



**UNIVERSIDADE FEDERAL DO CEARÁ**  
**INSTITUTO UNIVERSIDADE VIRTUAL**  
**CURSO DE BACHARELADO EM SISTEMAS E MÍDIAS DIGITAIS**

**ERIC FAÇANHA NASCIMENTO**

**FLOW, ESTÉTICA E EMOÇÃO: UMA ABORDAGEM CONTEMPORÂNEA NO**  
**DESIGN E DESENVOLVIMENTO DO JOGO DIGITAL SPACE VIBES**

**FORTALEZA**

**2025**

**ERIC FAÇANHA NASCIMENTO**

**FLOW, ESTÉTICA E EMOÇÃO: UMA ABORDAGEM CONTEMPORÂNEA NO  
DESIGN E DESENVOLVIMENTO DO JOGO DIGITAL SPACE VIBES**

Monografia apresentada ao Programa de Graduação em Sistemas e Mídias Digitais da Universidade Federal do Ceará, como requisito parcial à obtenção do título de Bacharel em Sistemas e Mídias Digitais. Área de concentração: Desenvolvimento de jogos.

Orientador: Prof. Dr. Antônio José Melo Leite Júnior.

**FORTALEZA**

**2025**

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)

---

N194f Nascimento, Eric Façanha.

Flow, estética e emoção : uma abordagem contemporânea no design e desenvolvimento do jogo digital Space Vibes / Eric Façanha Nascimento. – 2025.  
128 f.

Trabalho de Conclusão de Curso (graduação) – Universidade Federal do Ceará, Instituto UFC Virtual, Curso de Sistemas e Mídias Digitais, Fortaleza, 2025.

Orientação: Prof. Dr. Antônio José Melo Leite Júnior.

1. Jogos digitais. 2. Unity. 3. Teoria do Flow. 4. LoFi. 5. Retrowave. I. Título.

CDD 302.23

---

**ERIC FAÇANHA NASCIMENTO**

**SPACE VIBES: UM ESTUDO SOBRE NARRATIVAS VISUAIS E JOGABILIDADE**

Monografia apresentada ao Programa de Graduação em Sistemas e Mídias Digitais da Universidade Federal do Ceará, como requisito parcial à obtenção do título de Bacharel em Sistemas e Mídias Digitais. Área de concentração: Desenvolvimento de jogos.

Aprovada em: 04/08/2025.

**BANCA EXAMINADORA**

---

Prof. Dr. Antônio José Melo Leite Júnior (Orientador)  
Universidade Federal do Ceará (UFC)

---

Prof. Me. Wellington Wagner Ferreira Sarmiento  
Universidade Federal do Ceará (UFC)

---

Prof. Dr. José Gilvan Rodrigues Maia  
Universidade Federal do Ceará (UFC)

A Deus, a minha mãe e padrasto, que sempre me encorajaram e ao meu amigo e companheiro de desenvolvimento Lucas Franco Cunha.

## **AGRADECIMENTOS**

Agradeço à Universidade Federal do Ceará (UFC) pelo suporte e apoio durante a pandemia de COVID-19. Foram momentos difíceis para mim e para muitos que ingressaram na graduação naquele período, como foi o meu caso, no semestre 2020.2. Tive medo de não conseguir me capacitar e evoluir no curso, mas os colegas de turma e os professores foram fundamentais para superar todas as dificuldades. No fim, todos os desafios foram vencidos graças ao suporte técnico, à capacitação dos professores e dos demais profissionais da instituição, além da colaboração de nós, estudantes.

Agradeço à minha mãe, que sempre esteve ao meu lado nessa jornada, incentivando meus estudos e me motivando a nunca desistir. Tudo isso só foi possível por causa de você. Agradeço também ao meu padrasto, uma peça importante na formação do meu caráter, que sempre me aconselhou, defendeu e me acolheu da melhor forma. Você também faz parte dessa conquista.

Ao Prof. Dr. Antônio José Melo Leite Júnior, que me orientou com paciência e dedicação, compartilhando ensinamentos e conselhos essenciais para a realização deste trabalho. Sou imensamente grato pela compreensão e por ter me guiado nessa etapa tão importante da minha graduação.

Aos meus amigos de infância, Lucas Franco Cunha, Luis Otavio Franco Cunha e sua família, por sempre me incentivarem a buscar a grandeza de fazer parte de uma instituição como a UFC, ajudando-me a moldar meu caráter e me acolher como parte da família. Um agradecimento especial ao Lucas Franco, que teve participação fundamental neste trabalho como parte da equipe de desenvolvimento, muito obrigado pela sua contribuição.

Por fim, agradeço aos meus amigos que acompanharam minha trajetória acadêmica: Larissa Araújo, Jessé Freire, Egberto Júnior, Vicente Laio, Cainan Matheus, Eric Victor e Andryel Coelho. Vocês tornaram a graduação mais leve e contribuíram intelectualmente em trabalhos, estudos e provas.

“A vida não é o problema, é batalha, é desafio.  
Cada obstáculo é uma lição, eu anuncio...”  
(RACIONAIS MC’S, 2002)

## RESUMO

Este Trabalho de Conclusão de Curso tem como objetivo apresentar o processo de idealização, fundamentação teórica e desenvolvimento de um jogo digital autoral, concebido com base nas estéticas visuais e sonoras das culturas LoFi e Retrowave. A proposta surge da intenção de investigar como experiências sensoriais imersivas, relaxantes e emocionalmente significativas podem ser proporcionadas ao jogador por meio da articulação entre arte, narrativa, interatividade e som em ambientes digitais. A pesquisa apoia-se em um robusto referencial teórico que abrange os estudos sobre jogos, gamificação, interatividade, design emocional e teoria do Flow, além de abordar elementos técnicos essenciais como motores gráficos, shaders, partículas e efeitos visuais no motor gráfico Unity, com suporte de ferramentas como Adobe Photoshop e CorelDRAW. *Space Vibes* é um jogo experimental desenvolvido com estética Retrowave e atmosfera LoFi, que simula a experiência de flutuar no espaço em gravidade zero. O processo metodológico adotado para o seu desenvolvimento seguiu uma abordagem prática, com etapas de brainstorming, prototipagem, testes e ajustes finais. Foram desenvolvidas mecânicas centradas na movimentação em gravidade zero, geração procedural de elementos e integração harmônica de trilha sonora com efeitos audiovisuais. Como resultado, foi produzido um protótipo funcional que demonstra a viabilidade técnica e estética da proposta, evidenciando o potencial da união das estéticas Retrowave e LoFi no desenvolvimento de jogos digitais. O projeto também apresenta reflexões sobre o uso de inteligência artificial para otimizar o escopo e o tempo de produção, e propõe caminhos para a expansão de uma possível aplicação do jogo em contextos terapêuticos. Sendo uma monografia de conclusão de curso, este trabalho foi desenvolvido de forma a permitir sua replicabilidade, oferecendo uma estrutura clara, acessível e tecnicamente fundamentada para que outros interessados possam desenvolver seus próprios títulos autorais. Assim, contribui para o campo do design de jogos independentes ao propor uma abordagem sensível, criativa e viável, que integra conceitos de design, narrativa e mecânicas interativas conforme diferentes visões e objetivos.

**Palavras-chave:** design de jogos; estética digital; LoFi; Retrowave; interatividade.



## ABSTRACT

This Undergraduate Thesis aims to present the process of conceptualization, theoretical foundation, and development of an original digital game designed around the visual and sound aesthetics of LoFi and Retrowave cultures. The project explores how immersive, relaxing, and emotionally meaningful sensory experiences can be delivered to players through the integration of art, narrative, interactivity, and sound in digital environments. The research is supported by a solid theoretical framework, encompassing studies on games, gamification, interactivity, emotional design, and the Flow theory, while also addressing technical aspects such as game engines, shaders, particle systems, and visual effects using the Unity engine, with additional support from tools like Adobe Photoshop and CorelDRAW. *Space Vibes* is an experimental game developed with a Retrowave aesthetic and LoFi atmosphere, simulating the sensation of floating in space in zero gravity. The methodology followed a practical approach, including brainstorming, prototyping, testing, and final adjustments. The game features mechanics based on zero-gravity movement, procedural generation of elements, and the harmonic integration of soundtrack and audiovisual effects. As a result, a functional prototype was developed, demonstrating the technical and aesthetic viability of the proposal and the potential of combining Retrowave and LoFi aesthetics in digital game design. The project also discusses the use of artificial intelligence to optimize production scope and time and proposes its possible application in therapeutic contexts. As a final graduation project, this work was designed to be replicable, offering a clear, accessible, and technically grounded structure for other developers interested in creating their own independent titles. Thus, it contributes to the field of independent game design by proposing a sensitive, creative, and feasible approach that integrates design, narrative, and interactive mechanics according to diverse creative goals.

**Keywords:** game design; digital aesthetics; LoFi; Retrowave; interactivity.

## LISTA DE FIGURAS

Figura 1	-Capa do jogo Hyper Light Drifter .....	16
Figura 2	- Capa do jogo Katana ZERO .....	17
Figura 3	- Capa do jogo Journey .....	17
Figura 4	- Imagem da primeira versão do jogo Batalha Naval .....	21
Figura 5	- Imagem do modelo 3D da Spider Gwen com rig .....	38
Figura 6	- Sprite Sheets do jogo Megaman X .....	39
Figura 7	- Mapas de textura (Albedo, Normal, Metallic, Roughness) .....	41
Figura 8	- Imagem do Pipeline de desenvolvimento .....	58
Figura 9	- Gráfico de Gantt do projeto Space Vibes .....	62
Figura 10	- Imagens com estética a) LoFi e b) retrowave .....	63
Figura 11	- Imagens do jogo a) Retro Drive e b)Synthwave Fury .....	64
Figura 12	- Imagem do jogo Spirit City: LoFi Sesssions .....	66
Figura 13	- Imagem do jogo LoFi Katana .....	67
Figura 14	- Figura 14 - Imagens dos planetas gerados por IA .....	73
Figura 15	- Imagens do astronauta geradas por IA .....	74
Figura 16	- Asset final do personagem "Astronauta" .....	74
Figura 17	- Partes divididas do astronauta a) cabeça , b) dorso, c) perna direita, d) perna esquerda, e) braço direito e f) braço esquerdo .....	75
Figura 18	- Partes unidas e com o rigging .....	75
Figura 19	- Imagem contendo o joystick no jogo .....	77
Figura 20	- Imagem das partículas do astronauta .....	81
Figura 21	- Imagem de fundo do jogo na a) cor padrão e b) roxo escuro .....	82
Figura 22	- Imagem com a paleta de cores do jogo .....	83
Figura 23	- Imagem do astronauta e do background .....	84
Figura 24	- Imagem do rastro do astronauta .....	85
Figura 25	- Imagem do personagem no estado idle .....	90
Figura 26	- Imagem do hud inicial .....	92
Figura 27	- Imagem do astronauta a) parado e b) movimento .....	95
Figura 28	- Imagens do código do script de movimentação do personagem .....	96
Figura 29	- Modelos finais de cada planeta a) planetOk, b) OneTex, c) TwoTex e d) Standard .....	99
Figura 30	- Imagens do código do sprite dos planetas .....	99

Figura 31	- Imagem da aba de VFX da Unity .....	100
Figura 32	- Imagem da aba de prefabs .....	101
Figura 33	- Imagem dos planeta com partículas .....	102
Figura 34	- Imagem dos planetas .....	103
Figura 35	- Imagem do ícone na pasta de aplicativos .....	106
Figura 36	- Imagem exibe o layout do Particle System da Unity .....	118
Figura 37	- Imagem da ferramenta Shader Graph .....	121
Figura 38	- Imagem da ferramenta Visual Effect Graph .....	123

## **LISTA DE TABELAS**

Tabela 1	Tabela de elementos gerados neste trabalho e seus links de acesso .....	19
Tabela 2	Tabela de referência dos principais canais de LoFi no Youtube .....	69

# SUMÁRIO

<b>1</b>	<b>INTRODUÇÃO .....</b>	<b>16</b>
<b>1.1</b>	<b>Objetivos .....</b>	<b>18</b>
<i>1.1.1</i>	<i>Objetivo geral .....</i>	<i>18</i>
<i>1.1.2</i>	<i>Objetivos específicos .....</i>	<i>19</i>
<i>1.1.3</i>	<i>Delimitação da Contribuição .....</i>	<i>19</i>
<b>1.2</b>	<b>Estrutura do documento .....</b>	<b>20</b>
<b>2</b>	<b>REFERENCIAL TEÓRICO .....</b>	<b>21</b>
<b>2.1</b>	<b>A essência do jogo: reflexões sobre sua natureza, estrutura e funções .....</b>	<b>21</b>
<i>2.1.1</i>	<i>Compreendendo o design de jogos digitais .....</i>	<i>23</i>
<i>2.1.2</i>	<i>Programação para jogos digitais: conceitos fundamentais .....</i>	<i>25</i>
<b>2.2</b>	<b>O papel da interatividade no design de jogos .....</b>	<b>26</b>
<b>2.3</b>	<b>Cultura lofi e retrowave: estéticas sonoras e visuais na era digital e sua influência no youtube contemporâneo .....</b>	<b>28</b>
<i>2.3.1</i>	<i>LoFi como linguagem estética e emocional .....</i>	<i>28</i>
<i>2.3.2</i>	<i>Retrowave: Estética Visual e Musical .....</i>	<i>30</i>
<b>2.4</b>	<b>O motor gráfico Unity: fundamentos e aplicações no desenvolvimento de jogos .....</b>	<b>32</b>
<i>2.4.1</i>	<i>O que são assets e sua importância no desenvolvimento de jogos digitais .....</i>	<i>35</i>
<i>2.4.2</i>	<i>Modelos 3D no design de jogos: forma, volume e estilo .....</i>	<i>36</i>
<i>2.4.3</i>	<i>Sprites no desenvolvimento de jogos: técnica e estética .....</i>	<i>38</i>
<i>2.4.4</i>	<i>Texturas e materiais: composição visual no espaço virtual .....</i>	<i>39</i>
<i>2.4.5</i>	<i>Som nos jogos digitais: imersão, ritmo e feedback .....</i>	<i>42</i>
<i>2.4.6</i>	<i>Scripts de programação: fundamentos e aplicações .....</i>	<i>43</i>
<b>2.5</b>	<b>Shaders e efeitos visuais (vfx): fundamentos técnicos e aplicações em jogos digitais .....</b>	<b>45</b>
<b>2.6</b>	<b>O Adobe Photoshop como plataforma de criação visual digital .....</b>	<b>46</b>
<i>2.6.1</i>	<i>A importância do Photoshop para o design de jogos digitais .....</i>	<i>47</i>
<b>2.7</b>	<b>Coreldraw e gráficos vetoriais: fundamentos e aplicações na produção visual .....</b>	<b>47</b>
<i>2.7.1</i>	<i>A utilização do Coreldraw na criação de interfaces para jogos .....</i>	<i>48</i>
<b>2.8</b>	<b>O papel das emoções no design emocional: fundamentos e aplicações .....</b>	<b>48</b>

2.9	O papel das IA's na arte digital .....	50
2.10	A teoria do Flow: fundamentos e aplicações no universo dos jogos .....	52
2.10.1	<i>Equilibrando desafios e habilidades</i> .....	54
2.10.2	<i>Feedback imediato</i> .....	55
2.10.3	<i>Emoções e imersão</i> .....	55
3	<b>METODOLOGIA</b> .....	57
3.1	Referencial .....	58
3.2	Brainstorming .....	59
3.3	Estudo de ferramentas de desenvolvimento .....	60
3.4	Estudo de mecânicas e design de jogo .....	60
3.5	Estudo das emoções .....	61
4	<b>DESENVOLVIMENTO DO JOGO SPACE VIBES</b> .....	62
4.1	Semana 1: planejamento e conceituação - brainstorming e definição de ideias .....	62
4.2	Bases documentais e experiência emocional no design .....	68
4.3	Análise e seleção de ferramentas e tecnologias .....	70
4.4	Criação de protótipos e composição estética .....	71
4.5	Planejamento técnico do fluxo de desenvolvimento .....	76
4.5.1	<i>Movimentação adaptada ao mobile</i> .....	76
4.5.2	<i>Manipulação física do movimento com rigidbody 2D</i> .....	77
4.5.3	<i>Integração das animações com o sistema de input</i> .....	78
4.6	Implementação da geração procedural de corpos celestes .....	78
4.7	Integração de efeitos visuais .....	79
4.8	Desenvolvimento da identidade visual e sonora .....	81
4.8.1	<i>Criação do background</i> .....	81
4.8.2	<i>Aplicação de shading nos planetas</i> .....	83
4.8.3	<i>Rastro visual com shading no personagem</i> .....	84
4.8.4	<i>Escolha da trilha sonora</i> .....	85
4.9	Revisão conceitual .....	86
4.10	Semana 2: desenvolvimento e implementação .....	86
4.10.1	<i>Configuração inicial do projeto - estrutura inicial na Unity</i> .....	87
4.10.2	<i>Importação e organização dos assets</i> .....	87
4.10.3	<i>Criação dos prefabs de planetas</i> .....	88

4.10.4	<i>Montagem do personagem principal</i>	89
4.10.5	<i>Animações e configuração do animator</i>	90
4.10.6	<i>Criação do canvas e elementos da interface</i>	91
4.10.7	<i>Desenvolvimento de scripts base</i>	92
4.10.8	<i>Montagem e organização da cena</i>	93
4.11	<b>Mecânica de movimentação</b>	93
4.11.1	<i>Movimentação em ambiente de gravidade zero</i>	93
4.11.2	<i>Uso do joystick pack e controle direcional</i>	93
4.11.3	<i>Integração visual: rastro com shaders dinâmicos</i>	94
4.11.4	<i>Experiência imersiva e continuidade visual</i>	98
4.12	<b>Geração procedural de planetas</b>	98
4.13	<b>Design de efeitos visuais e áudio</b>	102
4.14	<b>Testes e ajustes</b>	103
4.15	<b>Publicação</b>	105
5	<b>RESULTADOS</b>	107
6	<b>CONSIDERAÇÕES FINAIS</b>	109
	<b>REFERÊNCIAS</b>	112
	<b>APÊNDICE A – FERRAMENTAS TÉCNICAS DE VFX NA UNITY</b>	118
	<b>APÊNDICE B – DOCUMENTAÇÃO TÉCNICA: ARQUIVOS DE DESENVOLVIMENTO E CÓDIGOS DO JOGO</b>	130

## 1 INTRODUÇÃO

O design de jogos digitais representa hoje uma das formas mais sofisticadas de unir arte, ciência e cultura. Ao longo das últimas décadas, os jogos digitais evoluíram ao ponto de serem vistos além de uma simples mídia de entretenimento, passando a ser estudados como sistemas interativos expressam valores, comportamentos e emoções humanas.

Como destaca Johan Huizinga, “o jogo é uma função significativa, que não pode ser dissociada da cultura” (HUIZINGA, 2000, p. 3). Essa afirmação define que o jogo é, antes de tudo, uma prática simbólica que atravessa eras, uma forma primária de criação e recriação de mundos, capaz de simular cenários, desafios e emoções com forte potencial de imersão e engajamento.

Figura 1 - Imagem da capa do jogo Hyper Light Drifter



Fonte: [https://www.nintendo.com/pt-br/store/products/hyper-light-drifter-special-edition-switch/?srsltid=AfmBOoqRDjnJf\\_xiCetuNF1JrbNDhWRsaHNSH5s6LWTC8jGLEUKoBaaw](https://www.nintendo.com/pt-br/store/products/hyper-light-drifter-special-edition-switch/?srsltid=AfmBOoqRDjnJf_xiCetuNF1JrbNDhWRsaHNSH5s6LWTC8jGLEUKoBaaw)

No cenário atual da indústria e dos estudos sobre jogos, é notável a ascensão de títulos independentes que exploram estilos culturais específicos como base para construção de seus mundos lúdicos. Jogos como Hyper Light Drifter (2016), figura 1, com paleta vibrante e visual pixelado inspirado no synthwave, Katana ZERO (2019), figura 2, que combina ação rápida, atmosfera neon e trilha eletrônica, e Journey (2012), figura 3, com ambientação envolvente e mecânicas fluidas, ilustram como elementos visuais e sonoros da cultura pop, cinema e música moldam a experiência do jogador. Tais jogos evidenciam uma abordagem sensível e intencional na construção do design, onde cada elemento comunica e reforça uma identidade própria.



Figura 2 - Capa do jogo Katana Zero



Fonte: [https://www.xbox-now.com/pt/game/13981/katana-zero-xbl#google\\_vignette](https://www.xbox-now.com/pt/game/13981/katana-zero-xbl#google_vignette)

Essa complexidade do design de jogos é evidente quando elementos técnicos, como regras, sistemas de física, programação e interatividade, aliam-se à direção artística e sonora, criando atmosferas sensoriais envolventes. Eric Zimmerman, importante teórico da área, define os jogos como “sistemas interativos compostos por regras, em que os jogadores agem para atingir objetivos em contextos de conflito” (ZIMMERMAN, 2003). Essa definição ressalta o caráter estrutural do design, abrangendo desde mecânicas até estética, som e narrativa.

Figura 3 - Capa do jogo Journey



Fonte: <https://www.techtudo.com.br/review/journey.ghml>

Este trabalho desenvolve o jogo Space Vibes, concebido como uma experiência interativa que integra duas correntes culturais contemporâneas: o Retrowave e o LoFi. O Retrowave, também chamado de Synthwave, é um movimento estético-musical que resgata

elementos visuais e sonoros dos anos 1980, com cores vibrantes, uso extensivo de neon, sintetizadores e uma atmosfera retrô futurista, como discutido por Tanner (2016) ao abordar a nostalgia mediada pelas mídias digitais.

Já o LoFi é um gênero musical e visual que valoriza a imperfeição sonora, a repetição hipnótica de ritmos lentos e uma estética minimalista, voltada para o relaxamento, introspecção e foco mental, seguindo perspectivas como as de Fisher (2014), que associa o gênero à busca por conforto e suspensão do tempo no cotidiano digital. Unir essas duas influências permite ao jogo construir um ambiente sonoro e visual com grande potencial imersivo e ao mesmo tempo tranquilizante.

A partir da união dessas estéticas, a proposta do jogo Space Vibes é provocar no jogador a sensação de Flow, conceito desenvolvido por Mihaly Csikszentmihalyi (1990), que descreve o estado de imersão total em uma atividade, em que o indivíduo perde a noção do tempo e se concentra plenamente em seu objetivo. De forma simplificada, essa experiência é possível quando há um equilíbrio ideal entre desafio e habilidade. Em um jogo digital, esse estado pode ser alcançado por meio da construção harmônica entre mecânicas bem ajustadas, ambientação sonora coerente e feedbacks visuais responsivos.

Por se tratar de uma monografia de conclusão de curso, este trabalho também busca oferecer uma base replicável para futuros desenvolvedores e pesquisadores interessados em unir estética, jogabilidade e experiência sensorial no design de jogos independentes.

## **1.1 Objetivos**

Diante desse panorama, este trabalho busca compreender como a combinação entre estética cultural e design de experiência pode favorecer o envolvimento emocional e sensorial do jogador. Para isso, foram estabelecidos objetivos que orientam o desenvolvimento da pesquisa e da aplicação prática.

### ***1.1.1 Objetivo geral***

Este trabalho tem como objetivo principal investigar de que forma elementos culturais e estéticos associados às vertentes Retrowave e LoFi podem ser integrados ao design de jogos digitais, visando intensificar a experiência imersiva e emocional do jogador. A proposta se concentra na criação de uma ambientação estilizada que une estética visual e sonora

em uma linguagem coesa, proporcionando ao jogador uma experiência sensorial marcante e alinhada às tendências culturais contemporâneas.

### ***1.1.2 Objetivos específicos***

Como objetivo específico, busca-se aplicar a teoria do Flow ao desenvolvimento da jogabilidade, avaliando como os elementos de dificuldade, responsividade e progressão podem ser ajustados para promover um estado de engajamento contínuo. Além disso, oferecer uma experiência emocional positiva, pautada na calma, no foco e na ausência de ansiedade, considerando sua viabilidade técnica e criativa especialmente pela capacidade de gerar *assets* de forma rápida e eficiente, o que torna possível a experimentação estética dentro de ciclos de produção reduzidos.

### ***1.1.3 Delimitação da Contribuição***

O grupo de desenvolvimento responsável pelo projeto integra a Sector 8 Studio (disponível em: <https://www.instagram.com/sector8studio/>), um estúdio independente voltado para produção de experiências interativas. O autor deste trabalho atuou como integrante da empresa durante o desenvolvimento do jogo, participando diretamente das etapas de concepção, design e implementação. A equipe, formada por dois profissionais de perfil multidisciplinar, reuniu conhecimentos em design gráfico, programação, direção de arte e teoria do jogo.

Para fins de documentação e experimentação, a versão protótipo de Space Vibes está disponível para download em formato APK, assim como o acesso aos demais arquivos desenvolvidos nos links presentes na tabela 1. Ressalta-se que, além de participar do desenvolvimento, o autor também realizou a análise presente neste trabalho.

Tabela 1 - Tabela de elementos gerados neste trabalho e seus links de acesso

Arquivos	Link de acesso
APK do jogo	<a href="https://drive.google.com/file/d/160kmNeorl0cnUA67uoVj-RPXAlWD3fob/view?usp=sharing">https://drive.google.com/file/d/160kmNeorl0cnUA67uoVj-RPXAlWD3fob/view?usp=sharing</a> .

<b>Assets, imagens</b>	<a href="https://drive.google.com/drive/folders/1rxp70nEvW2OYZ27iz4h0_BUdU8OHVdKU?usp=sharing">https://drive.google.com/drive/folders/1rxp70nEvW2OYZ27iz4h0_BUdU8OHVdKU?usp=sharing</a>
<b>Repositório</b>	<a href="https://github.com/eric02nasc/SpaceVibes-Sector8.git">https://github.com/eric02nasc/SpaceVibes-Sector8.git</a>

Fonte: Elaborado pelo autor (2025).

O jogo Space Vibes está protegido por licença exclusiva da Sector 8 Studio, que detém todos os direitos sobre sua venda, distribuição e comercialização. Nenhum uso comercial ou reprodução dos seus elementos, incluindo código, arte, áudio e demais componentes, é permitido sem autorização expressa da empresa. Para fins de documentação e experimentação, os arquivos e recursos do projeto estão disponíveis para acesso nos links fornecidos, garantindo transparência e facilidade de consulta, porém permanecem sob a proteção dos direitos da Sector 8 Studio.

## 1.2 Estrutura do documento

O trabalho está dividido em capítulos que abordam, inicialmente, uma revisão teórica sobre design de jogos e conceitos culturais que fundamentam o projeto (Capítulo 2); em seguida, a apresentação da metodologia usada para o desenvolvimento de Space Vibes (Capítulo 3); em seguida, a etapa de desenvolvimento com enfoque nas mecânicas, ambientação sonora e visual, e experiência do jogador (Capítulo 4); e, por fim, uma avaliação dos resultados obtidos e considerações finais que apontam possíveis desdobramentos e contribuições para a área (Capítulo 5).

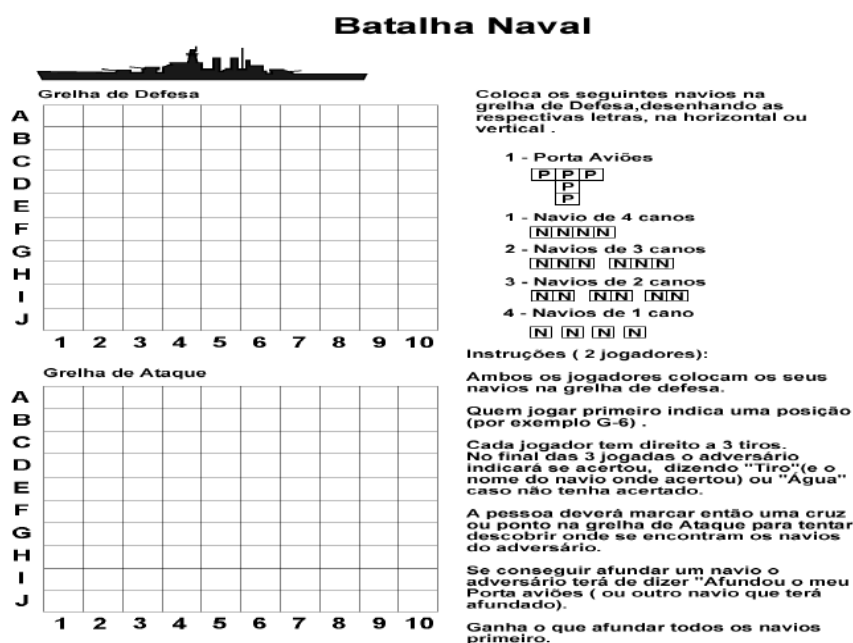
## 2 REFERENCIAL TEÓRICO

Para compreender todo o processo de desenvolvimento deste projeto, é essencial explorar os principais pontos de referência teóricos e técnicos que sustentam suas bases. Além de utilizar conceitos consolidados sobre design de jogos, torna-se relevante destacar as características das ferramentas empregadas em desenvolvimento de jogos e como elas contribuem para esse objetivo. Assim, esta seção dedica-se a explorar a definição de jogo, suas funções e elementos constitutivos, bem como as teorias que fundamentam o design.

### 2.1 A essência do jogo: reflexões sobre sua natureza, estrutura e funções

Um jogo pode ter várias definições. Já que, suas características não se restringem somente aos jogos digitais, existem muitas outras formas de um jogo se apresentar, antes de se tornar uma experiência digital. A exemplo disso, destaca-se o jogo Batalha Naval (Milton Bradley Company, 1931), um jogo de tabuleiro de dois jogadores, onde os jogadores adivinham os quadrados com os navios do rival. Ganha quem conseguir destruir todos os navios adversários. Sua primeira aparição foi nos cadernos de militares russos na primeira guerra mundial (1914-1918). Posteriormente, foi lançado oficialmente como um jogo pela Milton Bradley, ainda jogado no papel como pode ser visualizado na figura 4.

Figura 4 - Imagem da primeira versão do jogo Batalha Naval



Fonte: <https://papeis.blogs.sapo.pt/108139.html>

Em 1967, a Milton Bradley introduziu a versão em plástico rígido do Batalha Naval, cujo design básico persiste até hoje com pequenas variações entre edições (SMITH, 2010). Atualmente, adaptações digitais incorporam gráficos avançados e efeitos realistas, demonstrando como a tecnologia transformou jogos tradicionais em experiências imersivas.

Levando em consideração esse contexto, destaca-se algumas das referências mais significativas com relação a definição de jogo. Os especialistas, Katie Salen e Eric Zimmerman, no livro *Rules of Play: Game Design Fundamentals*, oferecem uma abordagem sistemática, enfatizando o jogo como um sistema interativo de regras:

"Um jogo é um sistema em que os jogadores se engajam em um conflito artificial, definido por regras, que resulta em um desfecho quantificável" (SALEN; ZIMMERMAN, 2004, p. 80).

Além disso, o historiador Johan Huizinga, em seu clássico *Homo Ludens*, destaca que o jogo é uma atividade primária da cultura humana, definida por sua característica de ser livre, separada da vida cotidiana e regida por regras próprias:

"O jogo é uma atividade ou ocupação voluntária, exercida dentro de certos limites fixos de tempo e espaço, segundo regras livremente aceitas, mas absolutamente obrigatórias, dotada de um fim em si mesma, acompanhada de um sentimento de tensão e alegria e de uma consciência de 'ser diferente' da 'vida cotidiana'" (HUIZINGA, 2000, p. 10).

Ainda nesse seguimento, Roger Caillois expande a visão de Huizinga, classificando os jogos em categorias baseadas em seus elementos essenciais:

"O jogo é uma atividade que deve ser: 1) livre, 2) separada (circunscrita dentro de limites de espaço e tempo), 3) incerta, 4) improdutiva, 5) regulamentada e 6) fictícia" (CAILLOIS, 1990, p. 24)."

Em contrapartida, uma visão mais moderna que envolve conceitos ligados a função emocional e social dos jogos apresentado por Jane McGonigal, em *Reality is Broken*, destaca os jogos como:

"Os jogos são sistemas projetados para criar felicidade genuína. Eles nos oferecem a oportunidade de nos sentirmos motivados, realizados e conectados a algo maior do que nós mesmos" (MCGONIGAL, 2011, p. 21).

Por fim, uma definição que destaca o parâmetro mais comum entre as diversas teorias citadas sobre jogos é a proposta de Bernard Suits. Ele argumenta que o elemento essencial e compartilhado em todas as definições de jogos é a presença de regras e obstáculos artificiais que orientam a experiência. Essa abordagem filosófica é frequentemente utilizada para identificar o ponto central que une as várias perspectivas teóricas sobre o que é um jogo. Assim Suits destaca:

"O jogo é a tentativa voluntária de superar obstáculos desnecessários, respeitando regras que limitam os meios permitidos para alcançar um fim pré-estabelecido" (SUITS, 1978, p. 43).

Entre as definições de Huizinga, Caillois, Salen e Zimmerman, e McGonigal, observa-se que as regras, a liberdade de escolha e o propósito interno (ou fim) aparecem consistentemente como aspectos centrais. Portanto, as regras são amplamente reconhecidas como o elemento fundamental que distingue jogos de outras atividades, pois são elas que criam os desafios e delimitam o espaço lúdico, que se trata de um ambiente, seja ele físico, digital ou simbólico, onde ocorre o jogo, ou seja, onde regras específicas são aceitas temporariamente pelos participantes e onde se suspende, de forma voluntária, a realidade cotidiana.

### ***2.1.1 Compreendendo o design de jogos digitais***

O design de jogos é o processo criativo que envolve a concepção, planejamento e desenvolvimento de todos os aspectos que compõem a experiência de um jogo. Ele engloba desde a criação de mecânicas, que são um conjunto de regras e ações que definem como um jogo funciona, até o que os jogadores podem e devem fazer. Essas mecânicas influenciam a jogabilidade, a estética, a narrativa e o feedback<sup>1</sup>.

---

<sup>1</sup> Feedback ou "experiência" na sua forma traduzida para o contexto, é a resposta do sistema ao comportamento do jogador, que fornece informações imediatas sobre suas ações, ajudando-o a entender as consequências dentro do jogo e ajustando sua estratégia ou desempenho.

Outro ponto que está inserido no estudo de design de jogos são os sistemas, que são uma coleção de mecânicas de jogo organizadas por jogadores ou criadas por designers. Esses sistemas podem ser compreendidos de forma isolada de outros sistemas e que controlam um grupo significativo de comportamentos não triviais relacionados de jogabilidade em direção a um resultado específico.

Todos esses pontos, incluindo a narrativa, os desafios e os elementos visuais, têm o papel de engajar o jogador. De acordo com Katie Salen e Eric Zimmerman, em *Rules of Play: Game Design Fundamentals*:

"Design de jogos é o ato de criar regras e interações que resultem em uma experiência significativa para o jogador" (SALEN; ZIMMERMAN, 2004, p. 11).

O design de jogos não se limita à estética ou à funcionalidade, mas busca equilibrar esses elementos para proporcionar uma experiência lúdica que seja ao mesmo tempo desafiadora, recompensadora e emocionalmente envolvente. Richard Rouse III, em *Game Design: Theory and Practice*, afirma:

"O design de jogos é uma prática interdisciplinar que combina arte, tecnologia e psicologia para criar mundos interativos que ressoem com os jogadores" (ROUSE, 2005, p. 22).

Projetar uma experiência de jogo significativa é o objetivo principal de um design bem-sucedido. Essa experiência surge da relação entre a ação do jogador e o resultado do sistema, tal como é percebido pelo jogador através da interação com o sistema do jogo (SALEN; ZIMMERMAN, 2004, p. 34). Na prática, o design de jogos pode ser dividido em diferentes áreas, como:

- **Design de mecânicas:** Definição das regras e sistemas que governam o jogo.
- **Design de níveis:** Criação de espaços interativos que equilibram desafio e progressão.
- **Design narrativo:** Desenvolvimento da história e dos elementos que conectam o jogador ao universo do jogo.

Para Jesse Schell, em *The Art of Game Design: A Book of Lenses*:



"O design de jogos é sobre criar experiências; é a arte de moldar como o jogador percebe, interage e se sente durante o jogo" (SCHELL, 2014, p. 5).

No processo de produção de um jogo, o design de jogo desempenha um papel essencial na criação de uma experiência e interatividade coesas e baseadas nas características dentro da aplicação.

### ***2.1.2 Programação para jogos digitais: conceitos fundamentais***

Essa etapa é responsável por “dar vida” aos elementos gráficos. Sob as condições certas, a programação de jogos envolve a criação de algoritmos e estruturas de dados que suportam a funcionalidade de um jogo, incluindo a renderização gráfica<sup>2</sup>, física, inteligência artificial e interação do jogador (GREGORY, 2018). É papel do programador implementar a lógica, as mecânicas e as interações.

Em Space Vibes, a programação foi crucial para criar sistemas interativos, comportamentos dinâmicos e proporcionar a experiência de jogo imersiva almejada. A ferramenta usada foi o motor gráfico Unity, um sistema capaz de funcionar em multiplataforma, utilizado para criar jogos e aplicações interativas em 2D, 3D, realidade aumentada e realidade virtual, oferecendo ferramentas integradas de design, programação e simulação em tempo real.

Dentro da Unity, a programação é realizada principalmente por meio da linguagem C#, que oferece uma combinação de desempenho e simplicidade. A Unity Technologies, empresa responsável pelo desenvolvimento do motor gráfico Unity, define programação para jogos como:

"O processo de transformar conceitos criativos em realidade interativa, utilizando scripts para controlar comportamentos de objetos, gerenciar eventos e responder às entradas do jogador" (UNITY TECHNOLOGIES, 2023).

A programação em jogos pode ser dividida em diversas áreas, incluindo:

- **Mecânicas de jogo:** Define as regras e interações fundamentais, como movimentação, ataques e interações com o ambiente.

---

<sup>2</sup> A renderização gráfica é o processo de finalização de uma imagem por meio de técnicas de computação gráfica, convertendo modelos 2D ou 3D em imagens digitais a partir de instruções e dados fornecidos por um software. Esse processo permite reunir elementos visuais como sombras, efeitos de iluminação e texturas, resultando na imagem final apresentada ao usuário (FOLEY et al., 1996).

- **Inteligência artificial (IA):** Determina os comportamentos dos personagens não jogáveis (Non Playable Characters ou NPC's).
- **Sistemas de física:** Controla simulações de colisões, gravidade e outros aspectos físicos.
- **Interface do usuário (User Interface ou UI):** Controla elementos visuais como menus, HUD (Head-Up Display) e barras de progresso.

Jeremy Gibson Bond, em *Introduction to Game Design, Prototyping, and Development*, destaca a importância da programação na construção de uma experiência coesa e funcional de jogos:

"A programação é o ponto onde o design se encontra com a execução. Ela traduz ideias em ações concretas e interativas, conectando o jogador ao mundo virtual de forma lógica e consistente" (BOND, 2014, p. 335).

Em um projeto de desenvolvimento de um jogo, a programação é utilizada para implementar sistemas de controle de movimento, aplicação de mudanças de características para sprites<sup>3</sup>, como cores e tamanho, controle de shaders<sup>4</sup> e VFX<sup>5</sup>, e controle de animações, entre outras aplicações. As principais áreas de aplicação desses códigos e o uso das ferramentas fornecidas pela Unity serão discutidos posteriormente, destacando como foram essenciais para alcançar os objetivos do projeto.

## 2.2 O papel da interatividade no design de jogos

No contexto do desenvolvimento de jogos digitais, a interatividade é o ponto principal, pois define a dinâmica entre o jogador e o sistema. É com base nesse parâmetro que o desenvolvedor molda a experiência para o jogo desenvolvido ser o mais imersivo e envolvente para o jogador. Essa perspectiva se tornou um dos principais pontos das aplicações de jogos na atualidade.

---

<sup>3</sup> Sprites, ou na sua tradução “figuras” ou “imagens bidimensionais”, são gráficos 2D utilizados para representar personagens, objetos ou elementos visuais em jogos.

<sup>4</sup> Shaders, ou na sua tradução “sombreados” ou “programas de sombreado”, são pequenos códigos executados na GPU que controlam como os elementos gráficos são renderizados, permitindo efeitos como iluminação e texturas dinâmicas.

<sup>5</sup> VFX (*Visual Effects*), ou na sua tradução “efeitos visuais”, referem-se a recursos gráficos aplicados em tempo real, como explosões, partículas e brilhos, que tornam a experiência visual mais imersiva.

A interatividade é a maneira pela qual os usuários podem se envolver e manipular ativamente o conteúdo das mídias (Bolter; Grusin, 2016), especialmente na área de desenvolvimento de jogos, onde essa característica é mais presente. Por se tratar de uma aplicação que depende da interação de um ou mais usuários, destacam-se não apenas os aspectos de programação, mas também os de arte e design.

Desse modo, é possível ver que um jogo vai além do simples ato de interagir; trata-se de um diálogo constante, no qual o jogador participa ativamente no ambiente digital, influenciando e sendo influenciado por ele. A interatividade é um elemento chave para a criação de experiências significativas em jogos, que tem como objetivo explorar a conexão entre o jogador e o universo lúdico por meio da arte, do design e da mecânica.

De acordo com Janet Murray, em seu livro *Hamlet on the Holodeck*, a interatividade é a característica distintiva dos meios digitais e está intrinsecamente ligada à ação do usuário:

"Interatividade é a capacidade de fazer escolhas significativas dentro de um ambiente que reage às ações do usuário" (MURRAY, 1997, p. 126).

Lev Manovich, em *The Language of New Media*, descreve a interatividade como um atributo essencial dos meios digitais, destacando seu papel na construção de experiências personalizadas:

"Interatividade é a capacidade do usuário de intervir em um fluxo de informações e modificar o resultado em tempo real" (MANOVICH, 2001, p. 55).

Jesper Juul, em *The Art of Failure: An Essay on the Pain of Playing Video Games*, enfatiza que a interatividade nos jogos não apenas envolve o controle direto do jogador sobre os elementos do sistema, mas também a criação de uma relação de aprendizado e adaptação constante:

"Os jogos são sistemas interativos onde o jogador aprende e se adapta, testando suas habilidades contra desafios impostos por regras e mecânicas" (JUUL, 2013, p. 45).

Adicionalmente, Katie Salen e Eric Zimmerman, no livro *Rules of Play*, reforçam que a interatividade nos jogos é fundamental para a criação de significado:

"A interatividade nos jogos surge da interação entre jogadores e sistemas, onde ações levam a reações que geram significado e engajamento" (SALEN; ZIMMERMAN, 2004, p. 34).

A interatividade é um elemento fundamental no design de jogos, pois possibilita a criação de experiências envolventes que promovem o engajamento e a imersão do jogador. No caso do jogo Space Vibes, essa interatividade busca-se ao integrar elementos audiovisuais e narrativos inspirados nos vídeos LoFi, transformando uma experiência passiva em uma vivência ativa e personalizada. O projeto demonstra como a combinação de estética, som e jogabilidade pode construir um contexto interativo que potencializa o estado de flow, essencial para o prazer e a concentração do usuário

### **2.3 Cultura lofi e retrowave: estéticas sonoras e visuais na era digital e sua influência no youtube contemporâneo**

Este capítulo aborda as culturas LoFi e Retrowave enquanto estéticas sonoras e visuais que ganharam destaque na era digital. Com forte presença em plataformas como o YouTube, essas correntes influenciam não apenas produções musicais e visuais, mas também experiências interativas, como jogos digitais.

#### ***2.3.1 LoFi como linguagem estética e emocional***

Este tópico explora as origens da cultura LoFi<sup>6</sup>, sua inserção e impacto na plataforma YouTube, e como ela molda experiências digitais e sociais contemporâneas. Além disso, é abordada a introdução de outra cultura também popular que está, em alguns casos, inserida juntamente com o LoFi, a cultura Retrowave.

A análise destaca como o LoFi, enquanto estética e fenômeno cultural, influencia as pessoas e como a cultura Retrowave atua em conjunto com ele, tanto em sua trilha sonora

---

<sup>6</sup> O nome LoFi vem de "low fidelity" (baixa fidelidade), termo que se refere originalmente à qualidade do som, em oposição ao HiFi "high fidelity" (alta fidelidade), que busca máxima precisão sonora. A estética LoFi também se estende à imagem, como na fotografia e na pós-produção, com imperfeições valorizadas como parte do estilo. Essa limitação estética, muitas vezes causada por restrições financeiras, leva artistas a utilizarem gravadores de fita cassete e outros equipamentos simples. Musicalmente, o gênero combina elementos de jazz, música eletrônica e hip-hop, sendo atualmente popularizado por transmissões ao vivo no YouTube em canais como LoFi Girl e Chillhop Music, que funcionam 24 horas por dia (BURKART; STONE, 2020).

quanto em suas decisões de design, que buscam proporcionar sensações de imersão e conforto ao usuário.

A cultura LoFi (abreviação de Low Fidelity) representa um movimento estético e musical caracterizado por sons de qualidade deliberadamente baixa, acompanhados de imperfeições, como ruídos ambientes ou distorções suaves, que evocam um sentimento de nostalgia e relaxamento. Originada na música independente dos anos 1990, essa estética deliberada pode ser compreendida como parte de um movimento cultural mais amplo, no qual escolhas musicais são influenciadas não apenas por convenções estéticas, mas também por contextos sociotecnológicos e pelas possibilidades criadas por avanços na gravação e distribuição musical (BORN, 1995).

No contexto do YouTube, o LoFi encontrou um espaço único, onde canais dedicados a transmissões ao vivo, como o icônico lofi hip hop radio - beats to relax/study to (<https://www.youtube.com/channel/UCSJ4gkVC6NrvII8umztf0Ow>), popularizaram o gênero a partir do início da pandemia, quando as pessoas passaram do trabalho presencial para o home office. Nesse período, esse tipo de conteúdo começou a ser consumido como uma forma de ampliar a concentração. Em fevereiro de 2020, a principal transmissão ao vivo "lofi hip hop radio - beats to relax/study to" havia sido vista 218 milhões de vezes, número que ultrapassou 668 milhões em julho de 2022 (WIKIPÉDIA, 2024).

. Essas transmissões combinam batidas melódicas e repetitivas com visuais minimalistas ou animados, muitas vezes remetendo a cenas de tranquilidade e introspecção, criando uma atmosfera acolhedora para os espectadores.

Segundo Neil Strauss, em *The Pop Life*:

"A cultura LoFi é uma celebração da imperfeição, um retorno ao artesanal em um mundo cada vez mais dominado pela perfeição digital" (STRAUSS, 1993, p. 45).

A experiência gerada pelo LoFi remete a saudade, e nesse contexto, ela ultrapassa a vivência direta, é um anseio por um tempo nunca vivido, mas imaginado. Muitas pessoas experimentam esse sentimento ao se conectarem com estéticas e sons que remetem a épocas passadas, como se buscassem um refúgio afetivo em um passado idealizado. É nesse ponto que Mark Fisher aborda uma reflexão sobre o LoFi e como ele contribuiu para compreender essa nostalgia fabricada e como ela transforma as músicas em um apelo emocional profundo:

"Os sons LoFi são impregnados de um senso de tempo perdido, evocando memórias fragmentadas e sensações de melancolia associadas ao passado" (FISHER, 2014, p. 20).

No YouTube, o LoFi amplia o formato musical, tornando-se parte de uma experiência multimídia. Canais como a ChilledCow, atualmente Lofi Girl, (<https://www.youtube.com/channel/UCSJ4gkVC6NrvII8umztf0Ow>) estabeleceram uma estética icônica associada ao gênero, com transmissões contínuas que atraem milhões de ouvintes buscando concentração ou relaxamento. Essa integração da música com visuais nostálgicos criou uma comunidade global em torno do gênero.

Para Jane McGonigal, em *Reality Is Broken*:

"O LoFi no YouTube é um exemplo de como as plataformas digitais podem criar espaços colaborativos que não apenas conectam pessoas, mas também promovem um estado de *flow* – um equilíbrio entre relaxamento e produtividade" (MCGONIGAL, 2011, p. 118).

A popularização do LoFi está fortemente ligada ao seu uso em contextos de estudo, trabalho e concentração. Estudos apontam que o gênero promove um ambiente sonoro que favorece o foco cognitivo, com batidas repetitivas e ausência de letras, reduzindo distrações e melhorando a retenção de informações (BURKART; STONE, 2020). De acordo com Campos (2021), a sonoridade relaxante do LoFi cria uma "atmosfera de conforto e acolhimento", funcionando quase como uma moldura sonora para atividades mentais intensas.

Canais como Lofi Girl se tornaram referência nesse cenário ao transmitirem continuamente esse tipo de música, consolidando uma relação entre LoFi e rotinas de estudo. Além disso, pesquisadores sugerem que a estética visual associada ao gênero, como animações suaves e repetitivas, também contribui para a manutenção da atenção (GOMES, 2022).

### **2.3.2 Retrowave: Estética Visual e Musical**

A cultura Retrowave, também conhecida como Synthwave, é um movimento cultural e estético que se inspira fortemente na nostalgia dos anos 1980, trazendo elementos característicos dessa época, como sintetizadores, neon, estética futurista e trilhas sonoras eletrônicas. Ela resgata a visão do futuro imaginada no passado, combinando tecnologia, design e música com uma abordagem contemporânea.

Atualmente, a Retrowave não está presente apenas na música e influencia diversas áreas, incluindo design gráfico, cinema, moda e jogos. No cenário digital, sua presença é particularmente marcante em plataformas como YouTube e redes sociais, onde vídeos e artes visuais utilizam paletas de cores vibrantes, imagens de paisagens urbanas e referências a videogames clássicos.

Essa cultura se tornou um fenômeno global, conectando pessoas que compartilham o fascínio pela estética retrowave, que se mistura com outras culturas, como no caso do LoFi, para criar experiências que proporcionam simultaneamente nostalgia e inovação. A sensação de nostalgia é frequentemente evocada por meio do uso de timbres analógicos, sintetizadores com efeitos de chorus e reverb, além de progressões harmônicas simples que remetem às trilhas sonoras de filmes e jogos dos anos 1980 (SANTOS, 2020).

Por outro lado, a inovação emerge da reinterpretação desses elementos com técnicas modernas de produção, combinando o passado estilizado com uma abordagem estética digital contemporânea (HARPER, 2019). No caso do Retrowave, por exemplo, são características marcantes o uso de sintetizadores digitais, linhas de baixo pulsantes, baterias eletrônicas marcadas (como as da Roland TR-808) e melodias com influência do synthpop<sup>7</sup> e da new wave<sup>8</sup>. Essa fusão de estilos, como a calma do LoFi com a energia visual e sonora do Retrowave, resulta em um novo tipo de expressão artística que captura a essência de um passado idealizado enquanto dialoga com as possibilidades do presente.

Segundo Vincent Limon, em *Retrowave Culture and Its Impact*:

"A Retrowave é mais do que nostalgia; é uma recriação idealizada dos anos 1980, combinando memórias culturais com as possibilidades tecnológicas contemporâneas" (LIMON, 2017, p. 32).

---

<sup>7</sup> *Synthpop* é um subgênero da música pop que surgiu no final dos anos 1970, marcado pelo uso predominante de sintetizadores, batidas eletrônicas e arranjos melódicos digitais. Tornou-se amplamente popular nos anos 1980 com bandas como Depeche Mode, Pet Shop Boys e Yazoo, sendo considerado uma das bases estéticas do som eletrônico contemporâneo (REYNOLDS, 2012).

<sup>8</sup> *New Wave* é um movimento musical que emergiu no final da década de 1970 como uma evolução do punk rock, incorporando elementos da música eletrônica, do pop e da estética visual moderna. Caracteriza-se por melodias cativantes, uso de sintetizadores, batidas dançantes e uma postura artística mais acessível e performática, influenciando diretamente gêneros como o synthpop e o synthwave (FRITH, 2004).

Para David Greig, em Synthwave: The Sound of Retro Future:

"O movimento Retrowave não é apenas sobre música, mas uma celebração de um estilo de vida baseado na estética e nos sentimentos de escapismo e euforia da cultura pop dos anos 1980" (GREIG, 2015, p. 22).

Como já mencionado, a cultura Retrowave também encontrou espaço em jogos, como em Hotline Miami (Dennaton Games, 2012), que combina gráficos estilizados, trilhas sonoras synth e narrativas envolventes.

Como Jane McGonigal sugere em Reality Is Broken:

"Elementos estéticos que evocam memórias ou nostalgia podem ser poderosos para criar experiências que conectam emocionalmente os jogadores e incentivam engajamento duradouro" (MCGONIGAL, 2011, p. 89).

É importante explorar as origens e características da cultura Retrowave, destacando sua relevância cultural e como ela foi adaptada no cenário contemporâneo. Isso será feito neste trabalho, demonstrando sua influência em conjunto com os outros elementos visuais e musicais já citados.

## 2.4 O motor gráfico Unity: fundamentos e aplicações no desenvolvimento de jogos

No processo de desenvolvimento de jogos digitais, existe a necessidade da utilização de um software próprio para produção de jogos, e as *engines*<sup>9</sup> são responsáveis por esse papel central, servindo como plataformas que simplificam a criação e o gerenciamento de elementos fundamentais, como gráficos, física, áudio e interatividade.

Para o desenvolvimento de um jogo, a escolha do motor Unity (<https://unity.com>) pode ser baseada no fato dela ter algumas funcionalidades a mais, como Shader Graph e Sistema de Partículas, comparado as demais disponíveis no mercado, como Godot<sup>10</sup> e GameMaker<sup>11</sup>, mesmo ambos acessíveis, pois a Unity possui uma versão gratuita (Unity Personal) voltada para

<sup>9</sup> A palavra "engine" em inglês significa "motor" em português. Uma engine de jogos, também conhecida como motor de jogo, é um programa de computador que fornece ferramentas para a criação de jogos.

<sup>10</sup> Godot é uma engine gratuita e de código aberto, usada para desenvolvimento de jogos 2D e 3D, com foco em acessibilidade e comunidade ativa.

<sup>11</sup> GameMaker é uma engine voltada para o desenvolvimento de jogos 2D, com interface amigável e ideal para projetos independentes.



estudantes e desenvolvedores que faturam menos de US\$100 mil por ano, ela ainda se mostra superior em relação a esses pontos específicos citados.

Já os planos pagos oferecem funcionalidades extras e suporte técnico, sem limitar o acesso às principais ferramentas essenciais na versão gratuita. Segundo o site oficial da Unity:

“Unity Personal é gratuito para criadores que tenham receita ou financiamento inferiores a US\$100 mil nos últimos 12 meses. Com ela, você pode desenvolver, lançar e operar seu jogo com acesso ao ecossistema Unity.”

Isso tornando-se a escolha ideal tanto para iniciantes quanto para desenvolvedores experientes.

Uma game engine é definida como uma estrutura de software projetada para facilitar a criação de jogos eletrônicos, oferecendo ferramentas e bibliotecas para lidar com as complexidades do desenvolvimento. Segundo Jason Gregory, em *Game Engine Architecture*:

"Uma game engine é um conjunto de subsistemas reutilizáveis que fornecem os elementos fundamentais necessários para o desenvolvimento de jogos, permitindo aos desenvolvedores focar no design e na experiência do jogador" (GREGORY, 2018, p. 23).

As engines modernas, como a Unity, são projetadas para atender a uma ampla gama de necessidades, suportando desde experiências 2D simples até jogos 3D complexos e imersivos. A Unity Technologies, empresa desenvolvedora do motor, descreve seu motor gráfico como:

"Uma plataforma de desenvolvimento que permite criar, operar e monetizar experiências interativas em 2D, 3D, realidade aumentada e realidade virtual" (UNITY TECHNOLOGIES, 2023).

No início da indústria dos jogos digitais, era comum a separação entre os sistemas de renderização gráfica e simulação física, o que exigia do desenvolvedor a integração manual entre diferentes ferramentas. Jogos utilizavam bibliotecas gráficas como o OpenGL<sup>12</sup> ou

---

<sup>12</sup> OpenGL é uma API gráfica de código aberto amplamente utilizada para renderização 2D e 3D em diferentes plataformas e sistemas operacionais.

DirectX<sup>13</sup>, e motores de física como o Havok<sup>14</sup>, o ODE<sup>15</sup> (Open Dynamics Engine) ou o Bullet Physics<sup>16</sup>, cada um com seu próprio ciclo de atualização, arquitetura e necessidades de sincronização.

Com a evolução do desenvolvimento de jogos e a crescente complexidade das experiências interativas, surgiram motores mais completos e integrados como a Unreal Engine<sup>17</sup>, a Godot, a GameMaker e a própria Unity, que passaram a incorporar tanto os sistemas gráficos quanto os físicos em um mesmo ambiente de forma nativa e sincronizada. Segundo Rabin (2010), um dos grandes avanços técnicos desse sistema foi:

“A Engines agora integram os diversos subsistemas anteriormente isolados, incluindo renderização, física e animação, tornando o desenvolvimento mais rápido, confiável e acessível” (RABIN, 2010, p. 15).

Outro destaque da Unity é sua comunidade global ativa e um vasto repositório de recursos, como o Unity Asset Store (<https://assetstore.unity.com>), que permite aos desenvolvedores integrar facilmente modelos<sup>18</sup>, texturas<sup>19</sup> e scripts<sup>20</sup>, elementos que serão explorados mais adiante. Isso facilita a prototipagem e acelera o desenvolvimento, elementos cruciais para o projeto atual de Space Vibes.

A seguir, serão abordados os conceitos fundamentais de motores gráficos de jogos, as principais características da Unity e o destaque de conceitos utilizados dentro do software no

---

<sup>13</sup> DirectX é um conjunto de APIs da Microsoft voltado para multimídia, especialmente jogos, usado em ambientes Windows, sendo conhecido pelo seu desempenho gráfico em tempo real.

<sup>14</sup> Havok é um motor de física proprietário amplamente utilizado na indústria de jogos digitais, especialmente em jogos AAA, que são produções de grande orçamento, alto padrão técnico e ampla divulgação comercial, semelhantes aos grandes filmes blockbusters. Títulos como *Halo*, *Assassin's Creed* e *The Elder Scrolls* utilizaram o Havok para simular física realista em ambientes complexos.

<sup>15</sup> ODE (Open Dynamics Engine) é um motor de física open source voltado para simulações de corpo rígido, frequentemente utilizado em jogos e aplicações de robótica.

<sup>16</sup> Bullet Physics é um motor de física de código aberto bastante adotado por sua flexibilidade e suporte tanto a corpos rígidos quanto moles, além de dinâmica de veículos.

<sup>17</sup> Unreal Engine é uma das engines mais poderosas do mercado, desenvolvida pela Epic Games, conhecida por seu realismo gráfico e alto desempenho.

<sup>18</sup> Modelos 3D são representações digitais tridimensionais de objetos, personagens ou cenários, criados em softwares como Blender, Maya ou 3ds Max. Eles são compostos por vértices, arestas e faces (malha poligonal) e podem ser animados por meio de rigging (esqueleto digital).

<sup>19</sup> Texturas são imagens 2D aplicadas sobre modelos 3D ou sprites para simular detalhes visuais, como cor, rugosidade, reflexo e transparência. Técnicas como Physically Based Rendering (PBR) utilizam mapas (Albedo, Normal, Metallic, Roughness) para realismo.

<sup>20</sup> Sprites são imagens 2D usadas em jogos para representar personagens, objetos ou elementos de interface. Podem ser estáticos ou animados (via sprite sheets).

desenvolvimento de um projeto. A escolha dessa ferramenta permitiu a implementação de mecânicas simples, porém eficazes, e a criação de uma experiência visual marcante, potencializada pelos recursos oferecidos pela Unity, evidenciando o potencial dessa ferramenta para construção de aplicações gamificadas.

#### ***2.4.1 O que são assets e sua importância no desenvolvimento de jogos digitais***

No desenvolvimento de jogos, os *assets*, palavra que vem do inglês e significa literalmente "bens" ou "recursos", referem-se aos recursos digitais utilizados no desenvolvimento, como imagens, sons, modelos e scripts. Esses elementos são fundamentais para criar uma experiência de interatividade e conexão do jogador com o jogo, alinhada com outros elementos como efeitos visuais (VFX), shaders, animações e entre outros.

Os assets podem ser definidos como todos os recursos utilizados no desenvolvimento de um jogo, incluindo modelos 3D, sprites, texturas, sons, trilhas sonoras, animações e até scripts de programação. Jason Gregory, em *Game Engine Architecture*, descreve assets como:

"Os assets são os componentes criativos que formam o conteúdo de um jogo, servindo como os blocos de construção que os desenvolvedores utilizam para criar mundos virtuais e interações" (GREGORY, 2018, p. 273).

Esses recursos podem ser criados diretamente pelos desenvolvedores ou baixados de bibliotecas especializadas como o Unity Asset Store (<https://assetstore.unity.com>), plataforma oficial da Unity Technologies que oferece recursos prontos (assets), como modelos, texturas, scripts e ferramentas, integrando-se diretamente à engine para agilizar o desenvolvimento do jogo. A Unity Technologies define os assets como:

"Arquivos e dados que você usa para criar seus jogos, incluindo gráficos, sons e quaisquer outros elementos que possam ser importados ou produzidos dentro da engine" (UNITY TECHNOLOGIES, 2023).

Além de suas funções visuais, os assets devem sempre ser otimizados para o processo de desenvolvimento. Assets bem produzidos garantem compatibilidade com o motor

gráfico, otimização de desempenho e estrutura padronizada, o que reduz falhas visuais, bugs<sup>21</sup> e retrabalho no desenvolvimento e permite que as equipes de desenvolvimento se concentrem na implementação de mecânicas e no ajuste da experiência do jogador, enquanto quem fica responsável pelo design adapta os elementos de acordo com o projeto. Segundo com Jeremy Gibson Bond, em *Introduction to Game Design, Prototyping, and Development*:

"Assets economizam tempo ao oferecer elementos pré-construídos que podem ser rapidamente integrados ao projeto, permitindo que os desenvolvedores dediquem mais tempo à inovação e ao refinamento do jogo" (BOND, 2014, p. 117).

A seguir, serão discutidos os principais tipos de *assets* utilizados no desenvolvimento de jogos digitais, como modelos, texturas, sons e scripts. Também será discutida a relevância desses recursos para a construção de experiências interativas, funcionais e visualmente consistentes.

#### **2.4.2 Modelos 3D no design de jogos: forma, volume e estilo**

Modelos 3D são representações tridimensionais de objetos, personagens ou cenários criados em softwares de modelagem e animação, como Blender<sup>22</sup>, Maya<sup>23</sup> ou 3ds Max<sup>24</sup>. Esses modelos são cruciais para jogos que utilizam gráficos tridimensionais. Os modelos 3D são compostos por vértices, arestas e faces que formam uma malha (*mesh*).

Essa malha pode receber a aplicação de texturas (imagens 2D) em superfícies 3D para adicionar detalhes visuais, como cores, padrões ou materiais. Os modelos 3D, especialmente personagens, são frequentemente “rigged”<sup>25</sup> com esqueletos para permitir movimento, o processo de adicionar um esqueleto digital (bones e joints) a um modelo 3D

<sup>21</sup> Bugs, na sua tradução "insetos", são erros ou falhas no código ou no funcionamento do jogo que causam comportamentos inesperados, como travamentos, gráficos incorretos ou mecânicas que não funcionam como previsto.

<sup>22</sup> Blender é um software livre e de código aberto para modelagem, animação e renderização 3D, mantido pela Blender Foundation. É amplamente utilizado na indústria criativa por sua versatilidade e por ser gratuito (BLENDER FOUNDATION, 2025).

<sup>23</sup> Maya é um software profissional desenvolvido pela Autodesk, voltado à animação e modelagem 3D, com forte presença nas áreas de cinema, televisão e jogos (AUTODESK, 2025a).

<sup>24</sup> 3ds Max é outra solução da Autodesk voltada para modelagem e visualização 3D, bastante utilizada em jogos, design arquitetônico e efeitos visuais (AUTODESK, 2025b).

<sup>25</sup> Rigged ou “Manipulado” na sua tradução, faz referência a possibilidade de movimentar e criar animações por meio de uma estrutura chamada “bone”(osso), semelhante ao que compõem um corpo humano, que tem a capacidade de influenciar o objeto 3D ou 2D que esteja sendo trabalhado.

permite a sua animação. Cada osso controla partes da malha, possibilitando movimentos como andar ou gesticular.. Esses modelos 3D são fundamentais para criar experiências visuais imersivas em jogos modernos, pois permitem a representação precisa e dinâmica de objetos no espaço tridimensional (WATT; POLICARPO, 2005).

O processo de rigging é essencial para transformar modelos estáticos em personagens interativos e animáveis dentro de um jogo digital. Ele permite que partes do corpo do personagem sejam controladas por um sistema de ossos, facilitando movimentos como andar, correr, pular e expressar emoções. Um exemplo desse processo é o rig desenvolvido para a personagem Spider-Gwen, da franquia Homem-Aranha.

O projeto aplicou técnicas avançadas como cinemática direta (FK), um método de animação em que os ossos são movimentados em ordem hierárquica, do topo (como o ombro) até a ponta (como a mão), e cinemática inversa (IK), permite controlar a posição final de um membro (como a mão ou o pé), e o sistema calcula automaticamente a posição dos ossos anteriores (como o braço ou a perna). Além disso, sistemas específicos como IK spline para movimentação da coluna e reverse foot para a correta articulação dos pés, características comuns em animações de personagens com alto nível de flexibilidade:

“O objetivo deste projeto foi criar um rig completo para essa personagem para animação ou jogos [...] funcionalidades: IK/FK para as pernas e braços, IK spline para a coluna, pé reverso para os pés.” (PRAGNON, 2024).

Essas técnicas demonstram como o rigging não apenas viabiliza a movimentação de personagens, mas também proporciona controle preciso e realismo na animação, elementos fundamentais para a imersão e qualidade visual em jogos digitais.

Figura 5 - Imagem do modelo 3D da Spider Gwen com rig



Fonte: <https://alexsalmar.gumroad.com/l/spidergwen>

#### ***2.4.3 Sprites no desenvolvimento de jogos: técnica e estética***

Sprites são imagens 2D utilizadas para representar objetos, personagens ou elementos em jogos bidimensionais. Eles são amplamente empregados em jogos com gráficos pixelados ou em interfaces de jogos 3D. O PNG<sup>26</sup> é o mais utilizado, devido à sua capacidade de suportar transparência. Também é possível realizar animações com sprites, feita por meio de “sprite sheets”<sup>27</sup> que contém várias imagens que, quando exibidas em sequência, criam o efeito de movimento.

<sup>26</sup> PNG (*Portable Network Graphics*) é um formato de dados utilizado para imagens, que surgiu em 1996 como substituto para o formato GIF, devido ao facto de este último incluir algoritmos patenteados.

<sup>27</sup> Sprite sheets são arquivos que contém sequências de imagens que representam o movimento de personagens e objetos em jogos ou aplicações.

Figura 6 - Sprite Sheets do jogo Megaman X



Fonte: [https://www.kindpng.com/imgv/mwToi\\_exemplo-de-spritesheet-sprite-sheet-megaman-png-transparent/](https://www.kindpng.com/imgv/mwToi_exemplo-de-spritesheet-sprite-sheet-megaman-png-transparent/)

Os sprites são elementos fundamentais em jogos 2D, responsáveis por representar visualmente personagens, objetos e ambientes. Eles permitem aos desenvolvedores construir mundos interativos com alta eficiência gráfica, pois podem renderizar imagens e animações consumindo poucos recursos do sistema, o que é ideal para dispositivos com hardware limitado. Além disso, o controle estético possibilita a personalização e estilização de cada sprite individualmente, permitindo uma grande variedade visual e coerência artística no jogo (KUMAR; RAHMAN, 2016).

#### ***2.4.4 Texturas e materiais: composição visual no espaço virtual***

Texturas são imagens aplicadas às superfícies de modelos 3D ou elementos 2D para adicionar detalhes visuais como cor, rugosidade ou brilho. O mapeamento UV é o processo de projetar uma textura em um modelo 3D. Os tipos de texturas são composto por:

O desenvolvimento visual de modelos 3D avançou significativamente graças a uma técnica chamada PBR (Physically Based Rendering), na sua tradução “Renderização Baseada em Física”, trata-se de um método de renderização que busca reproduzir de forma realista o comportamento da luz ao interagir com diferentes tipos de materiais, seguindo princípios físicos do mundo real.

Ao contrário das abordagens antigas, que dependiam de truques visuais manuais para simular reflexos, brilhos e texturas, o PBR permite criar superfícies visivelmente coerentes em qualquer condição de iluminação, seja sob o sol de um deserto ou na penumbra de uma caverna. Isso faz com que o material se comporte de forma previsível e natural, oferecendo consistência visual entre diferentes plataformas, motores gráficos e cenas.

A construção de materiais PBR é feita por meio de texturas específicas, também chamadas de mapas, que controlam aspectos como cor, relevo, reflexo, brilho e tipo de material. Os principais mapas utilizados neste processo são:

- **Albedo / Diffuse:** O mapa *Albedo*, também conhecido como *Diffuse*, na sua tradução “cor base do material”, define a aparência visual fundamental da superfície. Ele contém as cores e padrões do material, como manchas de sujeira em um tecido, veios de uma madeira ou desenhos de um azulejo. No PBR, o *Albedo* deve estar livre de sombras ou brilhos pintados, pois essas características serão geradas dinamicamente pela luz realista do sistema. Assim, ele funciona como a “pele pura” do objeto, onde estão apenas as informações visuais fixas da textura.
- **Normal:** O *Normal Map*, na sua tradução “mapa de normais”, adiciona riqueza de detalhes visuais ao objeto sem aumentar sua complexidade geométrica. Ele simula pequenas saliências, reentrâncias, vincos, rachaduras e texturas em baixo relevo que interagem com a luz, tudo isso sem adicionar polígonos ao modelo 3D. É como se o modelo tivesse uma textura em relevo esculpida, mas feita com informações visuais, o que economiza desempenho e mantém o realismo.
- **Metallic:** O *Metallic Map*, na sua tradução “mapa metálico”, determina quais partes do material são feitas de metal e quais não são. Os metais reagem à luz de forma específica: eles refletem quase toda a luz ambiente e muitas vezes “tingem” o reflexo com sua própria cor. Esse mapa costuma ser binário, na cor branca (valor 1): indica áreas metálicas, como ferro, ouro e alumínio. Na cor preta (valor 0): indica áreas não metálicas, como madeira, tecido ou plástico. Esse controle é essencial para diferenciar, por exemplo, uma espada de aço com cabo de couro — cada parte exige um comportamento de luz diferente.
- **Roughness:** O *Roughness Map*, na sua tradução “mapa de aspereza”, define o quão lisa ou áspera é a superfície do material. Ele afeta a nitidez dos reflexos: Superfícies lisas (valores escuros) refletem a luz de forma clara, como espelhos ou metais polidos. Superfícies ásperas (valores claros) dispersam a luz, gerando reflexos suaves ou difusos — como em concreto, couro ou pedra bruta. Esse mapa ajuda a transmitir visualmente a sensação tátil do objeto. Um vidro fosco, por exemplo, pode ser criado apenas ajustando a *roughness* de forma adequada.
- **Specular:** Alternativamente ao uso do *Metallic/Roughness*, alguns fluxos de trabalho usam o mapa *Specular*, na sua tradução “reflexo especular”. Ele serve para controlar o



grau e a cor do brilho reflexivo de cada parte do material, sendo mais flexível para simular superfícies não metálicas com brilho acentuado, como cerâmica envernizada, borracha ou plástico. O uso desse mapa depende do motor gráfico e do pipeline adotado, mas segue o mesmo princípio: controlar como a luz se comporta sobre a superfície.

Figura 7 - Imagem de mapas de textura



Fonte: <https://www.poliigon.com/pt-br/texture/concrete-cinder-block-wall-texture-charcoal-gray/9119>

Além dos mapas principais citados, existem outros mapas complementares que podem ser utilizados na composição de um material PBR, conforme as exigências visuais e técnicas de cada projeto. No exemplo da imagem anterior tem-se o *Ambient Occlusion* na sua tradução “oclusão de ambiente”, que simula sombras suaves em cavidades e áreas com pouca incidência de luz, contribuindo para uma maior profundidade visual.

Já o *Displacement Map*, na sua tradução “mapa de deslocamento”, modifica fisicamente a geometria da superfície, criando elevações reais no modelo, diferentemente do *Normal Map*, que apenas simula relevo. Há ainda o *ORM Map*, na sua tradução “mapa combinado de “Oclusão Rugosidade e Metálico”, uma técnica de otimização onde os três mapas

são compactados em um único arquivo, utilizando os canais de cor (RGB) para reduzir o número de texturas e melhorar o desempenho.

Diversos outros mapas também podem ser utilizados, e sua aplicação varia conforme o estilo visual e os efeitos pretendidos pelo artista, complementando o realismo e a expressividade da superfície. Cada mapa adiciona uma camada extra de controle visual ou artístico ao material, e seu uso consciente pode enriquecer profundamente a experiência visual. A aplicação de texturas adequadas pode transformar um modelo básico em um objeto visualmente rico e realista (MURDOCH, 2016).

#### ***2.4.5 Som nos jogos digitais: imersão, ritmo e feedback***

Os sons fornecem feedback auditivo, promovem a imersão e constroem a ambientação para o jogador. Os principais responsáveis por esses feedbacks são os efeitos sonoros (SFX - sound effects), que são sons curtos representando ações ou interações, como disparos, colisões ou interações com o ambiente.

Outro componente importante é a ambiência sonora, composta por sons contínuos que criam atmosfera, como vento, chuva ou ruídos espaciais. Em jogos 3D, os sons são frequentemente ajustados para transmitir sensações de direção e distância, contribuindo para uma percepção espacial mais realista. A qualidade sonora é crucial para reforçar a experiência emocional do jogador e aumentar sua imersão (COLLINS, 2008).

Outro elemento essencial na construção sonora de um jogo é a trilha sonora, que define o tom emocional e influencia o comportamento do jogador, criando conexões com o contexto narrativo em que está inserida. Ela também pode dialogar com a estética visual do jogo, complementando-a.

As trilhas podem ser compostas para repetir continuamente ou variar de acordo com o progresso do jogador, seja por mudança de fase ou situações específicas, como em uma *cutscene*. Sua função emocional é impactar diretamente o jogador, criando uma camada narrativa adicional e guiando sua jornada (JØRGENSEN, 2009).

No desenvolvimento de jogos digitais, a trilha sonora e os efeitos sonoros não desempenham apenas um papel técnico de ambientação, eles são centrais na construção da experiência sensível do jogador. A escolha de timbres, ritmos e estilos musicais influencia diretamente a forma como o jogador interpreta o espaço e os eventos do jogo. Gêneros como o

LoFi e o Retrowave, que são os estilos usados neste trabalho, são capazes evocar estados específicos de espírito, como nostalgia, introspecção ou tranquilidade.

O jogo *Unpacking* (2021), por exemplo, utiliza trilhas LoFi para induzir um clima de calma e acolhimento, aproximando o jogador de uma experiência quase meditativa. Por outro lado, o jogo *Far Cry 3: Blood Dragon* (2013) explora o Retrowave para criar atmosferas carregadas de energia e estética retrowave, resgatando emoções ligadas à cultura dos anos 1980.

A estética sonora, portanto, não apenas acompanha a narrativa visual: ela molda a experiência emocional e transforma o jogo em um espaço sensível e envolvente. Como afirma Karen Collins (2008, p. 130):

"O som nos jogos pode agir como um marcador emocional, guiando o jogador através da narrativa e influenciando a forma como ele interpreta cada momento da experiência lúdica."(COLLINS, 2008, p. 130)

Além disso, os efeitos sonoros intensificam a percepção sensorial do universo digital, funcionando como gatilhos emocionais imediatos. Eles marcam transições sutis de humor, sinalizam perigos iminentes e reforçam o sentimento de pertencimento ao ambiente do jogo.

#### ***2.4.6 Scripts de programação: fundamentos e aplicações***

Scripts são os arquivos responsáveis por armazenar toda a lógica de programação e interatividade dentro de um jogo digital. Eles definem como cada elemento, sejam eles sons, sprites, animações, modelos 3D, efeitos visuais ou a própria interface, deve se comportar e interagir com os demais. Em outras palavras, os scripts funcionam como o “cérebro” do jogo, transformando recursos estáticos em objetos dinâmicos e interativos.

A linguagem de programação utilizada varia conforme o motor gráfico adotado. Por exemplo, na Unity, a linguagem mais comum é o C#<sup>28</sup>, já integrada ao ambiente de desenvolvimento do motor gráfico desde sua instalação padrão (UNITY TECHNOLOGIES, 2023). Essa escolha se deve à sua sintaxe clara, boa performance e ampla documentação voltada

---

<sup>28</sup> C# (lê-se C-Sharp ou CSharp) é uma linguagem de programação orientada a objetos desenvolvida pela Microsoft como parte da plataforma .NET. Ela combina a expressividade da linguagem C++ com a simplicidade do Visual Basic, sendo amplamente utilizada no desenvolvimento de jogos por sua integração com a engine Unity (MICROSOFT, 2022).

para o desenvolvimento de jogos. Segundo Nestor (2013), os scripts não apenas dão vida ao jogo, mas também são essenciais para estabelecer regras, definir comportamentos e criar o fluxo de ações que tornam a experiência interativa envolvente e divertida.

A lógica de um jogo é organizada em diferentes tipos de scripts, cada um responsável por um aspecto específico da aplicação. Abaixo, destacam-se os principais:

- **Gameplay Scripts:** Controlam o comportamento dos personagens jogáveis, inimigos e elementos interativos do cenário. Definem ações como movimento, ataque, coleta de itens, interações contextuais, entre outras.
- **UI Scripts (User Interface Scripts):** São responsáveis por gerenciar todos os elementos visuais da interface do usuário (UI), como menus, botões, barras de vida, pontuação, cronômetros, caixas de diálogo e feedbacks visuais. A Interface do Usuário compreende todos os componentes gráficos com os quais o jogador interage diretamente. Esses scripts garantem que as interfaces respondam aos comandos e se atualizem de acordo com o progresso do jogo.
- **System Scripts:** Incluem scripts que operam funcionalidades mais complexas e sistêmicas, como física (gravidade, colisões, massa), inteligência artificial (comportamento de inimigos, pathfinding), controle de tempo, geração dinâmica de elementos (spawn) e simulações.
- **Audio Scripts:** Gerenciam a reprodução de efeitos sonoros e trilhas sonoras, controlando volume, ativação de sons contextuais, transições de áudio (fade in/out) e sincronização com ações do jogador.
- **Animation Scripts:** Coordenam a execução de animações em personagens e objetos, seja por meio de Animator Controllers<sup>29</sup> ou manipulando parâmetros diretamente via código.

---

<sup>29</sup> Animator Controllers são componentes da Unity responsáveis por gerenciar e organizar as animações de um personagem ou objeto. Eles permitem a definição de estados animados (como correr, pular ou atacar) e as transições entre esses estados, baseando-se em parâmetros configurados pelo desenvolvedor, como velocidade, direção ou comandos do jogador (UNITY TECHNOLOGIES, 2023).

- **Event Scripts:** Lidam com a lógica baseada em eventos, como cutscenes, gatilhos, checkpoints<sup>30</sup>, diálogos e transições de fases.
- **Input Scripts:** Capturam e interpretam os comandos do jogador (teclado, mouse, joystick, tela sensível ao toque), e os convertem em ações dentro do jogo.

Essa divisão modular facilita a organização do código e possibilita uma manutenção mais eficaz, permitindo que diferentes aspectos da aplicação sejam testados, ajustados e reaproveitados de forma independente (MCDONALD, 2020).

## 2.5 Shaders e efeitos visuais (vfx): fundamentos técnicos e aplicações em jogos digitais

Shaders e efeitos visuais (VFX) são utilizados no desenvolvimento de aplicações e contextos gráficos, seja em filmes, séries ou jogos, esses elementos são responsáveis por agregar realismo, estilo artístico e interação visual à experiência do jogador. Os shaders controlam propriedades visuais como cor, transparência, brilho, reflexo, textura e distorção, podendo simular materiais realistas como metal, vidro ou pele, ou criar efeitos estilizados como traços de pintura ou estética de desenho animado.

Já os VFX utilizam sistemas de partículas, luzes dinâmicas e animações visuais, tornam o ambiente mais vivo e responsivo, representando eventos como explosões, fumaça, feitiços, respingos de água ou alterações climáticas, além de reagirem a ações do jogador em tempo real.

Combinados, shaders e VFX são capazes de enriquecer a ambientação, como também funcionam como elementos de linguagem visual, reforçando a narrativa, o feedback das ações e a identidade estética do jogo. Segundo Jason Gregory, em *Game Engine Architecture*:

"Um shader é essencialmente um pequeno programa que executa no pipeline<sup>31</sup> gráfico, controlando a forma como as superfícies são renderizadas e como os efeitos de luz e sombra são aplicados" (GREGORY, 2018, p. 403).

<sup>30</sup> Checkpoints são pontos de salvamento intermediários dentro de um jogo que permitem ao jogador retomar o progresso a partir de determinado estágio, em vez de reiniciar desde o início. Eles são amplamente utilizados para balancear a dificuldade e proporcionar uma experiência mais fluida e menos punitiva (SCHELL, 2019).

<sup>31</sup> Pipeline, na sua tradução "linha de produção ou fluxo de processamento", refere-se à sequência organizada de etapas ou processos técnicos que os dados percorrem durante a produção de um conteúdo digital, como gráficos,

Na Unity, shaders são bastante configuráveis, permitindo aos desenvolvedores criar desde materiais simples até efeitos complexos, como sombras dinâmicas, reflexos em tempo real e distorções visuais. Assim, a Unity Technologies descreve os shaders como:

"Um conjunto de instruções que define como um material deve ser exibido. Eles são responsáveis pela aparência dos objetos no mundo virtual, indo além das texturas para criar interações visuais avançadas" (UNITY TECHNOLOGIES, 2023).

A respeito dos VFX, a visão de Jeremy Gibson Bond, em *Introduction to Game Design, Prototyping, and Development*:

"Os VFX são projetados para dar vida a eventos no jogo, aumentando a imersão e o impacto emocional ao transmitir de forma visualmente atraente o feedback das ações dos jogadores" (BOND, 2014, p. 249).

A Unity fornece ferramentas para a criação de VFX<sup>32</sup>, como o Particle System<sup>33</sup> e o Shader Graph<sup>34</sup>, o Apêndice A contém mais informações sobre esses componentes de forma detalhada, que permitem a criação de efeitos de partículas personalizados e shaders visuais interativos sem necessidade de codificação avançada.

## 2.6 O Adobe Photoshop como plataforma de criação visual digital

O Adobe Photoshop é uma das ferramentas mais consolidadas no campo do design gráfico e da criação digital. Sua versatilidade permite ao usuário explorar manipulação de imagens, pintura digital, composição fotográfica e criação de texturas com profundidade visual.

---

animações ou lógica de jogo. Em desenvolvimento de jogos, pode incluir desde a modelagem 3D até a renderização final, passando por texturização, rigging, efeitos visuais e integração com a engine (GOLINSKI; GRUBA, 2020).

<sup>32</sup> VFX ou Visual Effect Graph ou na sua tradução “Gráfico de Efeitos Visuais”, é um editor de efeitos visuais do Unity que permite criar efeitos em tempo real. Ele é baseado em nós e usa a unidade de processamento gráfico (GPU) para simular o comportamento de partículas. O VFX Graph pode ser usado para criar efeitos visuais como comportamentos simples de partículas, simulações complexas, neve, fumaça, fogo, entre outras.

<sup>33</sup> Particle System ou “Sistema de partículas”, na sua tradução, é um sistema versátil e robusto com muitas opções e configurações personalizáveis que ajudam você a criar uma quantidade ilimitada de efeitos altamente diversificados. Isso se deve a opção de criar uma a centenas de partículas de diferentes tamanhos e efeitos que podem variar de fogo a água, a simulações de fluidos ou explosões de desenho animado.

<sup>34</sup> O Shader Graph ou “Gráfico de Sombreador”, é uma ferramenta que permite que você crie shaders visualmente. Em vez de escrever código, você cria e conecta nós em uma estrutura de gráfico. O Shader Graph fornece feedback instantâneo que reflete suas alterações, e é simples o suficiente para usuários que são novos na criação de shaders.

Desde sua criação, tornou-se essencial para profissionais da arte e do design por possibilitar liberdade criativa com alto nível de detalhamento visual. Segundo Scott Kelby:

“O Photoshop não é apenas um programa de edição de imagem é uma extensão do olhar criativo, uma ferramenta que transforma ideias visuais em realidade gráfica”.(Scott Kelby, 2020, p. 15)

### ***2.6.1 A importância do Photoshop para o design de jogos digitais***

No campo do desenvolvimento de jogos digitais, o Photoshop disponibiliza recursos fundamentais para a criação de assets visuais que compõem o universo gráfico de um jogo abrangendo desde cenários atmosféricos até elementos de interface do usuário (UI). Como destaca Carlos Batti:

“O Photoshop permite a criação de ambientes digitais que não apenas ilustram, mas definem o tom e a ambientação de uma narrativa interativa, sendo uma ponte entre a imaginação e a materialidade visual no design de jogos”.(Carlos Batti, 2015, p. 44)

Diante do exposto, compreende-se que o Photoshop transcende a função tradicional de editor de imagens, consolidando-se como um instrumento essencial no repertório de qualquer designer que atua com jogos digitais.

Sua versatilidade permite não apenas a criação de elementos visuais altamente detalhados, mas também a manipulação precisa de texturas, interfaces e efeitos, aspectos fundamentais para a construção da identidade visual de um jogo. Ao proporcionar controle estético minucioso, o Photoshop contribui diretamente para o fortalecimento da experiência do usuário e para a imersão em universos narrativos e visuais.

## **2.7 Coreldraw e gráficos vetoriais: fundamentos e aplicações na produção visual**

O CorelDRAW é uma ferramenta voltada para a criação gráfica vetorial<sup>35</sup>, que permite a construção de elementos com alta precisão, escalabilidade e fidelidade visual. Essa ferramenta é amplamente utilizada no desenvolvimento de logotipos, interfaces e elementos

---

<sup>35</sup> Vetorial refere-se a um tipo de representação gráfica baseada em fórmulas matemáticas que descrevem formas por meio de pontos, linhas, curvas e polígonos. Diferente das imagens em bitmap (ou rasterizadas), os gráficos vetoriais são escaláveis sem perda de qualidade, o que os torna ideais para logotipos, ilustrações técnicas e design gráfico em geral (FARIN, 2002).

gráficos que exigem rigor na organização espacial e clareza visual. Para André Begosso:

“O CorelDRAW é uma ferramenta que possibilita ao designer organizar o espaço visual com exatidão, proporcionando controle absoluto sobre formas, cores e proporções vetoriais”.(André Begosso, 2017, p. 85)

### ***2.7.1 A utilização do Coreldraw na criação de interfaces para jogos***

A criação de interfaces em jogos digitais exige não apenas criatividade, mas também funcionalidade e clareza comunicacional. Ferramentas como o CorelDRAW permitem que elementos de interface, como botões, menus e HUDs, sejam desenvolvidos com rigor estético e técnico, contribuindo diretamente para a imersão do usuário e a usabilidade do jogo. A respeito disso:

“Um bom design gráfico em jogos deve unir forma e função, e o uso de ferramentas vetoriais como o CorelDRAW é fundamental para alcançar clareza, objetividade e beleza nos elementos visuais interativos”.(Robin Landa, 2011, p. 92)

No contexto do design de jogos, o CorelDRAW desempenha um papel igualmente relevante, especialmente na elaboração de elementos vetoriais. A clareza visual, a leveza dos arquivos e a facilidade de exportação para múltiplos formatos fazem com que a ferramenta continue sendo amplamente adotada por profissionais da indústria criativa. Sua aplicação no desenvolvimento de jogos digitais se evidencia na concepção de interfaces, logotipos, ícones e composições gráficas que precisam manter qualidade em resoluções variadas.

## **2.8 O papel das emoções no design emocional: fundamentos e aplicações**

É importante compreender e analisar as interconexões do design emocional e como introduzi-lo no design de jogos. O design emocional explora a influência do design na evocação de emoções e no impacto causado nos jogadores. Nesse sentido, são destacados dois pontos principais sobre o tema: a evolução histórica do Design Emocional e a relação entre emoção e design ao longo do tempo.

O termo "Design Emocional" foi popularizado pelo designer Donald Norman, que argumenta:



“Esse conflito entre diferentes níveis de emoção é comum no design: produtos reais oferecem um conjunto contínuo de conflitos. Uma pessoa interpreta uma experiência em muitos níveis, mas o que atrai em um nível pode não atrair em outro. Um design bem-sucedido precisa se destacar em todos os níveis” ( Donald Norman, 2008, p. 36).

Norman categoriza o design emocional em três níveis:

- **Visceral:** Refere-se à reação imediata, emocional e instintiva que ocorre ao se deparar com um produto. Está relacionado à estética e ao impacto visual. Um exemplo seria um smartphone, como o iPhone, que causa uma impressão visceral por sua construção em vidro e metal, bordas suaves e interface polida. Esse design comunica elegância e modernidade antes mesmo do usuário interagir com o dispositivo. Ele impacta visualmente de forma atraente, podendo gerar desejo ou curiosidade, aumentando a probabilidade de engajamento inicial com o produto.
- **Comportamental:** Nesse nível, o foco está no desempenho funcional e na usabilidade do produto. Aqui, é avaliada a experiência prática, como a facilidade de uso, eficiência e utilidade. Pode ser tomado como referência uma cafeteira Nespresso para exemplificar o design comportamental. É fácil de operar, com botões simples e cápsulas práticas, entregando café de qualidade com rapidez. O produto atende à necessidade funcional com um design intuitivo. Assim, a experiência positiva na interação garante satisfação funcional, incentivando o uso contínuo e fortalecendo a confiança no produto.
- **Reflexivo:** Este nível aborda os aspectos simbólicos e emocionais do produto, incluindo memórias, associações culturais e status social. Pode ser tomado como exemplo um relógio Rolex que vai além de ser funcional e esteticamente bonito. Ele carrega um significado reflexivo, representando status, sucesso e tradição. Para o usuário, ele pode simbolizar uma conquista pessoal ou uma herança familiar. O valor reflexivo cria uma conexão emocional duradoura, fidelizando o usuário e muitas vezes justificando um preço mais alto ou a lealdade à marca.

Dessa forma, compreende-se que o design não deve se limitar à funcionalidade e usabilidade<sup>36</sup>, mas também deve considerar o impacto emocional que um produto pode ter nos

---

<sup>36</sup> Usabilidade é a medida de quão fácil é usar um produto ou sistema para realizar tarefas. É um atributo de qualidade que avalia a facilidade de interação entre o usuário e a ferramenta. Essa técnica é importante para garantir que o usuário tenha uma boa experiência, seja produtivo e consiga alcançar os objetivos desejados.

usuários. A respeito disso, Norman descreve três tipos de design: o *Visceral Design*, que se refere à primeira impressão provocada pelo produto, envolvendo estética, aparência e primeiras percepções; o *Behavioral Design*, que engloba as interações do usuário com o produto, incluindo facilidade de uso, eficiência e eficácia nas tarefas; e, por fim, o *Reflective Design*, que se relaciona às emoções e significados mais profundos que o produto evoca após a interação, seja pelo impacto duradouro que tem na vida do usuário, seja pelo seu significado emocional ou simbólico.

Além disso, Paul Ekman, um renomado psicólogo e pesquisador das emoções humanas, destaca em *Emotions Revealed* que:

"As emoções são reações automáticas a estímulos externos, mas também são moldadas por experiências passadas e contextos culturais" (EKMAN, 2003, p. 45).

No contexto de jogos, essas definições se alinham com o conceito de criar experiências que não apenas entretêm, mas que também engajam emocionalmente os jogadores. Como Jesse Schell argumenta em *The Art of Game Design*:

"Jogos bem-sucedidos são aqueles que evocam uma ampla gama de emoções, desde a excitação de superar desafios até a empatia por personagens ou histórias" (SCHELL, 2014, p. 78).

A partir da análise sobre o design emocional, a integração dessa teoria em um projetos de desenvolvimento de jogos exige decisões conscientes no uso de elementos sonoros, visuais, mecânicos e outros. Esses elementos devem ser cuidadosamente planejados para evocar os sentimentos desejados, promovendo uma experiência imersiva que incentive o jogador a se engajar cada vez mais na interação com o jogo.

## 2.9 O papel das IA's na arte digital

A arte digital é um campo criativo que utiliza tecnologias digitais como meio de produção, expressão e disseminação de obras artísticas. Diferente de formas tradicionais de arte, como pintura ou escultura, a arte digital explora ferramentas computacionais para criar imagens, animações, sons, e até experiências interativas. Desde sua origem, com os primeiros

---

experimentos em computação gráfica, a arte digital evoluiu significativamente, incorporando avanços tecnológicos que ampliaram as possibilidades criativas. Segundo Christiane Paul, autora de *Digital Art*:

"A arte digital engloba práticas que utilizam processos digitais como parte essencial de sua produção e apresentação, conectando os limites entre a arte tradicional e as novas mídias" (PAUL, 2015, p. 12).

Com a chegada das Inteligências Artificiais (IAs), a arte digital entrou em uma nova fase. Ferramentas como geradores de imagens, assistentes criativos e algoritmos de aprendizado profundo tornaram-se os principais mecanismos de produção artística.

A arte generativa é o principal método de criação, que utiliza algoritmos para criar obras autônomas, onde o artista passa uma série de informações do que deve ser produzido em um prompt e a ferramenta é capaz de gerar várias imagens baseadas no que foi solicitado, em alguns casos, gerando coisas imprevisíveis. George Ness, considerado o "pai da arte generativa", destaca a singularidade desse processo:

"A arte generativa é uma colaboração entre o artista e a máquina, onde a programação não é apenas uma ferramenta, mas o próprio meio de expressão criativa" (NESS, 1974, p. 18).

Esse conceito transformou a maneira como artistas digitais interagem com suas criações, permitindo que algoritmos de IA contribuam para resultados inesperados e inovadores. Hoje, a arte generativa é amplamente utilizada em plataformas digitais e jogos, como um método mais rápido de produção de *assets*. Dessa forma, a utilização de IA na arte permite a colaboração entre humanos e máquinas para criar obras únicas.

Exemplos disso incluem ferramentas como DALL-E, MidJourney e Stable Diffusion, que geram imagens a partir de descrições textuais, ampliando a acessibilidade e a velocidade da produção artística. De acordo com Stephen McDowell, em *Artificial Creativity*:

"As IAs representam não apenas um novo instrumento para artistas digitais, mas também um agente que redefine o que significa ser criativo na era contemporânea" (MCDOWELL, 2019, p. 67).

O uso de Inteligências Artificiais na arte digital e no desenvolvimento de jogos tem proporcionado ganhos significativos em termos de agilidade e diversidade visual. Ferramentas de IA podem gerar personagens, texturas e cenários de forma rápida, facilitando o trabalho de pequenos estúdios e artistas independentes. Além disso, elas ampliam as possibilidades criativas, permitindo explorar estilos variados e automatizar tarefas repetitivas. Segundo McCormack em *Designing Creativity: Artificial Intelligence and the Future of Creative Work*:

“A IA pode ser vista como uma parceira criativa, expandindo a capacidade humana de imaginar e produzir” (MCCORMACK et al., 2019, p. 15).

Por outro lado, o uso da IA levanta preocupações éticas relacionadas à autoria, direitos autorais e à valorização do trabalho artístico. Muitos modelos são treinados com obras existentes sem o devido consentimento, o que pode comprometer a originalidade e a justiça na produção criativa. Além disso, há o risco de desvalorização do trabalho humano frente à facilidade de geração automatizada. Esses desafios destacam a importância de utilizar a IA como ferramenta de apoio e não como substituta do processo criativo humano.

No contexto de desenvolvimento de jogos, a arte digital e o uso de IAs desempenham um papel fundamental, contribuindo para a criação de elementos visuais. Por meio de ferramentas gráficas e algoritmos de geração automática, é possível produzir personagens, cenários, texturas e animações com maior rapidez e variedade.

Essa abordagem evidencia a transformação da prática artística e o impacto das tecnologias emergentes no campo do design e da estética contemporânea, destacando-se pela vasta produção de conteúdo de forma quase instantânea.

## **2.10 A teoria do Flow: fundamentos e aplicações no universo dos jogos**

A teoria do Flow, proposta por Mihaly Csikszentmihalyi na década de 1970, surgiu a partir de estudos em psicologia positiva com o objetivo de compreender como e por que as pessoas se envolvem profundamente em atividades que proporcionam prazer, engajamento e sentido. Csikszentmihalyi observou que, durante essas experiências, indivíduos relataram uma sensação de total envolvimento, perda da noção do tempo e foco absoluto na tarefa realizada.

É frequentemente descrita como "perder-se" em uma atividade, seja ela um jogo, livro, filme, ou outra forma de interação que capture completamente a atenção e as emoções do indivíduo (CSIKSZENTMIHALYI, 1990). Outro conceito abordado pela teoria é o

engajamento, que se trata do nível de atenção, interesse e envolvimento ativo que uma pessoa demonstra em relação a uma atividade, ideia, ou interação.

É uma resposta emocional e cognitiva que reflete o quão conectado e motivado alguém está em participar de algo, seja uma experiência individual ou coletiva. O conceito de Flow descreve uma experiência psicológica em que o indivíduo está totalmente concentrado em uma tarefa, sentindo-se motivado e desafiado, sem que o esforço pareça cansativo ou frustrante. Segundo Csikszentmihalyi:

"O Flow é o estado de total envolvimento em uma atividade, em que o ego desaparece, o tempo parece se distorcer e toda ação, movimento e pensamento seguem inevitavelmente da ação anterior" (CSIKSZENTMIHALYI, 1990, p. 71).

Nos jogos digitais, o Flow é frequentemente considerado essencial para proporcionar experiências gratificantes e envolventes aos jogadores. Para atingir esse estado, o design do jogo precisa equilibrar desafios e habilidades, garantindo que os jogadores se sintam constantemente estimulados, mas não sobrecarregados ou entediados. Jenova Chen, um designer de jogos renomado e criador de títulos como *Flow* (2006), *Flower* (2009) e *Journey* (2012) (CHEN, 2006; CHEN, 2009; CHEN, 2012), é um dos principais defensores da aplicação da teoria do Flow no design de jogos. Em sua dissertação de mestrado na University of Southern California, Chen adaptou o conceito de Flow para o desenvolvimento de jogos, destacando que:

"Jogos bem projetados devem funcionar como sistemas dinâmicos que ajustam automaticamente os desafios apresentados ao jogador com base em suas habilidades, mantendo-o no estado de Flow" (CHEN, 2007, p. 34).

A aplicação da teoria do Flow nos jogos é observada em mecânicas que permitem uma progressão constante e em sistemas que proporcionam feedback imediato, incentivando os jogadores a permanecerem engajados. Jogos como *Space Vibes*, por exemplo, podem se beneficiar ao integrar elementos que conduzem o jogador a esse estado. De acordo com Chris Bateman, autor de *Game Writing: Narrative Skills for Videogames*:

"O Flow nos jogos cria uma conexão emocional profunda entre o jogador e o sistema, permitindo que ele experimente satisfação contínua e senso de conquista" (BATEMAN, 2007, p. 129).

Para a produção de jogos digitais, a teoria do Flow melhora a experiência do jogador porque equilibra desafio e habilidade, mantendo-o engajado sem causar frustração ou tédio. Isso é feito ao ajustar as mecânicas de dificuldade e oferecer objetivos claros com feedback imediato, incentivando a continuidade da jogabilidade.

Além disso, o Flow reforça a imersão emocional porque promove um estado de concentração profunda, no qual o jogador se sente completamente inserido no universo do jogo. Essa imersão ocorre por meio de narrativas envolventes, ambientações consistentes e sistemas que recompensam o progresso de forma contínua.

### ***2.10.1 Equilibrando desafios e habilidades***

O primeiro ponto que se destaca no uso da teoria do Flow nos jogos é a capacidade de ajustar constantemente os desafios ao nível de habilidade do jogador. Quando a dificuldade de uma tarefa é muito baixa, o jogador pode se sentir entediado e desinteressado. Por outro lado, se a tarefa é excessivamente difícil, o jogador pode sentir frustração e desistir. O desafio é criar uma curva de aprendizagem que, ao mesmo tempo, mantenha o jogador em constante evolução, sem perder o engajamento (CSIKSZENTMIHALYI, 1990; SCHELL, 2019).

Um excelente exemplo disso é o jogo Celeste (2018), desenvolvido por Maddy Makes Games. Em Celeste, o jogador é desafiado por uma série de níveis complexos, cada um mais difícil que o anterior, mas com controles precisos e mecânicas que garantem um alto grau de controle sobre a ação. Essa combinação de desafio constante, aliado a uma habilidade crescente, mantém o jogador em um estado de Flow, pois ele sente que cada dificuldade superada é uma conquista que o aproxima ainda mais do seu objetivo final. De acordo com o criador do jogo, Maddy Thorson:

"O design de Celeste foi feito para incentivar os jogadores a se desafiarem sem sentir que estão sendo punidos por falhas" (THORSON, 2018).

Um exemplo desse processo em Celeste é o sistema de recomeço instantâneo após a morte. Quando o jogador falha, o jogo reinicia imediatamente do último ponto sem telas de carregamento ou penalidades. Isso reduz a frustração e incentiva a tentativa e erro, transformando cada falha em parte natural do aprendizado e da progressão.

### **2.10.2 Feedback imediato**

Outro princípio importante para alcançar o Flow nos jogos é o feedback imediato, um exemplo seria o efeito visual ou som ao coletar um objeto no cenário do jogo, que é essencial para criar um ciclo de ação e reação contínua, onde os jogadores podem rapidamente perceber o impacto de suas escolhas. Jogos como Super Mario Bros.

(1985) exemplificam bem esse aspecto. No jogo, cada salto, cada inimigo derrotado e cada moeda coletada oferece uma resposta imediata do sistema, que reforça o progresso e a experiência positiva do jogador, proporcionando uma sensação constante de recompensa e motivação ao longo da jogabilidade. Esse feedback constante cria um fluxo contínuo de ação e recompensa, que mantém o jogador engajado e envolvido com o jogo.

Em jogos mais recentes, como The Legend of Zelda: Breath of the Wild (2017), o feedback imediato é dado por meio da exploração, combate e resolução de quebra-cabeças. Por exemplo, ao escalar uma montanha e alcançar o topo, o jogador é recompensado com uma vista panorâmica que revela novos pontos de interesse no mapa, um estímulo visual que reforça a exploração.

No combate, ataques bem-sucedidos fazem inimigos recuar ou cair, com efeitos sonoros e visuais que sinalizam o impacto. Já nos quebra-cabeças, como nos santuários, resolver um enigma resulta na abertura de portas, ativações de mecanismos ou recompensas diretas, como orbes espirituais, proporcionando uma sensação clara de progresso e conquista.

### **2.10.3 Emoções e imersão**

O Flow não se limita apenas à mecânica do jogo, mas também se estende ao impacto emocional que ele gera no jogador. Isso é particularmente importante para jogos que buscam uma imersão profunda em um mundo fictício, como é o caso de Journey (2012). A ausência de palavras ou narrativas convencionais no jogo permite que o jogador se concentre totalmente na experiência sensorial e emocional. O designer Chen afirmou que:

"A emoção é um dos pilares centrais do design de Journey, onde o Flow não é apenas sobre mecânica, mas sobre como o jogador se conecta com o mundo" (CHEN, 2012).

Em Journey, o jogador atravessa vastos desertos, ruínas antigas e montanhas nevadas sem qualquer diálogo verbal, criando uma conexão emocional por meio da ambientação, da trilha sonora dinâmica e do encontro com outros jogadores anônimos, com quem é possível cooperar silenciosamente.

Assim, a teoria do Flow oferece uma metodologia para o design de jogos, sendo capaz de aprimorar a experiência do jogador e gerar uma maior imersão. Exemplos como Celeste, Super Mario Bros., e Journey ilustram como a aplicação do Flow pode criar experiências agradáveis ao jogador.

### **3 METODOLOGIA**



Este capítulo descreve o processo de desenvolvimento do jogo *Space Vibes*, um projeto autoral realizado pela Sector 8 Studio com o objetivo de testar a viabilidade da criação de um jogo digital em um curto período de duas semanas, utilizando ferramentas de IA como apoio ao desenvolvimento. A proposta central foi analisar se seria possível, dentro desse prazo, entregar uma experiência que combinasse as estéticas LoFi e Retrowave em um ambiente espacial imersivo, aplicando conceitos de design emocional e a teoria do Flow para evocar sensações de relaxamento e envolvimento profundo no jogador.

O jogo foi pensado para ser acessível a um público amplo, com mecânicas simples que permitissem uma fácil assimilação, seguindo a premissa de que "até uma criança conseguiria jogar e sentir as emoções propostas". Além disso, buscou-se explorar a gamificação em um contexto pouco usual, o dos vídeos LoFi no YouTube, investigando como elementos interativos poderiam ampliar a experiência contemplativa típica desse tipo de mídia.

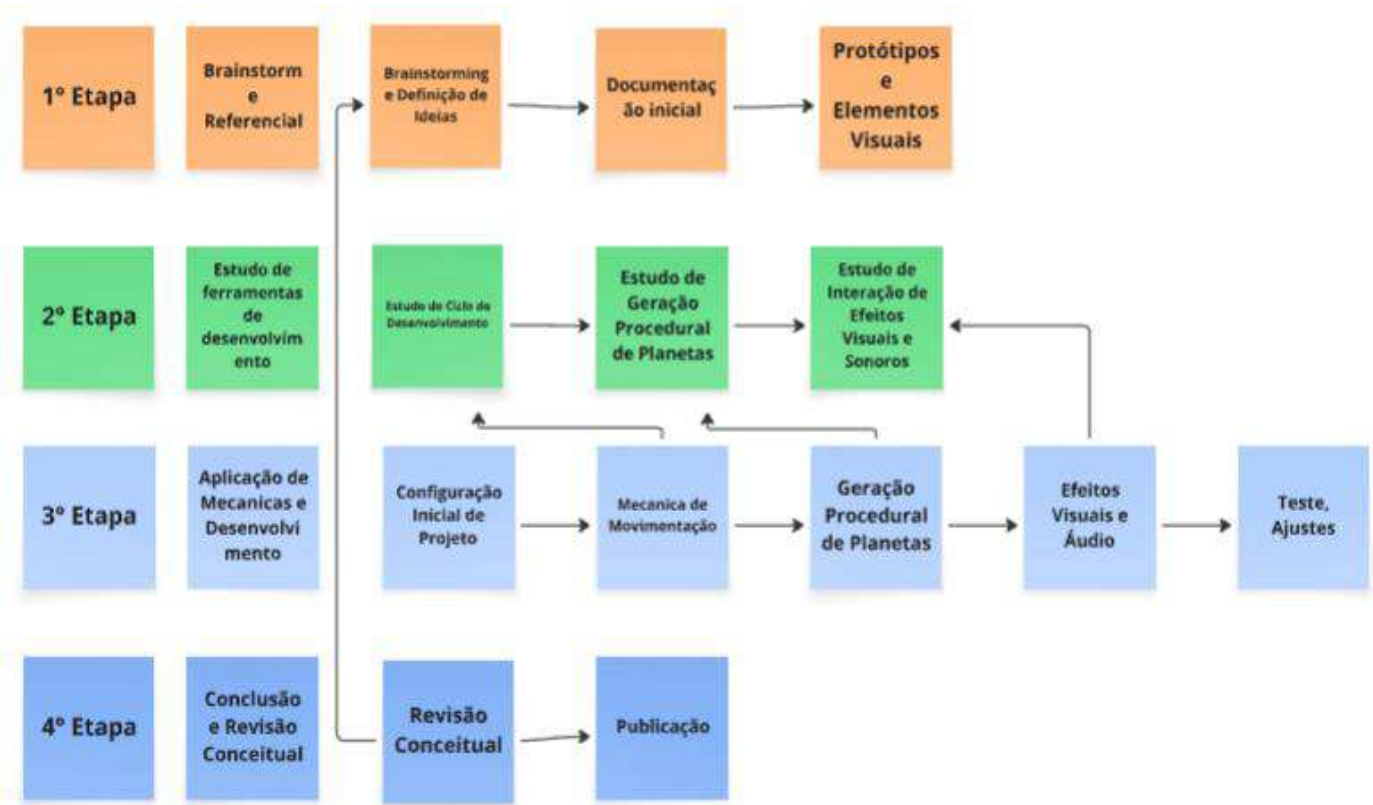
Antes do desenvolvimento do jogo, foi realizado um estudo aprofundado sobre as estéticas LoFi e Retrowave, analisando suas características visuais, como paletas de cores, elementos nostálgicos e efeitos de luz, e sonoras, batidas relaxantes, sintetizadores analógicos e atmosferas ambientais. Essa pesquisa incluiu a análise de artes conceituais e jogos que utilizam temáticas similares, com o objetivo de capturar a essência dessas culturas e traduzi-las de forma coesa na identidade visual e na trilha sonora de *Space Vibes*.

A escolha pela estética Retrowave em *Space Vibes* justifica-se pela sua capacidade de trazer cores vibrantes e contrastantes que estimulam visualmente o jogador, criando um ambiente ao mesmo tempo dinâmico e agradável. Essas cores fortes e marcantes são fundamentais para manter a atenção do usuário, atuando como um componente sensorial que complementa a proposta de combinar de forma coesa com a música e estrutura do LoFi.

De acordo com a teoria do Flow, o equilíbrio entre desafio e habilidade é essencial para que o jogador alcance um estado de concentração profunda e prazerosa. Nesse sentido, a estética Retrowave contribui para esse equilíbrio ao evitar que a experiência se torne monótona ou tediosa, estimulando visualmente sem causar sobrecarga.

A figura 8, apresenta o pipeline, o qual ilustra o fluxo de trabalho adotado durante o desenvolvimento de *Space Vibes*, desde a concepção inicial até a entrega final do protótipo jogável.

Figura 8 - Imagem do Pipeline de desenvolvimento



Fonte: Elaborado pelo autor (2022).

Para garantir a execução eficiente do projeto dentro do prazo estipulado, foi estabelecido um cronograma de desenvolvimento baseado em uma estrutura de pipeline, apresentando as etapas do processo, seu fluxo de implementação e como cada fase contribuiu para a evolução do jogo. Essa abordagem permitiu não apenas organizar as tarefas de forma sequencial e integrada, mas também identificar pontos de interação, onde certas etapas foram revisitadas para refinamento com base nos testes e feedbacks obtidos ao longo do desenvolvimento.

### 3.1 Referencial

A fase inicial do desenvolvimento do jogo Space Vibes consistiu na definição dos referenciais teóricos, técnicos e estéticos que nortearam o projeto. Foram selecionados autores e conceitos que abordam o design emocional, a teoria do Flow, gamificação, e a experiência estética nos jogos digitais. Paralelamente, foram definidos os recursos técnicos a serem utilizados: o motor gráfico Unity, a estética visual baseada no estilo LoFi, a utilização de inteligência artificial para geração de imagens e a concepção visual inspirada na estética retrô e minimalista.

A fundamentação teórica e técnica foi essencial para o desenvolvimento do dentro do cronograma apertado. A Teoria do Flow garantiu que as mecânicas fossem simples o suficiente para induzir imersão, mas envolventes o bastante para manter o jogador engajado. O Design Emocional direcionou cada escolha estética e interativa para evocar precisamente a atmosfera relaxante e nostálgica almejada.

Na esfera técnica, o Unity foi escolhido por sua versatilidade e eficiência, permitindo a criação de um jogo eficiente com o intuito desejado. O uso estratégico de IA otimizou a produção de assets visuais sem comprometer a identidade minimalista e retrô. Por fim, a estética LoFi e Retrowave se tornaram o principal referencial para o projeto. Essa combinação conseguiu unificar a sensação relaxante e nostálgica dos vídeos LoFi, e deu um novo significado: ao invés de somente assistir, o usuário agora pode interagir.

Essas definições fundamentaram a proposta do jogo como uma experiência contemplativa, pautada pela simplicidade mecânica, apelo visual e harmonia sonora. O referencial também incluiu estudos sobre a música LoFi, amplamente utilizada como ferramenta de relaxamento e foco, característica que se tornou central na construção da atmosfera do jogo.

### **3.2 Brainstorming**

Com os referenciais definidos, a fase de ideação foi iniciada com um brainstorming que consolidou os principais objetivos do projeto. A proposta central consistia na criação de um jogo desenvolvido em duas semanas, com mecânica simples e intuitiva, voltado à exploração livre em um ambiente cósmico gerado proceduralmente.

O jogo foi concebido como uma extensão interativa de vídeos musicais no estilo LoFi, com o intuito de oferecer uma experiência estética e sensorial, priorizando o acolhimento e a tranquilidade. A arte e o som foram considerados elementos fundamentais para a criação de uma atmosfera imersiva, funcionando como agentes de foco e relaxamento.

O processo de brainstorming para Space Vibes começou de forma colaborativa e descontraída, reunindo referências visuais e sonoras que capturassem a essência das estéticas LoFi e Retrowave. Através de sessões criativas, a equipe explorou desde paletas de cores e elementos visuais nostálgicos até trilhas sonoras. Ideias foram livremente compartilhadas e refinadas, priorizando aquelas que melhor traduziam a atmosfera calma desejada, enquanto

mantinham a acessibilidade para todos os públicos. Ferramentas visuais como mood boards e playlists temáticas ajudaram a consolidar a direção artística.

### **3.3 Estudo de ferramentas de desenvolvimento**

Com a concepção do projeto definida, optou-se pelo Unity como motor principal devido à sua flexibilidade e recursos especializados para jogos 2D/3D. Ferramentas como o Shader Graph foram cruciais para criar efeitos visuais personalizados, como a atmosfera de planetas e brilhos neon, típicos da estética Retrowave, sem exigir codificação complexa.

Já o VFX Graph permitiu a produção de sistemas de partículas eficientes para elementos cósmicos, como nebulosas e estrelas cadentes, essenciais para a ambientação espacial. A linguagem C#, adotada pela Unity, facilitou a implementação de mecânicas básicas, como a movimentação suave do astronauta, por ser intuitiva e amplamente documentada, acelerando o desenvolvimento dentro do prazo curto. A geração procedural de planetas, implementada via scripts em C#, ajudou a criar variações únicas de corpos celestes automaticamente, enquanto mantinha a identidade minimalista do jogo.

Para agilizar a criação de assets, utilizou-se o Leonardo.AI, ferramenta de IA generativa que produziu texturas e elementos de cenário com coerência estética, baseados em prompts alinhados ao tema LoFi/Retrowave. Isso evitou gargalos na produção manual de artes. O Photoshop e o CorelDRAW complementaram o processo, sendo empregados para editar e refinar as imagens geradas pela IA, ajustando cores, contrastes e composições para garantir harmonia visual.

### **3.4 Estudo de mecânicas e design de jogo**

A fase seguinte concentrou-se na definição das mecânicas e do design interativo do jogo. Optou-se por uma proposta de exploração livre, sem desafios complexos ou metas obrigatórias, priorizando a contemplação e a navegação suave pelo ambiente espacial.

As mecânicas foram cuidadosamente projetadas para promover um ritmo contínuo e fluido, em alinhamento com a Teoria do Flow, de Csikszentmihalyi. Esse conceito psicológico defende que atividades com desafios equilibrados, nem tão difíceis a ponto de frustrar, nem tão fáceis a ponto de entediar, induzem um estado de concentração e imersão profundas. Em Space Vibes, isso se traduziu em controles intuitivos (como movimentação suave e ausência de

obstáculos punitivos), permitindo que o jogador se concentrasse na experiência sensorial, e não em superar dificuldades artificiais.

A interface minimalista, com elementos de UI reduzidos ao essencial, e a interação não linear, em que o jogador explora o ambiente espacial livremente, sem objetivos obrigatórios ou sequências predeterminadas, reforçaram o caráter meditativo do jogo. Essa abordagem, inspirada na natureza aberta dos vídeos LoFi, convida à contemplação, transformando a jornada em uma experiência pessoal e subjetiva.

Por fim, a ambientação sonora e visual foi articulada para funcionar como extensão do conceito LoFi. As paisagens cósmicas, com tons suaves e efeitos de luz sutis, combinam-se com trilhas ambientais de sintetizadores e batidas relaxantes, criando um ciclo de feedback sensorial. Aqui, a interatividade diferencia o jogo de mídias passivas: o jogador não só ouve e observa, mas navega ativamente por essa atmosfera, tornando a imersão mais profunda e pessoal.

### **3.5 Estudo das emoções**

A última etapa da metodologia consistiu na análise das emoções que o jogo deveria provocar. Um dos seus objetivos é oferecer uma experiência emocional positiva, pautada na calma, no foco e na ausência de ansiedade. O jogo foi intencionalmente construído para evitar estímulos estressantes, adotando uma estética suave, trilhas sonoras relaxantes e a ausência de punições ou recompensas tradicionais.

A música LoFi foi utilizada como elemento central para indução de estados emocionais tranquilos, em sinergia com os componentes visuais. A Teoria do Flow, proposta por Csikszentmihalyi (1990), foi aplicada como embasamento conceitual para compreender e orientar o engajamento do jogador, mesmo em um ambiente que prescinde de desafios convencionais.

Dessa forma, Space Vibes configura-se como uma proposta de jogo sensorial e emocional, construída com base em estudos contemporâneos de design e experiência interativa, priorizando a imersão subjetiva e o bem-estar do jogador.

## 4 DESENVOLVIMENTO DO JOGO SPACE VIBES

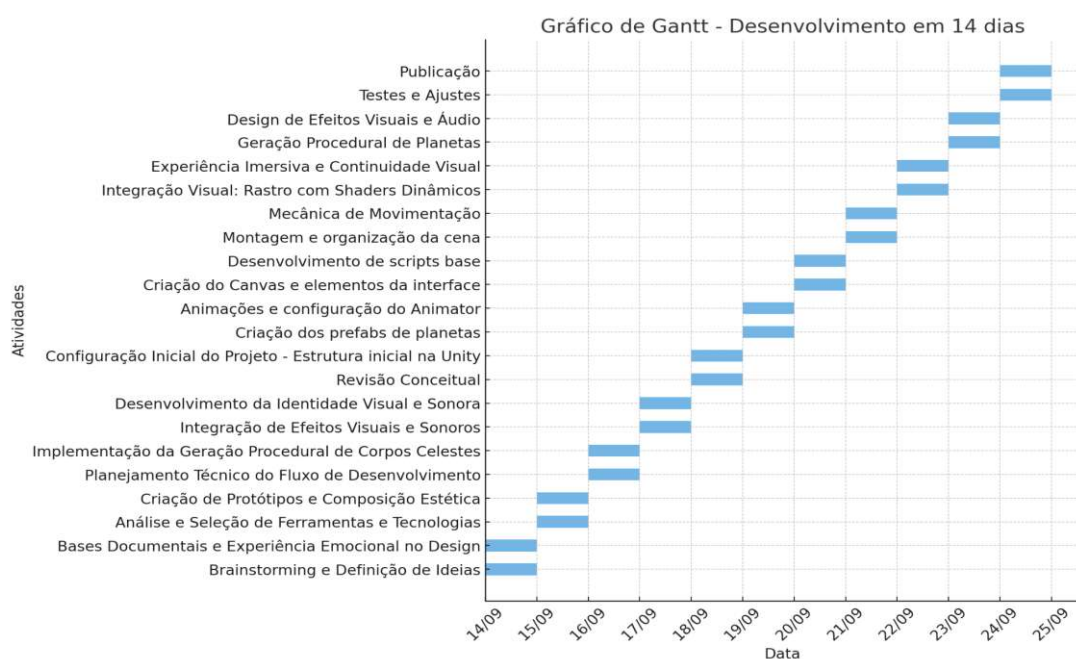
Este capítulo apresenta o processo de desenvolvimento do jogo *Space Vibes*, detalhando as etapas de concepção, design visual, implementação técnica e ajustes finais. A proposta busca integrar estética e jogabilidade de forma fluida, explorando referências das culturas LoFi e Retrowave para criar uma experiência imersiva e relaxante.

### 4.1 Semana 1: planejamento e conceituação - brainstorming e definição de ideias

O desenvolvimento de *Space Vibes* teve início em 14 de setembro de 2022, seguindo um cronograma intensivo de duas semanas, dividido em duas etapas principais de sete dias cada. A primeira semana foi dedicada à prototipagem, definição de assets e testes conceituais; a segunda concentrou-se na implementação técnica, refinamento visual e ajustes finais de jogabilidade.

Para garantir a eficiência do processo em um prazo tão restrito, foi adotada uma abordagem baseada em sprints curtos, com metas diárias alinhadas ao pipeline de produção. A utilização de ferramentas como Unity, inteligência artificial generativa e softwares de design assistido permitiu otimizar o tempo de desenvolvimento, sem comprometer a qualidade visual nem a fluidez da experiência proposta.

Figura 9 - Gráfico de Gantt do projeto Space Vibes



Fonte: Elaborado pelo autor (2022).

A gestão do tempo foi um fator decisivo para o cumprimento das metas estabelecidas. O gráfico de Gantt, apresentado na figura 9, mostra as tarefas realizadas ao longo das duas semanas, destacando as fases do desenvolvimento.

Figura 10 - Imagens com estética a) LoFi e b) retrowave

a)



Fonte: <https://dataismo.com.br/synth-boy-personagem-quando-a-lofi-girl-parou/>

b)



Fonte: <https://open.spotify.com/intl-pt/track/6ZlavGynMLNn29gZVXxhqi>

A pesquisa estética visual para o desenvolvimento de *Space Vibes* foi conduzida majoritariamente por meio de buscas direcionadas no Google Imagens, com o intuito de mapear os principais elementos gráficos das estéticas Retrowave e LoFi. Utilizaram-se palavras-chave

como “Retrowave art style”, “Synthwave backgrounds”, “LoFi aesthetic scene” e “LoFi ambient art”, o que permitiu reunir uma variedade de imagens com composições típicas dessas correntes visuais.

As imagens coletadas apresentavam padrões visuais recorrentes, figura 10 a) e b), como paletas de cores neon (predominantemente rosa, roxo e azul), elementos retrô-futuristas (grades geométricas, pores do sol estilizados, carros esportivos dos anos 1980), além de filtros que simulam ruídos visuais ou efeitos analógicos. Esses aspectos foram utilizados como base para a construção dos cenários e da interface do jogo. A seguir, apresenta-se um exemplo direto da busca por “Retrowave art style” no Google Imagens.

Complementarmente, utilizou-se também a plataforma Pinterest como fonte de curadoria estética. Foram observados diversos painéis dedicados à cultura visual LoFi, muitos dos quais apresentavam cenas intimistas com personagens em ambientes urbanos noturnos, iluminação suave, elementos climáticos (como chuva ou neblina) e objetos cotidianos dispostos de forma simbólica, como notebooks, fones de ouvido e janelas abertas para paisagens tranquilas.

Essa análise contribuiu significativamente para a definição da atmosfera que o jogo busca evocar, alinhando-se ao objetivo de construir uma experiência sensorial imersiva, próxima da cultura LoFi amplamente difundida em vídeos e transmissões ao vivo.

Figura 11 - Imagens do jogo a) Retro Drive e b) Synthwave Fury

a)



Fonte:

[https://store.steampowered.com/app/2142380/Retro\\_Drive\\_Revamped/?l=portuguese](https://store.steampowered.com/app/2142380/Retro_Drive_Revamped/?l=portuguese)



b)



Fonte:

<https://store.steampowered.com/agecheck/app/1984710/?l=brazilian>

Com relação às referências de jogos, foi realizada uma análise de títulos que adotam a estética Synthwave e elementos da cultura LoFi, com o objetivo de compreender como essas linguagens visuais e sonoras têm sido aplicadas em experiências interativas. Inicialmente, a busca foi feita por meio do Google, utilizando palavras-chave como “jogos Retrowave”, “jogos Synthwave”, “LoFi games” e variações como “games with LoFi aesthetics” ou “Retrowave indie games”.

Além disso, foram consultadas plataformas de distribuição digital, como a Steam, que possibilitaram o levantamento de jogos categorizados dentro dessas estéticas. Os critérios de seleção consideraram: presença marcante da identidade visual Synthwave ou LoFi, uso de trilha sonora coerente com essas estéticas e foco em ambientações imersivas ou experiências sensoriais.

Entre os títulos identificados, destacam-se *Retro Drive: Revamped*, desenvolvido pela Retro Drive Studio (RETRO DRIVE STUDIO, 2021), figura 11 a), que adota fortemente a estética Retrowave como elemento central, e *Synthwave FURY*, da New England Interactive (NEW ENGLAND INTERACTIVE, 2020), figura 11 b), que explora de forma intensa os visuais e trilhas sonoras Synthwave, ambos disponíveis na Steam. Essas análises serviram como base para identificar tendências, evitar abordagens redundantes e orientar a originalidade estética e sonora da proposta.

Figura 12 - Imagem do jogo Spirit City: LoFi Sessions



Fonte: [https://store.steampowered.com/app/2113850/Spirit\\_City\\_Lofi\\_Sessions/](https://store.steampowered.com/app/2113850/Spirit_City_Lofi_Sessions/)

Outro título identificado durante a pesquisa, voltado para a cultura LoFi, foi *Spirit City: LoFi Sessions* (MOONCUBE GAMES, 2024), figura 12, desenvolvido pela Mooncube Games e distribuído na plataforma Steam. O jogo propõe uma experiência focada na organização pessoal e no bem-estar do usuário, permitindo a personalização tanto do personagem quanto do ambiente virtual.

O principal objetivo é auxiliar na gestão do tempo e das atividades diárias, possibilitando ao jogador criar listas de tarefas, configurar temporizadores personalizados e manter diários de rotina. Além disso, enquanto essas tarefas são realizadas, o usuário pode ativar o modo de reprodução de músicas LoFi, mantendo o jogo em segundo plano como um ambiente sonoro e visual relaxante, semelhante às tradicionais lives da LoFi Girl.

Figura 13 - Imagem do jogo LoFi Katana



Fonte: [https://www.meta.com/pt-br/experiences/lofi-katana/6640896199298377/?srltid=AfmBOopCgA2-dQeEYlMoKRruYngMSPiiQ-fPnp3142fvLOoltLNo0O\\_d](https://www.meta.com/pt-br/experiences/lofi-katana/6640896199298377/?srltid=AfmBOopCgA2-dQeEYlMoKRruYngMSPiiQ-fPnp3142fvLOoltLNo0O_d)

Outro exemplo relevante que apresenta uma estética semelhante à de *Space Vibes* é *LoFi Katana* (LoFi Clan, 2023), figura 13, desenvolvido pelo estúdio LoFi Clan e disponibilizado na loja da Meta. O jogo combina elementos visuais do estilo Retrowave com uma trilha sonora LoFi, criando uma atmosfera imersiva que também busca relaxar o jogador por meio da estética audiovisual.

No entanto, *LoFi Katana* se diferencia significativamente na proposta mecânica: trata-se de uma experiência em realidade virtual, na qual o jogador precisa cortar objetos ou se movimentar no ritmo da música para acumular pontos, seguindo a lógica de jogos de ação rítmica.

Enquanto *Space Vibes* foca não só na estética, mas também em oferecer uma jogabilidade voltada para gerar a sensação de *Flow*, *LoFi Katana* demanda maior atenção e precisão física, oferecendo um desafio mais ativo e dinâmico.

Conforme destacado por Salen e Zimmerman (2004), um bom design de jogo deve promover interações significativas a partir de sistemas coerentes e mecânicas bem definidas. Com base nessa premissa, a etapa de prototipação visual de *Space Vibes* foi iniciada com foco na simplicidade e na experimentação livre. O primeiro esboço do jogo foi desenvolvido no aplicativo Paint, uma ferramenta de desenho vetorial básica, utilizada intencionalmente para registrar ideias iniciais de forma rápida e descomplicada. Essa abordagem permitiu testar a distribuição espacial dos elementos, a relação entre figura e fundo e a identidade visual do personagem principal.

A figura do astronauta é frequentemente associada à solidão e à introspecção, dada sua posição de distanciamento extremo da Terra e da convivência humana, sendo utilizada como metáfora em diversas produções culturais (IHMASTRONOMY, 2023). Essa mesma simbologia é observada em depoimentos de astronautas coletados pela NASA, nos quais relatam experiências de confinamento e reflexão durante longos períodos no espaço (CAPRICHIO, 2020).

A escolha por um astronauta vem da imagem arquetípica do explorador solitário, flutuando silenciosamente no espaço, uma metáfora visual que ressoa com a proposta de introspecção e relaxamento do jogo. Essa representação está profundamente enraizada na cultura visual e narrativa ocidental, onde o astronauta simboliza tanto o desconhecido quanto a jornada interior (REES, 2021).

Já o ambiente foi estruturado com base em composições espaciais amplas, fundos com estrelas, e uma paleta de cores que remete ao estilo Retrowave, como tons neon de roxo, azul e rosa. A combinação desses elementos buscou criar uma ambientação coerente e imersiva, que apoiasse tanto a mecânica quanto a proposta sensorial e estética do projeto.

## **4.2 Bases documentais e experiência emocional no design**

No dia 15 de setembro de 2022, foi elaborado o primeiro documento oficial do projeto. A equipe incorporou referências extraídas de vídeos LoFi no YouTube, imagens com estética Synthwave encontradas na internet e jogos lançados à época com propostas semelhantes, adaptando esses elementos ao objetivo de integrar componentes que evocassem relaxamento e contemplação.

Essa abordagem foi fundamentada na teoria do design emocional de Donald Norman (2008), já apresentada na seção 2.11, “O Papel das Emoções no Design Emocional: Fundamentos e Aplicações”, com os conceitos de design visceral, comportamental e reflexivo.

Para embasar a pesquisa estética do Space Vibes, foram analisados os principais canais de LoFi disponíveis no YouTube, que representam referências sólidas em termos de identidade visual, público engajado e relevância cultural. A seleção desses canais considerou fatores como número de inscritos, volume de visualizações recentes, taxa de engajamento e a coerência estética e diversidade cultural presente em cada conteúdo.

Essa análise permitiu identificar padrões visuais e sonoros que contribuem para a construção da atmosfera relaxante e contemplativa desejada no jogo, garantindo que a proposta

artística estivesse alinhada às tendências e expectativas do público-alvo da cultura LoFi. A tabela a seguir contém dados de engajamento dos canais utilizados como referência para este trabalho:

Tabela 2 - Tabela de referência dos principais canais de LoFi no Youtube

Nome do Canal	Nº de Inscritos (aproximado)	Destaques Visuais/Sonoros	Link para o canal
<b>LoFi Girl</b>	+14 milhões	Transmissões 24 horas por dia, personagem icônico, estética calma	<a href="https://www.youtube.com/c/lofigirl">https://www.youtube.com/c/lofigirl</a>
<b>Chillhop Music</b>	+3 milhões	Cores vibrantes, foco instrumental, cenas urbanas	<a href="https://www.youtube.com/c/Chillhopdotcom">https://www.youtube.com/c/Chillhopdotcom</a>
<b>College Music</b>	+1,3 milhão	LoFi com foco estudantil, transmissões longas	<a href="https://www.youtube.com/c/CollegeMusic">https://www.youtube.com/c/CollegeMusic</a>

Fonte: Elaborado pelo autor (2022).

### **Crítérios e Processo de Seleção**

- **Relevância histórica e consolidação estética:** os canais foram selecionados por sua importância na consolidação do gênero LoFi enquanto estética audiovisual, utilizando animações em loop, personagens estáticos e paletas suaves como assinatura visual. LoFi Girl, anteriormente conhecido como ChilledCow, é reconhecido como pioneiro nesse formato e estabeleceu o ícone visual da “garota estudando” que se tornou referência global (BROWN, 2021; LOFI GIRL, 2025).
- **Consistência na curadoria musical e identidade visual:** os canais Chillhop Music e College Music foram escolhidos pela qualidade e regularidade em suas curadorias musicais, aliadas a visuais animados que mantêm coerência com o ambiente introspectivo e urbano típico do LoFi. Chillhop se destaca por vinhetas sazonais que

reforçam o sentimento de reconexão emocional e nostálgica (MCINTYRE, 2020), enquanto College Music se concentra em trilhas contínuas voltadas ao foco e ao estudo (COLLEGE MUSIC, 2025).

- **Alinhamento com os princípios de imersão e foco do jogo:** os três canais apresentam elementos sonoros e visuais compatíveis com a proposta do jogo *Space Vibes*, que busca provocar estados de relaxamento, concentração e imersão através de estética espacial e ritmo contemplativo. Tais características estão associadas a estudos sobre experiência ótima e fluxo (CSIKSZENTMIHALYI, 2008).

Dito isso, pensou-se em trazer o contexto de interatividade dos jogos para um ambiente onde praticamente não há interatividade: os vídeos LoFi no YouTube. A temática espacial se encaixa com o estilo Retrowave e, consequentemente, com a estética LoFi por meio da música. Dessa forma, a ideia foi definida: o jogo será ambientado em um espaço iluminado por planetas e efeitos, tendo como personagem principal um astronauta perdido no espaço. Esses elementos apresentarão cores que remetem à estética Retrowave.

O propósito do jogo é vagar por esse espaço e admirar os planetas, nebulosas e outros fenômenos cósmicos, enquanto o usuário se imersa no contexto por meio da música e, consequentemente, se sente relaxado.

#### 4.3 Análise e seleção de ferramentas e tecnologias

No dia 16 de setembro de 2022, foi realizado um levantamento técnico para a definição das ferramentas a serem utilizadas no projeto, considerando o prazo, a complexidade do desenvolvimento e os objetivos estéticos e funcionais.

A escolha do motor Unity foi estratégica, devido à sua robustez no desenvolvimento de jogos 2D e à ampla compatibilidade multiplataforma, especialmente com dispositivos Android, plataforma-alvo do projeto. Além disso, a Unity oferece suporte avançado a efeitos visuais em tempo real, como o Particle System e o Shader Graph, recursos essenciais para a construção de elementos atmosféricos como estrelas cadentes, nebulosas e outros, que reforçam a ideia estética do jogo.

Para a criação dos assets visuais, foi adotado o uso da inteligência artificial Leonardo.AI, uma ferramenta que permite gerar imagens com base em prompts descritivos. A escolha se deu pela capacidade da IA de acelerar o processo de prototipação visual, fornecendo variações estilísticas e iconográficas coerentes com o conceito estético do jogo. Os elementos

gerados, como o astronauta, os planetas e o cenário espacial, foram cuidadosamente selecionados com base em sua adequação à identidade visual proposta.

Complementarmente, softwares de edição de imagem como Adobe Photoshop e CorelDraw foram utilizados para refinar os materiais criados pela IA. Essas ferramentas permitiram ajustes detalhados, como a separação de camadas do personagem e a vetorização de certos elementos para garantir a qualidade visual nas animações. O trabalho realizado nesses softwares teve como foco adaptar os assets para a criação das animações dentro do Unity Animator Controller, preservando a consistência estética e garantindo fluidez nas transições visuais.

#### **4.4 Criação de protótipos e composição estética**

A partir do dia 17 de Setembro de 2022, começaram os protótipos de design do astronauta, planetas e cenários. O uso de VFX e shaders foi analisado como uma opção para criar atmosferas imersivas, alinhando-se à estética planejada. Como destacado por Gregory (2018), elementos visuais bem planejados intensificam a experiência do jogador.

Para garantir uma identidade visual consistente e economicamente viável, optou-se pelo uso de ferramentas de inteligência artificial generativa, o Leonardo.AI, para a criação dos principais elementos gráficos do jogo. Essa abordagem permite explorar variações estéticas rapidamente, promovendo liberdade criativa e economia de tempo, ao mesmo tempo em que mantém o alinhamento com a proposta sensorial e contemplativa de Space Vibes.

A seguir, detalham-se os principais assets a serem gerados por IA e suas características:

##### **1. Personagem Principal (Astronauta)**

- Descrição: Figura estilizada, com traços suaves e uma paleta que mistura tons frios (azul, lilás, branco). Representa a solidão contemplativa e será posteriormente animado no Unity Animator Controller.
- Características: Design minimalista, expressão serena, traje espacial levemente futurista.

##### **2. Planetas**

- Descrição: Planetas com variações de cor e textura que compõem o pano de fundo da jornada.
- Características: Diversidade visual (um gasoso, um rochoso, outro com anéis), cores suaves, texturas atmosféricas.

### 3. Background

- Descrição: Um céu estrelado dinâmico, com nebulosas e galáxias ao fundo.
- Características: Tons escuros com manchas de cor neon (roxo, rosa, azul), deve sugerir imensidão e calma.

### 4. Satélites Naturais

- Descrição: Objetos pequenos se encontram próximos de alguns planetas.
- Características: Utiliza os *sprites* de um planeta simples, paleta monocromática, como tamanho reduzido.

Durante o processo de prototipação visual de Space Vibes, foi definida uma estética consistente com a proposta retrowave. As imagens foram produzidas por meio de prompts cuidadosamente elaborados em inglês, compostos por frases e palavras-chave que direcionavam o estilo, as cores e o nível de detalhe das composições. Cada uma dessas figuras geradas contribuiu com elementos únicos ao ambiente visual do jogo, e suas características foram planejadas para serem facilmente adaptáveis à lógica procedural dos sistemas implementados na Unity.

A figura 14 a) foi gerada com o seguinte prompt: “Planet 2d art, black background, small, white and silver colors, digital painting, retro aesthetic, stroke lines, moon around”. O resultado gerou planetas em preto e branco com detalhes sutis e traços marcados, ideais para a posterior aplicação de personalização via código. Os elementos derivados dessa figura foram convertidos em prefabs e coloridos dinamicamente por meio de scripts, utilizando uma lógica que assegura que cada planeta possua combinações únicas de cor e forma, contribuindo para a variabilidade visual do jogo.

A figura 14 b) utilizou o prompt: “Planet 2d art, black background, small, heavy strokes, neon colors, digital painting, retro aesthetic, stroke lines, similar earth planet”. Diferentemente da figura anterior, os planetas gerados já vinham com cores vibrantes, alinhadas à estética neon que permeia o jogo. Essa figura trouxe uma variedade de formas planetárias com forte presença visual, que remete a representações estilizadas da Terra em um contexto retrô digital.

Por fim, a figura 14 c) foi criada a partir do prompt: “A perfectly round circle planet 2d, black background, small, 4k, unreal engine 5, beautiful colors, atmosphere, neon colors, digital painting”. Essa composição resultou em uma imagem com aparência quase hipnótica, semelhante a um buraco negro envolto em luzes atmosféricas coloridas. As cores fortes e os



contrastes evidentes a tornaram um dos elementos centrais na ambientação espacial do jogo, fortalecendo a estética visual que mescla o LoFi com o Retrowave.

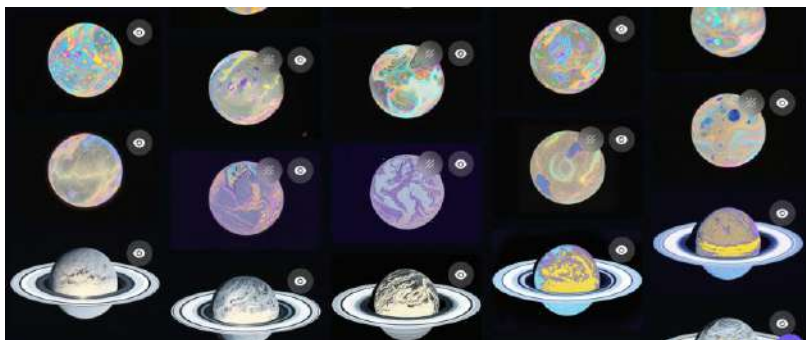
Figura 14 - Imagens dos planetas gerados por IA

a)



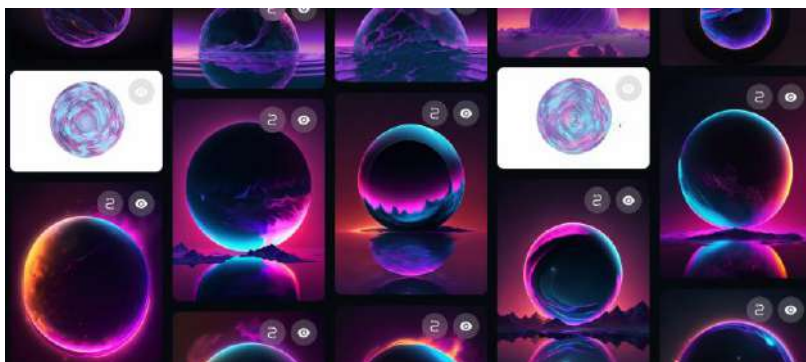
Fonte: <https://app.leonardo.ai/>

b)



Fonte: <https://app.leonardo.ai/>

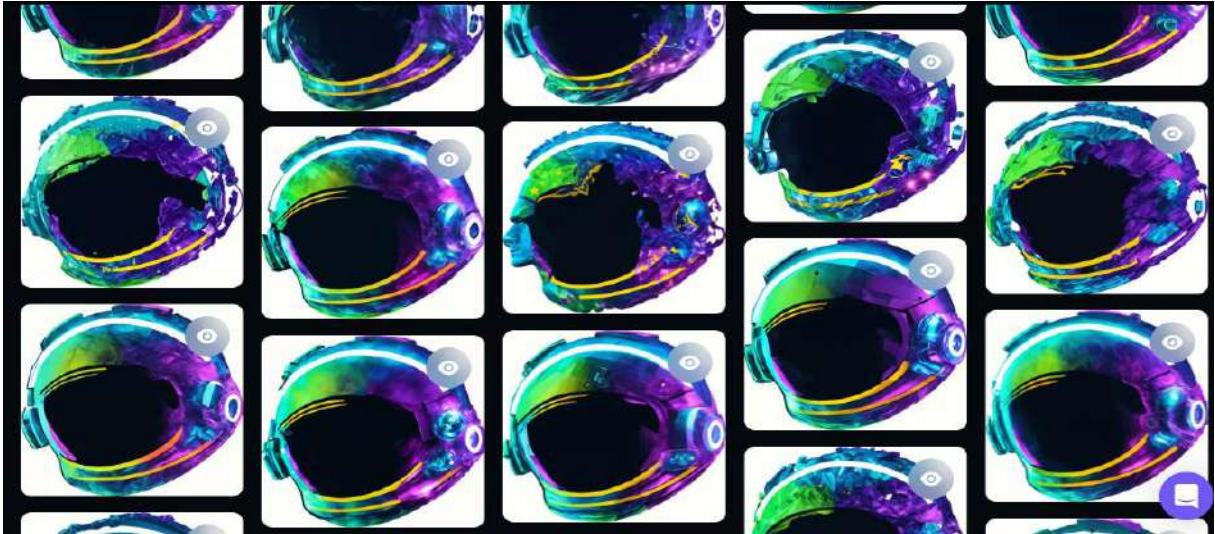
c)



Fonte: <https://app.leonardo.ai/>

Assim como os planetas, o personagem foi gerado por meio de prompt e seguiu a lógica estética proposta para o jogo. O astronauta foi dividido em partes para que pudesse ser melhor trabalhada a sua animação, utilizando a ferramenta Animator da Unity usando Bones para facilitar a movimentação ao criar as animações.

Figura 15 - Imagens do astronauta geradas por IA



Fonte: <https://app.leonardo.ai/>

Para a cabeça, foi usado o prompt: "2D vector illustration of a futuristic astronaut helmet in Retrowave style, featuring neon green, electric purple, and deep blue colors. Smooth shapes, glowing highlights, retro-futuristic details, clean vector lines, isolated on a dark background." e para o restante do corpo: "2D vector illustration of a full astronaut suit body (torso, arms, legs) in Retrowave style without the helmet. Includes neon green, electric purple, and deep blue elements, glowing lines, sleek sci-fi armor design, futuristic panels, clean vector art, isolated on dark background."

As imagens geradas por inteligência artificial, utilizadas como base para os planetas e demais elementos visuais do jogo, serão apresentadas posteriormente na seção de resultados.

Figura 16 - Asset final do personagem "Astronauta"

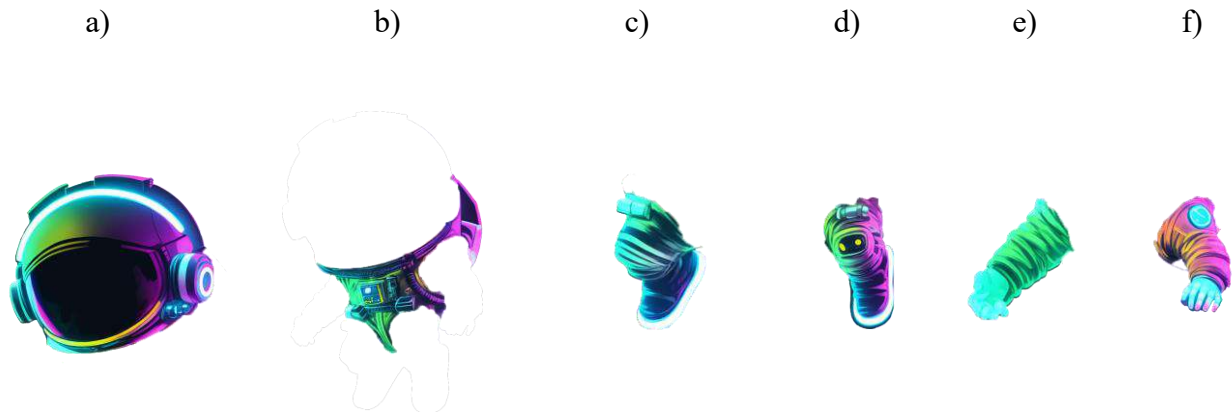


Fonte:

<https://drive.google.com/file/d/1NbX6ugVf9f5w5WNfCVBXuZG5xxl9U9hw/view?usp=sharing>

Como resultado, o asset final gerado pela ferramenta foi posteriormente dividido no software Photoshop, por meio da ferramenta laço, em cabeça, torso, braços e pernas para melhor usabilidade no processo de desenvolvimento de animações dentro da Unity.

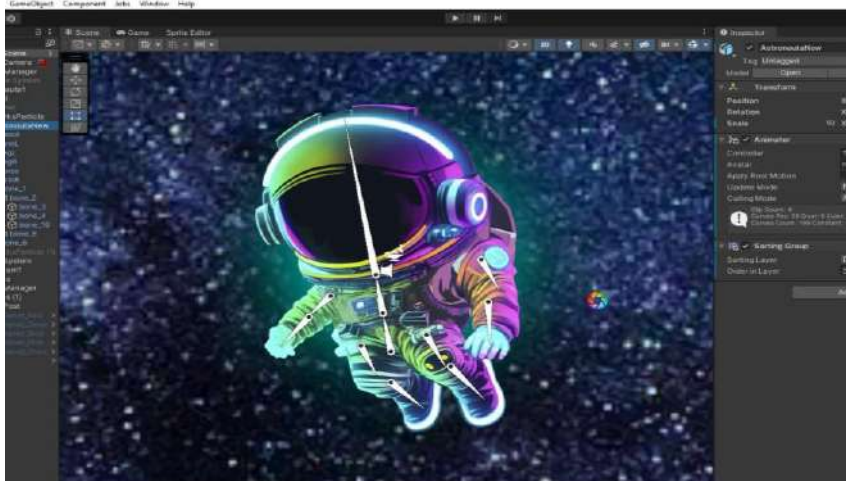
Figura 17 - Partes divididas do astronauta a) cabeça , b) dorso, c) perna direita, d) perna esquerda, e) braço direito e f) braço esquerdo



Fonte: [https://drive.google.com/drive/folders/1lf\\_eMeg4tYjMNXRy7hvXJ8QYZxfyLjB-?usp=sharing](https://drive.google.com/drive/folders/1lf_eMeg4tYjMNXRy7hvXJ8QYZxfyLjB-?usp=sharing)

As partes do personagem astronauta, representadas nas Figuras 17 (a a f), foram unidas dentro da Unity com o objetivo de criar um rigging funcional para animação. Para isso, utilizou-se a ferramenta 2D Animation Package, que permite configurar esqueletos e hierarquias de ossos em sprites 2D. Com essa abordagem, foi possível atribuir movimentos suaves e controlados ao personagem, resultando na animação apresentada na Figura 18, que representa o rigging completo e funcional dentro do ambiente do jogo.

Figura 18 - Partes unidas e com o rigging



Fonte:

<https://drive.google.com/file/d/1sGVgMT5D7td8OVG8gONIGHkiCxHGrDho/view?usp=sharing>

## 4.5 Planejamento técnico do fluxo de desenvolvimento

No dia 18 de setembro de 2022, a equipe de desenvolvimento deu início à organização do fluxo de trabalho do projeto Space Vibes, com foco na criação de uma arquitetura de desenvolvimento orientada a objetos robusta, clara e escalável. A intenção principal foi estabelecer uma base sólida para o código-fonte, de modo que o projeto pudesse evoluir de forma coesa e adaptável às futuras implementações. A estruturação modular visa não apenas facilitar a reutilização de componentes, mas também garantir maior controle sobre a complexidade crescente do sistema, considerando os desafios de sincronização entre gameplay, interface e visual. As decisões iniciais foram estruturadas da seguinte forma:

- **Definição das classes principais do jogo:** Foram identificadas as entidades centrais responsáveis pela lógica de funcionamento do jogo, como o jogador, os planetas, os obstáculos e os controladores de estado. Cada classe foi desenhada para manter um papel único, promovendo o princípio de responsabilidade única e facilitando testes unitários e manutenção do código. As relações entre essas classes estão detalhadas no Apêndice A, com esquemas explicativos.
- **Planejamento da ordem de implementação:** Com base em princípios de integração contínua e testes iterativos, definiu-se uma sequência lógica para o desenvolvimento dos módulos, priorizando aqueles que garantiriam testes mais rápidos e isolados. A ordem favorece a validação incremental de funcionalidades e permite ajustes mais ágeis conforme feedbacks internos são recebidos. A hierarquia de implementação leva em consideração os componentes do motor Unity, os quais também se encontram detalhados no Apêndice A.

### 4.5.1 Movimentação adaptada ao mobile

A entrada de dados, em vez de ser capturada pelos métodos tradicionais da Unity, passou a ser realizada por meio da leitura dos valores provenientes do componente da extensão Joystick Pack, fornecida pela Unity Asset Store. A adoção dessa abordagem justifica-se pela necessidade de adaptar a jogabilidade do Space Vibes à interface de dispositivos móveis, considerando que o projeto tem como plataforma-alvo o sistema Android.

Esse joystick virtual foi instanciado como uma variável pública no script `PlayerController`, permitindo o acesso às suas propriedades `Horizontal` e `Vertical`. Dessa forma,

o vetor de entrada (moveInput) era atualizado continuamente no método Update(), refletindo em tempo real a posição atual do joystick na tela, visível na figura 19.

Figura 19 - Imagem contendo o joystick no jogo



Fonte: <https://drive.google.com/drive/folders/1Rimtb6MJ7f1dYp-RFmkocwciT1Nr0NOE?usp=sharing>

#### ***4.5.2 Manipulação física do movimento com rigidbody 2D***

Embora a leitura do input fosse feita a cada quadro no método Update(), a aplicação do movimento físico no personagem foi implementada no FixedUpdate(). Essa escolha técnica é justificada pelo fato de que o FixedUpdate() é chamado em intervalos de tempo constantes, garantindo maior estabilidade e consistência nas simulações físicas, independentemente da taxa de quadros do dispositivo.

O movimento do personagem principal foi realizado por meio da manipulação direta da propriedade velocity do componente Rigidbody2D vinculado ao objeto jogador. Essa abordagem foi preferida em relação ao uso de métodos como AddForce(), pois oferece um controle mais imediato e previsível da movimentação, aspecto especialmente importante para uma jogabilidade voltada ao toque em dispositivos móveis.

Além disso, o vetor de entrada (moveInput) foi normalizado para assegurar uma velocidade constante mesmo durante movimentos diagonais, proporcionando uma resposta fluida e uniforme.

Essa implementação técnica também está alinhada à proposta estética e mecânica de Space Vibes, cuja ambientação remete à sensação de gravidade zero. O sistema de



movimentação foi concebido para simular uma inércia espacial: ao mover o joystick ao máximo em determinada direção e soltá-lo, o astronauta continua sua trajetória indefinidamente, como se estivesse à deriva no vácuo. Essa sensação de liberdade e deslocamento contínuo é fundamental para reforçar a imersão no universo do jogo.

#### ***4.5.3 Integração das animações com o sistema de input***

A movimentação do personagem em Space Vibes foi pensada para refletir não apenas um deslocamento físico, mas também uma resposta visual coerente com a fluidez espacial da ambientação proposta. Para isso, utilizou-se um sistema de animação baseado em bones (esqueletos articulados), implementado por meio do componente Animator, nativo da Unity.

As animações foram ativadas dinamicamente de acordo com a direção e intensidade do deslocamento gerado pelo joystick virtual. Em cada quadro, a magnitude e orientação do vetor de entrada eram avaliadas para determinar qual sequência animada seria executada. Por exemplo, ao mover o joystick para cima e rotacionar em 360° com intensidade máxima, o personagem executa uma animação de “pirueta”, simulando uma rotação livre no espaço, como se estivesse em gravidade zero.

Esse sistema de transições baseadas em movimento proporcionou uma sincronia precisa entre física e representação visual. A resposta do personagem é suave e expressiva, alinhando a estética do jogo à sua proposta sensorial e imersiva.

### **4.6 Implementação da geração procedural de corpos celestes**

Para garantir variedade no ambiente de exploração, foi desenvolvido um sistema dinâmico de geração procedural de planetas. O objetivo desse sistema é criar ambientes variados e imprevisíveis, promovendo uma sensação contínua de descoberta durante a navegação espacial. Cada planeta possui características únicas, incluindo variações de cor, presença de luas, nuvens de partículas e atmosferas distintas.

A implementação técnica da geração procedural foi realizada por meio de duas classes principais. A primeira, PlanetGenerator, é responsável pela lógica de criação e distribuição dos planetas no cenário durante a execução do jogo (runtime). A segunda,

PlanetData, estrutura os dados referentes às características dos planetas, como tamanho, cor e tipo — que podem variar entre modelos simples (apenas um sprite), sprites com textura única, sprites com múltiplas texturas ou sprites com cor sólida sem texturas.

Para compor o ambiente visual de *Space Vibes*, utilizou-se uma série de assets criados na plataforma Leonardo.AI, focados em elementos espaciais como planetas e corpos celestes. A função `Random.Range()` da Unity foi aplicada para introduzir aleatoriedade na posição e no tipo dos planetas, respeitando limites predefinidos que mantêm a consistência visual do cenário. A aplicação detalhada desse método será apresentada nas seções seguintes.

A instanciação dos planetas em tempo de execução foi feita por meio da função `Instantiate()`, utilizando prefabs predefinidos como base visual dos corpos celestes. Essa abordagem possibilitou a criação de uma grande quantidade de objetos sem necessidade de definição manual prévia, otimizando o desenvolvimento dentro do prazo estipulado de duas semanas.

A separação entre dados e lógica, seguindo princípios de abstração, trouxe maior organização e flexibilidade ao projeto. Dessa forma, ajustes nas propriedades dos planetas podem ser realizados facilmente sem impactar diretamente o código de geração. Essa metodologia contribuiu significativamente para a construção de um ambiente dinâmico, reforçando o caráter exploratório e contemplativo que fundamenta a proposta de *Space Vibes*.

Por exemplo, informações como cor, escala, rotação e tipo de cada planeta foram armazenadas em estruturas de dados independentes do script responsável pela instanciação. Assim, para modificar a paleta cromática de um planeta específico ou ajustar a probabilidade de surgimento de determinados tipos, basta alterar os dados correspondentes, sem interferir no funcionamento interno do código.

Isso acelerou o processo iterativo de testes visuais e viabilizou futuras expansões modulares do sistema de geração, como a inclusão de novos estilos de corpos celestes ou atmosferas animadas, sem comprometer a base já validada.

#### **4.7 Integração de efeitos visuais**

Com o intuito de ampliar a imersão do jogador e reforçar a expressividade do universo representado, foi integrada ao projeto uma camada de efeitos visuais e sonoros que responde diretamente às principais ações do jogo. A movimentação do personagem, as colisões com obstáculos e a destruição de elementos foram identificadas como pontos-chave para a

aplicação desses efeitos, contribuindo para uma experiência mais envolvente e responsiva, alinhada à proposta sensorial e contemplativa do jogo.

A implementação foi organizada em torno de duas classes principais: VFXManager e SFXManager. A primeira é responsável pelo controle centralizado dos efeitos visuais, organizando e ativando os sistemas de partículas que compõem os fenômenos visuais, como efeitos nos planetas, estrelas e no personagem. Já a segunda, SFXManager, administra os efeitos sonoros, assegurando que os sons correspondentes a cada interação sejam reproduzidos com precisão e em sincronia com as ações visuais. Ambas as classes funcionam como gerenciadores globais, mantendo o código organizado e evitando redundâncias na chamada e manipulação desses efeitos.

Para os efeitos visuais dinâmicos com baixo impacto no desempenho, foi utilizado o Particle System da Unity. Tradicionalmente, a renderização de partículas em tempo real demandava alto poder computacional, pois envolvia o processamento simultâneo de centenas ou milhares de elementos gráficos, comprometendo a performance em hardwares modestos (WATT & POLICARPO, 2007). Contudo, com as sucessivas atualizações do motor gráfico Unity, foi possível otimizar esses sistemas, possibilitando efeitos visuais sofisticados com consumo eficiente de recursos. Detalhes técnicos sobre o funcionamento e a evolução desses sistemas podem ser consultados no Apêndice A.

No contexto do jogo Space Vibes, os Particle Systems foram empregados para compor elementos como estrelas cadentes, atmosferas planetárias e a movimentação em alta velocidade do personagem. Nas atmosferas, utilizou-se a sobreposição de duas texturas em partículas com velocidades e níveis de opacidade distintos, criando uma sensação de profundidade e leveza nos campos gasosos.

As estrelas cadentes foram geradas por partículas que seguem trajetórias diagonais de curta duração, com brilho intenso e desvanecimento gradual, simulando corpos celestes atravessando o campo de visão do jogador. Quanto ao personagem, partículas são emitidas ao atingir sua velocidade máxima, simulando faíscas e chamas, como se o atrito extremo gerasse combustão. Esse recurso, aliado ao movimento acelerado das estrelas ao fundo, reforça visualmente a sensação de alta velocidade.



Figura 20 - Imagem das partículas do astronauta



Fonte: <https://drive.google.com/drive/folders/1Rimtb6MJ7f1dYp-RFmkocwciT1Nr0NOE?usp=sharing>

A ativação dos efeitos visuais e sonoros foi integrada a eventos programáticos nativos da Unity, como o `OnCollisionEnter2D`. Esse tipo de trigger possibilita o disparo preciso dos efeitos, como pode ser visualizado na figura 20, no exato momento em que a ação ocorre, garantindo maior sincronização entre a jogabilidade e a resposta audiovisual. Além disso, o uso de eventos contribui para a escalabilidade do código, facilitando a adição de novos efeitos sempre que novas interações forem implementadas no jogo.

#### 4.8 Desenvolvimento da identidade visual e sonora

Esta seção aborda a construção da identidade visual e sonora de *Space Vibes*, destacando as escolhas estéticas que definem o universo do jogo. Foram exploradas cores, formas, trilhas e efeitos sonoros inspirados nas estéticas LoFi e Retrowave, com o objetivo de reforçar a ambientação espacial e proporcionar uma experiência sensorial coesa e envolvente.

##### 4.8.1 Criação do background

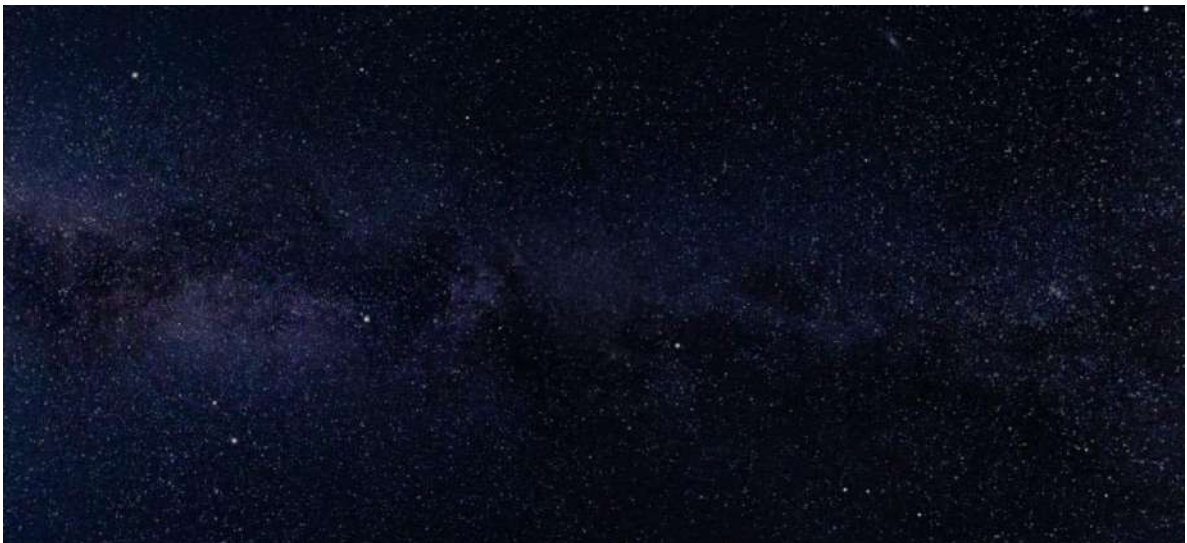
O background principal de *Space Vibes* foi concebido com foco em transmitir a estética retrowave e a imersão espacial desejada pelo projeto. Para isso, foi utilizada a plataforma Leonardo.AI, uma ferramenta de geração de imagens por inteligência artificial que permite controle estilístico por meio de prompts descritivos. A escolha por essa abordagem proporcionou agilidade na produção visual, além de permitir iterações rápidas até atingir o

resultado desejado.

O fundo foi planejado para harmonizar com a paleta néon do jogo, integrando-se visualmente com os demais elementos da interface. Como parte dessa ambientação, o background realiza uma transição sutil de cor entre o preto e o roxo ao longo do tempo, como pode ser visto na figura 21 a) e b) recurso que reforça a sensação de passagem temporal e contribui para uma atmosfera mais coesa em relação à estética do jogo.

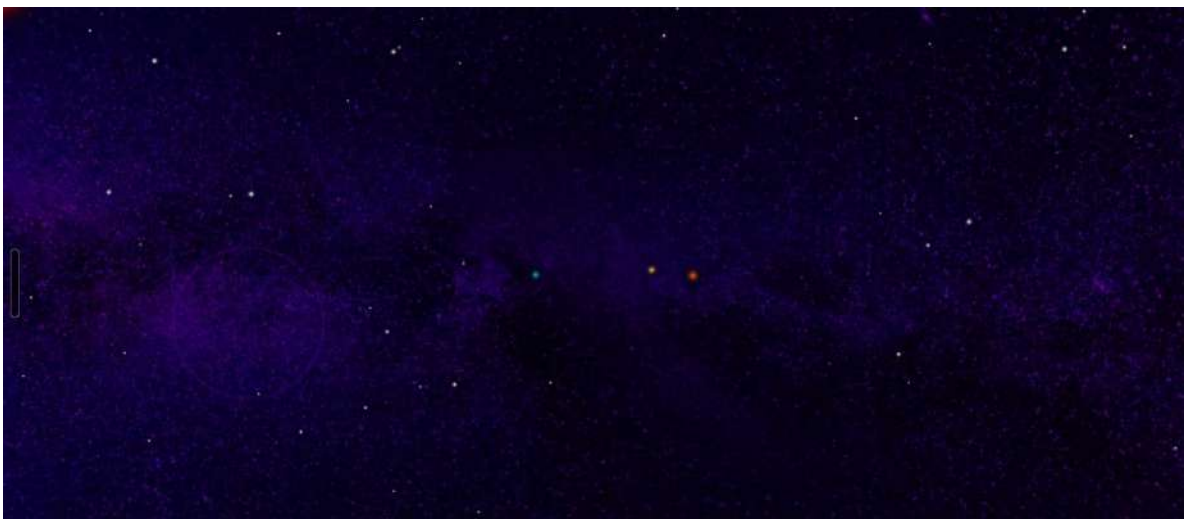
Figura 21 - Imagem de fundo do jogo na a) cor padrão e b) roxo escuro

a)



Fonte: <https://drive.google.com/drive/folders/1Rimtb6MJ7f1dYp-RFmkocwciT1Nr0NOE?usp=sharing>

b)



Fonte: <https://drive.google.com/drive/folders/1Rimtb6MJ7f1dYp-RFmkocwciT1Nr0NOE?usp=sharing>

Entre os prompts utilizados na criação do background, destacam-se descrições como: “retrowave space background with nebula, soft purple and pink tones, glowing stars,

grainy texture, 1980s sci-fi aesthetic”, assim como variações do tipo “lofi space scene with deep blues, pixel stars and subtle gradients, VHS glow style”. Esses comandos foram ajustados iterativamente com base no feedback visual, garantindo que o resultado final se integrasse de forma harmoniosa e coerente à proposta artística do jogo.

#### 4.8.2 Aplicação de shading nos planetas

Um dos aspectos fundamentais na consolidação da identidade visual de Space Vibes foi a aplicação de técnicas de shading para a composição estética dos planetas gerados proceduralmente. Para a definição das cores utilizadas nesses shaders, recorreu-se a uma paleta cuidadosamente selecionada, que inclui tons vibrantes de vermelho, magenta, ciano, roxo e amarelo, como pode ser visualizado na figura 22. Essa escolha cromática reforça a estética retrowave do jogo, conferindo aos corpos celestes uma aparência visualmente impactante e alinhada à atmosfera neon e futurista proposta.

Figura 22 - Imagem com a paleta de cores do jogo

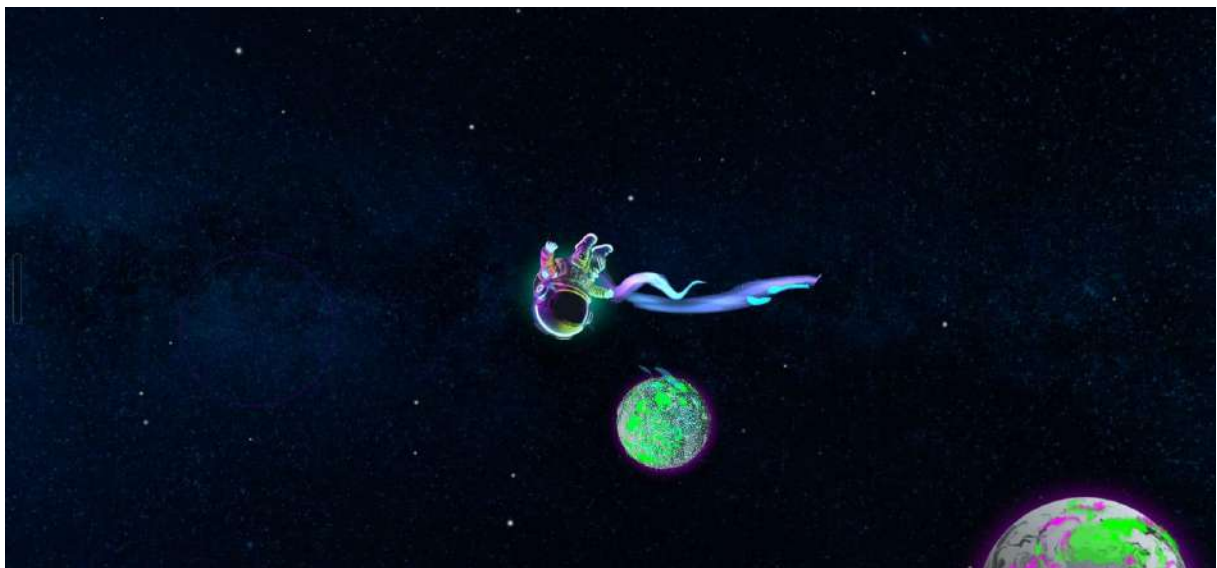


Fonte: <https://drive.google.com/drive/folders/1Rimtb6MJ7f1dYp-RFmkocwciT1Nr0NOE?usp=sharing>

Para gerar esse efeito visual, aplicou-se um gradiente de cor nos *shaders* dos planetas, com transições suaves entre branco e uma das cores da paleta, para gerar o efeito de uma iluminação neon. A iluminação foi manipulada a partir de uma fonte global controlada por script, combinada com cálculos de direção normal da malha das esferas planetárias. Esse arranjo permitiu criar uma falsa profundidade por meio de variação de brilho e sombra, mesmo em um ambiente 2D.

Esses efeitos foram desenvolvidos para interagir dinamicamente com a posição da câmera e os movimentos do jogador. Para isso, o shader foi programado para captar a posição da câmera principal e ajustar automaticamente a orientação do brilho e dos reflexos projetados sobre os planetas, a figura 23 exemplifica isso. À medida que o jogador se move ou muda a perspectiva, os reflexos simulados se reposicionam levemente, criando um efeito sutil de paralaxe<sup>37</sup> e iluminação reativa.

Figura 23 - Imagem do astronauta e do background



Fonte: <https://drive.google.com/drive/folders/1Rimtb6MJ7f1dYp-RFmkocweiT1Nr0NOE?usp=sharing>

#### 4.8.3 Rastro visual com shading no personagem

Além dos elementos cênicos, o personagem principal, o astronauta, também foi beneficiado por essa solução visual ao receber um rastro espacial com efeitos de *shading* e partículas neon. O rastro foi desenvolvido utilizando o sistema Trail Renderer<sup>38</sup>, acoplado ao personagem e modificado por um shader que manipula opacidade e brilho ao longo do tempo, esse efeito pode ser visualizado na figura 24, criando uma sensação dinâmica de velocidade e deslocamento em um ambiente digital.

<sup>37</sup> Paralaxe é o fenômeno óptico em que a posição aparente de um objeto varia conforme o ponto de observação. Na astronomia, por exemplo, mede-se a paralaxe estelar para calcular distâncias entre corpos celestes. Em interfaces digitais, o efeito de paralaxe cria profundidade visual ao mover elementos em velocidades diferentes durante a rolagem da tela.

<sup>38</sup> Trail Renderer é um componente visual do motor Unity que cria um efeito de rastro contínuo e fluido atrás de um objeto em movimento. Ele gera uma linha que segue o caminho percorrido pelo objeto, podendo ser configurada para variar em largura, cor, opacidade e tempo de duração, sendo amplamente utilizado para representar efeitos de velocidade, como rastros de luz, fumaça ou trilhas de partículas.



Figura 24 - Imagem do rastro do astronauta



Fonte: <https://drive.google.com/drive/folders/1Rimtb6MJ7f1dYp-RFmkocwciT1Nr0NOE?usp=sharing>

As cores do rastro seguem a paleta utilizada nos planetas, reforçando a unidade cromática e o impacto visual. O efeito resultante é contínuo e fluido, ampliando a percepção de movimento e imergindo o jogador em uma ambientação estilizada e coerente.

Esse efeito visual é responsivo ao comportamento do jogador: quanto maior a intensidade do input recebido pelo joystick, mais vibrante e alongado o rastro se torna. Para isso, os valores de velocidade foram vinculados aos parâmetros de width e time do Trail Renderer via script, possibilitando variações sutis no efeito conforme a movimentação do personagem. Essa interação dinâmica entre o movimento e o rastro reforça a imersão do jogador, acentuando a estética retrowave proposta pelo jogo.

#### ***4.8.4 Escolha da trilha sonora***

A seleção da trilha sonora foi realizada por meio de uma pesquisa no YouTube, utilizando o prompt "lofi retrowave space ambient instrumental", com o objetivo de encontrar faixas que unissem atmosferas relaxantes e elementos sonoros futuristas, alinhados com a proposta estética do projeto. Durante a curadoria, foram consideradas faixas como Resonance, de HOME (2014), Space Love, de Astrophysics (2019), e Astronomy, de Karl Casey (2022). A última foi escolhida por apresentar uma sonoridade envolvente, marcada por sintetizadores suaves e batidas lentas, que evocam a imersão espacial desejada para a experiência proposta em Space Vibes.

Embora Astronomy, de Karl Casey (2022), esteja protegida por direitos autorais e não possua licença aberta para uso comercial, sua utilização neste projeto se restringe a fins acadêmicos, sem fins lucrativos, como parte de um protótipo funcional. A faixa foi incorporada com o propósito de reforçar a ambientação sonora durante o desenvolvimento, e será substituída por trilha licenciada ou original caso o projeto venha a ser distribuído publicamente ou comercialmente.

#### **4.9 Revisão conceitual**

No dia 20 de setembro de 2022, o processo de desenvolvimento concentrou-se em uma revisão crítica dos conceitos teóricos que fundamentaram a construção da experiência interativa do jogo. Essa etapa teve caráter avaliativo, possibilitando verificar a efetiva aplicabilidade das bases teóricas adotadas, como a teoria do Flow, os princípios do design emocional, e as estéticas Retrowave e LoFi, no desenvolvimento do jogo. A análise buscou compreender se os fundamentos, especialmente aqueles relacionados ao equilíbrio entre desafio e habilidade, à responsividade e à clareza dos objetivos, estavam devidamente representados nos sistemas de *Space Vibes*, na narrativa implícita e na ambientação visual.

Como resultado, a revisão confirmou que os elementos conceituais selecionados foram não apenas pertinentes, mas também viáveis dentro do escopo técnico do projeto. A capacidade de gerar assets visuais rapidamente por meio de ferramentas baseadas em inteligência artificial, como a plataforma Leonardo.AI, mostrou-se um diferencial fundamental nesse processo.

Essa agilidade permitiu testar e aplicar variações visuais e funcionais de forma dinâmica, assegurando que o refinamento estético e a implementação dos conceitos ocorressem com eficiência. Conclui-se, portanto, que a convergência entre teoria e prática foi alcançada com êxito, viabilizando a entrega de uma experiência coesa e alinhada à proposta inicial.

#### **4.10 Semana 2: desenvolvimento e implementação**

Nesta etapa, foram iniciadas a implementação das mecânicas principais e a integração dos elementos visuais e sonoros ao projeto. O foco da semana esteve na estruturação da base do jogo, testes de funcionalidades e ajustes iniciais para garantir uma experiência jogável e coerente com a proposta de *Space Vibes*.

#### ***4.10.1 Configuração inicial do projeto - estrutura inicial na Unity***

No dia 21 de setembro de 2022, foi iniciada a configuração do ambiente de desenvolvimento do projeto *Space Vibes*, utilizando o motor Unity em sua versão 2020.3 (Unity 2020 LTS), que representava à época a versão mais recente e estável, com suporte a recursos avançados necessários para o desenvolvimento do projeto. Após a criação do projeto, foi estabelecida uma cena principal que serviria como base de testes e estruturação de elementos iniciais, sendo esta cena nomeada como Main Scene.

#### ***4.10.2 Importação e organização dos assets***

Foi realizada a importação manual de todos os arquivos gráficos previamente preparados com estilo vetorial bidimensional, organizados em subpastas específicas como personagens, cenários e planetas. Essa etapa visou garantir que os recursos visuais estivessem devidamente separados e acessíveis, facilitando tanto a prototipação quanto as fases posteriores de animação e programação.

A estrutura da aba Project no Unity foi construída com base em boas práticas já aplicadas pela empresa Sector 8 Studio, utilizando pastas nomeadas como Sprites, Animations, Prefabs, Scripts, Scenes, UI e Materials. A organização lógica dos arquivos também permitiu uma maior fluidez no pipeline de desenvolvimento, especialmente na criação de assets reutilizáveis, como no caso dos satélites naturais e na construção modular dos elementos do jogo.

##### **Listagem dos arquivos importados:**

- **Astronauta – Cabeça (PNG):** sprite separado da cabeça do personagem principal, usado para permitir animações autônomas.
- **Astronauta – Dorso (PNG):** parte central do corpo do personagem, funcionando como base de junção para os membros.

- **Astronauta – Braços e Pernas (PNG):** sprites independentes dos membros, facilitando o controle por cinemática direta.
- **Planeta Simples (PNG):** versão básica de planeta, sem textura, usada para composição de elementos de fundo ou objetos secundários.
- **Planeta com Uma Textura (PNG):** sprite com textura aplicada à superfície, conferindo profundidade visual.
- **Planeta com Duas Texturas (PNG):** planeta com sobreposição de duas texturas, permitindo mais detalhes e estilo personalizado.
- **Planeta Predefinido (PNG):** conjunto de planetas estilizados com aparência finalizada, prontos para aplicação direta em cena.
- **Background (PNG):** elementos de fundo, utilizados para construção de paralaxe e ambientação espacial da cena principal.

#### ***4.10.3 Criação dos prefabs de planetas***

A partir dos sprites importados, foi iniciada a criação de prefabs dos planetas. Cada planeta foi configurado como um GameObject, contendo um componente Sprite Renderer com a imagem vetorial respectiva. Foram introduzidas variáveis via código C# para complementar a lógica de funcionamento de cada planeta.

Entre elas, uma variável booleana chamada `haveMoon`, utilizada para realizar um sorteio no momento da criação e definir se o planeta que será instanciado na cena terá ou não um satélite natural ao redor, sendo utilizado um sprite simples sem variação, apenas o mesmo modelo para todos. Outro booleano chamado `planetOk` foi implementado para introduzir um planeta sem presença de texturas, apenas um sprite, que apresenta cores e características únicas.

Foram também declarados arrays do tipo `Color[]` chamados `colorBase`, `colorTexture` e `colorTexture2`, responsáveis por armazenar combinações de cores e variações de textura que serão aplicadas dinamicamente ao Sprite Renderer via código, conforme regras estabelecidas na identidade visual do jogo. Essas cores são pré-definidas com base na paleta



cromática retrowave do projeto, garantindo que cada planeta mantenha a coerência estética com o universo de Space Vibes.

A Unity trata a aplicação de cores da paleta como uma entrada vetorial (RGBA). Quando o script seleciona um valor de um dos arrays (como `colorBase`), esse valor pode ser imediatamente atribuído ao componente `SpriteRenderer.color`, o que atualiza visualmente o sprite renderizado na tela. Isso garante consistência visual sem a necessidade de múltiplos arquivos de imagem para cada variação de cor.

**Exemplo de código:**

```
spriteRenderer.color = colorBase[Random.Range(0, colorBase.Length)];
```

Essa flexibilidade é essencial para aplicar a identidade visual definida pela paleta retrowave, com suas cores neon, gradientes vibrantes e contraste alto. Também reduz a necessidade de múltiplos sprites com variações manuais, tornando o fluxo mais leve e adaptável para diferentes condições do jogo.

As variáveis `colorBase`, `colorTexture`, `colorTexture2`, e o próprio componente `SpriteRenderer` são todas configuráveis no Inspector da Unity. Essas variáveis tornam os planetas personalizáveis, permitindo a criação procedural de combinações visuais durante a geração de fases ou testes na Main Scene.

#### ***4.10.4 Montagem do personagem principal***

O personagem principal foi criado a partir de um `GameObject` vazio nomeado `Player`, que recebeu como filho um objeto visual com o componente `Sprite Renderer`, configurado com o sprite inicial em estado `idle`, pode ser visualizado na figura 25, onde o personagem se encontra parado. O objeto principal recebeu os componentes `Rigidbody 2D`, com gravidade ativada, `Box Collider 2D`, com um ajuste preciso no contorno do corpo por meio do uso do `outliner`, um recurso que adiciona uma linha de contorno ao redor do objeto, destacando sua silhueta, e o componente `Animator` para controle das animações.

Figura 25 - Imagem do personagem no estado idle



Fonte: <https://drive.google.com/drive/folders/1Rimtb6MJ7f1dYp-RFmkocwciT1Nr0NOE?usp=sharing>

O Animator Controller foi criado para gerar animações com o propósito de apresentar um movimento de flutuação do personagem, uma vez que ele está em um ambiente de gravidade zero. O personagem foi montado parte por parte e depois criada uma estrutura de Bones dentro do Animator para movimentá-lo, criar as animações e manter interligada a sua estrutura. A transição entre estados foi controlada por parâmetros gerados com a movimentação do joystick, de acordo com o deslocamento do personagem. Esses parâmetros foram manipulados pelo script PlayerController, com base na leitura de inputs horizontais e verticais via Input.GetAxis.

#### ***4.10.5 Animações e configuração do animator***

Os Animation Clips foram criados com base nas movimentações dos Bones. Cada clip foi configurado com frame rate constante e loop habilitado para os estados em que são feitos os movimentos de subida, descida e rotação na tela. O estado de “Up” foi configurado com uma sequência de movimentos que indica que o personagem está subindo, sem loop. Os clips foram inseridos no Animator, com transições condicionais conectadas por parâmetros definidos. O script responsável atualizava os parâmetros em tempo real, dependendo do movimento realizado no joystick.

Além da definição dos Animation Clips e dos parâmetros condicionais no Animator, foi configurada a interpolação entre os estados para garantir transições visuais suaves

durante a execução das animações. Utilizou-se o recurso de Transition Duration<sup>39</sup> para determinar o tempo de interpolação entre um estado e outro, evitando cortes abruptos ou mudanças bruscas nos movimentos.

Para os estados de subida e descida, aplicou-se um tempo de transição curto, preservando a fluidez do movimento sem comprometer a resposta do personagem ao controle do jogador. Já nas transições para estados de rotação ou repouso, foram utilizados tempos de interpolação ligeiramente maiores, reforçando a naturalidade do deslocamento e mantendo a coerência visual.

#### ***4.10.6 Criação do canvas e elementos da interface***

Foi criado um Canvas configurado no modo Screen Space – Overlay, no qual os elementos de interface são renderizados diretamente na camada mais alta da tela, acima de todos os objetos 3D da cena. Esse modo é ideal para HUDs (Heads-Up Displays) ou menus fixos, pois não depende da posição ou rotação da câmera, ou seja, os elementos permanecem sempre visíveis e estáticos na tela, como se fossem parte do sistema operacional do jogo. Essa configuração garante clareza, estabilidade visual e controle total sobre o layout da interface.

Dentro desse Canvas, foram inseridos elementos básicos de interface, como textos informativos e marcadores. Para a tipografia, foi utilizado o componente TextMeshPro<sup>40</sup>, que oferece alta qualidade de renderização textual, suporte a fontes personalizadas e efeitos visuais aprimorados, essenciais para manter a identidade visual do jogo alinhada à estética Retrowave e LoFi proposta.

Os elementos foram posicionados utilizando o sistema de ancoragem e organizados com Grid Layout Group e Horizontal Layout Group, quando necessário. Um Empty Object chamado UI Container foi utilizado como organizador visual dentro do Canvas, a união desses componentes pode ser visualizada na figura 26.

---

<sup>39</sup> Transition Duration é um parâmetro presente no Animator do Unity que define o tempo de interpolação entre uma animação e outra. Ele determina quanto tempo leva para que a nova animação substitua a anterior de forma gradual, possibilitando transições mais suaves e evitando mudanças abruptas no estado visual do personagem.

<sup>40</sup> TextMeshPro é um sistema de renderização de texto avançado da Unity baseado em Signed Distance Field (SDF), que permite escalabilidade sem perda de qualidade, aplicação de sombras, contornos e gradientes, além de maior precisão tipográfica em comparação ao sistema de texto padrão da engine.

Figura 26 - Imagem do hud inicial



Fonte: <https://drive.google.com/drive/folders/1Rimtb6MJ7f1dYp-RFmkocwciT1Nr0NOE?usp=sharing>

#### ***4.10.7 Desenvolvimento de scripts base***

Foi iniciado o desenvolvimento dos scripts principais, com base no comportamento esperado para o protótipo inicial. O script `PlayerController` foi criado, contendo funções para movimentação horizontal, salto, controle de física e atualização do Animator. A movimentação foi feita aplicando velocidade diretamente ao `Rigidbody`, utilizando a função `velocity`, enquanto o salto foi implementado com uma força vertical, utilizando `AddForce`. Os inputs foram lidos com `Input.GetAxis` e `Input.GetButtonMove` do joystick.

O script `PlanetBehavior` foi desenvolvido para configurar comportamentos específicos dos planetas, com destaque para a rotação dos corpos celestes. Ao serem instanciados na cena, os planetas recebem uma posição inicial aleatória no eixo Z, o que garante que cada um deles apresente uma orientação visual única desde o início. Essa rotação é mantida em tempo real através da função `Update()`, utilizando o comando `Transform.Rotate` no eixo Z.

O script `GameManager` foi iniciado com funções básicas, como iniciar jogo, reiniciar cena e criar e apagar os planetas que estão sendo visualizados na cena, utilizando variáveis públicas e funções estáticas acessíveis globalmente.

#### **4.10.8 Montagem e organização da cena**

A cena foi organizada visualmente, com os elementos alinhados em seu grid e com a câmera ajustada para aspect ratio de 16:9, o qual foi escolhido por ser o formato padrão mais amplamente utilizado em monitores, televisores e dispositivos móveis atuais. A aba Scene foi utilizada com ferramentas de snapping ativadas para posicionamento preciso. A aba Game foi configurada para exibir o jogo com qualidade proporcional ao formato de tela padrão, e os elementos visuais foram testados em múltiplas resoluções através da opção de simulação de resoluções da Unity.

### **4.11 Mecânica de movimentação**

Esta seção descreve o desenvolvimento da mecânica de movimentação do jogador, elemento essencial para a fluidez e imersão de *Space Vibes*. Foram considerados aspectos como responsividade, suavidade nos controles e alinhamento com a proposta relaxante do jogo, garantindo uma navegação intuitiva no ambiente espacial.

#### **4.11.1 Movimentação em ambiente de gravidade zero**

No dia 22 de setembro de 2022, a atenção do desenvolvimento se voltou à implementação da principal mecânica de controle do jogador: a movimentação do astronauta em um ambiente que simula o vácuo espacial e a ausência de gravidade. O objetivo foi criar uma física que transmitisse a leveza e a inércia próprias do espaço, estabelecendo um sistema de propulsão contínua com base em impulsos direcionais suaves. Em vez de uma movimentação rígida e direta, o personagem responde de forma fluida aos comandos, deslizando levemente antes de desacelerar, o que contribui significativamente para a sensação de imersão em um ambiente espacial realista.

#### **4.11.2 Uso do joystick pack e controle direcional**

A mecânica foi construída utilizando a extensão Joystick Pack, que fornece controles responsivos baseados em toque para jogos mobile. Esse recurso permitiu implementar um sistema de movimentação analógica, em que a intensidade do movimento depende da distância do toque em relação ao centro do joystick. Isso proporcionou um controle mais

orgânico e intuitivo, alinhado à proposta de promover o estado de Flow descrito por Jenova Chen (2007), que destaca a importância de uma resposta contínua e natural entre ação e reação dentro do jogo. A movimentação oferecida pelo Joystick Pack ajudou a manter esse equilíbrio, pois elimina travas e movimentos engessados que poderiam prejudicar a imersão do jogador.

Essas “travas” no movimento referem-se a limitações típicas de sistemas de movimentação digital com má implementação, nos quais o personagem se move apenas em direções fixas (como cima, baixo, esquerda e direita), em velocidades constantes e com mudanças bruscas de direção. Isso cria uma sensação de controle rígido, pouco fluido, como se o personagem estivesse “travado” em eixos invisíveis ou demorasse para responder aos comandos do jogador. Por exemplo, em jogos que não utilizam movimentação analógica, pressionar uma tecla como “W” move o personagem sempre na mesma velocidade e direção, sem variação. Isso pode resultar em curvas quadradas, mudanças de direção abruptas e uma sensação artificial, quebrando a imersão, especialmente em experiências que visam fluidez ou relaxamento, como Space Vibes.

#### ***4.11.3 Integração visual: rastro com shaders dinâmicos***

Para reforçar visualmente o deslocamento do personagem, foi implementado um sistema de rastro (trail) ativado por shaders customizados, que responde diretamente à movimentação do jogador. O rastro tem como função não apenas tornar a movimentação mais perceptível, mas também intensificar a estética retrowave com linhas pulsantes em cores neon que acompanham o fluxo do personagem.

Do ponto de vista técnico, a ativação desse rastro foi feita por meio de condicionais if e else, em que o sistema identifica se há movimento ativo e, em caso afirmativo, aciona o rastro. Quando o jogador deixa de se mover, o rastro é desativado, reforçando a ideia de ausência de propulsão em gravidade zero. As imagens abaixo exemplificam o funcionamento do trail por meio da figura 27 a) e b), que ilustram o efeito visual no jogo, e as demais imagens mostram a parte do código, com as variáveis e funções necessárias para o funcionamento.

Figura 27 - Imagem do astronauta a) parado e b) movimento

a)



Fonte: <https://drive.google.com/drive/folders/1Rimtb6MJ7f1dYp-RFmkocwciT1Nr0NOE?usp=sharing>

b)



Fonte: <https://drive.google.com/drive/folders/1Rimtb6MJ7f1dYp-RFmkocwciT1Nr0NOE?usp=sharing>

Os códigos responsáveis pela movimentação do personagem e pelos efeitos visuais associados a essa mecânica estão presentes na figura 28 a) até d). Esses scripts foram desenvolvidos em C# dentro da Unity e incluem funcionalidades como deslocamento contínuo, resposta aos comandos do jogador, animações sincronizadas e efeitos complementares, como partículas e trilhas sonoras dinâmicas.



Figura 28 - Imagens do código do script de movimentação do personagem

a)

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;
using UnityEngine.VFX;

public class PlayerController : MonoBehaviour
{
    [SerializeField] private FixedJoystick _joystick;

    public GameObject Player;
    public Animator animator;
    float horizontalMove = 0f;
    float verticalMove = 0f;
    public float speed;

    public Rigidbody2D rb;
    public SpriteRenderer spriteRenderer;
    public TrailRenderer trail;
    public TrailRenderer trail2;
    public VisualEffect visualEffect;
    public VisualEffect visualEffect2;
    public bool isShinning;
    public bool trailON;

    public Material[] shaders;
    public Material[] shaders2;
    public Color[] colors;

    public float spinForce;
    public float normalizeTime;
    private float currentSpinForce;
    private float spinTime;
    public float fatorChange;
    public Vector2 direction;
    public Color currentColor;
    public Color colorA;
    public Color colorB;
    public float velocidade;
    public bool particulasDeAtrito;
```

Fonte:

[https://drive.google.com/file/d/12BrbXcrpiEmK1uZmSZ\\_uiRN8GcdHhUpH/view?usp=sharing](https://drive.google.com/file/d/12BrbXcrpiEmK1uZmSZ_uiRN8GcdHhUpH/view?usp=sharing)

b)

```
public float rotationOffset = -90f;

public float rotationSpeed = 5f; // Velocidade de rotação da cabeça
public ParticleSystem particleSparks;
public ParticleSystem particleStrokes;
// public Vector3 strokeTransform;
public ParticleSystemRenderer systemRenderer;

private Vector2 lastVelocity; // Última velocidade registrada do astronauta
public Transform playerPos;

private void Start()
{
    // strokeTransform = particleStrokes.transform.position;
    particleStrokes.Stop();
    visualEffect2.Stop();
    particleSparks.Stop();
    playerPos = GetComponent<Transform>();
    spriteRenderer = GetComponent<SpriteRenderer>();
    rb = GetComponent<Rigidbody2D>();
    trailON = true;
    systemRenderer = particleSparks.GetComponent<ParticleSystemRenderer>();
    animator = Player.GetComponent<Animator>();
}
```

Fonte:

[https://drive.google.com/file/d/12BrbXcrpiEmK1uZmSZ\\_uiRN8GcdHhUpH/view?usp=sharing](https://drive.google.com/file/d/12BrbXcrpiEmK1uZmSZ_uiRN8GcdHhUpH/view?usp=sharing)



c)

```

if (particulasDeAtrito)
{
    if (velocidade > 1.9f)
    {
        particleSparks.Play();
        visualEffect2.Play();
    }
    else
    {
        particleSparks.Stop();
        visualEffect2.Stop();
    }
    if (velocidade > 1.98f)
    {
        visualEffect2.gameObject.SetActive(true);
        visualEffect2.Play();
    }
    else
    {
        visualEffect2.gameObject.SetActive(false);
        visualEffect2.Stop();
    }
}

```

Fonte:

[https://drive.google.com/file/d/12BrbXcrpiEmK1uZmSZ\\_uiRN8GcdHhUpH/view?usp=sharing](https://drive.google.com/file/d/12BrbXcrpiEmK1uZmSZ_uiRN8GcdHhUpH/view?usp=sharing)

d)

```

Vector3 localScale = transform.localScale;

if (_joystick.Horizontal > 0)
{
    spriteRenderer.flipX = true;
    systemRenderer.flip = localScale;
    // particleSparks.transform.rotation = Quaternion.Euler(0f, 0, 0f);
}
else
{
    spriteRenderer.flipX = false;
    // particleSparks.transform.rotation = Quaternion.Euler(0f, 180, 0f);
}
}
else if (rb.velocity.magnitude > 1.3f)
{
    // particleStrokes.gameObject.transform.parent = this.gameObject.transform;
    // particleStrokes.gameObject.transform.position = strokeTransform;
    particleStrokes.gameObject.SetActive(true);
    particleStrokes.Play();
}
}

```

Fonte:

[https://drive.google.com/file/d/12BrbXcrpiEmK1uZmSZ\\_uiRN8GcdHhUpH/view?usp=sharing](https://drive.google.com/file/d/12BrbXcrpiEmK1uZmSZ_uiRN8GcdHhUpH/view?usp=sharing)

Essas imagens, assim como outras que complementa essa etapa do código de movimentação, estão disponíveis no apêndice B por meio do link: [https://drive.google.com/file/d/12BrbXcrpiEmK1uZmSZ\\_uiRN8GcdHhUpH/view?usp=sharing](https://drive.google.com/file/d/12BrbXcrpiEmK1uZmSZ_uiRN8GcdHhUpH/view?usp=sharing)

#### 4.11.4 *Experiência imersiva e continuidade visual*

A combinação entre controle analógico por joystick, movimentação flutuante com física simulada e efeitos visuais ativados por comportamento cria uma experiência integrada e sensorialmente coesa. O jogador sente que está verdadeiramente navegando por um espaço livre, com feedback visual e físico que sustenta a ilusão de estar imerso em um ambiente de gravidade zero.

A presença do rastro neon que se projeta atrás do personagem durante o movimento não só contribui para o design visual do universo retrowave, mas também serve como reforço sensorial, pois amplifica a percepção da velocidade e direção de deslocamento. Essa sinergia entre estética e mecânica fortalece a imersão e torna a experiência mais envolvente e memorável.

#### 4.12 Geração procedural de planetas

No dia 23 de setembro de 2022, foi desenvolvido um sistema procedural para a geração de planetas únicos, utilizando combinações variadas de cores, atmosferas e satélites. As imagens-base foram criadas com o auxílio da ferramenta Leonardo.AI, no modelo 1 do capítulo 1.4, e cada planeta gerado apresentava características atmosféricas distintas. Essas variações foram extraídas e aplicadas como texturas personalizadas nos mundos presentes dentro do jogo.

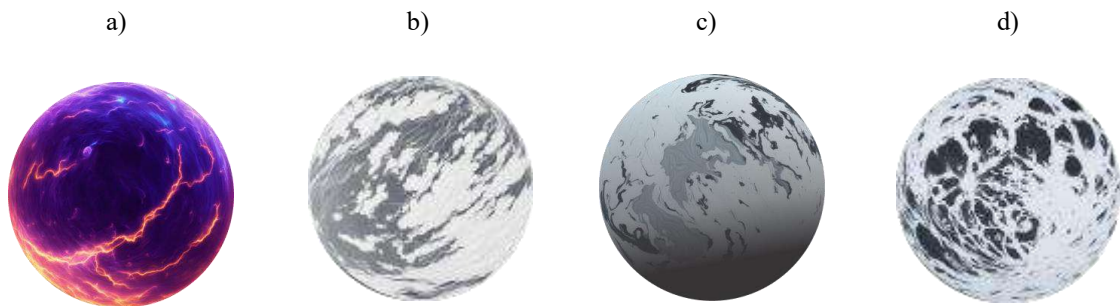
A criação dos corpos celestes obedece a uma lógica programada, onde cada tipo de planeta é definido por meio de um Enumerador. Os tipos são organizados da seguinte forma:

- **Standard** (Figura 29 d)): planetas compostos apenas por assets básicos, sem aplicação de texturas;
- **OneTex** (Figura 29 b)): planetas que utilizam um asset simples com a adição de uma única textura;
- **TwoTex** (Figura 29 c)): planetas que combinam o asset base com duas texturas distintas, enriquecendo visualmente sua superfície.

Além da composição visual, o sistema também inclui um Enumerador adicional que determina o tamanho de cada planeta: Small, Medium ou Big, definindo respectivamente se o planeta será pequeno, médio ou grande. Essa categorização influencia tanto o aspecto estético

quanto as propriedades físicas dos planetas no ambiente de jogo, contribuindo para a diversidade e riqueza do universo espacial gerado.

Figura 29 - Modelos finais de cada planeta a) planetOk, b) OneTex, c) TwoTex e d) Standard



Fonte: [https://drive.google.com/drive/folders/1lf\\_eMeg4tYjMNXRy7hvXJ8QYZxfyLjB-?usp=sharing](https://drive.google.com/drive/folders/1lf_eMeg4tYjMNXRy7hvXJ8QYZxfyLjB-?usp=sharing)

Esses modelos apresentados não possuem cores, exceto o modelo planetOk, figura 29 a), uma vez que ele já é um sprite que é introduzido no game sem alterações, diferente dos outros modelos de planetas utilizam da colorização e texturas que são mais complexas e utilizam de uma logica de código que realiza alterações na rotação, cor e qual o tipo da sua textura que aquele planeta deve ter ao ser instanciado na cena.

Figura 30 - Imagens do código do sprite dos planetas

a)

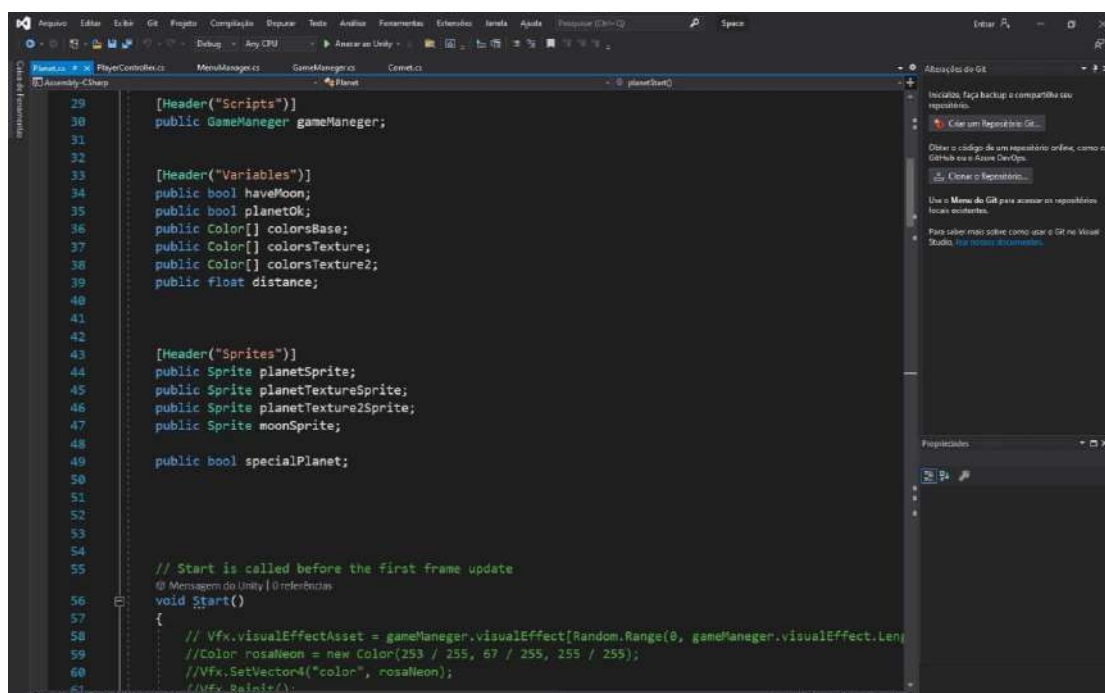
```

Arquivo  Editar  Exibir  Git  Projeto  Compilação  Depurar  Teste  Análise  Ferramentas  Extensões  Janela  Ajuda  Pesquisar (Ctrl+Q)  Space
[+] Assembly-CSharp
  Planet.cs
    1  using System.Collections;
    2  using System.Collections.Generic;
    3  using UnityEngine;
    4  using UnityEngine.VFX;
    5
    6
    7  Script do Unity (3 referências de ativo) | 16 referências
    8  public class Planet : MonoBehaviour
    9  {
    10
    11      12 referências
    12      public enum planetType { standart, oneTex, twoTex }
    13      public planetType PlanetType;
    14
    15      7 referências
    16      public enum planetSize { small, medium, big }
    17      public planetSize PlanetSize;
    18
    19      [Header("GameObjects")]
    20      public GameObject planet;
    21      public GameObject planetTexture;
    22      public GameObject planetTexture2;
    23      public GameObject moon;
    24      public GameObject player;
    25
    26      public VisualEffect Vfx;
    27      public GameObject efeitoAtmosfera;
    28
    29      private ParticleSystem planetParticles; // Referência para o sistema de partículas
    30      public Transform localEffect;
    31
    32      [Header("Scripts")]
    33      public GameManager gameManager;
    34
    35
  
```

Fonte: <https://drive.google.com/drive/folders/1NrVzLFiD5yKD8fbX37lthc7btpEnPO71?usp=sharing>

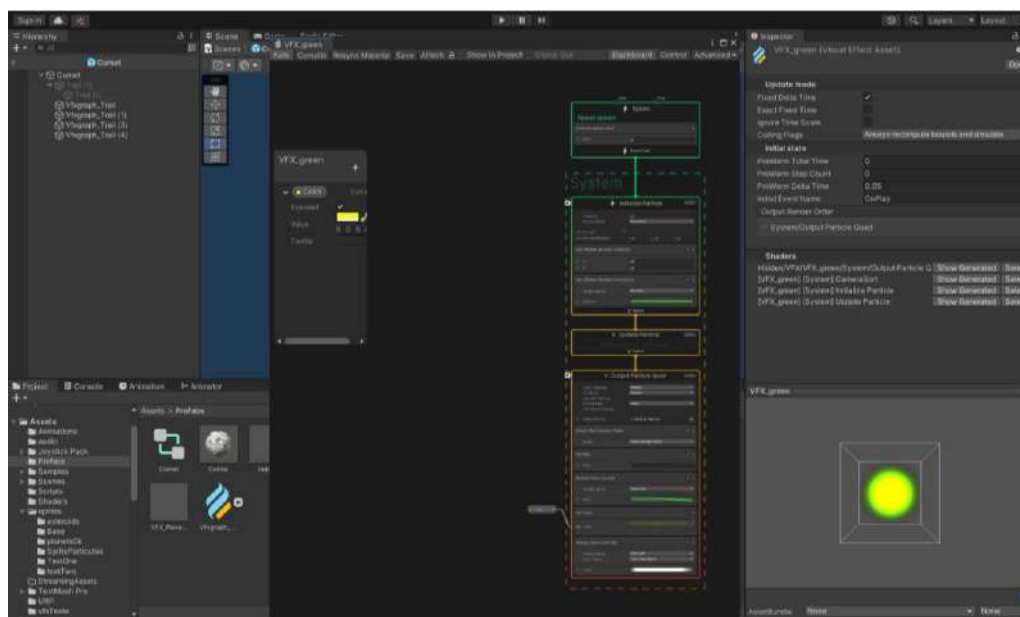
Para instanciar os elementos no jogo, foi criado um prefab que serve como modelo de referência manipulado pelo código gerador de planetas, visível na figura 32. Cada corpo celeste possui características como atmosferas geradas por VFX com cores predefinidas, possibilidade de satélites naturais instanciados aleatoriamente via Random, e partículas adicionadas por Particle Systems.

b)



Fonte: <https://drive.google.com/drive/folders/1NrVzLFiD5yKD8fbX37Ithc7btpEnPO71?usp=sharing>

Figura 31 - Imagem da aba de VFX da Unity



Fonte: <https://drive.google.com/drive/folders/1NrVzLFiD5yKD8fbX37Ithc7btpEnPO71?usp=sharing>

Para acesso de mais imagens referentes aos códigos desenvolvidos nessa etapa segue o link para o apêndice B: <https://drive.google.com/drive/folders/1LBIDM-eVNZkeJ-BqhbH7V0iRMiueiZcs?usp=sharing>

Figura 32 - Imagem da aba de prefabs



Fonte:

<https://drive.google.com/drive/folders/1ngdjaQp9X40Q8SBTos2zaIr6lhgRUpIN?usp=sharing>

Para o controle dos planetas em Space Vibes, foram desenvolvidos scripts responsáveis por gerenciar os diferentes tipos de planetas e suas respectivas características, como cor, tamanho, rotação, como pode ser visualizado nas figuras 30 a) e b). Esses códigos possibilitaram uma variedade visual e comportamental no ambiente do jogo, enriquecendo a experiência do jogador.

Além disso, o efeito de atmosfera ao redor dos planetas foi criado utilizando o Shader Graph, parte desse processo pode ser visualizado na figura 31, permitindo aplicar brilhos e camadas translúcidas que reforçam a estética do jogo.

Com relação aos prefabs utilizados para no desenvolvimento do jogo, o acesso as imagens e outros prefabs também utilizados pode ser feito no apêndice B com o link: <https://drive.google.com/drive/folders/1ngdjaQp9X40Q8SBTos2zaIr6lhgRUpIN?usp=sharing>

#### 4.13 Design de efeitos visuais e áudio

No dia 24 de setembro de 2022, foram integrados ao projeto os primeiros efeitos de partículas e sistemas de shaders, com o objetivo de enriquecer visualmente o ambiente espacial e reforçar a estética retrowave definida para o jogo. A criação desses efeitos teve início com a composição de estrelas em movimento e atmosferas com brilho dinâmico.

Utilizando o Particle System da Unity, foram desenvolvidas partículas com emissão contínua e gradientes personalizados, esse efeito pode ser visto na figura 33, simulando gases cósmicos em cores vibrantes, predominantemente em tons de ciano. Para acentuar o brilho característico do estilo neon, foi utilizado o blend mode additive, que permite a sobreposição luminosa dos elementos. Além disso, foram aplicados efeitos de fade in e fade out, garantindo transições visuais suaves e imersivas dentro do ambiente espacial.

Figura 33 - Imagem dos planeta com partículas



Fonte: <https://drive.google.com/drive/folders/1Rimtb6MJ7f1dYp-RFmkocweiT1Nr0NOE?usp=sharing>

Além disso, foram desenvolvidos shaders personalizados por meio do Shader Graph da Unity, permitindo manipulações visuais avançadas, como distorções dinâmicas e brilhos pulsantes. Esses efeitos foram aplicados a camadas de fundo e a sprites específicos, como estrelas cadentes, contribuindo para a atmosfera etérea do cenário.



Os planetas também foram aprimorados com o uso de efeitos visuais baseados em shader, incluindo brilhos atmosféricos rotacionais, bordas com efeito glow e uma sutil sensação de pulsos rítmicos, que conferem vida e profundidade às superfícies planetárias, efeitos visíveis na figura 34. Todos esses efeitos foram parametrizados com base na paleta de cores retrowave, garantindo coesão estética e reforçando a identidade visual do jogo.

Figura 34 - Imagem dos planetas



Fonte: <https://drive.google.com/drive/folders/1Rimtb6MJ7f1dYp-RFmkocwciT1Nr0NOE?usp=sharing>

Por fim, a trilha sonora foi inserida na cena principal utilizando um objeto do tipo Audio Source, configurado para reprodução em loop, com controle de volume e atenuação espacial, garantindo uma ambientação sonora contínua e envolvente. A música escolhida foi "*Astronomy*", de Karl Casey (2022), uma faixa que mescla elementos de LoFi Hip Hop com sintetizadores Retrowave, criando um ambiente sonoro imersivo e em total sintonia com a proposta visual do jogo.

A união entre o som ambiente e os efeitos visuais estabeleceu a base da imersão sensorial de *Space Vibes*, sendo esse um passo fundamental para definir o tom atmosférico e emocional do projeto.

#### 4.14 Testes e ajustes

Entre os dias 25 e 26 de setembro de 2022, o projeto *Space Vibes* foi submetido a uma fase inicial de testes, realizados pela equipe de desenvolvimento, com o objetivo de identificar falhas de comportamento, inconsistências visuais e problemas de desempenho. Essa

etapa foi essencial para validar as implementações anteriores e garantir que os sistemas de movimentação, colisão, partículas, shaders e interface respondessem de forma fluida e previsível durante a execução em tempo real.

Durante os testes, a equipe concentrou-se em validar a física do personagem, em especial o comportamento do sistema de movimentação horizontal, o movimento vertical e a resposta do Animator Controller às mudanças de estado. Foram observadas discrepâncias na transição entre a movimentação e o giro da direita para esquerda e vice-versa, que, em algumas situações, não ocorriam suavemente devido a atrasos nos parâmetros gerados para ativar os booleanos manipulados via script. Para corrigir esse comportamento, foram implementadas verificações condicionais adicionais e ajustes nos valores de interpolação das transições entre os animation clips.

Além disso, os testes também focaram na performance dos efeitos de partículas, principalmente aqueles relacionados ao rastro do personagem, aos elementos atmosféricos aplicados aos planetas e às estrelas no fundo. Foram utilizados três aparelhos distintos: o Samsung Galaxy S23 Ultra, o Motorola Moto E5 e o Xiaomi Redmi Note 8. O Galaxy S23 Ultra, da Samsung, é o mais potente entre os três, equipado com processador Snapdragon 8 Gen 2 e GPU Adreno 740, sendo ideal para aplicações gráficas exigentes. O Redmi Note 8, da Xiaomi, possui processador Snapdragon 665 e GPU Adreno 610, apresentando desempenho intermediário. Já o Moto E5, da Motorola, conta com processador Snapdragon 425 e GPU Adreno 308, representando um modelo de entrada com menor capacidade gráfica.

Apesar das diferenças de desempenho entre os dispositivos, em nenhum deles foram identificados travamentos, quedas de frame rate ou falhas de renderização durante a execução do protótipo. Ainda assim, visando garantir maior compatibilidade e prevenir possíveis problemas em dispositivos mais limitados ou não testados, foi implementado um sistema básico de quality settings<sup>41</sup> adaptativo, que reduz a emissão e a taxa de atualização dos sistemas de partículas em tempo real, de acordo com o nível de desempenho do dispositivo.

Outro ponto avaliado com atenção foi a aplicação dos shaders, sobretudo nos planetas e no personagem principal. Ajustes nos parâmetros de opacidade, intensidade de emissão e suavização dos contornos foram feitos diretamente no Shader Graph, com testes em

---

<sup>41</sup> Quality settings são configurações gráficas ajustáveis em tempo de execução, utilizadas em game engines como o Unity, que permitem equilibrar desempenho e qualidade visual conforme a capacidade de hardware do dispositivo, atuando em parâmetros como sombras, texturas, partículas e resolução.



tempo real para assegurar que os efeitos permanecessem coesos e dentro da estética retrowave proposta.

A troca de feedback entre os membros da equipe foi constante durante essa fase. Os resultados dos testes eram compartilhados em tempo real, de forma presencial, o que possibilitou um bom trabalho de refinamento. A cada falha ou comportamento inesperado identificado, a solução era rapidamente implementada e testada novamente. Essa dinâmica ágil e colaborativa permitiu o avanço rápido na estabilidade do protótipo e consolidou a base técnica sobre a qual as próximas etapas do desenvolvimento seriam construídas.

#### 4.15 Publicação

Como destacado por Schell (2014), o polimento final é essencial para criar uma experiência coesa e envolvente. Nesse sentido, os últimos ajustes foram direcionados não apenas à eliminação de bugs, mas à entrega de uma experiência estética e funcional refinada. Pequenas melhorias nos transitions de cena, nos tempos de resposta dos botões e na coerência sonora da trilha LoFi Retrowave contribuíram para uma sensação de completude e profissionalismo no produto final.

No dia 27 de setembro de 2022, foi iniciada a etapa final de desenvolvimento do projeto Space Vibes, voltada para a realização de testes de desempenho e adaptação do jogo visando sua publicação na Play Store. Este momento marcou o encerramento do ciclo de produção inicial, com foco no refinamento técnico, estabilidade geral e conformidade com os padrões exigidos pelas plataformas de distribuição.

Foram realizadas *builds*<sup>42</sup> específicas para dispositivos Android com diferentes configurações de hardware, utilizando as ferramentas de Profiler e Device Simulator integradas à Unity. Os testes avaliaram o uso de CPU e GPU, a alocação de memória, o tempo de carregamento das cenas e o desempenho dos scripts em execução contínua. A trilha de partículas, os efeitos visuais com shaders e os elementos vetoriais animados foram monitorados para identificar qualquer tipo de problema de renderização gráfico ou lógica redundante que pudesse comprometer a fluidez da experiência.

A interface foi otimizada para a proporção de tela 16:9, garantindo que as

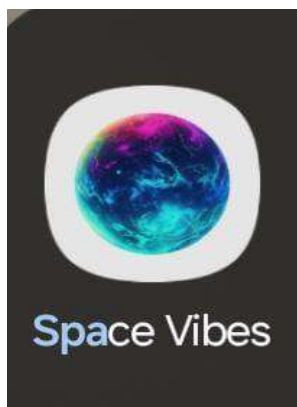
---

<sup>42</sup> Builds referem-se às versões compiladas ou montadas de um jogo durante seu processo de desenvolvimento. Essas versões, que podem variar em estágio de desenvolvimento (como alpha, beta ou release), são utilizadas para testes internos, demonstrações ou distribuição controlada, permitindo que desenvolvedores verifiquem funcionalidades, desempenho e correção de erros antes do lançamento final.

funcionalidades de toque funcionassem corretamente nessa resolução. Pequenos ajustes no posicionamento dos botões e das barras de status foram realizados para assegurar uma experiência consistente. As fontes utilizadas, originalmente com o TextMeshPro, foram convertidas para Asset Fonts a fim de garantir um carregamento rápido e manter a legibilidade, independentemente da densidade de pixels de tela.

A publicação na Google Play Store foi realizada sem fins comerciais, com o objetivo principal de disponibilizar o protótipo de forma pública e acessível para fins acadêmicos e de demonstração. Para isso, foram preparados os materiais exigidos pela plataforma, como o ícone do jogo, figura 35, a imagem da aba de apresentação do aplicativo (feature graphic), além da descrição otimizada com palavras-chave relevantes como LoFi, Retrowave, relaxing game, space, aesthetic, indie e visual experience.

Figura 35 - Imagem do ícone na pasta de aplicativos



Legenda: <https://drive.google.com/drive/folders/1Rimtb6MJ7fIdYp-RFmkocwciT1Nr0NOE?usp=sharing>

O arquivo APK foi gerado com a configuração adequada do keystore<sup>43</sup> para assinatura digital, conforme as diretrizes da plataforma. O projeto também foi testado com o Play Asset Delivery<sup>44</sup>, recurso que melhora a distribuição de arquivos maiores e permite atualizações escaláveis, especialmente em dispositivos com armazenamento limitado. Com o jogo estabilizado, otimizado e testado em diferentes dispositivos e cenários, a equipe considerou o desenvolvimento do protótipo concluído.

---

<sup>43</sup> Keystore é um arquivo criptográfico que contém as chaves usadas para assinar digitalmente o APK de um aplicativo Android, garantindo sua autenticidade e segurança no processo de distribuição.

<sup>44</sup> Play Asset Delivery é um sistema da Google Play que permite a entrega eficiente de recursos grandes e otimizados, como texturas e dados gráficos, por meio de métodos configuráveis como install-time, fast-follow e on-demand, reduzindo o tamanho inicial do aplicativo e melhorando a experiência do usuário.

## 5 RESULTADOS

O processo de desenvolvimento possibilitou a criação de elementos visuais e mecânicos que contribuíram significativamente para a construção do universo proposto. Os planetas foram gerados com atmosferas distintas, aplicando efeitos de shaders que simulam as vastas superfícies planetárias e a presença de uma camada translúcida que vibra com as cores da paleta Retrowave. Esses planetas, além de comporem o cenário, funcionam como âncoras visuais dentro da imensidão espacial, conferindo ritmo e contraste à movimentação do personagem.

Este, por sua vez, possui uma movimentação flutuante baseada na simulação de gravidade zero. A partir de impulsos leves, o astronauta é capaz de deslizar indefinidamente no espaço e, conforme sua velocidade aumenta, ativa automaticamente um efeito visual por meio de um sistema condicional (if/else) que intensifica o rastro de partículas deixado para trás. Esse rastro não é apenas decorativo, mas um componente expressivo da dinâmica do jogo, sugerindo velocidade, energia e continuidade.

Além disso, partículas foram aplicadas com o propósito de criar efeitos visuais no ambiente, como no fundo do jogo e nos planetas. As estrelas em movimento e as estrelas cadentes conferem profundidade ao ambiente, tornando o universo do jogo mais vivo e dinâmico, sem comprometer a clareza visual. A trilha sonora, composta por temas LoFi, reforça essa atmosfera ao proporcionar um ambiente sonoro contínuo, com batidas suaves e repetições rítmicas que induzem à concentração e ao relaxamento. Juntas, essas decisões permitiram que o jogo promovesse uma experiência meditativa e sensorial, em consonância com o conceito de Flow.

O desenvolvimento de *Space Vibes* parte de uma premissa bastante clara: oferecer ao jogador uma experiência sensorial que vá além do entretenimento tradicional, utilizando elementos visuais, sonoros e mecânicos como ferramentas para induzir um estado de imersão e relaxamento, inspirado na teoria do Flow de Mihaly Csikszentmihalyi (1990). A proposta central do projeto está na construção de um espaço simbólico, um ambiente onde o jogador possa se desconectar das exigências cotidianas e se reconectar com a própria percepção do tempo e do ritmo.

Apesar dos avanços e resultados consistentes, o desenvolvimento de *Space Vibes* também enfrentou limitações significativas, que merecem ser destacadas. A principal delas foi o tempo reduzido para a produção. Com apenas duas semanas disponíveis para planejar,

prototipar, desenvolver e ajustar o jogo, muitas decisões tiveram que ser tomadas de forma acelerada, o que limitou a profundidade dos testes e a quantidade de funcionalidades implementadas. A fase de testes, por exemplo, foi comprometida, o que impediu uma análise mais precisa da performance do jogo em diferentes plataformas e contextos de uso.

Outro ponto sensível foi o tamanho da equipe. Contando com apenas duas pessoas para todas as etapas do projeto, o desenvolvimento exigiu polivalência e constante adaptação entre áreas técnicas e criativas. Isso impactou diretamente na implementação de recursos extras, como menus mais complexos, ajustes de interface e melhorias de usabilidade. Além disso, algumas imagens geradas por inteligência artificial, inicialmente pensadas para enriquecer o universo visual, não foram utilizadas de forma plena, pois não se adequaram ao estilo visual final ou apresentaram inconsistências técnicas na integração com os demais elementos.

## 6 CONSIDERAÇÕES FINAIS

Ao longo do desenvolvimento de Space Vibes, o objetivo central do jogo foi investigar de que forma elementos culturais e estéticos associados às vertentes Retrowave e LoFi poderiam ser integrados ao design de jogos digitais. No final, esse objetivo foi alcançado.

A estética visual vibrante somou-se a uma trilha sonora relaxante e a uma movimentação espacial fluida, simulando uma jornada em gravidade zero. Essa experiência é complementada por uma lógica de jogabilidade que não exige ação constante, mas permite ao usuário contemplar o universo em seu próprio ritmo, reforçando a sensação de calma e imersão contínua.

A expectativa era que o projeto demonstrasse como o design de jogos, mesmo com recursos limitados, pode ser um canal eficaz para a construção de experiências subjetivas que toquem o jogador de maneira emocional e sensorial. Essa visão encontra respaldo na integração das culturas visuais e sonoras contemporâneas que orientaram a estética do projeto. Por meio de atmosferas vibrantes, trilhas minimalistas e mecânicas de movimentação contínua que simulam a leveza da gravidade zero, o jogo se posiciona como uma proposta sensível e inovadora dentro da cena indie, ao mesmo tempo contemplativa e acessível.

Do ponto de vista prático, Space Vibes também representou uma oportunidade de exploração técnica. O desenvolvimento permitiu a experimentação com ferramentas como Unity, sistemas avançados de partículas, lógica condicional para interações dinâmicas e criação de shaders personalizados. Essas ferramentas foram utilizadas não apenas como recursos técnicos, mas como meios de potencializar a expressividade do jogo. A integração fluida entre arte, música e jogabilidade reflete um processo consciente de design, que se preocupou desde o início com a coerência entre forma e função, entre estilo visual e comportamento sistêmico.

Em relação às expectativas, é possível afirmar que os principais objetivos estabelecidos no início do projeto foram alcançados. A resposta estética foi sólida, com atmosferas visualmente consistentes, fluidez na movimentação e presença de efeitos que capturam a atenção do jogador. A música, cuidadosamente integrada ao ritmo visual, desempenhou papel fundamental na ambientação. Embora o tempo de desenvolvimento tenha limitado a implementação de algumas ideias complementares, como modos de jogo alternativos, edições mais avançadas na HUD e recursos extras de personalização, a essência do projeto foi respeitada e entregue de forma completa.

O impacto do jogo, mesmo em fase inicial, já se mostra promissor. A recepção nos testes realizados indicou que o objetivo foi atingido: uma sensação de tranquilidade, envolvimento e curiosidade. A fluidez da experiência, aliada ao estilo visual Retrowave, transporta o usuário para uma jornada solitária e, ao mesmo tempo, reconfortante. Essa combinação de fatores faz de Space Vibes um jogo com potencial para alcançar públicos interessados em experiências alternativas, especialmente aqueles familiarizados com ambientes de estudo e relaxamento promovidos por playlists de LoFi, cenas noturnas urbanas e atmosferas oníricas digitais.

Com base nos aprendizados e limitações identificadas, diversas melhorias já foram projetadas para possíveis versões futuras do jogo. A primeira delas é a reformulação da interface inicial (HUD), que passará a oferecer mais opções de personalização e formas de entrada no universo do jogo, como a escolha de estilos de trilha sonora, ajustes de paleta de cores e modos de jogo alternativos. Já dentro da jogabilidade, planeja-se também oferecer maior liberdade estética ao jogador, permitindo a alteração da coloração do rastro deixado pelo personagem, da atmosfera de fundo, bem como do estilo visual geral da experiência.

Outro aprimoramento essencial será a adição de um sistema que possibilita ao jogador desligar a música original do jogo e escutar sua própria playlist enquanto joga. Essa escolha visa respeitar a individualidade sonora de cada jogador e reforçar a proposta meditativa e flexível do LoFi. Em complemento, está em desenvolvimento uma funcionalidade de captura de tela, que permitirá ao jogador registrar momentos significativos da sua jornada espacial e compartilhá-los, criando vínculos emocionais e incentivando a permanência no universo de Space Vibes.

Por fim, planeja-se expandir a mecânica com a adição de um novo modo de jogo, inspirado no clássico Asteroids, desenvolvido e publicado pela Atari, onde uma espaçonave deve destruir o máximo de asteroides possível sem ser atingida (ATARI, 1979). Nesse modo alternativo, o astronauta enfrentará corpos celestes em uma dinâmica mais desafiadora, baseada em reflexos e precisão. Essa expansão visa agregar valor à experiência do jogador, sem comprometer a proposta original de imersão e tranquilidade, mas oferecendo diferentes formas de engajamento dentro do mesmo universo estético.

Ao final deste processo, Space Vibes se apresenta como um projeto que cumpre com êxito sua proposta conceitual e entrega uma experiência coesa e sensorial. A integração entre elementos visuais, sonoros e mecânicos resultou em um jogo com identidade própria, capaz de estimular a concentração, promover relaxamento e, ao mesmo tempo, despertar

fascínio visual e sensorial no jogador. O uso de cores neon, as atmosferas dinâmicas dos planetas, a movimentação contínua em gravidade zero, os efeitos de partículas em nebulosas e meteoros, e a trilha sonora suave formam um conjunto harmônico que se diferencia dentro do universo dos jogos independentes.

Os ajustes finais realizados, especialmente aqueles voltados à otimização dos efeitos de partículas e à suavização da movimentação do personagem, foram determinantes para alcançar esse equilíbrio entre estética e funcionalidade. Embora ainda haja espaço para melhorias e expansões, como descrito anteriormente, o resultado atual já configura uma base sólida para futuras versões, mostrando o potencial que uma ideia bem fundamentada pode alcançar, mesmo com recursos e tempo limitados. O projeto reafirma, assim, o poder do design de jogos como uma linguagem expressiva, sensorial e, acima de tudo, cultural.

Em suma, Space Vibes não é apenas um jogo, mas uma proposta artística e experiencial. Ele se propõe a contribuir para o debate sobre como os jogos podem ser plataformas de introspecção e relaxamento, e não apenas de ação e competição. O projeto cumpre seu papel como protótipo funcional e conceitual, abre portas para futuras expansões e serve como evidência de que, mesmo em cenários com restrições de tempo e equipe, é possível criar algo que ressoe profundamente com seus jogadores.

Como responsáveis por todo esse processo esteve o time criativo do Sector 8 Studio, um coletivo independente de desenvolvimento que acredita na intersecção entre arte digital, cultura pop e tecnologia como motor para a criação de produtos inovadores. Mesmo composto por apenas duas pessoas durante este projeto específico, o estúdio demonstrou versatilidade e capacidade de execução, sendo responsável por todas as etapas do processo: concepção, design, programação, arte, sonorização e testes.

A limitação de equipe foi um desafio considerável, mas também uma força propulsora de criatividade e foco. O engajamento e o comprometimento com a proposta estética e conceitual permitiram que o jogo ultrapassasse barreiras técnicas e se afirmasse como um produto autoral e reflexivo.

## REFERÊNCIAS

- NORMAN, Donald A. *Design Emocional: porque adoramos (ou odiamos) os objetos do dia-a-dia*. São Paulo: Rocco, 2008.
- BATEMAN, Chris. *Game Writing: Narrative Skills for Videogames*. Hingham: Charles River Media, 2007.
- CAILLOIS, Roger. *Os jogos e os homens: A máscara e a vertigem*. Lisboa: Edições Cotovia, 1990.
- CHEN, Jenova. *Flow in Games (and Everything Else)*. Master's Thesis, University of Southern California, 2007. Disponível em: <https://www.jenovachen.com/flowingames>. Acesso em: 18 dez. 2024.
- CHEN, Jenova. *Flow* [jogo eletrônico]. Thatgamecompany, 2006.
- CHEN, Jenova. *Flower* [jogo eletrônico]. Thatgamecompany, 2009.
- CHEN, Jenova. *Journey* [jogo eletrônico]. Thatgamecompany, 2012.
- CSIKSZENTMIHALYI, Mihaly. *Flow: The Psychology of Optimal Experience*. New York: Harper & Row, 1990.
- CSIKSZENTMIHALYI, Mihaly; NAKAMURA, Jeanne. The concept of Flow. In: SNYDER, C. R.; LOPEZ, Shane J. (Eds.). *Handbook of Positive Psychology*. New York: Oxford University Press, 2002. p. 89–105.
- GALLOWAY, Alexander R. *Gaming: Essays on Algorithmic Culture*. Minneapolis: University of Minnesota Press, 2006.
- HUIZINGA, Johan. *Homo Ludens: O jogo como elemento da cultura*. São Paulo: Perspectiva, 2000.
- MANOVICH, Lev. *The Language of New Media*. Cambridge: MIT Press, 2001.
- MCGONIGAL, Jane. *Reality is Broken: Why Games Make Us Better and How They Can Change the World*. New York: Penguin Press, 2011.



- NES, George. *Generative Art: A Practical Guide Using Processing*. Shelter Island: Manning Publications, 2009.
- SALEN, Katie; ZIMMERMAN, Eric. *Rules of Play: Game Design Fundamentals*. Cambridge: MIT Press, 2004.
- UNITY TECHNOLOGIES. *Unity User Manual*. Disponível em: <https://docs.unity3d.com>. Acesso em: 18 dez. 2024.
- ZEILEMA, Bastiaan. *Designing for Flow in Games*. Amsterdam: Vrije Universiteit, 2017.
- SMITH, John. *A história dos jogos de tabuleiro: evolução e impacto cultural*. Nova York: Game History Press, 2010.
- COLLINS, K. *Game sound: an introduction to the history, theory, and practice of video game music and sound design*. Cambridge: MIT Press, 2008.
- FULTON, S.; SKINNER, J. *The art of 2D computer animation and imaging*. New York: Van Nostrand Reinhold, 2004.
- JØRGENSEN, K. *A comprehensive study of sound in games*. Aalborg: Aalborg University Press, 2009.
- MURDOCH, D. *3D modeling for beginners: learn everything fast*. New York: CreateSpace Independent Publishing, 2016.
- NESTOR, W. *Unity 4.x game development by example*. Birmingham: Packt Publishing, 2013.
- WATT, A.; POLICARPO, F. *3D games: real-time rendering and software technology*. Boston: Addison-Wesley, 2005.
- BORN, Georgina. *Rationalizing Culture: IRCAM, Boulez, and the Institutionalization of the Avant-Garde*. Berkeley: University of California Press, 1995.
- BATTI, Carlos. *Design Digital: fundamentos e prática para criação visual*. São Paulo: Novatec, 2015.
- BEGOSSO, André. *Design gráfico e comunicação visual*. São Paulo: Blucher, 2017.

GOLDSTONE, Will. *Unity Game Development Essentials*. Birmingham: Packt Publishing, 2009.

KELBY, Scott. *Adobe Photoshop Classroom in a Book*. San Francisco: Adobe Press, 2020.

LANDA, Robin. *Design gráfico: fundamentos e aplicações*. São Paulo: Cengage Learning, 2011.

TOTTEN, Christopher. *An Architectural Approach to Level Design*. Boca Raton: CRC Press, 2014.

FISHER, Mark. *Realismo capitalista: É mais fácil imaginar o fim do mundo do que o fim do capitalismo*. São Paulo: Autonomia Literária, 2020. (Título original: *Capitalist Realism*, 2009)

TANNER, Grafton. *Babbling corpse: vaporwave and the commodification of ghostly media*. Winchester, UK: Zero Books, 2016.

FOLEY, James D. et al. *Computer Graphics: Principles and Practice*. 2. ed. Reading: Addison-Wesley, 1996.

WIKIPÉDIA. *Lofi Girl*. Wikipedia, The Free Encyclopedia, 2024. Disponível em: [https://en.wikipedia.org/wiki/Lofi\\_Girl](https://en.wikipedia.org/wiki/Lofi_Girl). Acesso em: 11 jul. 2025.

BURKART, Patrick; STONE, Nathaniel. *Off the Network: Disrupting the Digital World*. Cambridge: MIT Press, 2020.

CAMPOS, João Victor. *Lo-Fi hip hop como trilha sonora para os estudos: estética e função na contemporaneidade*. Revista Eletrônica Música e Cultura, v. 12, n. 2, p. 45–58, 2021.

GOMES, Laura A. *A sonoridade do cotidiano: LoFi Beats e os novos usos da música ambiente*. Revista Vórtex, Curitiba, v. 10, n. 1, p. 1–20, 2022.

FRITH, Simon. *Popular Music: Critical Concepts in Media and Cultural Studies*. London: Routledge, 2004.

REYNOLDS, Simon. *Retromania: Pop Culture's Addiction to Its Own Past*. London: Faber & Faber, 2012.

UNITY TECHNOLOGIES. *Planos e preços*. 2025. Disponível em: <https://unity.com/pt/pricing>. Acesso em: 7 jul. 2025.

BLENDER FOUNDATION. *Blender: Free and Open Source 3D Creation Software*. 2025. Disponível em: <https://www.blender.org>. Acesso em: 7 jul. 2025.

AUTODESK. *Maya: 3D Animation Software*. 2025a. Disponível em: <https://www.autodesk.com/products/maya>. Acesso em: 7 jul. 2025.

AUTODESK. *3ds Max: 3D Modeling & Rendering Software*. 2025b. Disponível em: <https://www.autodesk.com/products/3ds-max>. Acesso em: 7 jul. 2025.

KUMAR, Hemanth; RAHMAN, Abdul. A deeper look at sprites. In: *Rapid Game Development Using Cocos2d-JS*. Berkeley, CA: Apress, 2016. p. 17–40. DOI: [https://doi.org/10.1007/978-1-4842-2553-0\\_3](https://doi.org/10.1007/978-1-4842-2553-0_3). Acesso em: 7 jul. 2025.

SHELL, Jesse. *The Art of Game Design: A Book of Lenses*. 3. ed. Boca Raton: CRC Press, 2019.

GOLIŃSKI, Michał; GRUBA, Piotr. Game Development Pipeline. In: GAMRAT, W.; KASPRZYK, T. (org.). *Advanced Topics in Game Design*. Warsaw: PJATK Press, 2020.

FARIN, Gerald. *Curves and Surfaces for CAGD: A Practical Guide*. 5. ed. San Francisco: Morgan Kaufmann, 2002.

MCCORMACK, J.; GAN, L.; GOLEMBEWSKI, M. et al. Designing Creativity: Artificial Intelligence and the Future of Creative Work. *Proceedings of the ACM on Human-Computer Interaction*, v. 3, n. CSCW, p. 1–27, 2019.

RETRO DRIVE STUDIO. *Retro Drive: Revamped* [jogo eletrônico]. Steam, 2021. Disponível em: [https://store.steampowered.com/app/1837080/Retro\\_Drive\\_Revamped/](https://store.steampowered.com/app/1837080/Retro_Drive_Revamped/). Acesso em: 17 jul. 2025.

NEW ENGLAND INTERACTIVE. *Synthwave FURY* [jogo eletrônico]. Steam, 2020. Disponível em: [https://store.steampowered.com/app/1451550/Synthwave\\_FURY/](https://store.steampowered.com/app/1451550/Synthwave_FURY/). Acesso em: 17 jul. 2025.

STEAM. *Spirit City: Lofi Sessions*. Mooncube Games. Disponível em:

[https://store.steampowered.com/app/2349630/Spirit\\_City\\_Lofi\\_Sessions](https://store.steampowered.com/app/2349630/Spirit_City_Lofi_Sessions). Acesso em: 17 jul. 2025.

META QUEST STORE. *LoFi Katana*. LoFi Clan. Disponível em:

<https://www.meta.com/experiences/6985798828173122/>. Acesso em: 17 jul. 2025.

REES, Dave. The solitary astronaut: exploring the aesthetics of space and self-reflection.

*Journal of Visual Culture*, v. 20, n. 3, p. 295–310, 2021. DOI: 10.1177/14704129211034277.

Disponível em: <https://journals.sagepub.com/doi/full/10.1177/14704129211034277>. Acesso em: 18 jul. 2025.

IHMASTRONOMY. *The astronaut as a symbol: why we romanticize space isolation*. 2023.

Disponível em: <https://ihmastronomy.com/astronauts-as-symbols>. Acesso em: 18 jul. 2025.

CAPRICHIO. *Astronautas contam como lidam com o isolamento em estações espaciais*. 2020.

Disponível em: <https://capricho.abril.com.br/comportamento/astronautas-contam-como-lidam-com-o-isolamento-em-estacoes-espaciais/>. Acesso em: 18 jul. 2025.

BROWN, Kat. *How LoFi Girl became an icon of YouTube's study culture*. The Guardian,

2021. Disponível em: <https://www.theguardian.com/music/2021/aug/26/how-lofi-girl-became-an-icon-of-youtubes-study-culture>. Acesso em: 18 jul. 2025.

LOFI GIRL. *YouTube Channel Overview*. 2025. Disponível em:

<https://www.youtube.com/c/LofiGirl>. Acesso em: 18 jul. 2025.

MCINTYRE, Hugh. *Why Chillhop Music Is One Of The Most Important Brands In LoFi Hip-Hop*. Forbes, 2020. Disponível em:

<https://www.forbes.com/sites/hughmcintyre/2020/05/28/why-chillhop-music-is-one-of-the-most-important-brands-in-lofi-hip-hop/>. Acesso em: 18 jul. 2025.

COLLEGE MUSIC. *YouTube Channel Overview*. 2025. Disponível em:

<https://www.youtube.com/c/CollegeMusic>. Acesso em: 18 jul. 2025.

OPEN HANDSET ALLIANCE. *Android Overview*. Disponível em:

<https://source.android.com>. Acesso em: 18 jul. 2025.

CASEY, Karl. *Astronomy*. White Bat Audio, 2022. Disponível em:  
<https://www.youtube.com/watch?v=Q7rN7DfxN6g>. Acesso em: 22 jul. 2025.

HOME. *Resonance*. 2014. Disponível em:  
<https://www.youtube.com/watch?v=8GW6sLrK40k>. Acesso em: 22 jul. 2025.

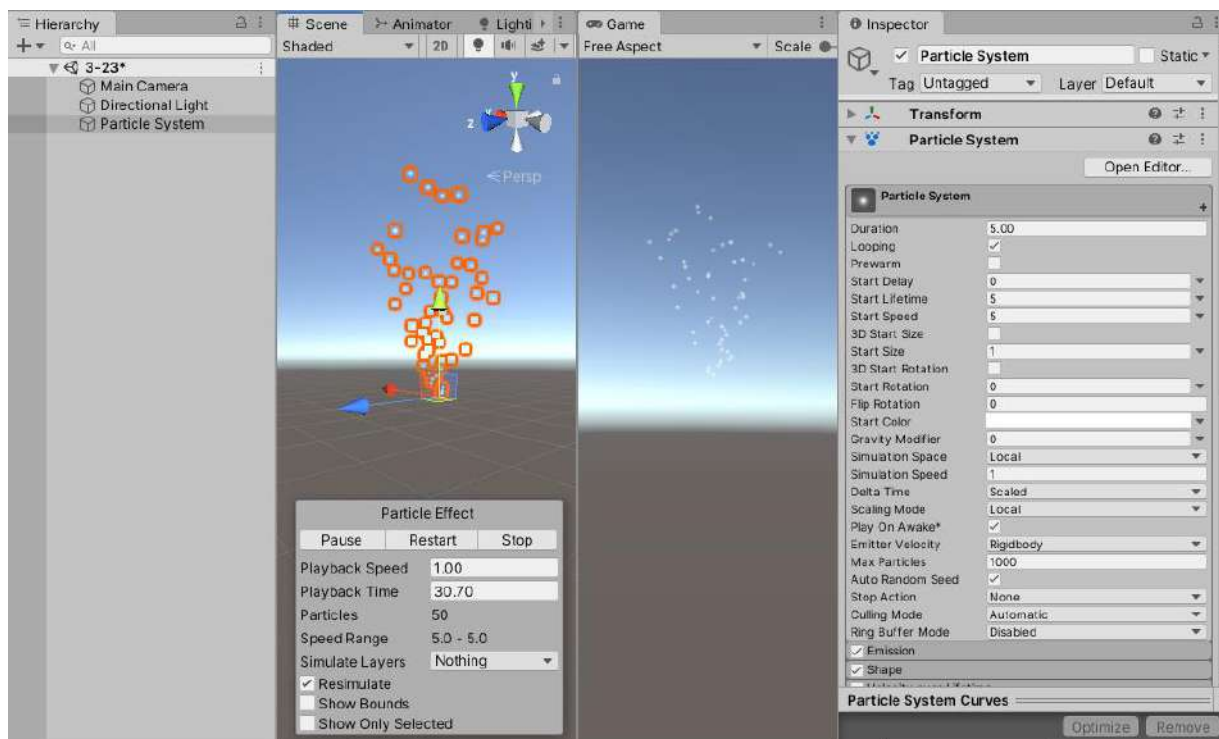
ASTROPHYSICS. *Space Love* (Album). 2019. Disponível em:  
<https://www.youtube.com/watch?v=ZfL4qivGnCM>. Acesso em: 22 jul. 2025.

## APÊNDICE A – FERRAMENTAS TÉCNICAS DE VFX NA UNITY

Este apêndice tem como objetivo apresentar, de forma detalhada, aspectos técnicos e específicos das ferramentas de efeitos visuais (Visual Effects – VFX) disponíveis na engine Unity, com ênfase no Particle System e no Shader Graph. Estas ferramentas são amplamente utilizadas na indústria de jogos digitais para a criação de efeitos visuais dinâmicos e expressivos, como explosões, fumaça, magia, iluminação, distorções e transições atmosféricas.

O Particle System permite a simulação de partículas em tempo real, possibilitando a criação de sistemas complexos com milhares de elementos gráficos leves que respondem a variáveis físicas como gravidade, vento, colisões e tempo de vida. Já o Shader Graph oferece uma interface visual baseada em nós para o desenvolvimento de shaders personalizados, códigos que controlam a aparência visual dos objetos, sem a necessidade de escrever scripts manuais, permitindo o desenvolvimento de materiais interativos, animados ou que respondem a estímulos externos.

Figura 36 - Imagem exibe o layout do Particle System da Unity



Fonte: <https://unity.com/pt>

A figura 36 demonstra a estrutura da aba de Particle System e de como ela funciona. O Particle System possibilita visualizar e ajustar os efeitos em tempo real no Editor<sup>45</sup>. Os desenvolvedores podem personalizar as partículas por meio de uma série de módulos que pode ser observado na imagem no canto direito, eles oferecem controle sobre comportamento, aparência e dinâmica. Outro ponto a se destacar é que as partículas são compatíveis com os shaders da Unity e materiais para adicionar texturas, transparências e outras propriedades visuais.

Ao criar um Particle System no Unity, ele aparece como um GameObject<sup>46</sup> com um componente "Particle System" anexo. O layout principal do Particle System inclui os seguintes elementos:

## INSPECTOR

No Inspector<sup>47</sup>, você encontrará o painel de controle do Particle System. Esse painel é dividido em seções chamadas módulos, que organizam as diversas configurações de forma lógica e acessível. Cada módulo controla um aspecto específico do sistema de partículas, como emissão, forma, velocidade, cor, entre outros. Essa estrutura facilita a personalização do efeito visual, permitindo que você ative apenas os módulos necessários e ajuste os parâmetros conforme o comportamento desejado para as partículas.

## MÓDULOS PADRÃO

Os principais módulos disponíveis no Inspector são:

- Main: Configurações básicas como duração, loop, taxa de emissão, e vida útil das partículas.

---

<sup>45</sup> Editor ou “Editor” na sua tradução, é uma ferramenta de desenvolvimento de jogos 2D e 3D que permite a visualização e edição desses projetos em tempo real, sem necessidade de interromper a simulação que está sendo observada. O editor é personalizável e permite a prototipagem rápida nele o que é produzido, enquanto mecanismo pode ser visualizado.

<sup>46</sup> GameObject ou “Objeto de jogo” na sua tradução, é um objeto fundamental no Unity que representa personagens, cenários e adereços. Ele atua como um contêiner para componentes que implementam a funcionalidade do objeto. Os GameObjects são os blocos de construção das cenas no Unity. Cada GameObject tem uma posição, rotação e escala, que definem onde e como ele aparece no jogo.

<sup>47</sup> Inspector ou “Inspetor” na sua tradução, é uma janela no Editor usada para exibir e editar as propriedades de objetos e componentes selecionados na Scene ou no Hierarchy. Ele permite configurar e personalizar objetos, ajustar valores e visualizar informações detalhadas sobre qualquer elemento.

- **Emission:** Controla a taxa de emissão (número de partículas geradas por segundo ou por evento).
- **Shape:** Define o formato inicial do emissor (esfera, cone, caixa, etc.).
- **Velocity over Lifetime:** Ajusta a velocidade e direção das partículas ao longo do tempo.
- **Color over Lifetime:** Permite transições de cor nas partículas enquanto estão vivas.
- **Size over Lifetime:** Modifica o tamanho das partículas ao longo de sua duração.
- **Renderer:** Configura os materiais, texturas, e aparência geral das partículas.

## VISUALIZAÇÃO NA JANELA DE CENA

O Particle System aparece na Scene View<sup>48</sup>, onde é possível ver as partículas em ação e ajustar os parâmetros em tempo real. Um Gizmo<sup>49</sup> indica a forma do emissor definido no módulo "Shape".

## BOTÕES DE CONTROLE NO INSPECTOR

**Play, Pause, Stop:** Botões para testar o Particle System sem a necessidade de executar o jogo.

**Resimulate:** Atualiza o efeito com as mudanças aplicadas.

## JANELA DO HIERARCHY

O GameObject contendo o Particle System pode incluir múltiplos sistemas de partículas, cada um com configurações específicas. Isso permite criar efeitos complexos ao empilhar diferentes sistemas.

## RECURSOS AVANÇADOS

**Sub Emitters:** Permite criar partículas secundárias acionadas por eventos, como explosões que geram faíscas.

---

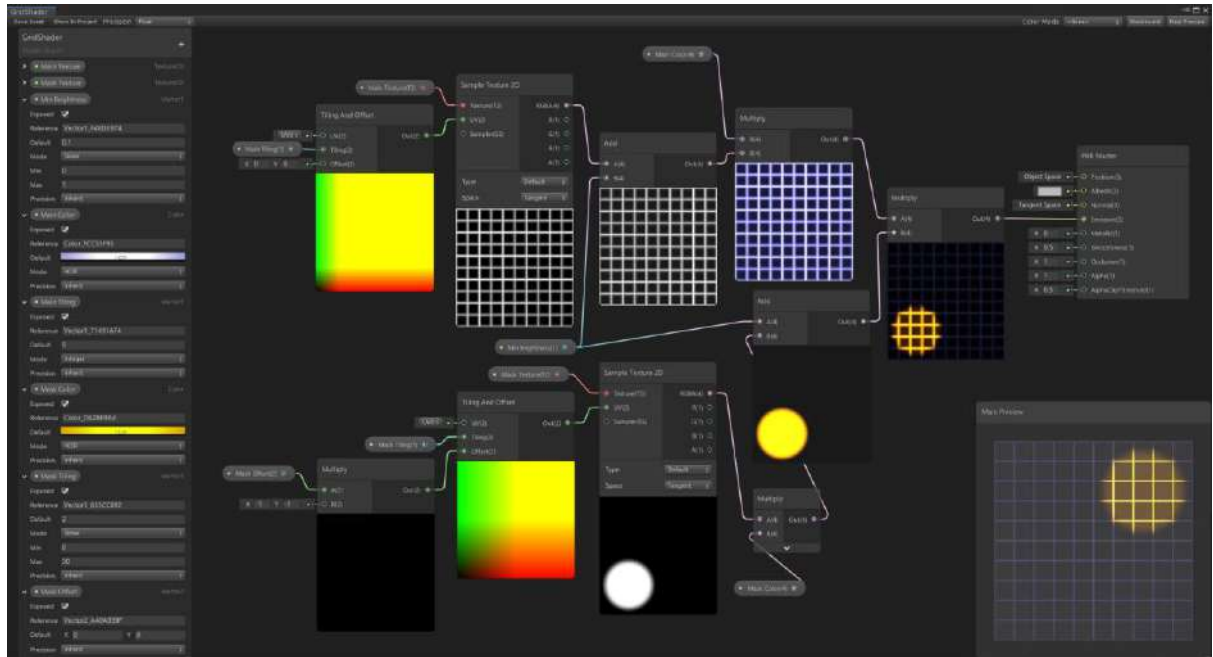
<sup>48</sup> Scene View ou “Visão da Cena” na sua tradução, se trata da janela onde pode ser visualizado a cena que está sendo criada.

<sup>49</sup> Gizmo é uma ferramenta visual usada no Editor para ajudar os desenvolvedores a visualizar e manipular objetos e componentes de forma mais intuitiva. Eles aparecem na Scene View como elementos gráficos que representam informações ou permitem interações específicas.



**Collision:** Configura interações físicas entre partículas e outros objetos da cena.

Figura 37 - Imagem da ferramenta Shader Graph



Fonte: <https://unity.com/pt>

## LAYOUT DO SHADER GRAPH NO UNITY

Ao abrir o Shader Graph, figura 37, a interface é apresentada de forma limpa e organizada, permitindo uma navegação eficiente. O layout é composto pela janela principal: Graph View, que é a área principal onde os nós são conectados para construir o shader.

Todos os nós partem do Nó Principal (Master Node) o ponto final do gráfico, onde todas as conexões convergem para definir a saída do shader. Entre os pontos que fazem composição dentro da interface temos:

- **Surface:** Define a aparência da superfície do material (cor base, transparência, reflexividade).
- **Emission:** Controla a luz emitida pelo material.
- **Normal:** Adiciona detalhes em relevo à superfície.

## JANELA DE BLACKBOARD

Localizada à esquerda, a Blackboard contém propriedades globais do shader, como texturas, cores e valores numéricos. As propriedades da Blackboard podem ser expostas ao Inspector para permitir ajustes diretos nos materiais.

## JANELA DE CONFIGURAÇÕES

Localizada no canto superior direito, onde é possível ajustar configurações gerais do shader, como tipo de material (Lit, Unlit, Subsurface), pipeline de renderização utilizado (URP ou HDRP), suporte para efeitos avançados, como transparência e sombreamento

## JANELA DE PRÉ-VISUALIZAÇÃO

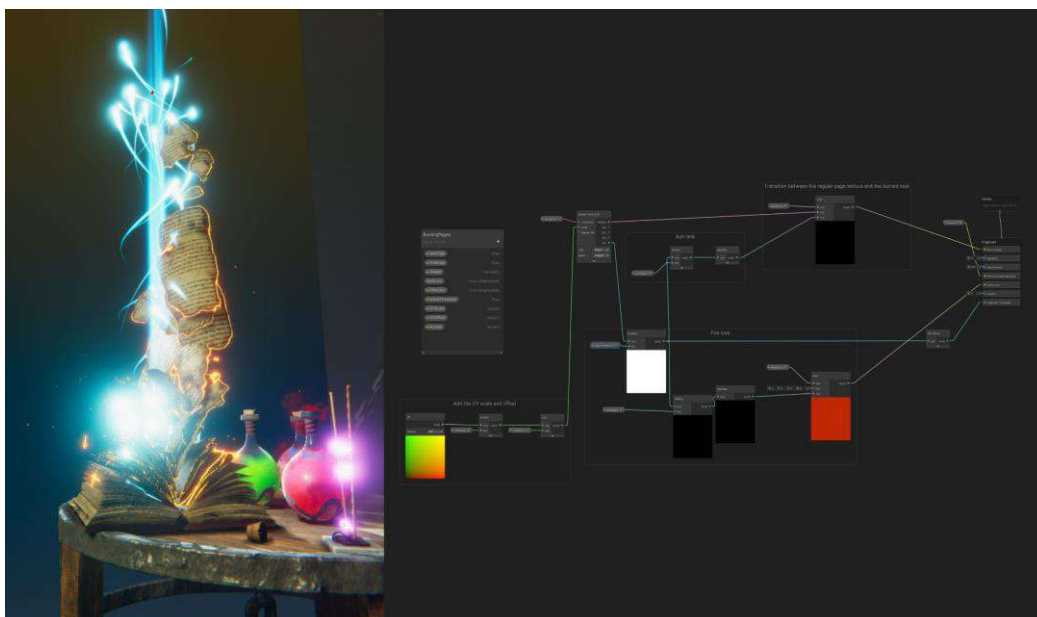
Exibe como o shader será renderizado em um modelo 3D, permitindo ajustes precisos de parâmetros.

## NODOS

Se tratam de blocos modulares usados para criar e personalizar comportamentos do shader. Alguns dos principais tipos de nós incluem:

- **Input Nodes:** Para receber informações de outros nodes contendo parâmetros de texturas, cores, e variáveis numéricas.
- **Math Nodes:** Operações matemáticas para manipular entradas (adição, multiplicação, etc.).
- **Utility Nodes:** Funções específicas, como máscaras, gradientes e lerp (interpolação linear).
- **Output Nodes:** Definem a saída do gráfico, conectando as propriedades ao nó principal.

Figura 38 - Imagem da ferramenta Visual Effect Graph



Fonte: <https://unity.com/pt>

## SUPORTE PARA SIMULAÇÕES COMPLEXAS

O VFX Graph, figura 38, permite a criação de simulações físicas detalhadas, como partículas que respondem à gravidade, vento, colisões e forças customizadas. Isso é possível graças ao uso de computação baseada em GPU<sup>50</sup>, o que possibilita cálculos complexos em tempo real para simular comportamentos naturais e dinâmicos. O VFX Graph permite configurar comportamentos físicos realistas nas partículas. Por exemplo:

- **Gravidade:** Partículas podem cair em direção a um ponto ou seguir uma direção específica, simulando objetos pesados ou leves.
- **Colisões:** Partículas podem colidir com superfícies na cena, refletir, parar ou até interagir com objetos físicos usando camadas de colisão definidas pelo usuário.
- **Forças:** É possível adicionar forças personalizadas, como turbulência, vento ou até campos de força gravitacional que afetam as partículas dinamicamente.

Essas simulações são realizadas na GPU, garantindo que grandes volumes de partículas possam ser processados em tempo real sem comprometer o desempenho.

<sup>50</sup> GPU é a sigla para Unidade de Processamento Gráfico, um circuito eletrônico que realiza cálculos matemáticos complexos em alta velocidade. As GPUs são utilizadas em computadores pessoais, estações de trabalho e videogames. Elas são responsáveis por renderizar gráficos e imagens, e são essenciais para a execução de tarefas como edição de vídeo, machine learning e renderização gráfica.

## USO DE MÚLTIPLOS SISTEMAS DE PARTÍCULAS

É possível criar efeitos compostos utilizando vários sistemas de partículas em uma única estrutura hierárquica. Isso permite combinar diferentes comportamentos em um único gráfico, como faíscas geradas por uma explosão acompanhadas por fumaça e detritos. Você pode conectar diferentes sistemas de partículas dentro de um único gráfico para criar efeitos complexos. Por exemplo:

- **Explosão:** Um sistema pode gerar as partículas de detritos, enquanto outro adiciona a fumaça e faíscas.
- **Fogo:** Um sistema pode cuidar das chamas e outro das brasas que sobem lentamente.

Cada sistema pode ter configurações únicas, como tempo de vida, forma de emissão, e comportamentos personalizados, mas todos coexistem de maneira integrada.

## EDIÇÃO VISUAL COM NODES

Assim como no Shader Graph, o VFX Graph utiliza uma interface nodal para criar os efeitos. A lógica do comportamento das partículas, como sua emissão, vida útil, cor e movimento, é configurada conectando nós, tornando o processo mais intuitivo para quem prefere uma abordagem visual ao invés de código. No editor do VFX Graph cada nó representa uma funcionalidade específica, como:

- **Emitir Partículas:** Controla como e quando as partículas são emitidas.
- **Atualizar:** Define comportamentos ao longo da vida das partículas, como mudança de cor, rotação ou escala.
- **Renderizar:** Define como as partículas serão exibidas, incluindo materiais, texturas e outros efeitos visuais.

Essa estrutura modular permite criar fluxos de trabalho intuitivos, ajustáveis diretamente no editor com visualização em tempo real.

## CONTROLE AVANÇADO DE PROPRIEDADES

O VFX Graph oferece controles personalizáveis para propriedades como cor, escala, rotação, velocidade e direção das partículas. Essas propriedades podem ser animadas com base no tempo, na posição da câmera ou em interações do jogador, permitindo criar efeitos

dinâmicos e responsivos ao ambiente. As propriedades das partículas podem ser manipuladas de várias formas:

- **Interações Baseadas no Tempo:** Por exemplo, partículas podem mudar de cor ou aumentar de escala à medida que envelhecem.
- **Curvas e Gradientes:** Oferecem controle preciso sobre como variáveis, como opacidade ou cor, evoluem ao longo do tempo de vida das partículas.
- **Parâmetros Externos:** Propriedades podem ser controladas dinamicamente em runtime, permitindo que efeitos sejam influenciados por variáveis do jogo, como a posição do jogador ou eventos específicos.

## INTEGRAÇÃO COM SHADERS PERSONALIZADOS

O VFX Graph pode ser combinado com shaders customizados criados no Shader Graph, permitindo ainda mais personalização na aparência das partículas. Isso é útil para criar materiais únicos que podem incluir transparência, brilho, texturas animadas e efeitos de distorção. Esse fato expande a flexibilidade visual dos efeitos. Exemplos:

- **Distorções:** Criar partículas que distorcem o fundo para simular calor ou ondas de impacto.
- **Transparência Dinâmica:** Aplicar gradientes ou texturas animadas para criar efeitos de dissolução ou transição.
- **Brilho e Emissão:** Adicionar efeitos de partículas brilhantes que interagem com o ambiente.

Essa integração é crucial para criar efeitos visuais únicos que se ajustem ao estilo artístico do jogo.

## SUPORTE PARA TEXTURAS E MAPAS 3D

O VFX Graph permite o uso de texturas e mapas 3D (como mapas de altura ou densidade) para definir comportamentos ou influenciar a aparência das partículas. Por exemplo, partículas podem seguir o formato de uma malha 3D ou reagir ao terreno do jogo.

- **Texturas 2D:** Podem ser usadas para animar partículas ou definir cores e padrões visuais.

- **Mapas de Densidade 3D:** Permitem que partículas sigam formas volumétricas, como fumaça fluindo dentro de um cubo ou partículas se moldando à silhueta de um objeto 3D.

Essa funcionalidade permite criar efeitos que interagem visualmente com o ambiente de forma sofisticada.

## EVENTOS E INTERATIVIDADE

A ferramenta suporta eventos, o que possibilita a criação de efeitos interativos. Por exemplo, um feitiço pode gerar partículas que mudam de comportamento ao colidir com um inimigo, ou uma explosão pode desencadear a emissão de fragmentos em diferentes direções. As partículas podem reagir dinamicamente a ações no jogo:

- **Exemplo 1:** Quando um projétil atinge o solo, ele pode acionar uma explosão com partículas de fumaça e detritos.
- **Exemplo 2:** O movimento do jogador pode influenciar o comportamento de partículas ao redor, como poeira sendo levantada.

Os eventos são configurados diretamente no gráfico e podem ser acionados via scripts ou interações no ambiente.

## SUPORTE A ILUMINAÇÃO E PÓS-PROCESSAMENTO

Os efeitos criados no VFX Graph podem interagir com a iluminação da cena, integrando-se aos sistemas de luz e sombra do Unity. Além disso, eles funcionam perfeitamente com efeitos de pós-processamento, como <sup>51</sup>bloom, <sup>52</sup>depth of field e <sup>53</sup>motion blur, para adicionar mais realismo e impacto visual. Os efeitos do VFX Graph podem interagir com as fontes de luz da cena e com sistemas de pós-processamento:

---

<sup>51</sup> O efeito bloom é um efeito de imagem que simula o vazamento de luz de uma lente imperfeita. Ele é usado para reproduzir o artefato de imagem de câmeras do mundo real.

<sup>52</sup> O efeito depth of field (DOF), ou profundidade de campo, é um efeito ótico que descreve o quanto um objeto está focado em relação aos outros objetos de uma cena.

<sup>53</sup> O efeito motion blur é um desfoque que ocorre quando um objeto se move mais rápido do que a capacidade de captura de uma câmera ou do olho humano.

- **Iluminação:** Partículas podem emitir luz própria (efeitos emissivos) ou refletir a iluminação do ambiente, criando maior realismo.
- **Pós-Processamento:** Efeitos como bloom, depth of field e motion blur podem ser aplicados para realçar as partículas e integrá-las melhor à cena.

Essa integração garante que os efeitos visuais fiquem harmoniosos com o restante do cenário.

## DEBUG E VISUALIZAÇÃO EM TEMPO REAL

O VFX Graph inclui ferramentas para depurar e visualizar a lógica dos efeitos diretamente no editor. Isso permite ajustar rapidamente o comportamento das partículas, ver como elas se comportam em diferentes condições e otimizar o desempenho em tempo real.

- **Visualização de Parâmetros:** É possível ver o comportamento das partículas em tempo real ao ajustar propriedades no editor.
- **Perfis de Performance:** O sistema exibe informações sobre o impacto do efeito na performance, permitindo ajustes para otimizar o uso da GPU.

Essas ferramentas ajudam a garantir que os efeitos não apenas pareçam incríveis, mas também sejam otimizados para diferentes plataformas.

## ESCALABILIDADE E OTIMIZAÇÃO

Projetado para rodar em uma ampla gama de dispositivos, o VFX Graph inclui opções para ajustar a complexidade dos efeitos, como reduzir o número de partículas em plataformas menos potentes. Ele também utiliza instâncias de partículas, reduzindo o impacto na performance. O VFX Graph foi projetado para ser escalável, oferecendo controle sobre:

- **LOD (Level of Detail):** Reduz automaticamente a complexidade dos efeitos em dispositivos menos potentes.
- **Número de Partículas:** É possível limitar a quantidade de partículas processadas sem perder a qualidade geral do efeito.
- **Otimização por Pipeline:** O VFX Graph funciona melhor com HDRP e URP, utilizando recursos do SRP para melhorar desempenho e visuais.

Isso permite que desenvolvedores adaptem seus projetos para diferentes plataformas, de consoles de última geração a dispositivos móveis.

## **UNITY ANIMATOR CONTROLLER COMO RECURSO PARA CONTROLE DE ANIMAÇÕES**

O Unity Animator Controller é uma ferramenta nativa da engine Unity que permite ao desenvolvedor controlar e manipular o comportamento de animações dentro do ambiente de jogo. Com ele, é possível organizar estados animados, condições de transição, sincronizações com áudio e lógica de gameplay. Will Goldstone destaca que:

“O Animator Controller transforma a lógica do jogo em movimento visual, sendo um elo entre o código e a experiência sensível do jogador”.(Will Goldstone, 2009, p. 88)

## **A RELEVÂNCIA DO ANIMATOR CONTROLLER NA IMERSÃO E FEEDBACK VISUAL NOS JOGOS**

No design de jogos, o uso estratégico de animações é uma das formas mais eficazes de fornecer feedback visual ao jogador, aprimorar a narrativa visual e reforçar o ritmo da gameplay. O Animator Controller é fundamental nesse processo, permitindo que animações sejam condicionadas a eventos específicos, intensificando o engajamento e a sensação de resposta imediata às ações do usuário.

“Animações bem estruturadas reforçam a imersão do jogador no universo do jogo, e o uso de ferramentas como o Animator Controller permite alinhar design visual, som e interatividade em tempo real”.(Christopher Totten, 2014, p. 159)

A Unity Animator Controller representa um avanço técnico e criativo na maneira como o comportamento de personagens e elementos animados são integrados aos jogos. Ao permitir a organização lógica de estados e transições de animações em uma interface visual intuitiva, ela proporciona ao desenvolvedor um controle preciso sobre os aspectos dinâmicos da narrativa e da jogabilidade. Em jogos que exigem fluidez e resposta emocional por meio de animações, essa ferramenta torna-se indispensável, viabilizando interações mais realistas e



coesas. Sua implementação impacta diretamente na imersão do jogador, transformando os recursos animados em parte orgânica do storytelling interativo.

## **APÊNDICE B – DOCUMENTAÇÃO TÉCNICA: ARQUIVOS DE DESENVOLVIMENTO E CÓDIGOS DO JOGO**

Este apêndice tem como objetivo fornecer acesso direto a materiais complementares relacionados ao desenvolvimento do jogo Space Vibes. Por meio do link disponibilizado abaixo, é possível visualizar imagens dos principais scripts utilizados, além de trechos de código, estruturas de prefabs, organização de assets e demais elementos técnicos que compõem o projeto.

Essa documentação serve para demonstrar a organização interna do desenvolvimento, bem como facilitar o entendimento de aspectos práticos abordados ao longo do trabalho. A disponibilização online visa também garantir transparência, reprodutibilidade e acesso rápido aos interessados no aprofundamento técnico da produção. Link de acesso: [https://drive.google.com/drive/folders/1rxp70nEvW2OYZ27iz4h0\\_BUdU8OHVdKU?usp=sharing](https://drive.google.com/drive/folders/1rxp70nEvW2OYZ27iz4h0_BUdU8OHVdKU?usp=sharing)