Membangun Game Dungeon Crawling Dungeon of Honor menggunakan Algoritma A*

Annisa'ul Maghfirah [1)], Amelia Yusnita [2)], Kusnandar [3)]

Teknik Informatika, STMIK Widya Cipta Dharma Samarinda
Jl. M. Yamin No.25, Samarinda, 75123
E-mail: annisamgfr.work@gmail.com, amelia@wicida.ac.id, kusnandar@gmail.com

ABSTRAK

Sehubungan dengan perkembangan jaman teknologi, media hiburan berupa permainan *video game* telah berkembang sesuai dengan teknologi yang ada. Permainan atau *video game* telah menjadi salah satu media hiburan yang cukup populer diantara anak-anak maupun orang dewasa. Salah satu cara untuk membuat *video game* lebih menarik adalah dengan mengimplementasikan kecerdasan buatan berupa *pathfinding* dengan menggunakan algoritma *A**. Dalam pembangunan *video game Dungeon of Honor*, algoritma *A** digunakan untuk membantu dalam pembuatan map acak dan juga pergerakan musuh dimana *A** digunakan sehingga musuh dapat mencari jalan terdekat untuk menemukan pemain. Metode pengembangan sistem yang digunakan adalah *Multimedia Development Life Cycle* (MDLC) versi Luther-Sutopo yang memiliki 6 tahap pengembangan, yaitu *concept, design, content, assembly, testing* dan *distribution*. Berdasarkan hasil dari penelitian, dalam *video game Dungeon of Honor* algoritma *A** berhasil diimplementasikan dan permainan layak untuk didistribusi.

Kata Kunci: *Game, RPG, Dungeon Crawling, pathfinding, Algoritma A-Star, MDLC*

Developing Game Dungeon Crawling Dungeon of Honor using A Algorithm*

ABSTRACT

Due to the growth period of technology, entertaintment media such as video game had developed alongside the technology that was present. Video game had become one of entertaintment media that was popular enough among children as well as adults. One way to make video game more appealing is by implementing artificial intelligence such as pathfinding by using A algorithm. When developing video game Dungeon of Honor, A* algorithm was used to help in making randomized map and also enemies movement where A* was used so enemies can search the shortest path to reach the player. Development method that was used was Multimedia Development Life Cycle* (MDLC) *version Luther-Sutopo that has 6 development steps, which is concept, design, content, assembly, testing and distribution. Based on the result of research, within video game Dungeon of Honor A* algorithm A* has been implemented succesfully and the game is suitable for distribution.*

Keywords*: Game, RPG, Dungeon Crawling, pathfinding, A-Star Algorithm, MDLC*

1. PENDAHULUAN

Video game atau lebih biasa disebut *game* merupakan permainan elektronik yang melibatkan interaksi dengan antarmuka pengguna atau alat pemasukan (seperti *joystick, controller, keyboard* atau alat sensor gerak) untuk mengasilkan masukan visual dari perangkat tampilan yang biasanya berupa video. Ada pun jenis-jenis *game* berdasarkan *genre*-nya seperti *action, action-adventure, adventure, puzzle, role-playing* (*RPG*), *simulation* dan lain-lain.

Sejarah *video game* dimulai sejak awal 1950-an, ketika ahli komputer merancang permainan dan simulasi sederhana sebagai bagian dari penelitian. Pada tahun 1960, professor dan mahasiswa di *M.I.T* memainkan *tic-tac-toe 3D* dan *Moon Landing* menggunakan input kartu berlubang. *Video game* tidak menjadi popular sebelum tahun 1970 sampai 1980-an, ketika permainan arkade, konsol, serta meningkatnya kemampuan grafis komputer tersedia secara umum. Permainan *arcade* awal dikembangkan dari tahun 1972 hingga 1978, dan konsol generasi pertama juga dirilis sekitar tahun 1970-an dengan game *Pong* yang merupakan salah satu permainan pertama. Pada era ini permainan *video game* arcade (*arcade*) menjadi sangat popular.

Hampir semua game memiliki unsur *game adventure* walaupun tidak spesifik. *Game adventure* biasanya memiliki satu tokoh atau obyek utama yang dimainkan dan dijalankan secara langsung dari awal sampai akhir (Ahmad & Widodo, 2017). *Dungeon crawl* (penjelajahan dungeon) adalah salah satu jenis *scenario* dalam *role-playing game* (*RPG*) bertema fantasi dimana para pahlawan (*player*) harus menavigasi lingkungan labirin

(sebuah *dungeon*), melawan berbagai *monster*, memecahkan teka-teki dan menjarah harta apa pun yang mungkin mereka temukan. Tujuannya adalah untuk mencari jalan keluar dari labirin menuju tangga yang akan memindahkan pemain ke *level* selanjutnya, sampai tangga terakhir yang biasanya terdapat sesuatu yang karakter pemain inginkan.

Seperti yang disebutkan sebelumnya, terdapat *monster* yang menjadi hambatan dalam permainan dan sebagai musuh yang harus dikalahkan. Setelah dikalahkan, *monster* tersebut akan memberikan pahlawan *experience point* (*exp*) dan menjatuhkan item yang bisa digunakan oleh pemain. Musuh-musuh tersebut biasanya akan tetap muncul dan mengejar para pahlawan hingga mereka keluar dari *dungeon* atau *game over*.

Kecerdasan buatan (*AI*) adalah sistem komputasi untuk meniru kecerdasan alami seperti manusi, hewan dan lain-lain. Kecerdasan buatan sering kali diterapkan di berbagai jenis permainan yang bertujuan untuk membuat permainan tersebut lebih menarik. *AI* lebih sering digunakan pada komponen *enemies* (musuh), target maupun *non-playable character* (*NPC*), dimana komponen pemain dikendali langsung dengan bebas oleh pemain. Pada permainan "*Dungeon of Honor*" akan menerapkan *AI* pada komponen musuh untuk membuat eksplorasi peta permainan lebih menarik.

Dalam pergerakan tokoh utama maupun tokoh lain dalam permainan, dibutuhkan kecerdasan buatan (*AI*) agar permainan lebih menarik. *AI* merupakan faktor terpenting dalam menigkatkan *gameplay*. Pada dasarnya ada dua teknik pencarian dan pelacakan yang digunakan, yaitu pencarian buta (*blindsearch*) dan pencarian terbimbing (*heuristicsearch*) (Fallo & Bulu, 2022).

Salah satu metode untuk membuat musuh-musuh mengejar pahlawan adalah menggunakan *path-finding* algoritma *A** (*A-star*). Pemakaian algoritma *A** karena algoritma ini terkenal akan keefektifitasnya untuk mencari jarak terdekat ke tujuan (Octavian & Hermawan, 2023). Karena itu dalam pembuatan permainan *Dungeon of Honor* akan menggunakan algoritma *A** sebagai metode kecerdasan buatan.

Berdasarkan hasil penelitian "Implementasi Metode Pathfinding dengan Algoritma A* pada Game Rogue-Like Menggunakan Unity", unit dapat bergerak melewati halangan yang diberikan menuju titik akhir, hanya dapat bergerak menuju 1 titik tujuan, dan menggunakan petak yang diberi nilai paling kecil yang akan digunakan sebagai jalur yang dilewati (Agung, dkk, 2022). Dalam *Dungeon of Honor*, *monster* akan mendeteksi posisi pahlawan di dalam *dungeon*, lalu menentukan jalan terdekat untuk mencapai posisi pahlawan tersebut. Jika pemain menggerakkan pahlawan, maka musuh akan mendeteksi ulang posisi pahlawan dan mengulang proses *path-finding* tersebut hingga *monster* tersebut bertemu dan *battle* dengan pahlawan tersebut.

2. RUANG LINGKUP

Dalam penelitian ini permasalahan mencakup:

1. Menggunakan *engine* Godot Engine
2. Banyak *level* yang dibuat adalah 5
3. Berupa *single player*
4. Berbasis *desktop*
5. Bersifat *offline*

3. BAHAN DAN METODE

Adapun bahan dan metode yang digunakan dalam penelitian ini adalah sebagai berikut:

3.1 Algoritma *A** (*A-Star*)

Algoritma *A** (*A-Star*) merupakan salah satu dari algoritma pathfinding yang menerapkan teknik *heuristic* (teknik pencarian terbimbing). Algoritma ini meminimalkan total biaya lintasan, dan pada kondisi yang tepat akan memberikan solusi yang terbaik dalam waktu yang optimal (Ahmad & Widodo, 2017).

Algoritma ini menghitung jarak berdasarkan petak yang akan diberikan nilai untukmembuat jalur ke titik tujuan (Agung, dkk, 2022).

Pemakaian algoritma *A** karena algoritma ini terkenal akan keefektifitasnya untuk mencari jarak terdekat ke tujuan. Sehingga algoritma *A** ini digunakan ke *NPC* musuh. Dibandingkan dengan algoritma *pathfinding* yang lainnya, *A** menggunakan fungsi heuristik yang mengutamakan *nodes* yang paling bagus dari yang lainnya sehingga proses cepat dan efektif (Octavian & Hermawan, 2023).

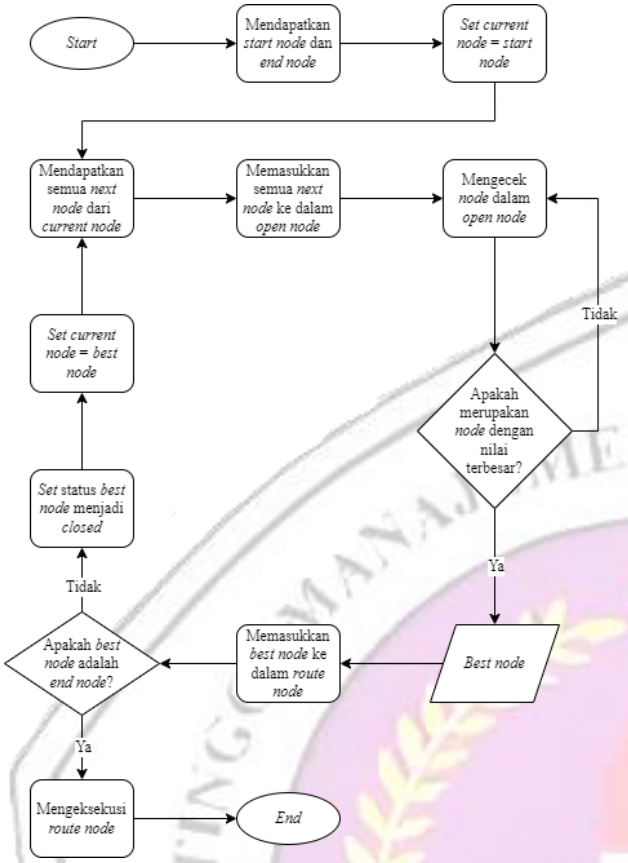
Algoritma A* adalah algoritma berbasis grid. Titik awal dan titik akhir dengan informasi jalur dalam peta yang dihasilkan dipilih untuk menjalankan algoritma A* (Agung, dkk, 2022).

$$f_{(v)} = h_{(v)} + g_{(v)} \quad (1)$$

Diketahui $h_{(v)}$ (1) adalah jarak heuristik dari sel ke titik akhir dan $g_{(v)}$ adalah panjang jalur dari titik awal ke titik akhir melalui urutan sel yang dipilih. Setiap node atau sel yang berdekatan dievaluasi oleh nilai $f_{(v)}$. Sel dengan nilai $f_{(v)}$ terendah dipilih sebagai urutan dari jalur (Agung, dkk, 2022).

Algoritma *A** *rise* (membangkitkan) *node* yang paling mendekaati solusi (*next node*). *Node* ini kemudian disimpan penerusnya (*successor*) ke dalam list (*open list*) sesuai dengan urutan yang paling mendekati solusi terbaik. Kemudian, *node* pertama pada *list* diambil dan menjadi *current node* (*node* sekarang), *successor*-nya dibangkitkan dan kemudian disimpan ke dalam *list* sesuai urutan yang terbaik. *Node list* ini disebut dengan *route node* (rute *node*). Ketika *current node* telah mencapai *end node* (*node* terakhir/tujuan), *route node* dieksekusi dengan membuat rute dari *start node* ke *end node* sesuai *node* yang ada di *route node*.

Berikut adalah diagram *flowchart* bagaimana algoritma *A** bekerja:



Gambar 1. Diagram *Flowchart* Algoritma *A**

Keuntungan dari algoritma ini adalah bahwa jarak yang digunakan dapat dimodifikasi atau jarak lain dapat ditambahkan kembali (Agung, dkk, 2022).

3.2 *Godot Engine*

Godot engine adalah salah satu alternatif *game engine* berbasis *opensource* berlisensikan *The MIT License* (*MIT*) *sponsored by Okam Studio*. *Game engine* ini sangat *powerfull* meski masih dalam tahap pengembangan, dengan *engine* ini kita dapat membuat *game 2D* atau *3D* (Rahayu, 2019).

Godot mengizinkan pengembang *video game* untuk membuat *game 2D* atau *3D* menggunakan beberapa bahasa pemrograman. *Godot* menggunakan *node* hirearki untuk mempermudah pengalaman dalam pengembangan. *Classes* dapat diperoleh dari tipe *node* untuk membuat tipe *node* yang lebih terspesialisasi yang dapat mewarisi perilaku *node* sebelumnya. *Node-node* tersebut akan diatur dalam "*scenes*", yang dapat digunakan ulang, *instanceable*, diwarisi, dan dapat dikelompokkan dalam sebuah sarang. Semua *game resource* termasuk skrip dan grafik aset disimpan sebagai bagian sistem *file* daripada *database*.

Godot memiliki basa pemrograman sendiri yang dinamakan *GDScript*, yang secara sintak mirip dengan *Python.*

3.3 *Flowchart*

Alat bantu dalam pembuatan permainan *Dungeon of Honor* adalah *flowchart*. Menurut Indrajani (2011), *flowchart* merupakan penggambaran secara grafik dari langkah-langkah dan urutan prosedur suatu program; Biasanya mempengaruhi penyelesaian masalah yang khususnya perlu dipelajari dan dievaluasi lebih lanjut.

Flowchart dapat digunakan untuk menyajikan kegiatan manual, kegiatan pemrosesan ataupun keduanya. *Flowchart* merupakan rangkaian simbol-simbol yang digunakan untuk mengkontruksi. Adapun petunjuk pembuatan *flowchart* adalah sebagai berikut:

1. *Flowchart* digunakan atau digambarkan dengan halaman atas ke bawah dan dari kiri ke kanan.
2. Kegiatan yang digambarkan harus dapat dimengerti oleh penggunanya.
3. Harus ada kejelasan untuk awal dan akhirnya.
4. Tahapan dari aktivitas harus diuraikan dengan menggunakan deskripsi kata kerja.
5. Tahapan langkah dari kegiatannya harus berada pada urutan yang tepat.
6. Ruang lingkup kegiatan yang berjalan harus ditelusuri dengan seksama.
7. Disarankan penggunaan symbol-simbol *flowchart* yang baku.

3.4 *Multimedia Development Life Cycle* (*MDLC*)

Metode pengembangan sistem yang diterapakan pada penelitian ini menggunakan metode *Multimedia Development Life Cycle* (*MDLC*) dari versi Luther-Sutopo. Metode ini memiliki 6 tahapan, yaitu *concept*, *design*, *material collecting*, *assembly*, *testing*, dan *distribution*.

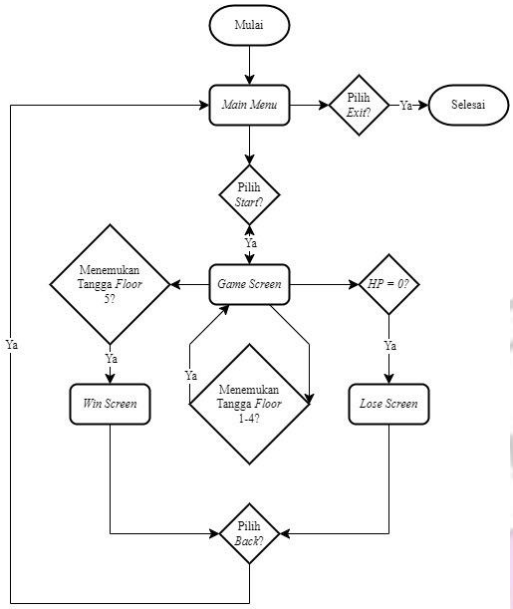
1. *Concept*: Merupakan tahapan awal dalam menentukan ide utama dari permainan, tujuan dan menentukan siapa yang akan memainkan game.

2. *Design*: Berisi tentang alur cerita, kode program, spesifikasi perangkat keras, tampilan *UI* (*User Interface*), kebutuhan material dan bahan yang akan digunakan dalam pengembangan permainan, serta pembuatan *storyboard*.

   Alur cerita dari permainan adalah seorang kesatria yang masuk ke dalam *dungeon* berisi penuh dengan *monster*, menjelajahi labirin *dungeon*, dan mencari tangga ke *level* selanjutnya untuk mendapatkan kehormatan diri.

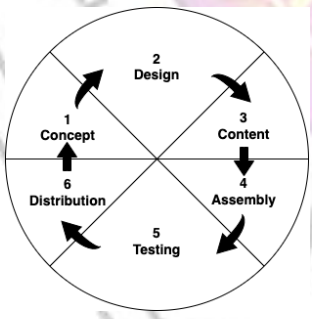
   Narasi dari game adalah "Seorang kesatria ingin menjadi kesatria bernama. Dia mendengar bahwa ada sebuah dungeon yang dapat mengabulkan keinginan seseorang yang berhasil tiba di akhir dungeon tersebut. Maka kesatria itu pergi ke dungeon tersebut untuk menghadapi tantangan tersebut. Apakah kesatria itu dapat tiba di akhir dungeon tersebut?"

   Berikut adalah *flowchart storyboard* yang digunakan untuk memudahkan pembuatan *video game*:



Gambar 2. Diagram *Flowchart Storyboard*

3. *Material Collecting/Content*: Pada tahapan ini dilakukan proses pengumpulan bahan-bahan dan alat-alat yang telah ditentukan dalam proses tahapan sebelumnya.
4. *Assembly*: Dalam tahapan ini, dilakukan peroses pengembangan permainan menggunakan bahan-bahan dan alat-alat yang telah dikumpulkan pada tahap sebelumnya.
5. *Testing*: dilakukan pengujian hasil dari proses tahapan assembly.
6. *Distribution*: Pada tahap ini dilakukan penyebarluasan permainan yang telah diuji dan tidak membutuhkan perbaikan.



Gambar 3. Diagram Metode Pengembangan Sistem *Multimedia Development Life Cycle* (MDLC) versi Luther-Sutopo

3.5 Kuesioner

Kuesioner dilakukan untuk mendapatkan data pada tahapan pengembangan testing. Pengajuan pertanyaan akan diberikan dengan menggunakan lembar kuisioner tester dimana responder akan menjawab pertanyaan terstruktur berupa kriteria-kriteria dalam permainan. Kriteria-kriteria tersebut adalah fungsional, stabilitas, banyak kesalahan, performa pada perangkat keras, navigasi UI, kontrol karakter, gerakan karakter, gerakan musuh, kesulitan musuh, dan nilai keseluruhan permainan tersebut.

Tabel 1. Skala Penilaian

Responsi	Nilai Angka
Sangat Baik	4
Baik	3
Kurang Baik	2
Sangat Kurang Baik	1

Dari penilaian di atas, maka setiap responsi dari responden akan dikumpulkan berdasarkan keriteria-kriteria yang telah ditentukan dan dicari rata-ratanya. Berdasarkan dari hasil penghitungan maka akan ditentukan kriteria mana yang perlu diperbaiki. Setiap kriteria dapat diberikan pertimbangan seperti berikut ini:

Tabel 2. Pertimbangan Revisi

Rentang Nilai	Hasil Pertimbangan
$3 \leq K < 4$	Tidak perlu revisi
$2 \leq K < 3$	Perlu sedikit revisi
$1 \leq K < 2$	Perlu banyak revisi
$0 \leq K < 1$	Perlu semua revisi

Permainan akan dianggap layak untuk distribusi jika pertimbangan pada semua kriteria adalah “Tidak perlu revisi”.

4. PEMBAHASAN

Design dari permainan yang ingin dibangun adalah sebagai berikut:

4.1 Kebutuhan Material dan Bahan

Berikut adalah gambar *sprite* dan *tileset* yang digunakan dalam pembangunan permainan:

Tabel 3. Penjelasan *Sprite*

Gambar	Keterangan
	*Sprite* Pemain (*Player*)
	*Sprite* Musuh 1 (*Enemy* 1)
	*Sprite* Musuh 2 (*Enemy* 2)
	*Tileset* Bayangan (*shadow*) area *unexplored*
	*Tileset* Pintu (*Door*) memisahkan lantai *room* dan *pathway*
	*Tileset* Tangga (*Stairs*) untuk ke *level* selanjutnya

Lanjutan Tabel 3. Penjelasan *Sprite*

Gambar	Keterangan
	*Tileset* Lantai (*Ground*) di mana *player* dapat berjalan
	*Tileset* Dinding (*Wall*) sebagai batas *room*
	*Tileset* Batu (*Stone*) sebagai area di luar *room* yang tidak bisa diakses
	*Sprite Food* 1 yang memiliki efek acak
	*Sprite Food* 2 yang memiliki efek acak
	*Sprite Food* 3 yang memiliki efek acak
	Sprite Food 4 yang memiliki efek acak
	Gambar *Impairment* yang digunakan untuk mengurangi penglihatan

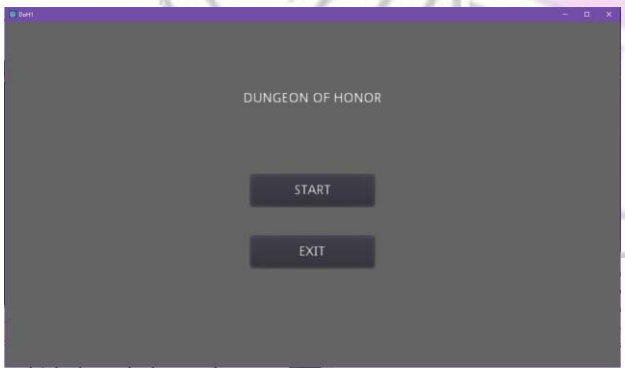
Berikut adalah daftar efek acak saat pemain (*player*) mengentuh salah satu *sprite food*:

Heal: nilai *HP* bertambah sejumlah 5.
Heal over time: nilai *HP* bertambah 1 selama 3 *turn*.
Poisoned: nilai *HP* berkurang 1 selama 3 *turn*.
Impairment: mengurangi penglihatan selama 10 *turn*.

Hasil *assembly* berdasarkan analisis dan perancangan adalah sebagai berikut:

4.2 Tampilan *Main Menu*

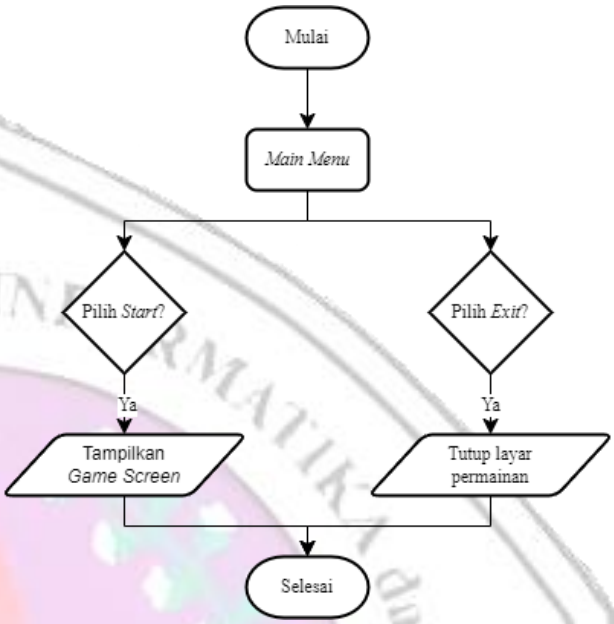
Pada tampilan ini akan menampilkan judul *game* "*Dungeon of Honor*", tombol *Start* dan tombol *Exit*. Tombol *Start* digunakan untuk memulai permainan dan memasuki *scene*/tampilan *game screen*. Sedangkan tombol *Exit* akan menutup layar *game*/permainan. Tampilan *main menu* dapat dilihat pada gambar 4.



Gambar 4. Tampilan *Main Menu*

Saat memasuki tampilan ini dari *main menu*, sistem akan memulai prosedur pembuatan map acak dimana bentuk map, letak *doors*, *pathway*, posisi awal *player*, *enemies*, *items*, dan *stairs* ditentukan.

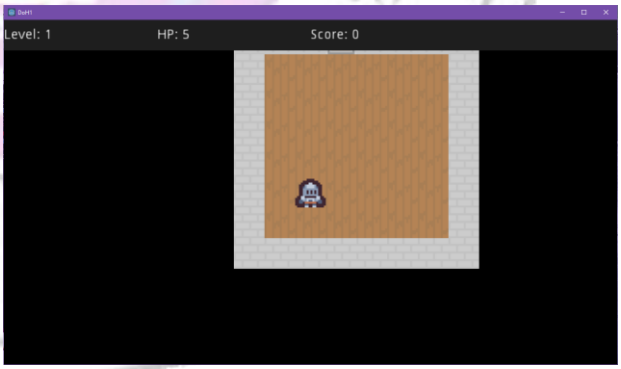
Berikut adalah gambar diagram *flowchart* dari tampilan *main menu*:



Gambar 5. Diagram *Flowchart Main Menu*

4.3 Tampilan *Game Screen*

Pada tampilan ini terdapat beberapa komponen utama, yaitu *status bar* dan *gamefield*. Pada *status bar* terdapat keterangan *Level*, *HP*, dan *Score*. Keterangan *level* digunakan untuk menunjukkan berapa kali tampilan ini dimasuki tanpa mencapai *hp* = 0. *hp* menunjukkan *Hit Point* yang akan menentukan jika *player* masih hidup atau tidak, dan *score* menentukan nilai skor pemain. Tampilan *game screen* dapat dilihat pada gambar 6.



Gambar 6. Tampilan *Game Screen*

Saat memasuki tampilan ini dari *main menu*, sistem akan memulai prosedur pembuatan map acak dimana bentuk map, letak *doors*, *pathway*, posisi awal *player*, *enemies*, *items*, dan *stairs* ditentukan.

Jika *player* menyentuh *enemies* maka status *hp* akan bertambah atau berkurang sesuai urutan gerak. Jika *player* menyentuh *items* maka beberapa hal dapat terjadi, seperti *hp* bertambah atau berkurang dan *field-of-view* berkurang. Hanya satu hal dapat terjadi pada satu *item*. Jika *player* menyentuh *stairs* maka salah satu dari dua hasil akan terjadi.

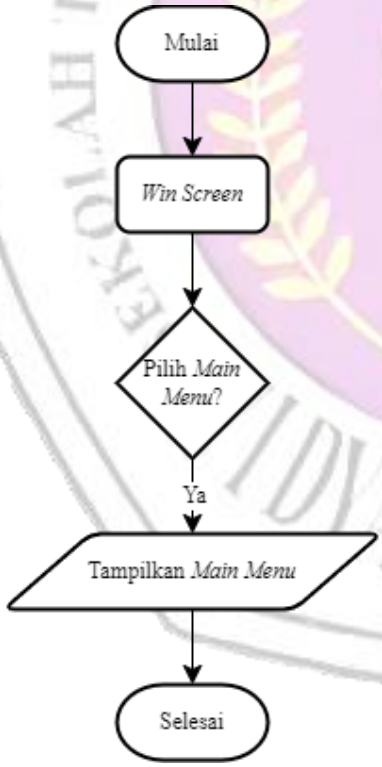
Setiap kali *player* memasuki ulang *game screen* maka status *level* akan bertambah satu sampai empat kali dimana selanjutnya akan memasuki *win screen*. Setiap kali *player* mengalahkan *enemies*, menyentuh *items* dan *stairs* maka status *score* akan bertambah.

Pembuatan map acak menggunakan algoritma *A** dimana algoritma digunakan untuk menentukan *pathway* sehingga setiap *room* terhubung satu sama lainnya. Algoritma *A** juga digunakan pada pergerakan *enemies* sehingga mereka dapat mengejar *player* setiap kali *player* bergerak.

Musuh akan *spawn* di dalam *room* dalam kondisi tidak aktif (tidak bergerak). Akan tetapi musuh akan mulai bergerak mengejar *player* saat *player* menyentuh *door* (pintu). Musuh akan selalu bergerak setelah *player* bergerak, baik berjalan atau menabrak dinding.

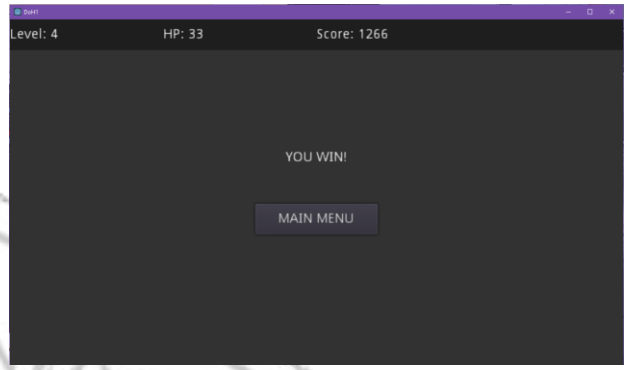
4.4 Tampilan *Win Screen*

Berikut adalah gambar diagram *flowchart* dari tampilan *win screen*:



Gambar 7. Diagram *Flowchart Win Screen*

Pada tampilan ini akan menampilkan tulisan ”*You Win!*” dan tombol *Main Menu*. Tombol *Main Menu* digunakan untuk memasuki *scene*/tampilan *main menu*. Tampilan *win screen* dapat dilihat pada gambar 8.

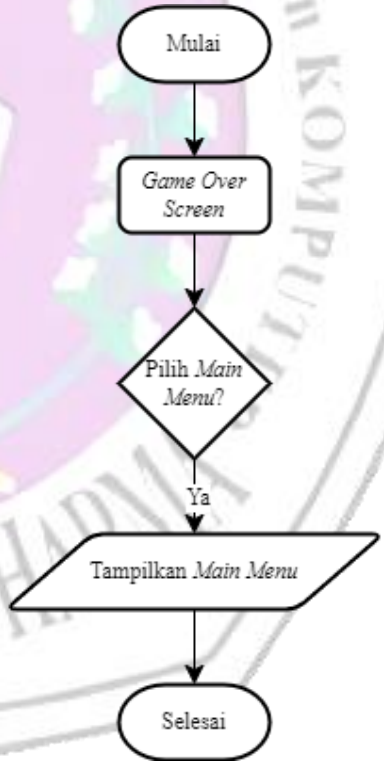


Gambar 8. Tampilan *Win Screen*

Player dapat masuk pada tampilan *win screen* jika status *hp* tidak 0 dan status *level* sudah menjadi 5 saat menyentuh *stairs* terakhir.

4.5 Tampilan *Game Over Screen*

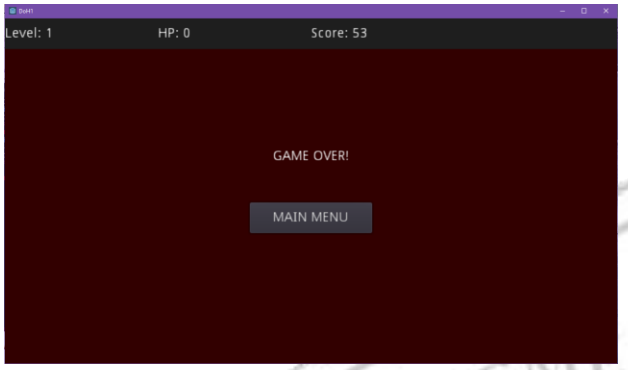
Berikut adalah gambar diagram *flowchart* dari tampilan *game over screen*:



Gambar 9. Diagram *Flowchart Game Over Screen*

Pada tampilan ini akan menampilkan tulisan ”*Game Over!*” dan tombol *Main Menu*. Tombol *Main Menu* digunakan untuk memasuki *scene*/tampilan *main menu*. Warna tampilan *game over screen* dibedakan dengan

tampilan *win screen* dengan menggunakan warna merah. Tampilan *win screen* dapat dilihat pada gambar 10.



Gambar 10. Tampilan *Game Over Screen*

Player dapat masuk pada tampilan *game over screen* jika status *hp* menjadi 0 saat disentuh oleh *enemies* atau karena mendapatkan salah satu efek menyentuh *items* yang dapat mengurangi *hp*.

Setelah *assembly* maka dilakukan proses *testing* dimana permainan (*video game*) yang telah dibuat akan dibagikan dengan kuesioner kepada 10 orang yang bersedia menjadi *tester* dan menjawab pertanyaan-pertanyaan pada kuesioner tersebut. *Tester* juga diminta untuk memberikan saran untuk pengembangan *game* kedepannya. Hasil *testing* ditampilkan pada tabel 3.

Tabel 4. Hasil *Beta Testing*

Pertanyaan	Penilaian				Total
	SB	B	KB	SKB	
Apakah *video game Dungeon of Honor* berfungsi dengan baik?	7	3			10
Bagaimana stabilitas *video game Dungeon og Honor*? Apakah ada *lag*?	10				10
Bagaimana banyak *glitch* atau kesalahan dalam *video game Dungeon of Honor*?	7	3			10
Bagaimana performa *video game Dungeon of Honor* pada komputer anda?	6	4			10
Bagaimana kemudahan navigasi *UI* dalam *video game Dungeon of Honor*?	4	6			10
Bagaimana kemudahan kontrol karakter pada *video game Dungeon of Honor*?	4	6			10
Bagaimana gerakan karakter pada *video game Dungeon of Honor*?	5	4	1		10
Bagaimana gerakan musuh pada *video game Dungeon of Honor*?	3	6	1		10
Bagaimana kesulitan musuh pada *video game Dungeon of Honor*?	5	3	2		10
Bagaimana menurut anda keseluruhan *video game Dungeon of Honor*?	4	6			10

Berdasarkan tabel 4 maka nilai rata-rata dari setiap pertanyaan adalah sebagai berikut:

Tabel 5. Perhitungan Rata-Rata Hasil *Testing*

No.	Perhitungan	Rata-Rata
1.	((7x4)+(3x3))/10	3,7
2.	(10x4)/10	4
3.	((7x4)+(3x3))/10	3,7
4.	((6x4)+(4x3))/10	3,6
5.	((4x4)+(6x3))/10	3,4
6.	((4x4)+(6x3))/10	3,4
7.	((5x4)+(4x3)+(1x2))/10	3,4
8.	((3x4)+(6x3)+(1x2))/10	3,2
9.	((5x4)+(3x3)+(2x2))/10	3,3
10.	((4x4)+(6x3))/10	3,4

Sesuai dengan penjelasan sebelumnya, maka pertimbangan dari hasi *testing* dapat dilihat pada tabel 5.

Tabel 6. Hasil Pertimbangan

Kriteria	Nilai Rata-Rata	Pertimbangan
Fungsionalitas	3,7	Tidak perlu revisi
Stabilitas	4	Tidak perlu revisi
Kesalahan	3,7	Tidak perlu revisi
Performa	3,6	Tidak perlu revisi
Navigasi *UI*	3,4	Tidak perlu revisi
Kontrol	3,4	Tidak perlu revisi
Gerakan Karakter	3,4	Tidak perlu revisi
Gerakan Musuh	3,2	Tidak perlu revisi
Kesulitan Musuh	3,3	Tidak perlu revisi
Nilai Keseluruhan	3,4	Tidak perlu revisi

Berdasarkan dari tabel di atas, maka *video game Dungeon of Honor* dianggap layak untuk tahap selanjutnya, yaitu tahap *distribution*. *Dungeon of Honor* akan didistribusikan secara *online* dan tidak dipungut biaya apapun.

5. KESIMPULAN

Dihasilkan aplikasi permaiana (*video game*) ber-*genre dungeon crawling* yang diberi nama *Dungeon of Honor* yang dapat berjalan pada perangkat *desktop*. *Game* ini mengangkat tema *role-playing game* (*RPG*) dimana pemain (*player*) bermain sebagai seorang satria yang menjelajah sebuah *dungeon* untuk mendapatkan kekuatan baru. *Game* ini dibangun dalam 6 tahap, yaitu *concept*, *design*, *content*, *assembly*, *testing*, dan *distribution*.

Algoritma *A** (*A-Star*) berhasil diterapkan sebagai alat pembuatan map acak dan juga alat menentukan arah pergerakan musuh (*enemies*). Algoritma *A** digunakan untuk menemukan dua titik *door* dan membuat *pathway* antara *room*. Algoritma *A** juga digunakan untuk menghitung jalan terdekat musuh untuk mengejar *player* agar permainan lebih menarik.

Dari hasil *testing* yang dilakukan dan diolah, *video game* ini layak untuk didistribusi dan dimainkan oleh orang-orang yang berminat.

6. SARAN

Aplikasi ini hanya sebatas Aplikasi *Game Dungeon Crawling* biasa. Apabila ingin dikembangkan dapat menambah misi seperti "*no item-run*" atau misi lainnya, menambahkan efek *items*, menambah variasi *enemies*, menambah *boss enemy*, menambah pilihan gerakan *player* dan lain-lain. Aplikasi ini hanya menampilkan *game dungeon crawling* yang sangat sederhana, dimana banyak hal-hal yang dapat ditambahkan untuk membuat *game* ini lebih kompleks dan menghibur.

7. DAFTAR PUSTAKA

Rozaq, A. (2019). Artificial Intelligence Untuk Pemula. http://eprint.unipma.ac.id/107/

Fallo, D. Y., & Bulu, V. R. (2022). Penerapan Algoritma A Star (A*) Pada Game Labirin. Jurnal Pendidikan Teknologi Informasi (JUKANTI), 5(1), 118-124. https://ojs.cbn.ac.id/index.php/jukanti/article/view/459

Agung, E. (2022). Implementasi Metode Pathfinding dengan Algoritma A* pada Game Rogue-like menggunakan Unity. Indonesia Journal on Computing (Indo-JC), 7(3), 81-94. http://socj.telkomuniversity.ac.id/ojs/index.php/indojc/article/view/677

Octavian, F., & Hermawan, L. (2023). Penerapan Algoritma Pathfinding A* dalam Game Dual Legacy berbasis Android. Jurnal Buana Informatika, 14(01), 20-29. https://ojs.uajy.ac.id/index.php/jbi/article/view/6928

Roguelike/RPG Pack, Kenney https://kenney.nl/assets/roguelike-rpg-pack

Tiny Dungeon, Kenney https://kenney.nl/assets/tiny-dungeon

Malabay, M. (2016). Pemanfaatan Flowchart Untuk Kebutuhan Deskripsi Proses Bisnis. J. Ilmu Komput, 12(1), 21-26.

Rahayu, P. (2019). Pengembangan Media Gapeto Berbasis Godot Engine sebagai alat evaluasi pembelajaran pada materi bangun ruang sisi datar. Digital Library: UIN Sunan Ampel, 1-96.

Widodo, W., & Ahmad, I. (2017). Penerapan algoritma A Star (A*) pada game petualangan labirin berbasis android. Khazanah Informatika: Jurnal Ilmu Komputer Dan Informatika, 3(2), 57-63. https://journals.ums.ac.id/index.php/khif/article/view/5221

DHARMA, W. C. (2015). Pedoman Penulisan Usulan Proposal dan Skripsi Jenjang Strata Satu (S1). Samarinda: STMIK Widya Cipta Dharma.