

**INSTITUTO
FEDERAL**
São Paulo

INSTITUTO FEDERAL DE SÃO PAULO
CAMPUS SÃO PAULO

Ensino Médio Técnico Em Desenvolvimento de Sistemas

Iris Silva Lorelli
Isabela Gomes de Oliveira
Mateus de Sousa Martins Ferreira Silva
Kenned Lucas Pereira da Silva
Thiago Daiki Sato

Entre Salas e Silêncios
Dentro da Culpa

São Paulo
2025

Entre Salas e Silêncios

Dentro da culpa

Documentação de Projeto Interdisciplinar apresentado como requisito parcial para

avaliação nas disciplinas do técnico em Desenvolvimento de Sistemas

Nome dos Professores (Banca):

Prof^a. Claudete Alves

Prof^a. Ana Lucia Grici Zacarin Mamede

Prof^a. Claudia Miyuki Werhmuller

Prof^a. Daniela dos Santos Santana

Prof^a. Gislene Pereira de Oliveira Martins

Prof^a. Paula Neves de Araújo

Prof. João Antonio Temochko Andre

Prof. Paulo Henrique Netto de Alcantara

Prof. André Evandro Lourenço

Prof. Antonio Ferreira Viana

São Paulo

2025

3.1 Design e Arte.....	8
3.2 Arquitetura e Código.....	9
3.3 O Site de Divulgação (HTML/CSS).....	13
Processo de Testes.....	15
Bugs Encontrados e Correções.....	15
Resultado Final.....	15
Vídeo de Gameplay.....	17

INTRODUÇÃO

O projeto interdisciplinar proposto pelos professores teve como principal desafio o desenvolvimento de um jogo digital autoral, integrando conhecimentos das disciplinas de programação, design e narrativa interativa. A proposta visava estimular a criatividade, o trabalho em equipe e a aplicação prática dos conteúdos aprendidos ao longo do semestre.

Este documento tem como objetivo apresentar todas as etapas do desenvolvimento do jogo “Entre Salas e Silêncios”, desde o planejamento inicial até os testes finais, passando pelo design visual, a estrutura de código, a construção narrativa e a criação de um site de divulgação.

A ideia central do projeto foi criar um jogo de terror psicológico com elementos narrativos profundos e mecânicas que quebram a quarta parede, desafiando a percepção do jogador sobre o que é real dentro da experiência. O enredo gira em torno de Clara, uma jovem que acorda em um hospital abandonado e, ao explorar ambientes como uma escola sombria e um hospital em ruínas, descobre fragmentos de sua própria história e enfrenta as consequências de seus atos passados. O jogo mistura exploração, suspense, puzzles e momentos de tensão, com múltiplas camadas narrativas e finais alternativos.

A escolha por esse tipo de jogo se deu pelo interesse da equipe em criar uma experiência imersiva e emocional, que fosse além da jogabilidade tradicional. O gênero de terror psicológico foi escolhido por permitir a construção de atmosferas densas e narrativas impactantes, além de possibilitar o uso criativo de recursos como a quebra da quarta parede. O público-alvo são adolescentes e jovens adultos que apreciam jogos com enredos profundos, estética retrô (pixel art) e mecânicas inovadoras.

Este documento apresenta o planejamento do projeto, as pesquisas realizadas, o design visual e sonoro, a arquitetura do código, os testes realizados e os resultados obtidos com o desenvolvimento do jogo “Entre Salas e Silêncios”. Também inclui reflexões sobre os aprendizados da equipe e sugestões para melhorias futuras.

PLANEJAMENTO DO PROJETO

Antes de iniciar o desenvolvimento do jogo “Entre Salas e Silêncios”, a equipe realizou uma série de pesquisas para definir o estilo, a narrativa e as mecânicas que seriam utilizadas. Como referência principal, foram analisados os jogos “MISADE” e “Little Nightmares”, ambos conhecidos por sua atmosfera sombria, narrativa fragmentada e elementos de terror psicológico. Esses títulos influenciaram diretamente a construção do enredo, a ambientação e o estilo visual do projeto.

O escopo do jogo foi definido como uma experiência narrativa interativa com foco em exploração, quebra da quarta parede e construção de suspense. O jogador controla Clara, uma personagem que percorre ambientes como hospital e escola coletando objetos, enfrentando perseguições e reconstruindo sua história por meio de fragmentos de memória. O jogo apresenta múltiplos finais, dependendo das escolhas feitas pelo jogador.

Para o desenvolvimento técnico, foi escolhida a engine **GameMaker**, por sua acessibilidade e eficiência na criação de jogos 2D. A linguagem utilizada foi **GML (GameMaker Language)**, que permitiu a implementação de mecânicas como perseguições, coleta de objetos e transições entre fases. A arte em pixel foi criada manualmente pela equipe, com foco em transmitir sensações de isolamento e tensão. Sons e trilhas foram selecionados e editados para reforçar a ambientação e os momentos de suspense.

A equipe se organizou por meio de um diário de bordo, registrando as principais decisões mensais:

- **Mês 1:** Definição do enredo, divisão de tarefas e escolha das ferramentas.
- **Mês 2:** Criação dos sprites, ambientações e primeiros testes com GameMaker.
- **Mês 3:** Implementação das mecânicas de coleta, perseguição e quebra da quarta parede.
- **Mês:** Finalização das fases, ajustes de som e testes de jogabilidade.

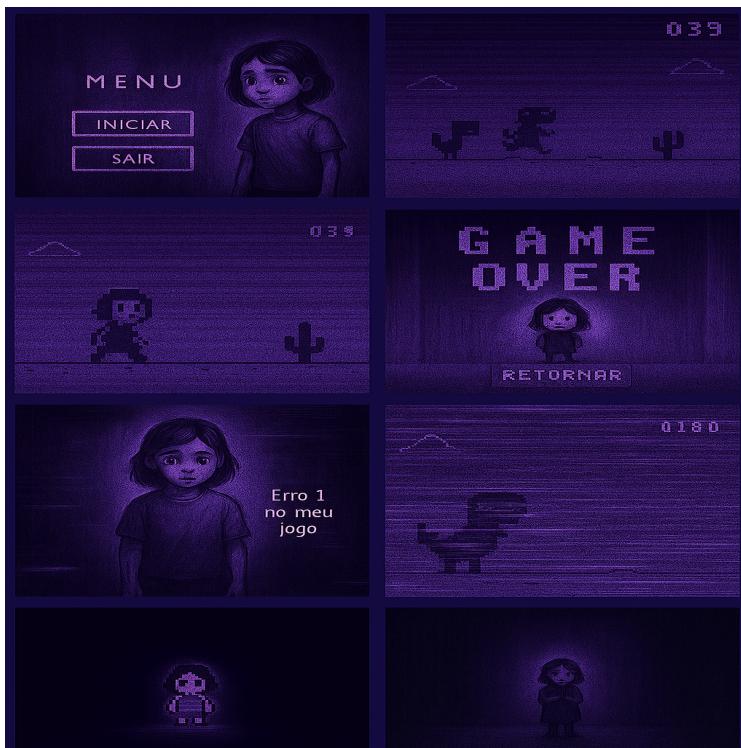
Storyboards:

A narrativa do jogo foi dividida em quatro capítulos principais, cada um com ambientações e objetivos específicos. Abaixo está a representação visual da sequência de fases e eventos que compõem a jornada da personagem Clara.

- *Figura 1 – Storyboard do Capítulo 1: Hospital*
- *Figura 2 – Storyboard do Capítulo 2: Escola*
- *Figura 3 – Storyboard do Capítulo 3: Verdade Fragmentada*
- *Figura 4 – Storyboard do Capítulo Final: Julgamento*

ENTRE SALAS e SILÊNCIOS

STORYBOARDS



DESENVOLVIMENTO

3.1 Design e Arte

O estilo visual do jogo foi desenvolvido em **pixel art**, com foco em transmitir sensações de isolamento, suspense e melancolia. A paleta de cores foi escolhida para reforçar o clima sombrio dos ambientes, com tons escuros e iluminação pontual, focando nas cores pretas e roxas.

Os cenários foram criados para representar locais como um hospital abandonado e uma escola deteriorada. Cada ambiente possui elementos visuais que dialogam com a narrativa e os eventos traumáticos da personagem Clara.

A personagem principal foi desenhada em duas versões: uma menina pequena, que representa a fase inicial do jogo, e uma versão maior, que aparece em momentos de quebra da quarta parede e revelações.



O monstro foi criado com formas distorcidas e olhos apagados, aparecendo em momentos-chave para provocar tensão e perseguições.



3.2 Arquitetura e Código

A lógica do jogo foi construída na engine **GameMaker**, utilizando a linguagem **GML** (**GameMaker Language**). A estrutura foi organizada em salas (rooms), objetos interativos e scripts que controlam os eventos narrativos e mecânicos.

A lógica principal envolve o uso de condicionais (**if/else**) para verificar interações com objetos, ativar eventos e controlar o comportamento do monstro. Laços (**while**) foram usados para monitorar estados do jogo, como a coleta de itens e o avanço de fases.

Uma função importante criada foi **verificarColeta(objeto)**, que identifica quando Clara coleta um item-chave e ativa a próxima etapa da narrativa. Também foram desenvolvidos scripts para controlar a perseguição do monstro, a quebra da quarta parede e os diálogos interativos.

Códigos importantes -

```
function estado() constructor
{
    static inicia = function() {};
    static roda = function () {};
    static finaliza = function () {};
}

function inicia_estado(_estado)
{
    estado_atual = _estado;
    estado_atual.inicia();
}

function roda_estado()
{
    estado_atual.roda();
}

function troca_estado(_estado)
{
    estado_atual.finaliza();
    estado_atual = _estado;
    estado_atual.inicia();
}

function define_sprite(_dir = 0, _sprite_right, _sprite_front, _sprite_back, _sprite_left)
{
    var _sprite;

    switch(_dir)
    {
        case 0: _sprite = _sprite_right; break;
        case 1: _sprite = _sprite_back; break;
        case 2: _sprite = _sprite_left; break;
        case 3: _sprite = _sprite_front; break;
    }
    return _sprite;
}
```

```

        troca_estado(estado_walk);
    }
}
#endifregion

#endifregion estado_walk
Estado_walk.inicia = function()
{
    // 1. Zera velocidades para garantir um novo movimento
    veloc_h = 0;
    veloc_v = 0;

    // 2. Define a velocidade com base na direção randomizada (direcao_patrulha)
    switch(direcao_patrulha)
    {
        case "direita":
            veloc_h = velocidade_base;
            break;

        case "esquerda":
            veloc_h = -velocidade_base; // Velocidade negativa para ir para a esquerda
            break;

        case "frente":
            veloc_v = velocidade_base;
            break;

        case "tras":
            veloc_v = -velocidade_base; // Velocidade negativa para ir para cima
            break;
    }

    // 3. Inicialização de Sprite e Timer
    // O sprite será definido na função .roda()
    image_index = 0;
    timer_estado = tempo_estado; // Define por quanto tempo ele andará
}

Estado_walk.roda = function()
{
    move_and_collide(veloc_h, veloc_v, Object6);

    // --- 2. Atualiza a Direção e o Sprite ---
    // previous_sprite_index = sprite_index; // Se quiser usar a verificação acima

    if (abs(veloc_h) > abs(veloc_v))
    {
        if (veloc_h > 0)
        {
            direcao_atual = "direita";
            sprite_index = elizabeth_direita;
        }
        else
    }
}
Ch:20

```

```

velocidade_base = 1;
distancia_percepcao = 150;
distancia_desengajamento = 200;
direcao_patrulha = "frente";

direcao_atual = "frente";
estado_idle = new estado();
estado_walk = new estado();
estado_chase = new estado();

tempo_estado = game_get_speed(gamespeed_fps)* 1;
timer_estado = tempo_estado;
veloc_h = 0;
veloc_v = 0;

#region estado_idle
estado_idle.inicia = function()
{
    switch(direcao_atual)
    {
        case "frente":
            sprite_index = spr_idle_frente;
            break;
        case "tras":
            sprite_index = spr_idle_tras;
            break;
        case "direita":
            sprite_index = spr_idle_direito;
            break;
        case "esquerda":
            sprite_index = spr_idel_esquerdo;
            break;
    }
    image_index = 0;
}
estado_idle.roda = function()
{
    timer_estado--;
    if(timer_estado <= 0)
    {
        var _escolha = irandom(3);
        switch(_escolha)
        {
            case 0: direcao_patrulha = "frente"; break; // Baixo
            case 1: direcao_patrulha = "tras"; break; // Cima
            case 2: direcao_patrulha = "direita"; break;
            case 3: direcao_patrulha = "esquerda"; break;
        }
    }
}

```

Esse trecho ativa a perseguição do monstro no hospital, criando um momento de tensão e urgência para o jogador.

```

1 min_tempo_cacto = 40;
2 max_tempo_cacto = 120;
3 alarm[0] = 300;
4
5 global.ID_CACTUS_PEQUENO = asset_get_index("obj_cactus_pequenos");
6 global.ID_CACTUS_GRANDE = asset_get_index("obj_cactus");
7
8 pontuacao = 0;
9 pontuacao_limite_inicio = 20;
10 pontuacao_limite_travamento = 50;
11 fase_dinossauro_travada = false;
12
13 global.invulneravel = true;
14 alarm[2] = 30;
15
16 global.pontuacao = 0;
17 global.forcar_gameover = false;
18 global.fase_menina = false;
19 global.pausar = false;
20 global.estado_jogo = "jogando";
21 global.bug_ativado = false;
22
23 if (room == rm_fase_escura) {
24     tempo = 0;
25     global.pausar = false;
26 }
27
28 if (room == rm_explica_menina_grande) {
29 }
30
31

```

Código acima controla todo o jogo, determinando onde cada objeto vai e determinando a pontuação no jogo.

```
var move_speed = 5;
velh = 0;

if (keyboard_check(vk_right)) {
    velh = move_speed;
    image_xscale = 1;
}
if (keyboard_check(vk_left)) {
    velh = -move_speed;
    image_xscale = -1;
}
x += velh;

velv += grav;
y += velv;

if( y >= 725){
    y = 725;
    velv = 0;
    if (keyboard_check_pressed(vk_space) or keyboard_check_pressed(vk_up)){
        velv -= 18;
    }
    image_speed = 1;
    if (velh != 0) {
        sprite_index = spr_menina_andando;
    }
}
```

```
image_speed = 1;

if (velh != 0) {
    sprite_index = spr_menina_andando;
}
else if (!keyboard_check(vk_down)) {
    sprite_index = spr_menina_parada;
}

if (keyboard_check(vk_down)){
    sprite_index = spr_menina_parada;
}

} else {

    image_speed = 0;

    if(keyboard_check(vk_down)){
        grav = 2;
    }else{
        grav = 0.7;
    }
}
```

Código acima gerênciando quando a personagem anda na fase inicial, mostrando como pula e posição dela no ambiente do jogo.

3.3 O Site de Divulgação (HTML/CSS)

O site foi criado com o objetivo de **divulgar o jogo**, apresentar a sinopse, os personagens e disponibilizar o link para download. A estética do site segue a identidade visual do jogo, com fundo escuro, fontes pixeladas e imagens dos cenários.



A screenshot of the 'Sobre' (About) page from the game's website. The page has a dark background with a central white text area. The title 'Sobre:' is at the top. Below it, there is a block of text describing the game: 'Entre Salas e Silêncios é um jogo de terror psicológico com elementos de suspense e quebra da Quarta Parede. Você joga como Clara, uma jovem que acorda em um hospital abandonado sem saber como morreu. Para alcançar a paz e seguir para o céu, ela precisa descobrir quem a matou e por que ainda está presa entre mundos. A jornada passa por locais marcantes como o hospital e a escola onde estudava, revelando memórias distorcidas, segredos sombrios e aparições de uma entidade misteriosa que a persegue. Conforme Clara coleta pistas e revive momentos do passado, ela se aproxima da verdade e de uma escolha final que determinará seu destino.' At the bottom of the text block, there is a smaller line of text: 'Prepare-se para uma experiência imersiva, cheia de tensão, sustos e reflexões sobre culpa, redenção e identidade.'

[íncio](#) [sobre](#) [personagens](#) [storyboards](#) [códigos](#)

Personagens:



[íncio](#) [sobre](#) [personagens](#) [storyboards](#) [códigos](#)

Códigos

```
function checkCollision(x1, y1, w1, h1, x2, y2, w2, h2) {
    let dx = x2 - x1;
    let dy = y2 - y1;
    let distance = Math.sqrt(dx * dx + dy * dy);
    if (distance <= w1 / 2 + w2 / 2) {
        return true;
    }
    return false;
}

function movePlayer() {
    let playerX = document.getElementById("player").x;
    let playerY = document.getElementById("player").y;
    let playerW = document.getElementById("player").width;
    let playerH = document.getElementById("player").height;

    let enemyX = document.getElementById("enemy").x;
    let enemyY = document.getElementById("enemy").y;
    let enemyW = document.getElementById("enemy").width;
    let enemyH = document.getElementById("enemy").height;

    if (checkCollision(playerX, playerY, playerW, playerH, enemyX, enemyY, enemyW, enemyH)) {
        alert("Game Over!");
        window.location.reload();
    }
}
```

```
function movePlayer() {
    let playerX = document.getElementById("player").x;
    let playerY = document.getElementById("player").y;
    let playerW = document.getElementById("player").width;
    let playerH = document.getElementById("player").height;

    let enemyX = document.getElementById("enemy").x;
    let enemyY = document.getElementById("enemy").y;
    let enemyW = document.getElementById("enemy").width;
    let enemyH = document.getElementById("enemy").height;

    if (checkCollision(playerX, playerY, playerW, playerH, enemyX, enemyY, enemyW, enemyH)) {
        alert("Game Over!");
        window.location.reload();
    }
}

function moveEnemy() {
    let enemyX = document.getElementById("enemy").x;
    let enemyY = document.getElementById("enemy").y;
    let enemyW = document.getElementById("enemy").width;
    let enemyH = document.getElementById("enemy").height;

    let playerX = document.getElementById("player").x;
    let playerY = document.getElementById("player").y;
    let playerW = document.getElementById("player").width;
    let playerH = document.getElementById("player").height;

    if (checkCollision(enemyX, enemyY, enemyW, enemyH, playerX, playerY, playerW, playerH)) {
        alert("Game Over!");
        window.location.reload();
    }
}
```

```
function movePlayer() {
    let playerX = document.getElementById("player").x;
    let playerY = document.getElementById("player").y;
    let playerW = document.getElementById("player").width;
    let playerH = document.getElementById("player").height;

    let enemyX = document.getElementById("enemy").x;
    let enemyY = document.getElementById("enemy").y;
    let enemyW = document.getElementById("enemy").width;
    let enemyH = document.getElementById("enemy").height;

    if (checkCollision(playerX, playerY, playerW, playerH, enemyX, enemyY, enemyW, enemyH)) {
        alert("Game Over!");
        window.location.reload();
    }
}

function moveEnemy() {
    let enemyX = document.getElementById("enemy").x;
    let enemyY = document.getElementById("enemy").y;
    let enemyW = document.getElementById("enemy").width;
    let enemyH = document.getElementById("enemy").height;

    let playerX = document.getElementById("player").x;
    let playerY = document.getElementById("player").y;
    let playerW = document.getElementById("player").width;
    let playerH = document.getElementById("player").height;

    if (checkCollision(enemyX, enemyY, enemyW, enemyH, playerX, playerY, playerW, playerH)) {
        alert("Game Over!");
        window.location.reload();
    }
}
```

O site foi desenvolvido com **HTML e CSS**, estruturado em seções como cabeçalho, conteúdo principal e rodapé. O CSS foi utilizado para definir o estilo visual, incluindo cores, fontes e animações sutis que reforçam o clima do jogo.

PROTÓTIPO E TESTES

Processo de Testes

Durante o desenvolvimento, o jogo passou por diversos testes para garantir que as mecânicas funcionassem corretamente e que a narrativa fosse compreensível. Foram realizados testes internos pela equipe e também testes externos com colegas de sala, que jogaram as fases e deram feedback sobre jogabilidade, ambientação e clareza dos objetivos.

Os testes foram divididos em três etapas:

- **Teste de mecânicas:** coleta de objetos, perseguições, transições entre fases.
- **Teste de narrativa:** compreensão da história, impacto das falas e quebra da quarta parede.
- **Teste de desempenho:** tempo de carregamento, fluidez das animações e resposta dos comandos.

Bugs Encontrados e Correções

Durante os testes, alguns bugs foram identificados:

- **Bug 1:** O monstro não aparecia após a coleta do terceiro documento. *Correção:* Ajuste na variável `item_coletado` e no evento de ativação do monstro.
- **Bug 2:** Clara atravessava paredes em algumas salas. *Correção:* Reconfiguração das colisões nos objetos de parede.
- **Bug 3:** O jogo travava ao tentar salvar na fase da escola. *Correção:* Remoção da função de salvamento temporário e inserção de mensagem narrativa (“Você não pode salvar o que já está perdido”).

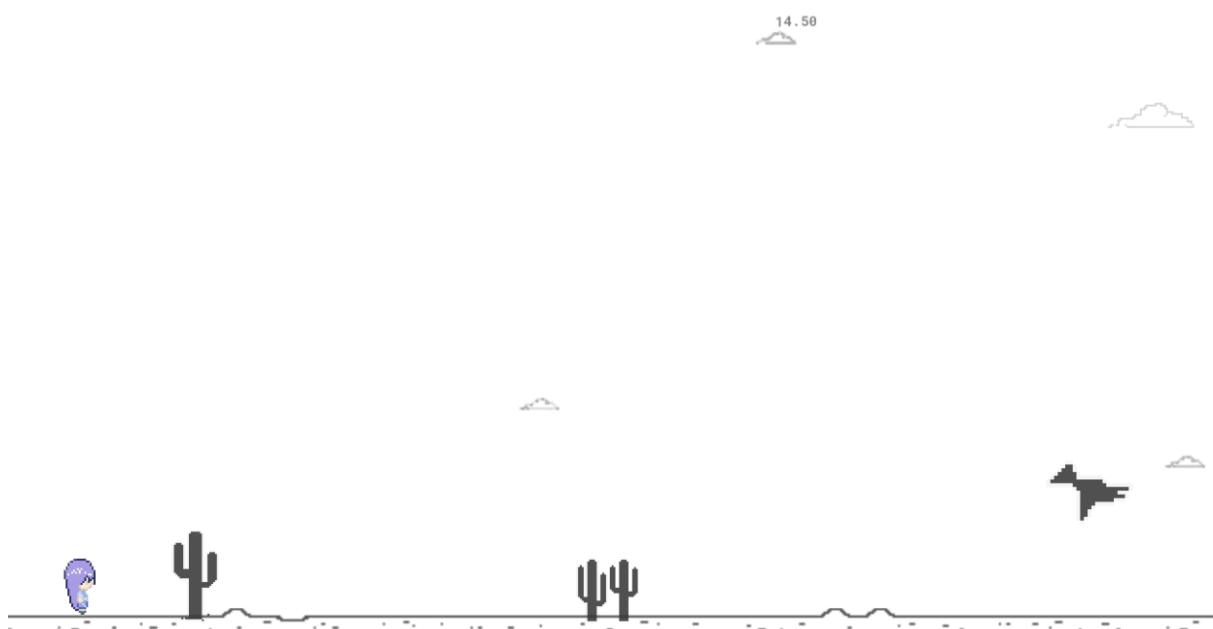
Resultado Final

O jogo final apresenta uma experiência completa, com três capítulos jogáveis, ambientações sombrias, narrativa interativa e múltiplos finais. A quebra da quarta parede foi implementada com sucesso, criando momentos de reflexão e surpresa para o jogador.

- *Figura 5 – Tela inicial do jogo*



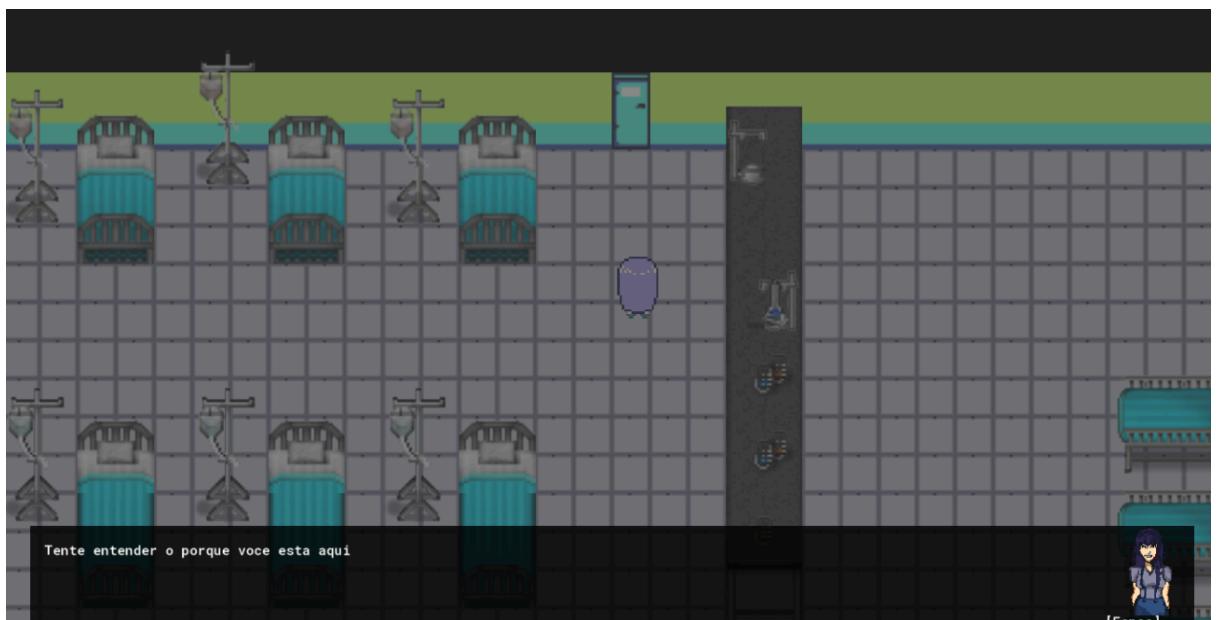
- *Figura 6 – Tela do jogo antes de iniciar história*



- *Figura 7 - Conversação da personagem principal*



- *Figura 7 - Fase do hospital em execução*



- *Figura 8 – Perseguição no hospital*



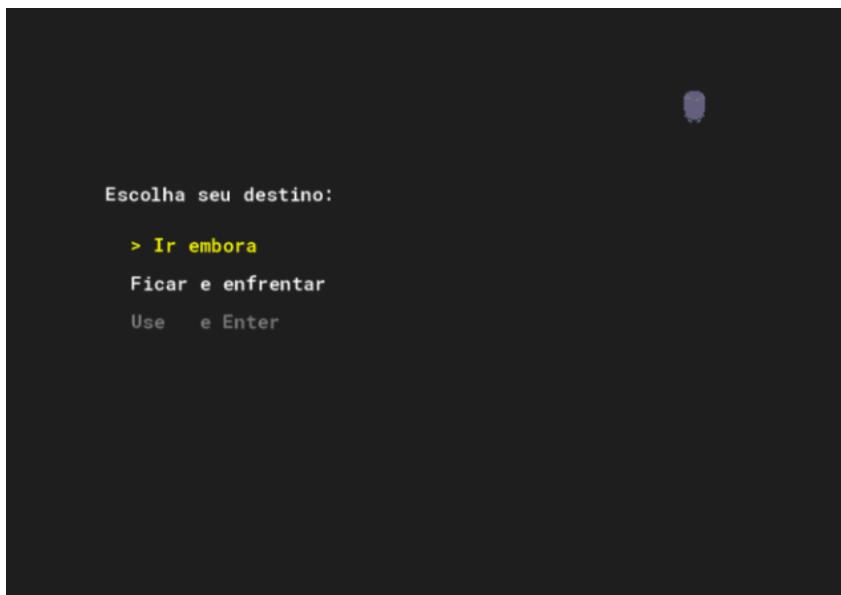
• Figura 9 – Transição hospital para escola



• Figura 10 – Escola



- *Figura 11 – Escolha final do jogador*



CONCLUSÃO

O desenvolvimento do jogo “Entre Salas e Silêncios” representou uma jornada intensa de aprendizado, criatividade e superação de desafios. Desde os primeiros passos, a equipe enfrentou dificuldades técnicas, especialmente na adaptação ao uso da engine GameMaker, que era completamente nova para todos os integrantes. A curva de aprendizado exigiu dedicação e pesquisa constante para compreender a lógica da ferramenta e implementar as mecânicas desejadas.

Outro desafio marcante foi a definição da história do jogo. A narrativa passou por diversas versões até chegar ao enredo final, que equilibra elementos de terror psicológico, quebra da quarta parede e escolhas morais. Essa instabilidade inicial refletiu o processo criativo da equipe, que buscava uma história impactante e coerente com a proposta do projeto.

Ao longo do desenvolvimento, a equipe aprendeu a trabalhar com organização, divisão de tarefas e adaptação às limitações técnicas. Foi possível compreender melhor conceitos de lógica de programação, design de interfaces, estrutura narrativa e integração de elementos visuais e sonoros. Além disso, o projeto permitiu explorar o uso de ferramentas modernas, como inteligência artificial para geração de storyboards e edição de áudio.

Entre os pontos altos do projeto, destacam-se a criação da atmosfera visual e sonora, a implementação das mecânicas de perseguição e a construção de momentos de quebra da quarta parede, que surpreendem o jogador e aprofundam a experiência narrativa.

Se a equipe tivesse mais tempo, seriam exploradas melhorias como a inclusão de dublagem, mais finais alternativos, fases extras e uma versão mobile do jogo. Ainda assim, o resultado final representa com fidelidade a proposta inicial e demonstra o crescimento técnico e criativo dos integrantes ao longo do ano.

REFERÊNCIAS

COPilot. Microsoft Copilot. Disponível em: <https://copilot.microsoft.com/>. . Acesso em: 3 de Maio 2025.

GEMINI. Google Gemini. Disponível em: <https://gemini.google.com/>. . Acesso em:. 21 de Agosto 2025.

MISADE. Jogo independente com narrativa interativa e quebra da quarta parede. Disponível em: <https://aihasto.itch.io/miside> Acesso em: 16 de Julho2025.

YOUTUBE. Como criar um jogo no GameMaker – Tutorial. Disponível em: <https://youtu.be/x2mUXtslgWY?si=7oU3p63mrAyrcpFp>. . Acesso em:14 de Abril 2025.

ITCH.IO. Sprites e cenários. Disponível em: <https://itch.io/games/store>. . Acesso em: 08 outubro 2025.