PENERAPAN METODE FINITE STATE MACHINE PADA NPC GAME TOPDOWN SHOOTER MENGGUNAKAN GODOT ENGINE

TUGAS AKHIR

Diajukan untuk Memenuhi Sebagian Syarat Guna Memperoleh
Gelar Sarjana Komputer
dalam Ilmu Teknologi Informasi

Oleh : **ANDRI ADITYA**
NIM : 2008096046

FAKULTAS SAINS DAN TEKNOLOGI
UNIVERSITAS ISLAM NEGERI WALISONGO
SEMARANG
2024

PERNYATAAN KEASLIAN

Yang bertanda tangan di bawah ini :

Nama	:	Andri Aditya
NIM	:	2008096046
Jurusan/Program Studi	:	Fakultas Sains Teknologi/ Teknologi Informasi

menyatakan bahwa skripsi yang berjudul :

PENERAPAN METODE FINITE STATE MACHINE PADA NPC GAME TOPDOWN SHOOTER MENGGUNAKAN GODOT ENGINE

secara keseluruhan adalah hasil penelitian/karya saya sendiri, kecuali bagian tertentu yang dirujuk sumbernya.

Semarang, 16 Mei 2024
Pembuat pernyataan,

Andri Aditya
NIM : 2008096046

KEMENTERIAN AGAMA R.I.
UNIVERSITAS ISLAM NEGERI WALISONGO
FAKULTAS SAINS DAN TEKNOLOGI
Jl. Prof. Dr. Hamka (Kampus II) Ngaliyan Semarang
Telp. 024-7601295 Fax. 7615387

PENGESAHAN

Naskah skripsi berikut ini :

Judul : **Penerapan metode Finite State Machine pada NPC Game topdown shooter menggunakan Godot Engine**
Penulis : Andri Aditya
NIM : 2008096046
Jurusan : Fakultas Sains Teknologi

Telah diujikan dalam sidang *tugas akhir* oleh Dewan Penguji Fakultas Sains dan Teknologi UIN Walisongo dan dapat diterima sebagai salah satu syarat memperoleh gelar sarjana dalam Ilmu Teknologi Informasi.

Semarang, 26 Juni 2024

DEWAN PENGUJI

Penguji I,

Dr. Khotibul Umam, M.Kom
NIP : 19790827 201101 1007

Penguji II,

Dr. Masy Ari Ulinuha, M.T
NIP : 19810812 201101 1007

Penguji III,

Wenty Dwi Yuniarti, S. Pd. M. Kom
NIP : 19770622 200604 2 005

Penguji IV,

Adzhal Arwani Mahfudh, M. Kom
NIP : 19910703 201903 1 006

Pembimbing I,

Dr. Masy Ari Ulinuha, M.T
NIP : 19810812 201101 1007

Pembimbing II,

Hery Mustofa, M.Kom
NIP : 19870317 201903 1 007

iii

NOTA DINAS

Semarang, 23 April 2024

Yth. Ketua Program Studi Teknologi Informasi
Fakultas Sains dan Teknologi
UIN Walisongo Semarang

Assalamu'alaikum warahmatullahi wabarakatuh

Dengan ini diberitahukan bahwa saya telah melakukan bimbingan, arahan dan koreksi naskah skripsi dengan:

Judul	:	Penerapan metode Finite State Machine pada NPC Game topdown shooter menggunakan Godot Engine
Nama	:	Andri Aditya
NIM	:	2008096046
Jurusan	:	Fakultas Sains Teknologi

Saya memandang bahwa naskah skripsi tersebut sudah dapat diajukan kepada Fakultas Sains dan Teknologi UIN Walisongo untuk diujikan dalam Sidang Munaqasyah.

Wassalamu'alaikum warahmatullahi wabarakatuh

Pembimbing I,

Dr. Masy Ari Ulinuha, ST., M.
NIP : 19810812 201101 1007

NOTA DINAS

Semarang, 30 April 2024

Yth. Ketua Program Studi Teknologi Informasi
Fakultas Sains dan Teknologi
UIN Walisongo Semarang

Assalamu'alaikum warahmatullahi wabarakatuh

Dengan ini diberitahukan bahwa saya telah melakukan bimbingan, arahan dan koreksi naskah skripsi dengan:

Judul	:	Penerapan metode Finite State Machine pada NPC Game topdown shooter menggunakan Godot Engine
Nama	:	Andri Aditya
NIM	:	2008096046
Jurusan	:	Fakultas Sains Teknologi

Saya memandang bahwa naskah skripsi tersebut sudah dapat diajukan kepada Fakultas Sains dan Teknologi UIN Walisongo untuk diujikan dalam Sidang Munaqasyah.

Wassalamu'alaikum warahmatullahi wabarakatuh

Pembimbing II,

Hery Mustofa, M.Kom
NIP : 19870317 201903 1 007

ABSTRAK

Permainan atau sering disebut game telah menjadi aktivitas hiburan dan kesenangan oleh kebanyakan orang. Berdasakan data dari situs data.ai Pada tahun 2022, dari total 90 miliar unduhan, Indonesia menjadi peringkat ketiga dibawah india dan brazil. Walaupun begitu, pengembang game lokal masih kesulitan bersaing dengan pengembang *game* asing. Menurut Kekominfo bersama NIKO Partners dan Asosiasi Gim Indonesia (AGI) pada tahun 2021, produk *game* lokal hanya menguasai 0,4% pasar dalam negeri. Dalam rangka meningkatkan produk game lokal. penulis bertujuan untuk membangun game tema top down Shooter dengan nama ”Survivor Shooter”. Game ini berfokus pada bagaimana menerapkan kecerdasan pada NPC dan objek terkait yang terdapat pada permainan. FSM digunakan untuk membangun kecerdasan pada NPC dikarenakan metode ini tidak terlalu kompleks untuk diterapkan. Metode pengembangan yang digunakan yaitu metode *Game Development LifeCycle*. GDLC merupakan *framework* dalam membangun perangkat lunak video game yang menerapkan pendekatan iteratif terdiri dari 6 fase pengembangan, dimulai dari fase inisialisasi/pembuatan konsep, preproduction, production, testing, beta dan realease. Pengujian dibagi menjadi 2 tahap, *alpha test* dan *beta test*. Hasil dari pengujian *alpha* yaitu pada pengujian unit diperoleh tingkat presentase 100%, pengujian integrasi dengan tingkat presentase 100%, serta pengujian sistem dengan total 7532 perangkat yang didukung. Sedangkan pada pengujian beta dilakukan dengan melakukan kuisioner diperoleh hasil ”baik” dengan tingkat presentase 61.25%

Kata kunci : Video game, Godot Engine, Finite State Machine

KATA PENGANTAR

Puji syukur kepada Allah SWT berkat Rahmat, Hidayah, dan Karunia-Nya kepada kita semua sehingga kami dapat menyelesaikan proposal skripsi dengan judul "Penerapan metode Finite State Machine pada NPC Game topdown shooter menggunakan Godot Engine". Laporan proposal skripsi ini disusun sebagai salah satu syarat untuk mengerjakan skripsi pada program Strata-1 di Jurusan Teknologi Informasi, Fakultas Sains Teknologi, Universitas Islam Negeri Walisongo.

Proses penyusunan skripsi ini tidak lepas dari doa, bantuan, bimbingan, motivasi dan peran dari banyak pihak. Sehingga penulis mengucapkan terimakasih kepada :

1. Bapak Prof. Dr. Nizar, M.Ag selaku Rektor UIN Walisongo
2. Bapak Dr. Khotibul Umam, M.Kom selaku Kajur teknologi informasi
3. Bapak Dr. Masy Ari Ulinuha, M.T sebagai Pembimbing I
4. Bapak Hery Mustofa, M.Kom sebagai Pembimbing II
5. Semua pihak yang tidak dapat penulis sebutkan satu persatu yang telah memberikan kontribusi hingga selesainya skripsi ini.

Atas segala kekurangan dan kelemahan dalam skripsi ini penulis mengharapkan saran dan kritik yang membangun. Semoga karya tulis yang sederhana ini dapat menjadi bacaan yang bermanfaat dan dapat dikembangkan bagi peneliti-peneliti selanjutnya.

DAFTAR ISI

DAFTAR TABEL

DAFTAR GAMBAR

DAFTAR LAMPIRAN

Halaman

BAB I

PENDAHULUAN

A. Latar Belakang Masalah

Permainan atau sering disebut *game* telah menjadi aktivitas hiburan dan kesenangan oleh kebanyakan orang. Semakin berkembangnya era digitalisasi juga turut membawa perubahan pada berbagai bidang. Salah satu wujud perkembangan ini yaitu makin berkembangnya industri *video game* yang menjadi salah satu sektor paling diminati dalam pasar hiburan.

Pada awalnya, pasar *video game* didominasi oleh *game* konsol (PS4, Xbox, dan Nintendo Switch), yang menjadi wajah dari *video game*. Namun dengan berkembangnya media digital dan akses internet semakin mudah. pasar game berevolusi menjadi format digital dan berkembang melampaui target konsolnya di mana sekarang dapat dinikmati dalam berbagai perangkat, baik komputer, *handphone, virtual reality*, dan sebagainya. (Mulachela, 2020)

Keberadaan pasar *game* di Indonesia menjadi pangsa baru yang berpotensi sejalan dengan canggihnya smartphone yang beredar. Berdasarkan laporan State of Mobile Gaming 2023 (2023) yang diterbitkan oleh data.ai, Indonesia menduduki peringkat 3 dengan total jumlah unduhan hingga 3.3 Miliar pada *play store*. Walaupun membuktikan jika Indonesia menjadi salah satu target pasar berbagai pengembang *game*, kenyataanmya pengembang game lokal masih kesulitan bersaing dengan pengembang *game* asing. Hal tersebut dibuktikan oleh catatan yang dimiliki Kementerian Pariwisata dan Ekonomi

Kreatif (Kemenparekraf) berdasarkan total pendapatan *game* nasional yang mencapai 25 Triliun, pendapatan pengembang industri permainan dalam negeri tercatat sangat sedikit, yakni hanya 0,5 persen (CNN Indonesia, 2023).

Terdapat beragam jenis *genre game* yang diminati, salah satunya adalah *genre* top-down shooter. *Genre* top-down shooter merupakan cabang dari *genre action*. Top-down adalah permainan di mana pemain melihat aksi permainan dari sudut pandang atas. Mirip ketika pemain melihat peta dalam permainan atau lebih dikenal dengan istilah *Bird's Eye View* (Andersen et al., 2021) . Guna menambah tantangan permainan, para pengembang *game* sering memasukan unsur musuh atau Non Player Character(NPC). NPC adalah istilah yang digunakan untuk mendefinisikan karakter yang digerakan oleh komputer dalam *video game* (Baffa et al., 2017). Umumnya NPC digunakan untuk membantu player, memberikan tujuan pada skenario permainan, dan juga aktivitas sejenisnya yang berhubungan dengan player. Diperlukan metode algoritma Dalam mengatur tingkah laku NPC, seperti bagaimana tujuan bergerak, mendeteksi musuh, atau mencapai titik target. Untuk menyelesaikan masalah tersebut, *Finite state machine* merupakan salah satu metode yang digunakan untuk mengatur sistem kecerdasan dalam pengambilan keputusan NPC.

Finite State Machine merupakan sebuah metodologi perancangan sistem kontrol yang menggambarkan tingkah laku atau prinsip kerja sistem dengan menggunakan tiga hal, yaitu: *state* (keadaan), *event* (kejadian) dan *action* (aksi) (Al Tahtawi et al., 2017). Secara sederhana, FSM berkaitan dengan perpindahan keadaan apabila syarat telah terpenuhi

lewat aksi yang telah diberikan sebelumnya. Berdasarkan hal tersebut, FSM sendiri tentu dapat dimanfaatkan dalam bidang pengembangan game khususnya pada game action maupun RPG (*Role Play Game*) karena progress permainan menentukan langkah yang akan terjadi berikutnya (Zagal and Altizer, 2014).

Tujuan dari penelitian ini adalah merancang sebuah *game* tema *top down Shooter* dengan nama *"Survivor Shooter"*. Game ini berfokus pada bagaimana menerapkan kecerdasan pada NPC dan objek terkait yang terdapat pada permainan. Penulis memilih menggunakan FSM untuk membangun kecerdasan pada NPC dikarenakan metode ini tidak terlalu kompleks untuk diterapkan. Untuk membuat *game*, umumnya dibutuhkan software khusus pengembangan *game* guna mempermudah pembuatan permainan. Godot engine adalah *game engine open source* yang bisa digunakan untuk membuat *game* untuk berbagai *platform* seperti desktop, android, serta konsol. Godot Engine menggunakan Bahasa GDScript yang merupakan turunan dari Bahasa python. Alasan memilih Godot Engine karena software ini relatif ringan, mudah digunakan serta memiliki fitur yang cukup lengkap dibandingkan dengan *game engine* lainnya.

B. Rumusan Masalah

Berdasarkan latar belakang yang telah dijabarkan diatas, diperoleh rumusan masalah dalam tugas akhir ini yaitu antara lain:

1. Bagaimana membangun permainan bertema *topdown shooter* dengan Godot Engine?

2. Bagaimana menerapkan *Finite State Machine* pada karakter musuh?

C. Batasan Masalah

Berdasarkan rumusan masalah yang dijelaskan diatas, maka ruang lingkup masalah penelitian ini adalah:

1. *Game* dibuat dengan menggunakan Godot Engine
2. Genre yang dibuat adalah *Action Shooter*
3. Sistem operasi yang menjadi target dari permainan ini adalah android(apk), dengan versi minimal 8.0
4. Metodologi pengembangan menggunakan *Game Development LifeCyle*

D. Tujuan Penelitian

Tujuan dari penelitian ini yaitu :

1. Membangun *game* bertema top down shooter menggunakan godot engine dengan menerapkan metodologi finite state machine pada NPC
2. Menguji fitur dan kelayakan *game* yang telah dibangun melalui pengujian internal dan survei kuisioner

E. Manfaat Penelitian

Manfaat penelitian pada proposal tugas akhir ini dibagi menjadi dua, yaitu:

1. Manfaat Teoritis

Manfaat teoritis pada penelitian ini diharapkan dapat bermanfaat dalam teori pengembangan game menggunakan *Godot engine*, serta mengetahui implementasi mekanisme kecerdasan pada karakter non-player menggunakan FSM.

2. Manfaat Praktis

a. Bagi peneliti

Manfaat penelitian ini bagi penulis yaitu untuk memahami proses pengembangan game menggunakan pengembangan GDLC.

b. Bagi pengembang game

1. Meningkatkan daya saing pengembang *game* lokal terhadap pengembang *game* luar
2. Memberi refrensi bagi perngembang *game* mengenai metodologi FSM pada permainan

F. Sistematika Penulisan

Dalam penyusunan proposal ini, agar mempermudah memahami pembahasan maka sistematika penulisan sebagai berikut:

BAB I : **Pendahuluan**

Pada bab ini menjelaskan mengenai latar belakang, rumusan masalah, batasan masalah, tujuan penelitian, manfaat penelitian dan sistematika penulisan.

BAB II : **Kajian Pustaka**

Berisi kajian dan tinjauan Pustaka, memaparkan kajian dari studi-studi terdahulu mengenai permasalahan yang berhubungan dengan penelitian ini. Serta memaparkan keranga berpikir pada penelitian ini.

BAB III : **Metodologi Penelitian**

Bab ini membahas rencana pengembangan *game* menggunakan metodologi *game development lifecycle*.

BAB IV : **Hasil dan Pembahasan**

Bab ini membahas mengenai analisis game, analisis kebutuhan perangkat, perancangan game, perancangan struktur menu, perancangan Finite State Machine, implementasi program, pengujian, serta proses rilis permainan.

BAB V : **Kesimpulan**

Berisi kesimpulan dari hasil penelitian yang dilakukan dan saran yang dapat digunakan untuk bahan pengembangan penelitian berikutnya.

BAB II

KAJIAN PUSTAKA

A. Tinjauan pustaka

1. *Video Game*

Permainan atau game merupakan aktivitas manusia dimana kegiatan tersebut terdapat aturan tertentu yang bertujuan sebagai media rekreasi dan bersenang-senang. permainan adalah sebuah sistem dimana pemain terlibat dalam konflik buatan, yang mana pemain berinteraksi dengan sistem dan konflik yang ada pada permainan merupakan buatan atau rekayasa pembuat permainan (Andersen et al., 2021). Pada permainan terdapat peraturan yang bertujuan untuk membatasi perilaku pemain dan menentukan alur permainan.

Video game merupakan permainan berupa multimedia (teks, ataupun gambar) yang dimainkan menggunakan media elektornik. Pada *video game* umumnya melibatkan interaksi antara perangtkat lunak permainan, orang yang memainkan dan juga perangkat keras (berupa televisi, komputer, handphone, dan lainnya) permainan tersebut dijalankan. Agar dapat mnimbulkan rasa puas dan menyenangkan, biasanya game dikembangkan agar semenarik mungkin dengan cara memberikan tantangan yang cukup sulit, menambahkan kompleksitas permainan, ataupun memberikan *reward* yang beragam.

a. Jenis-jenis *Video Game*

Game dikasifikasikan menjadi 3 jenis, yaitu berdasakan platform (perangkat), ruang dimensi, dan juga berdasarkan genre (Hoesen, 2022; Martono, 2015)

Platform Permainan

1. **Acarde game**, adalah perangkat mesin dimana membutuhkan koin atau kredit untuk dapat memainkan game tersebut.
2. **Pc game**, merupakan perangkat lunak yang dapat diinstall pada perangkat komputer pribadi (PC). Dikarenakan perangkat komputer dapat dikostumasikan oleh pengguna, PC game menjadi platform populer dikalangan gamer.
3. **Mobile game**, yaitu permainan yang dapat dimainkan melalui perangkat mobile ataupun smartphone seperti android dan Ios.
4. **Console game**, adalah game yang dijalankan pada perangkat khusus yang didesain untuk menjalankan game. Contih Perangkat konsol yang populer untuk saat ini yaitu *Sony PlayStation, Microsoft Xbox, dan Nintendo Switch.*
5. **Web browser game**, adalah permainan yang dapat dimainkan dengan browser seperti *Firefox, Chrome, Opera*.

Dimensi Permainan

1. **Game 2D**, merupakan game dilihat dari keadaan ruangnya hanya memiliki 2 elemen, yaitu X dan Y. terdapat dua pergerakan kamera pada game 2d, yang pertama yaitu

static view dimana semua objek berada pada satu bidang dan gerakan karakter utama hanya terbatas pada bidang itu saja, contoh dari *static view* adalah Candy Crush Saga, Tetris, pong. Sedangkan yang kedua yaitu *side scroll* view game yang kita mainkan mempunyai kamera yang dapat bergeser ke kanan atau ke kiri sesuai dengan gerakan dan kecepatan karakter, contohnya yaitu Super Mario Bros, Limbo.

2. **Game 2.5D**, yaitu game dimana menggunakan animasi dan latar 3D, tetapi menggunakan cara bermain gaya 2D. dalam game grafik ini sering disebut istilah *isometric*, *diametric* ataupun *trimetric projection*. umumnya game yang dikategorikan 2.5D memiliki gameplay yang cukup simpel (Sutanto dkk., 2020). Contoh game terkenal yang menggunakan grafik ini yaitu Little Nightmare, Clash of Clans.
3. **Game 3D**, yaitu game dengan grafis yang baik dalam penggambaran secara realita dimana ruang dimensinya memiliki 3 elemen, yaitu X, Y, Z. umumnya pemain dapat melihat suatu objek dari sudut 360° sehingga pemain dapat melihat bentuk keseluruhan dari objek tersebut.

Genre Permainan

1. **Aksi (Action)** , genre action menjadi salah satu genre yang paling popular pada saat ini. Ciri khas pada genre ini terdapat unsur berupa aksi menembak, melompat, dan saling pukul. Biasanya pemain memerlukan kecepatan reflek, akurasi dan waktu yang tepat untuk dapat

menyelesaikan tantangan. Contoh game yang terkenal yaitu Point Blank(PB), brawl star.

2. **Strategi** genre strategi umumnya menitikberatkan strategi pemain dalam menjalankan permainan. Dalam genre ini, pemain biasanya dapat mengendalikan hingga beberapa pasukan dan bangunan. Contoh game ber-genre ini yaitu *Age of Empire, Warcraft*.
3. **Teka-teki (puzzle)** Fokus utama pada genre ini adalah memecahkan teka-teki untuk dapat menyelesaikan permainan. Bentuk permaianan ini dapat berupa menyusun balok, menyamakan gambar atau bentuk, menyelesaikan perhitungan matematika, menggeser kotak ke tempat seharusnya. Contoh game genre ini antara lain Sudoku, Tetris.
4. **Role Playing Game** Pada genre ini, pemain akan memainkan peran sebagai tokoh cerita pada game, yang umumnya merupakan tokoh utama. Dalam genre ini Terdapat unsur keterampilan (experience) dimana seiring permainan, karakter pemain juga akan terus berkembang menjadi lebih kuat. Selain itu, pemain dapat menentukan statistik kemampuan karakter berupa kecerdasan, kekuatan, ketahanan, dan sebagainya. Contoh game popular pada genre ini adalah Final Fantasy, Ragnarok.
5. **Petualangan (adventure)** Permainan genre adventure merupakan permainan yang berfokus dalam menjelajahi berbagai tempat, seperti mendaki tebing, menyusuri hutan, dan berayun dari satu pohon kepohon lainnya. Contoh

game genre adventure diantaranya Limbo, the legend of Zelda.

6. **Simulasi** Genre game ini memiliki konsep mirip dengan kenyataan, dimana semua factor dalam game dibuat semirip mungkin dengan dunia nyata. Alur permainan game ini seringkali digambarkan dengan memulai dengan tidak mempunyai apa-apa dengan tujuan menjadi kaya ataupun memiliki usaha yang sukses. Selain alur tersebut, terdapat juga seperti simulasi mengendalikan sarana transportasi, maupun profesi seperti dokter dan teknisi. Contoh game ini yaitu The Sims, Tycoon.
7. **Olahraga(sport)** Game ini memiliki gameplay berbagai jenis olahraga di dunia nyata. dimana pemain akan melakukan pertandingan olahraga secara virtual. Game ini berupa kompetisi antara dua pemain atau lebih, baik secara individu ataupun tim. Contohnya adalah Pro Evolution Soccer (PES), Most Wanted, NBA.

b. Unsur-Unsur *game*

Video Game terdiri dari beberapa unsur dasar yang dikombinasikan sehingga terbentuk pengalaman yang menarik (Gunadi and Fatta, 2012; Fauzhan et al., 2020). Adapun unsur *game* antara lain:

1. Fitur adalah elemen yang menggambarkan permainan dalam bentuk visual ataupun di rasakan.
2. *Gameplay*, adalah cara kerja *game*, dimana fitur-fitur yang tersedia membentuk alur permainan.

3. *Interface*, merupakan tampilan yang tersedia dalam game. Antarmuka yang baik akan membantu pemain merasa betah dan tidak bosan saat memainkan game.
4. *Rules*, adalah seperangkat aturan yang ada untuk memainkan game
5. Desain Level, adalah tingkat kesulitan yang menggambarkan skenario permainan.

2. *Game Top-Down Shooter*

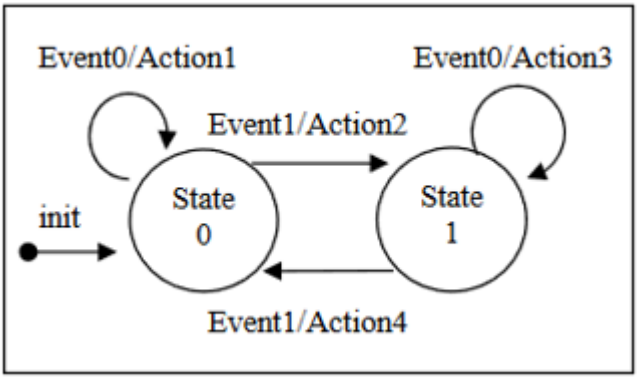
Top-down shooter merupakan cabang dari genre action dengan gameplay peperangan khususnya baku tembak. Perspektif pemain pada game ini diambil pada sudut pandang atas menuju ke bawah, sama seperti saat melihat peta pada game. kontrol Pemain biasanya mengendalikan gerakan karakter atau kendaraan menggunakan kontrol arah, seperti tombol panah atau joystick.

3. *Non Player Character*

Non player character (NPC) disebut juga *autonomous character* adalah karakter pada game yang tidak dapat dimainkan oleh pemain, tetapi dikontrol oleh program komputasi menggunakan sistem kecerdasan buatan. Tujuan dari NPC dapat digambarkan dalam berbagai bentuk. Npc yang bersifat kawan biasanya bertugas membantu pemain dalam menyelesaikan tujuan, menyediakan misi dan peralatan bagi pemain. Selain itu, npc juga dapat berbentuk musuh yang bertujuan agar game lebih seru dan menantang. (Baffa et al., 2017)

4. Finite State Machine

Finite State Machine atau juga disebut finite state automata merupakan sebuah metodologi perancangan sistem kontrol yang menggambarkan tingkah laku atau prinsip kerja sistem. FSM digunakan untuk mengatur aliran eksekusi program, Dengan menggunakan prinsip kerja tiga hal, yaitu: *state* (keadaan), *event* (kejadian) dan *action* (aksi) (Al Tahtawi et al., 2017). FSM sering digunakan dalam berbagai bidang, seperti perhitungan matematis, *artificial intelligence*, games, rambu lalu lintas, mesin minuman dan sebagainya.



Gambar 2.1. Ilustrasi Finite State Machine

Perancangan FSM biasanya memilki dua atau lebih keadaan (*state*), dimana hanya satu keadaan yang hanya boleh aktif secara bersamaan, sehingga mesin perlu berpindah dari keadaan awal ke keadaan lain agar dapat melakukan tindakan yang berbeda. Agar dapat melakukan perpindahan kondisi, sistem perlu mendapat event atau masukan tertentu. Saat sistem terjadi transisi keadaan ini juga disertai aksi, aksi

yang dilakukan dapat berupa aksi sederhana ataupun rangkaian proses yang relatif kompleks (Sifaulloh et al., 2021).

5. Godot Engine

Menurut Situs Godot Engine (2024) Godot engine merupakan game engine sumber daya terbuka (*Open Source*) berlisensi *The MIT License* (MIT) yang digunakan untuk membuat game 2D ataupun 3D dalam berbagai platform, baik window, android, Ios, web, maupun konsol.

Gambar 2.2. Ilustrasi godot engine

Sumber : https://godotengine.org/

Godot Engine sepenuhnya gratis dan tidak terdapat biaya tersembunyi ataupun royalti didalamnya, dikarenakan pengembangan godot ini sepenuhnya independent dan dikembangkan oleh komunitas godot sendiri. Sebagai pengembang indie (*independent*) tentunya ini menjadi kelebihan tersendiri dikarenakan pengembang game tidak

perlu menyediakan dana untuk alokasi lisensi komersial yang umumnya terdapat pada game engine lain seperti *Unity* dan *Unreal*. Bahasa pemograman yang digunakan untuk mengembangkan game di Godot terdapat 2 pilihan, yaitu C dan GDscript yang merupakan turunan dari Bahasa Python.

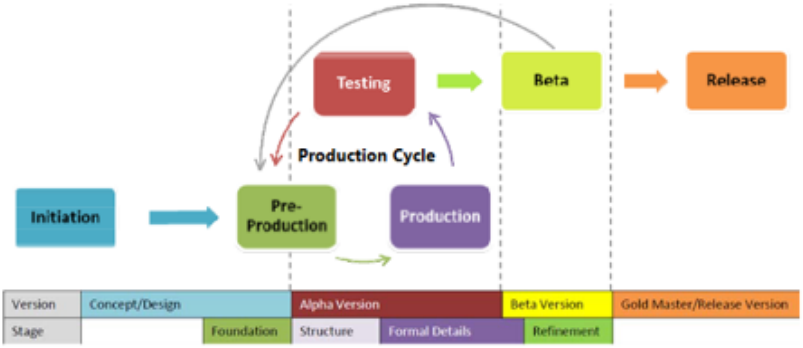
Godot engine mempunyai beberapa fitur yang menjadi nilai kelebihan dibandingkan dengan game engine lainnya, salahsatunya pada arsitektur Node. Node merupakan komponen dalam scene dalam membentuk sebuah game. Node memudahkan pengembang dalam membangun struktur yang komplek serta *scene reuseable*.

Godot engine juga cocok untuk membangun game 2d dikarenakan fitur-fitur yang disediakan dapat mempermudah dan mempercepat proses pengembangan game. fitur tersebut seperti Tilemap editor, yang memudahkan desain peta atau lingkungan pada game dengan *auto-tiling*, rotasi, *custom grid shapes*, dan *multiple layers*. Ukuran dari program Godot engine juga cukup ringan, yaitu hanya kurang dari 200 Mb. Maka dari itu godot engine dapat berjalan dengan lancar dan optimal pada mayoritas PC dengan ram 4 Gb.

Selain itu, Godot engine menyediakan fitur *Visual Scripting*. Dengan *visual scripting* yang disediakan pengembang tidak harus menguasai bahasa pemrograman untuk dapat membangun game di dalamnya. *Visual Scripting* menggunakan grafik visual berbasis node yang merepresentasikan alur logika pada game. jadi, pengembang hanya perlu Menyusun blok logika untuk membuat fungsi tertentu tanpa membutuhkan coding.

6. *Game Development LifeCycle*

Metode *Game Development LifeCycle* atau disingkat GDLC adalah Salah satu framework yang digunakan untuk mengembangkan aplikasi permainan atau biasanya disebut dengan video game yang menerapkan pendekatan iteratif yang terdiri dari 6 fase pengembangan, dimulai dari fase inisialisasi/pembuatan konsep, preproduction, production, testing, beta dan realease (Andriyat Krisdiawan, 2018).



Gambar 2.3. Ilustrasi gdlc

Sumber : Ariyana et al. (2022)

Tujuan dari pemodelan GDLC adalah untuk memanajemen setiap tahap dari dibuatnya game dari awal agar berjalan dengan efisien dan sesuai rencana, apa saja kriteria yang perlu dipertimbangkan dalam setiap tahapan serta untuk menghasilkan game yang berkualitas. Berdasarkan gambar 3.1 tahapan dari GDLC antara lain (Ramadan and Widyani, 2013;

Andriyat Krisdiawan, 2018; Luay and Apriandari, 2024; Ariyana et al., 2022):

a. Inisiasi

Inisiasi adalah tahap awal berupa perancangan konsep game yang akan dibuat. Seperti apa *game* yang dibuat, bagaimana *game* tersebut akan dibangun, serta identifikasi topik dan target user. Hasil dari proses inisiasi adalah konsep *game* dan deskripsi dari *game*. Pada tahap inisiasi akan dijelaskan tentang skenario game yang akan dibuat, karakter dan alur cerita pada *game* yang akan dibangun.

b. Praproduksi

Praproduksi merupakan tahapan perencanaan *game* yang terdiri dari penciptaan desain *game* dan *prototyping*. Desain *game* berfokus pada perancangan desain *game*, Menyusun alur cerita, genre, mekanik permainan, tantangan, alur cerota dan karakter pada *game*. pada tahap ini juga terjadi pembuatan prototipe, pembuatan prototipe bertujuan untuk membuat fondasi dan struktur game.

c. Produksi

Produksi adalah pembuatan *game* berdasarkan konsep, *prototyping*, dan desain yang telah dibuat. Tahap ini merupakan proses inti yang berputar di sekitar penciptaan aset, pembuatan kode sumber, dan integrasi elemen *game*. Tahapan produksi pada umumnya meliputi pembuatan dan penyempurnaan detail formal menyeimbangkan, meningkatkan kinerja, menambahkan fitur yang baru, serta memperbaiki bug (terkait dengan

fungsional dan penyelesaian internal kriteria kualitas).

d. Testing

Testing atau pengujian *alpha* merupakan tahapan untuk melakukan pengujian terhadap *game* yang telah dibuat. Pengujian dilakukan oleh tim internal pengembang *game*. Penggujian yang dilakukan yaitu terkait fungsionalitas fitur dan keseimbangan *gameplay* pada permainan. Metode yang digunakan untuk menguji fungsionalitas fitur adalah dengan melakukan *Playtesting*, yaitu mencoba untuk memainkan *game* beberapa kali. Sedangkan untuk menguji keseimbangan pada *game*, dilkukan dengan cara menguji game secara detail dengan cara melakukan *feedback* antar tim penguji apakah game tersebut membosankan, terlalu sulit, ataupun kurang menantang. Hasil dari testing berupa laporan *bug*, permintaan perubahan, serta keputusan pengembangan. kemudian akan memutuskan apakah sudah waktunya untuk maju ke fase berikutnya (*Beta*) atau mengulangi siklus produksi.

e. Beta

Setelah melalui tahap *alpha*, tahap selanjutnya yaitu melakukan testing ke pihak eksternal atau orang dari luar pihak pengembang. Pengujian beta masih menggunakan metode pengujian yang sama dengan metode pengujian sebelumnya. Pengujian beta terdiri atas dua jenis, yaitu **beta tertutup** dan **beta terbuka**. Pada beta tertutup pengujian hanya dilakukan peserta yang diundang untuk menguji permainan. Sementara beta terbuka, memungkinkan siapa saja yang mendaftar menjadi peserta penguji.

f. Rilis

Tahap rilis adalah tahap terakhir pada pengembangan *game*. *game* yang telah di produksi kemudian di *build* ke dalam ekstensi yang dapat dieksekusi oleh sistem tujuan. Pada tahap ini, game mulai dipasarkan ke publik secara langsung ataupun dengan melalui distributor pihak ketiga seperti *steam*, *app store*, ataupun *play store*.

7. Unified Modeling Language

Unified Modelling Language (UML) merupakan suatu metode dalam pemodelan secara visual yang digunakan sebagai sarana perancangan sistem berorientasi objek (*object oriented*). UML digunakan untuk membantu membuat rencana dan memahami sistem perangkat lunak yang rumit dengan lebih mudah. UML terdiri dari berbagai diagram yang masing-masing merepresentasikan aspek tertentu dari sistem. Diagram-diagram ini memberikan pandangan yang jelas tentang bagaimana komponen-komponen berhubungan satu sama lain dan bagaimana data dan informasi mengalir dalam sistem. Beberapa *diagram Unified Modeling Language* yang paling umum digunakan termasuk *class diagram, activity diagram*, dan lain sebagainya.

8. Blackbox Testing

Blackbox testing merupakan proses pengujian perangkat lunak yang berfokus pada fungsionalitas perangkat lunak tanpa mengetahui susunan kode dan desain dari perangkat lunak tersebut. Pengujian blackbox berfokus pada fungsi, masukan

dan keluaran dari perangkat lunak sesuai dengan spesifikasi yang dibutuhkan (Cholifah et al., 2018).

Pengujian black box bertujuan untuk menemukan fungsi yang tidak sesuai, kesalahan antarmuka, kesalahan pada struktur data, kesalahan performansi, serta kesalahan inisialisasi (Wijaya and Astuti, 2021).

B. Google Play Store

Google play store merupakan layanan konten digital khusus sistem operasi android yang memungkinkan pengguna untuk menelusuri dan mengunduh aplikasi yang dikembangkan dengan Android software development kit (SDK). Play store menjadi tempat penyedia produk-produk digital seperti aplikasi, musik atau lagu, buku, game maupun pemutar media berbasis cloud baik gratis atau berbayar.

C. Kajian Penelitian Relevan

Kajian hasil penelitian relevan merupakan refrensi mengenai penelitian yang sudah dilakukan oleh pihak lain dan mendapatkan hasil yang valid sesuai dengan judul dan tujuan peneliti. Tujuan dari kajian penelitian relevan di gunakan sebagai pendukung terhadap penelitian yang akan dilakukan.

Tabel 2.1. Tabel Kajian Penelitian Relevan

No	Papper	Motode	Uji Data	Hasil Pembahasan

1	PENERAPAN METODE FINITE STATE MACHINE PADA NON PLAYER CHARACTER (NPC) GAME ACTION STRATEGY "OUROBOROS" (Safitra et al., 2020)	Pengem-banggan Multimedia	Dilakukan dengan metode blackbox serta kuisioner	Ai pada enemy dan object game berfungsi melakukan state sesuai instruksi
2	Implementasi Finite State Machine Untuk Npc Pada Game 2d Side-Scroll Shooter (Enggar et al., 2022)	Flowchart sistem game, dan use case diagram	Dilakukan dengan metode blackbox serta kuisioner	kecerdasan buatan dapat diimplementasi dengan baik untuk mengatur tingkah laku dari setiap enemy NPC yang berbeda
3	Educational Game Based Role Playing Games with Finite State Machine Method (Winarno et al., 2022)	Literature study, Game design and development	-	Menampilkan misi berupa pertanyaan, jika benar akan berlanjut ke soal berikutnya. Jika salah akan mengulangi dari awal

4	Penerapan Metode Finite State Machine pada Game “Santri on the Road” (Sifaulloh et al., 2021)	Observasi pada game terkait, kualitatif	eksperiment fitur	menangani logika perilaku player saat state running, jump, menunduk, dan game over
5	Implementasi Finite State Machine (FSM) Dalam Game Monopoli 3D Teknologi Informasi dan Komunikasi dengan Algoritma Fisher-Yates Shuffle Berbasis Android (Setio Utomo and Arwin Dermawan, 2022)	Multimedia Development Live Cycle (MDLC).	Uji fungsionalitas, blackbox testing dan kuisioner	pemodelan AI Finite State Machine pada NPC digunakan untuk pengambilan keputusan, sedangkan Algoritma Fisher-Yates Shuffle berfungsi untuk melakukanpengacakan soal

6	Mathematics Education Game Using the Finite State Machine Method to Implement Virtual Reality in Game Platformer (Laksono and Susanto, 2020)	Multimedia Development Live Cycle (MDLC).	Pengujian blackbox serta User Aceptance Test (UAT) berupa kuisioner kepada guru dan siswa sd	Player bertemu dengan NPC yang mengharuskan menjawab pertanyaan dengan benar untuk memperoleh 1 item nya.
7	PENERAPAN METODE FINITE STATE MACHINE (FSM) PADA GAME AGENT LEGENDA ANAK BORNEO (Yulsilviana and Ekawati, 2019)	Flowchart serta storyboard dan blok diagram	Pengujian Blackbox dengan menguji semua fitur yang telah dibuat	NPC musuh yang telah diprogram untuk mendeteksi, mengejar dan menangkap player. Jika diluar jangkauan maka npc akan kembali ketempat awal
8	PENERAPAN A* PATHFINDING DAN FSM (FINITE STATE MACHINE) PADA GAME "LOST CIVILIZATION" BERBASIS ANDROID (Satrio et al., 2022)	Menggunakan metode GDLC	Pengujuan algoritma astar mengunakan whitebox sedangkan fitur game menggunakan blackbox	Penerapan finite state machine dan algoritma astar sebagai pathfinding jalur menghasilkan enemy yang akan mengejar player jika memenuhi jarak maksimal.

9	Pemodelan Perilaku Musuh Menggunakan Finite State Machine (FSM) Pada Game Pengenalan Unsur Kimia (Bimantoro and Haryanto, 2016)	Metode Game development lifecycle (GDLC)	Kuisioner, studi pustaka dan eksperimen	Metode finite state machine menghasilkan Model perilaku musuh akan digunakan untuk penghitungan pemodelan selanjutnya dan untuk merubah performa objek-objek yang ada didalam game
10	Design an enemy non-player character in maze game using finite state machine algorithm (Refnaldi, 2023)	Metode Game development lifecycle (GDLC)	Observasi program dan Kuisioner	FSM berjalan dengan baik dan musuh mampu menjalankan semua state *(State Idle, Patrolling, Chasing, Player run away from NPC, Room Restart*)

Berdasarkan penelitian ilmiah yang telah di pelajari diatas, diperoleh bahwa dalam mengimplementasikan Finite State Machine, diperlukan desain alur perilaku pada sistem yang dibuat. Selain itu, metode pengembangan yang umum digunakan dalam penelitian diatas adalah *Multimedia development lifecycle* (MDLC) dan *Game development lifecycle* (GDLC).

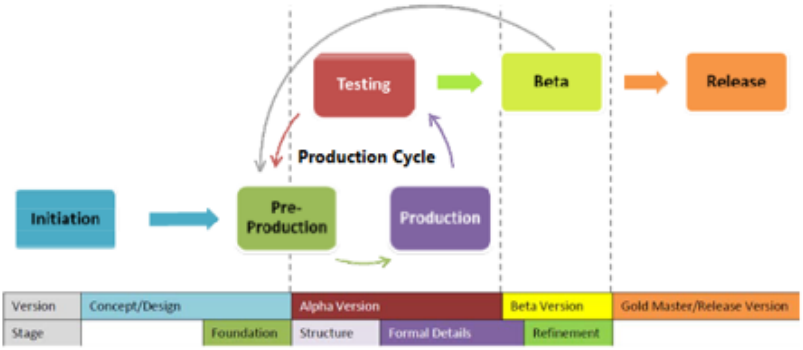
Perbedaan penelitian ini dibandingkan pada beberapa kajian penelitian pada tabel 2.1 yaitu penelitian ini membahas mengenai penerapan FSM yang diterapkan pada permainan *top down shooter*. Permainan ini dikembangkan menggunakan

perangkat lunak Godot Engine dengan metodologi penelitian GDLC.

BAB III

METODOLOGI PENELITIAN

Metode pengembangan sistem yang digunakan untuk membuat *game Survivor Shooter* ini adalah menggunakan *Game Development LifeCycle*. GDLC terdiri dari 6 tahap yaitu fase inisiasi/pembuatan konsep, *preproduction, production, testing, beta* dan *realease* Setiap tahap harus dikerjakan secara berurutan, namun dapat Kembali ketahap sebelumnya jika terjadi perbaikan, penambahan module, dan sebagainya.



Gambar 3.1. Ilustrasi gdlc

Sumber : Ariyana et al. (2022)

A. Inisiasi

Proses inisiasi diawali dengan melakukan analisis terhadap *game* ber-genre *action-shooter* 2D yang terdapat pada *Play*

store, yaitu *Beastria, Ailment*, dan *Us in Space*. ketiga *game* tersebut dipilih karena memiliki genre dan *gameplay* yang hampir sama.

Game yang akan dibangun merupakan game genre *action-shooter* 2D berjudul *Survivor Shooter*. Agar *game* lebih menantang, dibutuhkan musuh yang dapat bertindak sesuai dengan logika yang diterapkan.

Metode yang digunakan untuk mengatur perilaku musuh pada game ini adalah menggunakan *Finite State Machine*, karena metode ini sederhana dan mudah diterapkan. Dimensi grafis pada *game Survivor Shooter* menggunakan tipe 2D, dimana memiliki perspektif melihat permainan dari atas seperti melihat peta. Pemain dituntut untuk bertahan dan mengalahkan musuh hingga mencapai target tertentu. Berikut merupakan analisa dari game Survivor Shooter:

1. Genre pada game ini adalah action dengan subgenre shooter
2. Menggunakan grafis 2D
3. Tindakan karakter musuh di implementasikan dengan metode Finite State Machine
4. *Game* dibangun dengan Bahasa GDscript menggunakan Godot Engine.

Berdasarkan analisa topik game tersebut, pengembang perlu menjabarkan lebih lanjut guna mendapatkan spesifikasi yang terperinci, yaitu:

1. Analisis Kebutuhan Perangkat

Analisis kebutuhan perangkat menggambarkan kebutuhan perangkat lunak dan perangkat keras yang dibutuhkan oleh Sistem untuk mengembangkan permainan serta memainkan permainan *Survivor Shooter*

2. Konsep Permainan

Konsep Permainan akan menjabarkan mengenai ide, gambaran umum, serta gagasan mengenai sistem dan objek yang akan dibuat pada *game Survivor Shooter*

B. Pra-produksi

Praproduksi atau desain merupakan tahapan yang dilakukan setelah proses inisiasi. Pada tahap ini, dilakukan perancangan game serta rencana produksi game, yang terdiri atas *game design* yakni penyempurnaan konsep permainan dan pembuatan *prototype* dari *game*. Desain yang dibuat meliputi desain *class diagram*, *User Interface*, diagram FSM, serta *asset image* yang akan digunakan pada *game*.

C. Produksi

Production atau juga disebut implementasi merupakan tahapan inti pada metodologi GDLC. Pada tahap ini, desain yang telah dirancang kemudian disusun untuk menghasilkan suatu sistem sesuai dengan tujuan perancangan. Pada pengembangan game “Survivor Shooter”, perangkat lunak yang digunakan untuk membangun game ini yaitu Godot Engine.

Godot Engine dipilih karena peformanya yang cukup ringan dan tidak membutuhkan spesifikasi komputer yang tinggi. Selain itu, Godot engine merupakan *software open source* yang membuat perangkat lunak ini sepenuhnya gratis, hal tersebut berarti tidak terdapat biaya tersembunyi ataupun royalti didalamnya.

Bahasa pemogramman yang digunakan pada proses implementasi Finite State Machine serta keseluruhan permainan ini adalah dengan menggunakan GDScript. GDScript merupakan Bahasa pemogramman tingkat tinggi bersifat *object oriented*. Bahasa ini menggunakan sintak berbasis indentasi mirip seperti bahasa Python.

D. *Alpha Test*

Alpha test merupakan pengujian yang dilakukan oleh pengembang *game* atau pihak internal yang bertujuan untuk dapat mengidentifikasi kesalahan dan juga fitur yang tidak sesuai dengan persyaratan yang sebenarnya (Wibisono and Baskoro, 2002). Jika ditemukan *bug* ataupun kesalahan yang terdapat pada *software* maka tahapan akan kembali dalam proses *production*. Pengujian yang dilakukan untuk menguji *game* "Survivor Shooter" antara lain (Pressman, 2001; Rifai, 2015):

1. *Unit Testing*

Unit testing adalah pengujian yang difokuskan pada unit terkecil pada desain perangkat lunak (komponen atau modul perangkat lunak). *Unit testing* dilakukan untuk memastikan

bahwa setiap unit kode dan komponen perangkat lunak dapat bisa bekerja sesuai harapan (Hasibuan and Dirgahayu, 2021).

Pada tahapan ini, FSM yang telah diimplementasikan pada objek akan diuji fungsionalitasnya untuk mendapatkan hasil yang sesuai dengan *class diagram* yang telah ditentukan. Metode yang digunakan untuk melakukan *testing* ini adalah dengan menggunakan *blackbox testing*.

Hasil dari pengujian unit testing yang terdapat pada lampiran 1 kemudian dihitung rata-rata serta presentase ketercapaian dengan rumus:

$$\bar{x} = \frac{\sum x}{n} \times 100\% \tag{3.1}$$

keterangan:

$\bar{x}$ = Skor rata-rata
$\sum x$ = Skor toal item
n = Jumlah item

2. *Integrated Testing*

Integration testing adalah sebuah level dari pengujian perangkat lunak dimana modul-modul yang berdiri sendiri digabungkan dan diuji sebagai sebuah kesatuan. Sehingga pengujian ini dapat menampilkan kesalahan dalam interaksi antar unit. Pengujian integrasi dilakukan untuk mendeteksi kesalahan atau ketidaksesuaian antarmuka atau keluaran dari setiap modul yang telah diintegrasi ke dalam sistem. (Herlambang et al., 2019).

Pada tahap ini, desain *user interface* yang telah disusun akan diuji fungsionalitasnya apakah tampilan serta fitur yang ada pada modul tersebut sudah sesuai dengan tujuan. *Integration testing* dilakukan dengan menggunakan *blackbox testing*.

Tabel 3.1. Tabel rencana pengujian integrasi

Pengujian	**Skenario**	**Hasil yang diharapkan**
Membuka menu utama	membuka permainan	Halaman menu utama ditampilkan setelah boot selesai
Stage list	Pada halaman utama, menekan tombol jelajah	Menampilkan stage permainan yang telah terbuka
Workshop	Pada halaman utama, menekan tombol workshop	Menampilkan workshop yang terdapat statistik senjata serta biaya upgrade
Upgrade Senjata	Pada halaman workshop, melakukan upgrade senjata	Saat koin mencukupi, statistik senjata terpilih akan meningkat
Option	Pada halaman utama, menekan tombol option kemudian melakukan perubahan dan disimpan	Menampilkan stage permainan. setelah menyimpan konfigurasi, perubahan diterapkan

Halaman about	Pada halaman utama, menekan tombol about	Menampilkan profil permainan dan pengembang
Memilih stage	Pada halaman stage permaian, memilih salah satu stage yang telah terbuka	Menampilkan stage permainan sesuai yang dipilih
Virtual joystick	Menggerakan v*irtual joytick* yang terdapat pada permainan	*Virtual joystick* bergerak sesuai letak sentuhan

Hasil dari setiap pengujian tersebut kemudian dihitung rata-rata serta presentase ketercapaian dengan rumus:

$$\bar{x} = \frac{\sum x}{n} \times 100\% \tag{3.2}$$

keterangan:

$\bar{x}$ = Skor rata-rata
$\sum x$ = Skor total item tercapai
n = Jumlah item

3. *System Testing*

System Testing merupakan pengujian dimana perangkat lunak yang diuji sudah lengkap dan terintegrasi. Tujuan dari pengujian sistem untuk memastikan sistem memenuhi

semua persyaratan dan berperilaku seperti yang diharapkan oleh pengguna (Khoirunningsih, 2018).

Teknik pengujian yang dilakukan pada permainan ”Survivor Shooter” adalah dengan melakukan Pengujian Instalasi dan kompabilitas terhadap beberapa perangkat Android dengan versi sistem operasi dan resolusi yang berbeda, yang diantara lain:

a. Sistem operasi:

1. Android 9
2. Android 10
3. Android 12

b. Resolusi layar:

1. HD (720 x 1280 piksel)
2. Full HD (1920 x 1080 piksel)
3. QHD (1600x2560 piksel)

E. *Beta Test*

1. *User Acceptance Testing*

User acceptance testing adalah tahapan pengujian software untuk memastikan produk yang dikembangkan sesuai dengan kebutuhan dan harapan end user. Pada tahap ini, *end user* yang terpilih melakukan *testing* terhadap fungsi-fungsi aplikasi dan melaporkan permasalahan yang ditemukan. Tujuan dari

pengujian ini adalah untuk mengetahui tingkat kelayakan dari perangakat lunak serta kenyamanan user dalam menggunakan sistem serta dapat menyelesaikan permasalahan yang dihadapi sebelum dirilis (Hady et al., 2020).

Metode yang digunakan pada UAT ini adalah dengan melakukan kuisioner dengan pertanyaan sesuai dengan tabel 3.2

Tabel 3.2. Tabel kuisioner UAT

Kode	**Pertanyaan**
fungsionalitas	
F01	Tata letak komponen terlihat rapi
F02	Indikator (healthbar, amunisi, dan sebagainya) telihat dengan jelas
F03	Pemain mengerti mengenai istilah yang terdapat pada game
F04	Virtual analog berjalan dengan semestinya
F05	Penyajian audio visual mendukung permainan
F06	Proses loading permainan berjalan lancar
F07	Fitur upgrade senjata berjalan dengan semestinya
gameplay	
G01	Permainan memberikan tujuan misi dengan jelas

G02	Pemain mendapat hadiah yang sebanding dengan kesulitan permainan
G03	Karakter pemain dapat dikontrol dengan semestinya
G04	Tantangan yang semakin sulit seiring bertambahnya stage permainan
G05	Karakter musuh memiliki perilaku yang memberikan tantangan lebih pada permainan

Kuisioner tersebut kemudian diberikan kepada mahasiswa UIN Walisongo yang memliki hobi bermain video game, komunitas Google Developer School Club (GDSC) UIN Walisongo, serta kepada beberapa ahli dibidang *game dev*. Adapun skala jawaban tiap pertanyaan tersebut mengacu pada Mean Opinion Score (MOS) dengan ketentuan sebagai berikut (Nugroho and Wijayanto, 2022):

Tabel 3.3. Tabel Mean Opinion Score UAT

MOS	**Keterangan**	**Nilai**
SS	Sangat setuju	5
S	Setuju	4
C	Cukup	3
KS	Kurang setuju	2
TS	Tidak setuju	1

Setelah mendapat nilai dari kuisioner tersebut kemudian dihitung presentase kelayakan menggunakan rumus berikut:

$$\bar{x} = \frac{\sum x}{n} \times 100\% \quad (3.3)$$

keterangan:

$\bar{x}$ = Skor rata-rata
$\sum x$ = Skor total item
n = Jumlah item

Hasil dari presentase tersebut kemudian dicocokan dengan predikat skala Likert.

Tabel 3.4. Tabel skala likert

Presentase	**keterangan**
0% - 20%	Sangat kurang layak
21% - 40%	Kurang layak
41% - 60%	Cukup
61% - 80%	Layak
81% - 100%	Sangat layak

F. Rilis

Release adalah tahap dimana final build dari game resmi dirilis. Game yang sudah selesai dibuat dan lulus beta testing, maka game dinyatakan siap untuk dirilis ke publik. Tahap rilis akan dilakukan dengan melalui layanan google play console developer. Nantinya game yang telah berhasil di verifikasi oleh pihak google akan dapat diunduh melalui Google play store. Untuk dapat dirilis di play store, rencana data profil aplikasi adalah sebagai berikut:

Tabel 3.5. Tabel data aplikasi

Tipe aplikasi	Game
Nama aplikasi	Survivor Shooter
Pengembang	Supersemar dev
Bundle ID	Com.supersemar.survivor_shooter
Platform	Android

BAB IV

HASIL DAN PEMBAHASAN

A. Inisiasi

1. Analisis Kebutuhan

a. Kebutuhan pengembang permainan

Berikut Spesifikasi perangkat yang digunakan oleh pengembang dalam membangun game ini:

1. Sistem operasi Windows 10
2. Software Godot Engine versi 4.2
3. Software pixel viewer

b. Kebutuhan perangkat user

Spesifikasi perangkat yang direkomendasikan untuk memainkan game ”Survivor Shooter” adalah sebagai berikut:

1. Sistem operasi Android minimal 8.0
2. Ram minimal 1.0 Gb
3. Ruang penyimpanan minimal 500 Mb

2. Konsep Permainan

a. Tujuan Permainan

Tujuan pada *game* ”Survivor Shooter” ini yaitu bertahan hidup dan mencapai target kemenangan. Target kemenangan terdapat 3 mode, di mana hanya 1 mode yang dipakai untuk setiap stage permainan, yaitu:

1. Mengalahkan semua musuh
2. Bertahan hingga waktu habis
3. Mencapai goal point atau titik poin kemenangan.

b. Awal Permainan

Saat permainan pertama kali dimainkan atau di install, berikut merupakan ketentuan awal permainan:

1. Pemain memiliki level 1
2. *Health point* yang dimiliki pemain adalah 100, yang dapat ditingkatkan sesuai dengan level pemain.
3. Semua senjata memliki level 1
4. Pemain memiliki koin awal 1000, koin didapatkan dari mengalahkan musuh serta menyelesaikan misi

c. Aturan Permainan

Aturan dalam permainan Survivor Shooter yaitu:

1. Pemain dapat menggerakan karakter melalui *virtual joystick* yang terdapat pada layar bagian kiri bawah
2. Pemain dapat mengganti senjata pada saat permainan berlangsung
3. Pemain dapat mengisi ulang peluru dengan menekan tombol reload
4. Pemain dapat membidik lawan dengan cara menekan tombol bidik di bagian kanan bawah layer

5. Jika lawan terkena peluru pemain, maka *health point* lawan akan berkurang. Jika *health point* lawan kurang dari sama dengan (<=) 0, maka musuh mati.
6. *Health point* pemain akan berkurang Jika pemain terkena hit dari lawan. Jika *health point* pemain kurang dari sama dengan (<=) 0, maka permainan kalah dan berakhir
7. Musuh yang kalah memiliki kesempatan menjatuhkan barang, yaitu: *Medkit, repair kit, koin*
8. Setiap menggunakan senjata, durabilitas akan menurun. jika durabilitas sama dengan 0, maka senjata tidak bisa digunakan
9. Pemain dapat mengambil *repair kit* untuk mengisi durabilitas senjata
10. Pemain dapat mengambil *Medkit* untuk memulihkan *health point*
11. Pemain dapat mengambil koin untuk mendapatkan uang tambahan
12. Jika pemain berhasil menyelesaikan tujuan misi, maka permainan berakhir, pemain dinyatakan menang

d. Fitur permainan

Fitur yang terdapat pada permainan antara lain:

1. Terdapat 3 stage, pemain dapat mengakses stage selanjutnya jika berhasil menyelesaikan stage terakhir.
2. Pemain dapat mengumpulkan 3 senjata yang berbeda, yaitu: Pistol, Assault Rifle, dan Shotgun

3. Koin merupakan mata uang pada pada permainan, yang bisa didapatkan ketika mengalahkan musuh dan menyelesaikan stage.
4. Pemain dapat meningkatkan statistik senjata menggunakan koin

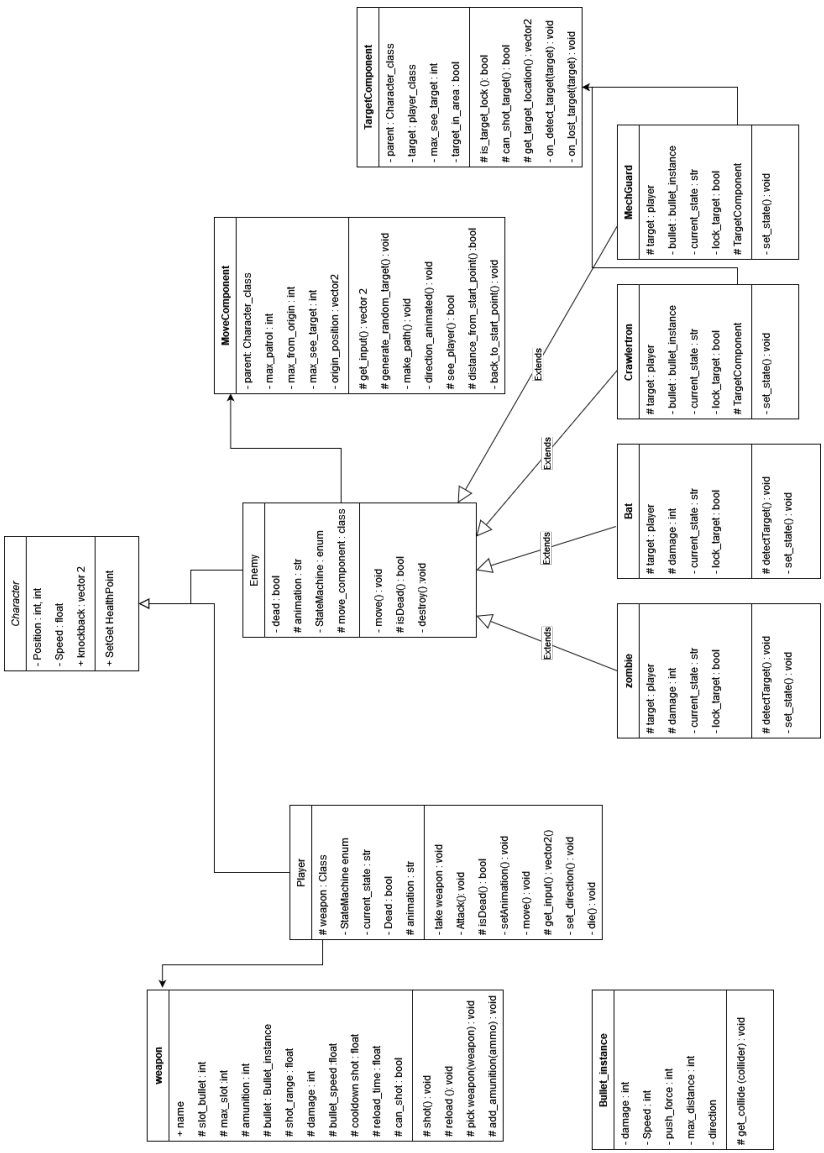
B. Pra-produksi

1. Perancangan Class Diagram

Class diagram merupakan salah satu jenis diagram pada UML (*unified modeling language*) yang menggambarkan dengan jelas struktur berupa deskripsi *class*, atribut, metode, dan hubungan dari setiap objek. Class diagram berfungsi untuk memudahkan merancang *class-class* yang nantinya dibutuhkan pada *game Survivor Shooter*.

Berdasarkan gambar 4.1 *Class character* merupakan *Class* utama yang dirancang. Dari *class character* ini, kemudian dibuat 2 *class* turunan (*inheritance*) yaitu *class player* serta *class Enemy* yang dapat diturunkan untuk merancang objek musuh baru. *Classs weapon* berfungsi untuk membangun objek senjata yang dapat digunakan oleh pemain. Sedangkan *class moveComponent* dan *TargetComponent* dibuat terpisah dari *class enemy* agar setiap objek hanya memiliki attribute dan method yang dibutuhkan saja.

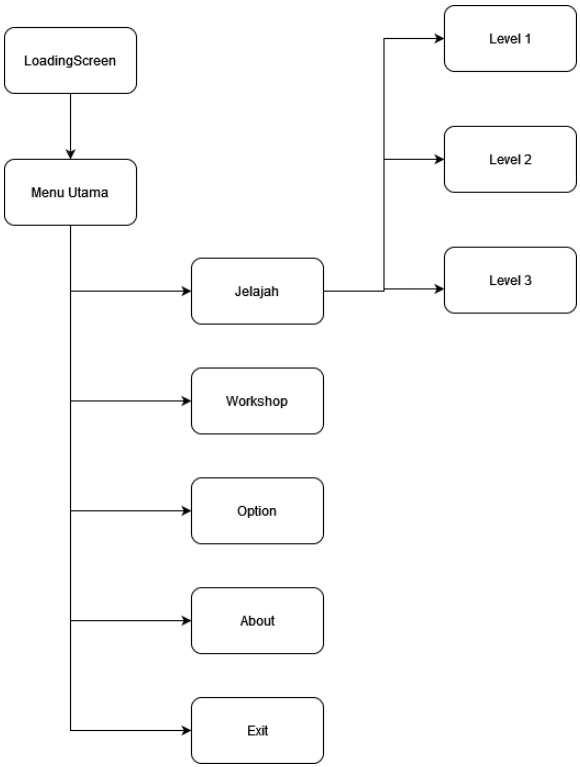
Selain *class character*, terdapat *class weapon* yang berfungsi untuk membentuk objek senjata. Serta *class bullet* sebagai *blueprint* peluru yang ada dipermainan.



Gambar 4.1. Class Diagram game survivor shooter

2. Struktur Menu

Struktur menu pada *game* ini memiliki 1 menu utama, yang memiliki terdapat 5 sub menu, yaitu jelajah, workshop, pengaturan, about, dan keluar.



Gambar 4.2. Struktur menu permainan

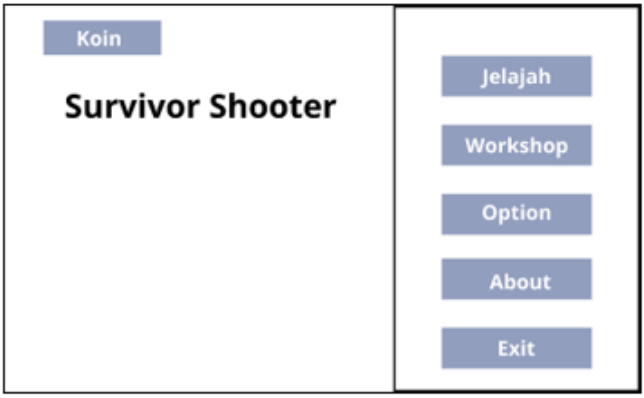
3. Perancangan User Interface

Perancangan user interface berfungsi untuk membuat gambaran mengenai tampilan yang akan dibangun pada game. Tujuan dari desain UI adalah untuk membuat desain

antarmuka yang membuat pengguna mudah untuk dipahami dan menyenangkan.

a. Menu utama

Menu utama merupakan halaman awal yang akan ditampilkan setelah *loading screen* selesai. pemain dapat berpindah tujuan halaman sesuai dengan button yang di klik pada halaman utama



Gambar 4.3. Desain menu utama permainan

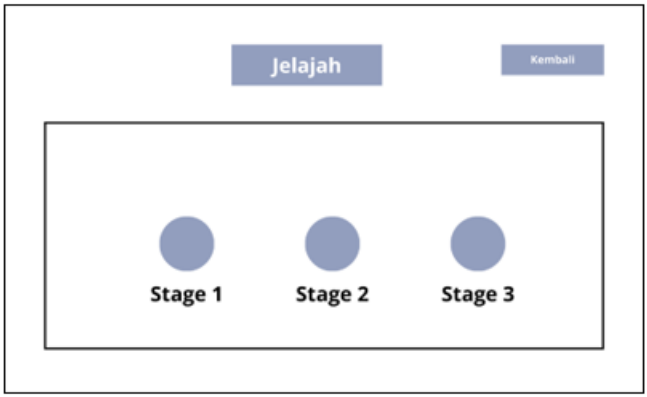
Penjelasan :

1. Pada bagian kiri terdapat navigasi utama yang menampilkan button untuk berpindah ke halaman terkait.
2. Tombol jelajah berfungsi untuk berpindah ke menu memilih level permainan
3. Tombol workshop akan mengarahkan pengguna ke halaman peningkatan senjata
4. Tombol option berguna untuk menampilkan menu pengaturan permainan.

5. Tombol about bertujuan untuk menampilkan halaman tentang yang berisi informasi permainan
6. Tombol exit berfungsi untuk keluar dari permainan
7. Terdapat informasi mengenai jumlah koin yang terletak pada kanan atas layer.

b. Menu jelajah

Pada Halaman jelajah, Pemain dapat memilih level yang tersedia. Ketika salah satu level disentuh, maka akan berpindah ke level yang dituju.



Gambar 4.4. Desain menu jelajah permainan

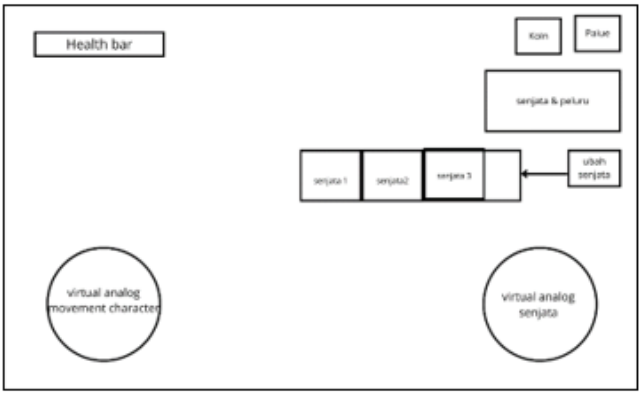
Penjelasan:

1. Terdapat tombol stage 1, yang Ketika dipilih maka akan menuju kehalaman level 1
2. Terdapat tombol stage 2, yang Ketika dipilih maka akan menuju kehalaman level 2

3. Terdapat tombol stage 3, yang Ketika dipilih maka akan menuju kehalaman level 3

c. Menu Level Permaian

Halaman yang menampilkan arena permainan.



Gambar 4.5. Desain menu level permainan

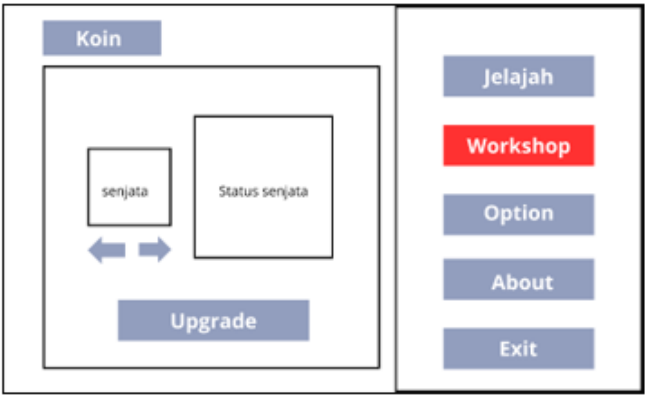
Penjelasan:

1. Presentasi healthpoint ditampilkan di kiri atas layar
2. Senjata dan peluru tersisa dapat dilihat pada kanan atas layer
3. Terdapat 2 analog yaitu analog untuk berjalana pada sisi kiri, dan analog untuk membidik pada sisi kanan
4. Untuk mengganti senjata dapat dilakukan melalui ubah senjata yang terdapat pada bagian bawah senjata

d. Menu Workshop

Halaman workshop menampilkan senjata serta statistik

senjata pada permainan.



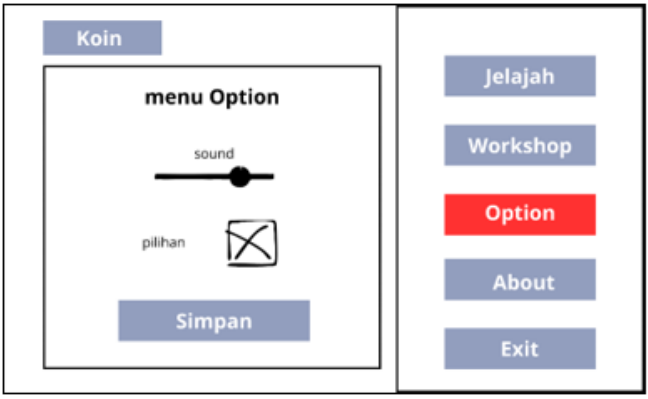
Gambar 4.6. Desain menu workshop permainan

penjelasan:

1. Terdapat gambar senjata yang akan di tingkatkan pada bagian kiri
2. Sedankan pada kanan akan menampilkan stastistik senjata awal serta stastistik setelah di tingkatkan
3. Tombol upgrade berfungsi untuk meningkatkan stastistik senjata sesuai harga koin

e. Menu Option

Halaman option berisikan pengaturan yang dapat diubah sesuai dengan keinginan pengguna.



Gambar 4.7. Desain menu option u permainan

penjelasan:

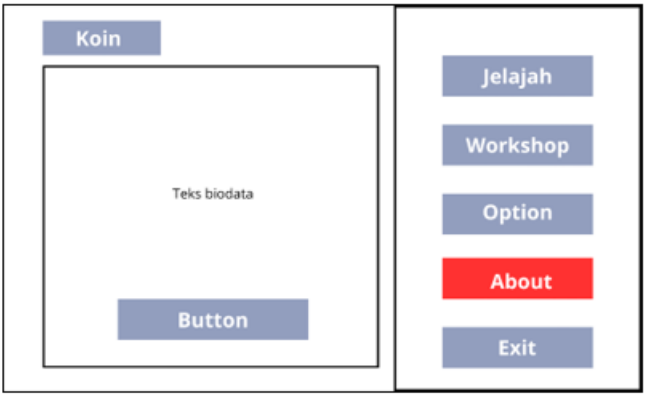
1. Untuk mengatur suara tinggi rendahnya suara, dapat menggunakan slider sound
2. Tombol simpan untuk menyimpan perubahan yang telah dilakukan

f. Menu About

Pada Halaman about terdapat informasi mengenai game, kontak ke pengembang, serta syarat dan ketentuan permainan.

Penjelasan:

1. Menampilkan data dan kredit pengembang
2. Menampilkan syarat dan ketentuan *game*



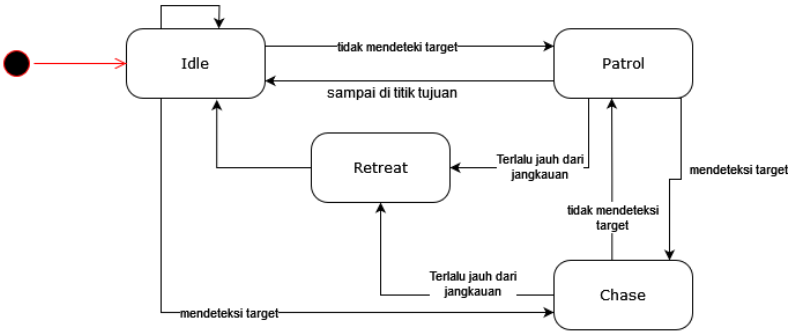
Gambar 4.8. Desain menu about permainan

4. Perancangan Finite State Machine

Implementasi metode *finite state machine* pada program *game* "*Survivor Shooter*" berfokus pada perancangan logika dan animasi karakter musuh serta objek senjata. Pada karakter musuh, pengembang membagi karakter musuh menjadi 2 tipe, yaitu tipe serangan jarak dekat dan tipe serangan proyektil.

a. Penerapan FSM pada karakter musuh jarak dekat

Penerapan FSM pada karakter musuh jarak dekat saat dimulainya program maka akan menuju *state idle*. Jika mendeteksi karakter pemain, maka *state* akan beralih ke *chase*. Sedangkan jika tidak mendeteksi pemain maka musuh akan berpratoli. Jika jarak karakter musuh terlalu jauh dari titik awal, maka unit akan mundur menuju titik awal.



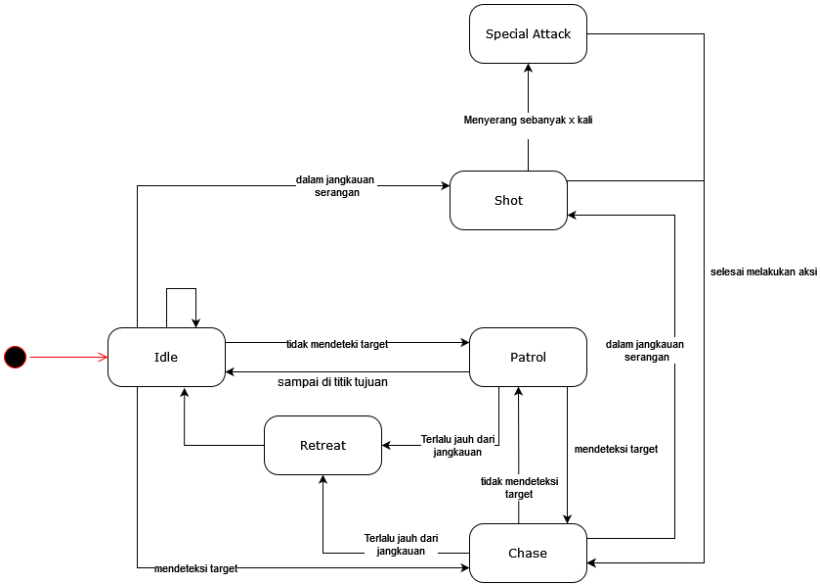
Gambar 4.9. Desain FSM musuh jarak dekat

Tabel 4.1. Tabel FSM karakter jarak dekat

Keadaan	**Kejadian**	**Tujuan**
Idle	Tidak mendeteksi target	Patrol
Idle	Mendeteksi target	Chase
Patrol	Sampai pada titik tujuan	Idle
Patrol	Mendeteksi target	chase
Patrol	Terlalu jauh terhadap titik awal	Retreat
Chase	Tidak mendeteksi target	Patrol
Chase	Terlalu jauh terhadap titik awal	Retreat
Retreat	Sampai pada titik awal	Idle

b. Penerapan FSM pada karakter musuh jarak jauh

Penerapan FSM pada karakter musuh jarak jauh saat dimulainya program maka akan menuju *state idle*. Jika mendeteksi karakter pemain, maka *state* akan beralih ke *chase*. Sedangkan jika tidak mendeteksi pemain maka musuh akan berpratoli. Saat karakter pemain berada pada jangkauan serangan, maka *state* akan berpindah ke *shoot*. Jika jarak karakter musuh terlalu jauh dari titik awal, maka unit akan mundur menuju titik awal.



Gambar 4.10. Desain FSM musuh jarak jauh

Tabel 4.2. Tabel FSM karakter jarak jauh

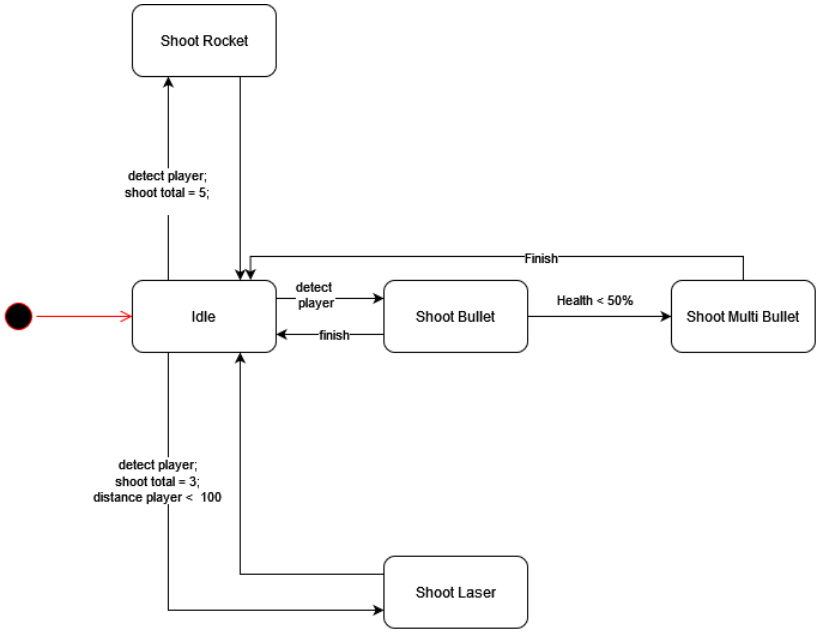
Keadaan	Kejadian	Tujuan

Idle	Tidak mendeteksi target	Patrol
Idle	Mendeteksi target	Chase
Idle	Dalam jangkauan serangan karakter	Shoot
Patrol	Sampai pada titik tujuan	Idle
Patrol	Mendeteksi target	chase
Patrol	Terlalu jauh terhadap titik awal	Retreat
Patrol	Dalam jangkauan serangan karakter	Shoot
Chase	Tidak mendeteksi target	Patrol
Chase	Terlalu jauh terhadap titik awal	Retreat
Chase	Dalam jangkauan serangan karakter	Shoot
Shoot	Telah melakukan aksi	Chase
Shoot	Telah menyerang beberapa kali	Special attack
Special Attack	Selesai melakukan special attack	chase
Retreat	Sampai pada titik awal	Idle

c. Penerapan FSM pada Boss

Penerapan FSM pada Boss saat dimulainya program maka akan menuju state idle. Jika mendeteksi karakter pemain,

dengan total tembakan tidak sama dengan 3 atau 5 maka state akan beralih ke Shoot bullet. Jika mendeteksi karakter pemain, dengan total tembakan sama dengan 3 dan jarak dengan pemain < 100 maka state akan beralih ke Shoot Laser. Sedangkan Jika mendeteksi karakter pemain, dengan total tembakan sama dengan 5 maka state akan beralih ke Shoot rocket.



Gambar 4.11. Desain FSM Boss

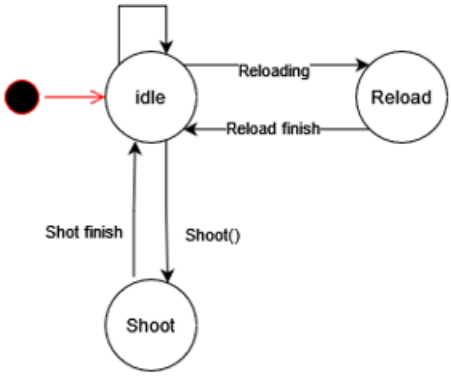
Tabel 4.3. Tabel desain fsm boss

Keadaan	**Kejadian**	**Tujuan**
Idle	mendeteksi target	Shoot Bullet

Idle	Mendeteksi target; Total serangan =3; Jarak dengan pemain <100	Shoot Laser
Idle	Mendeteksi target; Total serangan = 5	Shoot Rocket
Idle	mendeteksi target; health <50%	Shoot Multi Bullet
Shoot	Selesai melakukan serangan	Idle

d. Penerapan FSM pada objek senjata

Penerapan FSM pada objek senjata saat dimulainya program maka akan menuju *state idle*. Jika pemain melakukan *event input* untuk isi ulang amunisi senjata, maka state senjata berpindah ke *Reloading*, setelah selesai mengisi ulang akan langsung Kembali menuju state idle. Jika pemain melakukan *event input* untuk menembak, maka state akan berpindah ke *shoot*, lalu Kembali ke *state idle*.



Gambar 4.12. Desain FSM senjata

Tabel 4.4. Tabel FSM Senjata

Keadaan	Kejadian	Tujuan
Idle	Melakukan input isi peluru	Reload
Idle	Melakukan input tembak	Shoot
Reload	Selesai mengisi peluru	Idle
Shoot	Selesai menembak	Idle

5. Perancangan Asset game

Asset game adalah semua kebutuhan atau bahan- bahan yang akan digunakan dalam perancangan sebuah game itu sendiri. *Asset game* dapat berupa *sprite* 2D, 3D model dan animasi, gambar statis, ikon, teks, musik latar, efek suara dan sebagainya. Asset yang digunakan dalam proyek ini berasal dari situs http://itch.io.

a. Perancangan Karakter

Perancangan karakter merupakan pembahasan mengenai karakter yang terlibat dalam game Survivor Shooter.

Tabel 4.5. Tabel *asset* karakter permainan

No	Gambar	Keterangan
1		Karakter Utama/Player

2		Zombie
3		Bat
4		Crawltron
5		Mechguard
6		Boss

b. Perancangan Item dan Environment

Rancangan item dan Environment adalah pembahasan mengenai lingkungan dan item yang terdapat pada game Survivor Shooter.

Tabel 4.6. Tabel *asset* Item dan *Environment* permainan

No	Gambar	Keterangan

1		Koin
2		Amunisi
3		Senjata *Assault Rifle*
4		Senjata *Pistol*
5		Senjata *Shotgun*
6		medkit

C. Produksi

1. Perancangan State

State atau keadaan merupakan komponen utama dalam prinsip kerja FSM. Diperlukan *Abstrak Class* Agar memudahkan dalam mengimplementasikan berbagai *state* yang terdapat pada

permainan. *Abstrak Class* merupakan kelas yang terletak di posisi tertinggi dalam hierarki class. Berikut merupakan *script abstract class state*

```
extends Node
class_name Base_state

signal  Transitioned
@export var _state_name :String


func process_input(event: InputEvent):
        pass

func enter():
        pass


func exit():
        pass

func update(delta):
        pass

func physics_update(delta):
        pass

func get_state_name():
        return _state_name
```

2. Implementasi FSM pada Karakter

Proses implementasi FSM pada karakter dibuat dengan pendekatan *object oriented* yang mana *class Character State Machine* menjadi objek yang mengontrol *state* yang dimiliki karakter tersebut. Berikut *script* dari FSM karakter

```
extends Node
class_name Character_State_Machine

var states : Dictionary = {}
var current_state : State
@export var initial_state : State


# Called when the node enters the scene tree for the first time.
func init(parent, animation, movement, sprite):
        for child in get_children():
                if child is State:
                        states[child.name.to_lower()] = child
                        child.Transitioned
                                .connect( on_child_transition)
                        child.parent = parent
                        child.sprite = sprite
                        child.animations = animation
                        child.move_component = movement
                        child.SPEED = parent.SPEED
                        if initial_state:
                        initial_state.enter()
                        current_state = initial_state

func process(delta):
```

```
        if current_state:
                current_state.update(delta)

func  physics_process(delta):
        if current_state:
                current_state.physics_update(delta)

func _unhandled_input(_event: InputEvent):
pass

func on_child_transition(state, new_state_name):
        if state != current_state:
                return
        var new_state = states.get(new_state_name.to_lower())
        if !new_state:
                return
        if current_state:
                current_state.exit()
                current_state = new_state
                current_state.enter()
```

a. Implementasi State Idle

State Idle merupakan keadaan saat NPC dalam keadaan diam atau tidak sedang menjalankan suatu aksi.

```
extends State

var time_to_patrol : float = 2.0

func enter() -> void:
    super()
    time_to_patrol = randf_range(0.5, 2.5)
    parent.set_state(get_state_name())

func physics_update(delta):
    time_to_patrol -= delta
    if target_manager.is_detect_player() == false and time_to_patrol <= 0.0:
        Transitioned.emit(self,"patrol")
        return
    if get_movement_input() != Vector2.ZERO:
        if  target_manager.is_detect_player() == true and !move_component.distance_from_origin():
            Transitioned.emit(self,"chase")
            return
    if target_manager.can_shoot_target() == true:
        Transitioned.emit(self,"shoot")

    parent.velocity = parent.knockback * SPEED
    parent.move_and_slide()
    parent.knockback = lerp(parent.knockback, Vector2.ZERO, 0.2)
```

Gambar 4.13. State Idle

b. Implementasi State Chase

State Chase merupakan keadaan saat NPC mendeteksi pemain dan dalam jangkauan kejar.

```
extends State

func enter() -> void:
    super()
    move_component.timer.start()
    move_component.make_path()
    parent.set_state(get_state_name())

func physics_update(delta):
    if target_manager.can_shoot_target() == true:
        move_component.move_and_slide_navigation(Vector2.ZERO)
        Transitioned.emit(self,"shot")
        return
    if  move_component.check_navigation_finished() == true:
        Transitioned.emit(self,"idle")
    var movement_input = get_movement_input()
    if move_component.distance_from_origin() == true :
        Transitioned.emit(self,"retreat")

    move_component.move_and_slide_navigation((movement_input+parent.knockback )* SPEED)
    parent.knockback = lerp(parent.knockback, Vector2.ZERO, 0.2)

func exit():
    move_component.timer.stop()
```

Gambar 4.14. State Chase

c. Implementasi State Patrol

State Chase merupakan keadaan saat NPC tidak mendeteksi pemain dalam waktu yang ditentukan.

```gdscript
extends State

signal  make_random_target
var patrol_time  :float = 10
const MAX_PATROL_TIME : float = 10

func enter() -> void:
	patrol_time = MAX_PATROL_TIME
	super()
	move_component.generate_random_target()
	parent.set_state(get_state_name())

func physics_update(delta):
	patrol_time -= delta


	if patrol_time <= 0:
		Transitioned.emit(self,"idle")
	var movement_input = get_movement_input()
	if move_component.check_navigation_finished() == true:
		Transitioned.emit(self,"idle")
	if target_manager.is_detect_player() == true and !move_component.distance_from_origin():
		Transitioned.emit(self,"chase")


	move_component.move_and_slide_navigation((movement_input+parent.knockback )* SPEED)
	parent.knockback = lerp(parent.knockback, Vector2.ZERO, 0.2)
```

Gambar 4.15. State Patrol

d. Implementasi State Shoot

State Chase merupakan keadaan saat NPC jarak jauh mendeteksi pemain dan dalam jangkauan serangan.

```gdscript
extends State

@export var timer : Timer
@export var bullet_component : Bullet_Component
@export var pattern_component : Pattern_Component

func _ready():
	pass

func enter() -> void:
	super()
	timer.wait_time = parent.cd_shoot
	pattern_component.add_normal_attack()
	if pattern_component != null :
		if pattern_component.get_normal_attack() > pattern_component.special_attack:
			Transitioned.emit(self,"specialattack")
			return
	SoundManager.play_clip(parent.sound, SoundManager.SOUND_LASER)
	parent.set_state(get_state_name())
	move_component.direction_animation()
	bullet_component.shoot()
	timer.start()
	await timer.timeout
	Transitioned.emit(self,"chase")

func physics_update(delta):

	parent.velocity = parent.knockback * SPEED
	parent.move_and_slide()
	parent.knockback = lerp(parent.knockback, Vector2.ZERO, 0.2)
```

Gambar 4.16. State Shoot

e. Implementasi State Special Attack

State Chase merupakan keadaan saat NPC jarak jauh telah menyerang beberapa kali, maka akan melakukan serangan

spesial karakter tersebut.

```gdscript
extends State
@export var laser : Node2D
@export var pattern_component : Pattern_Component
@onready var interval = $interval
@export var laser_duration : float = 1

func _ready():
	pass

func enter() -> void:
	interval.wait_time = laser_duration
	super()
	pattern_component.reset_normal_attack()
	parent.set_state(get_state_name())
	move_component.direction_animation()
	var target = target_manager.get_target_location()
	laser.active(laser_duration, target)
	interval.start()
	await interval.timeout
	Transitioned.emit(self,"chase")

func physics_update(delta):

	parent.velocity = parent.knockback * SPEED
	parent.move_and_slide()
	parent.knockback = lerp(parent.knockback, Vector2.ZERO, 0.2)

func  exit():
	laser.deactive()

```

Gambar 4.17. State Special Attack

f. Implementasi State Die

State Die merupakan keadaan saat NPC memiliki *health point* <= 0 atau telah dikalahkan pemain.

```gdscript
extends State

var sound : AudioStreamPlayer2D

func _ready():
	if get_child_count() == 1:
		sound = get_child(0)

func enter() -> void:
	if sound  != null:
		sound.play()
	if parent != Player:
		if parent.is_dead != true:
			GameLoader.add_money(20)
			if parent.has_method("drop_item"):
				parent.drop_item()
			parent.is_dead = true
			SignalManager.emit_signal("enemy_killed", parent)
	super()


func physics_update(delta):

	pass

```

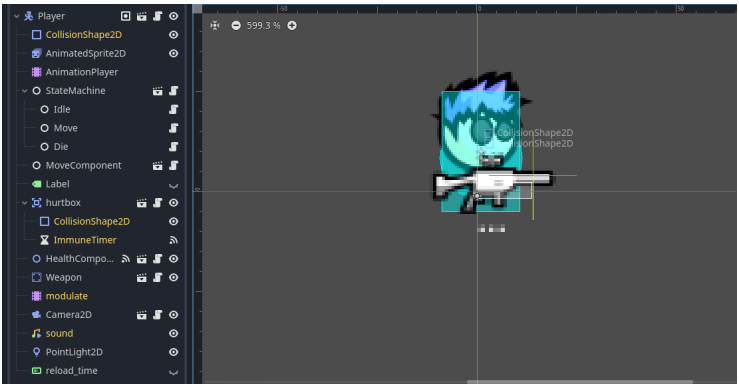
Gambar 4.18. State die

3. Implementasi Objek Karakter

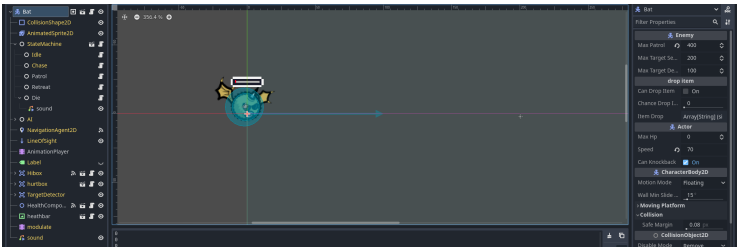
Setiap komponen pada karakter permainan disusun menggunakan *node*. Node merupakan blok penyusun terkecil pada *game* yang disusun hingga berbentuk seperti *tree*. Pada hirarkinya, *class character* memiliki komponen wajib diantaranya:

1. ***CharacterBody2D***, class yang dikhususkan dapat memproses tubuh fisik yang dapat dikontrol
2. ***AnimationSprite2D***, merupakan node yang digunakan untuk menampilkan *sprite* karakter
3. ***StateMachine***, node yang terdapat program FSM yang akan dieksekusi.
4. ***Hurtbox***, digunakan untuk mengatur *collision* area yang dapat menerima sinyal terluka atau diserang
5. ***HealthComponent***, *node* yang menampung dan memproses variabel *health* karakter
6. ***Sound***, berfungsi untuk menjalankan output suara terkait karakter

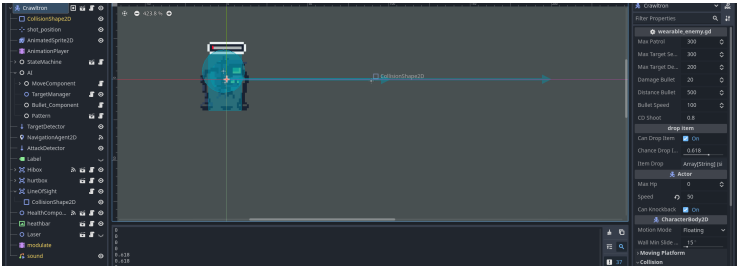
Berikut implementasi *scene* karakter pada permainan *Survivor Shooter*.



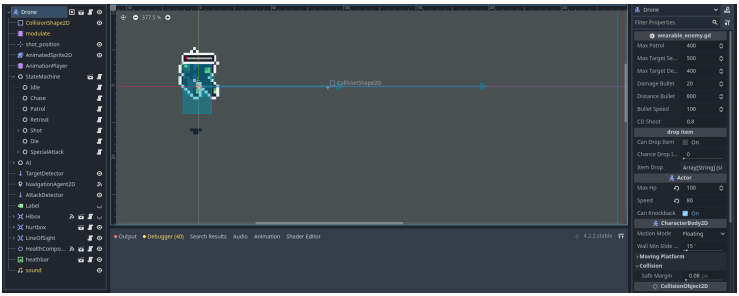
Gambar 4.19. Karakter player



Gambar 4.20. Karakter bat



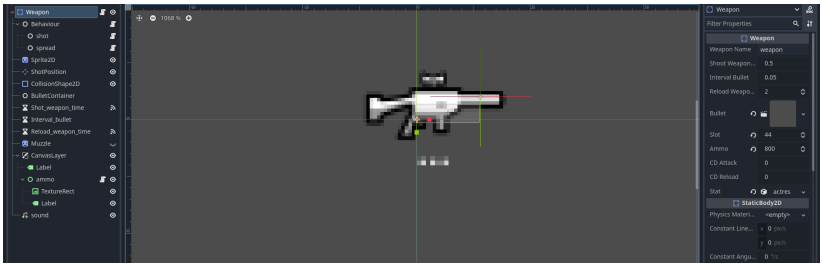
Gambar 4.21. Karakter Crawltron



Gambar 4.22. Karakter Drone

Gambar 4.23. Karakter Mechtron

Gambar 4.24. Karakter Boss



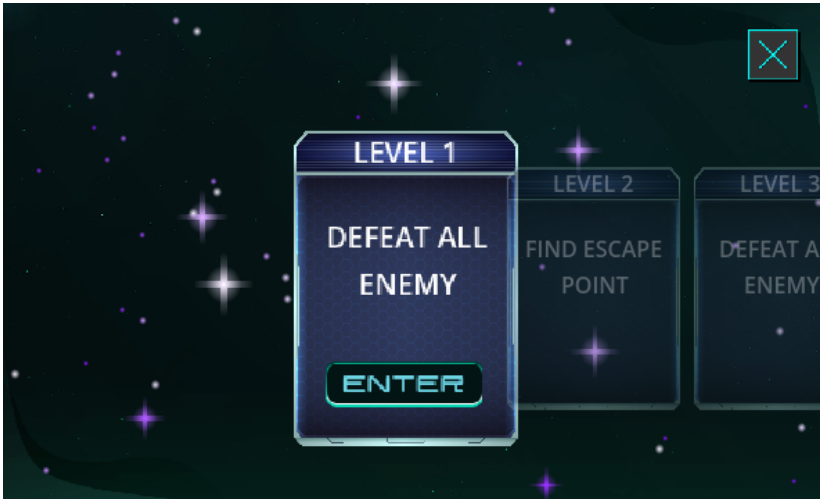
Gambar 4.25. Senjata

4. Implementasi Interface Permainan

Implementasi *interface* permainan meliputi tampilan menu pada *game Survivor Shooter*. Menu dalam game ini terdiri dari 6 menu, antara lain "Play", "Level List", "Workshop", "Option", "About" dan "Exit":

1. ***Play****, berfungsi untuk menampilkan scene level list*
2. **Level List**, Menampilkan daftar level yang tersedia
3. ***Workshop***, berisi *scene* yang menampilkan stastistik senjata, selain itu pemain dapat meningkatkan stastistik senjata menggunakan koin
4. ***Option***, *scene* yang mengatur konfigurasi permainan, yang diantaranya suara permainan, suara efek permainan.
5. ***About***, Menampilkan profil, syarat ketentuan permainan dan *credit* pengembang permainan.
6. ***Exit***, tombol untuk keluar dari permainan

Gambar 4.26. Menu Utama



Gambar 4.27. Menu Level List

Gambar 4.28. Menu Workshop

Gambar 4.29. Menu Option

Gambar 4.30. Menu About

D. Alpha Testing

1. Unit Testing

Proses pengujian unit dilakukan dengan metode blackbox, yaitu dengan melakukan percobaan beberapa kali terhadap FSM yang terdapat pada karater.

a. Pengujian Karakter Player

Pengujian *finite state machine* pada karakter *player* dilakukan dengan menguji input gerakan yang didapat dari *virtual controller*.

Tabel 4.7. Pengujian unit karakter player

Kode	State	Input	Output	Hasil
P001	*Idle*	Menggerakan karakter melalui virtual controller	Animasi = move; Pemain bergerak sesuai tujuan	Sesuai
P002	*Idle*	healthpoint kurang dari 0	Animasi player = fall; Pemain mati	Sesuai
P003	*Move*	Tidak melakukan input gerakan	Animasi = idle; Pemain diam ditempat	Sesuai
P004	*Move*	healthpoint kurang dari 0	Animasi player = fall; Pemain mati	Sesuai

b. Pengujian Karakter Musuh Jarak Dekat

Pengujian *finite state machine* pada karakter musuh jarak dekat dilakukan dengan menguji kemungkinan yang terjadi sesuai dengan lampiran 1.

Tabel 4.8. Pengujian unit karakter musuh jarak dekat

Kode	State	Input	Output	Hasil
EM01	*Idle*	Saat countdown patrol = 0 dan tidak mendeteksi player	Animasi = walk; state = patrol; karakter bergerak menuju titik patroli	Sesuai
EM02	*Idle / patrol*	Mendeteksi pemain	Animasi = walk; state = chase; karakter bergerak menuju target	Sesuai

EM03	*patrol*	Mencapai titik patroli	Animasi = Idle; state = idle; karakter diam ditempat	Sesuai
EM04	*Chase*	Kehilangan target	Animasi = Idle; state = idle; karakter diam ditempat	Sesuai
EM5	*idle* / *patrol* / *chase*	healthpoint kurang dari 0	Animasi = fall; state = dead; karakter mati	Sesuai

c. Pengujian Karakter Musuh Jarak Jauh

Pengujian *finite state machine* pada karakter musuh jarak jauh dilakukan dengan menguji kemungkinan yang terjadi sesuai dengan lampiran 1.

Tabel 4.9. Pengujian unit karakter musuh jarak jauh

Kode	**State**	**Input**	**Output**	**Hasil**
ER01	*Idle*	Saat countdown patrol = 0 dan tidak mendeteksi player	Animasi = walk; state = patrol; karakter bergerak menuju titik patroli	Sesuai
ER02	*Idle* / *patrol*	Mendeteksi pemain	Animasi = walk; state = chase; karakter bergerak menuju target	Sesuai
ER03	*patrol*	Mencapai titik patroli	Animasi = Idle; state = idle; karakter diam ditempat	Sesuai
ER04	*Chase*	Kehilangan target	Animasi = Idle; state = idle; karakter diam ditempat	Sesuai

ER05	*Chase*	Berada pada jarak tembak	Animasi = shoot; state = shot; karakter menembak kearah target	Sesuai
ER06	*Shoot*	Telah menyerang sebanyak X kali	Animasi = shoot; state = special attack; karakter menembak kearah target	Sesuai
ER07	*Shoot*	Target berada diluar jarak tembak	Animasi = walk; state = chase; karakter bergerak menuju target	Sesuai
ER08	*Special Attack*	Telah melakukan serangan	Animasi = walk; state = chase; karakter bergerak menuju target	Sesuai
ER09	*idle / patrol / chase*	healthpoint kurang dari 0	Animasi = fall; state = dead; karakter mati	Sesuai

d. Pengujian Senjata

Pengujian *finite state machine* pada senjata dilakukan dengan menguji kemungkinan yang terjadi sesuai dengan lampiran 1.

Tabel 4.10. Pengujian unit senjata

Kode	State	Input	Output	Hasil
S01	*Idle*	Pemain membidik melalui virtual controller	State = Shoot; Senjata menembak kearah tujuan	Sesuai

S02	*Idle*	Pemain melakukan input reload senjata	State = Reload; Peluru disenjata terisi ulang, tidak bisa melakukan input menembak dalam keadaan ini	Sesuai
S03	*Shoot*	Pemain berhenti melakukan input menembak	State = idle;	Sesuai
S04	*Reload*	waktu countdown isi ulang selesai	State = Idle;	Sesuai

Hasil dari semua pengujian unit dilakukan kemudian dihitung rata-rata sebagai berikut:

Tabel 4.11. Tabel Skor Ketercapaian

Objek	**Ketercapaian**	**Total Item**
Player	4	4
Musuh Jarak Dekat	5	5
Musuh Jarak Jauh	9	9
Senjata	4	4
Total	22	22

Berdasarkan data tabel 4.11 tersebut, kemudian dihitung skor presentase ketercapaian:

$$\frac{22}{22} \times 100\% = 100 \quad (4.1)$$

Hasil dari pengujian *Finite State Machine* yang diterapkan pada objek berjalan dengan baik dan sesuai dengan tingkat presentase 100%

2. Integrated Testing

Tabel 4.12. tabel pengujian integrasi

Pengujian	**Skenario**	**Output**	Hasil
Membuka menu utama	Membuka permainan	Menampilkan scene boot, dilanjutkan ke scene menu utama	Sesuai
Stage list	Pada halaman utama, menekan tombol jelajah	Menampilkan scene stage list	Sesuai
Workshop	Pada halaman utama, menekan tombol workshop	Menampilkan Scene Workshop	Sesuai
Upgrade Senjata	Pada halaman workshop, melakukan upgrade senjata (koin mencukupi)	Statistik senjata terpilih akan meningkat, koin berkurang	Sesuai
	Pada halaman workshop, melakukan upgrade senjata (koin tidak mencukupi)	Tidak bisa melakukan upgrade senjata	Sesuai
	Pada halaman workshop, senjata ditingkatkan level maksimal	Button upgrade senjata menjadi disable	Sesuai

Option	Pada halaman utama, menekan tombol option kemudian melakukan perubahan dan disimpan	Menampilkan stage permainan. setelah menyimpan konfigurasi, perubahan diterapkan	Sesuai
Halaman about	Pada halaman utama, menekan tombol about	Menampilkan scene profil permainan dan pengembang	Sesuai
Memilih stage	Pada halaman stage permaian, memilih salah satu stage yang telah terbuka	Menampilkan stage permainan sesuai yang dipilih	Sesuai
Virtual joystick	Menggerakan virtual joytick yang terdapat pada permainan	Virtual joystick bergerak sesuai letak sentuhan	Sesuai

Berdasarkan data tabel 4.12 tersebut, kemudian dihitung skor presentase ketercapaian:

$$\frac{10}{10} \times 100\% = 100\% \tag{4.2}$$

Hasil dari pengujian integrasi yang dilakukan pada *interface* dan menu permainan berjalan dengan baik dan sesuai dengan tingkat prosentase 100%

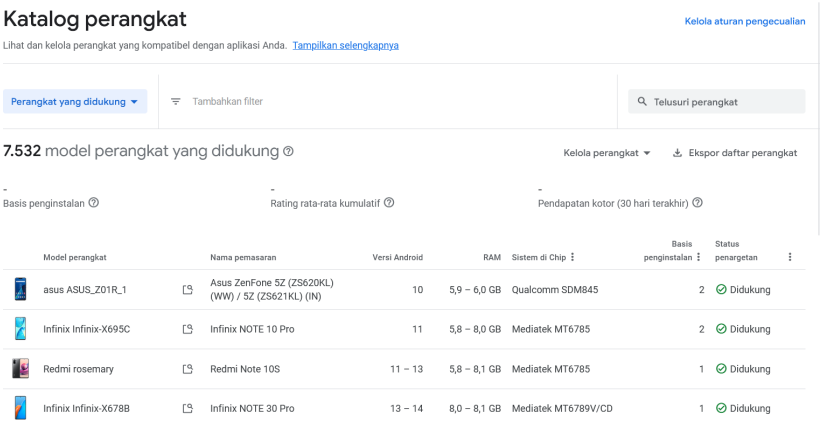
3. System Testing

Pengujian sistem dilakukan dengan melakukan instalasi pada beberapa perangkat android yang berbeda. Berikut merupakan hasil pengujian instalasi *game Survivor Shooter* :

Tabel 4.13. Pengujian Sistem

Perangkat	Versi OS	Resolusi	Hasil
Redmi Note 9	Android 13	2400 x 1080 px	Berhasil
Samsung galaxy A51	Android 13	2000 x 1080 px	Berhasil
Samsung galaxy A52s	Android 13	2400 x 1080 px	Berhasil
Itel p55	Android 13	1612 x 720 px	Berhasil
Redmi note 8	Android 12	2340 x 1080 px	Berhasil
Asus ZenFone 5Z	Android 10	2240 x 1080 px	Berhasil

Selain pengujian pada perangkat diatas, penguji mengamhil data katalog perangkat yang terdapat pada *google play console*. Pada gambar 4.31 didapatkan data bahwa 7532 perangkat dapat menjalankan *game Survivor Shooter*.



Gambar 4.31. Katalog Perangkat

E. Beta Test

Pengujian beta dilakukan dengan melakukan kuisioner sejumlah 15 responden. Hasil dari pernyataan responden dapat dilihat pada tabel berikut:

Tabel 4.14. Hasil Kuisioner UAT

KODE	TOTAL NILAI	RERATA
G01	57	3,8
G02	65	4,3
G03	62	4,1
G04	65	4,3
G05	60	4
F01	56	3,7
F02	57	3,8
F03	59	3,9
F04	62	4,1
F05	62	4,13
F06	63	4,2
F07	67	4,5
Total	735	

Berdasarkan data tabel 4.12 tersebut, kemudian dihitung skor presentase ketercapaian:

$$\frac{735}{15} \times 100\% = 61.25\% \tag{4.3}$$

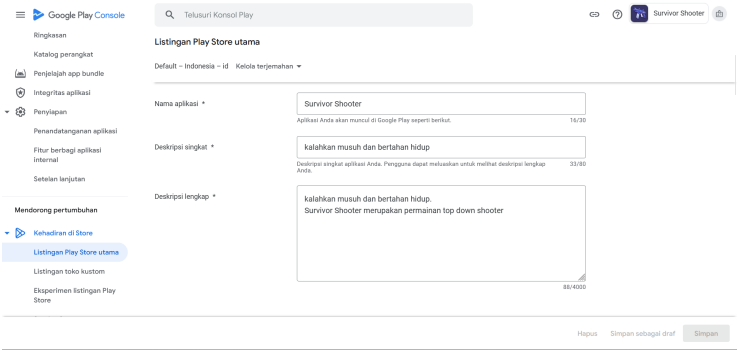
Hasil dari pengujian user yang dilakukan melalui kuisioner memperoleh hasil **baik** dengan tingkat presentase 61.25% .

F. Rilis

Tahap rilis dilakukan dengan mengunggah permainan ke *play store*. Supaya permainan dapat diverifikasi oleh pihak *google*, diperlukan data informasi mengenai *game Survivor Shooter* ini. Berikut informasi profil permainan:

Tabel 4.15. Tabel data aplikasi

Tipe aplikasi	Game
Nama aplikasi	Survivor Shooter
Pengembang	Supersemar dev
Bundle ID	Com.supersemar.survivor_shooter
Platform	Android
Paid App?	Gratis (*free*)
Min SDK	24
Target SDK	34
Icon	



Gambar 4.32. Profil Aplikasi

Berikut merupakan status setelah *game Survivor Shooter* telah diverifikasi oleh pihak *google*, *game* dapat dicari dan diunduh melalui *play store* perangkat android dengan versi OS android 8.0 (API 24) hingga android 14 (API 34).

Gambar 4.33. *Listing Play Store*

BAB V
KESIMPULAN DAN SARAN

A. Kesimpulan

Berdasarkan hasil penelitian yang telah dilakukan dalam implementasi *finite state machine* pada NPC *game topdown shooter* menggunakan Godot engine diperoleh kesimpulan yaitu:

1. Penelitian ini berhasil menerapkan *finite state machine* pada objek karakter *player* dan musuh. Hasil dari pengembangan ini telah sesuai seperti musuh dapat bergerak dengan otomatis dan sesuai dengan kecerdasan buatan yang diterapkan. Serta gerakan player sesuai inputan dari pemain sesuai.
2. Pengujian aplikasi dilakukan melalui dua tahap. Tahap internal dilakukan dengan menguji fungsionalitas didapatkan hasil 100%, serta pengujian instalasi perangkat dengan hasil 7532 perangkat yang didukung. Pada tahap eksternal dilakukan dengan melakukan kuisioner diperoleh hasil **"baik"** dengan tingkat presentase 61.25%

B. Saran

Setelah dilakukan pengujian terhadap game "Survivor Shooter" didapatkan ada kekurangan. Adapun saran sebagai acuan terhadap penelitian atau pengembangan selanjutnya diantaranya:

1. Level *game* dapat dikembangkan menjadi lebih banyak.

2. Perlu ditambahkan fitur *tutorial* serta penjelasan pada permainan.
3. Kecerdasan pada karakter musuh dapat ditingkatkan kompleksitasnya agar dapat memberi tantangan lebih pada permainan.
4. *Game* perlu memiliki alur cerita.

DAFTAR PUSTAKA

Al Tahtawi, A. R., Somantri, Y., and Haritman, E. (2017). Design and Implementation of PID Control-based FSM Algorithm on Line Following Robot. *Jurnal Teknologi Rekayasa*, 1(1):23.

Andersen, J., Pragantha, J., and Haris, D. A. (2021). PERANCANGAN GAME TOP DOWN ROGUELIKE SHOOTER "ARCANA MEMORIES" PADA PC. *Jurnal Ilmu Komputer dan Sistem Informasi*, 9(1):135.

Andriyat Krisdiawan, R. (2018). IMPLEMENTASI MODEL PENGEMBANGAN SISTEM GDLC DAN ALGORITMA LINEAR CONGRUENTIAL GENERATOR PADA GAME PUZZLE. *JURNAL NUANSA INFORMATIKA*, 12:9.

Ariyana, R. Y., Susanti, E., Ath-Thaariq, M. R., and Apriadi, R. (2022). Penerapan Metode Game Devlopment Life Cycle (GDLC) pada Pengembangan Game Motif Batik Khas Yogyakarta. 1(6).

Baffa, A., Sampaio, P., Feijo, B., and Lana, M. (2017). Dealing with the Emotions of Non Player Characters. In *2017 16th Brazilian Symposium on Computer Games and Digital Entertainment (SBGames)*, pages 76–87, Curitiba. IEEE.

Bimantoro, T. and Haryanto, H. (2016). Pemodelan Perilaku Musuh Menggunakan Finite State Machine (FSM) Pada Game Pengenalan Unsur Kimia Enemy Behaviour Modeling Using Finite State Machine (FSM) on Game of Chemical

Element Introduction. *Journal of Applied Intelligent System*, 1(3):210–219.

Cholifah, W. N., Yulianingsih, Y., and Sagita, S. M. (2018). Pengujian Black Box Testing pada Aplikasi Action & Strategy Berbasis Android dengan Teknologi Phonegap. *STRING (Satuan Tulisan Riset dan Inovasi Teknologi)*, 3(2):206.

CNN Indonesia, C. I. (2023). Cuan Industri Game Tembus Rp25 T, Tapi ke Kantong RI Hanya 0,5 Persen. https://bit.ly/3UoEtE9.

Enggar, A., Daru Kusuma, P., and Ratna Astuti (2022). Implementasi Finite State Machine Untuk Npc Pada Game 2d Side-Scroll Shooter.

Fauzhan, F., Alvian, E., Andreanto, S., and Amzy, N. (2020). Analisis Elemen Visual Game "Pamali" Dengan Menggunakan Pendekatan Teori Mimesis Plato. *Visual Heritage: Jurnal Kreasi Seni dan Budaya*, 2(02):89–95.

Gunadi, A. and Fatta, H. A. (2012). ANALISIS DAN PEMBUATAN GAME " PETUALANGAN SI ARGO" BERBASIS. 13(1).

Hady, E. L., Haryono, K., and Rahayu, N. W. (2020). User Acceptance Testing (UAT) pada Purwarupa Sistem Tabungan Santri (Studi Kasus: Pondok Pesantren Al-Mawaddah).

Hasibuan, A. N. and Dirgahayu, T. (2021). Pengujian dengan Unit Testing dan Test case pada Proyek Pengembangan Modul Manajemen Pengguna.

Herlambang, A. D., Rachmadi, A., Utami, K., Hakim, R. I., and Rohmah, N. (2019). Pengembangan Fitur E-Matur dengan V-Model sebagai Alat Pengaduan Publik untuk Website Badan Kepegawaian Negara. *Jurnal Teknologi Informasi dan Ilmu Komputer*, 6(5):467–474.

Hoesen, N. (2022). Rancang Bangun Game Berbasis Android Bertemakan Cerita Rakyat Betawi Si Pitung. *Jurnal Esensi Infokom : Jurnal Esensi Sistem Informasi dan Sistem Komputer*, 5(2):32–37.

Khoirunningsih, E. (2018). PENGEMBANGAN SISTEM INFORMASI KEARSIPAN BERBASIS MOBILE APPLICATION DI DINAS KOMUNIKASI DAN INFORMATIKA DAERAH ISTIMEWA YOGYAKARTA. *Jurnal Elektronik Pendidikan Teknik Informatika*.

Laksono, E. A. and Susanto, A. (2020). Mathematics Education Game Using the Finite State Machine Method to Implement Virtual Reality in Game Platformer. *Inform : Jurnal Ilmiah Bidang Teknologi Informasi dan Komunikasi*, 5(1):8–13.

Luay, D. M. and Apriandari, W. (2024). PENGGUNAAN METODE GDLC (GAME DEVELOPMENT LIFE CYCLE) UNTUK MENGENAL BENDERA DUNIA. 10(1).

Martono, K. T. (2015). PENGEMBANGAN GAME DENGAN MENGGUNAKAN GAME ENGINE GAME MAKER. *JURNAL SISTEM KOMPUTER*.

Mulachela, A. (2020). Analisis Perkembangan Industri Game di Indonesia Melalui Pendekatan Rantai Nilai Global (Global

Value Chain). *Indonesian Journal of Global Discourse*, 2(2):32–51.

Nugroho, A. K. and Wijayanto, B. (2022). EVALUATION OF THE QUALITY OF ACADEMIC INFORMATION SYSTEM UNSOED USING ISO 9126 AND MEAN OPINION SCORE (MOS).

Pressman, R. S. (2001). *Software Engineering: A Practitioner's Approach*. McGraw-Hill Series in Computer Science Software Engineering and Databases. McGraw Hill, Boston, Mass, 5. ed., 20. anniversary ed edition.

Ramadan, R. and Widyani, Y. (2013). Game development life cycle guidelines. In *2013 International Conference on Advanced Computer Science and Information Systems (ICACSIS)*, pages 95–100, Sanur Bali, Indonesia. IEEE.

Refnaldi, I. A. A. (2023). Design an Enemy Non-Player Character in Maze Game Using Finite State Machine Algorithm. *[CEPAT] Journal of Computer Engineering: Progress, Application and Technology*, 2(01):9.

Rifai, W. A. (2015). PENGEMBANGAN GAME EDUKASI LINGKUNGAN BERBASIS ANDROID.

Safitra, W., Faisol, A., and Wibowo, S. A. (2020). PENERAPAN METODE FINITE STATE MACHINE PADA NON PLAYER CHARACTER (NPC) GAME ACTION STRATEGY "OUROBOROS". *Jurnal Mahasiswa Teknik Informatika*, 4(2).

Satrio, I., Santi Wahyuni, F., and Rudhistiar, D. (2022). PENERAPAN A* PATHFINDING DAN FSM (FINITE STATE

MACHINE) PADA GAME ”LOST CIVILIZATION” BERBASIS ANDROID. *Jurnal Mahasiswa Teknik Informatika)*, 6(2).

Setio Utomo, D. and Arwin Dermawan, D. (2022). Implementasi Finite State Machine (FSM) Dalam Game Monopoli 3D Teknologi Informasi dan Komunikasi dengan Algoritma Fisher-Yates Shuffle Berbasis Android.

Sifaulloh, H., Fadila, J. N., and Nugroho, F. (2021). Penerapan Metode Finite State Machine pada Game Santri on the Road. *Walisongo Journal of Information Technology*, 3(1):11–18.

Situs Godot Engine, S. G. E. (2024). Situs resmi Godot Engine. https://godotengine.org/features/.

State of Mobile Gaming 2023, S. o. M. G. . (2023). State of Mobile Gaming 2023. https://www.data.ai/en/go/state-of-mobile-gaming-2023/.

Wibisono, W. and Baskoro, F. (2002). PENGUJIAN PERANGKAT LUNAK DENGAN MENGGUNAKAN MODEL BEHAVIOUR UML. *JUTI: Jurnal Ilmiah Teknologi Informasi*, 1(1):43.

Wijaya, Y. D. and Astuti, M. W. (2021). PENGUJIAN BLACKBOX SISTEM INFORMASI PENILAIAN KINERJA KARYAWAN PT INKA (PERSERO) BERBASIS EQUIVALENCE PARTITIONS. *Jurnal Digital Teknologi Informasi*, 4(1):22.

Winarno, E., Cicik Wijayanti, T., Hadikurniawati, W., and Solissa, E. M. (2022). Educational Game Based Role Playing Games with Finite State Machine Method. *International Journal Of Artificial Intelegence Research*, 6(1):2579–7298.

Yulsilviana, E. and Ekawati, H. (2019). PENERAPAN METODE FINITE STATE MACHINE (FSM) PADA GAME AGENT LEGENDA ANAK BORNEO. *Sebatik*, 23(1):116–123.

Zagal, J. P. and Altizer, R. (2014). Examining 'RPG Elements': Systems of Character Progression.

Lampiran 1. Daftar test case unit testing

Tabel 0.1. Tabel daftar test case unit testing

Objek	**Kode**	**State**	**Input**	**Hasil yang diharapkan**
Player	P01	*Idle*	Pemain melakukan input gerakan	Animasi = move; Pemain bergerak sesuai tujuan
	P02	*Idle*	Health pemain kurang dari sama dengan (<=) 0	Animasi = fall; Pemain mati
	P03	*Move*	Tidak menerima input gerakan	Animasi = idle; Pemain diam ditempat
	P04	*Move*	Health pemain kurang dari sama dengan (<=) 0	Animasi = fall; Pemain mati
senjata	S01	*Idle*	Pemain melakukan input menembak	State = shoot; Senjata menembak kearah yang ditentukan
	S02	*Idle*	Pemain melakukan input mengisi ulang peluru	State = reload; Peluru disenjata terisi ulang, tidak bisa melakukan input menebak selama proses ini
	S03	*Shoot*	Pemain berhenti melakukan input menembak	State = idle

	S04	*Reload*	Waktu countdown mengisi ulang selesai	State = idle
Musuh jarak dekat	EM01	*Idle*	Countdown patrol = 0, tidak mendeteksi pemain	Animasi = walk; State = patrol; Karakter bergerak menuju titik acak
	EM02	*Idle / patrol*	Mendeteksi pemain atau target	Animasi = walk; State = chase Karakter bergerak menuju pemain
	EM03	*Patrol*	Mencapai titik patroli	Animasi = idle; State = idle Karakter diam ditempat
	EM04	*Chase*	Kehilangan target	Animasi = idle; State = idle Karakter diam ditempat
	EM05	*Idle/ patrol/ chase*	Health karakter kurang dari sama dengan (<=) 0	Animasi player = fall; Pemain mati
Musuh jarak jauh	ER01	*Idle*	Countdown patrol = 0, tidak mendeteksi pemain	Animasi = walk; State = patrol; Karakter bergerak menuju titik acak
	ER02	*Idle / patrol*	Mendeteksi pemain atau target	Animasi = walk; State = chase; Karakter bergerak menuju pemain
	ER03	*Patrol*	Mencapai titik patroli	Animasi = idle; State = idle Karakter diam ditempat

	ER04	*Chase*	Kehilangan target	Animasi = idle; State = idle Karakter diam ditempat
	ER05	*Chase*	Berada pada jarak tembak	Animasi = shoot; State = shoot Karakter menembak kearah pemain
	ER06	*Shoot*	Target berada di jarak tembak; Telah Menyerang sebanyak x kali	Animasi = shoot; State = special_attack Karakter menembak kearah pemain
	ER07	*Shoot*	Target berada diluar jarak tembak;	Animasi = walk; State = chase Karakter bergerak menuju pemain
	ER08	*Special attack*	Telah melakukan serangan	Animasi = walk; State = chase Karakter bergerak menuju pemain
	ER09	*Idle / patrol /chase / shoot / special-attack*	Health karakter kurang dari sama dengan (<=) 0	Animasi = fall; karakter mati

Lampiran 2. Riwayat Hidup

RIWAYAT HIDUP

A. Identitas Diri

1	Nama Lengkap	:	Andri Aditya
2	Tempat, Tgl. Lahir	:	Kab. Semarang, 11 Maret 2002
3	Alamat	:	Rekesan, RT 05 RW 02, Beji Lor, Kec. Suruh, Kab. Semarang
4	HP	:	6289509401174
5	Email	:	andriaditya311@gmail.com

B. Riwayat Pendidikan

1. Pendidikan Formal

a. SD N 1 Beji Lor

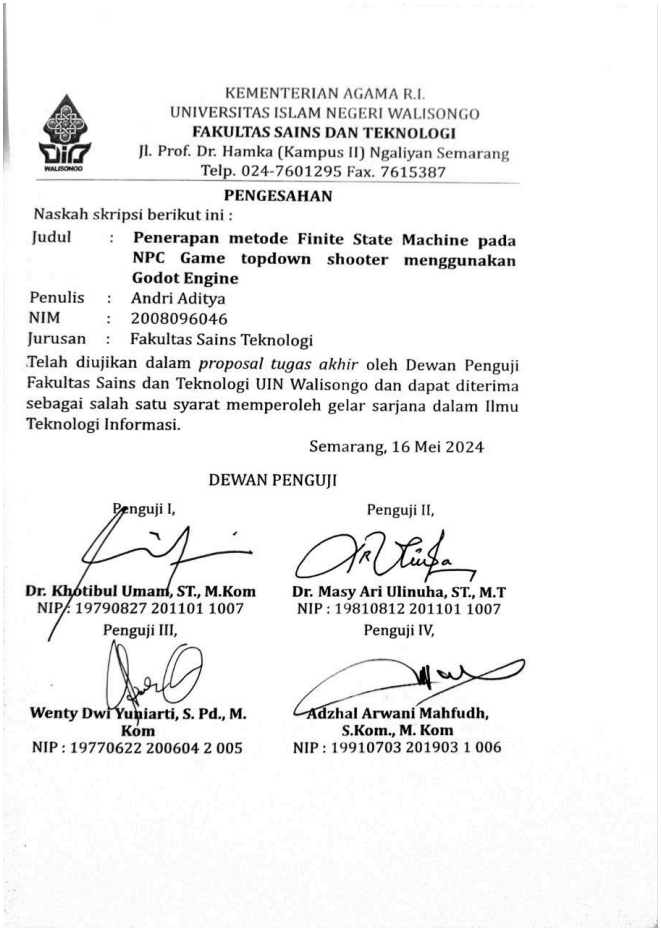
b. SMP N 2 Tengaran

c. SMK N 1 Tengaran

Semarang, 22 Juni 2024
Pembuat pernyataan,

Andri Aditya
NIM : 2008096046

Lampiran 3. Lembar Pengesahan Proposal



KEMENTERIAN AGAMA R.I.
UNIVERSITAS ISLAM NEGERI WALISONGO
FAKULTAS SAINS DAN TEKNOLOGI
Jl. Prof. Dr. Hamka (Kampus II) Ngaliyan Semarang
Telp. 024-7601295 Fax. 7615387

PENGESAHAN

Naskah skripsi berikut ini :

Judul : **Penerapan metode Finite State Machine pada NPC Game topdown shooter menggunakan Godot Engine**
Penulis : Andri Aditya
NIM : 2008096046
Jurusan : Fakultas Sains Teknologi

Telah diujikan dalam *proposal tugas akhir* oleh Dewan Penguji Fakultas Sains dan Teknologi UIN Walisongo dan dapat diterima sebagai salah satu syarat memperoleh gelar sarjana dalam Ilmu Teknologi Informasi.

Semarang, 16 Mei 2024

DEWAN PENGUJI

Penguji I,
Dr. Khotibul Umam, ST., M.Kom
NIP : 19790827 201101 1007

Penguji II,
Dr. Masy Ari Ulinuha, ST., M.T
NIP : 19810812 201101 1007

Penguji III,
Wenty Dwi Yuniarti, S. Pd., M. Kom
NIP : 19770622 200604 2 005

Penguji IV,
Adzhal Arwani Mahfudh, S.Kom., M. Kom
NIP : 19910703 201903 1 006

Gambar 0.1. *Lembar Pengesahan Proposal*