JITET (Jurnal Informatika dan Teknik Elektro Terapan)
Vol. 13 No. 3, pISSN: 2303-0577 eISSN: 2830-7062 http://dx.doi.org/10.23960/jitet.v13i3.7163

IMPLEMENTASI SISTEM CHARACTER PLAYER PADA GAME RPG 2D MENGGUNAKAN GAME ENGINE GODOT

Arifqi Kartadinata[1], Mutaqin Akbar[2]

[1,2]Universitas Mercu Buana Yogyakarta; Jl. Jembatan Merah No. 84C, Sleman, Yogyakarta, Indonesia; (0274) 584922

Keywords:
Character Player;
Game RPG 2D;
Godot Engine;
Character Controller;
Combat System.

Corespondent Email:
211110119@student.mercubuana-yogya.ac.id

Abstrak. *Perkembangan industri game mendorong kebutuhan akan sistem kendali karakter dan pertarungan yang responsif dalam genre Action RPG 2D. Penelitian ini bertujuan untuk mengimplementasikan sistem character player yang mencakup character controller dan combat system menggunakan Godot Engine. Character controller dirancang untuk memberikan kendali yang halus terhadap pergerakan karakter seperti bergerak, berlari, dan melompat, sedangkan combat system difokuskan pada mekanisme attack, deteksi musuh, upgrade stats, dan pengurangan damage musuh. Metode yang digunakan adalah rekayasa perangkat lunak dengan tahapan analisis, perancangan, implementasi, dan pengujian. Sistem kendali karakter dikembangkan menggunakan Hierarchical Finite State Machine (HFSM) agar modular dan mudah dikembangkan, sedangkan sistem pertarungan memanfaatkan area dan animasi yang sinkron. Hasil pengujian menunjukkan bahwa sistem berjalan dengan baik dan memberikan pengalaman bermain yang responsif dan stabil. Implementasi ini penting sebagai acuan pengembangan game Action RPG 2D berbasis open-source.*

Abstract. *The growth of the game industry drives the demand for responsive character control and combat systems, particularly in the 2D Action RPG genre. This study aims to implement a character player system that includes a character controller and a combat system using the Godot Engine. The character controller is designed to provide smooth control over character movements such as walking, running, and jumping, while the combat system focuses on attack mechanisms, enemy detection, stat upgrades, and enemy damage reduction. The research method employed is software engineering, consisting of analysis, design, implementation, and testing stages. The character control system is developed using a Hierarchical Finite State Machine (HFSM) to ensure modularity and ease of development, while the combat system utilizes synchronized areas and animations. Testing results indicate that the system performs well and provides a responsive and stable gameplay experience. This implementation serves as a valuable reference for developing open-source 2D Action RPG games.*

1. PENDAHULUAN

Game merupakan suatu sistem interaktif yang melibatkan pemain dalam suatu tantangan dengan aturan tertentu dan menghasilkan pencapaian yang dapat diukur. Game memiliki peran penting sebagai media hiburan, pembelajaran, dan pelatihan yang dapat dinikmati oleh semua kalangan, baik anak-anak maupun orang dewasa[1]. Untuk mewujudkan pembuatan game, pengembang membutuhkan

alat bantu yang dapat mendukung proses pengembangannya. Salah satu alat yang umum digunakan dalam proses ini adalah game engine[2]. Dalam pengembangan game indie, keterbatasan teknologi dan sumber daya kerap menjadi tantangan dalam menciptakan sebuah game yang dapat menyaingi game populer[3]. Maka dari itu peran game engine yaitu sebagai perangkat lunak utama yang digunakan untuk merancang dan membangun video game yang dapat memudahkan pengembang dalam pembuatannya. Umumnya, pemula dalam pengembangan game memanfaatkan game engine karena dapat mempermudah proses pembuatan game. Namun, bagi pengembang (developer) yang telah memiliki kemampuan teknis yang tinggi, mereka dapat menciptakan game engine mereka sendiri. Video game tidak hanya berfungsi sebagai media hiburan semata, tetapi juga dapat digunakan untuk melatih keterampilan berpikir dan mempelajari berbagai kompetensi secara interaktif[4], [5]. Game 2D dalam genre ini tetap populer berkat kemudahan pengembangan, kecepatan akses, serta daya tarik visual yang khas. Salah satu tools yang mendukung pengembangan game 2D secara efisien adalah Godot Engine, game engine open-source yang ringan, fleksibel, dan mendukung pengembangan multiplatform. IDE ini dilengkapi dengan berbagai pustaka yang diperlukan serta add-on tambahan untuk menunjang kompatibilitas dengan lingkungan perangkat lunak lainnya. Platform ini juga menyediakan berbagai alat bantu yang mendukung proses pengembangan game secara open-source. Bahasa pemrograman utama yang digunakan dalam IDE ini dikenal dengan nama GDScript[6].

Game bisa dibuat baik oleh individu maupun studio, namun yang paling penting adalah bagaimana pemain berinteraksi dengan isi game dan bagaimana interaksi tersebut memengaruhi berbagai elemen dalam game[7]. Game kini tidak hanya menjadi hiburan, tetapi juga bisa menjadi profesi yang menguntungkan. Selain itu, game dapat menyampaikan nilai-nilai seperti ketekunan, melatih logika, dan mengasah kreativitas[8].

Untuk menyederhanakan proses pengembangan, para developer masa kini semakin banyak memanfaatkan game engine, karena menyediakan kerangka kerja yang mencakup berbagai fungsi dan alat bantu dalam menciptakan interaksi, logika permainan, serta manajemen sumber daya secara efisien. Godot, misalnya, menawarkan bahasa pemrograman bawaan GDScript, serta dukungan untuk bahasa lain seperti C# dan C++, sehingga memungkinkan fleksibilitas tinggi dalam pengembangan game 2D maupun 3D.

Penelitian sebelumnya dalam pengembangan game 2D action RPG telah banyak membahas aspek desain level, sistem quest, hingga integrasi AI musuh. Namun, aspek fundamental yang secara langsung memengaruhi pengalaman bermain, seperti character controller yang mengatur pergerakan karakter, lompatan, lari, dan navigasi dalam lingkungan permainan serta combat system yang meliputi mekanisme serangan, pertahanan, animasi, dan interaksi dengan musuh masih sering diterapkan secara terbatas atau tidak terdokumentasikan secara mendalam. Beberapa pengembangan bahkan hanya menampilkan sistem dasar tanpa mempertimbangkan fleksibilitas dan skalabilitas kode untuk pengembangan lebih lanjut. Padahal, game tidak hanya berfungsi sebagai sarana hiburan, tetapi juga sebagai media untuk menguji kelincahan intelektual, ketangkasan, dan kecerdasan berpikir pemain.

Kesenjangan yang ditemukan dari tinjauan tersebut adalah masih minimnya dokumentasi teknis yang secara khusus mengimplementasikan sistem character controller dan combat system dalam kerangka kerja Godot Engine untuk game 2D action RPG secara menyeluruh. Selain itu, belum banyak referensi yang membahas penerapan kedua sistem tersebut secara modular dan efisien, yang dapat menunjang pengembangan game secara kompleks dan berkelanjutan. Keunikan dari penelitian ini terletak pada implementasi sistem pemain (character player) yang dirancang dengan memperhatikan prinsip modularitas serta integrasi langsung dengan fitur-fitur Godot Engine, seperti sistem animasi, input, dan state machine.

Berdasarkan latar belakang dan kesenjangan tersebut, penelitian ini bertujuan untuk merancang dan mengimplementasikan sistem character player pada game 2D action RPG, dengan fokus utama pada pengembangan character controller dan combat system menggunakan Godot Engine. Tujuan utamanya adalah menciptakan sistem kendali karakter dan

sistem pertarungan yang responsif serta modular dalam konteks pengembangan game 2D action RPG berbasis Godot Engine.

2. TINJAUAN PUSTAKA

2.1 Game

Game adalah suatu aktivitas yang disusun secara terstruktur maupun semi terstruktur yang pada awalnya ditujukan sebagai hiburan. Namun, seiring waktu, game juga berkembang menjadi alat pembelajaran yang efektif. Keunggulan ini berasal dari sifat game yang menarik dan menyenangkan, mampu memotivasi pemain, memberikan efek kecanduan yang bersifat positif dalam konteks edukatif, serta mendorong terjadinya interaksi dan kerja sama antar pemain. Game sebenarnya penting untuk perkembangan otak, untuk meningkatkan konsentrasi dan melatih untuk memecahkan masalah dengan tepat dan cepat karena dalam game terdapat berbagai konflik atau masalah yang menuntut kita untuk menyelesaikannya dengan cepat dan tepat[9]. Kombinasi karakteristik tersebut menjadikan game sebagai media yang diminati oleh berbagai lapisan masyarakat dan memiliki potensi besar untuk mendukung proses pembelajaran serta peningkatan kemampuan kognitif dan sosial.

2.2 Action platformer

Action platformer adalah genre game yang menggabungkan elemen aksi dan platforming, di mana pemain mengendalikan karakter untuk melompat, menghindari rintangan, serta melawan musuh secara langsung[10]. Game jenis ini menuntut ketepatan kontrol, refleks cepat, dan koordinasi yang baik antar elemen gameplay. Dalam pengembangannya, genre ini membutuhkan sistem kendali karakter yang responsif, desain level yang menantang, serta sistem pertarungan yang seimbang. Game-game action platformer modern sering kali menghadirkan kombinasi elemen eksplorasi, peningkatan kemampuan karakter, dan musuh dengan pola serangan yang bervariasi untuk meningkatkan tantangan dan daya tarik permainan.

2.3 Character Controller

Character controller merupakan sistem inti yang mengatur bagaimana pemain dapat mengontrol karakter utama dalam game, termasuk pergerakan dasar seperti berjalan, berlari, melompat, serta interaksi dengan lingkungan seperti deteksi lantai, gravitasi, dan kolisi[11]. Dalam game 2D, sistem ini umumnya dibangun dengan menggunakan pendekatan state machine untuk mengelola transisi antar kondisi karakter, seperti diam, bergerak, melompat, dan menyerang. Penerapan character controller yang baik harus memperhatikan respons input, kestabilan pergerakan karakter, dan integrasi dengan sistem animasi agar menghasilkan kontrol yang halus dan imersif.

2.4 Game Engine

Game engine merupakan perangkat lunak yang dirancang khusus untuk memfasilitasi proses pengembangan video game, mulai dari pengolahan grafis, simulasi fisika, scripting, hingga sistem animasi dan audio. Peran game engine dalam pengembangan game dapat disamakan dengan fungsi Integrated Development Environment (IDE) pada pengembangan perangkat lunak umum. Seperti halnya objek tiga dimensi yang memerlukan perangkat lunak pengolah objek tiga dimensi, pengembangan video game juga membutuhkan aplikasi khusus berupa game engine. Pengembang juga memiliki opsi untuk menggunakan game engine populer yang sudah tersedia atau mengembangkan engine mereka sendiri sesuai kebutuhan. Beberapa contoh game engine yang umum digunakan dalam industri game antara lain Unreal Engine, Unity Engine, CryEngine, dan Godot Engine. Godot Engine sendiri dipilih karena sifatnya yang open-source dan tidak membebankan biaya lisensi maupun royalti kepada penggunanya[12].

2.5 Godot Engine

Godot Engine (GE) dipilih sebagai platform utama dalam pengembangan game pada proyek ini. GE adalah mesin pengembangan game open-source yang memiliki keunggulan dalam hal kekuatan dan fleksibilitas. Dengan memanfaatkan GE, pengembang dapat mengakses berbagai fitur penting yang dibutuhkan untuk menciptakan game 2D berkualitas, termasuk dukungan untuk sistem fisika, animasi, scripting melalui Godot Script (GS), serta manajemen aset yang efisien. Game

yang dikembangkan umumnya mengacu pada situasi kehidupan nyata sebagai referensi dalam perancangannya. Pengelolaan elemen menggunakan sistem hirarki berbasis node, di mana elemen dapat ditempatkan di dalam scene atau dipindahkan ke scene lain. Hal ini memungkinkan pengembang untuk bekerja secara efisien. Selain itu, antarmuka serta editor dari GE dirancang agar mudah dipahami dan dipelajari oleh pengembang, termasuk mereka yang masih baru. Godot Engine merupakan aplikasi pengembangan video game yang bersifat open-source dan gratis, serta menyediakan sistem node dan scene untuk mendukung penciptaan objek dalam game secara modular. Engine ini juga memungkinkan pengembang untuk sepenuhnya memiliki hak atas game yang mereka buat tanpa perlu membayar royalti atas keuntungan yang diperoleh[13]. Layaknya Integrated Development Environment (IDE) untuk pengembangan aplikasi pada umumnya, game engine ini berfungsi sebagai alat untuk mengembangkan video game[14].

2.6 Role Playing Game (RPG)

Role Playing Game RPG merupakan jenis permainan yang menggunakan perangkat audiovisual dalam permainannya dan sering kali mengusung cerita fiksi. Saat ini, RPG telah berkembang menjadi bagian dari budaya, bentuk seni, media penceritaan, sarana edukasi, dan banyak fungsi lainnya[15]. RPG adalah jenis permainan yang menghadirkan pengalaman bermain seolah-olah pemain menjadi karakter di dalam permainan. RPG biasanya menekankan pada elemen pengembangan karakter yang terukur secara statistik, seperti peningkatan level atau atribut. Permainan RPG umumnya memiliki durasi yang panjang serta kedalaman cerita yang kompleks, menjadikannya tampak seperti narasi nyata. Fokus utama permainan ini adalah pada perkembangan karakter dan penyelesaian tantangan yang dihadapi selama permainan berlangsung[16].

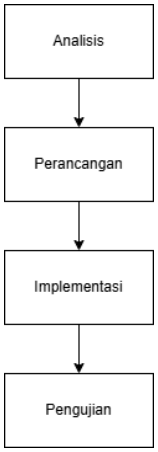
2.7 Combat System

Combat system dalam game action RPG mencakup mekanisme serangan, pertahanan, dan interaksi antara karakter pemain dengan musuh. Sistem ini biasanya mencakup deteksi tabrakan (collision detection), penghitungan damage, efek visual atau animasi serangan, dan sistem respons musuh. Dalam implementasi real-time combat, sinkronisasi antara input pemain, waktu animasi, dan reaksi musuh sangat krusial untuk menciptakan pengalaman bermain yang memuaskan. Desain combat system juga sering melibatkan kombinasi antara logika pemrograman dan desain gameplay agar sistem pertarungan terasa seimbang dan menyenangkan[17].

2.8 Hierarchical Finite State Machine (HFSM)

Hierarchical Finite State Machine (HFSM) adalah pengembangan dari Finite State Machine (FSM) yang dirancang untuk mengelola logika perilaku secara modular dan terstruktur melalui konsep hirarki antar state. Dalam implementasinya pada karakter pemain (player), HFSM memungkinkan pengelompokan state seperti “idle”, “run”, atau “attack” sebagai parent-state yang dapat memiliki sub-state seperti “attack_combo” atau “run_jump”, sehingga mempermudah pengaturan transisi dan pewarisan perilaku[17]. Pendekatan ini tidak hanya mengurangi kompleksitas logika, tetapi juga meningkatkan efisiensi dan keterbacaan kode, terutama ketika pemain memiliki banyak kondisi dan aksi yang saling terkait dalam gameplay.

3. METODE PENELITIAN



Gambar 1. Metode Penelitian

Penelitian ini mengadopsi metode rekayasa perangkat lunak untuk merancang, mengimplementasikan, dan menguji sistem character player pada game Action RPG 2D. Fokus pengembangan diarahkan pada penerapan Hierarchical Finite State Machine

(HFSM) menggunakan Godot Engine versi 4.3, untuk mendukung pengelolaan perilaku karakter yang kompleks secara modular dan terstruktur. Metode terdiri dari empat tahap utama, yaitu: analisis kebutuhan, perancangan sistem, implementasi, dan pengujian.

3.1 Analisis

Tahap awal ini bertujuan untuk mengidentifikasi kebutuhan karakter utama dari sisi kontrol dan sistem pertarungan. Analisis dilakukan dengan pendekatan teknis dan konseptual, yaitu: Studi pada game Action RPG yang telah ada, terutama pada pola kontrol karakter dan pertarungan waktu nyata. Salah satu contohnya adalah Hollow Knight, yang menggunakan pendekatan Hierarchical Finite State Machine (HFSM) untuk mengelola perilaku kompleks karakter dan musuh secara modular dan terstruktur. HFSM memungkinkan transisi antar-state seperti idle, run, jump, attack, hingga damage dilakukan secara fleksibel dan efisien. Identifikasi kebutuhan fungsional sistem, meliputi navigasi karakter (idle, jump, lari, dash), interaksi pemain (input keyboard dan respon visual), sistem serangan (ground attack, air attack, deteksi musuh, hit damage), serta manajemen animasi dan status karakter (bergerak, menyerang, damage).

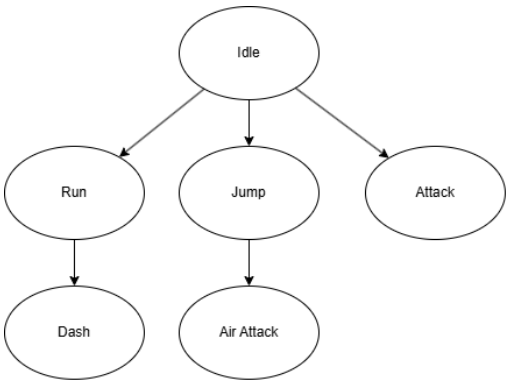
Dari analisis ini, disimpulkan bahwa sistem kontrol dan pertarungan karakter perlu dikelola secara hierarkis karena karakter memiliki state yang bercabang dan saling bergantung.

3.2 Perancangan/Desain

Secara umum, Finite State Machine (FSM) sering digunakan dalam pengembangan game karena sederhana dan fleksibel. Namun, saat digunakan untuk mengatur perilaku player controller yang kompleks, FSM dapat menjadi sulit dikelola akibat banyaknya state dan transisi. Untuk mengatasi hal ini, Hierarchical Finite State Machine (HFSM) menjadi solusi yang lebih efisien. HFSM menyederhanakan desain perilaku dinamis dengan sistem aturan yang mengurangi kompleksitas dan beban komputasi, sehingga lebih terstruktur dan mudah dikembangkan[18]. Struktur HFSM memberikan fleksibilitas dalam mengelola state yang saling bertumpuk, misalnya Combat memiliki sub-state Attack, Combo, dan Cooldown, sejalan dengan prinsip perancangan dan desain game yang mencakup tujuan, perangkat pendukung, serta alur dan aset visual game.

Struktur karakter pemain disusun dengan CharacterBody2D sebagai node utama. Komponen penting yang digunakan antara lain AnimationTree dan AnimationPlayer untuk mengatur animasi kompleks, Area2D untuk mendeteksi serangan terhadap musuh, Timer untuk mengelola durasi state dan cooldown, serta CanvasLayer untuk debug overlay selama pengujian.

Desain HFSM pada karakter ini mengatur alur transisi antar state secara hierarkis dan berjenjang, sehingga perilaku karakter dapat dibagi dan dikendalikan secara lebih terstruktur. Karakter memulai dari state dasar yaitu idle, yang menjadi titik pusat dari berbagai transisi. Dari idle, karakter dapat berpindah ke state run ketika terdapat input pergerakan. Selama dalam state run, jika pemain melakukan input tertentu seperti tombol dash, maka state akan berpindah ke dash, memungkinkan karakter melakukan gerakan cepat. Selain itu, dari idle, pemain juga dapat melakukan lompatan, yang membawa karakter ke state jump. Dalam kondisi melompat ini, jika tombol serang ditekan, maka state akan berubah ke air attack, memungkinkan terjadinya serangan udara. Sementara itu, dari kondisi idle juga terdapat jalur langsung ke state attack, yang mewakili serangan dasar saat karakter diam.



Gambar 2. Desain HFSM

Alur ini menunjukkan bahwa idle menjadi simpul utama dari seluruh transisi, dengan arah transisi yang bergantung pada konteks aksi yang diinginkan pemain, baik itu gerakan, lompat, dash, maupun serangan. Struktur ini tidak hanya memudahkan pengaturan transisi antar state, tetapi juga menjaga agar setiap perpindahan terasa logis dan sesuai dengan situasi karakter dalam permainan.

3.3. Implementasi/Pengkodean

Pengkodean dilakukan menggunakan Godot 4.3 dan GDScript 2.0. Implementasi terbagi menjadi dua modul besar: Character Controller dan Combat System. Tahapan implementasi ini merupakan tahap di mana dilakukan penulisan kode program yang membentuk sistem gameplay secara menyeluruh, dan membutuhkan kemampuan logika yang baik dalam penyusunan alur dan struktur kode.

1. Implementasi HFSM

Sistem HFSM diimplementasikan dengan sistem pendaftaran state secara dinamis menggunakan dictionary. Masing-masing state diwakili oleh skrip turunan dari BaseState.gd dan dimuat sebagai resource melalui preload() atau load().

2. Implementasi Character Controller

Sistem gerak seperti idle, run, dan dash dikelola melalui state MovementState. Input keyboard dipantau dan dikirimkan ke state aktif. Fungsi move_and_slide() digunakan untuk pergerakan, sedangkan animasi dikendalikan melalui AnimationPlayer.

3. Implementasi Combat System

Serangan dipicu dari input (Input.is_action_just_pressed("attack")) yang diteruskan ke state Attack. Damage musuh diberikan jika musuh berada dalam Area2D saat frame serangan aktif. Sistem combo diatur menggunakan urutan state Attack1, Attack2, dan Attack3, dengan waktu input terbatas untuk transisi antar combo. Setelah menyerang, karakter kembali masuk ke state idle.

3.4. Pengujian

Pengujian menggunakan Black Box (Black Box Testing) yaitu metode dalam pengujian perangkat lunak yang berfokus pada pengujian fungsionalitas sistem tanpa memperhatikan struktur internal dari perangkat lunak itu sendiri. Pengujian ini bertujuan untuk memastikan bahwa sistem berfungsi sesuai dengan spesifikasi atau kebutuhan yang telah ditentukan[19].

Proses pengujian dilakukan dengan menjalankan program atau sistem. Pengujian dilakukan menggunakan metode Blackbox Testing, yang bertujuan untuk memastikan bahwa sistem telah berjalan sesuai dengan fungsinya tanpa melihat struktur internal kode. Pengujian ini juga mencakup evaluasi tampilan antarmuka apakah sudah sesuai dengan ekspektasi pengguna, serta pengecekan terhadap kemungkinan error saat sistem dijalankan, guna memastikan performa optimal dan pengalaman bermain yang lancar bagi pengguna.

4. HASIL DAN PEMBAHASAN

4.1 Hasil Implementasi Character Controller

Implementasi sistem character controller dalam proyek ini dilakukan menggunakan Godot Engine versi 4 dengan memanfaatkan node CharacterBody2D sebagai komponen utama untuk mengatur pergerakan karakter dalam lingkungan 2D. Fokus utama dari implementasi ini adalah memberikan kendali gerak yang responsif dan visualisasi gerakan yang halus, sehingga menciptakan pengalaman bermain yang menyenangkan dan intuitif bagi pemain.

Fitur-fitur dasar yang berhasil diimplementasikan antara lain berjalan ke kiri dan kanan, melompat, serta deteksi lantai. Sistem pergerakan ini dikendalikan menggunakan skrip GDScript, yang memanfaatkan metode seperti move_and_slide() untuk menghasilkan gerakan yang realistis namun tetap terkontrol. Pengolahan input dilakukan melalui fungsi Input.is_action_pressed() dan Input.is_action_just_pressed() untuk memastikan respons yang akurat terhadap aksi pemain. Bahasa pemrograman yang digunakan dalam pengembangan ini adalah C#, sebuah bahasa yang dikembangkan oleh Microsoft dalam platform .NET dan dirancang dengan sintaks yang menyerupai C namun lebih sederhana dan mudah dipahami[20].

Karakter juga dilengkapi dengan sistem gravitasi dan deteksi tabrakan menggunakan properti seperti is_on_floor() yang dimiliki CharacterBody2D, sehingga karakter hanya bisa melompat saat berada di tanah dan tidak melayang di udara. Semua logika ini dirancang untuk bekerja mulus dalam berbagai kondisi permainan.

Dari sisi visual, sistem animasi karakter dikendalikan menggunakan node AnimatedSprite2D atau AnimationPlayer, yang terhubung dengan skrip kontrol untuk mencerminkan status karakter saat idle, berjalan, atau melompat. Dengan mengatur animasi secara dinamis berdasarkan kondisi kecepatan atau arah pergerakan, transisi antar

animasi dapat berjalan halus dan sesuai konteks.

Dengan memanfaatkan fitur-fitur yang tersedia dalam Godot 4 dan struktur node yang modular, sistem character controller ini mampu mengintegrasikan logika pergerakan, respons fisik, dan animasi visual secara menyatu, sehingga mendukung alur permainan secara optimal dan menghadirkan pengalaman bermain yang berkualitas.

Implementasi sistem character controller dilakukan dengan fokus pada kendali gerak dan respons visual dari karakter utama. Fitur-fitur utama yang berhasil diimplementasikan antara lain, pergerakan 4 arah yaitu karakter dapat bergerak ke arah atas, bawah, kiri, dan kanan dengan respons kontrol yang halus.

Gambar 3. Pergerakan 4 arah

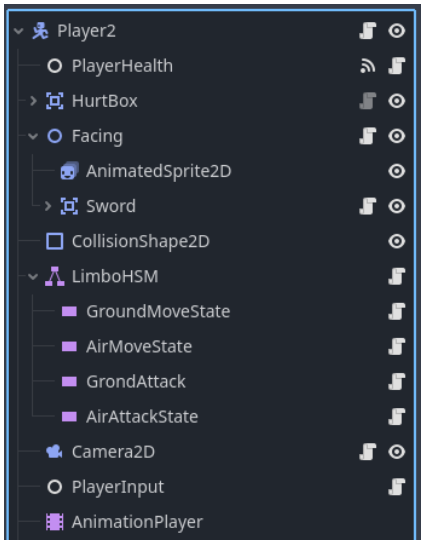
Kemudian fitur animasi karakter yaitu tersedianya animasi untuk berbagai kondisi seperti idle, run, jump, dash, attack123, dan air attack, yang akan aktif berdasarkan input pemain dan kondisi karakter.

Gambar 4. Animasi karakter

Selanjutnya deteksi objek interaktif yaitu sistem mampu mendeteksi interaksi dengan objek tertentu seperti objek klaim hadiah (claim rewards), healing zone, dan spawn trigger yang akan memicu event tertentu saat karakter berada dalam area tersebut.

Gambar 5. Deteksi object

Terakhir struktur node dalam Godot menggunakan pendekatan hierarki berbasis scene dan node, dengan node utama berupa CharacterBody2D sebagai induk karakter, dan node tambahan seperti AnimatedSprite, CollisionShape2D, serta Area2D untuk deteksi interaksi.



Gambar 6. Struktur node

4.2 Hasil Implementasi Combat System

Sistem pertarungan yang dikembangkan dalam game ini berhasil mengintegrasikan berbagai mekanisme interaktif yang mendukung gameplay bergenre aksi RPG. Sistem ini dirancang dengan pendekatan modular dan dikendalikan melalui skrip GDScript di Godot Engine versi 4, dengan memanfaatkan node utama CharacterBody2D sebagai basis pergerakan dan interaksi karakter. Implementasi sistem pertarungan difokuskan untuk menciptakan pengalaman bermain yang responsif, halus, dan terasa natural ketika pemain melakukan aksi bertarung.

Karakter pemain dapat melakukan serangan menggunakan pedang yang diposisikan sebagai bagian dari node visual karakter. Saat pemain menekan tombol serang, sistem akan menjalankan animasi serangan yang dikontrol menggunakan AnimatedSprite2D atau AnimationPlayer, sekaligus mengaktifkan deteksi tabrakan dari area pedang. Ketika serangan ini mengenai musuh yang memiliki sistem HurtBox, maka akan terjadi interaksi berupa pengurangan HP pada musuh. Efek ini memperkuat rasa impact dari setiap serangan yang dilakukan.

Seluruh logika pertarungan diatur menggunakan sistem state machine berbasis node, yang memungkinkan pengelompokan aksi berdasarkan kondisi seperti berada di tanah atau di udara. Misalnya, terdapat perbedaan antara status GroundAttack dan AirAttack yang masing-masing mengatur bagaimana karakter menyerang dalam posisi tersebut. Dengan memisahkan setiap aksi ke dalam status tersendiri, sistem menjadi lebih mudah untuk dikembangkan dan dikelola, termasuk untuk penambahan fitur seperti combo attack, jeda serangan, atau efek khusus di masa depan.

Selain menyerang, sistem juga mengatur bagaimana karakter bereaksi ketika terkena serangan. Ketika karakter pemain terdeteksi bersentuhan dengan hitbox serangan musuh, node HurtBox akan mengirimkan sinyal ke komponen PlayerHealth untuk mengurangi nilai HP. Proses ini dapat diikuti dengan animasi khusus atau efek visual seperti kedipan sprite, suara terkena pukulan, atau getaran kamera sebagai respons terhadap kerusakan yang diterima.

Salah satu aspek penting dalam sistem ini adalah integrasi antara input pemain dengan logika status. Input seperti gerakan, serangan, dan lompatan ditangani melalui node PlayerInput, lalu diteruskan ke sistem state machine yang menentukan status aktif karakter. Misalnya, saat pemain menekan tombol serang di udara, status AirAttack akan diaktifkan dan logika serta animasi yang sesuai akan dijalankan secara otomatis.

Dengan struktur dan logika yang dibangun secara menyeluruh, sistem pertarungan ini mampu menciptakan pengalaman yang dinamis dan mendalam. Pemain tidak hanya dapat bergerak dan menyerang, tetapi juga mengalami sistem reaksi, pembatasan aksi berdasarkan status, serta efek visual yang membuat pertarungan terasa hidup. Sistem ini juga telah dirancang agar fleksibel dan mudah dikembangkan lebih lanjut sesuai dengan kebutuhan gameplay dan desain level yang lebih kompleks.

Fitur sistem pertarungan yang berhasil diimplementasikan mencakup berbagai mekanisme interaktif yang mendukung gameplay RPG aksi, di antaranya yaitu HP dan UI pertarungan yang didalamnya tersedia tampilan total max HP dan current HP dari karakter utama seperti terlihat pada Gambar 7.

Gambar 7. HP dan UI bar

Kemudian mekanisme serangan yaitu pemain dapat melakukan serangan biasa (attack) dan serangan di udara (air attack), yang akan memicu animasi serangan seperti pada Gambar 8.

Gambar 8. Mekanisme serangan

Setelah itu terdapat juga fitur klaim coin dan upgrade stats yaitu setiap membuka chest akan mendapatkan koin. Koin ini dapat diklaim untuk melakukan peningkatan (upgrade) terhadap statistik karakter.

Gambar 9. Claim coin upgrade

Statistik yang dapat ditingkatkan dapat dilihat pada Tabel 1 yaitu pemain dapat meningkatkan nilai HP, attack, crit rate, dan crit damage melalui sistem peningkatan atribut. Tampilan dalam game dapat dilihat pada Gambar 10.

Tabel 1. Statistik stats

Stats	Fungsi
HP	Jumlah Health point
Attack	Jumlah attack point
Crit rate	Besarnya kesempatan terjadinya crit damage
Crit damage	Jumlah attack point x crit damage

Gambar 10. Upgrade stats

Skrip UpgradeData.gd dikelola melalui AutoLoad Godot agar fungsinya dapat diakses global, memungkinkan pengelolaan data pemain seperti max health, attack, crit rate, dan crit damage secara efisien.

Selanjutnya sistem damage yaitu musuh atau karakter yang terkena serangan akan menerima pengurangan HP sesuai nilai serangan dan pertahanan seperti terlihat pada Gambar 11. Kalkulasi damage juga dapat dilihat pada Tabel 2.

Gambar 11. System damage

Tabel 2. System damage calculation

Komponen	Penjelasan
Attack	Nilai dasar serangan. Damage bersifat acak ±3 dari nilai aslinya. Contoh: attack 100 = random 97–103
Crit Rate	Peluang (dalam %) serangan menjadi critical. Contoh: 25% berarti 1 dari 4 serangan kemungkinan besar menjadi crit
Crit Damage	Bonus damage saat critical terjadi. Contoh: 50% = attack × 1.5
Final Damage	Attack random × Crit damage (jika terjadi crit)

Sistem healing memungkinkan pemain memulihkan HP melalui item atau zona khusus. Fitur ini membantu pemain bertahan lebih lama saat menjelajah atau bertarung. Proses pemulihan dilakukan dengan menggunakan item yang tersedia dalam inventori atau dengan masuk ke area tertentu yang secara otomatis memulihkan HP, sebagaimana ditampilkan pada Gambar 12.

Gambar 12. Sistem healing

Tampilan visual damage ditampilkan dalam bentuk popup damage serta terdapat visual combo attack ketika serangan dilakukan secara beruntun. Pengurangan HP sesuai nilai serangan dan pertahanan seperti terlihat pada Gambar 11. Kalkulasi damage juga dapat ditunjukkan secara langsung melalui efek popup yang muncul. Saat player melakukan combo attack, setiap serangan menampilkan popup damage. Jika serangan dilakukan beruntun dalam 5 detik, sistem akan menghitung dan menampilkan total damage combo secara visual. Setelah 5 detik tanpa serangan, total damage akan direset. Efek ini memperjelas dampak serangan dan menambah kesan dinamis saat bertarung, seperti terlihat pada Gambar 13 dibawah ini.

Gambar 13. Damage popup

Feedback saat menerima serangan ditunjukkan melalui efek visual seperti animasi

getar, perubahan warna pada karakter, atau kemunculan partikel saat terkena hit. Efek ini membantu memberi tanda bahwa karakter sedang menerima damage dan meningkatkan respons visual pemain terhadap situasi dalam permainan. Visualisasi ini dapat dilihat pada Gambar 12.

Gambar 12. Hit feedback

4.3 Hasil Pengujian

Hasil dari beta test yang dilakukan menunjukkan bahwa game telah mencapai tahap akhir pengembangan dan siap untuk dievaluasi oleh pengguna eksternal, sekaligus mengungkap berbagai kesalahan serta memperoleh masukan yang bermanfaat dari pihak ketiga sebelum peluncuran secara luas[21]. Sejalan dengan itu, hasil dari pengujian blackbox menunjukkan bahwa fungsionalitas perangkat lunak telah berjalan sesuai dengan persyaratan yang ditetapkan, tanpa ditemukan kesalahan pada proses, antarmuka, maupun struktur data[21]. Hasil pengujian menunjukkan bahwa deteksi serangan dengan menggunakan Area2D mampu mengenali musuh dalam jangkauan serangan secara tepat. Selain itu, sistem pengurangan HP berfungsi sesuai harapan, dengan respons yang konsisten dan perhitungan yang stabil saat serangan berhasil mengenai musuh.

Tabel 3. Hasil pengujian control

Pengujian	Hasil
A untuk lari kiri	✓
D untuk lari kanan	✓
Space untuk lompat	✓
Lclick untuk attack	✓
Rclick untuk dash	✓
E untuk menu upgrade	✓
Esc untuk pause menu	✓

Tabel 4. Hasil pengujian state

Pengujian	Hasil
Saat tidak ada input	karakter berada di state idle
Saat tombol arah ditekan	karakter masuk ke state run
Saat serangan ditekan saat berjalan	pindah ke state attack

Tabel 5. Hasil pengujian combat

Pengujian	Kondisi	Hasil
Karakter melakukan attack	Musuh berada di AttackArea	Musuh menerima damage dan HP berkurang
Karakter melakukan air attack	Musuh di bawah karakter	Damage terdeteksi, musuh kehilangan HP
Serangan dilakukan saat musuh berada di luar area	Musuh di luar AttackArea	Tidak ada pengurangan HP
Serangan dilakukan dengan combo	Musuh tetap di posisi	Musuh menerima damage combo
Musuh terkena serangan lalu menghilang (mati)	Musuh HP <= 0	Musuh menghilang setelah HP mencapai 0
Karakter heal	Player HP di bawah 100%	HP bertambah
Musuh menyerang karakter	Player dalam Hurtbox musuh	Player menerima damage
Player menyerang dua musuh lebih	Dua musuh lebih dalam AttackArea	Semua musuh terkena damage

Tabel 6. Hasil pengujian UI

Pengujian	Hasil
Hp player bertambah/berkurang	HP player akan naik/turun
Player menyerang/terserang	Muncul popup damage number
Player melakukan combo attack	Damage number akan ditotalkan
Player mendapat koin	Koin akan bertambah

Pembahasan

Implementasi sistem character controller dilakukan dengan fokus pada kendali gerak dan respons visual dari karakter utama. Fitur-fitur utama yang berhasil diimplementasikan antara

lain, pergerakan 4 arah yaitu karakter dapat bergerak ke arah atas, bawah, kiri, dan kanan dengan respons kontrol yang halus sebagaimana ditunjukkan pada Gambar 3. Kemudian fitur animasi karakter yaitu tersedianya animasi untuk berbagai kondisi seperti idle, run, jump, dash, attack123, dan air attack, yang akan aktif berdasarkan input pemain dan kondisi karakter, seperti terlihat pada Gambar 4. Selanjutnya deteksi objek interaktif yaitu sistem mampu mendeteksi interaksi dengan objek tertentu seperti objek klaim hadiah claim rewards, healing zone, dan spawn trigger yang akan memicu event tertentu saat karakter berada dalam area tersebut, sebagaimana terlihat pada Gambar 5. Terakhir struktur node dalam Godot menggunakan pendekatan hierarki berbasis scene dan node, dengan node utama berupa CharacterBody2D sebagai induk karakter, dan node tambahan seperti AnimatedSprite, CollisionShape2D, serta Area2D untuk deteksi interaksi, seperti diperlihatkan pada Gambar 6.

Fitur sistem pertarungan yang berhasil diimplementasikan mencakup berbagai mekanisme interaktif yang mendukung gameplay RPG aksi, di antaranya yaitu HP dan UI pertarungan yang di dalamnya tersedia tampilan total max HP dan current HP dari karakter utama seperti terlihat pada Gambar 7. Kemudian mekanisme serangan yaitu pemain dapat melakukan serangan biasa attack dan serangan di udara air attack, yang akan memicu animasi serangan sebagaimana ditunjukkan pada Gambar 8. Setelah itu terdapat juga fitur klaim coin dan upgrade stats yaitu setiap membuka chest akan mendapatkan koin. Koin ini dapat diklaim untuk melakukan peningkatan upgrade terhadap statistik karakter, seperti yang ditampilkan pada Gambar 9. Statistik yang dapat ditingkatkan dapat dilihat pada Tabel 1 yaitu pemain dapat meningkatkan nilai HP, attack, crit rate, dan crit damage melalui sistem peningkatan atribut. Tampilan dalam game ditunjukkan pada Gambar 10. Skrip UpgradeData.gd dikelola melalui AutoLoad Godot agar fungsinya dapat diakses global, memungkinkan pengelolaan data pemain seperti max health, attack, crit rate, dan crit damage secara efisien.

Selanjutnya sistem damage yaitu musuh atau karakter yang terkena serangan akan menerima pengurangan HP sesuai nilai serangan dan pertahanan sebagaimana diperlihatkan pada Gambar 11. Kalkulasi damage juga dapat dilihat pada Tabel 2. Sistem healing dalam game memungkinkan pemain untuk memulihkan HP mereka melalui item atau zona khusus, seperti terlihat pada Gambar 12. Mekanisme ini memberikan dukungan penting bagi pemain dalam mempertahankan HP karakter, terutama saat menghadapi musuh atau menjelajahi area yang berbahaya. Saat player melakukan combo attack, setiap serangan menampilkan popup damage. Jika serangan dilakukan beruntun dalam 5 detik, sistem akan menghitung dan menampilkan total damage combo secara visual. Setelah 5 detik tanpa serangan, total damage akan direset, sebagaimana ditunjukkan pada Gambar 13. Fitur ini tidak hanya memberikan informasi penting kepada pemain, tetapi juga menambah kesan dramatis dan responsif saat pertarungan berlangsung, sebagaimana diperjelas melalui Gambar 13.

5. KESIMPULAN

Berdasarkan implementasi fitur-fitur di atas, sistem character controller dan combat system dalam game berhasil dibangun secara terstruktur dan modular menggunakan pendekatan Hierarchical Finite State Machine (HFSM). Pendekatan ini memungkinkan pemisahan logika antar state utama seperti Idle, Run, Jump, Attack, Air Attack, dan Interaksi, serta pengelompokan state turunan seperti animasi spesifik dalam kondisi bertarung maupun saat menjelajah lingkungan. Struktur HFSM ini memperjelas alur transisi antar kondisi karakter, sehingga sistem dapat merespons input pemain dan event lingkungan secara efisien dan adaptif. Misalnya, saat karakter berada pada state Run dan pemain menekan tombol serang, sistem akan berpindah ke state Attack dengan sub-state animasi yang sesuai. Selain itu, penggunaan HFSM juga memperkuat pengelolaan interaksi seperti klaim hadiah, healing zone, hingga mekanisme upgrade dan perhitungan damage, dengan menjaga transisi antar state tetap terorganisir tanpa konflik logika. Dengan demikian, implementasi berbasis HFSM memberikan pondasi yang kokoh untuk membangun sistem gameplay yang responsif, fleksibel, dan mudah dikembangkan lebih lanjut.

DAFTAR PUSTAKA

[1] Dwifa Yuda Pradana, Anik Vega Vitianingsih, Dwi Cahyono, Anggit Wikaningrum, Seftin Fitri Ana Wati, "Pengembangan Aplikasi Game Pengenalan Jenis-Jenis Virus Berbasis RPG," *Jurnal Teknologi Terpadu*, vol. 10, no. 2, hlm. 78, 2024.

[2] Ibzani Ilham Shagianto, Giri Wahyu Wriasto, Djul Fikry Budiman, Misbahuddin, Ni Made Seniari, "Aplikasi Game berbasis Android 2D dengan Logika Fuzzy pada NPC (Non-Player Character)," *JEITECH*, vol. 1, no. 1, hlm. 42, 2023.

[3] Bagus Tri Sasongko, "Pengembangan Game Escape From Basement dengan AI Berbasis Finite State Machine di Godot Engine Menggunakan Metode GDLC," *Prosiding KONSTELASI*, vol. 2, no. 1, hlm. 96, 2025.

[4] S. Y. Cheung dan K. Y. Ng, "Application of the Educational Game to Enhance Student Learning," *Front. Educ.*, vol. 6, hlm. 623793, Mar 2021, doi: 10.3389/feduc.2021.623793.

[5] G. Mingyu, M. Md Yunus, dan K. R. M. Rafiq, "Educational Games and Game-based Approaches in Higher Education: A Systematic Review (2014-2023)," *IJARPED*, vol. 13, no. 1, hlm. Pages 899-919, Jan 2024, doi: 10.6007/IJARPED/v13-i1/20555.

[6] Muhammad Fikriansyah, Giri Wahyu Wiriasto, A. Sjamsjiar Rachman, "Rancang Bangun Prilaku Buatan pada Non-Player Character dalam Game Pemadam Kebakaran menggunakan Finite State Machine dan Godot Script," *Dielektrika*, vol. 10, no. 1, hlm. 1, 2023.

[7] Leonard Winata, Muhammad Akbar Maulana, Joko Susilo, "Studi Perbandingan Pengembangan Game dalam GDScript dengan Godot dan C# dengan Unity," *Bit-Tech*, vol. 7, no. 3, hlm. 716, 2025.

[8] Zakiey Cahya Ardi Wahana, Suryo Adi Wibowo, Abdul Wahid, "Game Adventure Horror 'Let's Escape' dengan Unity Engine Berbasis Desktop Menggunakan Metode Finite State Machine," *JATI*, vol. 4, no. 2, hlm. 306, 2020.

[9] Nopi Ramsari, Gilang Ramadhan, "Pembuatan Game Side Scrolling 2D The Naila's Survival Berbasis Android," *FIKI*, vol. VIII, no. 2, hlm. 70, 2018.

[10] M. F. Arrazzaq, A. P. Sasmito, dan H. Z. Zahro', "PERANCANGAN GAME 2D PLATFORMER 'ADVENTURE QUEST' DENGAN METODE FINITE STATE MACHINE BERBASIS ANDROID," *JATI (Jurnal Mahasiswa Teknik Informatika)*, vol. 7, no. 4, Art. no. 4, Des 2023, doi: 10.36040/jati.v7i4.7537.

[11] S. Octodinata, J. Pragantha, dan D. A. Haris, "PEMBUATAN GAME 2D PLATFORMER 'SAVE THE FOXY' PADA WEBSITE," *JSSTK*, vol. 1, no. 2, hlm. 431–442, Okt 2023, doi: 10.24912/jsstk.v1i2.31034.

[12] Ferdi, Sasa Ani Arnomo, "Perancangan Game Platformer Pemburu Koin Menggunakan Godot Engine," *Jurnal Comasie*, vol. 6, no. 4, hlm. 112, 2022.

[13] Kevin Asgaryansyah, Giri Wahyu Wiriasto, "Rancangan Game 2D RPG Berbasis Android Dengan Metode Extreme Programming Menggunakan Godot Script," *Dielektrika*, vol. 10, no. 2, hlm. 76, 2023.

[14] Ricky, Mesri Silalahi, "Rancang Bangun Game Platformer Bintang Kecil Menggunakan Godot Engine," *Jurnal Comasie*, vol. 5, no. 2, hlm. 128, 2021.

[15] Vanessa Metayani, Robby Tan, "V-Fighter: Game Platformer 2D Protokol Kesehatan Dengan Menggunakan Godot (Strategi)," *Jurnal Strategi*, vol. 5, no. 1, hlm. 33, 2023.

[16] Made Agus Panji Sujaya, I Gede Mahendra Darmawuguna, Made Windu Antara Kesiman, "Pengembangan Game Rpg 2D Legenda Desa Trunyan," *INSERT*, vol. 2, no. 2, hlm. 88, 2021.

[17] E. N. F. Dewi, A. N. Rachman, dan R. H. Z. Hartadji, "Implementation of Hierarchical Finite State Machine for Controlling 2D Character Animation in Action Video Game," dalam *2023 Eighth International Conference on Informatics and Computing (ICIC)*, Manado, Indonesia: IEEE, Des 2023, hlm. 1–6. doi: 10.1109/ICIC60109.2023.10381978.

[18] M. Umar Yanto, Hani Zulfia Zahro, Renaldi Primaswara Prasetya, "The Legend of The Knight 2D Menggunakan Metode Hierarchical Finite State Machine (HFSM) Berbasis Android," *JATI*, vol. 8, no. 6, hlm. 12409, 2024.

[19] Diva Putri Kynta, Muhammad Rizky Pribadi, "Implementation of Fuzzy Logic in Educational Game on Manners and Morals for Kids Using Godot Engine," *Journal of Artificial Intelligence and Software Engineering*, vol. 5, no. 1, hlm. 344, 2025.

[20] Gina Rahma Fitriyani Asiqin, Yusuf Sumaryana, Cepi Rahmat Hidayat, "Game Edukasi Pembelajaran Budaya Flores Berbasis Android Dengan Menggunakan Metode Game Development Life Cycle (GDLC)," *JITET*, vol. 13, no. 1, hlm. 1324, 2025.

[21] Riza Ainun Jariyah, Moch. Lutfi, "Game Edukasi Pengenalan Alat Musik Tradisional

3D Untuk Anak Usia Dini," *JITET*, vol. 12, no. 3S1, hlm. 4173, 2024.