BAB III
ANALISIS DAN PERANCANGAN

3.1 Analisis

Analisis dilakukan untuk mendapatkan tujuan mengidentifikasi masalah dan kebutuhan yang dibutuhkan dari analisis data primer dan data sekunder. Analisis pada *game "The Fabrique"* yaitu:

1. Pergerakan NPC *(Non-Playable Character)* yang masih bergerak pada satu arah.
2. Peningkatan interaksi karakter dalam *game* untuk meningkatkan kualitas dari *game*.
3. Kepuasan pengguna dalam memainkan *game* yang memiliki gerakan yang *responsive* dan menantang.

3.1.1 Identifikasi Masalah

Berdasarkan identifikasi masalah yang sudah ditemukan di atas maka peneliti menggambarkan masalah secara umum yang akan dijabarkan melalui tabel berikut:

Tabel 3.1 Identifikasi Masalah

Masalah	Penyebab	Dampak
Perilaku NPC yang masih statis dalam pengulangan putaran jalur yang tetap	NPC tidak memiliki aturan sistem yang bergerak secara dinamis	NPC tidak mampu bergerak bebas saat terdapat halang rintang

3.1.2 Identifikasi Kebutuhan

Identifikasi kebutuhan dilakukan untuk mempersiapkan berbagai aspek yang mendukung jalannya penelitian. Berikut adalah beberapa kebutuhan yang telah diidentifikasi:

1. Pengembangan *Game*
   a. Platform Pengembangan : *Construct* 3 digunakan untuk merancang dan membangun *game "The Fabrique"*.

b. Desain Antarmuka Pengguna: Figma digunakan untuk mendesain UI/UX dari *game*.
c. Perangkat Pengujian: Smartphone digunakan untuk menguji *game* pada platform *mobile*.

2. Pengelolaan Proyek
   a. Dokumentasi Konsep:
      - Konsep *Game*: Deskripsi lengkap mengenai cerita, *gameplay*, karakter, dan dunia dalam *game*.
      - Konsep Algoritma: Penjelasan mengenai algoritma *Pathfinding* A* dan bagaimana diimplementasikan dalam *game*.
   b. *Library Pathfinding*: Modul yang mendukung implementasi algoritma *Pathfinding* dalam *Construct* 3.

3.1.3 Pemecahan Masalah

Berdasarkan uraian di atas, pemecahan masalah yang dapat dilakukan untuk mendukung pengembangan *game* dengan kecerdasan buatan adalah dengan menggunakan metode *Pathfinding*, yang penting untuk mengatasi permasalahan yang telah diidentifikasi. Metode *Pathfinding* digunakan untuk melakukan deteksi target yang akan dilakukan oleh NPC *(Non-Playable Character)*. Sehingga dengan implementasi metode *Pathfinding* ini diharapkan NPC *(Non-Playable Character)* dapat bergerak secara pintar.

3.2 Perancangan

Perancangan *game* yang dilakukan saat membuat *game "The Fabrique"* ini dengan memilih metode *Pathfinding* menggunakan algoritma A* sebagai solusi untuk deteksi jalur dalam pengembangan *game*. Pemilihan ini didasarkan pada kemampuan algoritma A* untuk menghasilkan jalur yang optimal, memungkinkan NPC *(Non-Playable Character)* untuk mencapai target dengan waktu yang lebih efisien dan tepat. Selain itu, penerapan metode *Pathfinding* menggunakan algoritma A* diharapkan dapat menghindari potensi *bug* yang dapat mempengaruhi pengalaman bermain pemain seperti bertabrakan dengan halangan. Dengan demikian, penggunaan metode *Pathfinding* menggunakan algoritma A* dalam

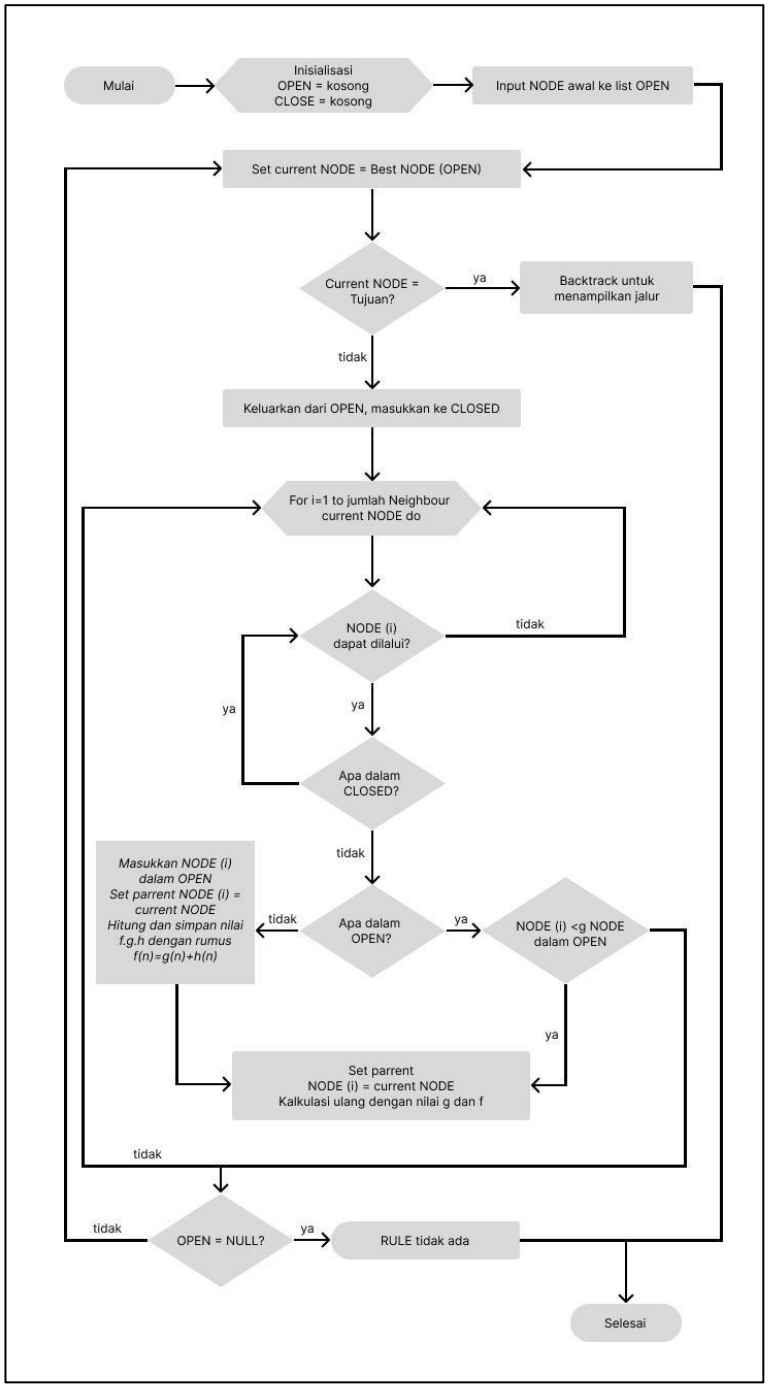
game The Fabrique diharapkan dapat menciptakan pengalaman bermain yang lebih dinamis dan interaktif.

3.2.1 Perancangan Sistem

Dari hasil identifikasi masalah pada *game* yang berjalan dapat menjadi acuan untuk merancang *game* yang dapat mempermudah peneliti dalam menghasilkan *game* dengan sistem yang optimal. Berikut adalah perancangan sistem yang akan digunakan dalam implementasi *Pathfinding* A* pada *game "The Fabrique"*.

3.2.1.1 *Flowchart* Algoritma A*

Dalam *Flowchart* algoritma A*, terdapat serangkaian langkah untuk menemukan jalur terpendek antara dua titik dalam peta *game*. Langkah pertama mencakup inisialisasi, di mana titik awal dan titik akhir ditetapkan. Kemudian, algoritma A* menggunakan heuristik dan jarak untuk mengevaluasi setiap langkah yang mungkin, memilih jalur dengan jarak terendah menuju tujuan. Langkah-langkah ini terus diulang hingga jalur terpendek ditemukan. Algoritma ini memungkinkan *game* untuk secara efisien menentukan jalur yang optimal antara posisi pemain dan target.

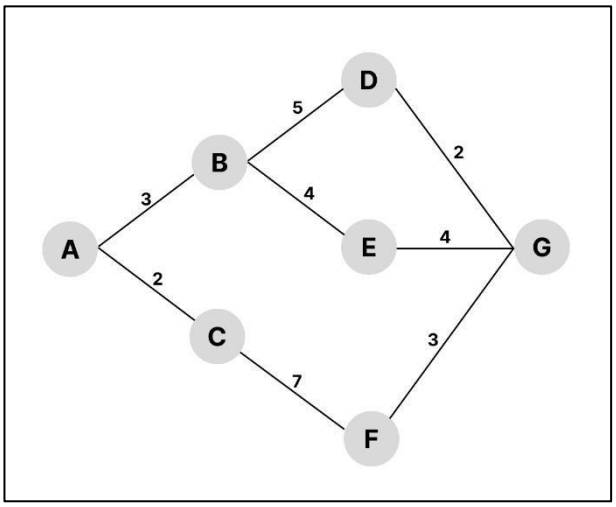


Gambar 3.1 *Flowchart* Algoritma A*

Dalam algoritma A*, perhitungan f(n) (Jarak Total) = g(n) (Jarak Aktual) + h(n) (Nilai Heuristik) adalah kunci dalam menentukan jalur terpendek. Fungsi f(n) digunakan untuk mengevaluasi total biaya suatu *node*, di mana f(n) adalah fungsi evaluasi total suatu *node*. g(n) adalah jarak aktual yang sudah ditempuh dari titik

awal ke *node* tersebut. h(n) adalah perkiraan jarak dari *node* saat ini ke *node* tujuan, yang biasanya dihitung menggunakan fungsi heuristik.

Saat menjalankan algoritma A*, nilai f(n) digunakan untuk memilih *node* selanjutnya dalam pencarian jalur. *Node* dengan nilai f(n) terendah dipilih sebagai kandidat terbaik untuk langkah berikutnya. Hal ini memungkinkan algoritma A* untuk mempertimbangkan jarak aktual yang telah ditempuh serta estimasi jarak yang tersisa menuju tujuan. Dengan demikian, algoritma A* dapat menentukan jalur terpendek antara titik awal dan tujuan dalam *game*. Untuk menghitung nilai f(n) = g(n) + h(n) hingga mencapai *node* tujuan G, diperlukan nilai g(n) dan h(n) untuk setiap *node* g(n) adalah jarak terkecil yang sudah ditempuh dari titik awal (A) ke *node* n. h(n) adalah estimasi jarak dari *node* n ke titik akhir (G). Dengan menghitung kedua nilai ini untuk setiap *node*, algoritma A* dapat mengidentifikasi jalur yang paling efisien dari titik awal ke tujuan.



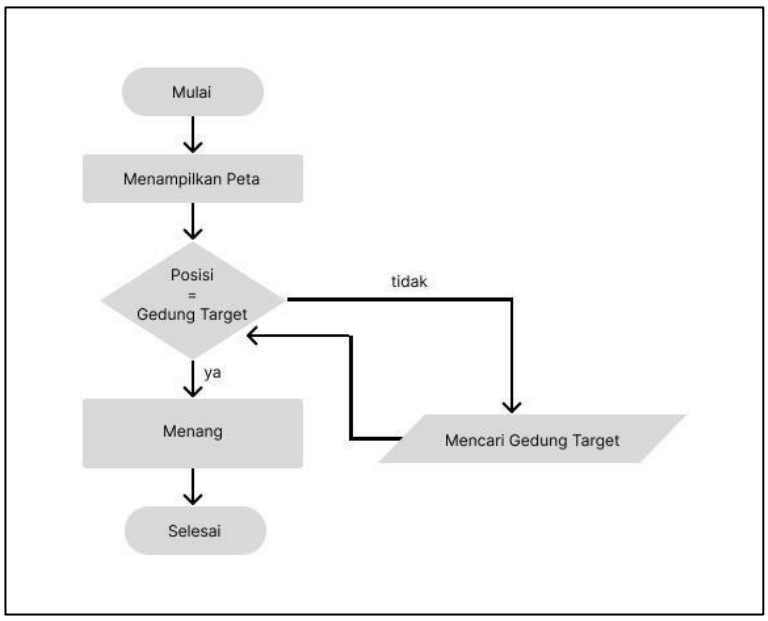
Gambar 3.2 Contoh *Node* Algoritma A*

Metode *Pathfinding* A* menggunakan pengecekan nilai di setiap *node* nya untuk menentukan jalur dengan nilai yang lebih efisien. Berikut adalah tabel perhitungan yang menjadi perhitungan *node* dalam metode *Pathfinding*.

Tabel 3.2 Perhitungan *Node* Algoritma A*

Keterangan	Perhitungan
Nilai g(n) dan h(n) untuk setiap *node*	g(A) = 0 (titik awal) g(B) = 3 (jarak dari A ke B) g(C) = 2 (jarak dari A ke C) g(D) = 8 (jarak dari A ke C, lalu dari C ke D) g(E) = 6 (jarak dari A ke C, lalu dari C ke E) g(F) = 9 (jarak dari A ke C, lalu dari C ke F) g(G) = 11 (jarak dari A ke C, lalu dari C ke F, lalu dari F ke G)
h(n) dihitung sebagai estimasi jarak dari setiap *node* ke titik akhir G	h(A) = 0 (karena A adalah titik akhir) h(B) = 6 (estimasi jarak dari B ke G) h(C) = 7 (estimasi jarak dari C ke G) h(D) = 2 (estimasi jarak dari D ke G) h(E) = 4 (estimasi jarak dari E ke G) h(F) = 3 (estimasi jarak dari F ke G) h(G) = 0 (karena G adalah titik akhir)
Dihitung f(n) = g(n) + h(n) untuk setiap *node*	f(A) = g(A) + h(A) = 0 f(B) = g(B) + h(B) = 3 + 6 = 9 f(C) = g(C) + h(C) = 2 + 7 = 9 f(D) = g(D) + h(D) = 8 + 2 = 10 f(E) = g(E) + h(E) = 6 + 4 = 10 f(F) = g(F) + h(F) = 9 + 3 = 12 f(G) = g(G) + h(G) = 11 + 0 = 11
Nilai f(n) untuk setiap *node*	f(A) = 0 f(B) = 9 f(C) = 9 f(D) = 10 f(E) = 10 f(F) = 12 f(G) = 11 (titik akhir)

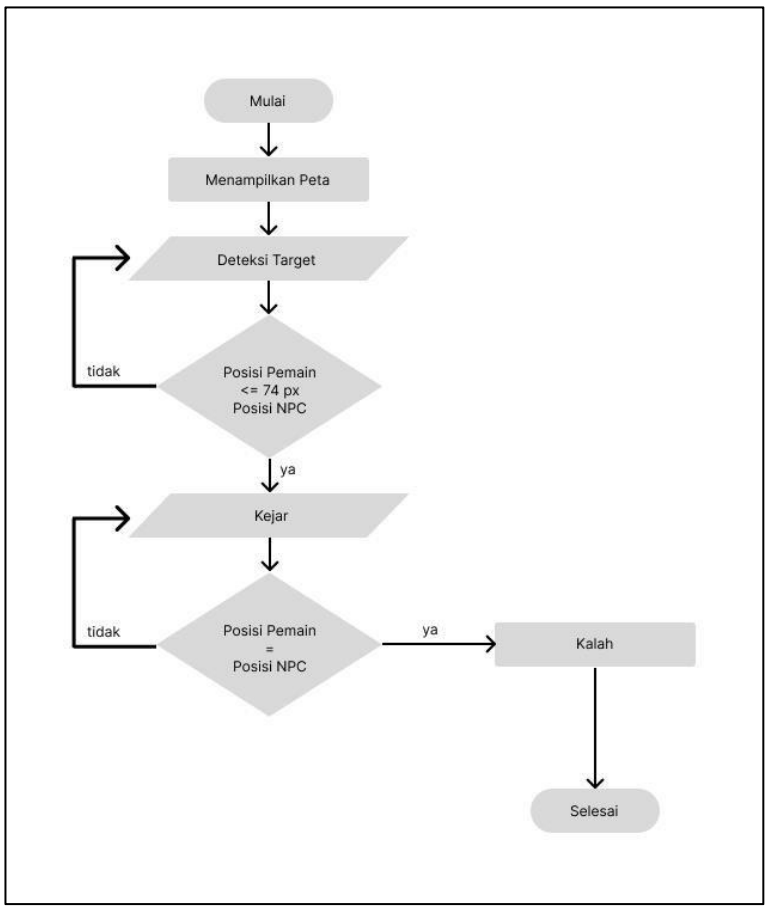
3.2.1.2 *Flowchart Player*



Gambar 3.3 *Flowchart Player*

Dalam *Flowchart Player*, terlihat bahwa pemain akan dihadapkan pada tampilan yang memungkinkan mereka untuk memilih pergerakan karakter. Pemain dapat dengan mudah mengontrol arah gerakannya dengan cara mengusap layar pada konsol yang tersedia sesuai dengan tujuan yang diinginkan. Proses ini memungkinkan pemain untuk bertemu dengan NPC *(Non-Playable Character)* atau memasuki target tujuan. Ini menggambarkan cara interaktif bagi pemain untuk menjelajahi map *game*.

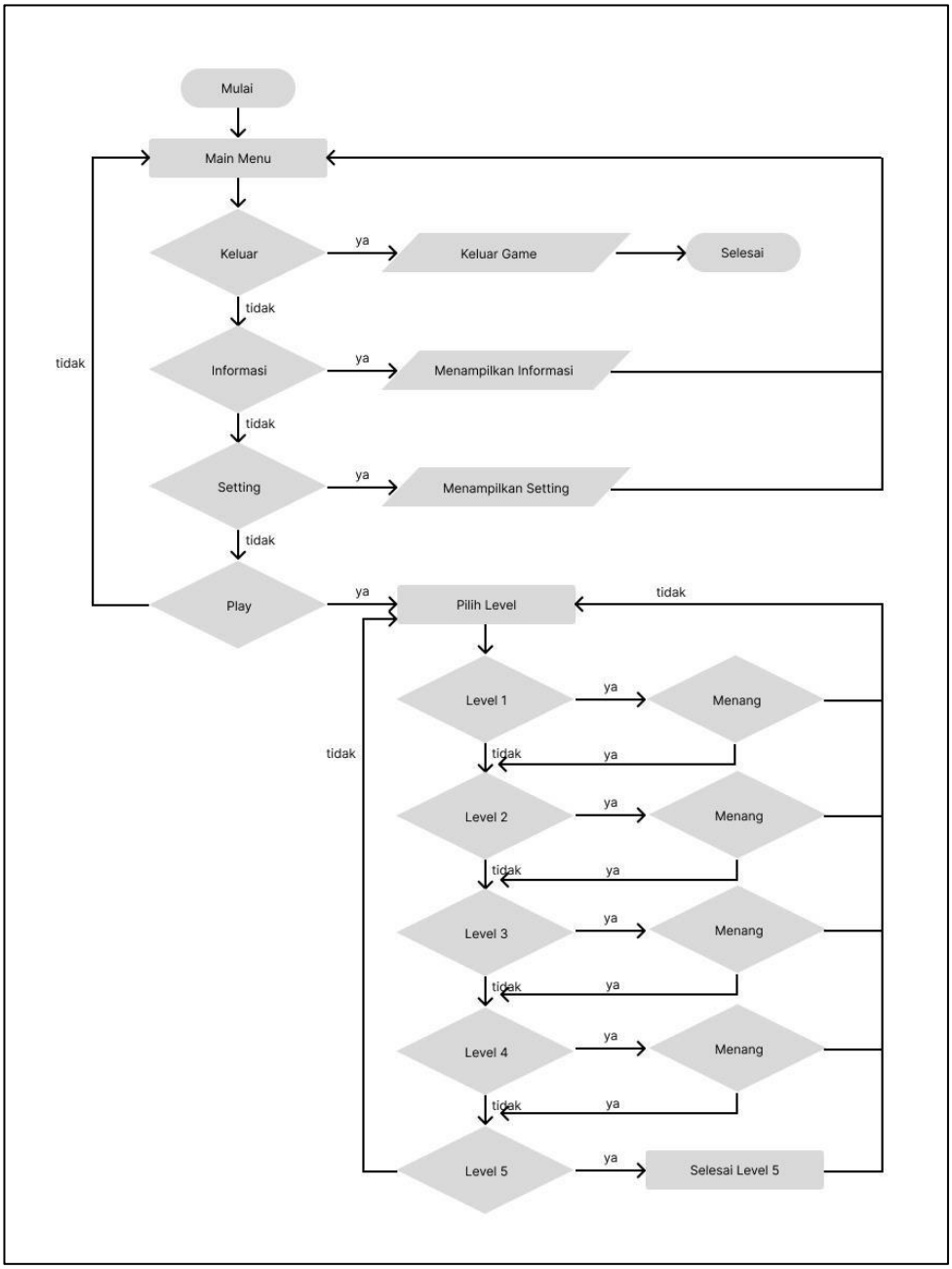
3.2.1.3 *Flowchart* NPC



Gambar 3.4 *Flowchart* NPC

Dalam *Flowchart* NPC *(Non-Playable Character)*, tergambar bahwa saat *game* dimulai, NPC akan mulai melakukan deteksi terhadap posisi pemain dengan menggunakan perhitungan heuristik. Melalui kondisi tersebut, NPC akan merespons dengan bergerak mendekati posisi pemain guna melakukan aksi yang mengakibatkan pengurangan skor koin atau uang dari pemain. Langkah ini mencerminkan perilaku NPC yang diatur untuk merespons keberadaan pemain.

3.2.1.4 *Flowchart Gameplay*

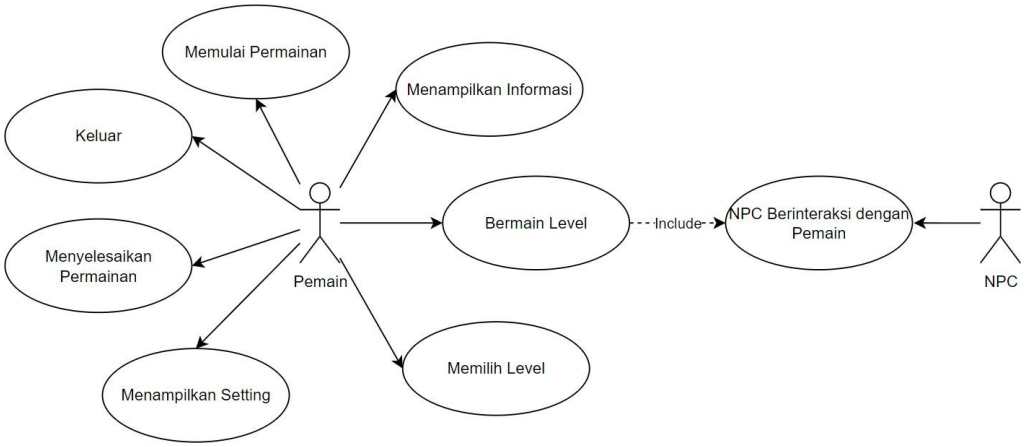


Gambar 3.5 *Flowchart Gameplay*

Dalam *Flowchart Gameplay* terdapat rancangan untuk mengatur alur dari isi *game "The Fabrique"*. Pada rancangan, diatur tentang informasi yang perlu diketahui oleh pemain, tingkat *level*, serta alur cerita dalam *game "The Fabrique"*.

Pada *game* juga dirancang tentang kondisi hasil dari permainan yaitu menang atau kalah.

3.2.1.5 *Use Case* Diagram



Gambar 3.6 *Use Case* Diagram

Dalam *Use Case* Diagram, dijelaskan interaksi yang terjadi antara tiga entitas utama, yaitu *player*, *system*, dan NPC *(Non-Playable Character)*. Diagram tersebut menggambarkan berbagai fungsi yang terlibat dalam *game*. Terdapat beberapa elemen yang muncul, seperti *menu* yang memberikan akses ke opsi *game*, petunjuk yang menyediakan panduan *game*, tombol *play* yang memulai *game*. Melalui diagram ini, dapat dipahami hubungan dan fungsi masing-masing entitas dalam menyediakan pengalaman bermain.

3.2.1.6 Penggunaan *Pathfinding* A*

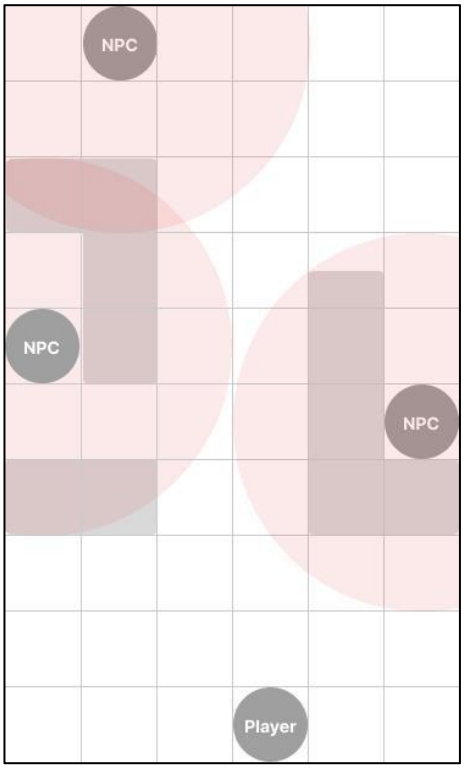
Dalam penelitian ini, diterapkan metode *Pathfinding* dengan menggunakan algoritma A* pada sistemnya. Penerapan metode *Pathfinding* dengan algoritma A* sangat penting untuk mencapai pergerakan NPC (*Non-Playable Character*) yang lebih dinamis dalam mengikuti target. Algoritma A* memungkinkan NPC untuk menemukan jalur terpendek dan paling efisien menuju target, sehingga memberikan perilaku yang lebih realistis dan responsif dalam permainan. Metode *Pathfinding* A* dipilih karena memiliki deteksi sederhana yang dapat dioptimalkan. Berikut

merupakan *script code* yang digunakan untuk menjalankan pergerakan NPC untuk menuju target.

Tabel 3.3 Rancangan *Event sheet*

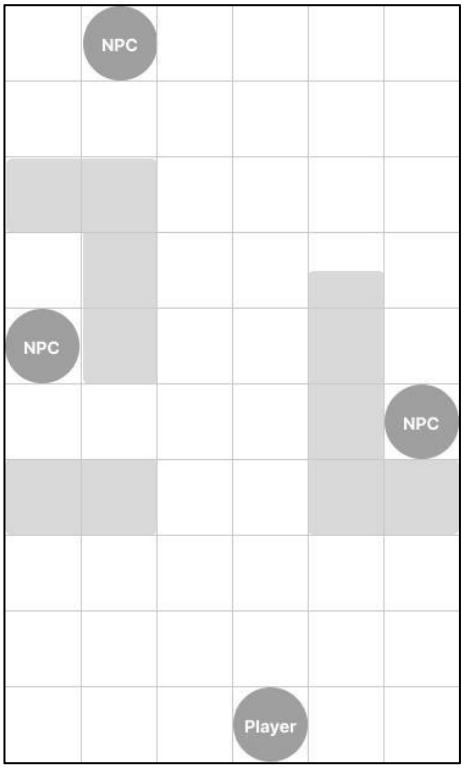
***Script Game* Pada NPC**
Event: Invert, Has LineOfSight to Player image point 0 Event: System, Every random(1,4) seconds **Action:** Enemy, Set Pathfinding enabled Action: Enemy, Pathfinding: Find path to (Enemy.X+random(-150,150), Enemy.Y+random(-150,150)) **Event:** Has LineOfSight to Player image point 0 Action: Enemy, Set Pathfinding disabled **Action:** Enemy, MoveTo: Move to Player image point 0 **Event:** Enemy, On Pathfinding path found **Action:** Enemy, Pathfinding: Move along path

Dalam *game* yang sedang dikembangkan, sistem pendeteksian menggunakan metode *Pathfinding* dengan algoritma *Pathfinding* A* akan diaktifkan saat target berada di luar deteksi NPC *(Non-Playable Character)*. Kondisi ini dirancang untuk menggambarkan situasi di mana musuh hanya akan mendeteksi keberadaan target jika target tersebut berada di depan atau di sekitarnya. Dengan demikian, implementasi ini bertujuan memberikan simulasi terkait bagaimana musuh akan mengenali dan merespons kehadiran target sesuai dengan jarak posisi dari karakter.



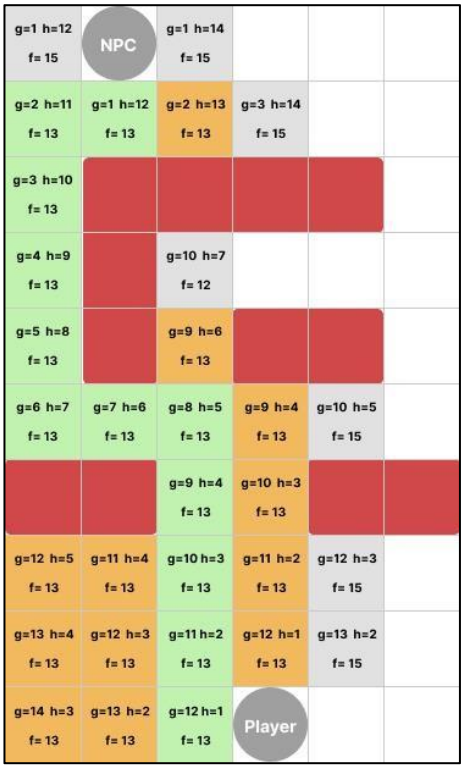
Gambar 3.7 Radius Sensor NPC

Pada implementasi metode *Pathfinding* menggunakan algoritma A*, proses pencarian *node* akan dilakukan dengan mempertimbangkan jarak terendah, sesuai dengan konsep yang diterapkan dalam algoritma A*. Hal ini dirancang untuk memastikan bahwa pergerakan NPC *(Non-Playable Character)* menuju target pemain dapat dijalankan dengan waktu yang singkat. Dengan memperhitungkan jarak terendah, algoritma A* akan memberikan solusi jalur yang optimal, meminimalkan waktu yang dibutuhkan oleh NPC (*Non-Playable Character*) untuk mencapai targetnya. Dengan demikian, penggunaan algoritma A* diharapkan dapat meningkatkan efisiensi pergerakan NPC, menciptakan pengalaman permainan yang lebih responsif dan dinamis bagi pemain.



Gambar 3.8 Simulasi Penempatan NPC

Dengan adanya implementasi ini, diharapkan game akan memberikan pengalaman yang lebih baik bagi pemain. Kecepatan dan ketepatan pergerakan NPC *(Non-Playable Character)* menuju pemain sebagai target akan menjadi faktor dalam meningkatkan pengalaman bermain dari *game*. Selain itu, dengan memanfaatkan *Pathfinding* algoritma A* juga memungkinkan adanya fleksibilitas dalam menyesuaikan rute pergerakan NPC dengan kondisi *game* yang dinamis. Dengan demikian, metode *Pathfinding* menggunakan algoritma A* tidak hanya berfokus pada efisiensi waktu, tetapi juga memberikan tantangan yang menarik dan dinamis dalam *game*.



Gambar 3.9 Proses Pencarian Graf

3.2.1.7 *Game* Aset

Game aset dapat berupa berbagai komponen seperti karakter, latar belakang, objek, efek suara, musik, animasi, dan elemen grafis lainnya yang digunakan dalam proses pengembangan *game*.

1. Karakter

   Pada pengembangan *game "The Fabrique"* nantinya dibagi menjadi dua jenis karakter. Terdapat karakter utama yang akan diceritakan dengan nama Brow Lee, dan dua karakter NPC yang diceritakan dengan nama Preman dan Senior.

Tabel 3.4 *Game* Aset Pada Karakter

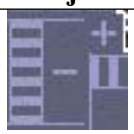
No	Karakter	Identitas	Keterangan
1		**Nama :** Brow Lee **Profil :** Karyawan baru **Peran :** Karakter utama	Brow Lee, karakter utama dalam *game*. Memiliki profil seorang pemuda yang baru menyelesaikan

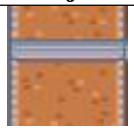
No	Karakter	Identitas	Keterangan
			pendidikannya dan akan memulai kariernya di tempat baru yaitu ibukota.
2		**Nama :** Preman **Profil :** Preman Ibukota **Peran :** NPC	Preman adalah karakter NPC yang akan menjadi lawan dari karakter utama karena membahayakan orang lain.
3		**Nama :** Senior **Profil :** Karyawan lama **Peran :** NPC	Senior adalah karakter NPC yang akan menjadi lawan dari karakter utama karena persaingan di kantor tempat Brow Lee bekerja.

2. *Tilemap*

Tilemap nantinya akan disusun sehingga menjadi sebuah jalan, bangunan, tumbuhan, dan komponen lainnya. Pada penerapannya nanti akan ada dua *Tilemap* yang berbeda. Antaranya seperti jalan aspal dan jalan rumput yang dapat dilewati. Kemudian ada bangunan, portal, dan pembatas jalan yang memiliki *behavior solid* yang nantinya seperti tembok pembatas ataupun rintangan objek.

Tabel 3.5 *Game* Aset Pada *Tilemap*

No	Objek	Identitas	Keterangan
1		Jalan Aspal	**Fungsi :** Sebagai lintasan yang bisa dilewati oleh semua karakter. ***Behavior* :** Tidak ada
2		Bangunan	**Fungsi :** Sebagai halangan dalam *game*. ***Behavior* :** *Solid*
3		Portal	**Fungsi :** Sebagai garis batas pergerakan karakter. ***Behavior* :** *Solid*

No	Objek	Identitas	Keterangan
4		Tempat target	**Fungsi :** Sebagai target tujuan dari pemain agar dapat memenangkan *level*. ***Behavior* :** Tidak ada
5		Halangan jalan	**Fungsi :** Sebagai penghambat yang dapat membuat karakter berjalan lebih lambat. ***Behavior* :** Tidak ada

3. *Audio*

Audio pada *game "The Fabrique"* yang berguna untuk memberikan penekanan emosi tertentu pada sebuah kejadian yang terjadi di dalam *game*. *Audio* nantinya akan ada beberapa perbedaan dalam penggunaan untuk beberapa *scene* seperti awal masuk, saat *gameplay*, saat menang, saat kalah, dan saat bersentuhan dengan NPC.

Tabel 3.6 *Game* Aset Pada *Audio*

No	Judul	Keterangan
1	*Peach* – Sakura *Girl*	*Audio* untuk latar utama
2	*Action* – Pixabay	*Audio* untuk *gameplay*
3	*Collision* – Pixabay	*Audio* untuk saat karakter utama bersentuhan dengan NPC
4	*Winner* – Pixabay	*Audio* untuk saat masuk ke hasil *game*
5	*Click* – Pixabay	*Audio* untuk saat menekan tombol pada *game*

4. *Level*

Perancangan *level* yang digunakan dalam *game "The Fabrique"* dengan ada beberapa tingkatan kesulitan yang akan menjadi tantangan bagi pemain. Dalam *level* nantinya juga terdapat halangan rintang yang dapat menambah keseruan dalam bermain.

Tabel 3.7 *Game* Aset Pada *Level*

Level	**Keterangan**	***Pathfinding***
Level 1	Brow Lee adalah seorang pemuda yang baru lulus dan akan bekerja di ibukota. Dalam perjalanan Brow Lee perlu untuk menuju ke tempat kos. Namun Brow Lee perlu berhati-hati pada Preman yang berkeliaran untuk menyerang.	Pada *level* 1 tempat tujuannya adalah tempat kos yang disimbolkan dengan bangunan kecil berwarna jingga. Pada *level* 1 ini Brow Lee perlu menghindari NPC bernama Preman dengan jarak tempuh yang pendek di 480 x 854 px.
Level 2	Setelah sampai di tempat kos, Brow Lee akan memulai hari pertamanya bekerja. Brow Lee harus segera berangkat bekerja. Namun Preman sudah mengincar Brow Lee, sehingga Brow Lee perlu lebih berhati-hati agar tidak bertemu dengan Preman.	Pada *level* 2 tempat tujuannya adalah kantor tempat Brow Lee bekerja yang disimbolkan dengan bangunan berwarna merah. Pada *level* 2 ini Brow Lee perlu menghindari NPC Bernama Preman dengan Jarak tempuh yang Panjang di 700 x 854 px.
Level 3	Ketika sudah sampai di tempat bekerja, Brow Lee perlu melakukan perkenalan dengan bos di tempatnya bekerja. Brow Lee perlu berhati-hati karena ada Senior Brow Lee yang memiliki niat jahat terhadap Brow Lee. Untuk itu Brow Lee perlu untuk menghindari Senior yang bisa membahayakan.	Pada *level* 3 tempat tujuannya adalah ruangan bos di tempat kerja Brow Lee bekerja yang disimbolkan dengan ruangan yang memiliki perabot kantor seperti meja kursi. Pada *level* 3 ini Brow Lee perlu menghindari NPC bernama Senior dengan halangan sederhana.

Level	Keterangan	*Pathfinding*
Level 4	Bekerja hari pertama telah usai, saatnya Brow Lee pulang ke kos untuk beristirahat. Namun dalam perjalanan pulang Brow Lee perlu berhati-hati karena masih ada Preman yang berkeliaran di luar sana.	Pada *level* 4 tempat tujuannya adalah tempat kos yang disimbolkan dengan bangunan kecil berwarna jingga. Pada *level* 4 ini Brow Lee perlu menghindari NPC bernama Preman dengan jarak tempuh yang pendek di 480 x 854 px. Namun pada *level* 4 ini terdapat halangan di jalan yang membuat semua karakter berjalan lebih pelan saat berada di atas halangan.
Level 5	Karena banyak sekali kesulitan yang dihadapi oleh Brow Lee. Brow Lee akan melaporkan kejadian yang dihadapinya ke polisi agar dapat mengatasi orang-orang yang membahayakan Brow Lee. Namun dalam perjalanan ke kantor polisi, Brow Lee dihadang oleh Preman dan Senior sekaligus.	Pada *level* 5 tempat tujuannya adalah kantor polisi yang disimbolkan dengan bangunan besar berwarna merah. Pada *level* 5 ini Brow Lee perlu menghindari NPC Bernama Preman dan Senior sehingga dalam *level* ini memiliki 2 karakter NPC yang berbeda.

3.2.2 Perancangan *User Interface*

Dalam tahap perancangan *user interface* atau tampilan pada *game*, desain harus mempertimbangkan tata letak elemen-elemen *visual*, penggunaan warna, ikon, dan navigasi yang mudah dipahami. Tujuannya adalah memberikan

pengalaman pengguna yang lancar dan memudahkan akses ke fitur-fitur aplikasi dengan estetika yang menarik, sehingga meningkatkan daya tarik aplikasi.

1. *Wireframe Main menu*

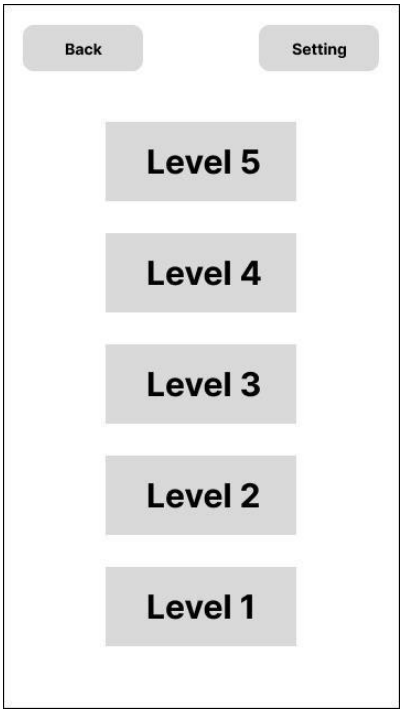
   Wireframe main menu adalah rancangan antarmuka pengguna yang pertama kali ditampilkan saat membuka aplikasi *game "The Fabrique"*.

Gambar 3.10 *Wireframe Main menu*

   Pada *user interface* main menu ini, akan terdapat tombol *play* untuk memulai *gameplay*, tombol informasi untuk melihat petunjuk *game*, dan tombol *setting* untuk mengakses pengaturan. Sehingga dengan adanya tampilan *main menu* pada *game* akan memudahkan pemain dalam memahami alur permainan, dan memberikan gambaran awal tentang permainan yang akan dimainkan. Hal itu didukung dengan adanya tombol *menu* untuk menuju informasi yang akan ada penjelasan tentang mekanisme berjalannya *game*. Kemudian juga ada tombol *menu* untuk menuju ke pengaturan yang akan ada tampilan untuk mengatur *audio* untuk aktif ataupun tidak aktif.

2. *Wireframe Menu Level*

Wireframe menu level merupakan rancangan *user interface* yang ditampilkan ketika pemain menekan tombol *play* pada *user interface main menu* untuk masuk ke *game*, sehingga sebelum masuk ke *gameplay*, pemain perlu memilih *level* untuk tingkat tantangan yang diinginkan antara *level* 1 sampai *level* 5.

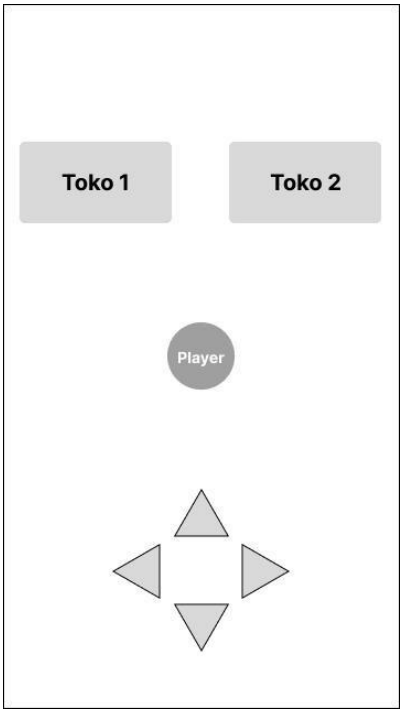


Gambar 3.11 *Wireframe Menu Level*

Nantinya pemain bisa memilih *level* untuk menentukan tingkat kesulitan yang diinginkan. Untuk mempermudah pemain juga terdapat tombol *menu* pengaturan untuk masuk ke pengaturan. Kemudian juga ada tombol *menu* untuk kembali ke *main menu*.

3. *Wireframe Gameplay*

   Wireframe gameplay ditampilkan setelah pemain memilih *level* yang ingin dimainkan pada *user interface menu level*. Pemain dapat menggerakkan karakter menggunakan kontrol console untuk menggerakkan ke arah atas, bawah, kanan, dan kiri. Pada *gameplay* juga terdapat bangunan yang menjadi halang rintang dan ada salah satu yang menjadi bangunan tempat tujuan karakter untuk memenangkan *level game*.



Gambar 3.12 *Wireframe Gameplay*

Sehingga pada *wireframe gameplay* ini nantinya akan dirancang untuk pemain agar dapat menggerakkan karakter untuk mencapai target tujuan pada *game*. Pada setiap *level*nya akan memiliki tantangan yang bervariasi tergantung dari *level* yang dipilih.

4. *Wireframe* Hasil *Game*

   Wireframe hasil *game* terdiri dari dua jenis tampilan yang bergantung pada hasil permainan. Jika pemain berhasil mencapai bangunan tujuan, *user interface* yang ditampilkan akan menunjukkan keterangan menang. Namun jika karakter yang dimainkan pemain dikejar oleh NPC kemudian ada di kondisi bersentuhan, maka akan ditampilkan *use*r *interface* dengan keterangan kalah.

Gambar 3.13 *Wireframe* Hasil *Game*

3.3 Rancangan Pengujian

Dalam penelitian ini, peneliti akan menggunakan beberapa metode pengujian untuk mengevaluasi kinerja dan keefektifan algoritma *Pathfinding* A* yang diimplementasikan dalam *game "The Fabrique"*. Rancangan pengujian mencakup Pengujian Fungsional yang digunakan untuk memastikan bahwa setiap komponen dari *game* berfungsi dengan benar sesuai dengan skenario yang telah ditentukan. Kemudian Pengujian Perilaku NPC untuk mengevaluasi apakah NPC dalam *game* dapat menggunakan algoritma *Pathfinding* A* dengan benar untuk navigasi dan interaksinya. Selanjutnya Pengujian *Pathfinding* A* untuk mengevaluasi kinerja dan efektivitas algoritma *Pathfinding* A* dalam berbagai skenario, termasuk interaksi antara NPC dan karakter pemain.

3.3.1 Rancangan Pengujian Fungsional

Pengujian fungsional bertujuan untuk memastikan bahwa setiap komponen dari *game "The Fabrique"* berfungsi sesuai dengan spesifikasinya. Langkah-langkah dalam pengujian fungsional meliputi:

1. Identifikasi komponen dengan menentukan komponen atau fitur yang akan diuji, seperti sistem navigasi NPC, interaksi antar karakter, dan mekanisme permainan lainnya.
2. Penentuan kondisi pengujian dengan menyusun berbagai skenario pengujian untuk setiap komponen, mencakup skenario sukses dan skenario kegagalan.
3. Pelaksanaan Pengujian: Melaksanakan pengujian sesuai dengan kasus uji yang telah ditentukan. Setiap pengujian akan mencatat apakah hasil yang diharapkan berhasil atau tidak.

Tabel 3.8 Rancangan Pengujian Fungsional

No	Komponen yang Diuji	Kondisi	Hasil
1	Berupa *list* komponen atau *interface* yang akan diujikan	Kondisi pada saat komponen yang diuji dijalankan	Kondisi komponen yang diuji sesuai atau tidak sesuai

3.3.2 Rancangan Pengujian Perilaku NPC

Pengujian perilaku NPC fokus pada bagaimana NPC menggunakan algoritma *Pathfinding* A* untuk navigasi di dalam *game*. Langkah-langkah dalam pengujian perilaku NPC meliputi:

1. Percobaan digunakan untuk menyusun serangkaian percobaan di mana NPC harus mencapai tujuan tertentu dalam berbagai skenario.
2. Keadaan digunakan untuk mengatur keadaan awal dari lingkungan *game* dan posisi NPC.
3. Kondisi digunakan untuk menentukan kondisi atau parameter spesifik yang harus dipenuhi selama percobaan.

4. Aksi digunakan untuk mengamati aksi yang diambil oleh NPC dalam usaha mencapai tujuannya.
5. Hasil digunakan untuk mencatat hasil dari percobaan, termasuk apakah NPC berhasil mencapai tujuan, waktu yang diperlukan, dan jalur yang diambil.

Table 3.9 Rancangan Pengujian Perilaku NPC

Percobaan	Keadaan	Kondisi	Aksi	Hasil
Informasi mengenai pengulangan percobaan pada setiap *level*	Keadaan awal dari NPC terhadap karakter Player	Posisi NPC terhadap karakter Player	Aksi yang dilakukan oleh NPC ketika ada pada kondisi tertentu	Kondisi komponen yang diuji sesuai atau tidak sesuai

3.3.3 Rancangan Pengujian *Pathfinding* A*

Pengujian ini bertujuan untuk menilai kinerja algoritma *Pathfinding* A* dalam berbagai kondisi. Langkah-langkah dalam pengujian *Pathfinding* A* meliputi:

1. *Level* digunakan untuk menentukan *level* atau peta yang akan digunakan untuk pengujian, termasuk kompleksitas dan ukuran peta.
2. Jenis rintangan digunakan untuk mengidentifikasi jenis dan posisi rintangan yang ada dalam *level* tersebut.
3. Percobaan digunakan untuk melakukan sejumlah percobaan untuk setiap *level*, menggunakan berbagai konfigurasi rintangan.
4. Start (x,y) digunakan untuk menentukan titik awal (koordinat x, y) dari NPC.
5. Tujuan (x,y) digunakan untuk menentukan titik tujuan (koordinat x, y) yang harus dicapai oleh NPC.
6. *Open list* digunakan untuk mencatat jumlah *node* yang dimasukkan ke dalam daftar terbuka (*open list*) selama proses pencarian jalur.
7. Jumlah *path* digunakan untuk menghitung jumlah jalur yang ditemukan selama pengujian.

8. Waktu digunakan untuk mencatat waktu komputasi yang diperlukan oleh algoritma untuk menemukan jalur dari titik awal ke titik tujuan.

Table 3.10 Rancangan Pengujian *Pathfinding* A*

Level	**Jenis Rintangan**	**Percobaan**	**Start (x,y)**	**Tujuan (x,y)**	***Open list***	**Jumlah *Path***	**Waktu**
1	Berupa rintangan dengan skenario sesuai dengan *level*	Informasi mengenai pengulangan percobaan pada setiap *level*	Berupa titik awal dari karakter NPC	Berupa titik awal dari karakter Player	Titik lintas yang tersedia untuk dilewati oleh NPC	Jumlah *path* yang dilalui oleh NPC untuk menuju Player	Waktu yang dibutuhkan NPC untuk menuju titik tujuan

Dengan rancangan pengujian yang telah disusun, diharapkan dapat diperoleh evaluasi yang mengenai kinerja dan efektivitas algoritma *Pathfinding* A* dalam *game "The Fabrique"*.