UNIVERSIDADE FEDERAL DO CEARÁ

CAMPUS DE QUIXADÁ

CURSO DE GRADUAÇÃO EM SISTEMAS DE INFORMAÇÃO

JEAN IGOR DE QUEIROZ PANTOJA

ESTUDO COMPARATIVO DE MOTORES DE JOGOS NO DESENVOLVIMENTO DE UM JOGO 2D PARA WEB.

QUIXADÁ

2022

JEAN IGOR DE QUEIROZ PANTOJA

ESTUDO COMPARATIVO DE MOTORES DE JOGOS NO DESENVOLVIMENTO DE UM JOGO 2D PARA WEB.

Trabalho de Conclusão de Curso apresentado ao Curso de Graduação em Sistemas de Informação do da Universidade Federal do Ceará Campus de Quixadá, como requisito parcial à obtenção do grau de bacharel em Sistemas de Informação.

Orientador: Prof. Dr. Jefferson de Carvalho Silva

Coorientadora: Profa. Dra. Valéria Lelli Leitão Dantas

QUIXADÁ

2022

Dados Internacionais de Catalogação na Publicação
Universidade Federal do Ceará
Sistema de Bibliotecas
Gerada automaticamente pelo módulo Catalog, mediante os dados fornecidos pelo(a) autor(a)

P222e Pantoja, Jean Igor de Queiroz.
Estudo comparativo de Motores de jogos no desenvolvimento de jogo 2D para web. / Jean Igor de Queiroz Pantoja. – 2022.
47 f.

Trabalho de Conclusão de Curso (graduação) – Universidade Federal do Ceará, Campus de Quixadá, Curso de Sistemas de Informação, Quixadá, 2022.
Orientação: Prof. Dr. Jefferson de Carvalho Silva.
Coorientação: Profa. Dra. Valéria Lelli Leitão Dantas.

1. jogos eletrônicos. 2. jogos - desenvolvimento. 3. motor de jogo. I. Título.

CDD 005

JEAN IGOR DE QUEIROZ PANTOJA

ESTUDO COMPARATIVO DE MOTORES DE JOGOS NO DESENVOLVIMENTO DE UM JOGO 2D PARA WEB.

Trabalho de Conclusão de Curso apresentado ao Curso de Graduação em Sistemas de Informação do da Universidade Federal do Ceará Campus de Quixadá, como requisito parcial à obtenção do grau de bacharel em Sistemas de Informação.

Aprovada em: ____/____/_____

BANCA EXAMINADORA

Prof. Dr. Jefferson de Carvalho Silva (Orientador)
Universidade Federal do Ceará (UFC)

Profa. Dra. Valéria Lelli Leitão Dantas (Coorientadora)
Universidade Federal do Ceará (UFC)

Prof. Dr. Victor Aguiar Evangelista de Farias
Universidade Federal do Ceará (UFC)

Prof. Dr. Bruno Góis Mateus
Universidade Federal do Ceará (UFC)

RESUMO

A partir dos avanços tecnológicos em conjunto com a instauração da internet, implicou-se na expansão das indústrias de jogos digitais, proporcionando empregos, faturamentos e possibilitando inovações dentro da esfera tecnológica. Diante disso, com o crescimento da indústria e o aumento de jogos mais complexos, sua produção passou a necessitar de equipes mais qualificadas com profissionais multiprofissionais, que com a disseminação dos computadores, tornou-se capaz de produzir jogos em sua plataforma. Fato esse corroborado com o surgimento de ferramentas com diversas funcionalidades e bibliotecas que auxiliam no desenvolvimento de jogos, dentre as ferramentas utilizadas no mercado, os motores de jogos se sobressaem por possuírem funcionalidades pré programadas que facilitam o processo de desenvolvimento. Além disso, os motores permitem distribuir o produto final para as mais diversas plataformas, como videogames, computadores, celulares e até mesmo para navegadores WEB. Nesse cenário, este trabalho propõe conduzir um estudo comparativo entre os motores de jogos mais pesquisados nos últimos 12 meses no brasil, através da implementação de um jogo 2D para WEB, com o objetivo de identificar qual motor possui o melhor custo benefício para usuários iniciantes em desenvolvimento de jogos. Em virtude disso, este trabalho dispõe-se a auxiliar o usuário programador a selecionar o melhor motor de jogos a partir de suas necessidades.

Palavras-chave: jogos eletrônicos; jogos - desenvolvimento; motor de jogo

ABSTRACT

From technological advances in conjunction with the establishment of the internet, it was implied the expansion of the digital gaming industries, providing jobs, revenues and enabling innovations within the technological sphere. Given this, with the growth of industry and the increase of more complex games, its production has needed more qualified teams with multiprofessional professionals, who with the dissemination of computers, has become capable of producing games on their platform. This is corroborated with the emergence of tools with various features and libraries that help in the development of games, among the tools used in the market, game engines stand out because they have pre -programmed features that facilitate the development process. In addition, engines allow you to distribute the final product to the most diverse platforms, such as video games, computers, cell phones and even web browsers. In this scenario, this paper proposes to conduct a comparative study between the most searched game engines in the last 12 months in Brazil, by implementing a 2D web game, with the objective of identifying which engine has the best cost benefit for novice users in Game development. As a result, this work is willing to assist the programmer user to select the best game engine from their needs.

Keywords: electronic games; games - development; game engine.

SUMÁRIO

1 INTRODUÇÃO

Com o advento da globalização e o estabelecimento da Internet, notou-se o avanço no âmbito tecnológico. Fato este que implicou na expansão das indústrias de jogos digitais em conjunto com as políticas públicas de incentivo à cultura com o foco em desenvolvimento econômico, no qual, a criatividade é o ponto de partida (AMÉLIO, 2018).

Os jogos digitais são vistos como bens informacionais com componentes multidisciplinares o que possibilita a incorporação de valor ao longo da cadeia de produção com potencial de geração de empregos e rendas, além de dispor da capacidade de mobilização social e cognitiva por meio de fomento da cultura local e nacional (AMÉLIO, 2018).

Historicamente na indústria de jogos digitais, as políticas públicas para o setor foram estimuladas devido às suas características multidisciplinares e convenientes para o desenvolvimento econômico, tendo diferentes característica e mudanças ao longo dos anos (AMÉLIO, 2018).

As primeiras mudanças foram voltadas para tecnologia, no qual, tinha como base restrições de importações e bens de consumo o que implicava em um cenário em que as instituições não tinham espaço para articulação sendo de suma importância o foco em promover, organizar, fortalecer a indústria brasileira. Nesse cenário surgem as políticas culturais com o foco em edital voltado unicamente para o setor de jogos, em que os jogos digitais passaram a ser avaliados de maneira única (AMÉLIO, 2018).

Com a relevância dos meios de comunicação as políticas de comunicação se encaixam no audiovisual e linguagens a fim de promover conteúdos interativos, habilidades cognitiva, motoras e criativas, assim os jogos digitais passaram a serem avaliados sob a perspectiva do entretenimento (AMÉLIO, 2018).

Tendo isso como, base de acordo com Becker (2018), além da habilidade de promover empregos e rendas, os jogos digitais, apresentam suma importância ao possibilitar inovações no âmbito tecnológico. Este fato se estende para as mais diversas áreas de atuação assim mostrando a necessidade de estudo.

Assim, os jogos começaram a fazer parte da rotina da população nas últimas décadas, sendo fonte de entretenimento entre jovens, tornando-se usual, atualmente nas mais diversas faixas etárias, resultando-se em fontes de aprendizagem em escolas (MOTTA; JUNIOR, 2013).

Com o crescimento do mercado e o aumento de jogos mais estruturados, a produção, que antes pequenos grupos organizavam-se a fim de idealizar, desenvolver e publicar, passa a ter

equipes mais qualificadas com mais de cem integrantes, havendo a necessidade de uma equipe multiprofissional (MOTTA; JUNIOR, 2013).

Com a popularização dos computadores e o desenvolvimento de jogos em sua plataforma, passou a ser possível ensinar e aprender desenvolvimento de jogos, resultando que algumas produtoras passaram a disponibilizar ferramentas e até mesmo o código fonte. Isto possibilitou que os usuários pudessem criar jogos com mais facilidade (COSTA *et al.*, 2019).

Assim surgem ferramentas que permitiam que os usuários pudessem participar no desenvolvimento de jogos mesmo não possuindo conhecimento avançado em programação, podemos citar dentre essas ferramentas os motores de jogos (do inglês, *Game Engine*) (COSTA *et al.*, 2019).

Os motores por sua vez auxiliam os usuários no desenvolvimento de jogos, por possuir uma coleção de funcionalidades pré-programadas. Nesse contexto, existem diversas motores diferentes, desde os modelos mais simples com suporte ao desenvolvimento de jogos 2D, até motores mais complexas, suportando gráficos em 3D (SILVA, 2020).

O trabalho Becker (2018) cita que diversos segmentos compõem o mercado dos jogos digitais, tais como: *consoles*, jogos na nuvem, jogos para televisão digital, jogos de mídia física, jogos por *download*, mercado *mobile* e por fim, jogos para WEB. Deste modo, sendo desenvolvidos para as mais diversas finalidades e objetivos, tendo desde caráter educativo até aplicações sociais.

Com o avanço das tecnologias *WEB*, com novas *APIs* e melhorias de desempenho em tecnologias de compilação *just-in-time* para *JavaScript*, tornou-se possível desenvolver jogos para WEB. Isto permitiu que os desenvolvedores possam publicar seus jogos em navegadores nos mais diversos dispositivos: *smartphones*, *tablets*, *computadores* e até *smart tvs* (MOZILLA, 2022).

De acordo com Cointelegraph (2022) o mercado brasileiro de jogos é detentor de cerca de US 2,3 bilhões de dólares, apresentando-se possivelmente como responsável pelo crescimento dos jogos baseados em *tokens* não fungíveis (NFT) no país.

Os jogos NFTs por sua vez se diferenciam dos demais a partir de itens colecionáveis escassos, atraindo jogadores que enxergam a possibilidade de lucrar a partir de peças raras e exclusivas, já que cada item NFT dentro do jogo é único (CANALTECH, 2022). Até janeiro de 2022, os jogos mais populares de NFTs em sua grande maioria podem ser jogados em navegadores *WEB*, abrangendo uma enorme quantidade de usuários (COINTELEGRAPH, 2022).

Nesse cenário, a partir da relevância dos jogos WEB, este trabalho irá desenvolver um jogo de plataforma 2D com o objetivo de avaliar os motores de jogo utilizados no desenvolvimento, almejando identificar qual motores de jogo possui o melhor custo benefício para um usuário iniciante na área de desenvolvimento de jogos. Em razão disso, espera-se que esse presente trabalho possa auxiliar o usuário a escolher o seu motor de jogo para desenvolvimento de jogos WEB de acordo com suas necessidades.

1.1 OBJETIVOS

1.1.1 OBJETIVO GERAL

O objetivo geral deste trabalho é conduzir um estudo comparativo entre tecnologias de desenvolvimento de jogos através da implementação de um jogo 2D para Web.

1.1.2 OBJETIVOS ESPECÍFICOS

1. Definir os requisitos do jogo a ser implementado.
2. Selecionar os motores a serem avaliados.
3. Desenvolver um jogo simples utilizando os motores de jogo escolhidos.
4. Definir métricas, a partir da literatura, a serem usadas na comparação dos motores de jogo.
5. Avaliar os motores de jogos de acordo com as métricas.

2 FUNDAMENTAÇÃO TEÓRICA

Nesta seção é apresentado todo conteúdo teórico essencial para o entendimento do presente trabalho. Nas seções 2.1 e 2.2 é demonstrado conceitos importantes para o entendimento de jogos digitais. Nas seções 2.3 e 2.4, apresenta-se toda a parte teórica a respeito das tecnologias que será utilizada neste trabalho. Na seção 2.5 é apresentado o conceito teórico do Documento de *Design* de jogo.

2.1 TIPOS DE JOGOS DIGITAIS

Nessa seção será abordado os tipos de jogos digitais, sendo eles: jogos em duas dimensões (2D) e em três dimensões (3D).

JOGOS 2D: Sendo a primeira geração de jogos de computadores, os jogos 2D são estruturas desenvolvidas em 2 planos dimensionais em eixo X e Y usando gráficos de caráter plano chamados de *sprites* assim tendo sua imagem atraída de forma planar e suas movimentações ocorrendo com base no eixo X e Y, normalmente acontecendo da esquerda para a direita, para cima e para baixo, conforme apresentado na Figura 1 (NGUYEN, 2021).

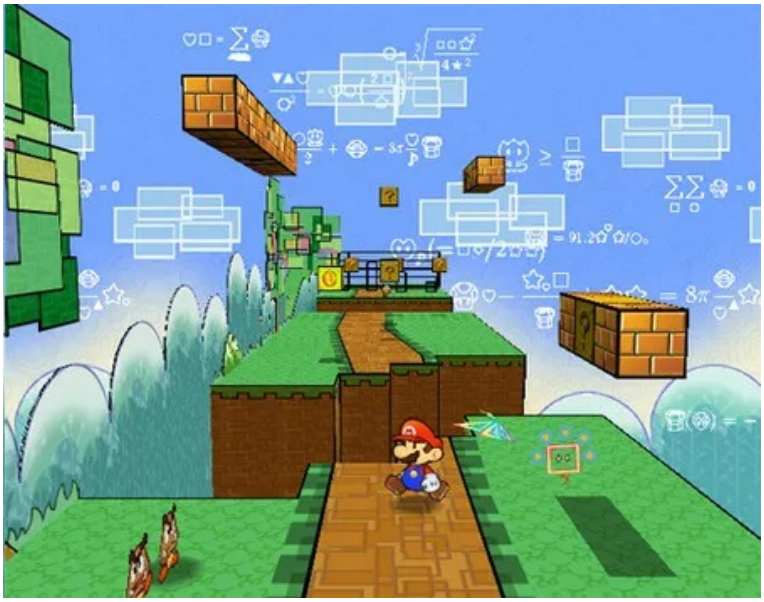
Além disso, é notória a importância da preferência dos jogadores, no qual, ao escolher um jogo 2D mostra-se preferir jogos mais leves e casuais, em contrapartida a outras classe de jogos que necessitam de mais espaço no disco rígido (NGUYEN, 2021).

Figura 1 – Exemplo de um jogo 2D

Fonte: Imagem retirada da página crie seus jogos. Link: https://www.crieseusjogos.com.br/e-melhor-criar-um-jogo-2d-ou-3d/

JOGOS 3D: Outra classe de jogos com bastante relevância é a 3D, no qual, faz-se o uso do plano tridimensional levando o usuário a sua imersão o que implica em uma simulação realística do mundo, fato que é consequência das texturas e materiais renderizadas na superfície do jogo. Nesse contexto, os objetos presentes nos jogos 3D apresentam profundidade e textura operando nos eixos X, Y e Z, como pode ser visto na Figura 2 (NGUYEN, 2021).

Figura 2 – Exemplo de um jogo 3D



Fonte: Imagem retirada da página crie seus jogos. Link: https://www.crieseusjogos.com.br/e-melhor-criar-um-jogo-2d-ou-3d/

2.2 GÊNEROS DE JOGOS DIGITAIS

Nessa seção será abordada os gêneros de jogos digitais: Ação, Simulação, Aventura, Estratégia, Quebra Cabeça, RPG e Luta.

JOGOS DE AÇÃO Silva *et al.* (2009) destacam que os jogos de ação consequentemente trazem experiências emocionantes, fazendo-se necessário ter bastante concentração e agilidade por parte do jogador para reagir instantaneamente aos acontecimentos que surgem em tempo real durante o jogo. O jogo *Metroid Dread* da empresa *Nintendo* como exemplo de jogo de ação.

JOGOS DE SIMULAÇÃO Jogos de simulação tem como objetivo retratar um acontecimento que ocorre no mundo real buscando passar as sensações reais do cenário sem apresentar riscos ao jogador(SILVA *et al.*, 2009). O jogo *EuroTruck* é um exemplo de jogo de simulação.

JOGOS DE AVENTURA Em sua essência jogos de aventura possuem um enredo bastante recheado com diversos cenários que desafiam o jogador a solucionar problemas que surgem ao decorrer da história (SILVA *et al.*, 2009). *Tomb Raider* é um exemplo de jogo de aventura.

JOGOS DE ESTRATÉGIA Silva *et al.* (2009) informa que os jogos de estratégias exigem que os jogadores gerencie recursos limitados a fim de conquistar seu objetivo, jogos de estratégias comumente são relacionados a tomada de decisão, podendo ser em tempo real ou baseado em turnos. *Team Fight Tatics* é um exemplo de jogo de estratégia.

JOGOS DE QUEBRA-CABEÇA O gênero de jogos de quebra-cabeça possui como ponto fundamental a solução de problemas ou enigmas (SILVA *et al.*, 2009). O Sudoku é um exemplo de jogo de quebra-cabeça.

JOGOS DE RPG Os jogos de RPG por sua vez permitem que o jogador controle um único personagem principal ou um grupo de personagens com o objetivo de realizar uma missão, possuindo um universo com enredo bem elaborado, além de possuir sistema de evolução dos personagens, permitindo que o jogador evolua a força, poderes e até mesmo utilizar itens melhores (LEMES *et al.*, 2009). *Tibia* é um exemplo de jogo de *RPG*.

JOGOS DE LUTA Lemes *et al.* (2009) Informa que esse gênero exige que no mínimo há dois jogadores (humano ou máquina) para que cada um possa controlar um personagem a sua escolha para um combate, onde cada personagem possui diferentes combinações de ataque, defesa e poderes. O jogo *Street Fighter* é um exemplo de jogo de luta.

2.3 TECNOLOGIAS DE DESENVOLVIMENTO DE JOGOS

Costa *et al.* (2019) introduzem que inicialmente o desenvolvimento de jogos era muito simples, pois diversas ferramentas de apoio não existiam, tendo seus jogos desenvolvidos em editores de texto e em ambientes de desenvolvimento integrado, implicando em uma grande necessidade de conhecimento computacional por parte dos desenvolvedores.

A partir disso, os primeiros videogames utilizavam jogos contidos em um cartucho com código-fonte imutável, sendo necessário conectar fisicamente o cartucho ao videogame para jogar, apesar disso, mesmo com diversas dificuldades, surgem usuários que faziam modificações ou *MODS* (do inglês, *modification*) nos jogos originais a fim de adicionar funcionalidades e interações que não existiam originalmente (COSTA *et al.*, 2019).

Posteriormente com a popularização do computador e o desenvolvimento de jogos

para a plataforma, passou a ser possível ensinar e aprender desenvolvimento de jogos, resultando que algumas produtoras passaram a disponibilizar ferramentas de criação de cenários para seus jogos, conhecidas como *level editing* ou editores de níveis, somado a isso algumas empresas chegaram a disponibilizar o código fonte de seus jogos após certo tempo, como *Duke Nuken* 3D, da 3D *Realms* e o *Quake*, da *ID Software* (COSTA *et al.*, 2019).

O jogo *Quake* recebeu alterações em suas armas e classes, possibilitando aos jogadores de escolher armas únicas e personagens com características específicas, originando o *MOD* chamado *Team Fortress*. Paralelo a isso a empresa responsável continuava disponibilizando ferramentas úteis para que os usuários continuassem inventando e modificando, sendo possível adicionar texturas e até mesmo objetos 3D (COSTA *et al.*, 2019).

Logo em seguida, a empresa ID Software[1] disponibilizou a ferramenta chamada Quake C, baseada na sintaxe e estrutura da linguagem de programação C, permitindo que usuários que possuíam conhecimentos não tão avançados de programação pudessem modificar pequenos trechos de códigos (COSTA *et al.*, 2019).

Por fim diversas ferramentas com alto grau de abstração passaram a surgir, permitindo que os usuários criassem *scripts* com objetivo de alterar diversas características do jogo. Ao decorrer disso, o nível de conhecimento do usuário evoluiu ao ponto de conseguirem criar seus próprios jogos utilizando essas ferramentas, assim alguns desenvolvedores passaram a utilizar motores de jogo em busca de mais liberdade em seu desenvolvimento (COSTA *et al.*, 2019).

2.4 MOTOR DE JOGO

Segundo Silva (2020), motores de jogo frequentemente são utilizadas no desenvolvimento de jogos, dado que possuem uma coleção de funcionalidades pré-programadas que auxilia os desenvolvedores, nesse contexto, a vários motores de jogo diferentes, tendo suas vantagens e desvantagens.

Os modelos mais simples apresentam, geralmente, gráficos em dimensões 2D, enquanto que os mais avançados utilizam gráficos 2D, 3D e mudanças nos aspectos visuais, sonoros e entre outros. Além disso, os motores de jogo, dispõem de licença de uso, tendo motores públicos e gratuitos para a criação e venda de jogos.

Diante disso, motor de jogo é composto por um conjunto de bibliotecas que foram desenvolvidas com o objetivo de colaborar no desenvolvimento de jogos, abstraindo diversos

[1] https://www.idsoftware.com/en-gb

estágios da construção de jogos por meio de editores e ferramentas gráficas (CAVALCANTE; PEREIRA, 2018). Algumas dessas bibliotecas e ferramentas serão abordadas logo a seguir.

DOCUMENTAÇÃO OFICIAL DO MOTOR DE JOGO: Flomén e Gustafsson (2020) informam que a documentação oficial do motor de jogo corresponde a um conjunto de documentos e informações que descrevem as funcionalidades de sua aplicação para o usuário por meio de manuais técnicos e instruções.

AMBIENTE DE DESENVOLVIMENTO INTEGRADO: Ambiente de desenvolvimento integrado (do inglês *integrated development environment* (IDE)) é um software que auxilia o usuário a desenvolver suas aplicações, sendo comumente composto por um conjunto de ferramentas como: editor de código-fonte, depurador de erros, gerenciador de arquivos, compilador, interpretador e alguns ambientes chegam a suportar controle de versionamento de código (FLOMÉN; GUSTAFSSON, 2020).

INTERFACE DE USUÁRIO: A Interface gráfica de usuário (do inglês *Graphical user interface* (GUI)) permite que o usuário interaja com o software mediante a *layouts* e ícones gráficos (FLOMÉN; GUSTAFSSON, 2020).

PERSONAGEM NÃO JOGÁVEL: Silva e Ribeiro (2021) o personagem não jogável é definido por um personagem que não está sob controle de um jogador humano, possuindo a responsabilidade de imitar o comportamento de um jogador real.

OBJETOS DO JOGO: Objetos do jogo são componentes utilizados para criar toda a estrutura do jogo. Os objetos do jogo podem ser qualquer coisa como por exemplo o jogador, cenário, câmera, personagens não jogáveis, efeitos especiais e etc (FLOMÉN; GUSTAFSSON, 2020).

SISTEMA DE ENTRADA: O sistema de entrada permite que o jogador controle seu aplicativo a partir de dispositivos como teclado, mouse, controle, telas sensíveis ao toque e entre outros Unity (2022).

SCRIPTS: *Scripts* é um pedaço de código que contem ações a serem adicionadas a um objeto do jogo, permitindo que o usuário gerencie eventos, controle o sistema de entrada e até mesmo implementar o comportamento de um personagem não jogável (FLOMÉN; GUSTAFSSON, 2020).

FÍSICA: Para Godot (2022) a física permite que o desenvolvedor saiba quando dois ou mais objetos colidem entre si durante o jogo. Unity (2022) complementa que a física do jogo ajuda o desenvolvedor a simular aceleração de objetos, gravidade e várias outras forças.

2.5 DOCUMENTO DE DESIGN DE JOGO

O Documento de design de jogo é um artefato elaborado pelo game designer, no qual descreve diversos aspectos sobre o jogo a ser desenvolvido, a partir de ideias fundamentais do jogo, conceitos, enredo, personagens, cenários e até mesmo possuindo informações mais detalhadas como design de níveis, identidade visual e sons (MOTTA; JUNIOR, 2013).

Souza (2011) escreve em seu texto que uma das principais funções do documento de design de jogo é evitar que a equipe de produção trabalhe isoladamente, formalizando o consenso da equipe de desenvolvimento, além disso essa ferramenta ajuda os desenvolvedores a tirarem dúvidas sobre o projeto.

Para os pesquisadores de desenvolvimento de jogos, o documento de design de jogo é necessário para estruturar as informações com a finalidade de guiar a equipe do projeto. Além do fato de que o documento facilita o entendimento do jogo para os investidores, chegando a ser utilizado como base contratual para definir as etapas do projeto (MOTTA; JUNIOR, 2013).

Este trabalho utiliza o modelo de documento de design de jogo elaborado por Hira *et al.* (2016) com a finalidade de auxiliar na etapa de desenvolvimento a fim de facilitar o entendimento do jogo que será desenvolvido nos motores de jogo selecionados buscando comparar qual possui o melhor custo-benefício.

O modelo conceitual de documento de design de jogo desenvolvido por Hira *et al.* (2016) em seu trabalho apresenta sua estrutura com os seguintes elementos:

1. Sumário.
2. Visão Geral/Resumo.
3. Visão Geral da História.
4. Análise de jogos semelhantes.
5. Lista de Diferenciais.
6. Viabilidade Técnica.
7. Público-alvo e Marketing.
8. Mecânica do jogo.
9. Inteligência artificial.
10. Progressão do Jogo.
11. Arte e Vídeo.
12. Música.
13. Interface.

14. Elementos do jogo.
    a) Personagens.
    b) Itens.
    c) Objetos/Mecanismos.
15. Material bônus.
16. Definições.

3 TRABALHOS RELACIONADOS

Neste capítulo, será abordado e discutido os trabalhos relacionados. Pontos de semelhanças e diferenças serão relacionados com este trabalho, assim como cada trabalho estará se relacionando com o vigente trabalho. Barczak e Woźniak (2019), Christopoulou e Xinogalos (2017) e Flomén e Gustafsson (2020) são os trabalhos relacionados.

3.1 COMPARATIVE STUDY ON GAME ENGINES Barczak e Woźniak (2019)

O trabalho de Barczak e Woźniak (2019) realizam uma análise comparativa entre os motores de jogo *Unity*[10] e *Unreal*[11] e *CryEngine*[1], utilizando capacidades técnicas e fatores que influenciam na popularidade dos motores de jogos como principais critérios para avaliação. Resumidamente, os autores têm como objetivo mostrar os pontos fortes e fracos de cada engine, além de apresentar suas diferentes características.

Para realizar a análise comparativa os autores se basearam nas seguintes características: multiplataforma, licenças, linguagens de programação, ferramentas de desenvolvimento, documentação, facilidade de aprendizado, multifuncionalidade, inteligência artificial, motor de física, vídeo, módulos de rede, *scripts*, som, *level design*, efeitos gráficos, modelagem, texturização, Bibliotecas e *plugins*, lojas/mercados, animações, interfaces de usuário e desenvolver um jogo protótipo nos três motores de jogo para realizar testes de desempenho.

O trabalho destaca o motor de jogo *Unity*[10] como a melhor escolha para desenvolvedores iniciantes, por possuir muitos cursos, templates prontos e uma documentação bem escrita. Por fim, os autores ressaltam que *Unreal*[11] em conjunto com o uso da ferramenta *BluePrints*[2] torna-se uma boa opção para usuários que não possuem habilidades com programação. Por fim, o trabalho relata que a *CryEngine*[1] permite criar jogos com gráficos com altíssima qualidade, além de se sobressair nos testes de desempenho com mais eficiência.

O trabalho faz uma comparação sobre diversos pontos a respeito dos motores de jogos, entretanto, mesmo com o desenvolvimento de um protótipo para os testes de desempenho, o presente trabalho se diferencia propondo o desenvolvimento de um jogo web que será utilizado como parâmetro para analisar os motores de jogo.

[1] https://www.cryengine.com/
[2] https://docs.unrealengine.com/5.0/en-US/blueprints-visual-scripting-in-unreal-engine/

3.2 OVERVIEW AND COMPARATIVE ANALYSIS OF GAME ENGINES FOR DESKTOP AND MOBILE DEVICES Christopoulou e Xinogalos (2017)

Em Christopoulou e Xinogalos (2017) é feita a comparação de diversos motores de jogo como: *Game Maker*[3], *JMonkey*[4], *Marmalade*[5], *Ogre3D*[6], *Shiva*[7], *Sio2*[8], *Turbulenz*[9], *Unity*[10] e *Unreal*[11], elencando suas principais características e suas respectivas capacidades com a finalidade de auxiliar o usuário a escolher o motor de jogo com base em suas necessidades.

O trabalho utiliza diversas métricas agrupadas em tópicos, as quais são: fidelidade audiovisual, fidelidade funcional, composição, kits de ferramentas para desenvolvedores, acessibilidade, rede, recursos de desenvolvimento, plataformas de implantação, que são utilizadas para avaliar os motores de jogo.

A metodologia do trabalho consiste em realizar uma análise qualitativa em todas os motores de jogos selecionados, após a análise, o trabalho seleciona dois motores de jogos que se sobressaíram sob os demais para uma nova comparação, cujo a metodologia consiste em desenvolver dois jogos de tiro seguindo as documentações e os tutoriais em seus respectivos sites oficiais.

O trabalho relata que os motores de jogo *Unity*[10] e *unreal*[11] são melhores com relação às concorrentes, sendo mais poderosas e apresentando bom desempenho nas métricas utilizadas na avaliação.

Por fim, Christopoulou e Xinogalos (2017) conclui que não é possível determinar qual dos motores de jogo se sobressai, alegando que cada uma possui suas vantagens e desvantagens e que a escolha dependerá do perfil do usuário.

O trabalho se diferencia do presente trabalho com relação a metodologia, já que o jogo a ser desenvolvido neste trabalho é um jogo 2D produzido pelo autor, no qual será especificado o documento de design de jogo com a finalidade de garantir que o mesmo jogo será desenvolvido nos motores de jogos a serem avaliados.

[3] https://gamemaker.io/
[4] https://jmonkeyengine.org/
[5] https://www.marmaladegamestudio.com/
[6] https://www.ogre3d.org/
[7] https://shiva-engine.com/
[8] http://www.sio2interactive.com/index.php
[9] http://biz.turbulenz.com/home
[10] https://unity.com/
[11] https://www.unrealengine.com/

3.3 GAME DEVELOPER EXPERIENCE A COGNITIVE TASK ANALYSIS WITH DIFFERENT GAME ENGINES Flomén e Gustafsson (2020)

O trabalho de Flomén e Gustafsson (2020) têm como objetivo identificar como a experiência ao desenvolver jogos pode ser afetada ao selecionar um motor de jogo específico, para isso os autores propuseram que duas equipes desenvolvessem o mesmo jogo em motores de jogos diferentes *unity*[10] e *godot*[12], .

As duas equipes foram submetidas a seis atividades diferentes que consistem em: configuração do ambiente, criar o cenário do jogo, desenvolver a movimentação do jogador, adicionar física e colisões, permitir interação do jogador com objetos e por último os efeitos sonoros, além disso durante o desenvolvimento, os participantes foram submetidos a sub-tarefas que consistiam em realizar as atividades de ambas a equipes ao mesmo tempo, paralelo a isso, também foi realizado a tarefa de pensamento em voz alta[13] e ao final da atividade cada participante foi submetido a um questionário elaborado pelo autor sobre a atividade em que foram submetidos. Ao final das atividades, todas as equipes participaram de um questionário de avaliação de usabilidade de software de acordo com *SUS*[14].

O trabalho utiliza o tempo de duração que cada equipe levou para concluir as atividades e ambos questionários como métricas para avaliar os motores de jogos. Com base nisso, os autores concluíram que *Godot*[12] é um bom motor de jogos para iniciantes que possuem acompanhamento de um professor, entretanto à medida que as tarefas vão ficando mais complexas, *Unity*[10] passa a ser a melhor escolha, por possuir um material mais acessível disponibilizado pela própria comunidade.

Diferente do que foi apresentado em Flomén e Gustafsson (2020), o presente trabalho se diferencia pelas métricas em que os motores de jogos foram avaliadas, pois o intuito do trabalho atual é utilizar métricas quantitativas, tais como: uso de processador, uso de memória ram, uso de rede, uso de placa gráfica, uso de disco rígido; e métricas qualitativas: linguagem de programação, paradigma, quantidades de plataformas suportadas, documentação, licença, código livre e funcionalidades presentes .

[12] https://godotengine.org/

[13] O método pensar em voz alta consiste em que o usuário é requisitado a falar seus pensamentos em voz alta durante a solução de um problema e a execução de uma atividade (LEHNHART *et al.*, 2019).

[14] https://measuringu.com/sus/

3.4 COMPARATIVO ENTRE OS TRABALHOS RELACIONADOS COM O PRESENTE TRABALHO

O comparativo entre o presente trabalho e os trabalhos pesquisados, é apresentado na Tabela 1. Na Tabela, são apresentados os motores de jogos utilizados, métricas analisadas e a finalidade de cada trabalho.

Sobre os motores de jogos utilizados, este trabalho se relaciona com Barczak e Woźniak (2019) utilizando *Unity* e *Unreal*, já Christopoulou e Xinogalos (2017) os motores *Game Maker*, *Unity* e *Unreal* também são utilizados no presente trabalho e por fim Flomén e Gustafsson (2020) também utiliza os motores *Unity* e *Godot*.

Acerca das métricas avaliadas, Barczak e Woźniak (2019) apresenta as seguintes métricas em comum: linguagem de programação, multiplataformas, licenças, ferramentas de desenvolvimento, documentação e desempenho; em Christopoulou e Xinogalos (2017) as métricas utilizadas em ambos projetos são: Kits de ferramentas para desenvolvedores, recursos de desenvolvimento e plataformas de implantação; e Flomén e Gustafsson (2020) não possui métricas em comum com o presente trabalho.

Por último, todos os trabalhos apresentaram finalidades diferentes, entretanto com algumas semelhanças, são elas: Barczak e Woźniak (2019) propõe um comparativo entre três motores de jogos, mas não desenvolve jogos 2D para web; em Christopoulou e Xinogalos (2017) é realizado um comparativo e é desenvolvido jogos semelhantes com base na documentação de seu respectivo motor de jogo, o presente trabalho propõe desenvolver jogos idênticos em todos os motores de jogos com base no documento de design de jogo elabarado pelo autor; e por último, o trabalho de Flomén e Gustafsson (2020) possui finalidade totalmente distinta do que é proposto neste trabalho, tendo como objetivo identificar como a experiência do desenvolvedor pode ser afetada ao selecionar determinado motor de jogo.

Tabela 1 – Comparativo entre os trabalhos relacionados com o presente trabalho

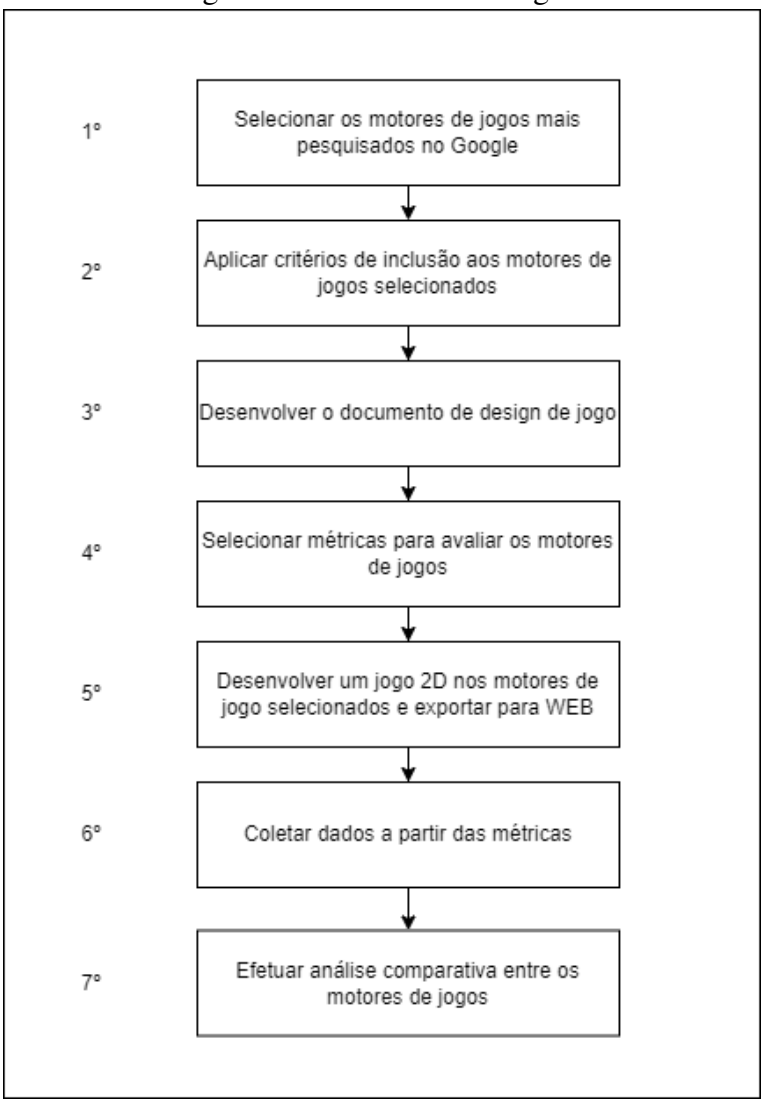
	Motores de jogos Utilizados	**Métricas Avaliadas**	**Finalidade do trabalho**
Presente Trabalho	- Unity - Godot - Game Maker - Unreal - RPG Maker - Construct Blender.	- linguagem de programação - paradigma - plataformas suportadas - documentação - licença - código livre - funcionalidades presentes - uso de processador - uso de memória ram - uso de rede - uso de placa gráfica - uso de disco rígido.	Analisar e selecionar os motores de jogos que possuem o melhor custo benefício para desenvolvimento de jogos 2D web a partir do desenvolvimento de jogos idênticos em ambos motores que será utilizado como parâmetro.
Barczak e Woźniak (2019)	- Unity - Unreal - CryEngine.	- multiplataforma - licenças - linguagens de programação - ferramentas de desenvolvimento - facilidade de aprendizado - multifuncionalidade - inteligência artificial - motor de física - vídeo - som - animações - módulos de rede - desempenho - documentação - *scripts* - *level design* - efeitos gráficos - modelagem - texturização - Bibliotecas - *plugins* - lojas - interfaces de usuário .	Realizar análise comparativa entre três motores de jogos de desenvolvimento de jogos
Christopoulou e Xinogalos (2017)	- GameMaker -JMonkey - Marmalade - Ogre 3D - Shiva - Sio2 - Turbulenz - Unity - Unreal.	- fidelidade audiovisual - fidelidade funcional - composição - kits de ferramentas para desenvolvedores, - acessibilidade - rede, - recursos de desenvolvimento - plataformas de implantação.	Realizar o comparativo entre os motores de jogos selecionadas e desenvolver jogos semelhantes em ambos motores com base na documentação.
Flomén e Gustafsson (2020)	- Unity - Godot.	- tempo de desenvolvimento - questionário a respeito das atividades desenvolvidas - questionário de usabilidade.	Identificar como a experiência ao desenvolver jogos pode ser afetada ao selecionar um motor de jogo específico

Fonte: Autor 2022

4 METODOLOGIA

Nesta seção será abordado todas as etapas do processo metodológico que o presente trabalho irá empregar. A Figura 3 apresenta os seguintes passos para execução do trabalho: Selecionar os motores de jogos mais pesquisados no *Google*; Aplicar critérios de inclusão aos motores de jogos selecionados; Desenvolver o documento de design de jogo; Selecionar métricas para avaliar os motores de jogos; Desenvolver um jogo 2D nos motores de jogo selecionados e exportar para WEB;Coletar dados a partir das métricas e por último Efetuar análise comparativa entre os motores de jogo.

Figura 3 – Passos metodológicos



Fonte: Elaborada pelo autor

4.1 Selecionando os motores de jogo mais pesquisados no Google

Na primeira etapa deste trabalho realizou-se uma pesquisa utilizando o termo de busca *Game Engines* no site *google trends*, com o objetivo de obter os motores de jogos mais pesquisados. A pesquisa foi filtrada para motores de jogos mais pesquisadas no Brasil nos últimos 12 meses com o objetivo de obter resultados de quais motores de jogos o público brasileiro possui mais interesse.

4.2 Aplicando critérios de inclusão aos motores de jogos selecionados

Nesta etapa foram descartadas os motores de jogos que não satisfaziam os critérios de inclusão. Os critérios de inclusão deste trabalho consiste em:

1. O motor de jogo exporta jogos para web?
2. O motor de jogo possui licença básica gratuita?
3. O motor de jogo atende aos requisitos mínimo?

Os requisitos mínimos em que a pesquisa foi realizada consiste em: um notebook Dell Inspiron 5537 Intel(R) Core(TM) i7-4500U CPU @ 1.80GHZ (4 CPUs) 2.40GHZ, 8GB Memória RAM, sistema operacional Windows 10 pro 64bits, DirectX 12 e 1TB de armazenamento HD.

4.3 Desenvolver o documento de design de jogo

A partir desta etapa foi gerado o artefato de documento de design de jogo a partir do *template* proposto por Hira *et al.* (2016), este documento tem como objetivo levantar os requisitos do jogo, servindo como base teórica de como o jogo deve ser desenvolvido em ambos motores de jogo.

4.4 Selecionar métricas para avaliar os motores de jogos

Nesta etapa, as métricas foram selecionadas a partir da disposição em que ambos motores de jogos possuíam a disposição de serem avaliados, para uma coleta de dados mais fiel e abrangente, foram coletados dados a respeito do código fonte gerado durante o desenvolvimento dos jogos e coletado dados a partir de cenários de testes que tem como finalidade simular a jogatina de um usuário comum.

4.5 Desenvolver um jogo 2D nos motores de jogo selecionados e exportar para *WEB*

A etapa de desenvolvimento do jogo consistiu em analisar o documento de design de jogo e a partir dele desenvolver o jogo nos motores de jogos selecionados. Para o desenvolvimento do jogo se fez necessário consultar a documentação oficial dos motores com o objetivo de entender como a ideia proposta no documento de design de jogo seria desenvolvida com as tecnologias escolhidas. Além disso, também foram consultadas canais no *youtube*, fórums, *blogs* e comunidades de desenvolvimento para sanar dúvidas a respeito de impedimentos que surgiam durante o desenvolvimento. Por fim, o jogo foi exportado para WEB e foi feita a publicação online na plataforma *itch.io*.

4.6 Coletar dados a partir das métricas

Nesta etapa, em consequência da dificuldade de encontrar uma aplicação que coletasse métricas durante os cenários de teste, surgiu a necessidade de desenvolver um *script* em *python* com o objetivo de coletar os dados de memória RAM e CPU a partir dos cenários pré-estabelecidos.

4.7 Efetuar análise comparativa entre os motores de jogos

A análise comparativa dos dados coletados consistiu em realizar a coleta de dados e exportá-los para uma planilha do *Excel*. No primeiro momento foi realizado uma análise de qual motor de jogo teve maior grau de dificuldade no processo de desenvolvimento, em seguida será avaliado os dados coletados dos cenários de testes para verificar qual motor exportou jogo para *WEB* com melhor desempenho.

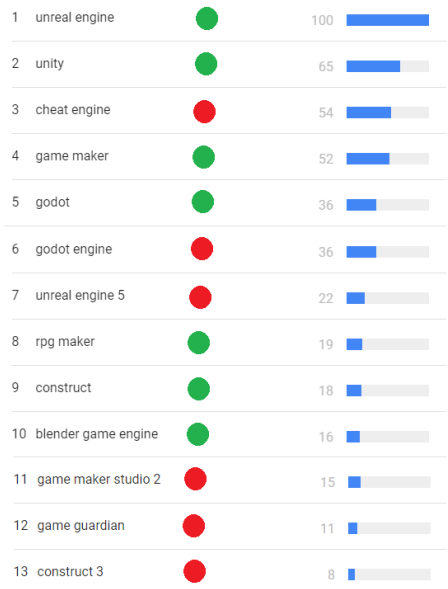
5 RESULTADOS

Os resultados obtidos deste trabalho foram coletados no período de junho de 2022 a novembro de 2022. Nesta seção é apresentado todos os resultados obtidos dentre os passos metodológicos supracitados.

5.1 Selecionando os motores de jogos mais pesquisados no Google

Este trabalho selecionou os motores de jogos mais pesquisados no Brasil nos últimos 12 meses. Conforme apresentado na Figura 4, o presente trabalho selecionou os seguintes motores de jogo (Pontuadas na cor verde os resultados da pesquisa que são motores de jogos): *UNREAL*[11],*Unity* [10], *GAME MAKER*[3],*Godot* [12], *RPG Maker*[1], *Construct*[2] e *Blender*[3]. Os resultados da pesquisa que estão pontuados na cor vermelha foram excluídos do presente trabalho por repetição (*godot engine*[12], *unreal engine 5*[11], *game maker studio 2*[3] e *construct 3*[2]) e por não serem motores de jogos (*cheat engine*[4] e *game guardian*[5]).

Figura 4 – Resultado da pesquisa do *Google Trends* sobre os motores de jogos mais utilizados



Fonte: *Google Trends*. Link: https://trends.google.com.br/trends/explore?geo=BR&q=Game%20Engine/. Acesso em: 25/06/2022

[1] https://www.rpgmakerweb.com/
[2] https://www.construct.net/en
[3] https://www.blender.org/
[4] https://www.cheatengine.org/
[5] https://gameguardian.net/forum/files/file/2-gameguardian/

5.2 Aplicação dos critérios de inclusão

Nesta seção é apresentado os resultados obtidos a partir dos critérios de inclusão.

5.2.1 O motor exporta jogos para web?

A Tabela 2 apresenta os motores de jogos que exportam os jogos desenvolvidos em seus ambientes para plataforma web. Todos os motores de jogos apresentaram portabilidade para desenvolvimento de jogos Web. *Unreal*[11], *Game Maker*[3] e *Construct*[2] possuem suporte a desenvolvimento de jogos para web com *HTML5*, já *Unity* [10] e *Godot* [12] utilizam *WebGL* e *HTML5*, o motor de jogo *RPG Maker*[1] utiliza *node.js*[6] e *browsersync*[7], por fim, para construir jogos web com *Blender* se faz necessário utilizar o *framework* chamado *Blend4Web*[8].

Tabela 2 – O motor exporta jogos para web?

	Unreal	**Unity**	**Game Maker**	**Godot**	**RPG Maker**	**Construct**	**Blender**
Exporta jogos para web?	sim	sim	sim	sim	sim	sim	sim

Fonte: Autor 2022

5.2.2 Possui licença básica gratuita?

A Tabela 3 mostra os motores de jogos que possuem licença básica gratuita, sendo eles: *Unreal*[11], *Unity*[10], *Godot*[12] e *Blender*[3] possuem licença básica gratuita com recursos ilimitados, já *Game Maker*[3] dispõe de licença básica gratuita limitando o usuário a exportar seus jogos somente para a plataforma web via navegador *Opera GX*[9] na plataforma *GX.games*[10], o motor *Construct* contém licença básica gratuita com limitações em seus recursos, como por exemplo: quantidade máxima de eventos limitada a cinquenta, quantidade máxima de efeitos especiais limitada a dois e entre outras restrições. Por fim, o motor de jogo *RPG Maker*[1] não possui licença básica gratuita, havendo apenas um teste de trinta dias com recursos limitados.

6 https://nodejs.org/en/
7 https://www.browsersync.io/
8 https://www.blend4web.com/en/downloads/
9 https://www.opera.com/pt-br/gx
10 https://gxc.gg/pt-br/

Tabela 3 – Possui licença básica gratuita?

	Unreal	**Unity**	**Game Maker**	**Godot**	**RPG Maker**	**Construct**	**Blender**
Possui licença básica grauita?	sim	sim	sim	sim	não	sim	sim

Fonte: Autor 2022

5.2.3 Atende aos requisitos mínimos?

O motor *Unreal*[11] recomenda que o usuário possua as seguintes especificações: sistema operacional *Windows* 10 64-bit, processador *Quad-core Intel* ou *AMD* com *clock* de 2.5 GHz ou superior e placa gráfica compatível com *Directx* 11 ou *DirectX* 12.

Unity [10] informa que os requisitos mínimos para rodar o motor irá depender da complexidade do projeto e apresenta as seguintes características mínimas: sistema operacional *Windows* 7 (SP1+), *Windows* 10 e *Windows* 11, versões de 64 bits, processador com arquitetura X64 com suporte a conjunto de instruções SSE2, placa gráfica com suporte a *Directx* 10, 11 e 12 e os devidos *Drivers* instalados.

Game Maker[3] possui os seguintes requisitos mínimos: sistema operacional *Windows 7* SP1 ou superior, processador *Dual Core*, placa gráfica integrada compatível com OpenGL 4.

O motor de jogos *Godot* [12] requer as seguintes configurações mínimas: sistema operacional *Windows* 7, processador *Intel Core 2 Duo* E84000, com 4GB de memória *RAM* e placa gráfica *NVIDIA GeForce* 6200.

RPG Maker[1] requer uma máquina com: sistema operacional *Windows* 7, processador *Intel Core 2 Duo*, 2GB de memória RAM, placa gráfica que suporte *DirectX* 9 e *OpenGL* 4.1.

O motor *Construct*[2] permite que o usuário desenvolva jogos a partir de abas, tais como: *Google Chrome* com versão 61 ou superior, *Firefox* com versão 60 ou mais, *Safari* versão 13 ou superior e *Microsoft Edge* com versão 79 ou mais.

Blender[3] recomenda as seguintes especificações mínimas: processador *Quad Core* com suporte SSE2 64-bit com *clock* de 2GHz, 8GB de memória RAM, placa gráfica com 2GB de RAM e suporte a *OpenGL* 4.3.

Diante do que foi dito, a Tabela 4 mostra que *Unreal*[11] e *Blender*[3] não atendem aos requisitos mínimos de *hardware* e *software*.

Tabela 4 – Atende aos requisitos mínimos?

	Unreal	**Unity**	**Game Maker**	**Godot**	**RPG Maker**	**Construct**	**Blender**
Atende aos requisitos mínimos?	não	sim	sim	sim	não	sim	não

Fonte: Autor 2022

5.2.4 Resultado dos critérios de inclusão e exclusão

A partir dos dados obtidos, os seguintes motores de jogos não foram selecionados para a próxima fase de desenvolvimento, são eles:

Unreal e *Blender*, por possuírem requisitos mínimos próximos ou superiores ao ambiente de pesquisa tendo potencial de acarretar em gargalos não propositais[11] ao decorrer do desenvolvimento.

O motor de jogo *Game Maker*[3] mesmo exportando jogos para web, sua licença básica limita o usuário a utilizar o navegador *Opera GX*[9].

Construct[2] por haver limitações de recursos durante o desenvolvimento em sua licença básica gratuita.

Por fim o motor de jogo *RPG Maker*[1] por não conter uma licença básica gratuita, disponibilizando apenas um teste de trinta dias.

Diante do que foi abordado, os motores de jogos *Unity* [10] e *Godot* [12] foram selecionadas para a próxima fase de desenvolvimento.

5.3 Desenvolvimento do documento de design de jogo

Este trabalho desenvolveu o documento de design de jogo com o objetivo de levantar os requisitos dos jogos a serem desenvolvidos em ambos motores, ressaltando a importância de documentar as principais características que o jogo deve conter. O Documento de design de jogo desenvolvido neste trabalho descreve o jogo intitulado *Os mortos-vivos Quixadá*, cujo a história do jogo baseia-se em uma invasão zumbi ao campus da UFC de Quixadá. O objetivo principal do jogo é sobreviver o maior tempo possível. O Documento de design de jogo deste trabalho encontra-se no APÊNDICE A.

[11] gargalos que ocorrem sem a intervenção de testes de desempenho

5.4 Selecionando métricas para avaliar os motores de jogos

As métricas selecionadas para avaliação dos motores de jogos são divididas em dois grupos: O primeiro grupo de métricas está relacionado ao código fonte gerado durante o desenvolvimento; O segundo grupo é um conjunto de cenários de testes que foram estabelecidos a partir das métricas a serem coletadas.

Para realizar a avaliação, foram selecionadas métricas que estivessem presentes em ambos motores. O primeiro grupo de métricas relacionado ao código fonte consiste em obter os seguintes dados:

1. Tamanho final do arquivo.
2. Quantidade de arquivos de código.
3. Quantidade de linhas.
4. Quantidade de linhas de código.
5. Complexidade do código.

Para a coleta de dados com relação aos jogos desenvolvidos, a partir do fato de que o jogador possa ter diferentes quantidades de memória RAM disponível e diferentes quantidades de abas abertas, criou-se cenários de testes a fim de simular a jogatina um usuário comum. No total, foram especificados 16 cenários para coleta de dados, são eles:

1. Memória RAM disponível: 4GB, 8GB e 12GB.
2. Total de abas abertos: 1, 5 e 10.
3. Jogos a serem testados: Os dois jogos desenvolvidos nos motores *Unity* [10] e *Godot* [12].

Para garantir que cada cenário fosse respeitado, utilizou-se uma Máquina Virtual com sistema operacional Linux, com 80GB de armazenamento, 2 núcleos de processador e utilizando o navegador *mozilla firefox*. Os dados coletados nesses cenários consistem em:

1. Tempo de carregamento de jogo.
2. Uso de CPU mínimo.
3. Uso de CPU máximo.
4. Média do uso de CPU.
5. Uso de memória RAM mínimo.
6. Uso de memória RAM máximo.
7. Média do uso de memória RAM.

5.5 Desenvolvimento do jogo "Os mortos-vivos Quixadá"nos motores Unity e Godot e coleta de dados do código fonte

Antes de iniciar a etapa de desenvolvimento, foi feita uma análise do documento de design de jogo para verificar os requisitos a serem desenvolvidos. A partir disso foram definidas as seguintes funcionalidades: Som de fundo; Tela de menu; Tela de créditos; Tela de Controles; Mapa do jogo; Personagem; Personagens não jogáveis / inimigos; Projétil da arma; Colisão; Pontuação de jogo e fim de jogo.

Para desenvolver o jogo no motor *Godot* [12] utilizou-se a linguagem de programação *GDScript* gerando um total de 8 arquivos, totalizando 188 linhas, sendo elas 149 de código, tendo como tamanho final do jogo exportado 24,467kb. Já o *Unity* [10] utiliza *C#* gerando 10 arquivos de *script*, com um total de 245 linhas, 288 de código e 7,157kb de tamanho final. Além disso o código fonte foi armazenado no site *Github* e os arquivos exportados foram hospedados na plataforma de jogos chamada *itch.io* gerando um link para acessar o jogo de seus respectivos motores: *Unity* [10] [12] [13] e *Godot* [12] [14] [15].

Tabela 5 – Comparativo entre Unity e Godot com relação a desenvolvimento

	Unity	**Godot**
Linguagem de programação	C#	GDScript
Tamanho final do jogo exportado	7,157kb	24,467kb
Quantidade de arquivos	10	8
Quantidade de linhas	345	188
Quantidade de linha de códigos	288	149

Fonte: Autor 2022

Conforme apresentado na Tabela 5, é possível observar que o motor de jogo *Unity* [10] gerou uma quantidade maior de linhas de código, uma vez que a linguagem de programação *C#* é mais verbosa que *GDScript*. Entretanto, utilizando o motor *Unity* [10] o tamanho final do jogo exportado para *WEB* foi em torno de quatro vezes menor do que foi gerado em *Godot* [12].

A Tabela 6 apresenta um levantamento de dados a respeito da complexidade ciclomática dos motores de jogos. A Tabela mostra que o motor *Unity* [10] possui maior complexidade nos arquivos *Enemy*, *Game*, *Player*, *EnemyHealthBar* e *CameraFollow*. Já Godot apresentou complexidade superior nos arquivos *Bullet* e *Menu* e por fim, os arquivos *GameOver*, *PlayerStatus* e

[12] https://srpantoja.itch.io/mortos-vivos-quixada-unity
[13] https://github.com/srpantoja/mortos-vivos-quixada-unity
[14] https://srpantoja.itch.io/mortos-vivos-quixada-godot
[15] https://github.com/srpantoja/mortos-vivos-quixada-godot

BackgroundAudio apresentaram a mesma pontuação de complexidade.

Tabela 6 – Comparativo de complexidade ciclomática entre os motores

Arquivos	Complexidade Unity	Complexidade Godot
Bullet	4	6
Enemy	11	6
Game	4	3
GameOver	2	2
PlayerStatus	3	3
Menu	1	4
Player	17	14
EnemyHealthBar	3	0
CameraFollow	1	0
BackgroundAudio	2	2
Total	48	38

Fonte: Autor 2022

A partir da Tabela é possível notar que há uma diferença considerável de 10 pontos entre os motores. Isso ocorre pois existem funcionalidades presentes no *Godot* [12] que simplificam o desenvolvimento, fazendo com que em alguns casos não haja a necessidade de programar, sendo observado na ausência dos arquivos *EnemyHealthBar* e *CameraFollow*.

5.6 Coletar dados a partir das métricas

Para realizar o desenvolvimento do *script* em *python* [16] que coleta as métricas de uso da CPU e memória RAM, foram utilizadas as bibliotecas *psutil* [17] e *selenium webdriver* [18]. A biblioteca *psutil* [17] foi utilizada para coletar em porcentagem o uso de memória RAM e CPU, já o *selenium webdriver* [18] foi utilizado para criar um ambiente controlado seguindo os cenários pré-estabelecidos. O código produzido para este trabalho pode ser encontrado na plataforma *Github* [19].

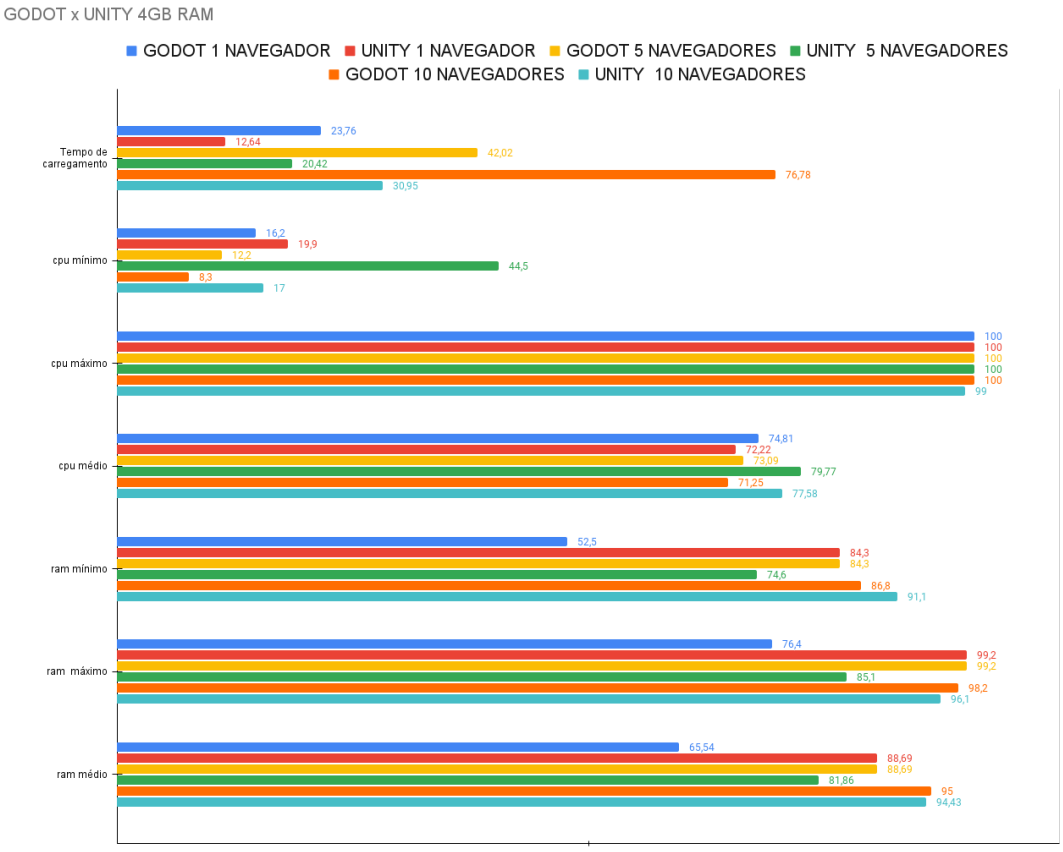
Ao executar o código, primeiramente é preciso informar qual motor de jogo será testado, após isso, o código irá abrir o navegador e redirecionar o usuário para a URL do motor selecionado. Paralelo a isso, é feita a coleta do tempo de inicialização de jogo de forma manual, cujo o cronômetro se inicia assim que o navegador é aberto e se encerra quando o jogo está completamente carregado. O próximo passo é jogar o jogo por dois minutos, enquanto

[16] https://www.python.org
[17] https://psutil.readthedocs.io/en/latest/
[18] https://www.selenium.dev/documentation/webdriver/
[19] https://github.com/srpantoja/pc$_{m}onitor$

isso, o *script* coleta os dados a respeito do uso de CPU e memória RAM. Ao final do tempo cronometrado de dois minutos jogando, é necessário fechar o navegador para que a coleta seja encerrada.

A Figura 5 apresenta os dados coletados no ambiente com 4GB de memória RAM disponível. Com relação ao tempo de carregamento, é possível observar que o motor *Unity* [10] apresentou maior velocidade com relação ao *Godot* [12] quando comparados com a quantidade de abas abertas.

Figura 5 – Cenários em ambiente com 4GB de memória RAM



Fonte: Autor 2022

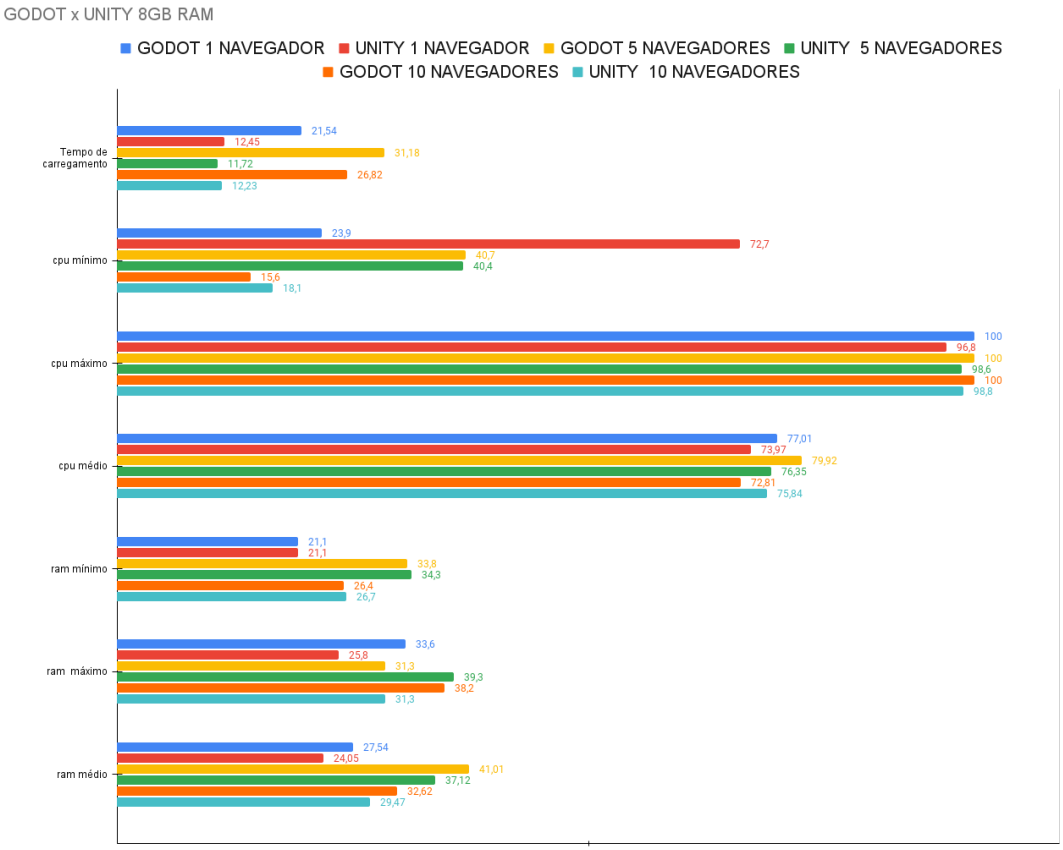
Já o uso mínimo de CPU, novamente o motor *Unity* [10] apresenta valores superiores ao *Godot* [12], entretanto, a respeito do uso máximo de CPU, ambos motores de jogos obtiveram valores próximo a 100 porcento, sendo assim, um empate técnico. Por fim, a média de uso de CPU também apresentou valores próximos entre os motores, porém o motor *Unity* [10] apresentou médias maiores nos cenários em que havia 5 e 10 abas abertas.

Acerca das métricas coletadas referentes ao uso de memória RAM, o motor *Unity* [10] apresentou porcentagem maior no tocante uso de RAM mínimo. Entretanto, o motor *Godot* [12]

obteve valores maiores nos cenários com 5 e 10 abas, sendo eles bem próximo de 100 porcento. Com relação ao uso médio de memória RAM o motor *Godot* [12] se sobressai no primeiro cenário com uma aba e *Unity* [10] apresenta melhores médias nos cenários restantes.

A partir da Figura 6, cujo cenário refere-se ao ambiente com 8GB de RAM, é possível observar que o tempo de carregamento no motor *Unity* [10] continuou superior aos do *Godot* [12], além de apresentar valores bem próximos entre si, independente da quantidade de abas abertas.

Figura 6 – Cenários em ambiente com 8GB de memória RAM



Fonte: Autor 2022

Com relação ao uso mínimo, o motor *Godot* [12] apresentou valores com porcentagem inferiores, dando destaque principalmente ao cenário com um único navegador, havendo uma diferença de quase 50 pontos percentuais. Entretanto, o motor *Unity* [10] sobressaiu nos valores de uso máximo de CPU, apresentando valores abaixos em todos os cenários, *Godot* [12] por sua vez, apresentou 100 porcento de uso máximo de CPU em todos os cenários. A respeito do uso médio de CPU, ambos motores obtiveram valores bastante próximos entre si, ressaltando que o motor *Unity* [10] obteve valores menores nos cenários com 1 e 5 abas abertas.

Por ter mais memória RAM disponível, os valores referentes ao uso de memória RAM caíram drasticamente. No tocante do uso mínimo de RAM, houve um empate técnico em todos os cenários, com uma pequena vantagem por parte do motor *Godot* [12] que obteve uma diferença de 0,5 pontos percentuais no cenário com 5 abas e uma diferença de 0,3 pontos percentuais com 10 abas. Porém, esse cenário não se matem no uso máximo de memória RAM, tendo o *Unity* [10] com menor valor nos cenários com 1 e 10 abas. Por fim, com relação ao uso médio de memória RAM, o motor de jogo *Unity* [10] foi superior em todos os cenários, apresentando uma diferença considerável de 3,49 pontos percentuais no cenário com uma aba, 3,89% com 5 abas e 3,15% com 10 abas.

Os dados coletados referentes ao ambiente com 12GB estão apresentados na Figura 7. Referente ao tempo de carregamento, o motor *Unity* [10] continua superior e bastante estável. O uso mínimo de CPU apresenta um empate técnico entre os motores, com uma leve vantagem de 0,3% para *Godot* [12] no cenário com uma aba. Porém, nos demais cenários *Unity* [10] apresentou uma ligeira vantagem, principalmente no cenário com 10 abas, chegando a ter uma vantagem em torno de 30 pontos percentuais a menos que *Godot* [12].

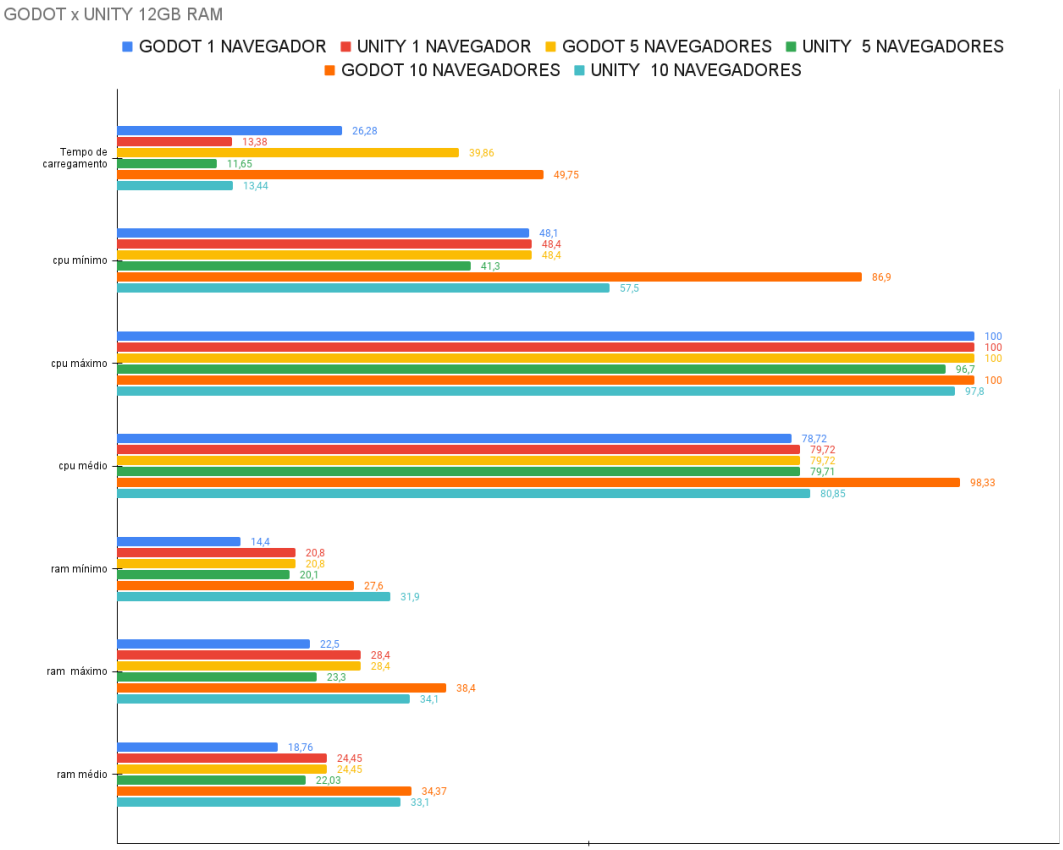
No tocante do uso máximo houve empate técnico no cenário com uma aba, entretanto nos demais cenários houve uma pequena vantagem para *Unity* [10]. Por fim, ambos motores apresentaram o uso médio de CPU bem próximos entre si nos cenários com 1 e 5 abas, mas com uma grande vantagem para *Unity* [10] no cenário com 10 abas.

Com relação ao uso mínimo de memória RAM, o motor de jogo *Godot* [12] obteve uma pequena vantagem nos cenários com 1 e 10 abas, porém no uso máximo, o motor de jogo *Unity* [10] apresenta valores menores nos cenários com 5 e 10 abas. Por último, o uso médio de memória RAM no cenário com uma aba o motor de jogo *Godot* [12] atingiu melhores resultados, entretanto deixou a desejar nos cenários seguintes.

5.7 Análise comparativa dos dados coletados

A partir dos dados coletados, foi realizada uma coleta de dados em dois grupos distintos de métricas, sendo eles: dados coletados durante o processo de desenvolvimento a respeito do código fonte e cenários de testes para observar qual motor de jogo apresentou melhor desempenho. Com relação aos dados de código fonte, é possível observar que o motor de jogo *Godot* [12] se sobressaiu por ter menor complexidade ciclomática, além de precisar escrever menos linhas de código para desenvolver o mesmo jogo.

Figura 7 – Cenários em ambiente com 12GB de memória RAM



Fonte: Autor 2022

Fato esse corrobora com a ideia de que o motor *Godot* [12] é a ferramenta ideal para usuários iniciantes em desenvolvimento de jogos. Entretanto, ao exportar o jogo para WEB, *Unity* [10] apresenta uma compactação avassaladora com relação ao seu concorrente, tendo uma diferença de tamanho em torno de quatro vezes menor. No tocante de desenvolvimento de jogos simples e com fins de aprendizagem, o tamanho final do arquivo torna-se irrelevante, porém, é importante ressaltar que a medida que o projeto vai crescendo o tamanho final também cresce.

A respeito dos dados coletados durante o cenário de teste, nota-se que em todos os cenários de tempo de carregamento o motor *Unity* [10] se saiu melhor, realidade essa fundamentada a partir da diferença do tamanho final do arquivo. Quanto ao uso de CPU mínimo, o motor *Godot* [12] apresentou em sua grande maioria valores menores, entretanto, por sua vez, também foi o responsável por valores altíssimos no uso máximo de CPU, chegando a ter 100% em diversos cenários.

É importante destacar que o uso máximo de CPU próximos a cem porcento pode acarretar em gargalos e travamentos durante o jogo. Por outro lado, *Unity* [10] apresenta melhores

resultados no uso médio de CPU, mantendo-se bastante constante e com valores próximos entre si, independe da quantidade de memória RAM disponível, diferentemente visto em *Godot* [12] que obteve valores médios bastante variáveis.

No tocante do uso memória RAM, o motor *Godot* [12] apresentou valores menores em cenários em que havia somente uma aba aberto, porém não se manteve nos demais cenários. *Unity* [10] por sua vez, em cenários com 5 e 10 abas abertas, apresentou um constante crescimento ao uso de memória RAM, mas sempre abaixo de *Godot* [12]. Com relação as métricas de desempenho, o motor *Unity* [10] apresentou melhores resultados com valores constantes e mantendo-se melhor em boa parte dos cenários, tendo assim um melhor desempenho comparado com *Godot* [12].

6 CONCLUSÕES

Mediante aos dados obtidos, fica notório que há diferenças significantes entre os motores de jogos *Unity* [10] e *Godot* [12]. Ambos motores possuem características únicas para públicos diferentes. *Godot* [12] por sua vez, com quantidade menores de linhas de código em conjunto com sua baixa complexidade durante o desenvolvimento, facilita o uso do motor para usuários iniciantes em desenvolvimento de jogos.

Em contraposição a isso, o motor *Unity* [10] apresentou maior complexidade durante a produção do jogo, com uma linguagem de programação mais verbosa (quando comparada com *Godot* [12] e com maior quantidade de arquivos e linhas de códigos geradas. Com base nisso, *Unity* [10] torna-se mais recomendável para usuários que já possuem noção básica em programação na linguagem C#.

Entretanto, durante os testes dos jogos, nota-se que o motor *Unity* [10] apresentou vantagens com relação ao desempenho durante o jogo, mantendo-se bastante estável em todos os cenários. Diante disso, visando a experiência do jogador, recomenda-se utilizar *Unity* [10] para jogos com funcionalidades complexas e pesadas.

Diante do conteúdo supracitado, conclui-se que o motor *Godot* [12] é recomendável para usuários iniciantes em desenvolvimento de jogos e *Unity* [10] por apresentar bons testes de desempenho, sendo bastante indicado para usuários que buscam desenvolver jogos pensando na experiência do jogador.

REFERÊNCIAS

AMÉLIO, C. de O. **A indústria e o mercado de jogos digitais no brasil**. XVII SBGames, Foz do Iguaçu, Paraná, Brasil, p. 1497–1506, 2018.

BARCZAK, A. M.; WOŹNIAK, H. **Comparative study on game engines**. [S.l: s.n], n. 1–2, 2019.

BECKER, K. **Direito autoral e jogos digitais:** um estudo acerca da regulamentação do direito autoral dos jogos digitais no direito brasileiro. [S.l: s.n], 2018.

CANALTECH. **O que são e como funcionam os jogos NFT**. [S.l], 2022. Disponível em: https://canaltech.com.br/games/o-que-sao-e-como-funcionam-os-jogos-nft/. Acesso em: 10 nov. 2022.

CAVALCANTE, C. H. L.; PEREIRA, M. L. A. **Comparativo entre Game Engines como Etapa Inicial para o Desenvolvimento de um Jogo de Educação Financeira**. [S.l: s.n], 2018.

CHRISTOPOULOU, E.; XINOGALOS, S. **Overview and comparative analysis of game engines for desktop and mobile devices**. [S.l: s.n], 2017.

COINTELEGRAPH. **Web3 deve favorecer economia de jogos play-to-earn e o crescimento deste mercado no Brasil**. [S.l], 2022. Disponível em: https://bit.ly/web3-should-favor-play-to-earn-games. Acesso em: 10 nov. 2022.

COSTA, D. P.; GOMES, F. F. B.; DUARTE, R. **Estudo comparativo entre as game engines unity e ogre**. Revista Computação Aplicada-UNG-Ser, v. 5, n. 1, p. 18–25, 2019.

FLOMÉN, R.; GUSTAFSSON, M. **Game developer experience:** a cognitive task analysis with different game engines. [S.l: s.n], 2020.

GODOT, E. **Godot Engine Documentation**. [S.l], 2022. Disponível em: https://docs.godotengine.org/en/stable/about/introduction.html. Acesso em: 26 jun. 2022.

HIRA, W. K.; MARINHO, M. V. P.; PEREIRA, F. B.; JR, A. B. **Criação de um modelo conceitual para Documentação de Game Design**. Proceedings of SBGames, p. 329–336, 2016.

LEHNHART, E. D. R.; LÖBLER, M. L.; TAGLIAPIETRA, R. D. Discussão e aplicação do protocolo think aloud em pesquisas sobre processo decisório. **Revista Alcance**, v. 26, n. 1, p. 13–29, 2019.

LEMES, D. d. O. *et al.* **Games Independentes:** fundamentos metodológicos para criação, planejamento e desenvolvimento de jogos digitais. Pontifícia Universidade Católica de São Paulo, 2009.

MOTTA, R. L.; JUNIOR, J. T. **Short game design document (SGDD)**. Anais do XII Simpósio Brasileiro de Jogos e Entretenimento Digital, p. 115–121, 2013.

MOZILLA, D. **Introdução ao desenvolvimento de jogos para a Web**. [S.l], 2022. Disponível em: https://developer.mozilla.org/pt-BR/docs/Games/Introduction. Acesso em: 10 nov. 2022.

NGUYEN, T. T. **Introduction to 2D game development with Unity**. [S.l: s.n], 2021.

SILVA, A. V. d. S. **Implementação de um sistema de componentes para um motor de jogos**. Mossoró: Universidade Federal Rural do Semi-Árido, 2020.

SILVA, G. A. da; RIBEIRO, M. W. de S. **Development of Non-Player Character with Believable Behavior:** a systematic literature review. Anais Estendidos do XX Simpósio Brasileiro de Jogos e Entretenimento Digital, p. 319–323, 2021.

SILVA, M. P. R.; COSTA, P. D. P.; PRAMPERO, P. S.; FIGUEIREDO, V. **Jogos Digitais:** definições, classificações e avaliação. Campinas: Universidade Estadual de Campinas-UNICAMP, 2009.

SOUZA, L. José Barbosa de M. **Multimídia como alternativa para documentação no desenvolvimento de jogos digitais**. Universidade Federal de Pernambuco, 2011.

UNITY. **Unity Our Company**. [S.l], 2022. Disponível em: https://unity.com/pt/our-company. Acesso em: 18 jun. 2022.

APÊNDICE A – DOCUMENTO DE DESIGN DE JOGO: OS MORTOS-VIVOS QUIXADÁ

A.1 Sumário

A.2 Visão Geral/Resumo

Quixadá está passando por uma endemia apocalíptica cuja doença é responsável por transformar as pessoas em zumbis. Você é um segurança na Universidade Federal do Ceará campus Quixadá e seu objetivo seu objetivo é não permitir a entrada de zumbis em massa para que um caos não se instaure na universidade, você possui dez minutos até que a ajuda chegue.

A.3 Visão Geral da História

Quixadá se encontra em uma endemia apocalíptica cuja doença é responsável por converter os seres vivos em infectados carniceiros. A primeira vítima foi uma jovem adolescente de 18 anos chamada Maria que estava com seus amigos em uma trilha aos arredores da pedra da galinha choca, até que certo momento Maria encontra e pega com suas mãos uma pedra brilhante próximo a um rio de água parada. Embaixo da pedra vivia uma colônia de parasitas que controlam os seres vivos após a infecção. Estes parasitas se alojam na cabeça da vítima e se alimentam de seu cérebro. Após pegar na pedra, Maria sente fortes tonturas e começa a convulsionar no meio da estrada da trilha, seus amigos sentem-se preocupados e correm até ela para ajudar. Ao chegar no local, os amigos de Maria percebem que seu corpo estava frio, edemaciado e cianótico e na tentativa de salvá-la, o grupo que estava com ela a leva em direção ao início da trilha para que possam levá-la à UPA de carro. Após chegar a UPA, Maria acorda bastante agressiva e ataca os profissionais do local desferindo chutes, mordias e arranhões como medida protetiva a médica plantonista seda Maria e coloca na Zona de observação pois acreditava ser apenas um surto psicótico fruto de um possível uso de drogas, o que não se esperava era o fato dos profissionais terem sido feridos por Maria durante o ataque o que provocou minutos depois uma série de convulsões e sintomatologias similares a dela. Em apenas 8 horas 70% da população de Quixadá estava infectada. Paralelo a isso você estava tranquilo na universidade federal do ceará campus quixadá descansando a barriga após o almoço até que avistou a primeira onda de zumbis, rapidamente você se prepara e chama por ajuda. seu objetivo é não permitir a entrada de zumbis em massa para que um caos não se instaure na universidade.

A.4 Inteligência artificial

Pato: este zumbi é o mais comum e compõe 70% da população de zumbi em Quixadá, possui movimentação lenta e possui um nível de dificuldade fácil.

1. zumbi nível 1: Pato.
    a) dano: 1
    b) vida: 3
    c) velocidade: 2

A.5 Progressão do Jogo

Duração da partida: O jogo tem duração infinita, tendo sua dificuldade aumentada a cada nível adquirido.

Subir de nível: o jogador deve adquirir a pontuação necessária para passar de nível, o cálculo efetuado para definir a próxima pontuação necessária a ser adquirida para subir de nível é: **PRÓXIMO NÍVEL = (100 * NÍVEL ATUAL) * 1.5.**

Adquirir pontos: O jogador adquire 10 pontos ao matar um zumbi.

A.6 Arte e Vídeo

Os personagens do jogo foram retirados do site itch.io, sendo eles:

1. zombie game sprite sheet: https://puszke.itch.io/zombie-game-sprite-sheet
2. medieval weapons pack: https://pixelhole.itch.io/medieval-weapons-pack

A.7 Música

As músicas e efeitos sonoros foram retirados do site itch.io, sendo eles:

1. https://darkworldaudio.itch.io/sound-effects-survival-i
2. https://leaflegion.itch.io/leaflegion-zombie-game-music-1
3. https://magic-eyes.itch.io/apocalypse-city-music

A.8 Interface

As interfaces e os tilesets para construir o jogo são respectivamente feitos por maxparata e szadiart. As interfaces descritas abaixo são protótipos de baixa fidelidade.

1. interface - GOREUI : https://maxparata.itch.io/goreui
2. tilesets: postapo lands: https://szadiart.itch.io/postapo-lands-demo

Figura 8 – Tela de menu inicial

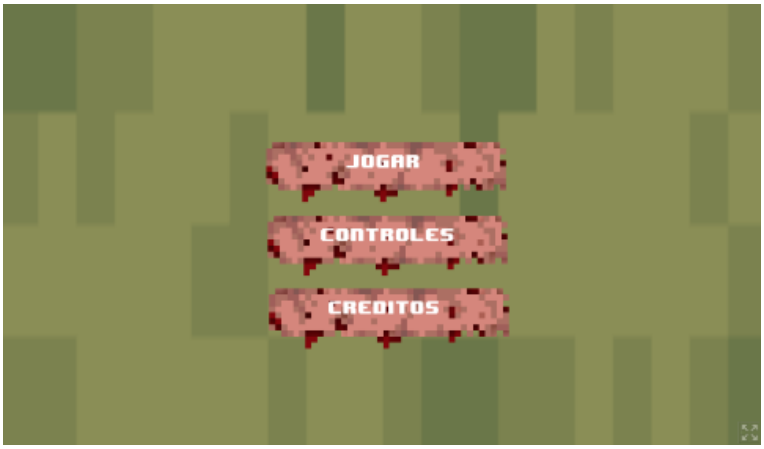


Figura 9 – Tela de menu inicial

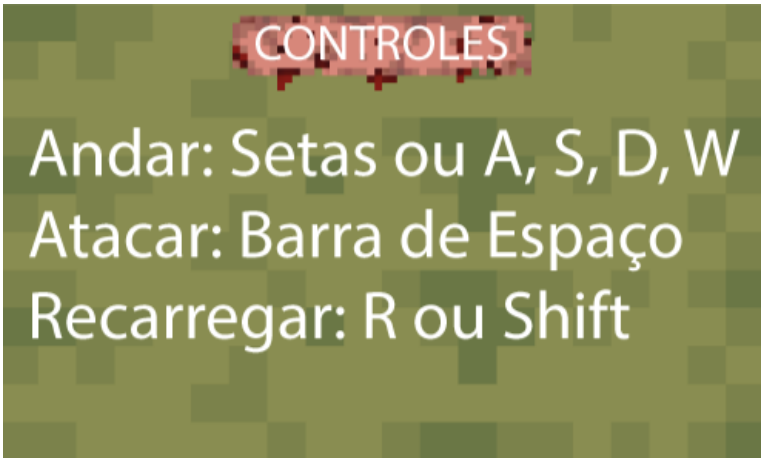
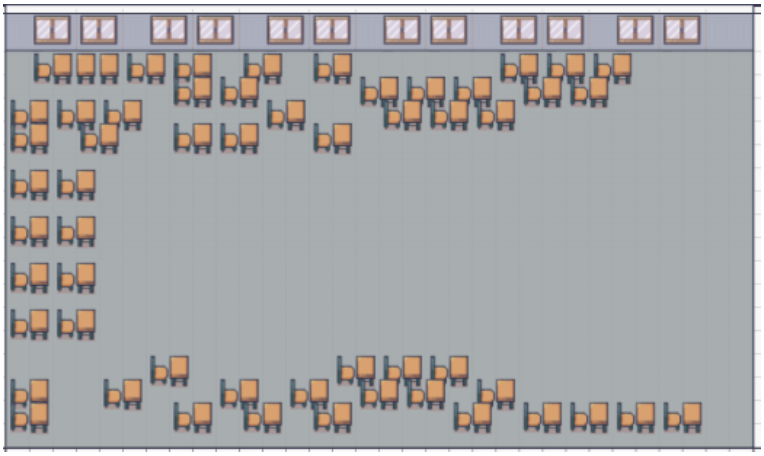


Figura 10 – Tela de menu inicial

Figura 11 – Tela de menu inicial

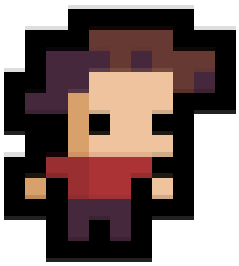


A.9 Elementos do jogo

Os elementos do jogo consistem em: Herói principal e arma de fogo.

O herói principal possui as seguintes características:

1. vida: 5
2. velocidade: 2



A arma de fogo possui as seguintes características:

1. carga: infinita
2. dano: 1

A.10 Objetos/Mecanismos

Movimentação do personagem: W A S D.

Atirar: Barra de espaço.