UNIVERSIDADE DE MOGI DAS CRUZES

THIAGO YUKIO MURAYAMA YASUE

OPERAÇÃO BRAILLE: JOGO SÉRIO ACESSÍVEL PARA USUÁRIOS COM DEFICIÊNCIA VISUAL PARA O AUXÍLIO DO ENSINO DE BRAILE

Mogi das Cruzes, SP

2024

UNIVERSIDADE DE MOGI DAS CRUZES

THIAGO YUKIO MURAYAMA YASUE

OPERAÇÃO BRAILLE: JOGO SÉRIO ACESSÍVEL PARA USUÁRIOS COM DEFICIÊNCIA VISUAL PARA O AUXÍLIO DO ENSINO DE BRAILE

Dissertação apresentada ao Programa de pós-graduação da Universidade de Mogi das Cruzes como parte dos requisitos para obtenção do grau de Mestre em Engenharia Biomédica.
Área de concentração: Informática Biomédica e Engenharia Clínica.

Orientadora: Prof.ª Dra. Silvia Cristina Martini

Mogi das Cruzes, SP

2024

FINANCIAMENTO:



Ficha catalográfica
Universidade de Mogi das Cruzes - Biblioteca Central

Yasue, Thiago Yukio Murayama
Y29o Operação Braille: jogo sério acessível para usuários com deficiência visual para o auxílio do ensino de braile / Thiago Yukio Murayama Yasue. -- Mogi das Cruzes, SP: Universidade de Mogi das Cruzes, 2024.
74 f. : il.

Dissertação (Mestrado em Engenharia Biomédica) -- Universidade de Mogi das Cruzes, 2024.
Orientação: Profa. Dra. Silvia Cristina Martini.
Inclui bibliografia.

1. Educação de pessoas com deficiência visual 2. Educação em Saúde 3. Ensino de Braille 4. Jogos de vídeo I.Martini, Silvia Cristina II.Universidade de Mogi das Cruzes III. Título

CDD 610.28

Elaborada por Maria Cristina Alves Silva Xavier – CRB-8/10077

ATAS

ATA DA SESSÃO PÚBLICA DE APRESENTAÇÃO DE DISSERTAÇÃO DE MESTRADO EM ENGENHARIA BIOMÉDICA DA UNIVERSIDADE DE MOGI DAS CRUZES

Às quatorze horas e trinta minutos do dia trinta e um de outubro de dois mil e vinte e três, pela Universidade de Mogi das Cruzes, realizou-se a defesa de dissertação intitulada "OPERAÇÃO BRAILLE: JOGO SÉRIO ACESSÍVEL PARA USUÁRIOS COM DEFICIÊNCIA VISUAL PARA O AUXÍLIO DO ENSINO DE BRAILE" para obtenção do grau de mestre pelo(a) candidato(a) **Thiago Yukio Murayama Yasue.** Tendo sido o número de créditos alcançados pelo(a) mesmo(a) no total de 48 (quarenta e oito), a saber: 24 unidades de crédito em disciplinas de pós-graduação e 24 unidades de crédito no preparo da dissertação, o(a) aluno(a) perfaz assim os requisitos para obtenção do grau de mestre. A comissão examinadora estava constituída dos senhores professores doutores: Silvia Cristina Martini, e Alessandro Pereira da Silva, da Universidade de Mogi das Cruzes, e Francisco Douglas Lima Abreu, da Faculdade de Informática e Administração Paulista, sob a presidência da primeira, como orientadora da dissertação. A sessão pública da defesa de dissertação foi aberta pela senhora presidenta da comissão que apresentou o(a) candidato(a). Em seguida o(a) candidato(a) realizou uma apresentação oral da dissertação. Ao final daapresentação da dissertação, seguiram-se as arguições pelos membros da comissãoexaminadora. A seguir a comissão, em sessão secreta, conforme julgamento discriminado por cada membro, considerou o(a) candidato(a)

Aprovado (aprovado(a) /reprovado(a)) por Unanimidade (unanimidade/maioria)

Mogi das Cruzes, 31 de outubro de 2023.

Comissão Examinadora	Julgamento
Documento assinado digitalmente gov.br FRANCISCO DOUGLAS LIMA ABREU Data: 31/10/2023 17:01:41-0300 Verifique em https://validar.iti.gov.br	
Prof. Dr. Francisco Douglas Lima Abreu	Aprovado (aprovado(a)/reprovado(a))
Documento assinado digitalmente gov.br ALESSANDRO PEREIRA DA SILVA Data: 02/11/2023 00:35:26-0300 Verifique em https://validar.iti.gov.br	
Prof. Dr. Alessandro Pereira da Silva	Aprovado (aprovado(a)/reprovado(a))
Documento assinado digitalmente gov.br SILVIA CRISTINA MARTINI Data: 31/10/2023 16:56:10-0300 Verifique em https://validar.iti.gov.br	
Profª Drª Silvia Cristina Martini	Aprovado (aprovado(a)/reprovado(a))

Dedico este trabalho a meus pais, minha tia e meus tios que me incentivaram a chegar até aqui.

Agradeço

À Prof.ª Dr.ª Silvia Cristina Martini, minha professora de graduação e orientadora no mestrado.

Aos amigos que me ajudaram com este projeto de alguma forma.

O presente trabalho foi realizado com o apoio da Coordenação de Aperfeiçoamento de Pessoal de Nível Superior - Brasil (CAPES) - Código de Financiamento 88887.644444/2021-00

“Quando tiveres feito uma boa ação, e outra pessoa beneficiar com ela, porquê pedir ainda mais — aplauso para tua bondade, ou um favor em troca — como fazem os insensatos?”

(Marco Aurélio)

RESUMO

O braile é um sistema de escrita utilizado por pessoas cegas ou com baixa visão, sendo este um importante instrumento de alfabetização. Seu aprendizado é sempre associado a instrumentos como reglete e punção ou máquinas de escrever em braile. Com o avanço das novas tecnologias, já é possível desenvolver softwares que sejam capazes de ajudar no ensino do braile de modo a tornar o aprendizado mais acessível, já que o acesso a alguns materiais, como a máquina de escrever em braile, é financeiramente restritivo. Neste contexto, este projeto possui o objetivo de minimizar essas dificuldades através do desenvolvimento de "Operação Braille", um jogo sério com foco no ensino de braile, aplicando mecânicas para tornar o jogo acessível para pessoas cegas e com baixa visão, como cores em alto contraste e narração de telas. Após o desenvolvimento, o jogo foi testado por especialistas da área da informática, validando o projeto conforme sua usabilidade, utilizando um questionário com a escala *Likert*, obtendo uma nota aceitável.

Palavras-chave: deficiência visual; educação em saúde; jogos de computador; tecnologia da informação.

ABSTRACT

Braille is a writing system used by blind or visually impaired individuals, making it an important literacy tool. Its learning process is typically associated with tools like a slate and stylus or Braille typewriters. With the advancement of new technologies, it is now possible to develop software that can assist in Braille education, making learning more accessible, especially since access to certain materials, such as Braille typewriters, can be financially restrictive. In this context, this project aims to alleviate these challenges through the development of "Operação Braille", a serious game focused on Braille education, incorporating features to make the game accessible to blind and visually impaired individuals, such as high-contrast colors and screen narration. After development, the game was tested by experts in the field of computer Science, validating the project's usability, using a Likert scale questionnaire, getting a favorable score.

Keywords: visual impairment; health education; computer games; information technology.

LISTA DE FIGURAS

Figura 1 – Representação de uma célula braile 17
Figura 2 – EAP do projeto 25
Figura 3 – Escala SUS 29
Figura 4 – Diagrama de Caso de Uso: Menu 40
Figura 5 – Diagrama de Caso de Uso: In-Game 40
Figura 6 – Diagrama de Redes: Definição de Escopo 47
Figura 7 – Diagrama de Redes: Desenvolvimento do Game 47
Figura 8 – Diagrama de Redes: Avaliação do Game 48
Figura 9 – Diagrama de Redes: Conclusão do Game 48
Figura 10 – Emblemas 51
Figura 11 – Jogador observa a tela 53
Figura 12 – Jogador utiliza o teclado 53
Figura 13 – Jogador confirma a célula e outra letra aparece para ser digitada .. 53
Figura 14 – *Game Loop* 54
Figura 15 – Diagrama de Classes 54
Figura 16 – Método *PreparaScriptablePalavra()* 55
Figura 17 – Método PreparaGetLetra() 55
Figura 18 – Método *GetLetra()* 56
Figura 19 – Função InterpretaCharPorAlfabeto() 56
Figura 20 – Método SetLetra() 57
Figura 21 – Sistema de Acreção de Pontos 58
Figura 22 – Fluxograma do Armazenamento de Pontos 59
Figura 23 – Fluxograma de Verificação de Pontos 60
Figura 24 – Telas do Jogo 60
Figura 25 – Tela de Pré-Menu 61
Figura 26 – Tela de Menu Principal 61
Figura 27 – Tela de Seleção de Fases 62
Figura 28– Tela de Pontuações 62
Figura 29 – Tela de Créditos 62
Figura 30 – Tela de *Gameplay* 63
Figura 31 – Funcionamento da Progressão de Clipes de Áudio 64
Figura 32 – Narração de Interface 64
Figura 33 – Telas do aplicativo móvel 65
Figura 34 – Resultados do teste de usabilidade 67

LISTA DE QUADROS

Quadro 1 – Alfabeto em braile 17
Quadro 2 – Lista Geral de Símbolos (Ordem Braile) 17
Quadro 3 – Lista de softwares analisados 20
Quadro 4 – Requisitos Funcionais: Iniciar Jogo 31
Quadro 5 – Requisitos Funcionais: Ver Pontuações 31
Quadro 6 – Requisitos Funcionais: Ver Créditos 31
Quadro 7 – Requisitos Funcionais: Voltar ao Menu 32
Quadro 8 – Requisitos Funcionais: Sair do Jogo 32
Quadro 9 – Requisitos Funcionais: Digitar 32
Quadro 10 – Requisitos Funcionais: Confirmar Célula 33
Quadro 11 – Requisitos Funcionais: Verificar Célula 33
Quadro 12 – Requisitos Funcionais: Pontuações 33
Quadro 13 – Requisitos Funcionais: Salvar Progresso 34
Quadro 14 – Requisitos Funcionais: Sair do Nível 34
Quadro 15 – Requisitos Funcionais: Acessar Pré-Menu 34
Quadro 16 – Requisitos Funcionais: Acessar Como jogar 35
Quadro 17 – Requisitos Funcionais: Acessar Vogais 35
Quadro 18 – Requisitos Funcionais: Acessar Fases 35
Quadro 19 – Requisitos Não Funcionais: Engine 36
Quadro 20 – Requisitos Não Funcionais: Especificações 36
Quadro 21 – Requisitos Não Funcionais: Salvar Jogo 36
Quadro 22 – Requisitos Não Funcionais: Configurações de Som 36
Quadro 23 – Requisitos Não Funcionais: Configurações de Interface 37
Quadro 24 – Requisitos Não Funcionais: Verificar Arquivos Salvos 37
Quadro 25 – Regras de Negócio: RN001 38
Quadro 26 – Regras de Negócio: RN002 38
Quadro 27 – Regras de Negócio: RN003 39
Quadro 28 – Regras de Negócio: RN005 39
Quadro 29 – Especificação de Caso de Uso: Menu Principal 40
Quadro 30 – Especificação de Caso de Uso: In-Game 41
Quadro 31 – Tarefas de Definição de Escopo – Dependências 42
Quadro 32 – Tarefas de Desenvolvimento: Roteiro – Dependências 42
Quadro 33 – Tarefas de Desenvolvimento: Programação – Dependências 43
Quadro 34 – Tarefas de Desenvolvimento: Arte – Dependências 43
Quadro 35 – Tarefas de Desenvolvimento: Áudio – Dependências 44
Quadro 36 – Tarefas de Avaliação do Game – Dependências 44
Quadro 37 – Tarefas de Conclusão – Dependências 44
Quadro 38 – Tarefas de Definição de Escopo – Duração 44
Quadro 39 – Tarefas de Desenvolvimento: Roteiro – Duração 44
Quadro 40 – Tarefas de Desenvolvimento: Programação – Duração 45

Quadro 41 – Tarefas de Desenvolvimento: Arte – Duração 46
Quadro 42 – Tarefas de Desenvolvimento: Áudio – Duração 46
Quadro 43 – Avaliação do Game – Duração..... 46
Quadro 44 – Conclusão do Game – Duração 46
Quadro 45 – Risco 01..... 49
Quadro 46 – Risco 02..... 49
Quadro 47 – Risco 03..... 50
Quadro 48 – Conversão binário-caractere da célula de braile..... 52

LISTA DE TABELAS

Tabela 1 – Caminho crítico: Definição de Escopo 48
Tabela 2 – Caminho crítico: Desenvolvimento do Game..... 48
Tabela 3 – Caminho crítico: Avaliação do Game..... 48
Tabela 4 – Caminho crítico: Conclusão do Game..... 49

SUMÁRIO

1. INTRODUÇÃO 16
1.1 CONCEITOS INICIAIS 16
1.2 JUSTIFICATIVA 18
1.3 ESTADO DA ARTE 19
1.3.1 Software de computador para ensino de braile 21
1.3.2 Aplicativo mobile para o ensino de braile 21
1.3.3 Jogo digital acessível para deficientes visuais 23
2. MATERIAL E MÉTODOS 25
2.1 DEFINIÇÃO DE ESCOPO 25
2.1.1 Levantamento de requisitos 25
2.1.2 Requisitos funcionais, não funcionais e regras de negócio 26
2.1.3 Divisão de Tarefas 26
2.1.4 Análise de Riscos 26
2.1.5 Elaboração do Cronograma 27
2.2 DEFINIÇÃO DO GAME 27
2.2.1 Roteiro 27
2.2.2 Design 28
2.2.3 Áudio 28
2.2.4 Ambiente de Desenvolvimento 28
2.3 AVALIAÇÃO DO GAME 29
2.3.1 Testes do game 29
2.3.2 Refinamento de Mecânicas 30
2.4 CONCLUSÃO DO GAME 30
2.4.1 Discussão de Resultados 30
2.4.2 Entrega do Projeto 30
3. RESULTADOS 31
3.1 DOCUMENTAÇÃO DE CONCEITO DO JOGO 31
3.2 ROTEIRO 50
3.3 DESENVOLVIMENTO DO JOGO 51
3.3.1 Sistema de Digitação 51
3.3.2 Game Loop 54
3.3.3 Gameplay e pontuação 57
3.3.4 Persistência de Dados 58
3.3.5 Fluxo de Telas 60

3.4 DESIGN **61**
3.5 ÁUDIO **63**
3.6 DISPONIBILIDADE **65**
3.7 APLICATIVO MÓVEL: PROVA DE CONCEITO **65**
3.8 TESTE DE USABILIDADE **66**
4 DISCUSSÃO **68**
5 CONCLUSÃO **69**
5.1 LIMITAÇÕES **69**
REFERÊNCIAS **70**
APÊNDICE A – Questionário de Usabilidade **73**

1. INTRODUÇÃO

1.1 CONCEITOS INICIAIS

A acessibilidade é um tema recorrente e atual dentro do contexto dos jogos digitais, pois com quase 400 milhões de jogadores com deficiências no mundo, é importante criar experiências de jogos que sejam inclusivas e acessíveis por design para o máximo possível de jogadores (Microsoft, c2024).

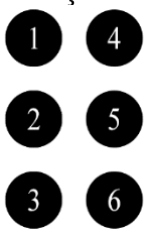
O estatuto da pessoa com deficiência considera como acessibilidade, "possibilidade e condição de alcance para utilização, com segurança e autonomia, de espaços, mobiliários, equipamentos urbanos, edificações, transportes, informação e comunicação, inclusive seus sistemas e tecnologias, bem como de outros serviços e instalações abertos ao público, de uso público ou privados de uso coletivo, tanto na zona urbana como na rural, por pessoa com deficiência ou com mobilidade reduzida" (Brasil, 2015). No contexto dos jogos, essa aplicação se dá através de recursos que facilitem o uso do programa por uma pessoa com deficiência, como dicas visuais (legendas, alto contraste e filtros para daltonismo), recursos de áudio (narração e indicações sonoras), recursos de entrada (remapeamento de teclas e suporte a outros tipos de controle), entre outros. Neste projeto, o foco principal será tratar da acessibilidade voltada para pessoas com deficiência visual.

A deficiência visual é classificada através da CID-11 (Classificação Internacional de Doenças) em seis níveis baseados na acuidade visual, isto é, a clareza da visão do indivíduo, onde se denota visão subnormal quando ocorre perda de visão parcial ou 30% de visão no melhor olho, e cegueira quando há a perda quase total ou total da visão (WHO, 2023). No Brasil, segundo dados do censo do Instituto Brasileiro de Geografia e Estatísticas (IBGE) realizado em 2010, 18.8% da população, ou cerca de 35 milhões de pessoas, possuem alguma deficiência visual, dos quais, cerca de 29 milhões possuem perda de visão parcial, 6 milhões possuem perda quase total de visão e 506.377 possuem perda total da visão.

Para tanto, o braile é utilizado por estas pessoas para o processo de leitura e escrita. Sendo criado no início do Século XIX por Louis Braille, o sistema é constituído por 63 sinais, formados a partir do conjunto matricial com duas colunas e três linhas, sendo este

conjunto de seis pontos, o sinal fundamental (Brasil, 2018). A figura 1 demonstra uma célula com a enumeração de cada ponto:

Figura 1 – Representação de uma célula braile



No Brasil, os sinais que se empregam na escrita corrente de textos possuem a significação do quadro 1 conforme documento normatizador do braile no Brasil, Grafia Braille para a Língua Portuguesa (Brasil, 2018).

Quadro 1 – Alfabeto em braile

A	B	C	D	E	F	G	H	I	J
⠁	⠃	⠉	⠙	⠑	⠋	⠛	⠓	⠊	⠚
K	L	M	N	O	P	Q	R	S	T
⠅	⠇	⠍	⠝	⠕	⠏	⠟	⠗	⠎	⠞
U	V	W	X	Y	Z				
⠥	⠧	⠺	⠭	⠽	⠵				

Hildebrandt *et al.* (2005) elaboram um documento com o objetivo da criação de uma grafia uniforme do braile e suas aplicações dentro do ambiente virtual, tais como o sistema do braile de seis pontos e a padronização do sistema braile de oito pontos. O quadro 2 retrata uma lista geral de símbolos utilizados no braile de seis pontos, onde há uma coluna de “descrição”, que apresenta a posição de cada ponto da célula.

Quadro 2 – Lista Geral de Símbolos (Ordem Braile)

Símbolo Braille	**Descrição**	**Símbolo em Tinta**	**Denominação**
⠁	(1)	a	a minúsculo
⠃	(12)	b	b minúsculo
⠉	(14)	c	c minúsculo
⠙	(145)	d	d minúsculo
⠑	(15)	e	e minúsculo

⠹	(124)	f	f minúsculo
⠻	(1245)	g	g minúsculo
⠱	(125)	h	h minúsculo
⠊	(24)	i	i minúsculo
⠚	(245)	j	j minúsculo
⠅	(13)	k	k minúsculo
⠇	(123)	I	l minúsculo
⠍	(134)	m	m minúsculo
⠝	(1345)	n	n minúsculo
⠕	(135)	o	o minúsculo
⠏	(1234)	p	p minúsculo
⠟	(12345)	q	g minúsculo
⠗	(1235)	r	r minúsculo
⠎	(234)	s	s minúsculo
⠞	(2345)	t	t minúsculo
⠥	(136)	u	u minúsculo
⠧	(1236)	v	v minúsculo
⠭	(1346)	x	x minúsculo
⠽	(13456)	y	y minúsculo
⠵	(1356)	z	z minúsculo
⠯	(12346)	ç	c com cedilha minúsculo
⠿	(123456)	é	e agudo minúsculo

Fonte: Hildebrandt *et al.* (2005)

1.2 JUSTIFICATIVA

Ainda que seja um instrumento importante, tanto na alfabetização, quanto no uso geral do dia a dia, o braile ainda é pouco utilizado. Segundo o Censo de 2010, das 35 milhões de pessoas com algum tipo de deficiência visual, 17% (cerca de 6 milhões) não estavam alfabetizadas. Apesar dos avanços na tecnologia proverem leitores de tela e livros digitais falados, tais recursos não substituem o braile por se tratar da única maneira de alfabetização da criança cega, sendo esta uma importante ferramenta de inclusão. Segundo Mosquera (2012), os portadores de deficiência não vivem num mundo à parte, mas interagem com o meio e precisam se sentir integrados à sociedade, sendo a alfabetização a melhor ferramenta de inclusão.

Existem hoje alguns instrumentos utilizados para o aprendizado do braile, dos quais cabe citar, por exemplo, a reglete, uma ferramenta básica, que consiste em uma prancha, utilizada para afundar uma punção em uma folha de papel, escrevendo assim as letras. Além desta, vale citar também a máquina de escrever braile, um dispositivo que se

assemelha a uma máquina de datilografar, na qual, a partir de seis botões, realiza-se a escrita no papel, em relevo. Máquinas mais avançadas como a *SMART Brailler* (Perkins, c2024) já possuem funções mais tecnológicas, com recursos voltados para a área educacional, por exemplo, um feedback de áudio quando uma letra ou palavra é inserida e um display digital para pessoas com baixa visão. Por fim, o *Focus Braille Display* (Freedom Scientific, c2024) é outra tecnologia assistiva utilizada para a escrita do braile, porém mais portável e compatível com dispositivos móveis através de *bluetooth*.

Todavia, a máquina de escrever braile é um dispositivo que possui um custo bem elevado, com modelos recentes custando mais de 12 mil reais enquanto a *Focus Braille Display* possui modelos a partir de 7 mil reais. Com a informatização, usuários de braile acabaram recorrendo aos vários softwares de computadores que simulam a digitação de cada um dos seis botões da máquina nas teclas do teclado, utilizando o padrão "s-d-f-j-k-l", tendo em vista que as teclas "f" e "j" possuem, como padrão, linhas de relevo, servindo como referência ao apoiar os dedos em um teclado. Ademais, existem impressoras capazes de converter arquivos de textos para braile, utilizando tanto editores de texto comuns quanto softwares específicos com simulador de teclado braile, como o Braille Fácil (INTERVOX, c2022).

1.3 ESTADO DA ARTE

Atualmente existem alguns softwares, tanto para computadores quanto para mobile, que, de alguma forma, possuem relação com o tema deste projeto. Abaixo, estes estão descritos, separados em três tipos: "software de computador para ensino de braile", "aplicativo mobile para o ensino de braile" e "jogo digital acessível para deficientes visuais". Foi feita uma pesquisa em sites como *ResearchGate* e Google Acadêmico utilizando as palavras-chave "*braille", "software", "game", "education"*. Também foi feita uma pesquisa nas lojas de aplicativos *Google Play* e *App Store*, utilizando apenas o termo "*braille".* Por fim, foi feita uma pesquisa por jogos de entretenimento em lojas digitais de jogos, como a *Steam*, com o objetivo de descobrir jogos que possuem público-alvo em deficientes visuais ou que fossem acessíveis através da sua jogabilidade. O quadro abaixo sintetiza os aplicativos analisados, classificando-os quanto a sua plataforma, tipo e tipos de acessibilidades.

Quadro 3 – Lista de softwares analisados

Autor	Plataforma	Tipo	Acessível para deficientes visuais?
Santana *et al.* (2019)	Windows (PC)	Software de computador para ensino de braile	Sim (*Input* que simula a máquina de escrever braile)
Thichuea e Wongjai (2019)	Windows (PC)	Software de computador para ensino de braile	Sim (*Input* que simula a máquina de escrever braile)
Ugedo (2020)	Android	Aplicativo mobile para o ensino de braile	Não
Santander (2020)	Android	Aplicativo mobile para o ensino de braile	Não
Techopolis Online Solutions, LLC (2018)	iOS	Aplicativo mobile para o ensino de braile	Não
R&A (2017)	iOS	Aplicativo mobile para o ensino de braile	Não
Gertz (2023)	iOS	Aplicativo mobile para o ensino de braile	Não
Dong Digital (2021)	Android, iOS	Aplicativo mobile para o ensino de braile	Sim (dicas táteis, dicas sonoras)
Kawakami (2014)	iOS	Aplicativo mobile para o ensino de braile	Sim (dicas sonoras)
Needham (2020)	Android	Aplicativo mobile para o ensino de braile	Sim (dicas táteis)
Milne *et al.* (2013)	iOS	Aplicativo mobile para o ensino de braile	Sim (dicas táteis, dicas sonoras, navegação por gestos)
Hatzigiannakoglou; Kampouraki (2016)	Android	Aplicativo mobile para o ensino de braile	Não
Kleck (2014)	iOS (somente *iPad)*	Aplicativo mobile para o ensino de braile	Sim (dicas sonoras, *input* que simula a máquina de escrever braile)
Sonnar Interactive Ltd (2016)	Android, iOS	Jogo digital acessível para deficientes visuais	Sim (dicas sonoras, navegação por gestos)

Dowino (2015)	Android, iOS, PC	Jogo digital acessível para deficientes visuais	Sim (dicas sonoras)
Team Zeroth (2023)	PC	Jogo digital acessível para deficientes visuais	Sim (dicas sonoras)

1.3.1 Software de computador para ensino de braile

Santana *et al.* (2019) desenvolveram um software simulando uma máquina de *Braille*, apelidada de *Braillearning*. Com a justificativa de que a máquina de *Braille* possui um preço elevado, o software foi produzido e testado por um grupo de pessoas com deficiência visual, sendo validado pela maioria dos usuários. Porém, é denotado que uma das limitações da execução da pesquisa se deu à análise do uso do software em relação ao ensino de braile, de modo que a maioria dos pesquisados já conheciam este sistema, então utilizaram a ferramenta apenas para simulação e não para o aprendizado em si. Além deste, o *Brailler Fighters* (Pooltrub, 2019), desenvolvido na Universidade de Tecnologia de *Rajamangala Lanna*, também é um software acessível para o ensino de braile, com a diferença de ser focado para o idioma tailandês, que utiliza um abugida ao invés de um alfabeto.

1.3.2 Aplicativo *mobile* para o ensino de braile

Existem vários aplicativos para o ensino de braile encontrados na *Google Play*, a maioria deles, porém, não é acessível para deficientes visuais, consistindo em: *flashcards* para estudo do braile, como no *Braille App* (Santander, 2020); aplicativo de escrita de braile através de toques na tela sem nenhuma descrição sonora, como o *Brailliac: Braille Tutor* (Needham, 2020) e aplicativo com exercícios de tradução de texto em braile para texto em inglês, como o *LEARN BRAILLE* (Ugedo, 2020) e texto em alemão, como o *Braille Trainer* (Gertz, 2023). O *Brailliac: Braille Tutor*, apesar de ser um aplicativo com a maior diversidade de recursos (dicionário, modo desafio, modo treino), carece de métodos de acessibilidade para deficientes visuais, tais como: dicas sonoras – pois nos modos de desafio e treino, há um texto na tela, que deve ser escrito em braile, mas não há indicadores sonoros, além do método de "ajuda": um botão que revela um *flashcard* de como a célula deve ser preenchida; vibração para o toque na célula de braile – que, apesar de existir vibração no app, é ativada apenas ao usuário errar uma célula. Por fim, o *Braille*

Academy: Play & Learn (Dong Digital, 2023), mesmo que possua som e vibração dos botões de teclado, possui um aviso na loja digital dizendo que o aplicativo não foi projetado especialmente para pessoas com deficiência visual.

Dentro da loja de aplicativos para iOS, a *App Store* possui substancialmente mais aplicativos com a proposta do ensino de braile, contudo, possuem as mesmas carências analisadas nos aplicativos da *Google Play* no que tange o aspecto da acessibilidade para deficientes visuais. Porém, mesmo com tantos aplicativos no iOS, é importante avaliar que o preço de um *iPhone*, na realidade brasileira, é bem restritivo. A loja oficial da *Apple* oferece atualmente como opção mais barata somente o *iPhone SE* de 3ª geração, a partir de R$ 4.299,00 (Apple, c2024), sendo que os aplicativos da *Google Play* analisados, além de serem todos gratuitos, podem ser executados até mesmo em um computador com o auxílio de um emulador *Android* gratuito.

Na área de jogos digitais, existem alguns projetos desenvolvidos exclusivamente para o ambiente mobile, dentre os quais pode-se citar: o *BraillePlay* (Milne *et al.*, 2013), um conjunto de jogos baseados na entrada em braile através do *touch* do celular, que simula uma célula de braile, como o *VBGhost*, uma versão digital do jogo de palavras "*ghost*", o *VBHangman*, uma versão do jogo "forca", o *VBWriter*, onde o usuário deve digitar a letra através de uma dica de áudio e o *VBReader*, onde através da vibração do celular, o jogador consegue sentir a célula de braile e tentar descobrir qual é aquela letra; o *Learn Braille* (Hatzigiannakoglou; Kampouraki, 2016) também é um conjunto de jogos, dentre eles um jogo de forca e um jogo de escrita, focado no ensino de braile no idioma grego, no entanto, não apresenta nenhum recurso de acessibilidade. Por fim, o aplicativo da *App Store*, *Exploring Braille with Madilyn and Ruff* (Kleck, 2014), é o único software analisado nesta seção, encontrado em lojas de aplicativo, que possui o ensino da escrita de braile aliado a sistemas de acessibilidade, aqui, pode ser destacado o *input* feito pelo jogador como se ele estivesse em uma máquina de escrever braile, além do uso de dicas sonoras. No entanto, é válido destacar mais uma vez que, principalmente no Brasil, o preço do *hardware* da *Apple* é bem elevado, ainda mais considerando que este último jogo só pode ser jogado em um dispositivo *iPad*, com preço que pode variar entre R$3.899,00 (*iPad* mais barato, de 9ª geração, sem nenhum periférico, sem *AppleCare Protection Plan*, com as piores especificações de hardware) a R$28.499,00 (*iPad Pro* mais recente sem nenhum periférico, com *AppleCare Protection Plan*, com as melhores especificações de hardware) de acordo com o site oficial da empresa (Apple, c2024).

1.3.3 Jogo digital acessível para deficientes visuais

A respeito de jogos voltados para deficientes visuais, existem diversos aplicativos com recursos de acessibilidade. O *Audio Game Hub 2* (Sonnar Interactive Ltd, 2016) é um aplicativo mobile produzido em uma parceria entre a *Royal New Zealand Foundation of the Blind*, *Auckland University of Technology* e *Leuphana University of Lüneburg*. É composto de oito jogos baseados em áudio, com entradas do usuário sendo executadas através de toques, duplo-toques e arrastar com o dedo na tela do celular. O jogo possui narração de menus, tutorial de navegação e interface visual em alto contraste, para usuários sem deficiência visual ou com baixa visão.

Sobre os jogos para computador, ambos os jogos se qualificam na modalidade de áudio games, ou seja, jogos que possuem apenas recursos sonoros para a sua *gameplay*. *A Blind Legend* (Dowino, 2016), é um jogo de ação-aventura em primeira pessoa, onde o jogador controla um cavaleiro cego que teve sua mulher raptada. Junto com a sua filha, o cavaleiro explora diversos cenários, como cavernas, cachoeiras e florestas. Para tanto, o jogo possui um recurso de áudio binaural, para o jogador conseguir se localizar no ambiente. Além disso, o jogador possui uma espada e um escudo, sendo capaz de realizar combos de golpes em inimigos, onde é emitido um som em alguma direção do jogador e cabe a ele acertá-los na posição correta. Breu: Ataque das Sombras (Team Zeroth, 2023), o segundo jogo da série Breu, é um jogo brasileiro de suspense onde o jogador controla um jovem cego que está à procura de seu avô e deve explorar uma mata enquanto é perseguido por criaturas perigosas.

Como visto, existem algumas ferramentas de software com o objetivo específico de simular a escrita de braile e jogos que possuem recursos inclusivos para deficientes visuais. No entanto, os aplicativos para dispositivos móveis analisados que possuem o objetivo do ensino de braile não possuem como público-alvo pessoas cegas e os poucos que possuem, são pouco acessíveis. Além disso, pouco se fala ainda a respeito do ensino do sistema de escrita dentro do universo computadorizado principalmente no que tange o universo dos jogos digitais.

Ademais, torna-se evidente que a dificuldade de acesso ao aprendizado de braile é um problema social que permeia a sociedade mesmo nos dias de hoje, com o auxílio de novas tecnologias, com isso, este projeto possui o objetivo de minimizar essas dificuldades através do desenvolvimento de um jogo feito para computadores,

observando as mecânicas analisadas dos softwares de computador que já existem, ainda diminuindo os custos com dispositivos de escrita.

Sendo assim, o objetivo deste trabalho é o desenvolvimento de um jogo sério focado no auxílio do ensino da escrita em braile, simulando os *inputs* do jogador como se ele estivesse utilizando um dispositivo como a máquina de braile.

Além disso, foram determinados alguns objetivos específicos dentro do desenvolvimento do jogo:

1. Programar um sistema de digitação em braile, baseado em entradas do teclado.
2. Criar os elementos visuais do jogo em alto contraste, para jogadores com baixa visão.
3. Incluir um recurso de narração dos menus de navegação e *gameplay.*
4. Realizar a validação do jogo sério por cinco especialistas:
    - 4 especialistas na área de informática
    - 1 especialistas na área de informática e deficiente visual

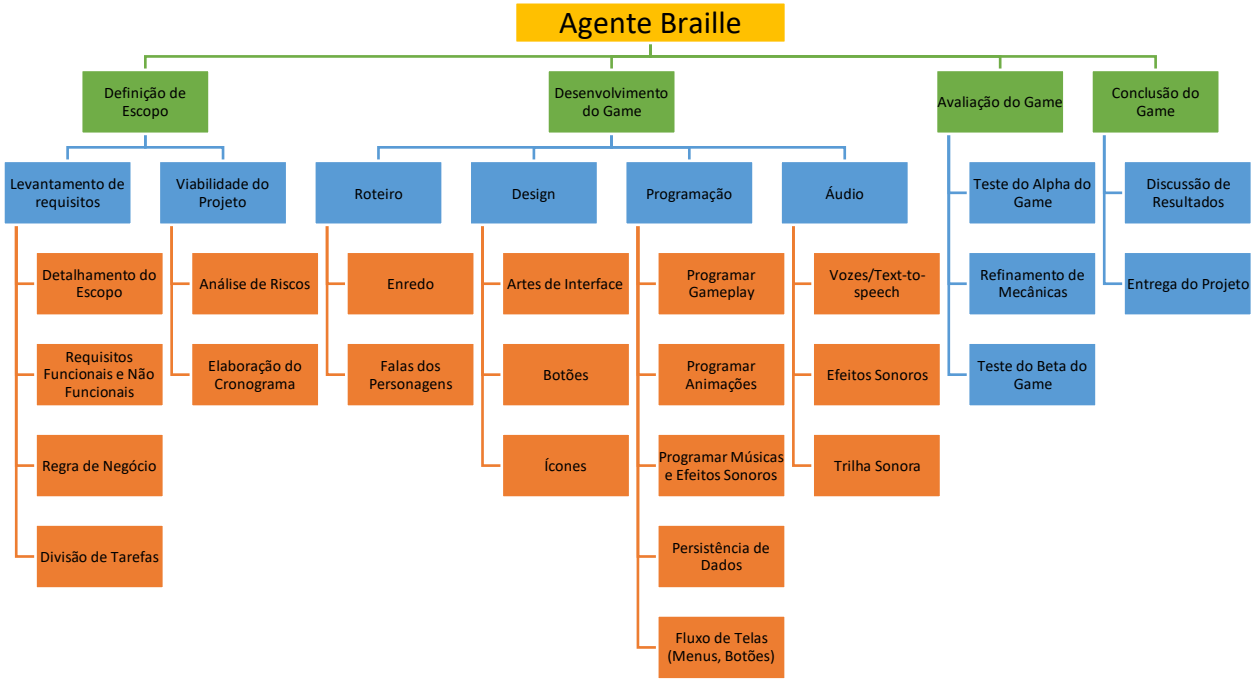
E posteriormente, após o desenvolvimento do jogo, um novo objetivo foi adicionado, a respeito da pesquisa relacionada à justificativa deste projeto:

- Desenvolver uma prova de conceito do jogo para responder à pergunta: “O quão difícil é fazer um aplicativo mobile acessível para o ensino de braile?”

2. MATERIAL E MÉTODOS

A princípio, foi-se elaborada uma EAP (Estrutura Analítica do Projeto) a fim de organizar o escopo do projeto de uma forma visual e hierárquica, dividindo-a em vários processos. É possível perceber a partir deste diagrama (Figura 2), que existem quatro partes principais: Definição de Escopo, Desenvolvimento do Game, Avaliação do Game e Conclusão do Game. Para tanto, foi utilizada uma metodologia baseada no modelo iterativo de *Royce* (Royce, 1987) — consistindo em um método em cascata com iterações, para o desenvolvimento.

Figura 2 – EAP do projeto



2.1 DEFINIÇÃO DE ESCOPO

2.1.1 Levantamento de requisitos

O jogo foi desenvolvido na *Unity*, um ambiente de desenvolvimento integrado (IDE), isto é, um software com diversas ferramentas integradas, este, com foco para a criação de jogos. A linguagem de programação a ser utilizada é o C#, utilizando o *Microsoft Visual Studio Community* como editor de código-fonte. Para a criação da arte,

foi utilizado o software *Inkscape* para a elaboração de botões, ícones e ilustrações, já que, vale ressaltar, o jogo poderá ser utilizado por pessoas com baixa visão ou mesmo pessoas videntes. Os efeitos sonoros foram baixados através do site *OpenGameArt*. Após isso, foi feito um tratamento no áudio utilizando o software *Audacity*.

A respeito dos requisitso técnicos do jogo, foram definidos requisitos mínimos, sendo esses, sistema operacional Windows 7, 8, 8.1 ou 10 com 64-32 bits, processador Intel Core2 Duo, memória 4GB de RAM, memória de vídeo de 512MB, DirectX 9.0 ou superior e armazenamento de 2GB. Já que o jogo não possui elementos tridimensionais, o jogo não requer um processador e placa de vídeos de topo de linha.

2.1.2 Requisitos funcionais, não funcionais e regras de negócio

Segundo Chung *et al.* (2000), visando apurar todas as funções que o sistema, neste caso, o jogo, deverá executar, foi necessário realizar uma análise com objetivo de demonstrar como as mecânicas funcionam. Os quadros que apresentam os requisitos funcionais do jogo foram feitos a partir de Sommerville (2017).

Além dos requisitos funcionais, ainda foram apontados os requisitos não-funcionais e as regras de negócio. Dentro do contexto de um jogo, as regras de negócio também incluem a própria mecânica de funcionamento das fases.

Para auxiliar na visualização do funcionamento dos requisitos funcionais, foram desenvolvidos diagramas de caso de uso (Booch *et al.*, 2005).

2.1.3 Divisão de Tarefas

Foi feita a divisão de cada fase de elaboração do jogo, sendo desenvolvida uma EAP (estrutura analítica de projeto), com o objetivo de contribuir na organização do projeto. Ademais, foram produzidos quadros que mostram as dependências de cada fase de produção.

2.1.4 Análise de Riscos

Foram estimados os riscos provenientes da produção jogo, portanto, foram criados quadros de estudo de caminho crítico, a partir da produção de diagramas de rede elaborados com base na etapa de divisão de tarefas.

Conforme a estruturação dos diagramas de rede, foi analisado o caminho crítico de cada etapa criando-se quadros para sintetizá-los. Além do tempo de cada atividade e suas dependências, foi considerado também se os funcionários responsáveis estarão disponíveis ou se estão realizando outras tarefas. Caso não estejam disponíveis, aquela tarefa ficará em espera.

Por fim, nesta etapa, foram analisados alguns possíveis riscos para o desenvolvimento do jogo e com base neles, foram estipulados seus impactos, prioridade e nosso plano de ação, caso este risco acabe ocorrendo.

2.1.5 Elaboração do Cronograma

Foi criado um cronograma com a duração de todas as tarefas, tanto as elaboradas em base no EAP quanto as atividades provenientes ao mestrado. A partir do cronograma, foi elaborado um estudo de caminho crítico, estimando o tempo de produção de cada fase de desenvolvimento.

2.2 DEFINIÇÃO DO GAME

2.2.1 Roteiro

O roteiro foi elaborado no aplicativo *Notion*, apresentando um *script* do que cada cena deve possuir, além do objetivo de estudo (por exemplo, uma fase focada na escrita de vogais, outra fase focada nas primeiras dez letras etc.).

A documentação e o *Game Design Document* foram desenvolvidas no aplicativo *Microsoft Word*, sendo que algumas figuras da documentação foram feitas no *Microsoft Power Point*, tais como diagramas de rede e o modelo de estrutura analítica de projeto.

Ademais, as fases introdutórias, que serão apresentadas logo mais nos resultados, foram feitas com base na cartilha “O Sistema Braille” (DREZZA, 2022), sendo extraídas as palavras utilizadas na seção “Orientações para uso da cartilha”.

2.2.2 Design

A arte do jogo foi feita toda em 2D, isto é, não há partes tridimensionais no jogo (modelagem 3d, texturas 3d, animações 3d). Por este motivo, foram utilizados softwares de arte 2d, como o *Inkscape*, para a interface, artes vetoriais e o *Aseprite* para as artes em *pixel*. Estas artes dentro de jogo foram feitas em alto contraste, isso significa que as cores utilizadas são o preto e o branco, para criar o máximo de contraste, destacando os *assets* no cenário, para a fácil utilização do jogo por usuários com problemas de baixa visão.

2.2.3 Áudio

Foram utilizados efeitos sonoros provenientes do site *OpenGameArt* para toda a composição da *user interface*, como o som do *click* do *mouse*, som de acerto e som de erro.

Inicialmente, o jogo iria possuir um sistema nativo de *text-to-speech*, isto é, um sistema capaz de converter *strings* de texto em áudio. No entanto, por não haver *APIs* de *tts* para a *engine* em questão, todas as falas necessárias foram produzidas através de um *site* conectado com a *Amazon Polly TTS*, um serviço em nuvem da *Amazon* de *text-to-speech*. Assim, foram inseridos os textos e estes foram exportados em arquivos de mídia de áudio. Esses áudios foram cortados e tratados utilizando o software *Audacity*, corrigindo quaisquer palavras que o *TTS* possa ter falado errado ou em uma entonação errada (exemplo: ao falar para o jogador pressionar a tecla “Q”, pode ser que o *TTS* não entenda o prompt “Q” e pule esta palavra, ou diga em uma entonação errada, como “qué”). Por fim, os áudios foram exportados em arquivos de áudio no formato *Waveform*.

2.2.4 Ambiente de Desenvolvimento

A programação foi feita utilizando o *Microsoft Visual Studio Community 2019*. Os inputs de célula de braile foram baseados nos softwares apresentados no estado da arte (sistema sdf-jkl).

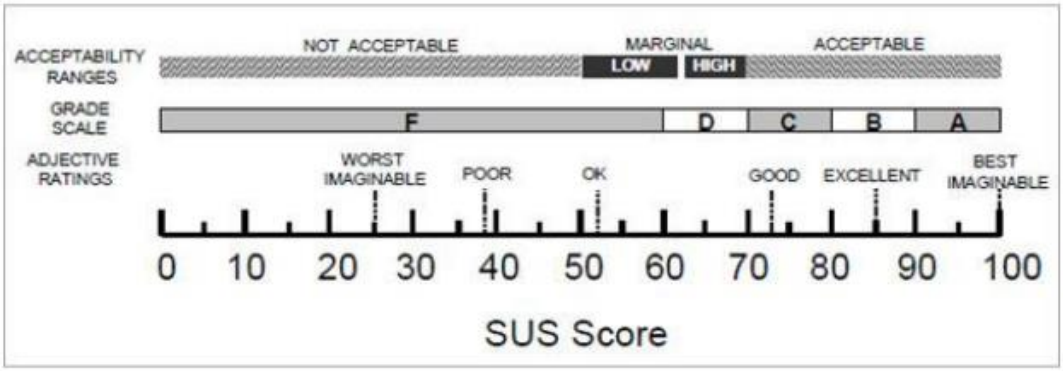
2.3 AVALIAÇÃO DO GAME

2.3.1 Testes do game

Após o desenvolvimento, serão feitos inicialmente os testes de caixa-branca, no que tange identificar problemas com a navegação e testes de caixa-preta na *gameplay*. Posteriormente, esta versão será testada por três especialistas da área da informática, que analisarão o nível de usabilidade do jogo e logo depois, responderão anonimamente um questionário do tipo *System Usability Scale* (Brooke, 2013), que consiste em dez perguntas com uma escala que varia entre 1 e 5, onde 1 significa "Discordo Completamente" e 5 significa "Concordo Completamente" (Ver questionário no Apêndice A). Esse questionário avalia a usabilidade de um sistema em três critérios: efetividade, eficiência e satisfação do usuário, seguindo a ISO 9241-11 – Orientações sobre Usabilidade (Associação Brasileira de Normas Técnicas, 2002).

Dentro da escala SUS (figura 16), cada questão é avaliada da seguinte forma: das alternativas pares (2,4,6,8,10), é pego o valor da alternativa e subtraído em 1. Das ímpares (1,3,5,7,9), subtrai-se 5 do valor da alternativa. Isso porque, as questões pares são as questões "afirmativas" e as questões ímpares são as "negativas". Logo, somam-se todos os valores, multiplica-se por 2,5 e obtém-se o score final, que pode ser comparado na ilustração representada pela figura 6. Sendo assim, o objetivo é alcançar, dentro dessa escala de usabilidade, uma nota de 70 ou superior para dar prosseguimento com os testes do jogo.

Figura 3 – Escala SUS



Fonte: Brooke (2013)

2.3.2 Refinamento de Mecânicas

Foi recolhido um feedback dos especialistas, com comentários acerca do que poderia ser melhorado ou implementado no projeto. Se a primeira versão do protótipo apresentasse uma nota inferior a 70 na escala SUS, nesse caso, seria necessário, após a correção de erros, uma nova bateria de testes com especialistas para que eles avaliem a versão nova do jogo.

2.4 CONCLUSÃO DO GAME

2.4.1 Discussão de Resultados

Por fim, foi realizada uma discussão de resultados acerca do projeto, com base no feedback dos especialistas e na avaliação utilizando o *System Usability Scale*, sendo esta uma parte essencial para a redação do artigo final.

2.4.2 Entrega do Projeto

A seguir, o jogo foi exportado em sua versão final para computadores, utilizando a própria ferramenta de "*build and export*" da *Unity*.

3. RESULTADOS

3.1 DOCUMENTAÇÃO DE CONCEITO DO JOGO

Quadro 4 – Requisitos Funcionais: Iniciar Jogo

Requisitos Funcionais	Prioridade
[RF001] Iniciar jogo	Essencial
Descrição: O jogo deve permitir iniciar um novo jogo.	
Detalhes da implementação prevista: **Entradas e pré-condições:** O jogador deve estar no menu inicial e pressionar o botão "jogar". **Saídas e pós-condições:** O sistema abre a cena "jogo".	

Quadro 5 – Requisitos Funcionais: Ver Pontuações

Requisitos Funcionais	Prioridade
[RF002] Acessar Pontuações	Essencial
Descrição: O jogo deve abrir a cena "pontuações".	
Detalhes da implementação prevista: **Entradas e pré-condições:** O jogador deve estar no menu inicial e pressionar o botão "pontuações". **Saídas e pós-condições:** O sistema abre a cena "pontuações".	

Quadro 6 – Requisitos Funcionais: Ver Créditos

Requisitos Funcionais	Prioridade
[RF003] Acessar Créditos	Essencial
Descrição: O jogo deve permitir iniciar um novo jogo. Caso haja um save, o jogo deve carregá-lo. Caso não haja um save, o jogo deve começar do início.	
Detalhes da implementação prevista:	

Entradas e pré-condições: O jogador deve estar no menu inicial e pressionar o botão “créditos”. **Saídas e pós-condições:** O sistema abre a cena “créditos”.

Quadro 7 – Requisitos Funcionais: Voltar ao Menu

Requisitos Funcionais	Prioridade
[RF004] Voltar para o Menu Principal	Essencial
Descrição: O jogo deve permitir que o jogador acesse o menu principal	
Detalhes da implementação prevista: **Entradas e pré-condições:** O jogador deverá estar na tela de Fases, Créditos ou Pontuações e acessar o botão de voltar. **Saídas e pós-condições:** O jogador é direcionado ao menu principal.	

Quadro 8 – Requisitos Funcionais: Sair do Jogo

Requisitos Funcionais	Prioridade
[RF005] Sair do Jogo	Essencial
Descrição: O jogo deve permitir o jogador a sair do jogo.	
Detalhes da implementação prevista: **Entradas e pré-condições:** O jogador deve selecionar a opção “Sair do Jogo” no menu principal. Para isso o menu principal deve estar aparecendo na tela. **Saídas e pós-condições:** A aplicação é encerrada.	

Quadro 9 – Requisitos Funcionais: Digitar

Requisitos Funcionais	Prioridade
[RF006] Digitar na Célula	Essencial
Descrição: O jogo deve permitir o jogador a escrever na célula de braile	
Detalhes da implementação prevista: **Entradas e pré-condições:** O jogo deve estar na fase de questões, com o teclado desbloqueado para o jogador poder dar *inputs*. As teclas válidas são s, d, f, j, k, l, h.	

Saídas e pós-condições: Caso o jogador tenha pressionado s, d, f, j, k, l, esses valores devem ser armazenados binariamente, sendo posteriormente, convertidos em uma saída decimal, que indicará ao sistema qual foi a letra escrita através do input do jogador. Se a tecla for “h”, a célula é limpa, retornando ao estado binário 000000.

Quadro 10 – Requisitos Funcionais: Confirmar Célula

Requisitos Funcionais	Prioridade
[RF007] Confirmar a Célula	Essencial
Descrição: O jogo deve permitir o jogador a confirmar sua entrada de célula.	
Detalhes da implementação prevista: **Entradas e pré-condições:** O jogo deve estar na fase de questões, com o teclado desbloqueado para o jogador poder dar inputs. Ele deve ter pressionado [espaço]. **Saídas e pós-condições:** O sistema irá interpretar o valor binário, convertendo-o através de um sistema para ver se a letra está correta.	

Quadro 11 – Requisitos Funcionais: Verificar Célula

Requisitos Funcionais	Prioridade
[RF008] Verificar Célula	Essencial
Descrição: O jogo deve permitir que o jogador responda à questão.	
Detalhes da implementação prevista: **Entradas e pré-condições:** Após digitar na célula, o jogador pressiona a tecla correspondente ao envio da resposta [ESPAÇO]. **Saídas e pós-condições:** Inicia-se uma validação entre o que foi lido e a resposta esperada pelo sistema. Caso a resposta esteja igual à resposta esperada, o jogador ganha um “ponto” e passa para a próxima questão. Caso contrário, o jogador não ganha um “ponto”, mas passa para a próxima questão mesmo assim.	

Quadro 12 – Requisitos Funcionais: Pontuações

Requisitos Funcionais	Prioridade
[RF009] Atribuir Pontuação Final	Essencial

Descrição: O jogo deve atribuir a pontuação final de cada fase.
Detalhes da implementação prevista: **Entradas e pré-condições:** O jogador deve completar uma fase com sucesso. **Saídas e pós-condições:** Ao término da fase, o sistema calcula a pontuação final do jogador.

Quadro 13 – Requisitos Funcionais: Salvar Progresso

Requisitos Funcionais	Prioridade
[RF010] Salvar Progresso	Essencial
Descrição: O jogo deve salvar o progresso do jogador.	
Detalhes da implementação prevista: **Entradas e pré-condições:** O jogador deve completar uma fase com sucesso. **Saídas e pós-condições:** O sistema deve salvar o progresso, levando em conta: pontuação e fases desbloqueadas. Isso será salvo em um arquivo .bin.	

Quadro 14 – Requisitos Funcionais: Sair do Nível

Requisitos Funcionais	Prioridade
[RF011] Sair do nível	Essencial
Descrição: O jogo deve permitir o jogador a sair do nível ou fase.	
Detalhes da implementação prevista: **Entradas e pré-condições:** O jogador deve estar na seleção de níveis. **Saídas e pós-condições:** O jogador é direcionado à tela de menu principal e o seu progresso é salvo automaticamente.	

Quadro 15 – Requisitos Funcionais: Acessar Pré-Menu

Requisitos Funcionais	Prioridade
[RF012] Acessar Tela de Pré-Menu	Essencial
Descrição: O jogo deve permitir o jogador a acessar a “Tela de Pré-Menu”	

Detalhes da implementação prevista: **Entradas e pré-condições:** O jogador inicia o jogo. **Saídas e pós-condições:** O jogador é direcionado à tela de Pré-Menu.111

Quadro 16 – Requisitos Funcionais: Acessar Como jogar

Requisitos Funcionais	Prioridade
[RF013] Acessar Tutorial: Como Jogar	Essencial
Descrição: O jogo deve permitir o jogador a acessar a tela “Como Jogar”	
Detalhes da implementação prevista: **Entradas e pré-condições:** O jogador entra na tela “Seleção de Fases” e seleciona o botão “Como Jogar” **Saídas e pós-condições:** O jogador é direcionado à tela de “Como Jogar”.	

Quadro 17 – Requisitos Funcionais: Acessar Vogais

Requisitos Funcionais	Prioridade
[RF014] Acessar Tutorial: Vogais	Essencial
Descrição: O jogo deve permitir o jogador a acessar a tela “Tutorial Vogais”	
Detalhes da implementação prevista: **Entradas e pré-condições:** O jogador entra na tela “Seleção de Fases” e seleciona o botão “Tutorial Vogais” **Saídas e pós-condições:** O jogador é direcionado à tela de “Tutorial Vogais”.	

Quadro 18 – Requisitos Funcionais: Acessar Fases

Requisitos Funcionais	Prioridade
[RF015] Acessar Seleção de Fases	Essencial
Descrição: O jogo deve permitir o jogador a acessar a tela “Tutorial Vogais”	
Detalhes da implementação prevista: **Entradas e pré-condições:** O jogador entra na tela “Seleção de Fases” e seleciona o botão “Tutorial Vogais”	

Saídas e pós-condições: O jogador é direcionado à tela de “Tutorial Vogais”.

Quadro 19 – Requisitos Não Funcionais: Engine

Requisitos Não Funcionais	Prioridade
[RNF001] *Engine* do Jogo	Essencial
Funcionamento do jogo	
Descrição: O jogo deverá ser desenvolvido na *engine Unity*, versão 2021.2.19f1.	

Quadro 20 – Requisitos Não Funcionais: Especificações

Requisitos Não Funcionais	Prioridade
[RNF002] Especificações mínimas do jogo	Essencial
Compatibilidade	
Descrição: O jogo deverá ser executado em computadores com especificações mínimas de: sistema operacional *Windows* 7, 8, 8.1 ou 10 com 64-32 bits, processador *Intel Core2 Duo*, memória 4GB de RAM, memória de vídeo de 512MB, *DirectX* 9.0 ou superior e armazenamento de 2GB.	

Quadro 21 – Requisitos Não Funcionais: Salvar Jogo

Requisitos Não Funcionais	Prioridade
[RNF003] Salvar/Carregar o Jogo	Essencial
Confiabilidade	
Descrição: O sistema deverá se comunicar com o banco de dados para realizar: (1) salvamento e carregamento de dados de jogadores e (2) seleção de questões de forma aleatória	

Quadro 22 – Requisitos Não Funcionais: Configurações de Som

Requisitos Não Funcionais	Prioridade
[RNF004] Ajuste de som	Importante
Desempenho	

Descrição: O jogo deve permitir ao jogador configurar os níveis de som e efeitos sonoros do jogo.
Detalhes da implementação prevista: **Entradas e pré-condições:** O jogador deverá acessar o botão de configurações no menu inicial. **Saídas e pós-condições:** Abre-se uma tela sobreposta ao menu principal, onde o jogador poderá fazer as alterações necessárias de áudio.

Quadro 23 – Requisitos Não Funcionais: Configurações de Interface

Requisitos Não Funcionais	Prioridade
[RNF005] Ajuste de interface	Importante
Desempenho	
Descrição: O jogo deve permitir o jogador a configurar a interface do jogo: resolução de tela e modo de exibição.	
Detalhes da implementação prevista: **Entradas e pré-condições:** O jogador deverá acessar o botão de configurações no menu inicial. **Saídas e pós-condições:** Abre-se uma tela sobreposta ao menu principal, onde o jogador poderá fazer as alterações necessárias da interface.	

Quadro 24 – Requisitos Não Funcionais: Verificar Arquivos Salvos

Requisitos Não Funcionais	Prioridade
[RNF006] Verificar Arquivos Salvos	Essencial
Disponibilidade	
Descrição: Após iniciar um jogo, caso haja um registro de salvamento, o jogo deve carregá-lo. Caso não haja um *save*, o jogo deve criar um.	
Detalhes da implementação prevista: **Entradas e pré-condições:** O jogador deve ter acabado de entrar no jogo. **Saídas e pós-condições:** Nesse momento, o sistema deverá buscar se há ou não há um *save*. Caso haja um *save*, deve carregá-lo e apresentar as informações na tela,	

tais como: fases concluídas e medalhas recebidas, além de desbloquear as fases habilitadas no *save* previamente.

As regras de negócio estão apresentadas nos quadros 26 a 29:

Quadro 25 – Regras de Negócio: RN001

Regras de negócio
[RN001] Desbloqueio de Fases
Descrição: O jogador deverá desbloquear os próximos níveis somente quando ele atingir 70% do nível atual.
Fonte: Thiago Yasue
Histórico: 26/09/2022

Quadro 26 – Regras de Negócio: RN002

Regras de negócio
[RN002] Pontuação Atribuída
Descrição: A cada acerto por caractere, o jogador recebe (1000 * *streak)* pontos e em seguida, o *streak* é aumentado da seguinte ordem: 1, 2, 4, 6 e 8. Se o contador estiver em 8 (valor máximo), o *streak* não é mais aumentado. A cada erro de caractere, o jogador não recebe pontos e seu *streak* é reiniciado para 1. A cada palavra acertada completamente, o jogador recebe 1000 * *streak* pontos. Ao término daquela fase, é analisado se o jogador atingiu ao menos 70% de precisão ao teclar corretamente, além de mostrar a quantidade de palavras corretas e palavras erradas.
Fonte: Thiago Yasue
Histórico: 26/09/2022

Quadro 27 – Regras de Negócio: RN003

Regras de negócio
[RN003] Randomização
Descrição: Cada questão que o jogador irá responder será dada através de um banco de questões armazenados em classes do tipo *Scriptable Object* que irá randomizar e enviar uma questão, com o propósito de minimizar o problema de o usuário acertar a questão apenas por decorar, mitigando este viés.
Fonte: Thiago Yasue
Histórico: 26/09/2022

Quadro 28 – Regras de Negócio: RN005

Regras de negócio
[RN005] Desbloqueio de fase
Descrição: A cada fase concluída com sucesso, isto é, concluída com 70% ou mais de precisão, o jogador deverá ser recompensado com uma insígnia/emblema/medalha, subindo-o de “cargo” na “agência secreta”.
Fonte: Thiago Yasue
Histórico: 26/09/2022

Figura 4 – Diagrama de Caso de Uso: Menu

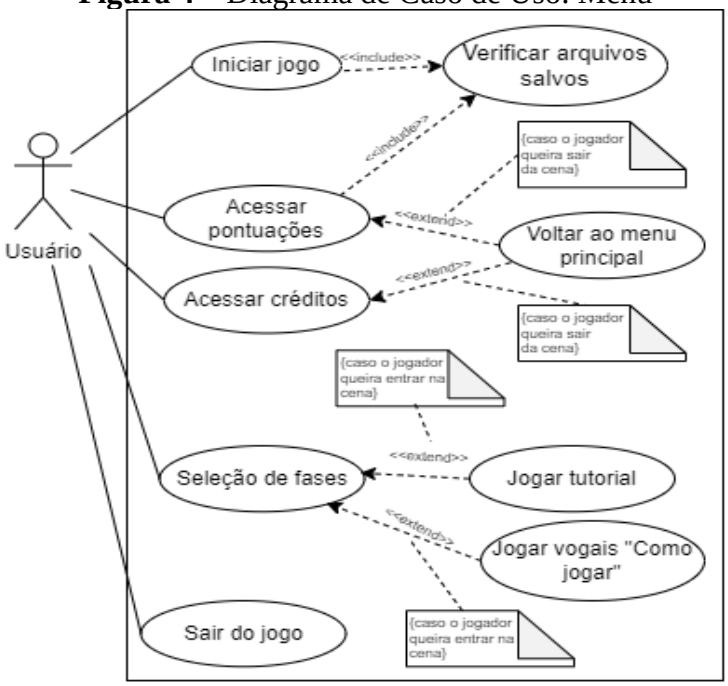
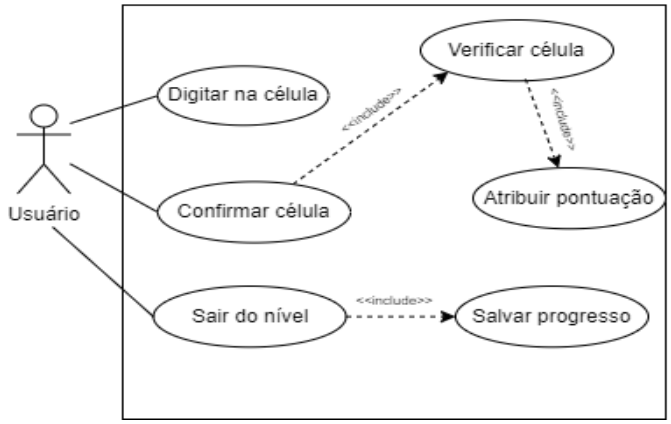


Figura 5 – Diagrama de Caso de Uso: In-Game



Os quadros abaixo, 30 e 31, apresentam a descrição expandida de cada diagrama de caso de uso apresentados anteriormente, indicando cada requisito funcional a ser utilizado e mostrando um fluxo básico como exemplo. É importante salientar que o funcionamento dos tutoriais segue o mesmo caso de uso da *gameplay* padrão, então para ambos os casos, é referido o caso de uso do quadro 31.

Quadro 29 – Especificação de Caso de Uso: Menu Principal

Especificação de Caso de Uso: Menus
Referências:

RF001, RF002, RF003, RF004, RF005, RF012, RF013, RF014, RF015
Descrição Geral: Interação do usuário com o menu principal do aplicativo.
Atores: Usuário, Sistema
Pré-Condições: O usuário se encontra na tela de “Menu Principal”.
Fluxo Básico: 1. O usuário inicia um jogo [RF001] 2. O usuário acessa a tela de Pré-Menu [RF012]
Fluxo Alternativo: 1. O usuário acessa as pontuações [RF002] 2. O sistema o retorna para o menu principal [RF004] 3. O usuário acessa os créditos [RF003] 4. O sistema o retorna para o menu principal [RF004] 5. O usuário acessa a seleção de fases [RF015] 6. O usuário acessa o tutorial [RF013] 7. O sistema o retorna para o menu principal [RF004] 8. O usuário acessa a seleção de fases [RF015] 9. O usuário acessa a fase de vogais [RF014] 10. O sistema o retorna para o menu principal [RF004] O usuário fecha o aplicativo [RF005]

Quadro 30 – Especificação de Caso de Uso: In-Game

Especificação de Caso de Uso: *In-Game*
Referências: RF006, RF007, RF008, RF009, RF010
Descrição Geral: Interação do usuário com o jogo, ou *gameplay*.
Atores: Usuário, Sistema
Pré-Condições: O usuário se encontra no jogo, após tê-lo iniciado [RF001];

Fluxo Básico:

1. O usuário digita na célula de braile [RF006]
2. O sistema verifica se aquela célula está correta [RF007]
3. O sistema atribui a pontuação correspondente, caso haja [RF008]
4. O jogador decide sair do nível, voltando ao menu principal [RF010]
5. O sistema salva o progresso até a última fase completa [RF009]

Quadro 31 – Tarefas de Definição de Escopo – Dependências

#	Atividade	Dependência
A	**Divisão de tarefas**	**Nenhum**
B	**Levantamento de Requisitos**	**Nenhum**
1	Elaboração dos Requisitos Funcionais e Não-Funcionais	
2	Elaboração das Regras de Negócio	
C	**Elaboração do Cronograma**	**A**
D	**Análise de Riscos**	**C**

Quadro 32 – Tarefas de Desenvolvimento: Roteiro – Dependências

#	Atividade	Dependência
A	**Desenvolver o enredo**	**Nenhum**
1	História Geral do Jogo	
2	Vozes, Narrativa etc.	
B	**Documentar áudios/falas dos personagens**	**A**
C	**Listar os efeitos sonoros e músicas**	**A**
D	**Conceitualizar a Gameplay**	**A**
1	Definir mecânica de *input*	
2	Definir mecânica de transição de fases de gameplay	
3	Definir como funcionará a pontuação, progressão de nível e score geral.	
E	**Conceitualizar Level Design**	**A,D**
1	Conceitualizar Fase “Como Jogar”	
2	Conceitualizar Fase tutorial de vogais	
3	Conceitualizar Fase Teste	
4	Conceitualizar Fase Texto Livre	

F	**Conceitualizar a Interface**	**D,E**
1	Conceitualizar Tela "pré-jogo"	
2	Conceitualizar Tela do Menu Principal	
3	Conceitualizar Interface do Menu Principal	
4	Conceitualizar Tela de “Pontuações”	
5	Conceituar Interface da Gameplay	
6	Conceituar Interface da Tela "Créditos"	

Quadro 33 – Tarefas de Desenvolvimento: Programação – Dependências

#	**Atividade**	**Dependência**
G	**Programar *In-Game***	**D**
1	Narração *in-game*	
2	Interface *in-game*	
3	Sistema de inputs na célula de braile	
4	Sistema de pontuação e progressão de fases	
5	Sistema de Leitura de células baseado no contexto narrativo (integração de partes)	
H	**Programar Áudios, Sons, Controle de Áudio**	**B**
I	**Programar Interface e Botões**	**F**
J	**Programar os sistemas de persistência de dados**	**G**

Quadro 34 – Tarefas de Desenvolvimento: Arte – Dependências

#	**Atividade**	**Dependência**
K	***Wireframe* das interfaces**	**E**
L	**Desenvolver a composição *in-game***	**D**
1	Cenário	
2	Itens	
M	**Desenvolver a Interface**	**E**
1	Desenvolver Arte do Logo do *Game*	
2	Desenvolver Arte dos Ícones	
3	Desenvolver Arte da tela de Créditos	

Quadro 35 – Tarefas de Desenvolvimento: Áudio – Dependências

#	Atividade	Dependência
N	**Desenvolvimento dos Efeitos Sonoros**	**d**
O	**Desenvolvimento das Músicas**	**d**
P	**Desenvolvimento das Vozes**	**c**

Quadro 36 – Tarefas de Avaliação do Game – Dependências

#	Atividade	Dependência
A	**Teste do Alpha do *Game***	**Nenhum**
B	**Refinamento de Mecânicas**	**A**
C	**Teste do Beta do Game**	**B**

Quadro 37 – Tarefas de Conclusão – Dependências

#	Atividade	Dependência
A	**Discussão de Resultados**	**Nenhum**
B	**Documentação Final**	**A**

Quadro 38 – Tarefas de Definição de Escopo – Duração

#	Atividade	Duração
A	**Divisão de tarefas**	**1 dia**
B	**Levantamento de Requisitos**	**2 dias**
1	Elaboração dos Requisitos Funcionais e Não-Funcionais	1 dia
2	Elaboração das Regras de Negócio	1 dia
C	**Elaboração do Cronograma**	**1 dia**
D	**Análise de Riscos**	**1 dia**

Quadro 39 – Tarefas de Desenvolvimento: Roteiro – Duração

#	Atividade	Duração
a	**Desenvolver o enredo**	**10 dias**
1	História Geral do Jogo	2
2	Vozes, Narrativa etc.	8

b	**Documentar áudios/falas dos personagens**	**10 dias**
c	**Listar os efeitos sonoros e músicas**	**2 dias**
d	**Conceitualizar a *Gameplay***	**3 dias**
1	Definir mecânica de input	1
2	Definir mecânica de transição de fases de gameplay	1
3	Definir como funcionará a pontuação, progressão de nível e score geral.	1
e	**Conceitualizar *Level Design***	**2 dias**
1	Conceitualizar Fase “Como Jogar”	1
2	Conceitualizar Fase “Vogais”	1
3	Conceitualizar Fase Teste	
4	Conceitualizar Fase Texto Livre	
f	**Conceitualizar a Interface**	**2 dias**
1	Conceitualizar Tela pré-menu principal	2
2	Conceitualizar Tela do Menu Principal	
3	Conceitualizar Interface do Menu Principal	
4	Conceitualizar Tela de “Pontuações”	
5	Conceituar Menu de Configurações	
6	Conceituar Interface da Gameplay	
7	Conceituar Interface da Tela "Créditos"	

Quadro 40 – Tarefas de Desenvolvimento: Programação – Duração

#	Atividade	Duração
G	**Programar *In-Game***	**91 dias**
1	Narração *in-game*	28 dias
2	Interface *in-game*	
3	Sistema de inputs na célula de braile	28 dias
4	Sistema de pontuação e progressão de fases	28 dias
5	Sistema de Leitura de células baseado no contexto narrativo (integração de partes)	7 dias
H	**Programar Áudios, Sons, Controle de Áudio**	**1 dia**
I	**Programar Interface e Botões**	**1 dia**
J	**Programar os sistemas de persistência de dados**	**2 dias**

Quadro 41 – Tarefas de Desenvolvimento: Arte – Duração

#	Atividade	Duração
K	**Wireframe das interfaces**	**14 dias**
L	**Desenvolver a composição *in-game***	**14 dias**
1	Cenários	7 dias
2	Itens	7 dias
M	**Desenvolver a Interface**	**6 dias**
1	Desenvolver Arte do Logo	1 dia
2	Desenvolver Arte dos Ícones	4 dias
3	Desenvolver Arte da Tela de Créditos	1 dia

Quadro 42 – Tarefas de Desenvolvimento: Áudio – Duração

#	Atividade	Duração
N	Desenvolvimento dos Efeitos Sonoros	7 dias
O	Desenvolvimento das Vozes, Narração etc.	7 dias
P	Desenvolvimento de Músicas	7 dias

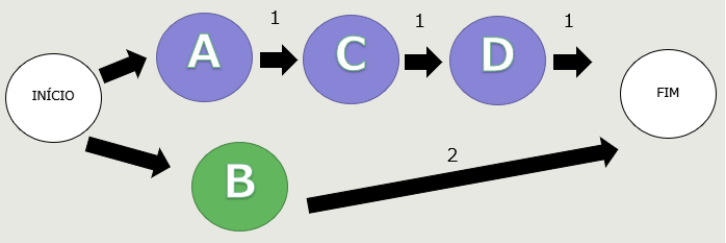
Quadro 43 – Avaliação do Game – Duração

#	Atividade	Duração
A	**Teste do *Alpha* do *Game***	**2 dias**
B	**Refinamento de Mecânicas**	**7 dias**
C	**Teste do *Beta* do *Game***	**2 dias**

Quadro 44 – Conclusão do Game – Duração

#	Atividade	Duração
a	**Discussão de Resultados**	**1 dia**
b	**Documentação Final**	**1 dia**

Figura 6 – Diagrama de Redes: Definição de Escopo



No diagrama de redes que diz respeito ao desenvolvimento do game (Figura 7), foram utilizadas quatro cores nas atividades: lilás, azul-naval, verde-água e verde-floresta, que representam cada etapa do desenvolvimento, respectivamente nessa ordem: roteiro e conceitualização, programação, arte e áudio.

Figura 7 – Diagrama de Redes: Desenvolvimento do Game

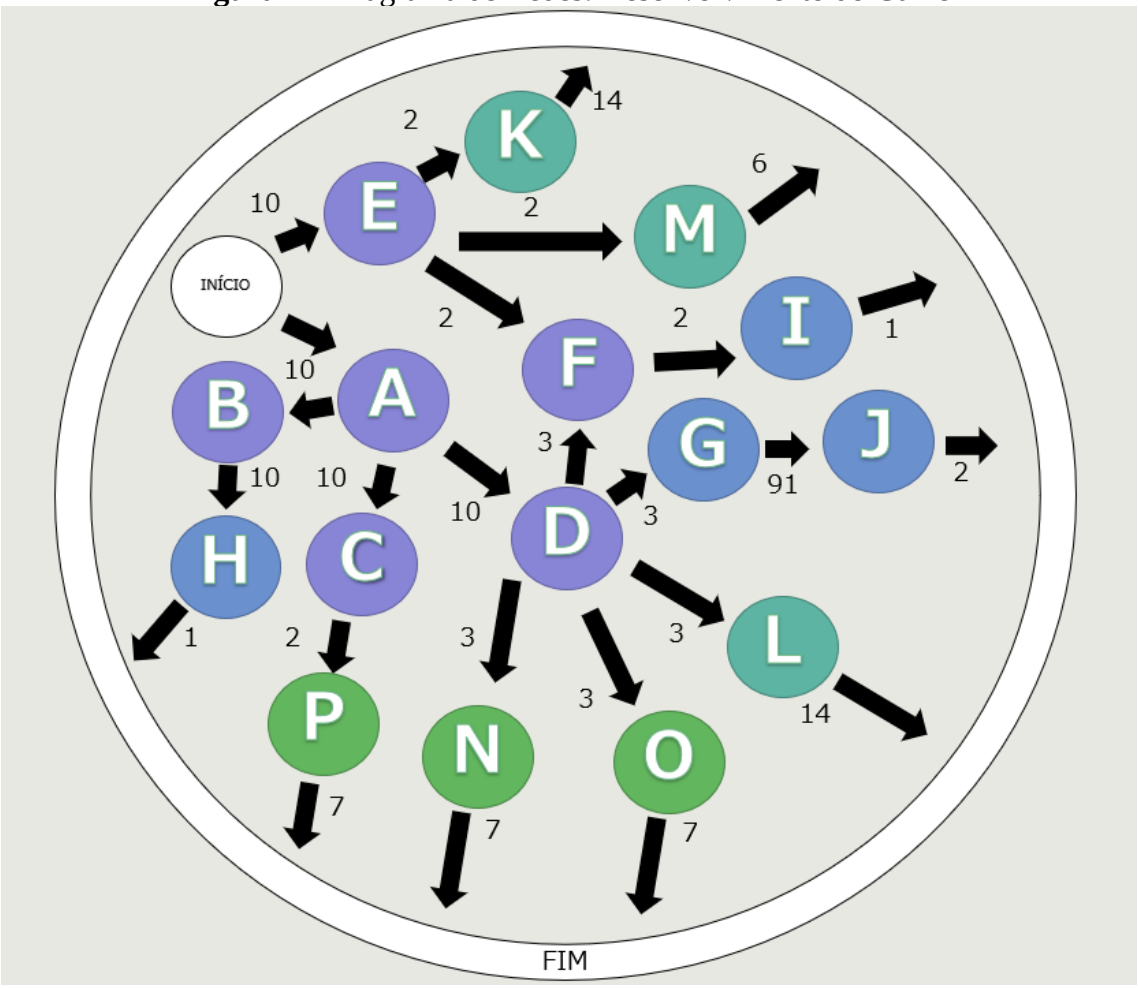


Tabela 1 – Caminho crítico: Definição de Escopo

Atividade	Duração (dias)
A	1
C	1
D	1
Total:	**3 dias**

Tabela 2 – Caminho crítico: Desenvolvimento do Game

Atividade	Duração (dias)
A	10
D	3
G	91
J	2
Total:	**106 dias**

Figura 8 – Diagrama de Redes: Avaliação do Game

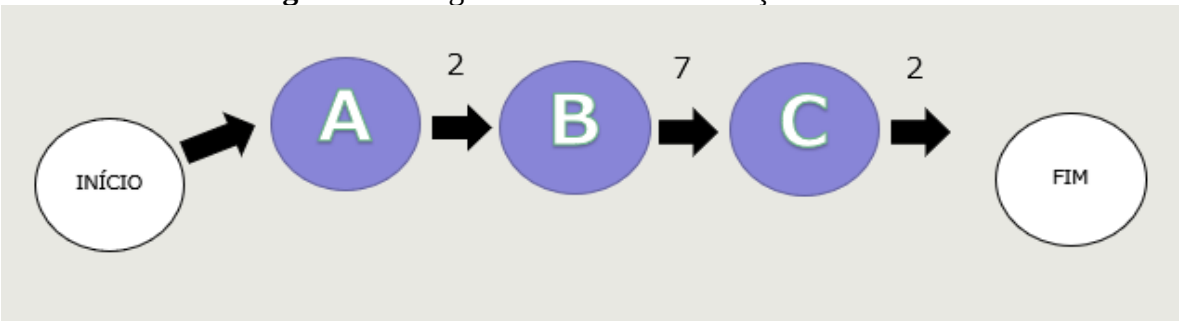


Figura 9 – Diagrama de Redes: Conclusão do Game

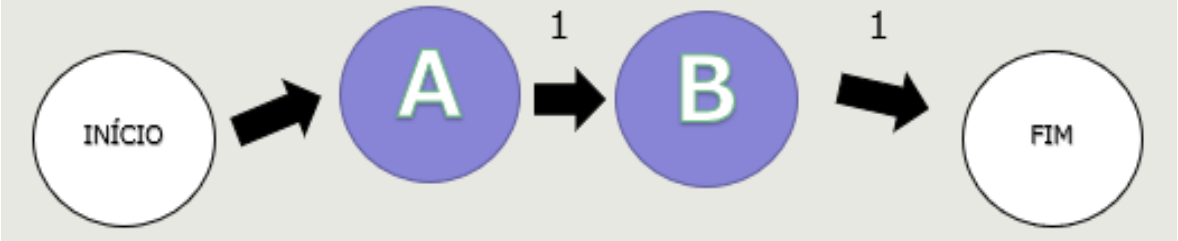


Tabela 3 – Caminho crítico: Avaliação do Game

Atividade	Duração (dias)
A	2
B	7
C	2
Total:	**11 dias**

Tabela 4 – Caminho crítico: Conclusão do Game

Atividade	Duração (dias)
A	1
B	1
Total:	**2 dias**

Quadro 45 – Risco 01

Risco	Probabilidade	Impacto	Prioridade
#001 – Erros	26% a 50%	Severo	**Alta**
Descrição: O projeto final pode conter *bugs*, falhas graves de navegação, entre outros erros.			
Plano de ação: Evitar o risco: O jogo será testado (ver etapa de Avaliação do Game): através de testes de caixa branca, caixa preta e testes de usabilidade.			

Quadro 46 – Risco 02

Risco	Probabilidade	Impacto	Prioridade
#002 – Prazos	26% a 50%	Moderado	**Alta**
Descrição:			

É possível que o jogo não seja entregue no prazo estipulado devido a atrasos e gargalos na produção do game (arte, modelagem, animação, programação, *game design*, documentação).
Plano de ação: Mitigar o risco: Serão criadas metas mensais e cronogramas. Assim, os possíveis atrasos podem ser reduzidos ou até mesmo eliminados.

Quadro 47 – Risco 03

Risco	Probabilidade	Impacto	Prioridade
#003 – Equipamento da Equipe	26% a 50%	Moderado	**Alta**
Descrição: O desenvolvimento do jogo pode ser comprometido caso o projeto seja apagado, o hardware seja comprometido etc.			
Plano de ação: Mitigar o Risco: Todos os arquivos possuem pelo menos três cópias, ou em nuvem ou em mídia de armazenamento física.			

3.2 ROTEIRO

Dando um contexto em meio a uma narrativa, o jogador irá aprender o básico de braile, sendo instruído a digitar utilizando as teclas S-D-F-J-K-L, como se ele estivesse aprendendo um código para se comunicar com agentes secretos que estarão em bases secretas. Vale dizer mais uma vez que essas teclas estão sendo utilizadas, pois a tecla “F” e a tecla “J” possuem um relevo, que as torna reconhecíveis em qualquer teclado, tornando o posicionamento dos dedos fáceis e intuitivo. O jogo começará com um tutorial, onde o jogador irá entender os conceitos básicos de braile e do game. Por exemplo: os pontos de uma célula e as teclas correspondentes a cada ponto. Cabe acrescentar que além das seis teclas básicas, o jogador ainda irá possuir teclas para deletar e avançar pontos e utilizar a barra de espaço para estruturar frases.

Dentro do game, o jogador ocupa o papel de um agente secreto de uma grande organização de espiões. O objetivo desta organização é proteger a fictícia cidade de *Hamana*,

uma capital de um importante país situado ao extremo oriente, anotando informações importantes utilizando um código confidencial – nesse caso, o braile, que será ensinado na fase de introdução. Ao longo da *gameplay*, o objetivo do jogador será ouvir determinada palavra e escrevê-la. Por exemplo, o objetivo da fase é ele descobrir a cor de um carro: na fase, ele consegue ouvir que o carro é azul, então ele digita “azul” para concluir a fase. Existem ainda fases de tutorial, que são mais simples, com o objetivo de ensinar os controles, a posição dos dedos no teclado e as letras individualmente, para que o jogador possa, de fato, jogar as fases. O objetivo principal do jogador é ir sendo promovido dentro da organização de agentes secretos, à medida em que progride em cada nível utilizando o sistema braile. A figura 10 apresenta as figuras dos emblemas que o jogador desbloqueia conforme conclui as fases.

Figura 10 – Emblemas



3.3 DESENVOLVIMENTO DO JOGO

3.3.1 Sistema de Digitação

O sistema de digitação em braile, mencionado nos objetivos do projeto, diz a respeito de um sistema capaz de converter os inputs do usuário em uma variável *string* (uma letra ou conjunto de letras). Desse modo, partiu-se do princípio de que o braile pode ser visto como um código binário, sendo “ponto não pressionado”, 0, e “ponto pressionado”, 1. Assim, foi criado um quadro com a descrição de cada letra em binário e sua correspondente versão de saída como um caractere. O dígito menos significativo, ou seja, o primeiro dígito do binário, mais à direita, é o ponto 1 da célula, e o dígito mais significativo, ou seja, o mais à esquerda, é o ponto 6. Desse modo, a letra “A”, que é representada pelo ponto 1 pressionado e os outros cinco pontos não pressionados, seria, em binário: 000001. O quadro abaixo, ordenado em binário, demonstra todas as saídas possíveis no jogo.

Quadro 48 – Conversão binário-caractere da célula de braile

Binário	Saída
000001	a
000011	b
000101	k
000111	l
001001	c
001010	i
001011	f
001101	m
001110	s
001111	p
010001	e
010011	h
010101	o
010111	r
011001	d
011010	j
011011	g
011101	n
011110	t
011111	q
100101	u
100111	v
101101	x
101111	ç
110101	z
111010	w
111101	y
111111	é

Esse sistema pode ser visto em funcionamento na próxima sequência de figuras, 11, 12 e 13, onde foi criado um sistema de testes, onde o jogador deve escrever a letra que aparece na tela, recebendo um “O” se estiver correto e um “X” se estiver errado. Assim, o jogador observa que deve escrever a letra “A” (figura 11), depois pressiona os pontos correspondentes à letra A (o ponto 1, canto superior esquerdo, como mostra a figura 12), e confirma a entrada de dados na figura 13 pressionando a barra de espaço. Logo após a confirmação, o jogador recebe um “O” no topo da tela, indicando que ele acertou. Em seguida, uma nova letra aparece para que ele digite novamente.

Figura 11 – Jogador observa a tela

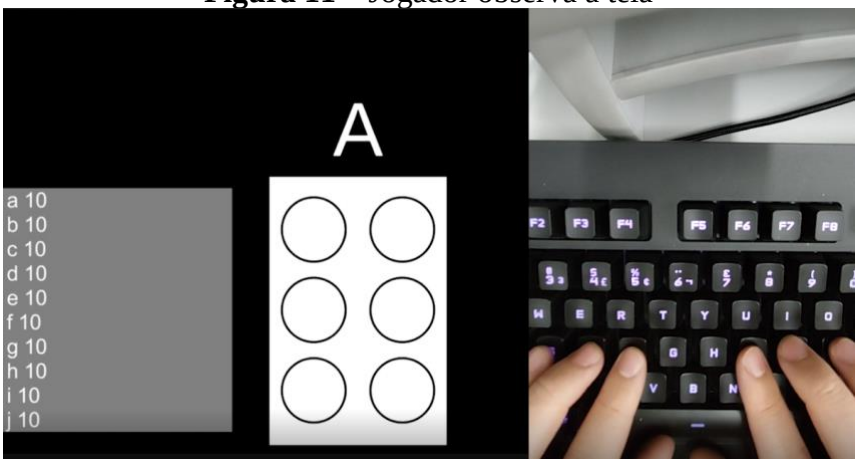


Figura 12 – Jogador utiliza o teclado

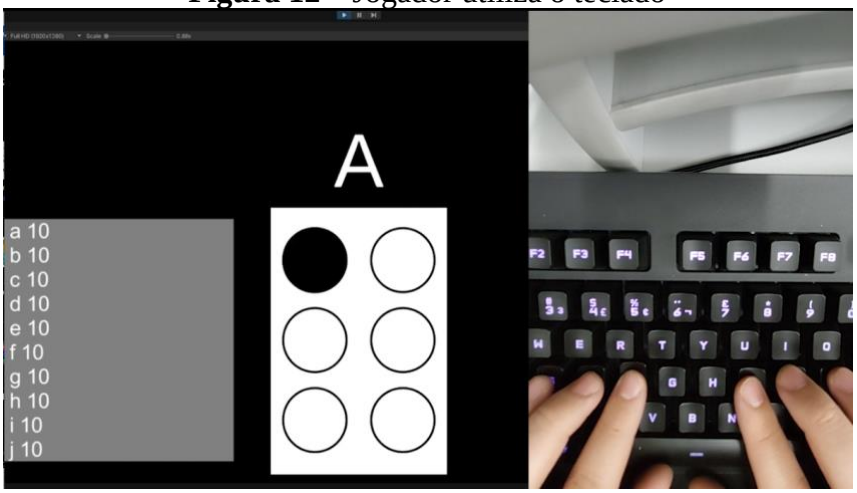
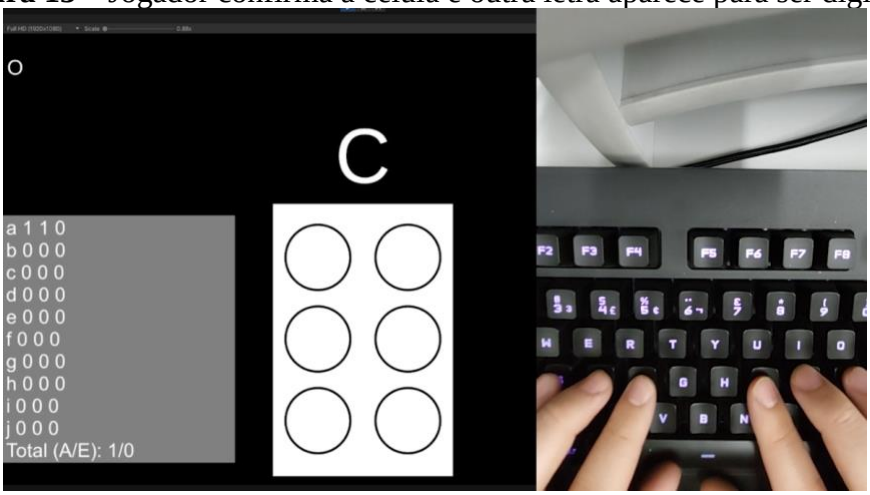


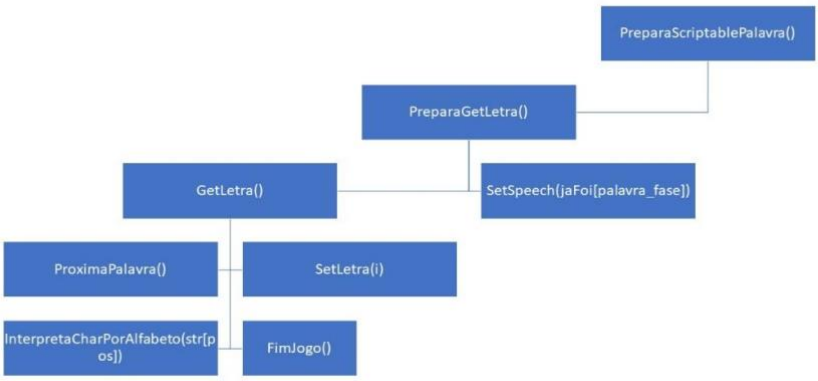
Figura 13 – Jogador confirma a célula e outra letra aparece para ser digitada



3.3.2 Game Loop

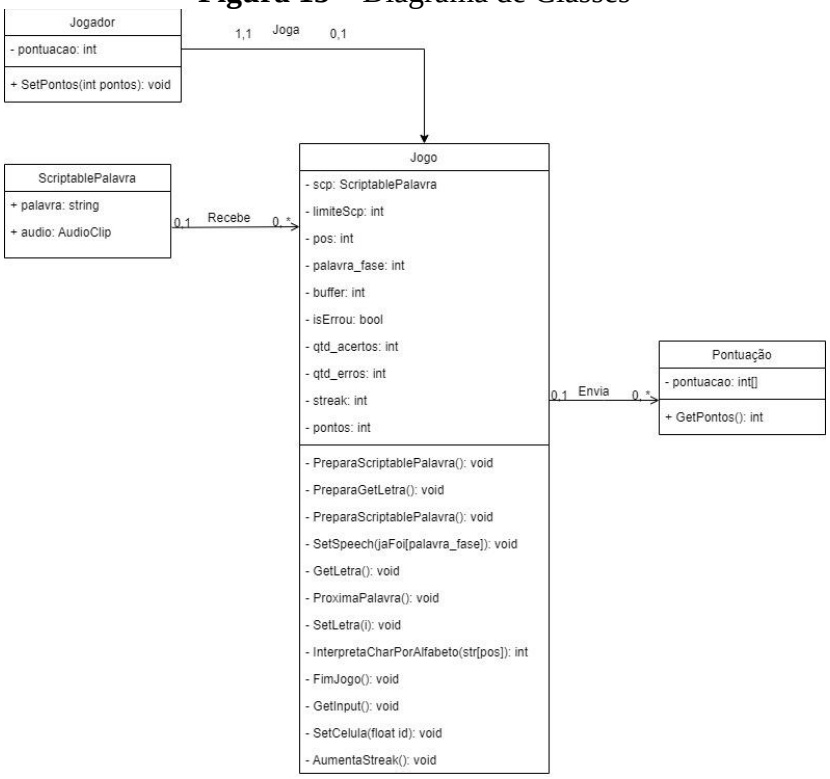
O funcionamento do sistema de jogabilidade do jogo forma um *game loop*, isto é, um sistema que se repete até o fim de jogo. A figura 14 apresenta os métodos que compõem o *game loop*.

Figura 14 – *Game Loop*



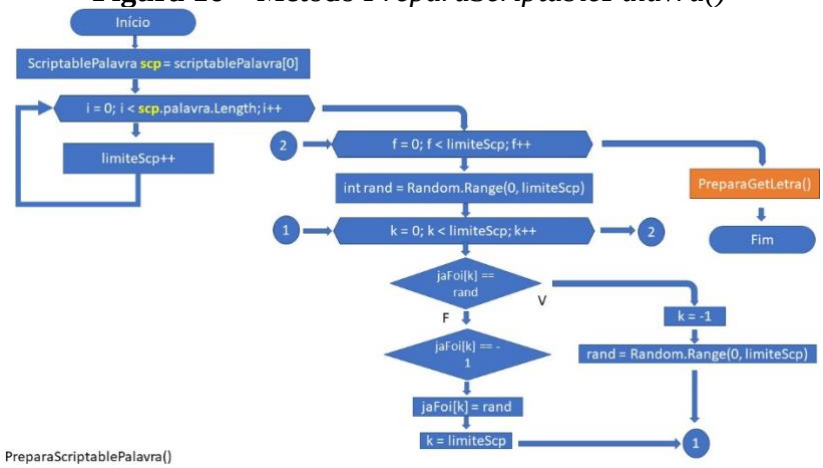
Esse *game loop* tem o objetivo primário de fazer com que um grupo de palavras, já estabelecido previamente, seja selecionado, então embaralhar essas palavras e em seguida, apresentar cada palavra, de letra em letra, para que o jogador as escreva. A figura 15 mostra visualmente as classes do jogo, apresentando métodos e variáveis de Operação Braille.

Figura 15 – Diagrama de Classes



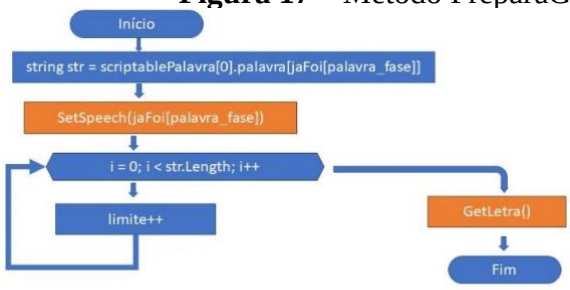
O método *PreparaScriptablePalavra()* é responsável por receber o *scriptablePalavra* a ser utilizado naquela fase. Após a instância, é necessário coletar o número total de palavras, pois é com este valor que o sistema saberá quando finalizar a fase. A seguir, é feito um embaralhamento das palavras, de modo que cada fase seja sempre randômica quanto à ordem de palavras a serem digitadas, armazenando-as em um vetor chamado *jaFoi[]*. Um fluxograma deste método é apresentado na figura 16.

Figura 16 – Método *PreparaScriptablePalavra()*



Logo em seguida, é necessário indicar qual é a palavra a ser escrita pelo jogador, uma de cada vez, respeitando a ordem do vetor *jaFoi[]*. Nesse caso, a palavra é selecionada e um áudio correspondente à palavra é emitido ao jogador. Além disso, uma variável coleta o tamanho da palavra, pois é com essa variável que o sistema saberá quando encerrar essa palavra e partir para a próxima. A figura 17 apresenta o fluxograma do método *PreparaGetLetra()*.

Figura 17 – Método PreparaGetLetra()



Após estabelecer o tamanho da palavra, o sistema irá avançar, letra por letra, o jogo, identificando a letra correta a ser digitada pelo jogador com base na variável "*pos"*, que coletará a posição da letra onde o jogador está a digitar neste momento. Esse sistema está representado na figura 18.

A interpretação da letra fica responsável pela função *InterpretaCharPorAlfabeto()*, que recebe *str[pos]*, ou seja, a posição da palavra. Esse método, apresentado na figura 19, tem o objetivo de transformar a letra em um número de 1 a 26, correspondente a cada letra do alfabeto.

Por fim nesta etapa, o método *SetLetra()*, apresentado na figura 20, é responsável por apresentar, na tela, a letra a ser digitada para o jogador vidente.

Figura 18 – Método *GetLetra()*

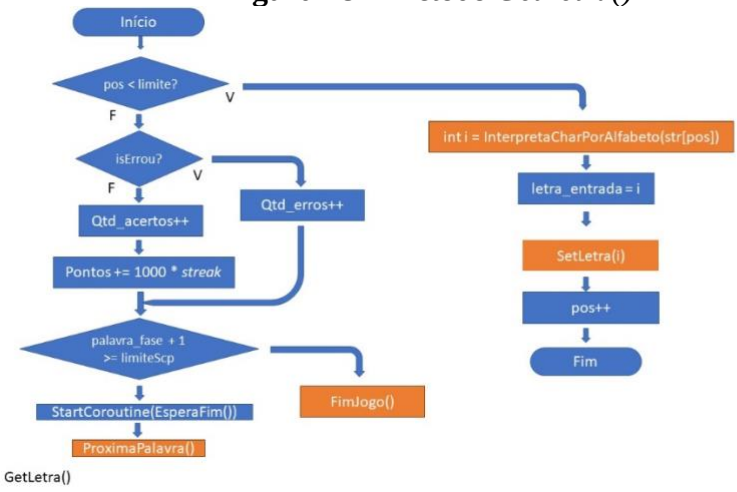


Figura 19 – Função InterpretaCharPorAlfabeto()

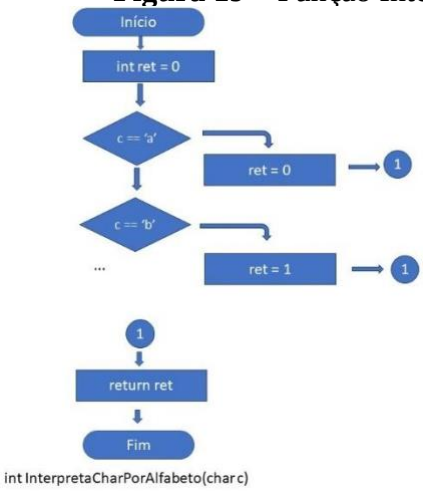
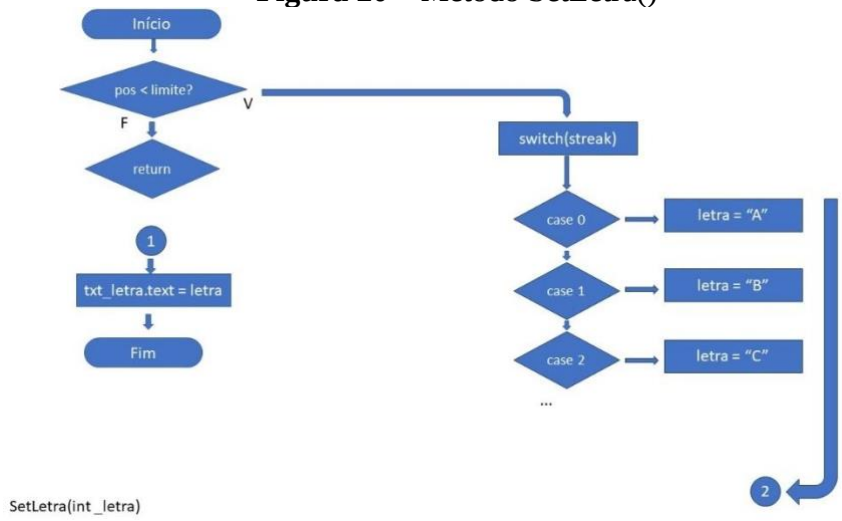


Figura 20 – Método SetLetra()



Ainda no método *GetLetra()*, caso a palavra tenha chegado ao final, o jogo irá analisar se o jogador acertou a palavra: para ter acertado a palavra, ele não deve ter errado nenhuma letra, caso contrário a *flag* "*isErrou"* será devidamente acionada. Se *isErrou* for verdadeiro, o jogador não recebe nenhum ponto por aquela palavra, porém, se acertou, ele receberá uma pontuação por ela.

Aqui, há mais uma condição: se essa foi a última palavra do conjunto de *scriptables*, o jogo encerra no método *FimFase()*, que apenas irá mostrar os resultados na tela. Caso contrário, o jogo se repete.

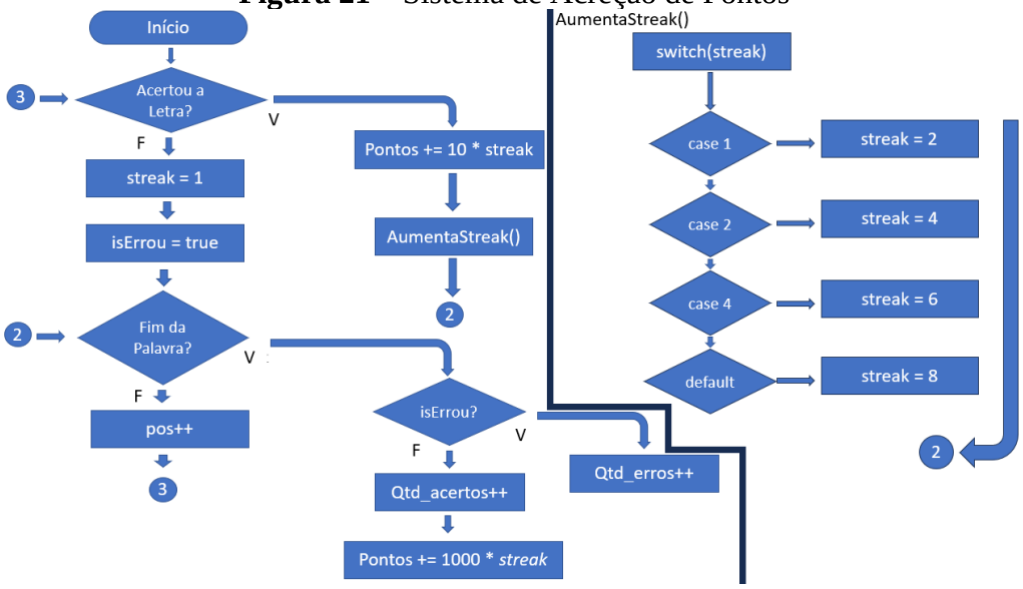
3.3.3 Gameplay e pontuação

Durante a *gameplay*, existem duas etapas de *gameplay* intercaladas: a etapa de "narrativa" e a etapa de "escrita". Durante a narrativa, o jogador vai explorar o universo, enquanto uma voz narra a história do jogo. Durante a fase de escrita, o usuário deve escrever corretamente em uma célula de braile uma letra ou conjunto de letras (palavras) e caso consiga atingir uma taxa de precisão por caractere de 70% ou superior naquela fase, ele é "promovido" e receberá um emblema por completar o nível. Apesar do jogo ser dividido em níveis, sendo eles: tutorial (instruções de controles do teclado); e fases comuns. As fases comuns serão progressivas, isto é, a medida em que o usuário é "promovido", ele terá acesso a fases mais difíceis, que darão continuidade à história do jogo. Essa promoção será feita por um

ranqueamento baseado na pontuação final em cada fase daquele jogador, que poderá ser visto na tela “Pontuações”, acessada do menu principal do jogo.

A pontuação do jogador é feita por um sistema de *streak*, ou sequência, que prioriza a precisão contínua do usuário, onde a cada caractere escrito corretamente, o jogador recebe um multiplicador de pontos baseado na quantidade da sequência de caracteres corretos. Se o que foi digitado estava errado, uma *flag* irá sinalizar e o sistema irá zerar o multiplicador, recomeçando-o. Além do *streak* de caracteres, também há uma pontuação bônus caso o jogador acerte a palavra por completo, pois, como a intenção nas fases é a escrita de palavras, a pontuação se baseia mais no acerto delas. A figura 21 apresenta como é feita a acreção de pontos por meio de um fluxograma.

Figura 21 – Sistema de Acreção de Pontos



3.3.4 Persistência de Dados

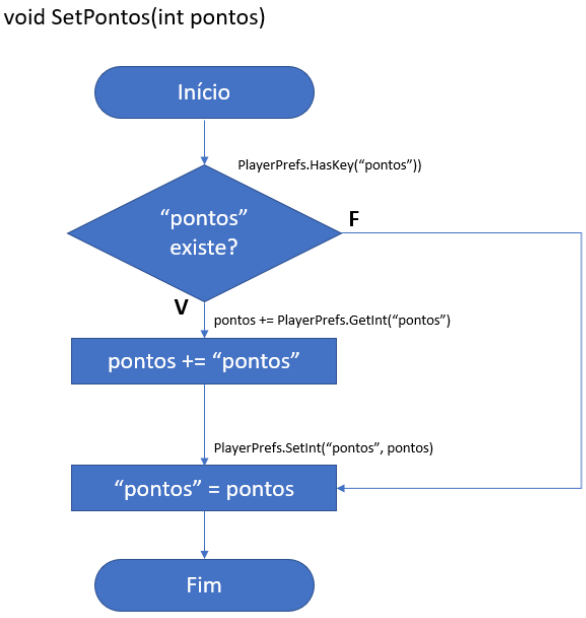
A parte do jogo com propósito de ensinar palavras é feita a partir de uma série de palavras randomicamente selecionadas que o jogo dá ao jogador, sendo assim, apesar do jogo ser linear, caso o jogador rejogue a mesma fase, ele não irá apenas decorar as entradas pré-feitas, criando um viés. O armazenamento de conjuntos de palavras foi feito utilizando objetos do tipo *Scriptable Objects* da própria *Unity*, onde são armazenados em pares de chave-valor os vetores com a *string* e sua respectiva linha de áudio, aliado a um código sorteador.

Cada uma das palavras dentro de um *Scriptable* possui um *id*, para fácil identificação com o arquivo de mídia de áudio e sua *string* de texto com o valor esperado a ser escrito. Caso

o texto escrito pelo usuário seja o mesmo valor esperado pelo sistema, isso significa que ele acertou a pergunta.

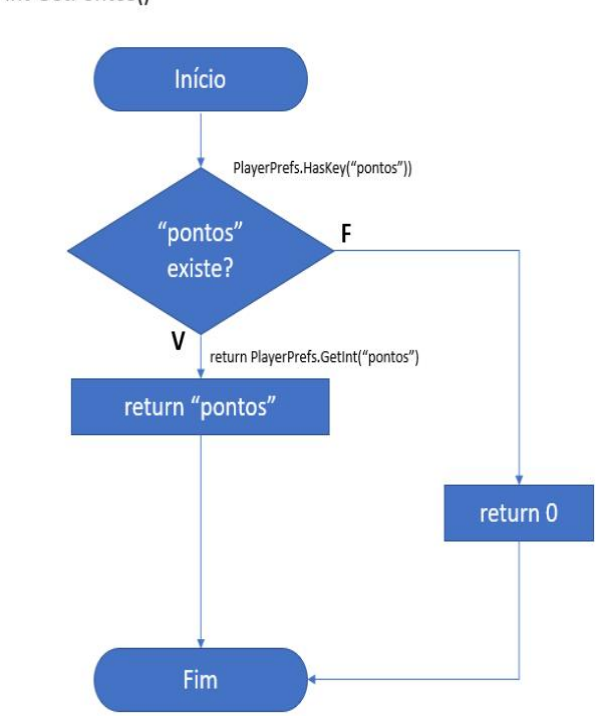
Durante a *gameplay*, os pontos que o jogador recebe ao completar uma letra ou palavra com sucesso são armazenados em uma chave utilizando a classe *PlayerPrefs*. A imagem abaixo representa um fluxograma do método de salvamento da pontuação total do jogador. Caso não haja uma chave com o nome "pontos", o sistema irá cria-la e armazenar a pontuação do jogador. Caso contrário, irá somar a última pontuação ao valor da chave "pontos". É importante destacar, neste fluxograma, a diferença entre *pontos* – sem aspas, correspondente à variável local e "*pontos*" – com aspas, correspondente à chave em *PlayerPrefs*.

Figura 22 – Fluxograma do Armazenamento de Pontos



Após o jogador ter completado uma fase, o resultado é registrado e exportado em um arquivo de texto com a sua pontuação, taxa de erros e acertos, sendo implementado utilizando a classe *PlayerPrefs* da própria *Unity*. Essa informação também fica visível para ser acessada nos menus, através de uma tela de pontuações, por uma função exemplificada em formato de fluxograma na figura 23, onde o sistema verifica se há uma chave "pontos" para ser exibida. Caso não haja, será considerado que o jogador possui zero pontos.

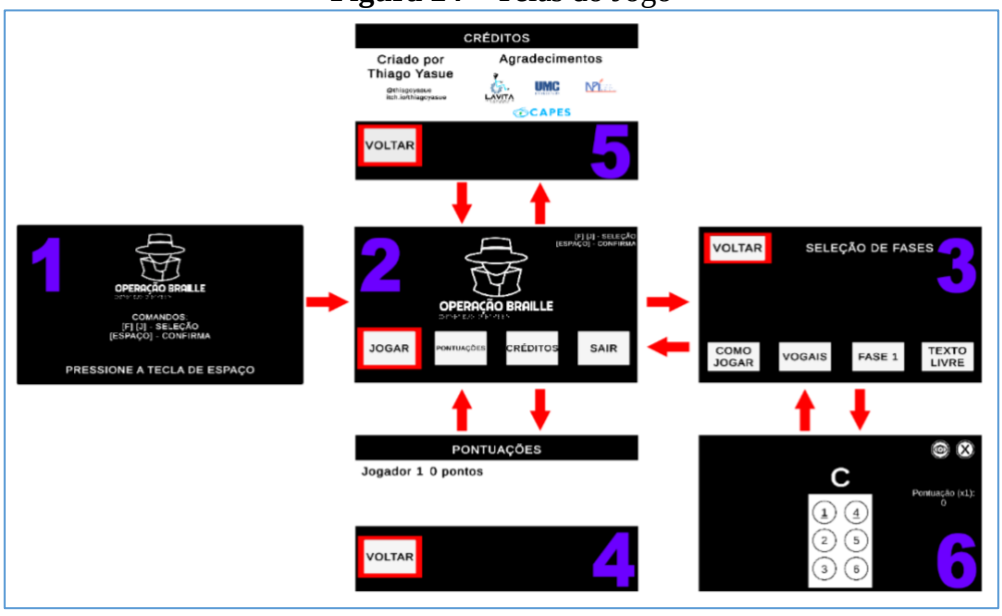
Figura 23 – Fluxograma de Verificação de Pontos



3.3.5 Fluxo de Telas

Foram conceitualizadas seis telas para o jogo. A figura 24 oferece uma visão de como o usuário pode acessá-las, sendo que as setas indicam as próximas telas que o jogador pode ir.

Figura 24 – Telas do Jogo



Enumeradas, em ordem, as telas são: Tela de Pré-Menu, Tela de Menu Principal, Tela de Seleção de Fases, Tela de Pontuações, Tela de Créditos e Tela de *Gameplay*.

3.4 DESIGN

Conforme a explicação anterior, a arte do jogo foi concebida em alto contraste (branco no preto). Além disso, para destacar a posição dos botões nas telas, optou-se por utilizar uma borda vermelha. As próximas imagens apresentam as telas do jogo.

Figura 25 – Tela de Pré-Menu



Figura 26 – Tela de Menu Principal



Figura 27 – Tela de Seleção de Fases



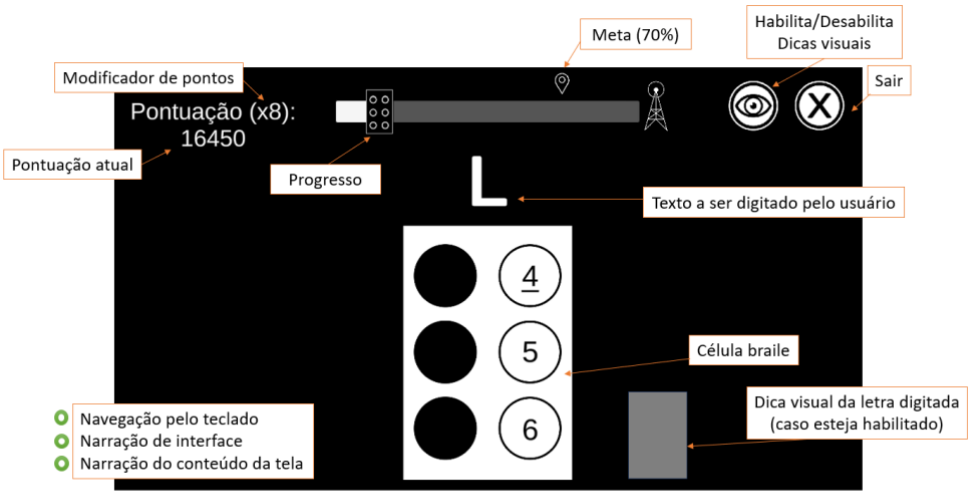
Figura 28– Tela de Pontuações



Figura 29 – Tela de Créditos



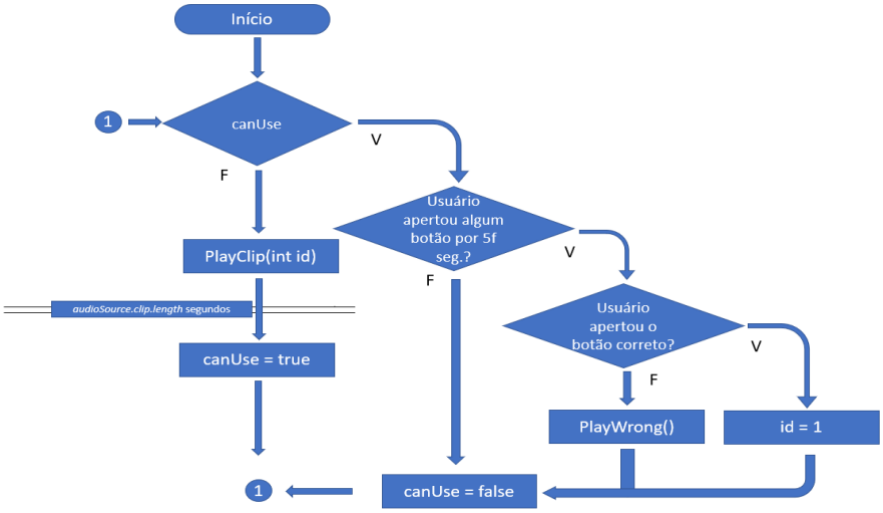
Figura 30 – Tela de *Gameplay*



3.5 ÁUDIO

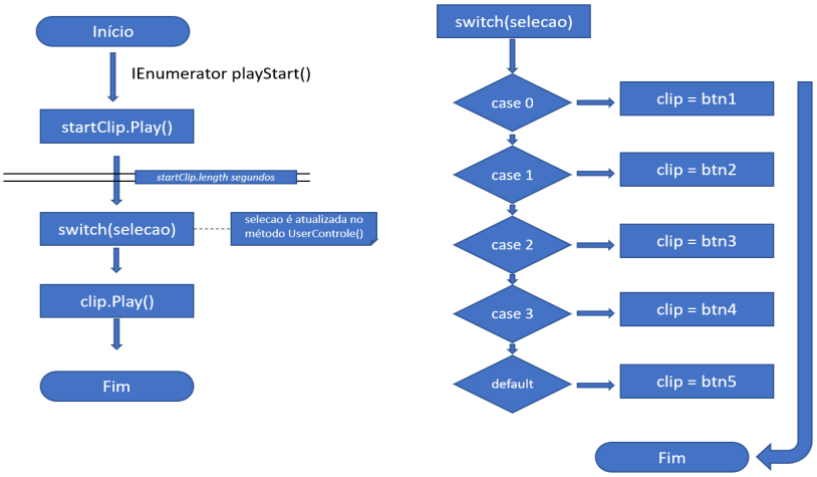
Aliado à programação, o sistema implementado de narração de interface foi concebido utilizando *Courotines* (corrotinas), rotinas capazes de operar cooperativamente (sob demanda), utilizando, especificamente para a aplicação deste jogo, um retorno do tipo *WaitForSeconds*, que dita a quantidade de tempo que a corrotina permanece em pausa, nesse caso, até que o clipe de áudio seja executado por completo (duração de *WaitForSeconds* equivale a *audioSource.clip.length*) sem que o sistema inteiro entre em pausa – por exemplo, o método de entrada de dados, que necessita constante checagem – e em seguida, o próximo áudio é tocado caso haja. Adicionalmente, isso impede que um comando de iniciar um clipe sobreponha outro clipe que ainda possa estar sendo executado, tocando duas faixas ao mesmo tempo. A figura 31 apresenta o funcionamento da narração dentro de jogo durante as fases através de um fluxograma, onde o sistema executa uma corrotina que controla um método que aciona os áudios (o método *PlayClip()* inicialmente com a variável *id = 0)* explicando qual tecla apertar, e caso o jogador erre a tecla ou comandos de tecla pressionados, um áudio dizendo que a digitação está incorreta é executado (o método *PlayWrong()*). Caso acerte, a sequência de áudios avança *(*nesse exemplo, o *id aumenta o estado em mais um, saindo de 0 e indo pra 1)* e caso não faça nada por um certo período, o áudio atual é executado novamente. É importante ressaltar que o usuário só pode digitar quando a *flag canUse* se torna verdadeira, assim como os áudios só serão executados quando *canUse* for falso.

Figura 31 – Funcionamento da Progressão de Clipes de Áudio



A respeito ainda da narração de interfaces, foi implementado outro recurso com o intuito de explicar o botão que está ativo logo quando o usuário acessa a página que contém botões. Por exemplo, a tela de "Menu Principal" possui quatro botões, sendo que, no método de inicialização, o sistema executa dois áudios em sequência: (1) a página onde o jogador está (o jogo diria "Menu Principal") e (2) o botão ativo (o jogo diria "Seleção de Fases"). Nesse caso, como o áudio do botão ativo é executado por último, o jogador pode trocar a seleção do botão, utilizando as teclas F e J, antes do áudio emitir "Seleção de Fases", fazendo-o se confundir com a posição em que se encontra a seleção. Para que essa confusão não aconteça, foi criado um sistema que atualiza, em tempo real, a posição do botão selecionado pelo jogador, como apresentado na figura 32.

Figura 32 – Narração de Interface



Como mostra a figura, após o jogo começar, uma corrotina (*playStart()*) é executada, de modo a executar a primeira faixa de áudio, que narra o nome da página atual, e em seguida, aguarda a execução do áudio por inteiro. Após a pausa, há um *switch case* que analisa qual é a posição (variável *selecao*) do botão que o jogador está e dependendo do botão, uma faixa diferente de áudio é executada. É importante dizer que a variável de posição (variável *selecao*) está sempre sendo atualizada em um método à parte (*UserControle()*), que está funcionando através do método *Update()*, que é chamado a cada *frame.*

3.6 DISPONIBILIDADE

O jogo foi registrado no instituto nacional da propriedade industrial (INPI), número de processo BR512023002433-3. Sua disponibilidade de acesso é restrita, ficando o código-fonte e seus componentes somente a disposição de uso dos autores, em repositório fechado.

3.7 APLICATIVO MÓVEL: PROVA DE CONCEITO

Como mencionado nos objetivos, foi desenvolvido um protótipo apenas para validação da ideia de “o quão difícil é fazer um aplicativo móvel acessível para o ensino de braile?”. Para tanto, foi feita uma adaptação do jogo para *Android*, utilizando o mesmo código-fonte. A figura 33 apresenta imagens das telas que foram exportadas: o menu principal e o tutorial de vogais, respectivamente nessa ordem.

Figura 33 – Telas do aplicativo móvel



Poucas alterações tiveram que ser feitas para a implementação móvel, sendo essas: sistema de navegação de menu por toque, ao invés do uso do teclado e sistema de digitação em braile por toque, ao invés do uso do teclado simulando a máquina de digitar braile.

O sistema de navegação de menu foi feito utilizando dois botões: o botão maior, mais ao centro da tela, serve para mudar a seleção do botão que está sendo pressionado, já o botão inferior serve para confirmar a seleção, fazendo o usuário ir para a tela selecionada.

O sistema de digitação em braile por toque foi fortemente inspirado em todos os aplicativos analisados, onde há um botão em cada ponto da célula de braile, que caso sejam pressionados, retornam uma dica tátil (vibração) e fazem a célula ser pressionada.

Há muitos pontos a serem analisados a respeito deste aplicativo. A primeira, já dita anteriormente, é que através da *Unity*, a portabilidade (ISO/IEC 9126) foi muito simples de ser feita. No entanto, existem dois pontos que devem ser ressaltados sobre a *Unity*, já que ela, primariamente, não é uma IDE focada no desenvolvimento móvel. Esses pontos são, primeiro, a eficiência (ISO/IEC 9216) da exportação do jogo, pois, apesar de jogos para computadores serem aceitáveis terem um tamanho de arquivo alto, que varia entre dezenas até centenas de *gigabytes*, como *Persona 5 Royal* (41 *gigabytes*), *EA SPORTS FC™ 24* (100 *gigabytes)* e *Call of Duty*® (125 *gigabytes*), aplicativos de celular tendem a possuir tamanhos de arquivo bem menores, porém, jogos feitos na *Unity* exportados para dispositivos móveis possuem geralmente grande tamanhos de arquivo: *Pokémon GO* e *Magic: The Gathering Arena* requerem entre 1 a 5 *gigabytes* de armazenamento e até mesmo jogos bidimensionais possuem armazenamento alto, como *Soul Knight* (500 *megabytes)*.

O segundo ponto é um aspecto técnico, que equivale à responsividade (Marcotte, 2010), ou seja, capacidade de resposta ou redimensionamento da estrutura do jogo, pois é notável que a ferramenta de *User Interface* da *Unity* é extremamente limitada: ela funciona bem em um jogo voltado para computadores, onde ela exercerá pouco papel na *gameplay*, porém, em tarefas mais complicadas, a ferramenta é insuficiente em execução e responsividade.

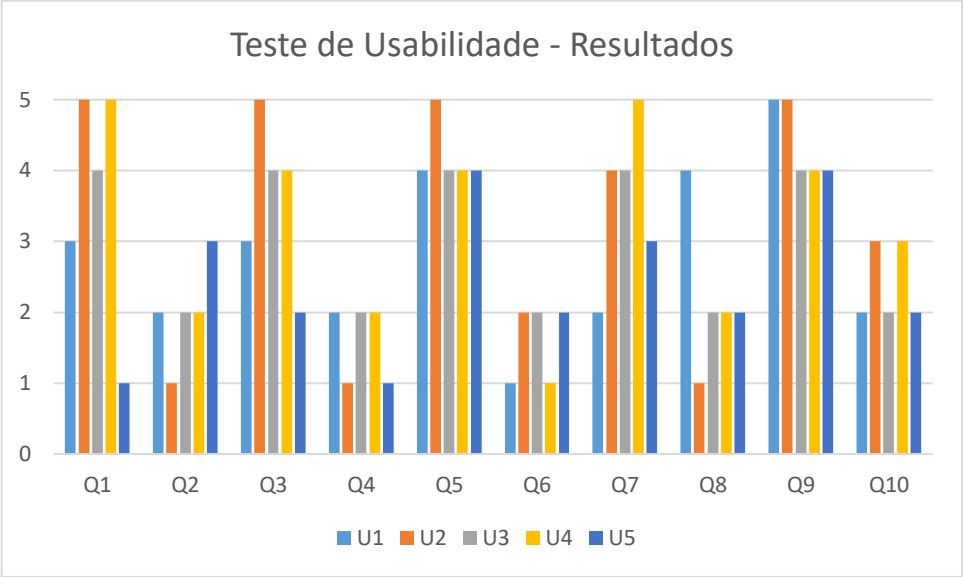
3.8 TESTE DE USABILIDADE

Para a realização do teste, o jogo foi disponibilizado para ser baixado em uma pasta compartilhada no *Google Drive*, consistindo em um arquivo compactado com a *build* do jogo. Além disso, foi disponibilizado um documento escrito no *Notion*, com instruções do teste, apresentando o objetivo, duração, fatores a serem analisados e exemplos de navegação; e logo

abaixo, o questionário, criado no *Google Forms*. O jogo foi testado e o questionário respondido anonimamente.

A figura 34 apresenta os resultados em formato de gráfico, mostrando a pontuação de cada uma das dez questões entre cada um dos cinco usuários que responderam.

Figura 34 – Resultados do teste de usabilidade



Ao analisar os resultados, é importante destacar que o cálculo da pontuação na escala SUS é baseado na intercalação de questões positivas, as ímpares, e questões negativas, as com números pares. Logo, examinando a figura, é possível observar esse padrão acontecendo.

Após calcular os valores para escala SUS, foi alcançada a média de 74 pontos para o teste de usabilidade, classificando o jogo entre “Bom” e “Excelente” na faixa de aceitabilidade “aceitável”.

Ademais, também foi solicitado que os *testers* dessem um *feedback* escrito após a realização de testes. A maioria disse que o jogo cumpre bem a sua função e apesar de ser simples, é funcional, porém, o maior problema relatado foi relacionado ao áudio do jogo, que poderia confundir os jogadores quanto a sua pronúncia, por ser uma voz muito robótica.

4 DISCUSSÃO

Após as pesquisas durante a etapa de escrita da seção de justificativa, ficou constatado que, não apenas softwares com objetivo do ensino à escrita de braile quase não existem, como, os softwares disponíveis ao público, em lojas de aplicativo, estão restritos apenas para uso dos usuários videntes, ou seja, carecendo de funções de acessibilidade.

Após os testes de Operação Braile, foi comprovado que o aplicativo possui boa usabilidade. Contudo, ainda é necessário testar o jogo para comprovar sua eficácia no aprendizado de braile.

Depois de desenvolver a prova de conceito do aplicativo móvel, foi possível inferir que criar aplicativos móveis acessíveis para o ensino de braile para pessoas não videntes é possível, porém não há grande interesse no desenvolvimento.

Quanto ao resultado do desenvolvimento de Operação Braille, foi possível solucionar vários dos problemas mencionados no estado da arte, a respeito dos aplicativos que já existem no mercado, o principal sendo a respeito da falta de dicas sonoras, já que foram adicionados efeitos sonoros e narrações em todo o jogo, tornando-o acessível para usuários não videntes. Ademais, o jogo também simula a digitação como uma máquina de escrever braile, sendo esse um recurso útil para facilitar a navegação e digitação – sem a necessidade de utilizar o mouse ou algum controle especial, por exemplo.

5 CONCLUSÃO

O jogo foi desenvolvido durante o tempo proposto com as mecânicas previstas: foi criado um sistema de digitação em braile, baseado nas entradas do teclado pelo usuário. Além disso, os elementos visuais do jogo em alto contraste foram criados e estão dispostos em todas as telas ao longo da *gameplay*. A respeito das telas do jogo, estas incluem o recurso de narração previsto anteriormente, tanto a narração de menus e botões, quanto a narração durante a *gameplay*. Por fim, foi realizada uma validação do jogo a partir da avaliação dos especialistas.

Ademais, é importante ressaltar que futuras alterações do projeto serão realizadas e futuros novos testes, principalmente no que tange testes com deficientes visuais, pois, após a coleta dos *feedbacks*, foi possível constatar que a opinião do testador deficiente visual proporcionou um panorama muito diferente dos outros *testers*, onde ele havia sugerido recursos que poderiam tornar o jogo mais intuitivo, além de recursos extras para personalização do jogo.

5.1 LIMITAÇÕES

Como limitação do projeto, é possível dizer que o jogo não possui nenhum tipo de dica tátil, sendo útil somente para a escrita de braile, e não a leitura. Também é importante enfatizar que o jogo ainda não foi testado conforme sua eficácia no aprendizado do ensino de braile.

Por fim, cabe ressaltar que antes de tudo, o objetivo principal deste projeto é de diminuir a dificuldade de acesso ao aprendizado de braile, possibilitando à pessoa cega e com deficiência visual independência e interação com a sociedade.

REFERÊNCIAS

APPLE, **iPad – Apple (BR).** c2024. Disponível em: https://www.apple.com/br/ipad/. Acesso em: 11 mar. 2024.

APPLE, **iPhone SE – Apple (BR)**. c2024. Disponível em: https://www.apple.com/br/iphone-se/. Acesso em: 11 mar. 2024.

ASSOCIAÇÃO BRASILEIRA DE NORMAS TÉCNICAS. **NBR9241-11**: Requisitos Ergonômicos para Trabalho de Escritórios com Computadores: Orientações sobre Usabilidade. Rio de Janeiro, RJ: ABNT, 2002.

INSTITUTO BRASILEIRO DE GEOGRAFIA E ESTATÍSTICA. **Censo Demográfico 2010**: características gerais da população, religião e pessoas com deficiência. Rio de Janeiro: IBGE, 2010. *E-book.* Disponível em: https://biblioteca.ibge.gov.br/visualizacao/periodicos/94/cd_2010_religiao_deficiencia.pdf. Acesso em: 07 fev. 2023.

BRASIL. **Lei nº 13.146, de 6 de julho de 2015**. Institui a Lei Brasileira de Inclusão da Pessoa com Deficiência (Estatuto da Pessoa com Deficiência). Brasília, DF: Presidência da República, 2015. Disponível em: https://www.planalto.gov.br/ccivil_03/_ato2015-2018/2015/lei/l13146.htm. Acesso em: 16 jan. 2023.

BRASIL. Ministério da Educação. Secretaria de Educação Continuada, Alfabetização, Diversidade e Inclusão. Diretoria de Políticas de Educação Especial. **Grafia Braille para a Língua Portuguesa**. Brasília, DF: MEC, 2018. *E-book*. Disponível em: http://portal.mec.gov.br/docman/dezembro-2018-pdf/104041-anexo-grafia-braille-para-lingua-portguesa/file. Acesso em: 19 mar. 2023.

BOOCH, Grady; RUMBAUGH James, JACOBSON, Ivar. **UML Guia do Usuário**. 6ª Edição. Rio de Janeiro: Elsevier, 2005.

BROOKE, John. SUS: A Restrospective. **Journal of Usability Studies**, v.8, n.2, p. 29-40, feb. 2013.

CHUNG, L.; NIXON, B.; YU, E.; MYLOPOULOS, J. **Non-Functional Requirements in Software Engineering**. Boston, Dordrecht, London: Kluwer Academic Publishers, 2000.

DONG DIGITAL. **Braille Academy**: Play & Learn. c2021-2023. Disponível em: https://apps.apple.com/br/app/braille-academy-play learn/id1561530189 . Acesso em: 19 mar. 2023.

DOWINO. **A Blind Legend**. 2016. Disponível em: https://store.steampowered.com/app/437530/A_Blind_Legend. Acesso em: 29 jul. 2022.

DREZZA, Érika Rack. **O Sistema Braille**. 2022. São Paulo, SP. Fundação Dorina, 2018. *E-book.* Disponível em: https://trocandosaberes.com.br/wp-content/uploads/2019/02/Cartilha-O-sistema-braille.pdf. Acesso em: 24 fev. 2023.

FREEDOM SCIENTIFIC. **Braille displays & notetakers**. c2024. Disponível em: https://store.freedomscientific.com/collections/braille-display-and-notetakers. Acesso em: 09 ago. 2023.

GUERREIRO, Patrícia. **Instituto Benjamin Constant**: 150 anos. Rio de Janeiro: Instituto Benjamin Constant, 2007.

GERTZ, Martin. **Braille Trainer**. 2019. Disponível em: https://apps.apple.com/br/app/braille-trainer/id1464977255?platform=iphone. Acesso em: 19 mar. 2023.

HATZIGIANNAKOGLOU, Paul D.; KAMPOURAKI, Maria T., "Learn Braille": A Serious Game Mobile App for sighted Braille Learners. **Journal of Engineering Science and Technology Review**, v.9, n.1, p. 174-176, abr. 2016.

HILDEBRANDT, Antonio Carlos; COUTO, Januário Pereira do; GONÇALVES, José Francisco; FREITAS, José Maria de; ROSA, João Bosco Dias Santa; FONTES, João Loures; SANT'ANNA, Laércio; SPELTA, Lêda Lucia; PENIDO, Márcio Neves. **Grafia braille para informática**, 2005. Brasília: MEC; SEESP, 2004.

INTERVOX. **Braille Fácil 4.0**. Rio de janeiro: NCE-UFRJ. c2022. (Programa Ministério da Educação Boa Escola para Todos). Disponível em: http://intervox.nce.ufrj.br/brfacil/. Acesso em: 19 mar. 2023.

KAWAKAMI, Takuya. **Braille Study Lite**. 2014. Disponível em: https://apps.apple.com/br/app/braille-study-lite/id881345749. Acesso em: 19 mar. 2023.

MARCOTTE, Ethan. **Responsive Web Design**. 2010. Disponível em: https://alistapart.com/article/responsive-web-design/. Acesso em: 15 out. 2023.

MICROSOFT. **Conceitos básicos da acessibilidade de jogos**. c2024. Disponível em: https://docs.microsoft.com/pt-br/training/paths/gaming-accessibility-fundamentals/. Acesso em: 15 set. 2022.

MILNE, Lauren R.; BENNETT, Cynthia L.; LADNER, Richard E.; AZENKOT, Shiri. **BraillePlay: Educational Smartphone Games for Blind Children**. In: ASSETS '14: THE 16TH INTERNATIONAL ACM SIGACCESS CONFERENCE ON COMPUTERS AND ACCESSIBILITY, 16., 2014, Rochester, Nova York, Estados Unidos. **ASSETS'14: Proceedings of The 16th International ACM SIGACCESS Conference on Computers and Accessibility,** Nova York, NY: Association for Computing Machinery, 2014. p. 137-144.

MOSQUERA, Carlos Fernando França. **Deficiência Visual na Escola Inclusiva**. Curitiba: Editora Intersaberes, 2012.

NEEDHAM, L. **BRAILLIAC:** BRAILLE TUTOR. 2018. Disponível em: https://brailliac.com. Acesso em: 29 jul. 2022.

PERKINS. **SMART Brailler from Perkins**. c2024. Disponível em: https://brailler.perkins.org/pages/smart-brailler. Acesso em: 11 mar. 2024.

POOLTRUB, Apinya**. Brailler fightrer Braille typing training game learning Society**. 2019. Disponível em: https://www.rmutl.ac.th/news/11118-brailler-fighter-braille-typing-training-game. Acesso em: 11 mar. 2024**.**

R&A. **Touch Braille**. 2017. Disponível em: https://apps.apple.com/br/app/touch-braille/id1199928789. Acesso em: 19 mar. 2023.

ROYCE, Winston W. **Managing the development of large software systems: concepts and techniques**. In: 9TH ANNUAL INTERNATIONAL CONFERENCE ON SOFTWARE ENGINEERING, 9., 1987, Monterey, Califórnia, Estados Unidos**. ICSE '87: Proceedings of the [...].** Washington, DC: IEEE Computer Society Press, 1987. p. 328 - 338

SANTANA, Kayo; PEREIRA, Claudia Pinto; SANTANA, Beatriz Silva de; **Braillearning: Software para simular a máquina de escrever em Braille.** In: XXX SIMPÓSIO BRASILEIRO DE INFORMÁTICA NA EDUCAÇÃO (BRAZILIAN SYMPOSIUM ON COMPUTERS IN EDUCATION), 30., 2019, Brasília. **Anais [...]**. Brasília, DF: SBIE, 2019. p. 1101-1110.

SOMMERVILLE, Ian. **Engenharia de software**. São Paulo: Pearson, 2007.

SONNAR INTERACTIVE LTD. **Audio Game Hub 2**. 2016. Disponível em: https://play.google.com/store/apps/details?id=com.AUT.AudioGameHub. Acesso em: 29 jul. 2022.

TEAM ZEROTH. **Breu:** Ataque das Sombras. 2023. Disponível em: https://store.steampowered.com/app/2213890/Breu_Ataque_das_Sombras/?l=brazilian. Acesso em: 09 ago. 2023.

TECHOPOLIS ONLINE SOLUTIONS, LLC. **PocketBraille Reference Lite**. 2018. Disponível em: https://apps.apple.com/us/app/pocketbraille-reference-lite/id1371786238?platform=iphone. Acesso em: 09 ago. 2023.

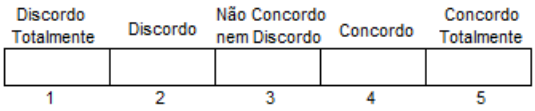
UGEDO, M. **LEARN BRAILLE**. 2020. Disponível em: https://play.google.com/store/apps/details?id=com.comoelagua.android.braille. Acesso em: 29 jul. 2022.

WHO. **Blindness and vision impairment**. 2023. Disponível em: https://www.who.int/news-room/fact-sheets/detail/blindness-and-visual-impairment. Acesso em: 20 ago. 2023.

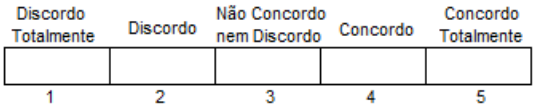
APÊNDICE A – Questionário de Usabilidade

Questionário de Percepção da Usabilidade – SUS (System Usability Scale)

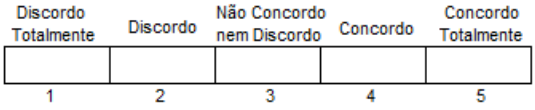
1. Se eu precisasse, eu acho que eu usaria este jogo com frequência.



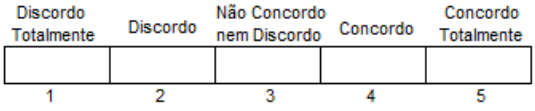
2. Eu achei as funcionalidades do jogo desnecessariamente complexas.



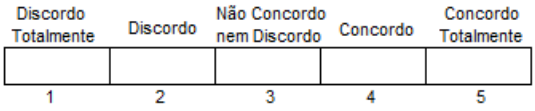
3. Eu acho que este jogo é fácil de usar.



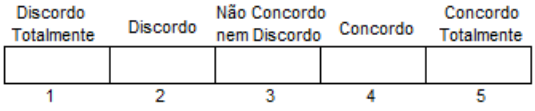
4. Eu acho que eu precisaria de suporte técnico para poder usar este jogo frequentemente.



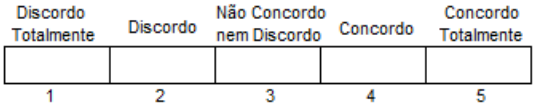
5. Eu acho que eu precisaria de suporte técnico para poder usar este jogo frequentemente.



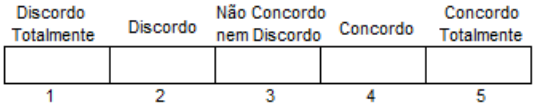
6. Eu acho que eu precisaria de suporte técnico para poder usar este jogo frequentemente.



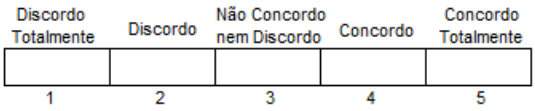
7. Eu acho que eu precisaria de suporte técnico para poder usar este jogo frequentemente.



8. Eu senti que inicialmente, precisaria da ajuda de um terceiro capacitado para me instruir a utilizar as funções básicas do jogo.



9. Eu não me senti perdido ao navegar pelo jogo.



10. Eu sinto que precisaria aprender muitas coisas fora do jogo antes de eu poder jogá-lo.

