pada permainan. Bentuk kecurangan yang dapat terjadi adalah *save editing* di mana pemain dengan sengaja memanipulasi data *save* game untuk menambah jumlah uang, menaikkan *level* aktor sampai dengan menambahkan jumlah barang tertentu.

Namun untuk menganalisa berkas *save game* yang ada pada gim ini, akan dilakukan proses deskripsi untuk mengubah format .rpgsave menjadi format .txt. Proses deskripsi berkas dapat dilakukan melalui *website* saveeditonline.comatau save-editor.com.



##### Gambar 3.15 Proses Deskripsi Dari .rpgsave

Jadi dalam berkas .rpgsave terdapat data jumlah uang yang dimiliki pemain, daftar aktor yang ada dalam kelompok, nama aktor, nama panggilan aktor, id kelas aktor, level aktor, jumlah exp aktor, kemampuan dan sihir yang dimiliki aktor, jenis barang dan jumlah barang yang dimiliki aktor, data *map* di mana pemain melakukan *save game,* judul gim dan lama waktu bermain.

## 3.3 Perancangan Visual

Dalam pembuatan sebuah gim, unsur visual sangatlah penting untuk menonjolkan kualitas dari gim yang akan dibuat*.* Perancangan visual yang menarik dan tepat akan membuat pemain lebih tertarik akan gim yang mereka mainkan. Perancangan visual dalam gim akan dibagi menjadi *character, tile map, special effect* dan *UI.*

### 3.3.1 Desain karakter

Desain karakter dalam gim ini akan dibagi menjadi 2 yaitu *bust character* dan *sprites character.*

* ***Bust character / Face actor***

*Bust character / face actor* adalah tampilan karakter berbentuk gambar 2D yang muncul saat karakter melakukan interaksi dengan karakter lain. Gambar ini bisa berupa wajah karakter, setengah badan atau seluruh badan karakter. *Bust character* biasanya digunakan untuk menandakan bahwa karakter itu memiliki peran yang vital pada alur cerita gim atau akan sering dijumpai oleh pemain di dalam gim. Pada unsur estetika, *bust character* dapat menambah daya tarik pemain pada gim yang mereka mainkan.

###### Tabel 3.3 Bust character / face actor

|  |  |  |  |
| --- | --- | --- | --- |
| No | Karakter | Nama | Keterangan |
| 1 |  | Kinen | Tokoh utama dalam gim. Ia bekerja sebagai kesatria bayaran / *mercenary* untuk mencari uang. |
| 2 |  | Raja Abigail | Tokoh antagonis utama dalam cerita. Ia seorang raja yang jahat dan tamak akan kekuasaan |
| 3 |  | Kapten Russell | Salah satu bos musuh di dalam gim. Ia memiliki watak yang sombong karena kedudukannya di kerajaan dan memandang kinen sebagai orang rendahan. |
| 4 |  | Penyihir Matilda | Seorang penyihir misterius yang membantu kinen dalam mengalahkan Raja Abigail. |
| 5 |  | Aran | Pembunuh bayaran yang bekerja sebagai mata – mata Raja Abigail. Ia merupakan musuh yang harus dihadapi kinen. |

* ***Sprite Character***

*Sprite character* adalah kumpulan gambar 2D karakter yang disusun sedemikian rupa sebagai animasi pada karakter ketika karakter tersebut melakukan gerakan atau tindakan di dalam gim. *Sprite* harus dibuat identik dengan *face actor* sehingga pemain tidak kebingungan saat berinteraksi dengan *NPC.* Penggunaan *sprite* dapat membuat gim terasa semakin hidup di mata pemain. *Sprite character* diperuntukan pada karakter yang dapat bergerak atau melakukan suatu tindakan tertentu di dalam gim. Dalam pembuatan *sprite* terdapat 2 unsur utama yang harus diperhatikan saat membuat *sprite*, yaitu:

1. ***Frame***

***Frame*** di sini bisa diartikan sebagai berapa jumlah gambar yang akan digunakan untuk membuat *sprite character.* Semakin banyak jumlah gambar yang digunakan, maka akan semakin baik *sprite* yang dihasilkan.

1. **Duras*i***

**Durasi**di sini merupakan berapa *frame* yang harus dijalankan / dianimasikan tiap satuan detik. Pada dasarnya semakin banyak jumlah *frame* yang digunakan dan semakin sedikit durasinya maka akan semakin baik *sprite* yang dihasilkan. Kunci utama dari pembuatan *sprite* yang baik adalah adanya keseimbangan antara *frame* dan durasi.

###### Tabel 3.4 Sprite Character

|  |  |  |  |
| --- | --- | --- | --- |
| **No** | ***Sprite*** | **Nama** | **Keterangan** |
| 1. |  | Kinen berjalan | *Sprite* milik tokoh utama kinen. *Sprite* ini digunakan saat pemain menggerakan kinen di dalam gim |
| 2 |  | Kinen memanah | *Sprite* tokoh utama kinen saat ia menggunakan busur dan panah |
| 3 |  | Kinen menahan serangan | *Sprite* kinen saat ia menahan serangan musuh menggunakan perisai |
| 4 |  | *Bat* | *Sprite* musuh pada *level* awal. Musuh ini akan bergerak mendekati pemain dan akan menyerang pemain. |
| 5 |  | *Slime* | *Sprite* musuh pada *level* awal. Musuh ini bergerak secara random namun akan menyerang pemain saat pemain berada di dekat musuh. |
| 6 |  | *Ghost* | *Sprite* musuh pada *level* awal. Musuh ini bergerak dengan lambat namun dapat menembakkan bola api ke pemain |
| 7 |  | *Minotour* | *Sprite* musuh pada *level* pertengahan. Musuh ini akan bergerak lebih cepat jika *HP* musuh ini kurang dari 50%. |
| 8 |  | *Mage skull* | *Sprite* musuh pada *level* akhir. Musuh ini akan melompat menjauh dari pemain jika pemain bergerak mendekati musuh ini. |

### 3.3.2 Tile *map*

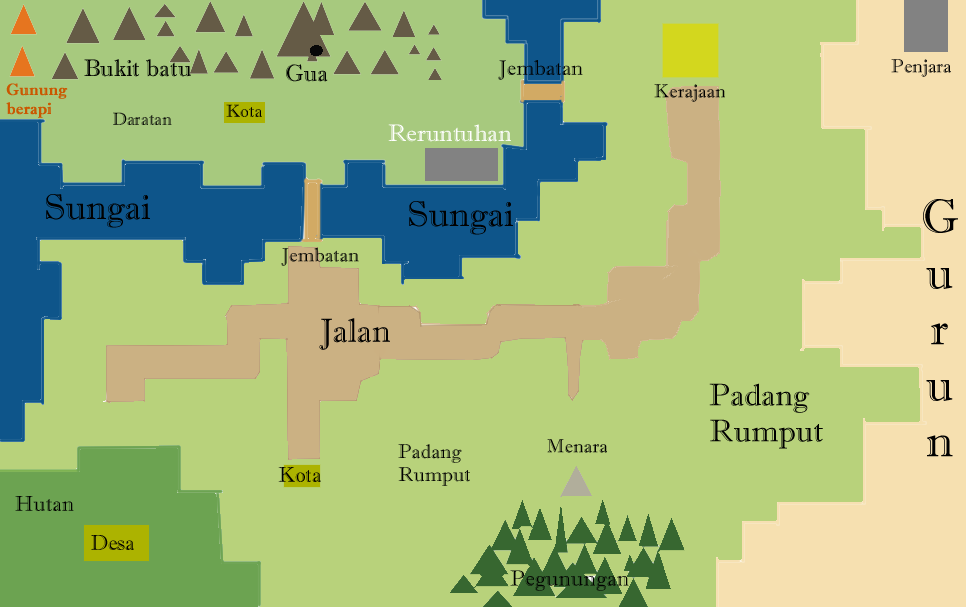
Desain *map* pada gim ini akan menggunakan *tileset* yang dibagi menjadi 4 yaitu bagian lantai, penghalang, objek dan bangunan. Lantaidigunakan untuk membuat tanah dan lantai yang dapat dilalui pemain. Penghalangdigunakan untuk membuat *tile* yang tidak dapat dilalui pemain namun beberapa penghalangdapat dihancurkan dengan serangan biasa atau dengan menggunakan bom. *Tile* objek merupakan *tile* yang digunakan untuk fungsi tertentu seperti menyimpan data atau menaruh peti harta karun. Sedangkan bangunan merupakan *tile* yang digunakan pemain untuk mengakses *map* / bangunan tersebut. Berikut daftar material *tileset* yang akan digunakan untuk mendesain *map* pada gim.

###### Tabel 3.5 *tileset*

|  |  |  |  |
| --- | --- | --- | --- |
| **No** | ***Tileset*** | **Kategori** | **Keterangan** |
| 1 |  | Lantai | Lantaimerupakan lantai atau tanah yang dapat dilalui oleh pemain ataupun NPC. |
| 2 |  | Penghalang  (Batu coklat) | Penghalangini merupakan penghalang bagi pemain untuk bergerak namun dapat dihancurkan dengan menggunakan bom. |
| 3 |  | Bangunan | Bangunan ini merupakan *tile* yang memindahkan posisi pemain ke *map* kastillain jika diinjak oleh pemain. |
| 4 |  | Lantai | Lantai yang berfungsi sebagai jebakan bagi pemain. Jika pemain menginjak lantai ini maka pemain akan terkena *damage*. |
| 5 |  | Objek  (Peti bagus) | Objek ini berfungsi untuk menyimpan suatu barangdidalamnya. |
| 6 |  | Objek  (Kristal merah) | Objek ini berfungsi untuk melakukan penyimpanan data di dalam gim (*save game).* |
| 7 |  | Penghalang | Objek dalam gim yang tidak dapat dihancurkan pemain dengan cara apapun. |

#### 3.3.2.1 Perancangan *world* *map*

Dengan menggunakan *tileset* yang ada maka sebuah *world map* dapat dibuat. *World map* ini merupakan area di mana pemain dapat menjelajahi dunia yang ada pada gim dengan bebas. *World map* berfungsi sebagai *map* yang menghubungkan *map* yang satu dengan *map* yang lainnya. *World map* berisi *tile* kota atau kastil kerajaan dan *dungeon* yang berupa gua, menara, ataupun reruntuhan.



##### Gambar 3.16 Perancangan *World Map*

Pada *world map* di atas jika pemain menginjak *tile* yang berupa kota ataupun *dungeon* maka pemain akan langsung ditransfer ke *map* yang bersangkutan.

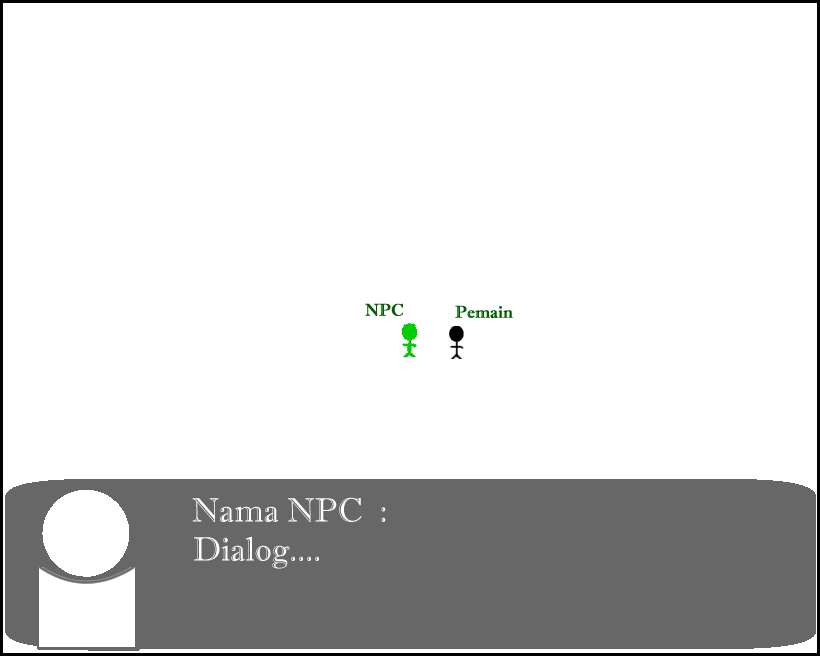
#### 3.3.2.2 Perancangan *map* kota

*Map* kota dibuat dengan menggunakan *tileset* yang berisi bangunan seperti gedung, rumah dan pepohonan. Pada *map* kota, pemain dapat menggunakan *gold /* uanguntuk membeli barang*,* alat*,* persenjataan ataupun perlengkapan baru. *Map* ini juga menyediakan penginapan untuk mengisi HP, MP dan TP yang dimiliki aktor sampai penuh. *Event* yang bersangkutan dengan alur cerita utama pada gim juga akan terjadi pada *map* kota.



##### Gambar 3.17 Perancangan *Map* Kota

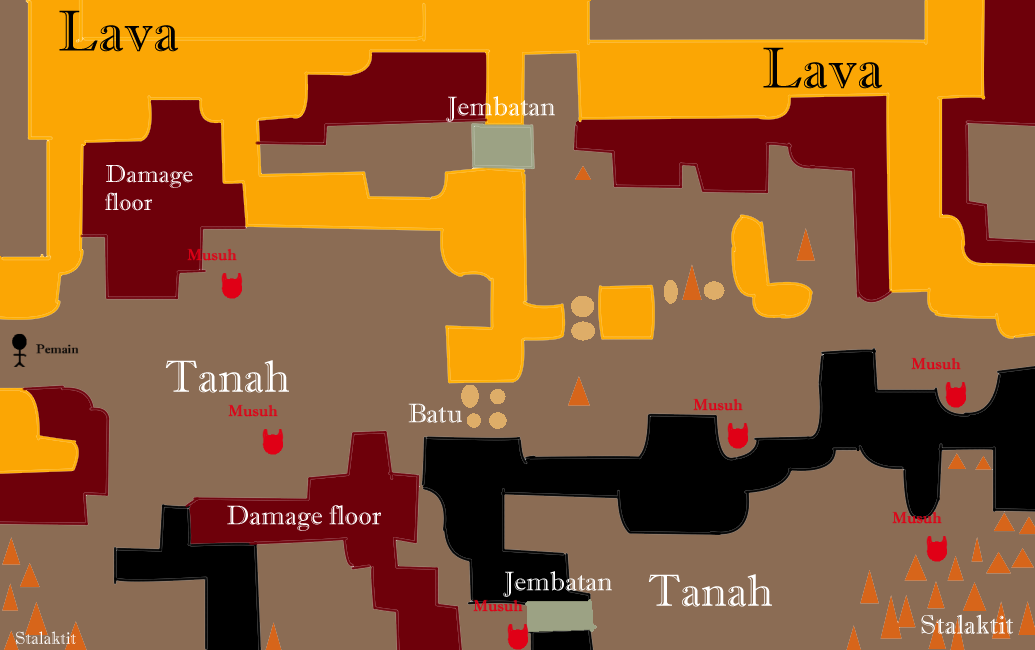
*Map* ini juga berisi NPC di mana pemain dapat berinteraksi dengan mereka ataupun menerima misi (*quest)* yang diberikan NPC*.* Jika pemain menyelesaikan misi tersebut maka pemain akan diberikan imbalan yang dapat berupa *gold, items,* persenjataan, perlengkapan, ataupun kemampuanbaru.



##### Gambar 3.18 Contoh Interaksi Antara Aktor Dengan NPC

#### 3.3.2.3 Perancangan *map* *dungeon*

M*ap dungeon*  akan berisi berbagai musuh yang menjadi lawan dari pemain. *Map* ini juga merupakan *map* di mana akan terjadinya *event* untuk melanjutkan cerita pada gim. Beberapa misi *(quest)* yang diberikan NPC hanya dapat diselesaikan pada *map dungeon*.



##### Gambar 3.19 Perancangan *Dungeon* Gunung Berapi

Selain berisi musuh, *map* ini juga terdapat peti harta karun. Jika pemain membuka peti harta karun tersebut, pemain dapat memperoleh isi dari peti tersebut. Peti harta karun dapat berisi barang*,* alat*,* persenjataan ataupun perlengkapan baru yang dapat digunakan oleh pemain untuk mempermudah mereka dalam bertarung melawan musuh – musuh yang ada.

### *3.3.3 Special effect*

*Special effect* di dalam gim berfungsi untuk memperkaya nilai visual yang ada di dalam gim. *Special effect* biasa digunakan bersama dengan *sound effect* untuk menghasilkan efek yang maksimal. Berikut adalah daftar *special effect* yang akan digunakan di dalam gim.

###### Tabel 3.6 Special effect

|  |  |  |  |
| --- | --- | --- | --- |
| **No** | ***Special effect*** | **Nama** | **Keterangan** |
| 1 |  | Ledakan | *Special effect* yang digunakan ketika pemain menggunakan bom dan meledak. |
| 2 |  | Hantaman | Special effect untuk serangan yang melibatkan penggunaan senjata tumpul. |
| 3 |  | Tebasan | Special effect untuk serangan yang menggunakan senjata tajam. |
| 4 |  | Api | Digunakan ketika pemain / musuh terkena sihir api |
| 5 |  | Cahaya | Digunakan ketika pemain menggunakan sihir cahaya |
| 6 |  | *Guard* | Digunakan saat serangan musuh mengenai pemain yang bertahan. |

### 3.3.4 *User Interface*

*User Interface* atau *UI* adalah perancangan tampilan menu dalam gim yang dapat dilihat oleh pemain. *UI* biasa digunakan sebagai mekanisme komunikasi antara gim dengan pemain. Penataan *UI* yang baik dapat meningkatkan pengalaman pemain dalam memainkan gim. *UI* akan dibuat sedemikian rupa agar pemain dapat memahami gim dengan baik.

HUD itu sendiri merupakan kepanjangan dari *Heads up display*, merupakan metode di mana informasi secara visual disampaikan kepada pemain sebagai bagian dari *user interface* dalam gim*. HUD* biasanya menunjukkan bar/kotak HP(*Hit Point*) ataupun MP(*Mana Point*) dan biasanya diletakkan di sudut layar. Fungsi HUD ini adalah untuk memberikan informasi kepada pemain akan kondisi di dalam gim.

###### Tabel 3.7 *HUD*

|  |  |  |  |
| --- | --- | --- | --- |
| **No** | **HUD** | **Nama** | **Keterangan** |
| 1 |  | *Actor* | Menunjukkan wajah dari tokoh utama dalam gim |
| 2 |  | *Status* | Sebagai bingkai untuk meletakkan  *HP, MP, TP* dan *Exp bar.* |
| 3 |  | *HP Bar* | Sebagai HUD untuk memberi informasi kepada pemain berapa HP yang dimiliki pemain |
| 4 |  | *MP Bar* | Sebagai HUD untuk memberi informasi kepada pemain berapa MP yang dimiliki pemain |
| 5 |  | *TP Bar* | Sebagai HUD untuk memberi informasi kepada pemain berapa TP yang dimiliki pemain |
| 6 |  | *EXP Bar* | Sebagai HUD untuk memberi informasi kepada pemain berapa EXP yang dimiliki pemain. |
| 7. |  | *Enemy HP bar* | Digunakan untuk menunjukkan berapa banyak HP yang dimiliki musuh. |

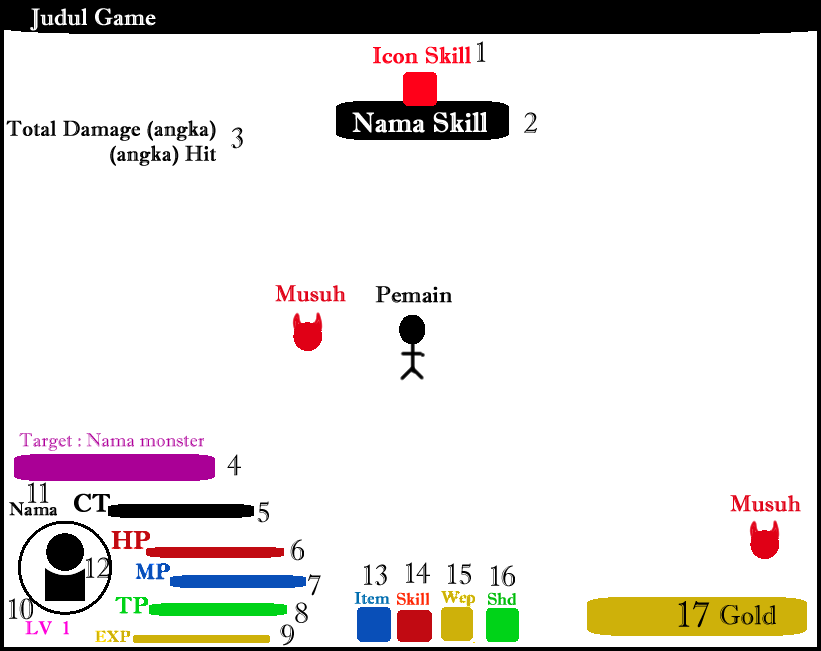
Selain *HUD*, ada jugaikon atau simbol yang digunakan di dalam gim. Ikon merupakan gambar dengan ukuran yang sangat kecil yang biasa digunakan sebagai simbol untuk mewakili fungsi tertentu. Ikon sering digunakan sebagai pengganti teks karena pemain lebih mudah dan cepat memahami makna dari gambar daripada harus membaca teks.

###### Tabel 3.8 Ikon / simbol

|  |  |  |  |
| --- | --- | --- | --- |
| **No** | **Ikon** | **Nama** | **Keterangan** |
| 1 |  | *frame item* | Digunakan sebagai wadah untuk menunjukkan barang apa yang digunakan pemain. |
| 2 |  | *Frame skill* | Digunakan sebagai wadah untuk menunjukkan kemampuanapa yang digunakan pemain. |
| 3 |  | *Frame weapon* | Digunakan sebagai wadah untuk menunjukkan senjata apa yang digunakan pemain. |
| 4 |  | Ikon *weapon* | Digunakan untuk menunjukkan jenis senjata yang ada di dalam gim. |
| 5 |  | Ikon *skill / magic* | Digunakan untuk menunjukkan jenis sihir dan kemampuan yang ada di dalam gim. |
| 6 |  | Ikon *item* | Digunakan untuk menunjukkan jenis barang yang ada di dalam gim. |
| 7 |  | Ikon ekspresi | Digunakan untuk menunjukkan emosi karakter di dalam gim. |

#### 3.3.4.1 Perancangan User Interface aktor

*User interface* akan dirancang untuk memberikan informasi yang dibutuhkan oleh pemain. *HUD* akan diletakkan pada bagian bawah layar agar tidak mengganggu pandangan pemain pada *map. HUD* akan menjadi transparan jika pemain berada pada belakang *HUD* sehingga pemain akan tetap mengetahui posisi aktor. *UI* aktor terdiri dari parameter aktor (nama, *CT, HP, MP, TP, EXP,* dan *Lv), item,* kemampuan*, equipment* dan *gold.*

.

##### Gambar 3.20 Perancangan Desain *UI* Aktor

Keterangan :

1. *Icon Skill* : Menunjukkan ikon kemampuan yang digunakan pemain
2. Nama *Skill* : Menunjukkan nama kemampuanyang digunakan pemain
3. Total *hit* : Menunjukkan total kerusakan dan jumlah serangan yang dilakukan pemain kepada musuh.
4. Target Nama monster : Menunjukkan nama dan jumlah *HP* yang dimiliki musuh
5. CT : Menunjukkan *charge time* saat aktor hendak menggunakan serangan sihir atau energi di saat pemain menggunakan serangan biasa.
6. HP : Menunjukkan *hit point /* nyawa yang dimiliki aktor. Bar *HP* akan berkurang jika pemain terkena serangan musuh atau menginjak lantai jebakan.
7. MP : Menunjukkan *mana point* yang dimiliki aktor. Bar *MP* akan berkurang jika pemain menggunakan sihir.
8. TP : Menunjukkan *tech point* yang dimiliki aktor. Bar *TP* akan berkurang saat aktor menggunakan kemampuan khusus atau bertahan.
9. Exp : Menunjukkan *experience point* yang dimiliki aktor. Level Aktor akan meningkat jika bar ini penuh.
10. Nama aktor : menunjukkan nama dari karakter utama.
11. Lv : Menunjukkan *level* karakter yang dimainkan
12. Foto : Menunjukkan foto *bust* dari aktor.
13. Item : Menunjukkan barang */* alatapa yang digunakan oleh pemain.
14. Skill : Menunjukkan kemampuan */*sihir apa yang digunakan oleh pemain.
15. Wep : Menunjukkan jenis senjata apa yang digunakan oleh pemain.
16. Shd : Menunjukkan perisai yang dimiliki pemain.
17. Gold : Menunjukkan jumlah uang yang dimiliki pemain.

#### 3.3.4.2 Perancangan User Interface Skill

*User interface skill* akan berisi opsi untuk melihat dan memilih kemampuan */* sihir yang dimiliki aktor. *UI* ini berisi simbol, nama dan keterangan dari kemampuan */* sihir tersebut.



##### Gambar 3.21 Perancangan *UI* Kemampuan

Menu ini dapat digunakan sebagai *shortcut* bagi pemain untuk mengganti kemampuan */* sihir yang akan digunakan. Saat pemain memilih kemampuan */* sihir tertentu, maka HUD kemampuanaktor akan langsung berganti dengan simbol dari kemampuan */* sihir yang dipilih.

#### 3.3.4.3 Perancangan User Interface Item

*User interface item* akan berisi opsi untuk melihat dan memilih barang*/* alat yang dimiliki aktor. *UI* ini berisi simbol, nama dan keterangan dari barang*/* alat beserta jumlah alat yang dimiliki.

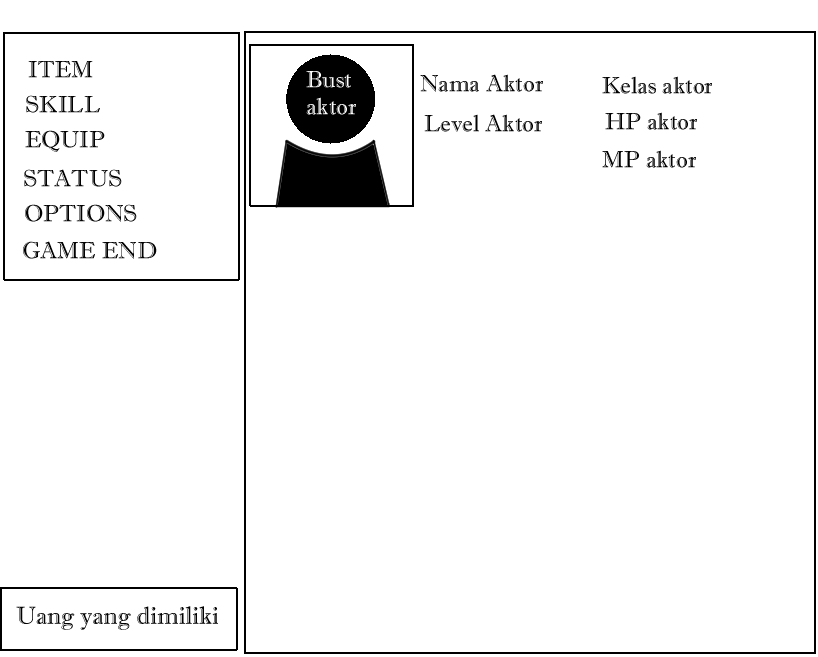


##### Gambar 3.22 Perancangan Desain *UI Item*

Menu ini dapat digunakan sebagai *shortcut* bagi pemain untuk mengganti barang */* alatyang akan digunakan. Saat pemain memilih barang */* alat tertentu, maka HUD *item* aktor akan langsung berganti dengan simbol dari */* alat yang dipilih. Pada gambar 3.6 dapat dilihat bahwa pemain memiliki 10 bom yang dapat digunakan untuk menghancurkan batu.

#### 3.3.4.4 Perancangan User Interface *status screen*

*User interface status screen* akan dirancang agar pemain dapat melihat barang *(Item)* yang dimiliki aktor, kemampuan / sihir (*skill)* yang dapat digunakan aktor, perlengkapan *(equipment)* yang dikenakan aktor, layar (*status)* yang memperlihatkan parameter aktor secara lengkap, opsi *(option)* untuk mengganti pengaturan pada gim, kembali ke *main menu* (*game end),* HP&MP yang dimiliki aktor, jumlah uang yang dimiliki aktor.



##### Gambar 3.23 Perancangan UI *Status Screen*

## 3.4 Perancangan desain program

Pada tahap perancangan desain program akan dijelaskan mengenai *game plan, flowchart, use case* dan *activity diagram.*

### 3.4.1 *Game plan*

*Game plan* merupakan gambaran dari alur gim dalam bentuk diagram. *Game plan* menjelaskan perancangan kerangka dari gim secara umum.

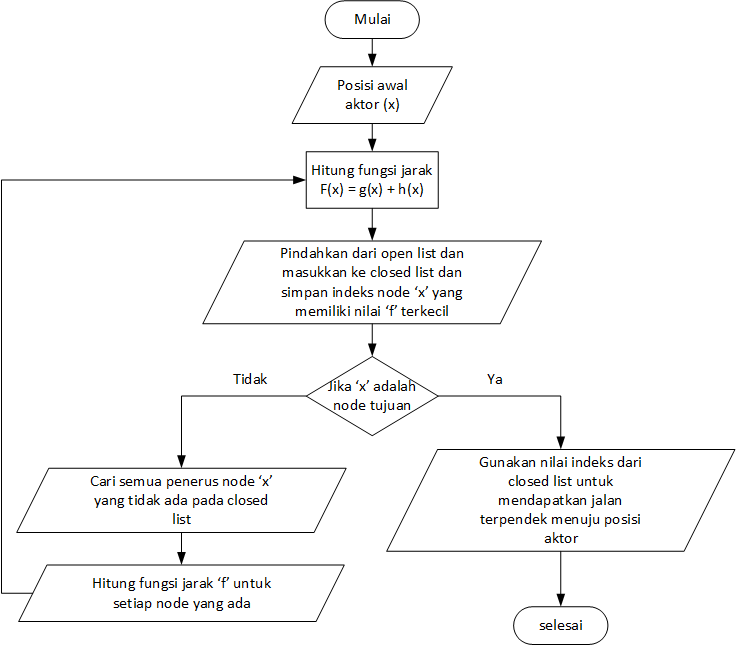


##### Gambar 3.24 *Flowchart* Game Plan

Pada saat pemain mulai menjalankan gim, yang pertama kali pemain lihat adalah ***splash screen***. *Splash screen* akan berisi logo STIKI pada latar bewarna hitam. Setelah *splash screen* selesai, pemain akan dibawa ke menu utama***.*** Pada menu utama*,* pemain dapat melakukan konfirmasi untuk memulai permainan. Setelah gim dimulai dan prolog cerita pada gim selesai maka pemain dapat menggerakkan aktor untuk menjelajahi kota dan *dungeon* yang ada di dalam gim. Jika aktor bertemu NPC musuh maka pemain dapat mencoba melawan dan mengalahkannya. Jika aktor berhasil mengalahkan musuh tersebut maka permainan dapat dilanjutkan, namun jika aktor kalah melawan musuh maka permainan berakhir. Permainan akan terus dilanjutkan sampai pemain dapat mengalahkan bos terakhir pada gim. Ketika bos tersebut berhasil dikalahkan maka epilog cerita akan dimulai untuk menandakan bahwa gim telah tamat sehingga permainanpun telah selesai.

### 3.4.2 Flowchart algoritma A\* pada gim

Berikut ini adalah *flowchart* penggunaan algoritma A\* pada *pathfinding* yang dilakukan musuh untuk menemukan posisi aktor.

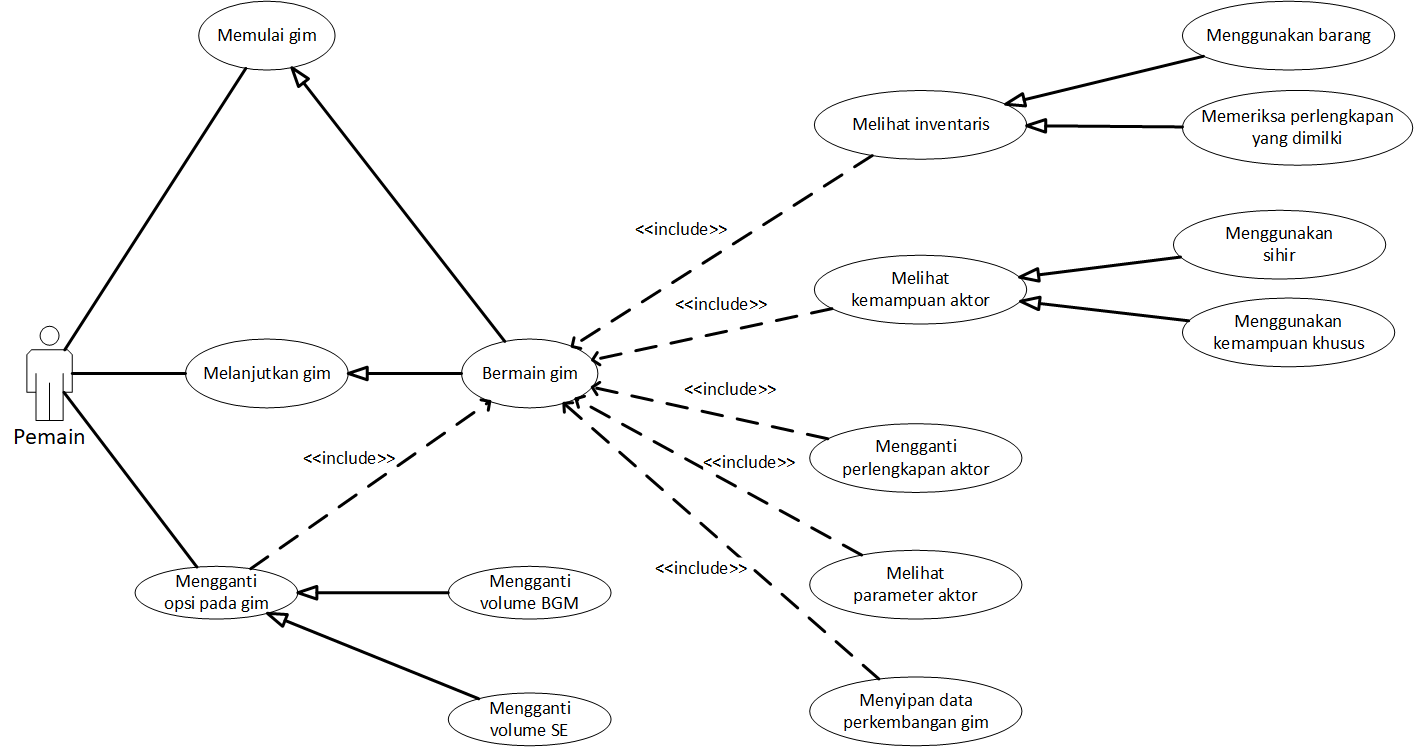


##### Gambar 3.25 *Flowchart* Algoritma A\*

Pada saat musuh berada di *map,* mereka akan melakukan *pathfinding* dengan menggunakan algoritma A\*. Musuh akan mendeteksi di mana letak posisi aktor untuk menentukan jalan paling optimal ke posisinya. Setiap *node* akan dihitung nilainya menggunakan rumus *f(x) = g(x)+h(x).* Nilai f yang terkecil akan diambil dan dimasukkan ke dalam *closed list.* Hal ini akan diteruskan sampai dengan didapatkannya jalan menuju posisi aktor.

### 3.4.2 Use case

Diagram *use case* dapat menunjukkan fungsi apa saja yang ada pada gim yang dapat diakses oleh pemain. Diagram *use case* dari pembuatan gim action adventure RPG 2D dengan menggunakan algoritma A\* yang akan dikembangkan dapat dilihat pada gambar di bawah.



##### Gambar 3.26 *Use Case* Pada Gim Forsena

Dari diagram *use case* di atas dapat dijelaskan sebagai berikut :

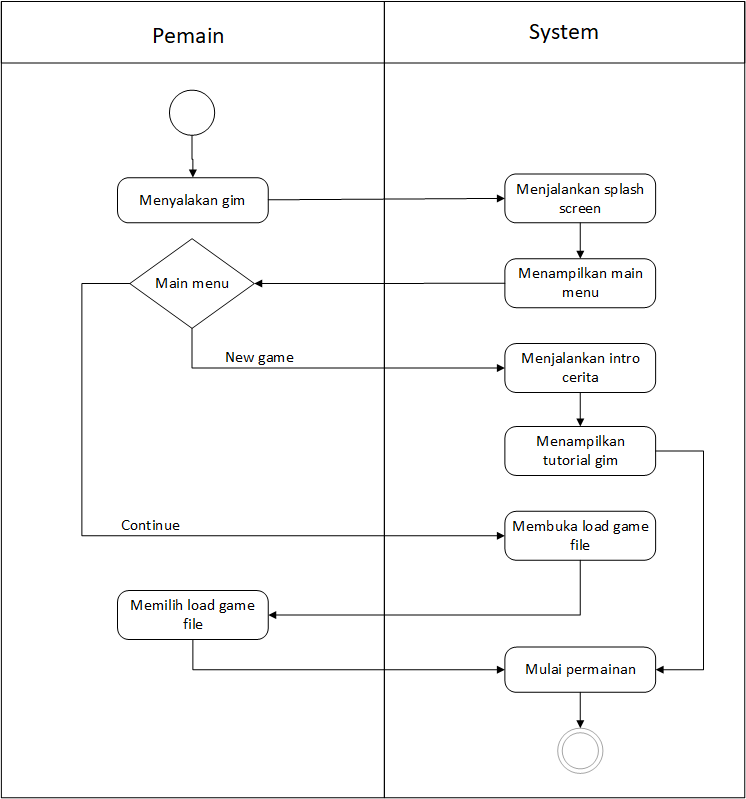
1. Pemain adalah satu – satunya aktor pada gim ini.
2. Saat pemain menjalankan gim, pemain dapat :
   1. Memulai gim
   2. Melanjutkan gim
   3. Mengganti opsi gim
3. Untuk bermain gim, pemain harus memulai gim atau melanjutkan gim terlebih dahulu.
4. Di saat bermain gim, pemain dapat :
   1. Melihat inventaris
   2. Melihat kemampuan aktor
   3. Mengganti perlengkapan aktor
   4. Melihat parameter aktor
   5. Menyimpan data perkembangan gim
5. Menggunakan barang dan melihat perlengkapan yang dimiliki merupakan bagian dari melihat inventaris.
6. Menggunakan sihir dan menggunakan kemampuan khusus dapat diakses di saat pemain melihat kemampuan aktor
7. Pemain dapat mengganti opsi pada gim sebelum atau di saat pemain bermain gim
8. Mengganti volume BGM dan Mengganti volume SE merupakan bagian dari mengganti opsi pada gim.

### 3.4.3 Activity diagram

Pada bagian ini terdapat *activity diagram* untuk setiap fitur yang ada pada gim berdasarkan *use case diagram* yang telah dibuat sebelumnya. Diagram ini akan digunakan untuk menggambarkan proses aktivitas pengguna dan sistem dimulai dari pengguna sebagai pemain memulai menjalankan gim, proses ketika pemain mulai bermain gim, proses menyimpan data dalam gim, proses dimana pemain mengganti pengaturan pada gim, dan proses interaksi antara musuh dengan pemain.

#### 3.4.4.1 Activity diagram memulai gim

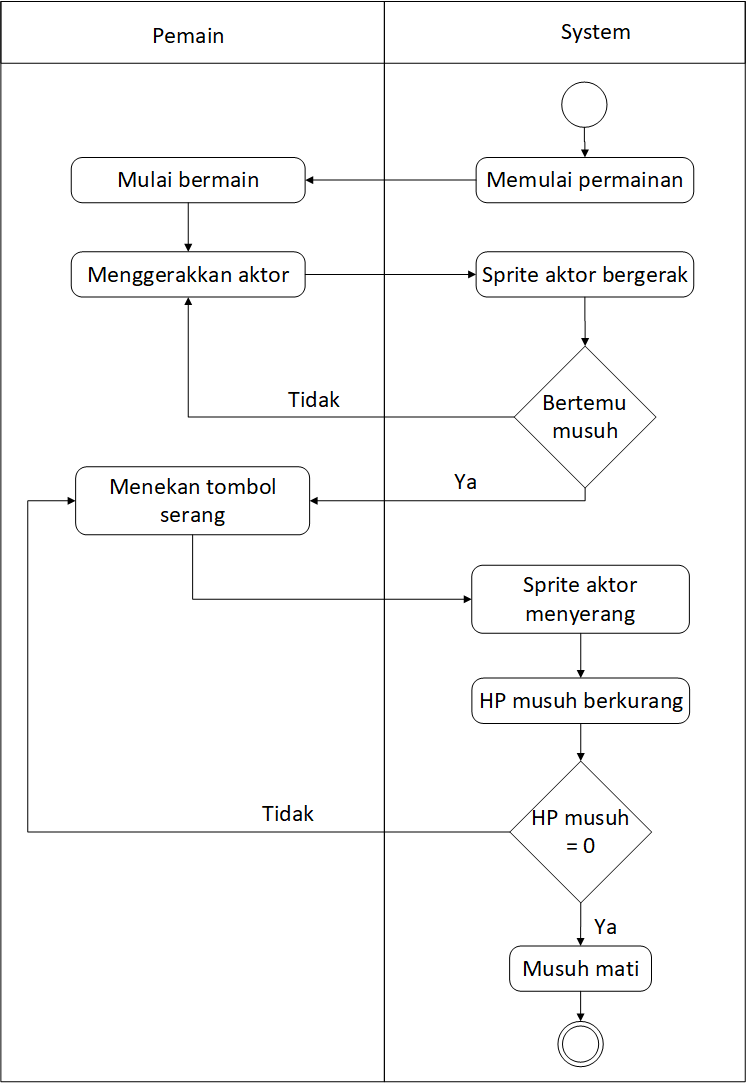
Diagram aktivitas memulai gim akan menunjukkan proses mulai dari pemain menjalankan program gim sampai dengan sistem memulai permainan yang ada. Saat program gim dijalankan maka sistem akan menjalankan *splash screen* lalu menampilkan *main menu*. Pada *main menu* pemain dapat memulai gim baru (*new game)* atau melanjutkan gim (*continue)* dari *save game* yang ada. Jika pemain memulai gim baru maka sistem akan menjalankan intro cerita dan tutorial gim. Setelah tutorial gim selesai maka sistem akan memulai permainan. Namun jika pemain memilih melanjutkan gim maka sistem akan menampilkan *load game file.* Setelah pemain memilih salah satu *load game file* yang ada maka sistem akan memulai permainan.



##### Gambar 3.27 *Activity Diagram* Memulai Gim

#### 3.4.4.2 Activity diagram bermain gim

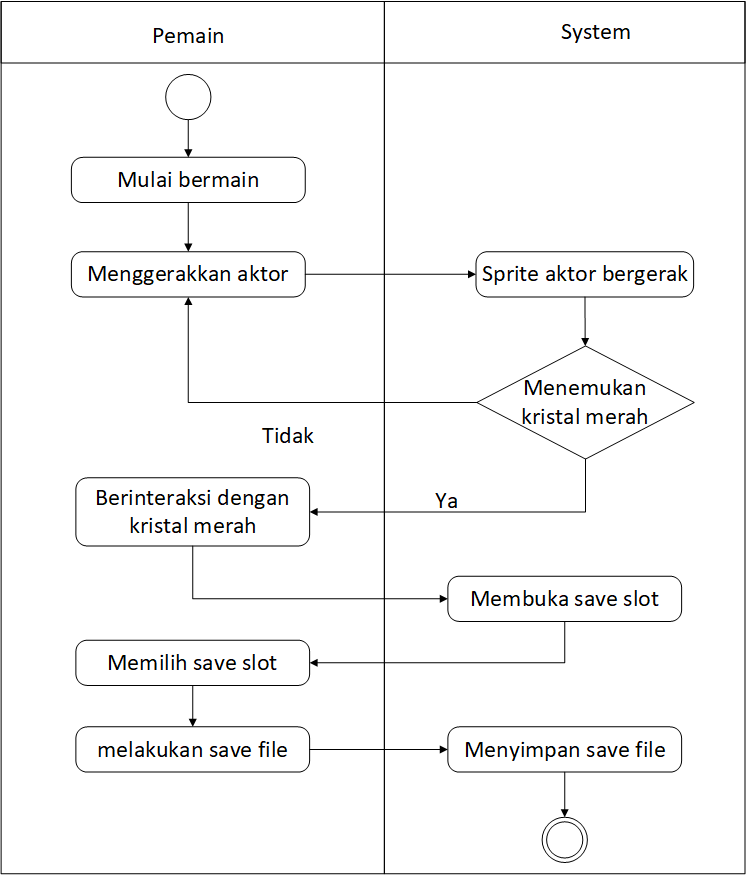
Diagram aktivitas bermain gim dimulai setelah sistem memulai permainan. Setelah permainan dimulai, pemain dapat bermain gim. Di saat bermain gim, pemain dapat menggerakan *sprite* aktor dengan menekan tombol panah pada *keyboard*. Jika pemain bertemu dengan musuh maka pemain dapat menekan tombol serang (tombol Z) agar *sprite* aktor melakukan serangan pada *sprite* musuh. Jika HP musuh sudah habis maka *sprite* musuh akan menghilang dari *map.*



##### Gambar 3.28 *Activity Diagram* Bermain Gim

#### 3.4.4.3 Activity diagram menyimpan data gim

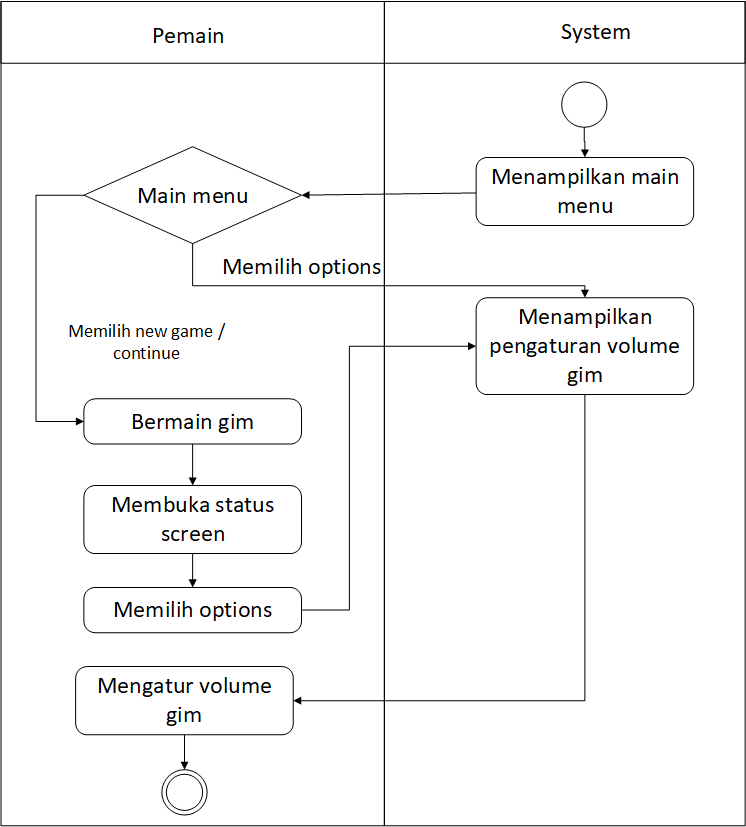
Diagram aktivitas penyimpanan data gim (*save game)* akan menjelaskan proses bagaimana pemain melakukan *save game* di saat pemain bermain gim. Pemain dapat menemukan kristal merah yang berfungsi untuk melakukan *save game*. Ketika pemain berinteraksi dengan kristal merah tersebut, sistem akan membuka *save* slot di mana pemain dapat memilih tempat untuk melakukan *save game*.



##### Gambar 3.29 *Activity Diagram* Menyimpan Perkembangan Gim

#### 3.4.4.4 Activity diagram mengganti pengaturan pada gim

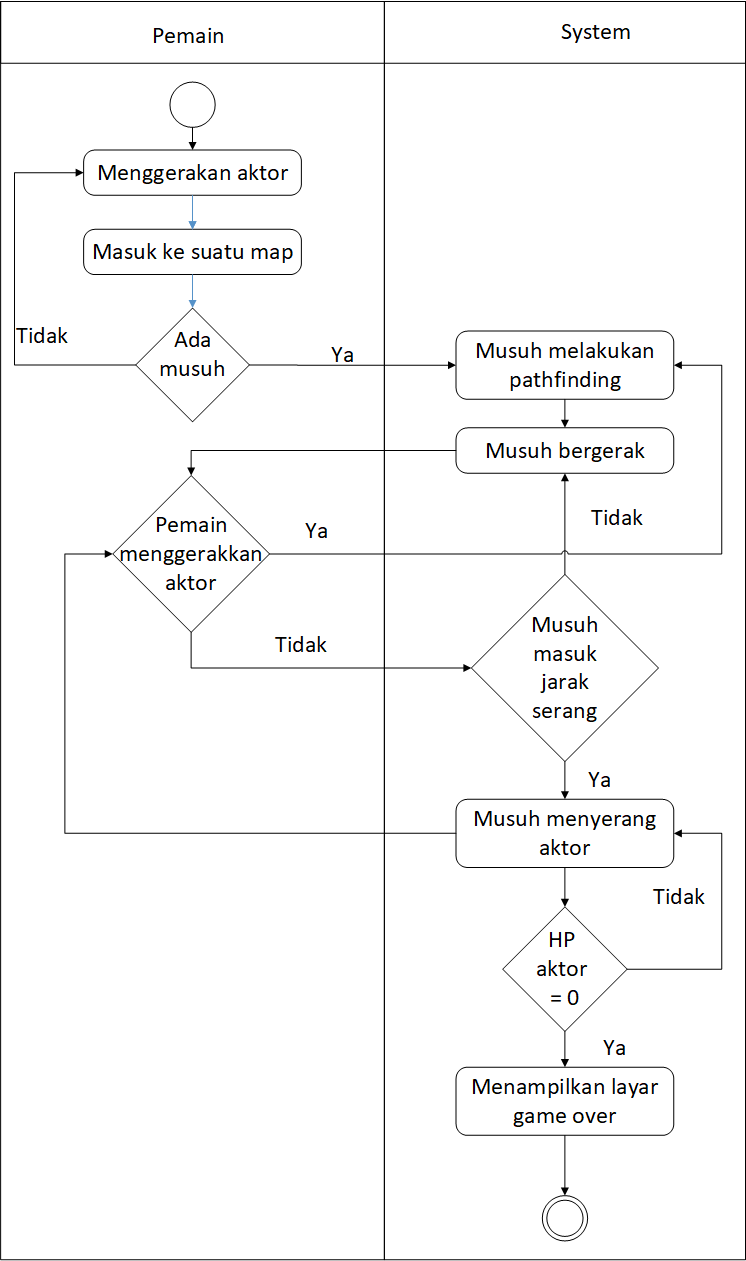
Diagram aktivitas mengganti pengaturan pada gim akan menunjukkan bagaimana pemain mengakses *menu options.* Untuk mengakses *menu options,* pemain dapat mengaksesnya melalui *main menu* secara langsung atau *status screen* dengan menekan tombol X di saat gim sudah dimulai*.* Pada *menu options* pemain dapat mengganti volume gim.



##### Gambar 3.30 *Activity Diagram* Mengganti Pengaturan Pada Gim

#### 3.4.4.5 Activity diagram interaksi musuh dengan pemain

Diagram aktivitas ini akan menunjukkan bagaimana musuh yang digerakan sistem berinteraksi dengan aktor yang dijalankan oleh pemain. Ketika aktor memasuki *map* yang terdapat musuh, maka musuh akan melakukan *pathfinding* menggunakan algoritma A\* untuk menentukan jalan terpendek menuju posisi aktor. Musuh akan bergerak melalui jalan hasil dari *pathfinding.* Jika pemain menggerakan aktor maka musuh akan mengulangi melakakan *pathfinding* untuk menemukan lokasi aktor yang baru dan jalan yang baru. Jika musuh berada di dekat aktor atau masuk dalam jarak serangan musuh maka musuh akan melakukan serangan terhadap aktor. Musuh akan terus melakukan serangan sampai HP aktor menjadi 0. Jika HP aktor mencapai angka 0 maka akan terjadi *game over* dan permainan akan selesai.



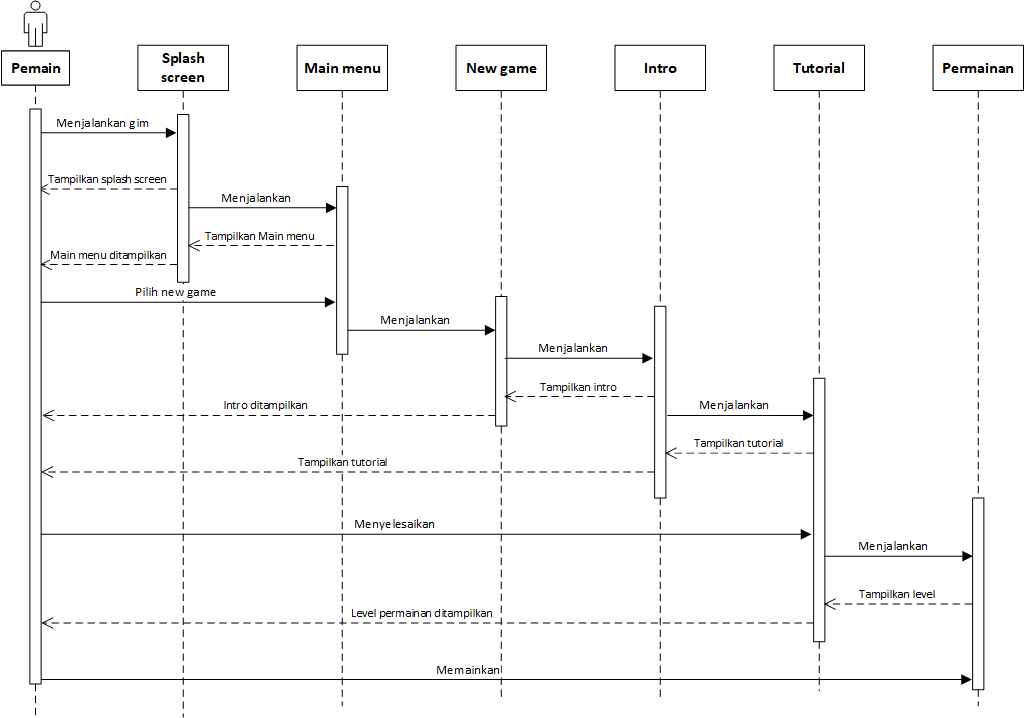
##### Gambar 3.31 *Activity Diagram* Interaksi Musuh Dengan Pemain

### 3.4.4 Sequence diagram

Berikut adalah *sequence diagram* yang akan digunakan untuk menggambarkan urutan proses dari fitur yang ada pada gim beserta interaksinya secara runtut dan jelas. Diagram ini akan digunakan untuk menjelaskan proses aliran pesan ketika pemain memulai permainan baru (*new game)*, pemain melanjutkan permainan (*continue),* mengganti pengaturan dan jalannya proses penyimpanan data pada gim.

#### 3.4.4.1 Sequence diagram new game

*Sequence diagram new game* menggambarkan interaksi pemain dengan objek – objek dalam gim yang diperlukan untuk memulai permainan baru.

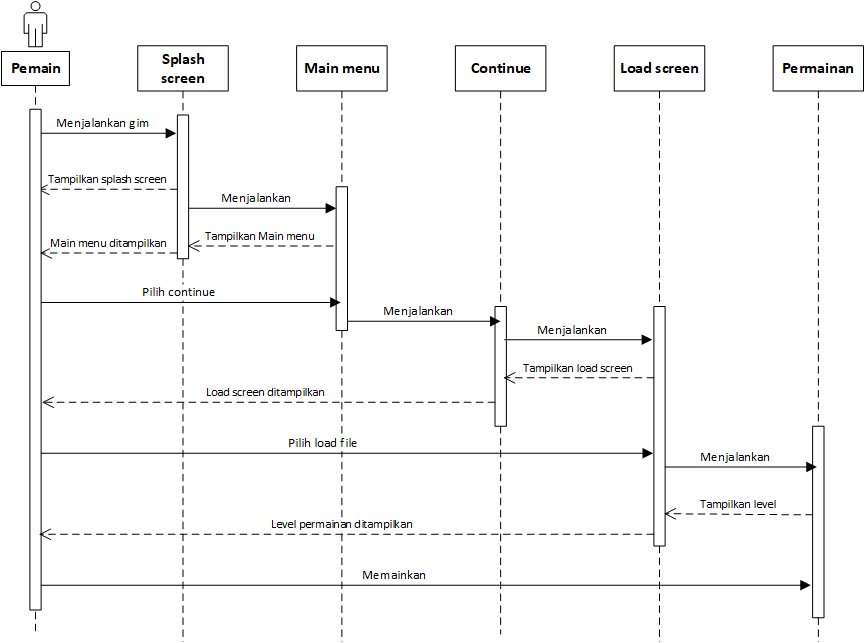


##### Gambar 3.32 *Sequence Diagram* Untuk *New Game*

Diagram di atas menggambarkan aliran pesan untuk memulai permainan baru. Diagram dimulai dari pemain menjalankan aplikasi gim sampai dengan pemain dapat memainkan gim.

#### 3.4.4.2 Sequence diagram continue

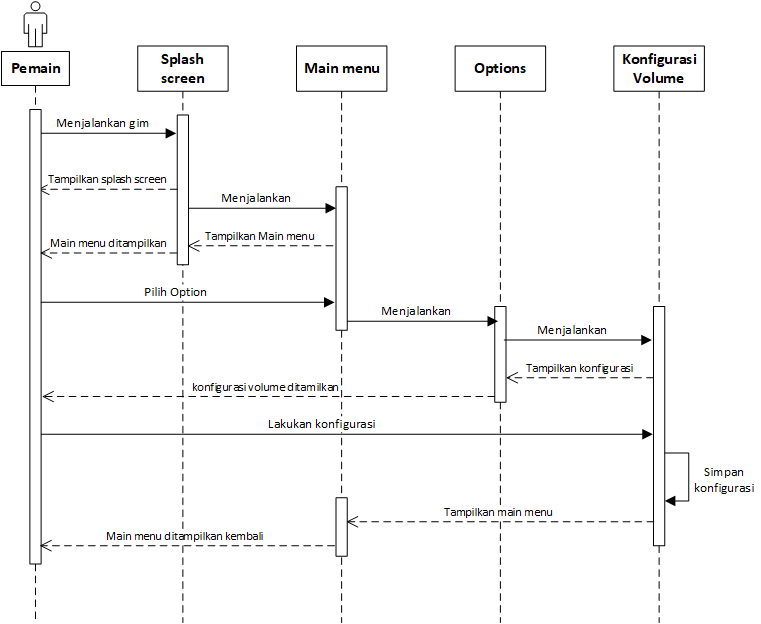
*Sequence diagram continue* hampir mirip dengan *sequence diagram new game.* Perbedaannya adalah pada objek *main menu,* pemain memilih *continue* untuk membuka *load screen* yang berisi *load file*. Ketika *load file* sudah selesai maka permainan dapat mulai dimainkan.



##### Gambar 3.33 *Sequence Diagram* Untuk *Continue*

#### 3.4.4.3 Sequence diagram options

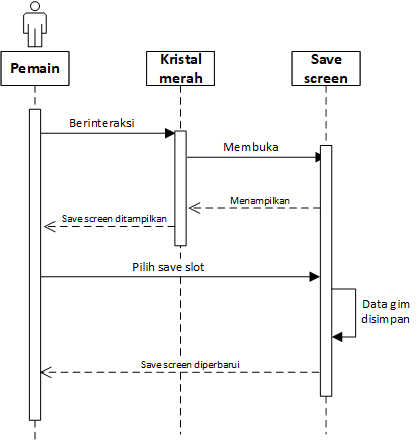
*Sequence diagram options* menggambarkan aliran pesan yang dimasukkan pemain untuk mengatur konfigurasi volume pada gim. Pada *main menu* pemain memilih *options* sehingga gim akan menampilkan layar di mana pemain dapat mengatur konfigurasi volume pada gim. Setelah konfigurasi selesai dilakukan, hasil konfigurasi itu akan disimpan dan gim akan menampilkan kembali layar *main menu* kepada pemain.



##### Gambar 3.34 *Sequence Diagram* Untuk *Options*

#### 3.4.4.4 Sequence diagram save game

*Sequence diagram save game* akan menunjukkan aliran pesan ketika pemain hendak menyimpan data permainannya di dalam gim. Diagram dimulai ketika pemain berinteraksi dengan objek untuk melakukan *save game* (kristal merah). Objek tersebut akan membuka *save screen* di mana pemain dapat memilih *save* slot untuk menyimpan data permainan mereka. Setelah data berhasil disimpan maka *save screen* akan diperbarui berdasarkan data permainan dari pemain.



##### Gambar 3.35 *Sequence Diagram* Untuk *Save Game*