



Curso de Análise e Desenvolvimento de Sistemas

Cauã Viana

Filipe Affonso

Documentação de Desenvolvimento de Software Logic Shuffle

Sorocaba
Novembro - 2025



**Cauã Viana
Filipe Affonso**

Documentação de Desenvolvimento de Software Logic Shuffle

Trabalho de Graduação apresentado à Faculdade de Tecnologia de Sorocaba, como parte dos pré-requisitos para obtenção do título de Tecnólogo em Análise e Desenvolvimento de Sistemas.

Orientadora: Prof^a M^a Denilce de A. O. Veloso

Sorocaba
Novembro - 2025

Dedicatória

Dedicamos este trabalho aos nossos professores, pela transferência inestimável de conhecimento e pela orientação essencial ao longo de nossa formação. Aos nossos familiares, pelo seu amor incondicional, pela perseverança e pelo apoio constante, que serviram de alicerce para a conclusão desta jornada. E, por fim, aos nossos amigos, que tornaram a caminhada mais leve e nos incentivaram em todas as etapas da realização deste projeto.

Agradecimentos

A concretização deste Trabalho de Graduação é resultado do esforço coletivo e do apoio fundamental de diversas pessoas e instituições aos quais expressamos nossa sincera gratidão.

Em primeiro lugar, somos gratos à nossa orientadora, Professora M^a Denilce de A. O. Veloso, e aos professores Antônio Cesar Munari e Maria Angélica Cardieri, cuja dedicação, paciência e valiosa experiência guiaram o desenvolvimento desta pesquisa e aprimoraram a qualidade do nosso trabalho. A precisão de suas orientações foi essencial para que superássemos os desafios.

Agradecemos também ao Ítalo Barros Pereira da SuperGeeks, pelas valiosas dicas no desenvolvimento do projeto.

Aos nossos familiares, especialmente aos nossos pais, por serem a nossa maior bênção, pelo amor incondicional, dedicação e incentivo infinito, que nos deram o suporte emocional e a infraestrutura necessária ao longo de toda a graduação.

Aos nossos amigos e colegas de curso, em especial ao Adriano Bertanha, que colaboraram de diversas formas na elaboração deste trabalho, compartilhando ideias e oferecendo um ambiente de apoio mútuo.

Por fim, agradecemos a todas as pessoas que, direta ou indiretamente, contribuíram para a realização deste projeto, fornecendo informações e colaborando com sua dedicação para o êxito do trabalho. O reconhecimento de vocês é parte integrante desta conquista.

Resumo

O ensino da disciplina de Estruturas de Dados apresenta um elevado grau de dificuldade entre estudantes de Computação, devido à necessidade de alta abstração lógica e à visualização dos conceitos. Diante desse problema, este trabalho teve como objetivo o desenvolvimento do Logic Shuffle, um aplicativo educacional composto por minijogos voltados à aprendizagem de árvores, pilhas, filas e deque. A elicitação de requisitos iniciou-se por meio da aplicação de um formulário, cujo objetivo foi identificar dificuldades recorrentes e preferências quanto a recursos visuais e dinâmicos de aprendizagem. Com base nesses dados, e na pesquisa de outras ferramentas do mercado, definiu-se o escopo e as funcionalidades do sistema, que foi desenvolvido utilizando a ferramenta Godot Engine na interface cliente, Python (Flask) no back-end e PostgreSQL como banco de dados, empregando uma arquitetura em camadas e comunicação via API REST. Foi utilizada a linguagem UML (*Unified Modeling Language*) para elaboração da documentação. A análise comparativa entre *game engines* embasou a escolha pela Godot, devido ao seu desempenho em projetos 2D e ao custo-benefício favorável. Os resultados obtidos demonstram que o Logic Shuffle atende aos objetivos propostos, apresentando potencial para complementar o ensino tradicional de Estruturas de Dados por meio de visualizações dinâmicas e interatividade. Conclui-se que o protótipo é funcional e que, em versões futuras, poderão ser adicionados novos jogos, níveis e mecânicas para aprimorar a eficácia pedagógica do aplicativo.

Palavras Chaves: Estruturas de Dados; Gamificação; Godot.

Lista de Figuras

Figura 1. Lista encadeada	16
Figura 2. Funcionamento da estrutura de uma pilha	18
Figura 3. Funcionamento da estrutura de uma fila.....	19
Figura 4. Estrutura de uma árvore e seus conceitos básicos	20
Figura 5. Estrutura de uma Hash Table	21
Figura 6 - Formação predominante dos pesquisados	28
Figura 7 – Curso predominante dos pesquisados	28
Figura 8 - Contato prévio com a disciplina Estruturas de Dados.....	29
Figura 9 – Estruturas de dados consideradas mais difíceis.....	30
Figura 10 - Grau de dificuldade percebido na disciplina (escala 1–5).....	30
Figura 11 - Avaliação da eficácia dos métodos de ensino tradicionais	31
Figura 12 - Principais dificuldades relatadas	32
Figura 13 - Experiência prévia com jogos e aplicativos educacionais	32
Figura 14 - Concordância sobre a utilidade de minigames no aprendizado	33
Figura 15 - Recursos interativos mais desejados	34
Figura 16 - Dispositivos preferidos para acesso à aplicação	34
Figura 17 – Dispositivos preferidos para acesso à aplicação.....	35
Figura 18 – Estrutura mais necessária de visualização prática	36
Figura 19 - LootCode	37
Figura 20 – Tela de jogo do LootCode.....	38
Figura 21 - The Farmer Was Replaced.....	39
Figura 22 - Hour of Code.....	40
Figura 23 – Diagrama de Caso Uso.....	48
Figura 24 - Arquitetura do Software	53
Figura 25 – Modelo Conceitual.....	58
Figura 26 – Modelo Lógico	59
Figura 27 – Diagrama de atividades do processo de autenticação do usuário.....	60
Figura 28 – Diagrama de atividades do fluxo pós-autenticação	61
Figura 29 – Tela de Login.....	62
Figura 30 – Tela de Cadastro	63
Figura 31 – Tela de Recuperação de Senha	64
Figura 32 – Tela Inicial.....	65
Figura 33 – Tela de Edição de Perfil	66
Figura 34 – Seletor de Avatar	67
Figura 35 – Tela de Ranking Global	68
Figura 36 – Tela de Configurações	69
Figura 37 – Tela Sobre o Jogo	70
Figura 38 – Tela de Seleção de Jogo	71
Figura 39 – Tela de Seleção de Fases (Dungeon Cave Quest)	72
Figura 40 – Tela de Introdução (Nível 1)	73
Figura 41 – Interface de Jogo (Nível 1).....	74
Figura 42 – Janela de Seleção de Elementos	75
Figura 43 – Janela de Nível Concluído	76
Figura 44 – Tela de Progressão de Fases.....	77
Figura 45 – Ranking Global com Posição do Usuário	78
Figura 46 – Janela de Verificação de Erro.....	79
Figura 47 – Janela de Confirmação para Mostrar Solução	80
Figura 48 – Tela de Seleção de Fases (Todos os Níveis Concluídos)	81
Figura 49 – Tela de Seleção de Fases (Pyramid Maker)	82
Figura 50 – Tela de Introdução (Pyramid Maker Nível 1)	83
Figura 51 – Interface de Jogo (Pyramid Maker Nível 1).....	84
Figura 52 – Exemplo de Construção da Árvore (Pyramid Maker)	85
Figura 53 – Tela de Progressão de Fases (Pyramid Maker)	86

Figura 54 – Janela de Verificação de Erro (Pyramid Maker)	87
Figura 55 – Janela de Confirmação para Mostrar Solução (Pyramid Maker)	88
Figura B.1 - Tela de Introdução (Nível 2)	108
Figura B.2 - Interface de jogo (Nível 2)	109
Figura B.3 - Tela de Introdução (Nível 3)	109
Figura B.4 - Interface de jogo (Nível 3)	110
Figura B.5 - Tela de Introdução (Nível 4)	110
Figura B.6 - Interface de jogo (Nível 4)	111
Figura B.7 - Tela de Introdução (Nível 5)	111
Figura B.8 - Interface de jogo (Nível 5)	112
Figura B.9 - Tela de Introdução (Nível 6)	112
Figura B.10 - Interface de jogo (Nível 6)	113
Figura B.11 - Tela de Introdução (Nível 7)	113
Figura B.12 - Interface de jogo (Nível 7)	114
Figura B.13 - Tela de Introdução (Nível 8)	114
Figura B.14 - Interface de jogo (Nível 8)	115
Figura B.15 - Tela de Introdução (Nível 9)	115
Figura B.16 - Interface de jogo (Nível 9)	116
Figura B.17 - Tela de Introdução (Nível 10)	116
Figura B.18 - Interface de jogo (Nível 10)	117
Figura C.1 - Tela de Introdução (Nível 2)	118
Figura C.2 - Interface de jogo (Nível 2)	118
Figura C.3 - Tela de Introdução (Nível 3)	119
Figura C.4 - Interface de jogo (Nível 3)	119
Figura C.5 - Tela de Introdução (Nível 4)	120
Figura C.6 - Interface de jogo (Nível 4 Resolvido)	120
Figura C.7 - Tela de Introdução (Nível 5)	121
Figura C.8 - Interface de jogo (Nível 5)	121
Figura C.9 - Tela de Introdução (Nível 6)	122
Figura C.10 - Interface de jogo (Nível 6)	122
Figura C.11 - Tela de Introdução (Nível 7)	123
Figura C.12 - Interface de jogo (Nível 7 Resolvido)	123
Figura C.13 - Tela de Introdução (Nível 8)	124
Figura C.14 - Interface de jogo (Nível 8)	124
Figura C.15 - Tela de Introdução (Nível 9)	125
Figura C.16 - Interface de jogo (Nível 9 Resolvido)	125
Figura C.17 - Tela de Introdução (Nível 10)	126
Figura C.18 - Interface de jogo (Nível 10)	126

Lista de Tabelas

Tabela 1: Custos de Infraestrutura.....	91
Tabela 2: Custos de Desenvolvimento	92

Lista de Quadros

Quadro 1: Trabalhos relacionados.....	24
Quadro 2: Comparação entre Games Engines	43
Quadro 3. Caso de uso – Cadastro	49
Quadro 4. Caso de uso – Login.....	49
Quadro 5. Caso de uso – Logout.....	49
Quadro 6. Caso de uso – Editar Perfil	50
Quadro 7. Caso de uso – Consultar Pontuação	50
Quadro 8. Caso de uso – Alterar Configurações.....	50
Quadro 9. Caso de uso – Selecionar Jogo.....	51
Quadro 10. Caso de uso – Selecionar Fase.....	51

Índice

1.	Introdução.....	12
2.	Embasamento teórico.....	14
2.1	Histórico.....	14
2.2	Conceito de Estrutura de Dados.....	14
2.3	Tipos de Estruturas de Dados.....	15
2.3.1	- Listas	16
2.3.2	- Pilhas	17
2.3.3	- Filas	18
2.3.4	- Árvores	19
2.3.5	- Hash Tables (tabelas de espalhamento)	20
2.4	Teorias de Aprendizagem e Gamificação	22
2.5	Trabalhos relacionados	22
3.	Análise de Requisitos	25
3.1	Visão geral do Produto	25
3.2	Descrição da técnica utilizada para levantamento dos requisitos	26
3.2.1	Levantamento e Análise dos Resultados do Questionário.....	27
3.2.2	- Ferramentas encontradas no mercado	36
3.2.3	- Comparação entre Game Engines.....	41
3.4	Requisitos de Software	44
3.4.1	Requisitos Funcionais.....	44
3.4.2	Requisitos Não Funcionais	45
3.4.3	Diagrama de Casos de Uso e Descrição dos Casos de Uso	48
4.	Projeto Detalhado do Software.....	52
4.1	Arquitetura da aplicação	52
4.2	Tecnologias utilizadas e APIs	53
4.2.1	Godot Engine	53
4.2.2	Python + Flask	54
4.2.3	PostgreSQL	54
4.2.4	SendGrid	54
4.2.5	SQLAlchemy (ORM)	55
4.2.6	Flask-JWT-Extended	55
4.2.7	Passlib	55
4.2.8	Python-dotenv	56
4.2.9	UUID	56
4.2.10	Hashlib.....	56
4.2.11	Godot ConfigFile.....	56
4.2.12	Godot Audio Engine.....	57

4.3 Modelo de dados	58
4.3.1 Modelo Conceitual	58
4.3.1 Modelo Lógico	59
4.4 Diagrama de Atividades	59
4.5 Interfaces com o usuário	62
5. Implantação	89
5.1 Instalação e repositórios	89
5.2 Custos	91
5.3 Testes e Validação da Aplicação.....	92
6. Conclusão.....	94
Referências.....	96
Glossário	100
Apêndice A – Cópia do Questionário.....	103
Apêndice B – Níveis subsequentes Dungeon Cave Quest.....	108
Apêndice C – Níveis subsequentes Pyramid Maker	118

1. Introdução

No contexto do ensino de Computação, especialmente nas disciplinas de Estruturas de Dados, observam-se dificuldades recorrentes entre os estudantes, resultantes da necessidade de articular raciocínio lógico e visualização abstrata. Zingaro et al. (2018) apontaram que alunos apresentam obstáculos significativos na compreensão dos conceitos fundamentais de pilhas, filas e listas encadeadas, evidenciando lacunas na assimilação conceitual e na transposição entre teoria e prática.

Diante dessas dificuldades, este trabalho de graduação apresenta o desenvolvimento do Logic Shuffle, um aplicativo educacional voltado ao ensino de Estruturas de Dados por meio de minijogos que simulam o funcionamento de árvores, pilhas, filas e deque.

A proposta integrou elementos lúdicos e objetivos pedagógicos, com o intuito de unir aprendizado teórico e prática interativa. O sistema incluiu funcionalidades voltadas ao acompanhamento individual do progresso dos usuários, sem necessidade de intermediação docente direta.

A motivação do projeto reforça esse diagnóstico, derivando das dificuldades observadas entre estudantes na assimilação dos conceitos das Estruturas de Dados por meio de métodos expositivos tradicionais. O avanço do ensino remoto e híbrido intensificou a demanda por recursos digitais que complementassem o conteúdo formal, reforçando a pertinência do desenvolvimento de ferramentas educacionais gamificadas (COOK et al., 2023).

Sendo assim, o objetivo geral consistiu em desenvolver um jogo educacional que auxiliasse o ensino e a aprendizagem de Estruturas de Dados por meio de recursos visuais e exercícios práticos. Como objetivos específicos, destacaram-se: (i) pesquisar e selecionar as tecnologias adequadas ao desenvolvimento do sistema; (ii) modelar os requisitos funcionais e não funcionais; (iii) implementar módulos interativos e minijogos educativos; e (iv) validar a usabilidade e a eficácia do sistema com usuários.

Para atingir esses objetivos, a metodologia adotada envolveu o levantamento de dados com potenciais usuários, trabalhos acadêmicos relacionados, pesquisa de aplicações existentes no mercado, o planejamento e o desenvolvimento iterativo da aplicação.

O sistema foi implementado utilizando Godot Engine para o desenvolvimento dos minijogos e da interface gráfica, Python (Flask) para o back-end e PostgreSQL como banco de dados. A arquitetura adotada seguiu o padrão em camadas, com comunicação via API REST, permitindo integração entre cliente e servidor.

O trabalho foi estruturado em seis capítulos. O Capítulo 2 apresenta o embasamento teórico, abordando conceitos de Estruturas de Dados, fundamentos da gamificação e estudos relacionados a jogos educacionais. O Capítulo 3 detalha a análise de requisitos e o projeto de software, incluindo a modelagem de casos de uso e o desenho da arquitetura do sistema. O Capítulo 4 descreve as tecnologias utilizadas no desenvolvimento e os diagramas técnicos que representaram a lógica de funcionamento dos módulos. O Capítulo 5 trata das etapas de implementação, levantamento de custos e também teste de validação. Por fim, o Capítulo 6 apresenta as conclusões obtidas e as perspectivas de aprimoramento da solução proposta.

Dessa forma, o trabalho pretende contribuir para a área de tecnologia educacional, propondo uma ferramenta inovadora de apoio ao ensino-aprendizagem de Estruturas de Dados, capaz de promover maior interatividade, engajamento e eficácia no processo educacional.

2. Embasamento teórico

2.1 Histórico

O avanço da tecnologia da informação, intensificado a partir da segunda metade do século XX e acelerado com a expansão da internet no início dos anos 2000, transformou profundamente a maneira como o conhecimento é produzido e compartilhado. Segundo Castells (1999), vivemos atualmente em uma sociedade em rede, na qual o acesso à informação se tornou elemento essencial para a inserção social, educacional e profissional.

No cenário contemporâneo, o mercado não se restringe apenas à adoção de tecnologias, mas exige a compreensão crítica e a aplicação sistemática dessas ferramentas em diferentes cenários. Nesse contexto, o ensino de disciplinas ligadas à Computação, como Estruturas de Dados, ganhou relevância estratégica, uma vez que o domínio desses conceitos é fundamental para a resolução de problemas, otimização dos recursos computacionais e o desenvolvimento de soluções inovadoras.

Assim, observa-se que a formação sólida em fundamentos computacionais, associada a práticas pedagógicas que favoreçam a aplicabilidade do conhecimento, constitui um diferencial relevante na formação de profissionais aptos a atuar em um cenário de intensas mudanças tecnológicas. Tal perspectiva evidencia a necessidade de integrar a dimensão teórica e prática no processo de ensino-aprendizagem, de modo a potencializar a formação crítica e inovadora dos estudantes da área de Computação.

2.2. Conceito de Estrutura de Dados

O conceito de Estrutura de Dados pode ser facilmente definido como uma forma organizada de armazenar, manipular e gerenciar informações em sistemas computacionais, a fim de garantir o acesso a algumas operações (inserção, remoção, busca e atualização), que são essenciais para estruturar os dados no computador na memória de modo a otimizar o processamento e a utilização de recursos.

Do ponto de vista conceitual, as estruturas de dados são classificadas em estruturas estáticas, como vetores e matrizes, e estruturas dinâmicas, como listas encadeadas, pilhas, filas e árvores. Enquanto as primeiras possuem tamanho fixo definido em tempo de compilação, as segundas oferecem maior flexibilidade ao se adaptarem à necessidade do programa durante sua execução (Cormen et al., 2009).

A importância do estudo de estruturas de dados reside no fato de que elas constituem a base para a construção de algoritmos eficientes e soluções escaláveis. Assim, compreender suas propriedades e aplicações é essencial para qualquer profissional da área de Computação, uma vez que o desempenho de softwares, sistemas e aplicações está diretamente relacionado à escolha adequada da estrutura utilizada para representar as informações (Knuth, 1997).

De modo mais complexo, pode-se dizer que estrutura de dados se refere a um modelo abstrato que define não apenas a forma de organização e armazenamento da informação, mas também o conjunto de operações válidas que podem ser realizadas sobre ela. Nesse sentido, sua definição envolve tanto aspectos lógicos relacionados à modelagem e à representação matemática dos dados, quanto aspectos físicos, associados à forma como esses dados são efetivamente implementados e manipulados na memória. Essa dualidade é fundamental, pois permite estabelecer um elo entre a teoria dos algoritmos e a prática da programação, fornecendo a base conceitual para o desenvolvimento de soluções computacionais eficientes e robustas.

2.3 Tipos de Estruturas de Dados

Conceitos como listas, pilhas, filas, árvores e tabelas de espalhamento são a base para o desenvolvimento de sistemas eficientes, bem como uma compreensão mínima necessária para atuar em partes importantes. Contudo, ainda representam um dos grandes desafios pedagógicos no ensino superior, devido ao nível de abstração exigido e à dificuldade de visualizar o comportamento dinâmico dessas estruturas.

2.3.1 - Listas

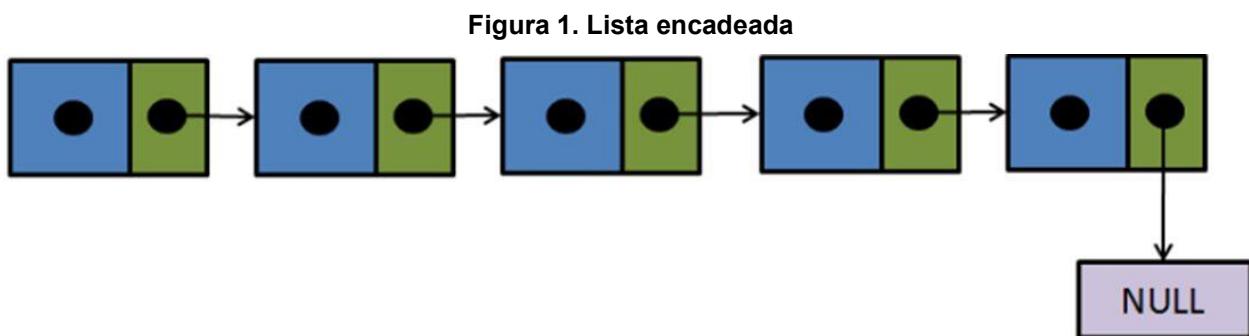
As listas constituem coleções lineares de elementos que podem ser organizadas de forma contígua (vetores) ou encadeada, permitindo inserções e remoções em diferentes posições, o último elemento aponta para o nulo, para indicar o seu fim.

Enquanto as listas contíguas oferecem acesso direto a qualquer posição por meio de índices, apresentam custo elevado para inserções e remoções em posições intermediárias, devido à necessidade de deslocamento dos elementos. Por outro lado, as listas encadeadas reduzem esse custo ao possibilitar a alteração de ponteiros, mas sacrificam a eficiência no acesso aleatório, que passa a depender da navegação sequencial (Cormen et al., 2009).

A aplicação das listas se estende a diversos contextos, como implementação de editores de texto, gerência de memória em sistemas operacionais e gerenciamento de dados em tempo de execução, tornando seu estudo essencial para a formação de profissionais de Computação.

Dessa forma, compreender as listas permite ao estudante desenvolver habilidades de organização e manipulação de dados, consolidando conceitos que serão utilizados no estudo de estruturas subsequentes e no desenvolvimento de algoritmos eficientes.

A figura 1 apresenta um exemplo de lista encadeada.



Fonte: Adaptado de (GRAN CURSOS ONLINE, 2025)

2.3.2 - Pilhas

As pilhas (*LIFO – Last In, First Out*) seguem a disciplina de acesso em que o último elemento inserido é o primeiro a ser removido, sendo amplamente aplicadas em algoritmos de retrocesso, chamadas de funções recursivas e controle de execução em compiladores (Cormen et al., 2009).

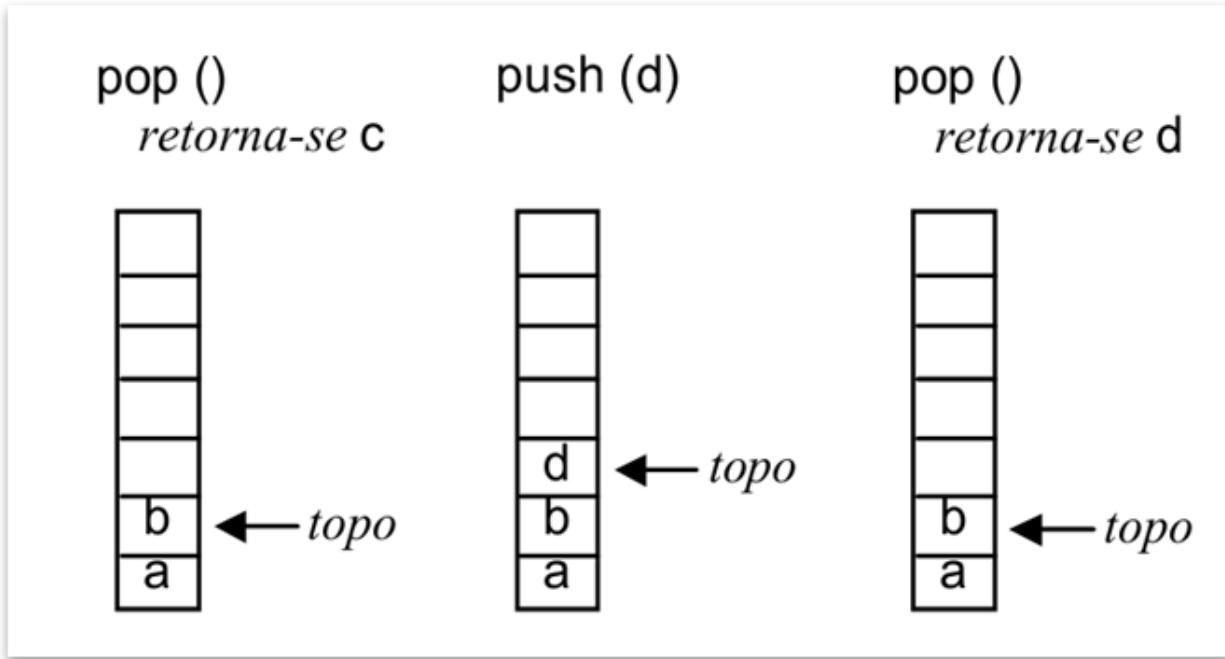
Sua implementação pode ser realizada tanto por meio de listas contíguas quanto encadeadas, oferecendo diferentes vantagens quanto a acesso, inserção e remoção de elementos. O conhecimento dessas alternativas é fundamental para o desenvolvimento de algoritmos que dependem da ordem reversa de processamento.

Na prática, as pilhas são essenciais em sistemas que exigem controle temporário de informações, como a pilha de execução de linguagens de programação, o gerenciamento de histórico em navegadores e operações de desfazer/refazer em editores.

Portanto, o domínio das pilhas permite ao estudante compreender padrões de acesso restrito a dados, fortalecendo a capacidade de análise de problemas computacionais que envolvem processos sequenciais e recursivos.

A figura 2 apresenta o exemplo do funcionamento de estrutura de pilha.

Figura 2. Funcionamento da estrutura de uma pilha



Fonte: Adaptado de (Parreira Junior, 2014)

2.3.3 - Filas

As filas (*FIFO – First In, First Out*) obedecem à ordem de chegada dos elementos, sendo essenciais em sistemas de escalonamento de processos, gerenciamento de recursos e redes de computadores (Cormen et al., 2009).

Podem ser implementadas de forma contígua ou encadeada, permitindo o gerenciamento eficiente de elementos em sequência. Essa flexibilidade é importante para aplicações que exigem manutenção da ordem de chegada dos dados.

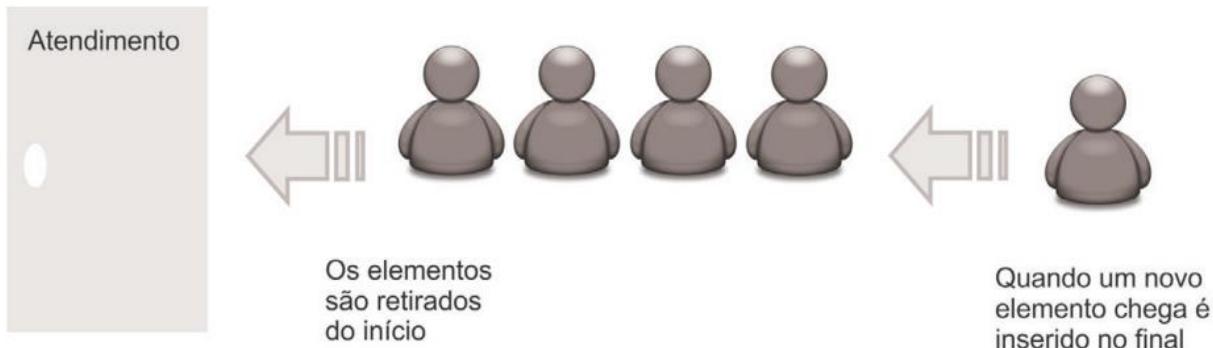
Filas são amplamente aplicadas em sistemas de impressão, buffers de comunicação, filas de atendimento e processos de execução concorrente em sistemas operacionais. O estudo de suas propriedades permite compreender melhor a dinâmica de recursos limitados e a organização de dados em tempo real.

Assim, o entendimento das filas fornece aos estudantes habilidades essenciais de organização sequencial de dados, preparando-os para o estudo de estruturas mais

complexas e para o desenvolvimento de soluções escaláveis em sistemas computacionais.

A figura 3 apresenta o exemplo do funcionamento de estrutura uma fila.

Figura 3. Funcionamento da estrutura de uma fila



Fonte: Adaptado de (Cortés, 2015)

2.3.4 - Árvores

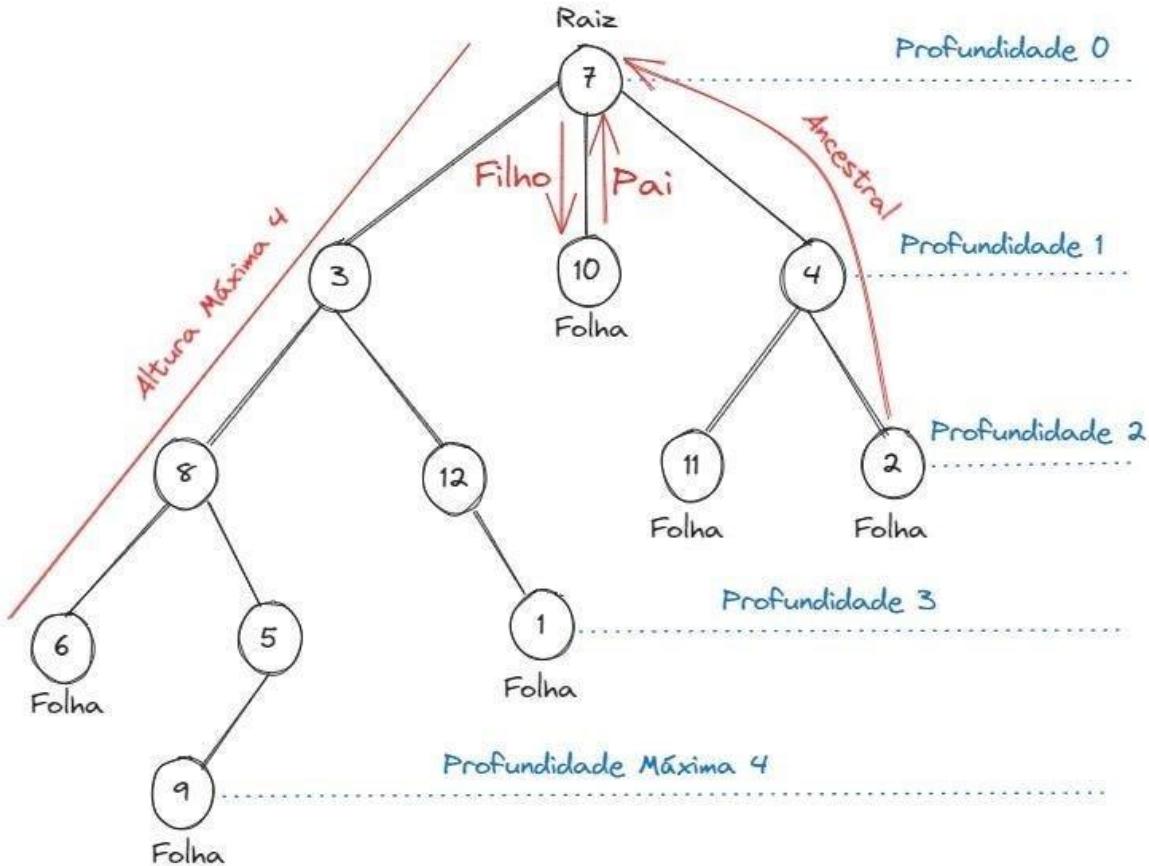
As árvores, por sua vez, representam estruturas hierárquicas que permitem organizar informações de forma eficiente, sendo utilizadas em sistemas de arquivos, bancos de dados e algoritmos de busca e ordenação. Dentro deste grupo, destacam-se árvores平衡adas, como AVL e rubro-negras, que garantem desempenho estável em operações de inserção, remoção e busca.

Na prática, árvores são fundamentais em sistemas que exigem organização hierárquica de dados, como diretórios de arquivos, índices de banco de dados e algoritmos de compressão. Compreender suas propriedades possibilita projetar soluções que otimizem armazenamento e tempo de processamento.

Dessa forma, o estudo de árvores oferece aos estudantes habilidades avançadas de abstração e raciocínio lógico, sendo um alicerce para o desenvolvimento de algoritmos complexos e de estruturas derivadas, como grafos e heaps.

A figura 4 apresenta o exemplo da estrutura de uma árvore e seus conceitos básicos.

Figura 4. Estrutura de uma árvore e seus conceitos básicos



Fonte: Adaptado de (Manzano, 2023)

2.3.5 - Hash Tables (tabelas de espalhamento)

As hash tables são estruturas que associam chaves a valores, utilizando funções de hash para permitir acesso rápido a dados, geralmente em tempo próximo ao constante (Cormen et al., 2009).

São fundamentais para o acesso rápido a dados, empregando funções de hash que permitem localizar informações em tempo próximo ao constante, o que as torna

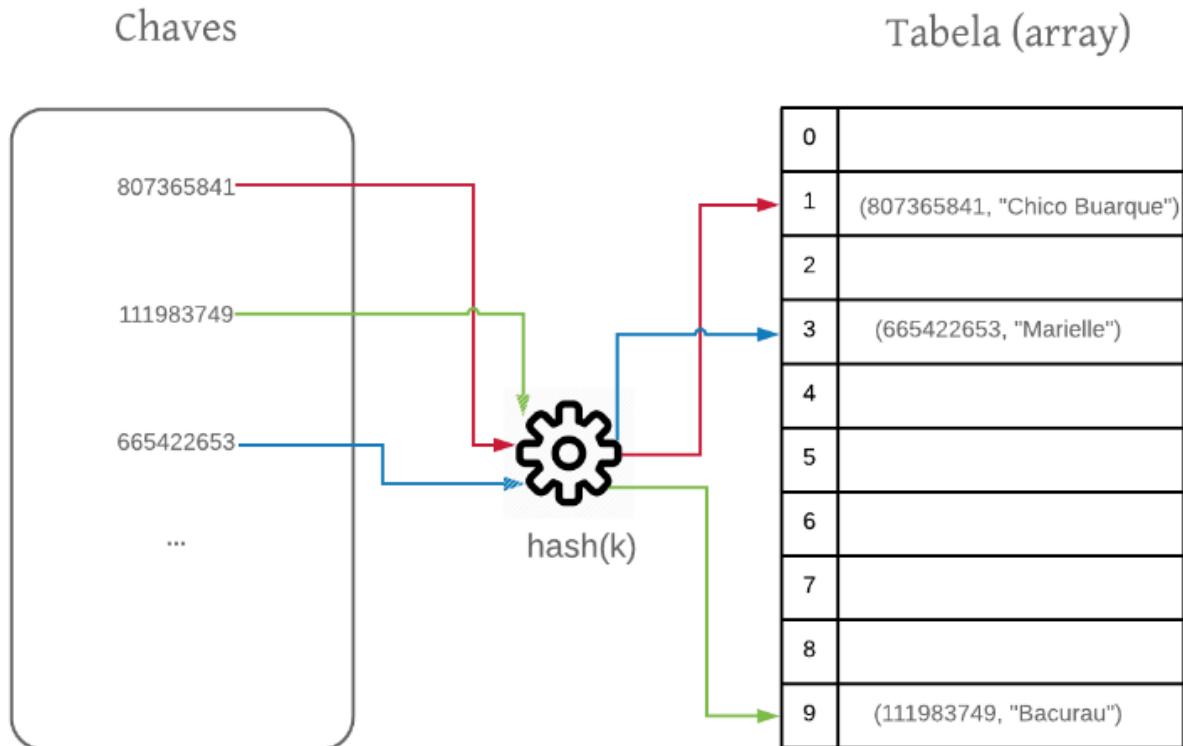
essenciais em aplicações de grande escala. Dessa forma, o domínio desses tipos de estruturas não apenas fornece ao estudante ferramentas práticas para a solução de problemas computacionais, mas também fortalece sua capacidade de raciocínio lógico e abstração, elementos indispensáveis para o desenvolvimento de software robusto e escalável.

Sua implementação envolve o gerenciamento de colisões e técnicas de redimensionamento dinâmico, que garantem eficiência e confiabilidade mesmo em aplicações de grande escala.

Em termos práticos, são utilizadas em sistemas de indexação, bancos de dados, linguagens de programação (como implementações de dicionários) e qualquer aplicação que exija buscas rápidas e armazenamento eficiente de dados.

A figura 4 apresenta o exemplo de uma Hash Table.

Figura 5. Estrutura de uma Hash Table



Fonte: Adaptado de (Brunet, 2019).

2.4 Teorias de Aprendizagem e Gamificação

Nesse cenário, o uso de jogos digitais educacionais surge como alternativa promissora. Estudos de Prensky (2012) destacam que a aprendizagem baseada em jogos promove maior engajamento e retenção do conhecimento, pois aproxima o conteúdo teórico de experiências práticas interativas. Além disso, o desenvolvimento de minigames tem se consolidado como estratégia eficaz para a fixação de conceitos abstratos em áreas como Matemática, Física e Programação, plataformas interativas cada vez mais utilizadas por docentes em suas respectivas áreas (Kohler et al., 2023).

Com base nisso, o projeto Logic Shuffle propõe-se a explorar o potencial do Godot Engine, aliado a um back-end em Python e banco de dados PostgreSQL em nuvem, para criar uma plataforma gamificada voltada ao ensino de estruturas de dados. A proposta não apenas visa tornar o aprendizado mais acessível e lúdico, mas também contribuir para a formação de estudantes mais preparados, oferecendo ferramentas escaláveis que, em versões futuras, poderão incluir funcionalidades como ranking de usuários, relatórios de desempenho e notificações personalizadas.

Assim como a Política Nacional de Educação da Lei nº 9.394/1996 (BRASIL, 1996) estabeleceu a necessidade de metodologias inovadoras para ampliar o acesso e a qualidade do ensino, iniciativas como o Logic Shuffle alinharam-se às diretrizes de modernização pedagógica, reforçando o papel da tecnologia digital como aliada no processo educacional. Dessa forma, o projeto busca unir ciência da computação, design educacional e inovação tecnológica, em resposta a um dos maiores dilemas contemporâneos: como ensinar conteúdos complexos de maneira eficiente e motivadora para estudantes da era digital.

2.5 Trabalhos relacionados

A pesquisa realizada em bases acadêmicas como: SBC OpenLib (SOL), Google Acadêmico, SciELO, BRAPCI e o Portal de Periódicos CAPES, identificou diversos

trabalhos que exploram o uso de jogos digitais no ensino de algoritmos e estruturas de dados, validando a relevância e o potencial de abordagens lúdicas nesta área. A análise desses projetos forneceu *insights* valiosos sobre metodologias, tecnologias e resultados alcançados, servindo como referência e base comparativa para o desenvolvimento do Logic Shuffle.

Souza et al. (2024) apresentam o SpaceCode, um jogo educacional ambientado no espaço que estimula a resolução de problemas algorítmicos de forma lúdica. A narrativa do jogo envolve desafios progressivos em que o jogador deve aplicar conceitos de lógica e programação para avançar em missões espaciais, explorando planetas e superando obstáculos. Utilizando a engine Unity, C#, JSON, Sprites 2D e Prefabs, o jogo destaca-se pela interatividade e pelo incentivo à autonomia do estudante, transformando o estudo de algoritmos em uma experiência motivadora através de elementos visuais atrativos.

Em uma abordagem alinhada à Educação 5.0, Barbosa et al. (2023) propõem um protótipo de jogo que combina gamificação e *storytelling* para o ensino de estruturas de dados. Construído também com Unity e C#, o jogo busca não apenas transmitir conteúdos técnicos, mas também desenvolver competências socioemocionais, como colaboração e criatividade. A ferramenta insere o estudante em uma narrativa interativa onde a progressão depende tanto da compreensão de conceitos técnicos quanto da capacidade de interagir em grupo e resolver problemas de forma criativa, integrando aspectos cognitivos e comportamentais ao processo de aprendizagem.

Focando diretamente no ensino de estruturas de dados, Battistella, Wangenheim e Wangenheim (2012) desenvolveram o Sortia, um jogo educativo voltado ao ensino de algoritmos de ordenação, em especial o Heapsort. O objetivo do jogo é permitir que o estudante compreenda o funcionamento interno dos algoritmos ao executar manualmente, de forma interativa, o processo de ordenação de um conjunto de números inteiros. A aplicação do Sortia na disciplina de Estrutura de Dados da Universidade Federal de Santa Catarina demonstrou resultados positivos quanto à motivação e ao aprendizado dos estudantes, com mais de 60% dos participantes atribuindo notas elevadas (1 e 2 em escala Likert de -2 a +2) à eficácia do jogo na aprendizagem e

adequação ao seu modo de aprender. O estudo também apontou que a maioria dos alunos considerou o jogo fácil de compreender e capaz de manter a atenção durante as atividades, reforçando o potencial da gamificação como ferramenta complementar no ensino de conteúdos técnicos abstratos.

Todos esses trabalhos comprovam a eficácia dos jogos digitais no processo de ensino-aprendizagem de conceitos computacionais, o Logic Shuffle se diferencia por sua implementação na Godot Engine e pela proposta de reunir vários minijogos dedicados a diferentes estruturas de dados (listas, pilhas, filas e árvores) em uma única plataforma. Essa abordagem busca oferecer uma experiência de aprendizado mais abrangente, escalável, acessível e adaptada às demandas contemporâneas da educação em Computação.

O Quadro 1 apresenta um comparativo sumarizado das características principais dos trabalhos relacionados mencionados.

Quadro 1: Trabalhos relacionados

Autor/Ano	Objetivo	Ferramenta Desenvolvida	Tecnologias Utilizadas
Souza et al. (2024)	Estimular a resolução de problemas relacionados a algoritmos de forma lúdica	SpaceCode – jogo educacional com temática espacial	Unity, C#, JSON, Sprites 2D, Prefabs
Barbosa et al. (2023)	Ensinar estruturas de dados e desenvolver competências socioemocionais (colaboração, criatividade)	Protótipo de jogo educativo com gamificação e storytelling	Unity, C#, gamificação, storytelling
Battistella, Wangenheim e Wangenheim (2012)	Apoiar o ensino de algoritmos de ordenação por meio de simulação interativa e avaliação empírica	<i>Sortia</i> – jogo educativo para ensino de ordenação (Heapsort)	JavaScript, HTML, Moodle, vídeos instrucionais

Fonte: Autoria Própria

3. Análise de Requisitos

3.1 Visão geral do Produto

O Logic Shuffle consiste em uma plataforma educacional digital voltada ao ensino de estruturas de dados, desenvolvida em formato de jogos interativos. A proposta do sistema é permitir que o estudante aprenda e pratique conceitos fundamentais da computação de forma visual, dinâmica e intuitiva. Nesta primeira versão, o escopo contempla a implementação de jogos voltados para árvores, pilhas, filas e deque, estruturas escolhidas por sua relevância e por serem frequentemente apontadas como de maior dificuldade pelos alunos em cursos de computação.

O sistema disponibilizará uma interface acessível e gamificada, em que o usuário poderá navegar por diferentes níveis de aprendizado. Cada jogo foi projetado para representar uma estrutura de dados específica, com visualização gráfica das operações realizadas, como inserção, remoção e busca de elementos. Dessa forma, o estudante poderá acompanhar, em tempo real, o funcionamento interno de cada estrutura, compreendendo não apenas a teoria, mas também sua aplicação prática.

Entre as opções oferecidas pelo sistema, destacam-se o cadastro de usuário, que possibilitará a criação de perfis individuais, e o registro automático de progresso, permitindo ao estudante retomar atividades de onde parou. O usuário poderá selecionar o jogo desejado e escolher diferentes fases ou desafios, que variam em complexidade e são elaborados para promover a consolidação gradual do conhecimento. Além disso, é disponibilizado um sistema de ranking, por meio do qual os participantes podem visualizar suas pontuações e compará-las com outros jogadores, incentivando o aspecto lúdico e competitivo sem comprometer o caráter pedagógico.

Outro diferencial do produto é a presença de feedback imediato, visual e sonoro, que orienta o usuário sobre acertos e erros durante a execução das atividades. Esse recurso amplia a percepção sobre o impacto de cada ação dentro da estrutura de dados simulada, contribuindo para uma experiência de aprendizado mais clara e eficiente.

Assim, o LogicShuffle caracteriza-se como uma ferramenta educacional complementar ao ensino de Estruturas de Dados, ao unir conteúdo técnico, interatividade e elementos de gamificação em um mesmo espaço. A proposta visa não apenas facilitar a compreensão de conceitos abstratos, mas também tornar o processo de aprendizagem mais atrativo e envolvente, de modo que o estudante desenvolva habilidades cognitivas essenciais para sua formação acadêmica e profissional.

3.2 Descrição da técnica utilizada para levantamento dos requisitos

Para o levantamento dos requisitos do Logic Shuffle, foi adotada a técnica de pesquisa quantitativa por meio de questionários online, utilizando a ferramenta Google Forms. Essa abordagem permitiu coletar dados de maneira prática e organizada, alcançando um público diversificado e obtendo informações relevantes sobre o perfil dos usuários potenciais do sistema.

O questionário foi composto por perguntas fechadas de múltipla escolha e escala de opinião, complementadas por algumas questões abertas que permitiram a coleta de percepções qualitativas. As perguntas abrangeram aspectos como o grau de dificuldade enfrentado pelos estudantes no aprendizado de estruturas de dados, os conteúdos considerados mais complexos, os métodos pedagógicos que geram maior engajamento e as funcionalidades desejadas em ferramentas de apoio educacional. Dessa forma, buscou-se identificar as necessidades do público-alvo de maneira objetiva, sem desistir de capturar insights relevantes por meio das respostas discursivas.

O instrumento de pesquisa foi divulgado entre estudantes de cursos relacionados à Computação, como Ciência da Computação, Sistemas de Informação, Engenharia de Software e cursos técnicos em programação, abrangendo ainda autodidatas interessados em aprofundar seus conhecimentos. O período de aplicação estendeu-se por 14 dias, durante os quais foi possível reunir uma amostra representativa do público pretendido.

A adoção dessa técnica mostrou-se adequada por permitir a obtenção de dados quantitativos consistentes, ao mesmo tempo em que forneceu subsídios qualitativos que

auxiliaram na priorização das funcionalidades a serem implementadas. Os resultados coletados serviram como base para a definição do escopo do projeto e para a identificação das estruturas de dados que exigem maior atenção no desenvolvimento dos minigames, conforme será detalhado na seção seguinte.

3.2.1 Levantamento e Análise dos Resultados do Questionário

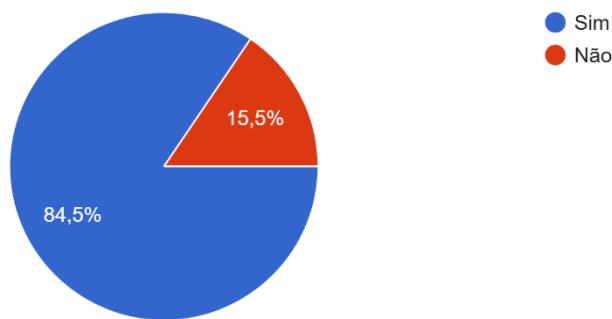
Com o objetivo de identificar o perfil dos usuários potenciais e compreender suas principais dificuldades no aprendizado de Estruturas de Dados, foi aplicado um questionário online contendo treze questões, distribuídas entre perguntas de identificação, experiência acadêmica e profissional, dificuldades enfrentadas e preferências quanto a recursos interativos. Ao todo, 58 pessoas responderam ao questionário, compondo a amostra analisada neste estudo. A seguir, apresenta-se a análise dos resultados obtidos.

O primeiro bloco do questionário teve como objetivo compreender o perfil dos respondentes. Observou-se que a maior parte possui vínculo com a área de Tecnologia da Informação, seja como estudante ou profissional atuante, ou seja 84,5% dos entrevistados conforme figura 6. Dentre os cursos mencionados, destacou-se o de Análise e Desenvolvimento de Sistemas, citado por 82,8% dos participantes (figura 7), representando a formação predominante entre os respondentes.

Figura 6 - Formação predominante dos pesquisados

1) Você é formado, cursa algum curso ou trabalha na área de Tecnologia de Informação?

58 respostas



Fonte: Autoria Própria, elaborado a partir de dados coletados via Google Forms

Figura 7 – Curso predominante dos pesquisados

2) Caso seja formado ou esteja cursando, qual é o curso:

58 respostas



▲ 1/2 ▼

Fonte: Autoria Própria, elaborado a partir de dados coletados via Google Forms

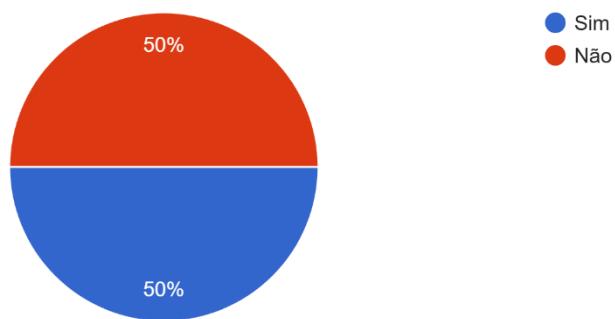
Na sequência, procurou-se identificar o contato prévio dos respondentes com a disciplina de Estruturas de Dados. Constatou-se que a metade dos entrevistados já

cursou ou está cursando a disciplina (figura 8), o que reforça a relevância do levantamento para validar as dificuldades enfrentadas em sala de aula.

Figura 8 - Contato prévio com a disciplina Estruturas de Dados

3) Já cursou a disciplina de Estrutura de Dados?

58 respostas



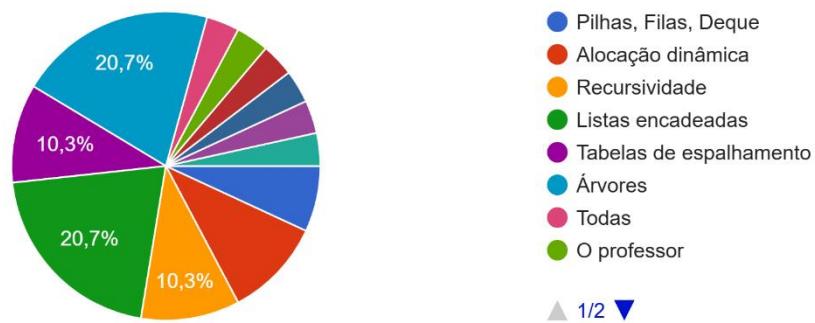
Fonte: Autoria Própria, elaborado a partir de dados coletados via Google Forms

Quando questionados sobre as estruturas de dados consideradas mais difíceis, destacou-se a predominância de árvores, listas encadeadas, recursividade e tabelas de espalhamento entre as respostas, evidenciando que esses conteúdos apresentam maior nível de abstração e, portanto, maior necessidade de recursos didáticos complementares (figura 9).

Figura 9 – Estruturas de dados consideradas mais difíceis

4) Qual parte da disciplina de Estrutura de Dados você achou mais difícil?

29 respostas



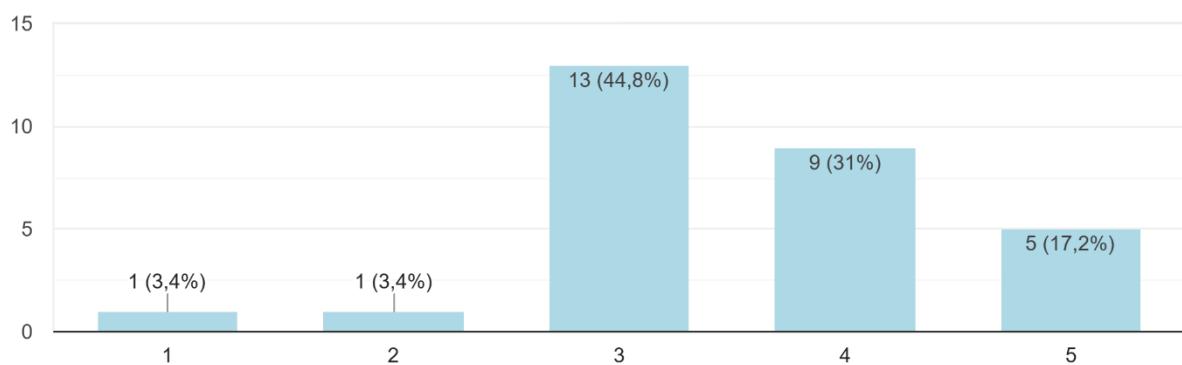
Fonte: Autoria Própria, elaborado a partir de dados coletados via Google Forms

O grau de dificuldade geral na disciplina também foi avaliado em uma escala de 1 a 5, sendo que a maior concentração se deu nos níveis 3 a 5, indicando percepção de dificuldade média a alta entre os estudantes (figura 10).

Figura 10 – Grau de dificuldade percebido na disciplina (escala 1–5)

5) Em uma escala de 1 a 5, qual foi o seu nível de dificuldade geral na disciplina?

29 respostas

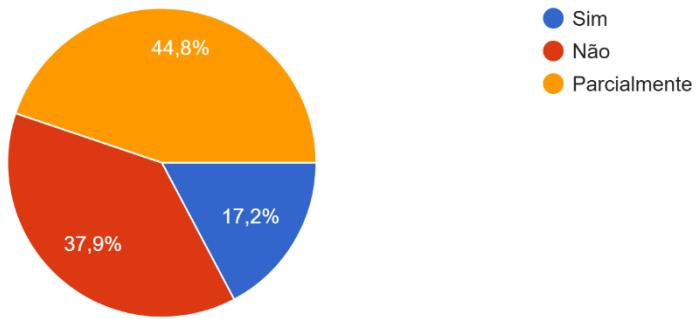


Fonte: Autoria Própria, elaborado a partir de dados coletados via Google Forms

Quanto à eficácia dos métodos tradicionais de ensino (aula expositiva e exercícios práticos), a maioria dos respondentes considerou que tais métodos foram insuficientes ou apenas parcialmente eficazes para a assimilação do conteúdo, reforçando a necessidade de abordagens alternativas (figura 11).

Figura 11 – Avaliação da eficácia dos métodos de ensino tradicionais

- 6) Você considera que os métodos tradicionais (aula + exercícios) foram suficientes para o aprendizado?
29 respostas



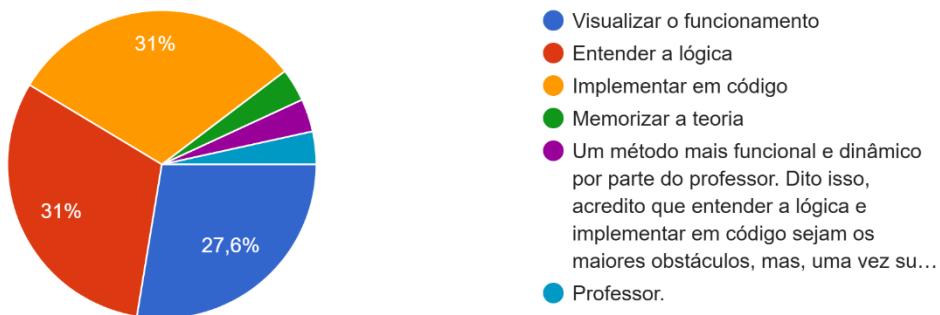
Fonte: Autoria Própria, elaborado a partir de dados coletados via Google Forms

No que se refere às maiores dificuldades relatadas, os resultados mostraram que a interpretação da lógica, seguida da implementação em código, foram os aspectos mais apontados (figura 12). Isso evidencia a importância de recursos que permitam uma melhor representação visual e prática dos conceitos.

Figura 12 – Principais dificuldades relatadas

7) Qual é a maior dificuldade ao aprender Estruturas de Dados?

29 respostas



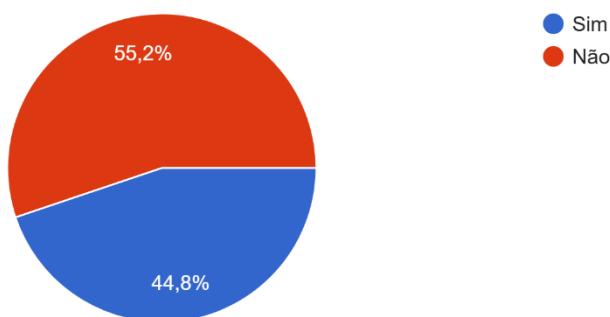
Fonte: Autoria Própria, elaborado a partir de dados coletados via Google Forms

O questionário também investigou experiências prévias com jogos e aplicativos educacionais. Parte significativa dos participantes declarou não ter utilizado esse tipo de recurso (figura 13), porém a maioria afirmou acreditar que minigames poderiam facilitar o aprendizado de Estruturas de Dados (figura 14), atribuindo notas altas na escala de concordância.

Figura 13 - Experiência prévia com jogos e aplicativos educacionais

8) Você já utilizou algum jogo ou aplicativo para estudar conteúdos acadêmicos?

58 respostas

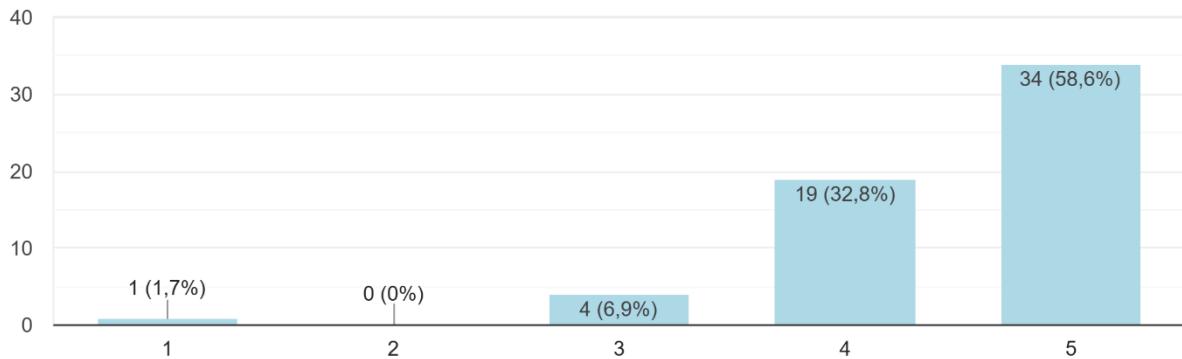


Fonte: Autoria Própria, elaborado a partir de dados coletados via Google Forms

Figura 14 - Concordância sobre a utilidade de minigames no aprendizado

9) Em sua opinião uma aplicação no formato de minijogos poderiam facilitar o aprendizado de Estruturas de Dados?

58 respostas



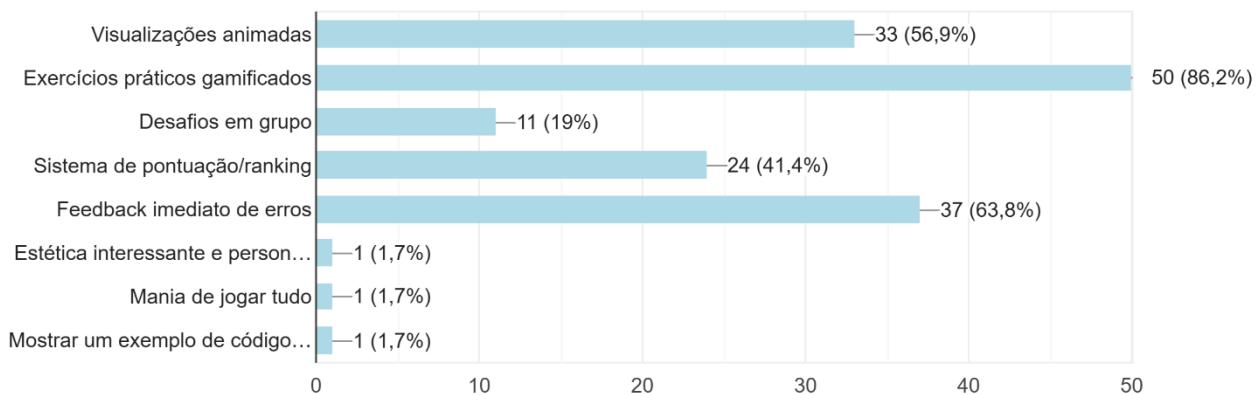
Fonte: Autoria Própria, elaborado a partir de dados coletados via Google Forms

Com relação aos recursos interativos mais desejados, destacaram-se as visualizações animadas, os exercícios práticos gamificados e o feedback imediato sobre erros (figura 15). Essas funcionalidades foram citadas como as que mais poderiam auxiliar na compreensão e fixação dos conteúdos.

Figura 15 – Recursos interativos mais desejados

10) Que tipo de recurso interativo ajudaria mais no seu aprendizado?

58 respostas



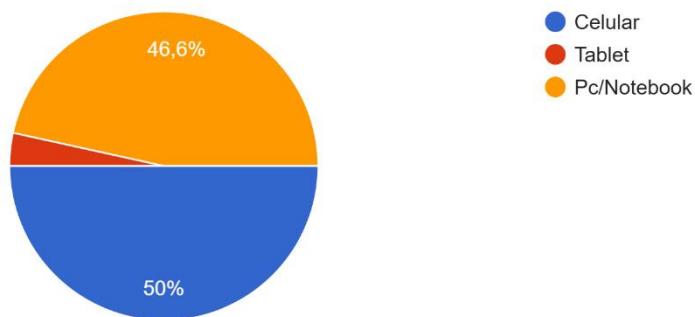
Fonte: Autoria Própria, elaborado a partir de dados coletados via Google Forms

Foi questionado ainda, qual dispositivo seria mais utilizado para acessar a aplicação. Os resultados indicaram predominância do uso de celulares, embora computadores pessoais e notebooks também tenham sido apontados como opção relevante (figura 16), o que reforça a necessidade de responsividade na plataforma.

Figura 16 – Dispositivos preferidos para acesso à aplicação

11) Qual dispositivo você mais usaria para jogar/aprender?

58 respostas

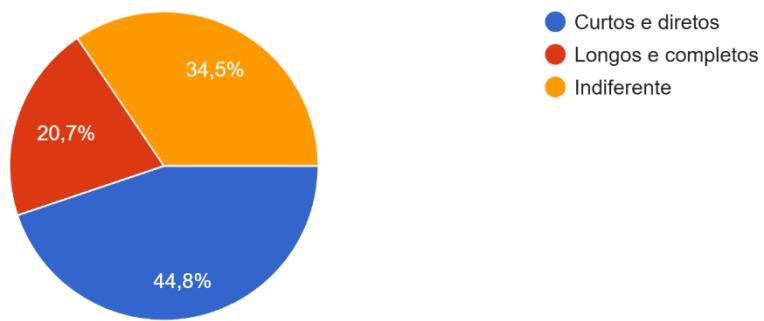


Fonte: Autoria Própria, elaborado a partir de dados coletados via Google Forms

Em relação ao formato dos minigames, a preferência predominante foi por jogos curtos e diretos, seguidos pelos participantes indiferentes à duração, enquanto os desafios mais longos e completos tiveram menor adesão (figura 17). Esses resultados indicam a importância de oferecer diferentes modos de interação, capazes de atender aos variados perfis de estudantes.

Figura 17 – Dispositivos preferidos para acesso à aplicação

12) Você preferiria minijogos mais curtos e diretos ou desafios mais longos e completos?
58 respostas



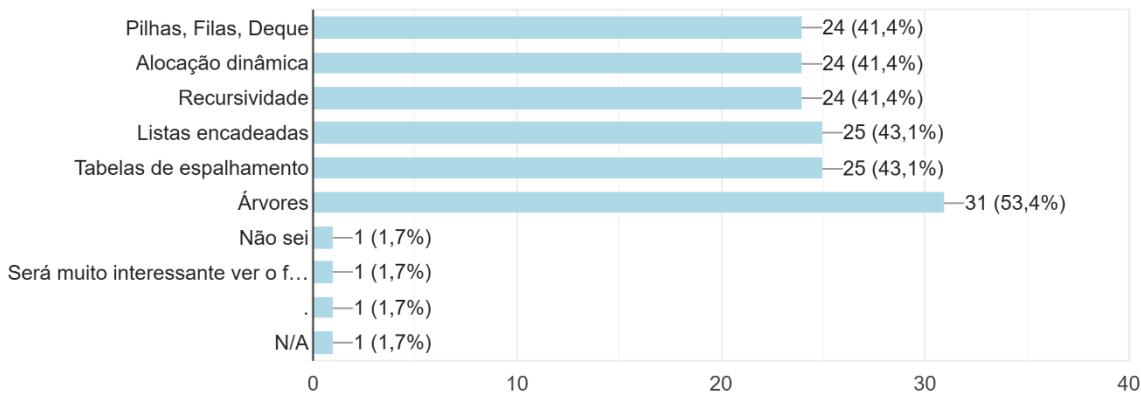
Fonte: Autoria Própria, elaborado a partir de dados coletados via Google Forms

Por fim, ao indicar qual estrutura de dados mais necessita de visualização prática para ser compreendida, as respostas reforçaram os resultados anteriores, destacando novamente árvores, listas encadeadas e tabelas de espalhamento como os maiores desafios pedagógicos, seguidos por recursividade, pilhas/filas/deques, e alocação dinâmica, conforme figura 18.

Figura 18 – Estrutura mais necessária de visualização prática

13) Qual estrutura de dados você acha que mais precisa de visualização prática para aprender?

58 respostas



Fonte: Autoria Própria, elaborado a partir de dados coletados via Google Forms

Dessa forma, a análise do questionário permitiu mapear não apenas o perfil e as dificuldades dos estudantes, mas também suas expectativas quanto a recursos didáticos inovadores. Esses resultados fornecem subsídios fundamentais para a definição das funcionalidades prioritárias do sistema Logic Shuffle, garantindo maior alinhamento com as necessidades reais de seu público-alvo.

3.2.2 - Ferramentas encontradas no mercado

A pesquisa direta com potenciais usuários foi complementada pela análise de ferramentas e recursos educacionais já existentes no mercado que possuem características e propostas semelhantes às do Logic Shuffle. Essa análise foi essencial para a construção do projeto, pois permitiu identificar boas práticas de gamificação, tipos de interação e metodologias de ensino que poderiam ser incorporadas aos minijogos, assegurando que a aplicação desenvolvida fosse, simultaneamente, educativa, motivadora e alinhada às necessidades reais dos estudantes. O estudo de projetos acadêmicos e soluções comerciais, conforme detalhado na Seção 2.5 deste trabalho,

validou a eficácia dos jogos digitais no processo de ensino-aprendizagem, reforçando a pertinência da abordagem adotada.

Entre as plataformas analisadas, destaca-se o LootCode (figuras 19 e 20), um ambiente de aprendizagem interativo voltado ao ensino de programação e lógica por meio de desafios gamificados. A plataforma propõe exercícios progressivos em formato de missões, nas quais o usuário escreve e executa trechos de código diretamente no navegador, recebendo feedback automático e recompensas pela conclusão das etapas. O LootCode adota princípios de aprendizagem baseada em desafios (*Challenge-Based Learning*), associando a resolução de problemas a um sistema de progressão e pontuação que estimula o engajamento contínuo.

Figura 19 - LootCode



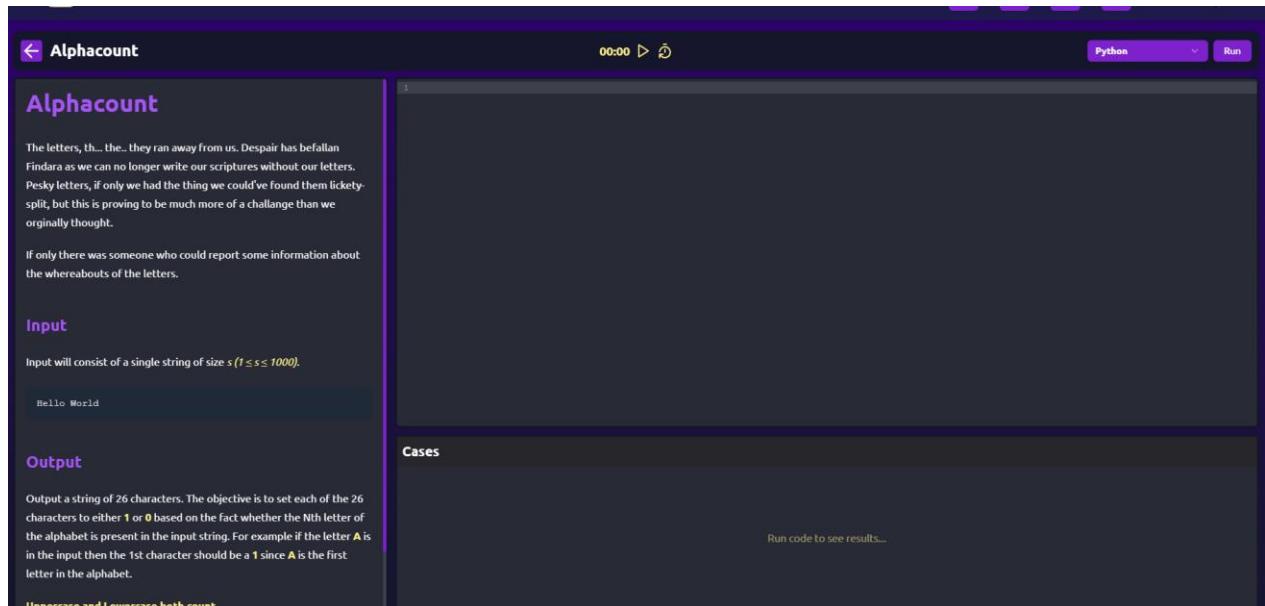
Fonte: LootCode¹

O sistema diferencia-se por oferecer uma experiência de codificação acessível e intuitiva, que combina práticas de programação com mecânicas de jogo, tornando o aprendizado mais dinâmico. A cada desafio concluído, o usuário acumula pontos e avança em um ranking global, o que incentiva a competitividade e a melhoria de desempenho. O design da plataforma valoriza a imersão e a clareza visual, apresentando

¹ Disponível em: <https://www.lootcode.dev>.

uma interface responsiva e estruturada em torno de missões temáticas que reforçam o aspecto lúdico da aprendizagem.

Figura 20 – Tela de jogo do LootCode



Fonte: LootCode

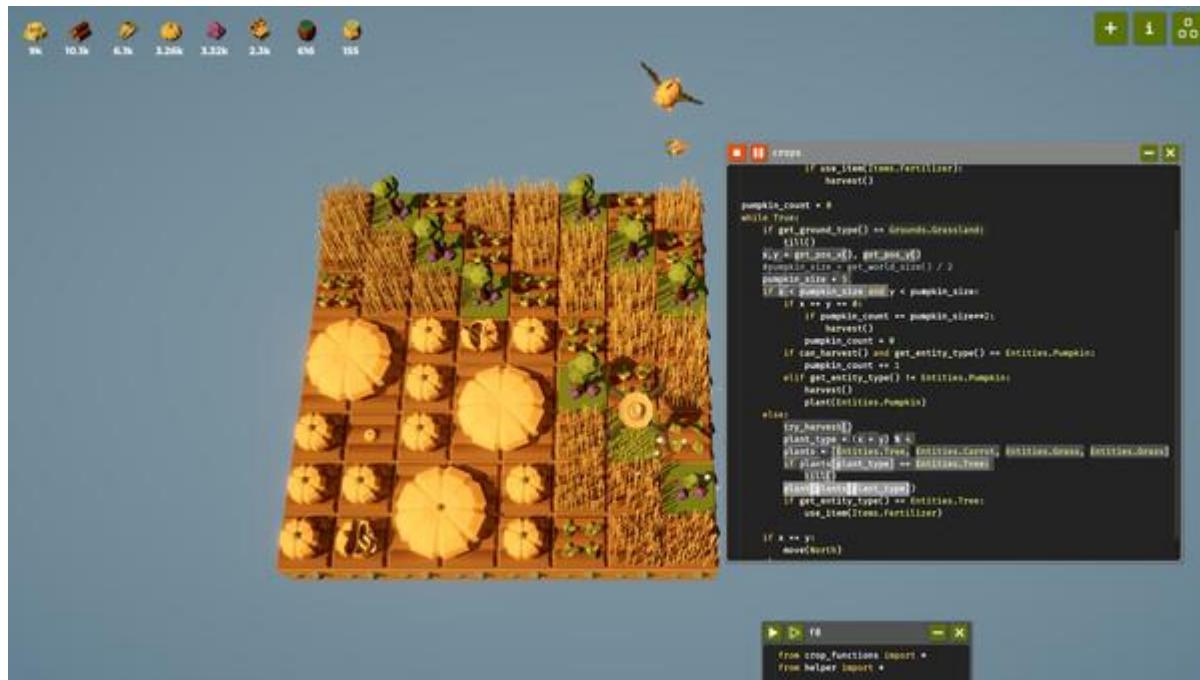
A análise do LootCode serviu como referência para o desenvolvimento do Logic Shuffle, principalmente na incorporação de elementos de progressão, pontuação e feedback imediato. No entanto, enquanto o LootCode tem como foco principal a escrita e execução de código, o Logic Shuffle se diferencia por priorizar a visualização e manipulação gráfica das Estruturas de Dados, tornando mais tangível o comportamento interno das estruturas e auxiliando na compreensão de seus princípios fundamentais.

A partir dessa comparação, foi possível identificar oportunidades de aprimoramento e delinear uma proposta que unisse o rigor conceitual da programação com a abordagem visual e interativa dos jogos digitais, consolidando o Logic Shuffle como uma solução educacional complementar.

Um outro jogo estudado foi o The Farmer Was Replaced (figura 21), uma experiência educativa que ensina fundamentos de lógica de programação e automação por meio da simulação de um ambiente agrícola controlado por robôs. O jogador deve

criar sequências de comandos para que os robôs executem tarefas como plantar, colher e armazenar produtos, aprendendo de forma prática conceitos como repetição, condição e sequência lógica. O jogo adota uma estética minimalista e progressiva, em que cada fase apresenta novos desafios que estimulam o raciocínio computacional e a resolução de problemas.

Figura 21 - The Farmer Was Replaced



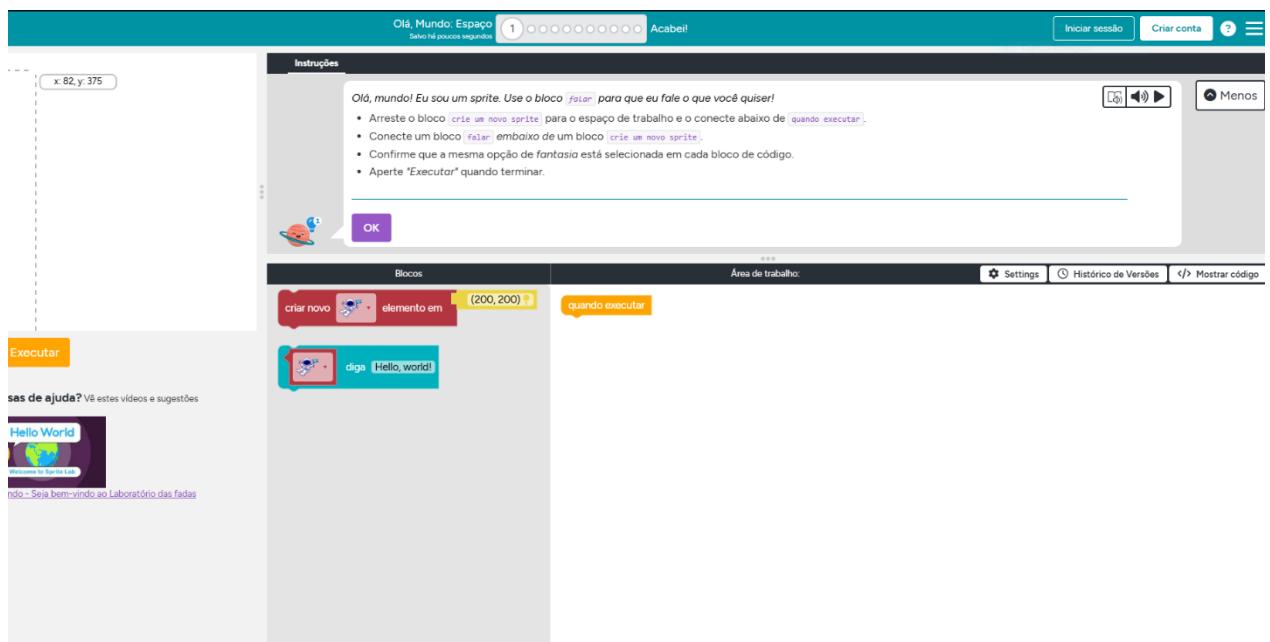
Fonte: The Farmer Was Replaced²

Essa abordagem de aprendizado baseado em experimentação do The Farmer Was Replaced influenciou diretamente o desenvolvimento do Logic Shuffle, que também busca ensinar por meio da prática e da resolução de desafios interativos. No entanto, enquanto o The Farmer Was Replaced trabalha com lógica de programação em um contexto de automação, esse projeto aplica a mesma lógica de aprendizagem incremental para o ensino de Estruturas de Dados, permitindo ao usuário visualizar o comportamento interno das estruturas e compreender sua aplicação em situações práticas.

² Disponível em: <https://thefarmerwasreplaced.com>

Outra iniciativa relevante é o Hour of Code (figura 22), um movimento global que promove o ensino introdutório de programação de maneira acessível e lúdica. A plataforma oferece tutoriais interativos que utilizam personagens populares e narrativas visuais para ensinar os fundamentos da lógica computacional, controle de fluxo e estruturas básicas de repetição e condição. O público-alvo inclui estudantes de diferentes faixas etárias e níveis de conhecimento, o que contribui para democratizar o acesso ao aprendizado em Computação.

Figura 22 - Hour of Code



Fonte: Hour of Code³

O Hour of Code foi uma das referências mais importantes na concepção da experiência visual do Logic Shuffle, principalmente no uso de elementos lúdicos e feedbacks imediatos para reforçar o aprendizado. Contudo, o projeto proposto neste trabalho diferencia-se ao concentrar-se exclusivamente em Estruturas de Dados, abordando conteúdos de maior complexidade técnica e apresentando desafios voltados à visualização e manipulação direta dessas estruturas. Assim, enquanto o Hour of Code introduz conceitos gerais de programação, o Logic Shuffle aprofunda o aprendizado em

³ Disponível em: <https://code.org/pt-BR>

um campo específico da computação, tornando-se uma ferramenta complementar às práticas de ensino tradicionais.

Sendo assim, a investigação dessas aplicações do mercado foi essencial para compreender as estratégias de ensino aplicadas em soluções interativas e como a gamificação pode potencializar o aprendizado de conceitos abstratos em computação. A análise comparativa permitiu identificar boas práticas de design e interatividade, bem como lacunas metodológicas que o projeto buscou preencher ao focar especificamente no ensino de Estruturas de Dados.

3.2.3 - Comparação entre Game Engines

Para o desenvolvimento do sistema de minigames proposto, foi realizada também uma análise comparativa entre diferentes motores gráficos (game engines) disponíveis no mercado. O objetivo dessa comparação foi identificar a ferramenta mais adequada em termos de recursos técnicos, curva de aprendizado, comunidade e compatibilidade com os requisitos do projeto. As plataformas analisadas foram Unity, Unreal Engine, CryEngine e Godot Engine.

A Unity é uma das engines mais utilizadas na indústria, amplamente adotada tanto para jogos 2D quanto 3D. Oferece grande variedade de recursos, vasta biblioteca de assets e documentação robusta. Contudo, sua interface pode apresentar complexidade para iniciantes, e a dependência de licenciamento em projetos de maior porte pode representar uma limitação. (Business of Esports, 2022)

A Unreal Engine, por sua vez, destaca-se pela qualidade gráfica de alto nível, sendo amplamente utilizada em produções AAA (ou Triple A). Apesar de seu poder, a engine demanda hardware mais robusto e possui curva de aprendizado mais íngreme, fatores que a tornam menos atrativa para projetos acadêmicos que priorizam simplicidade e eficiência no desenvolvimento. (Harpooner, 2019)

A CryEngine também se destaca pela qualidade gráfica avançada, sendo utilizada em jogos de grande porte que exigem realismo visual. Entretanto, apresenta documentação menos acessível, menor comunidade de suporte quando comparada a

Unity e Unreal, e complexidade elevada para desenvolvedores em fase inicial, o que a torna pouco indicada para projetos educacionais de escopo limitado. (Crytek, s.d.)

A Godot Engine, por sua vez, foi escolhida como plataforma para o desenvolvimento do Logic Shuffle. Entre os fatores determinantes estão sua interface intuitiva, curva de aprendizado acessível e a forte ênfase em projetos 2D, que é o foco principal deste trabalho. Além disso, a Godot é totalmente gratuita e de código aberto, permitindo maior liberdade de customização sem custos de licenciamento. Sua comunidade ativa e em crescimento também proporciona suporte adequado, aliado a uma documentação clara e objetiva. (Godot Engine, s.d.)

Dessa forma, embora Unity, Unreal e CryEngine apresentem recursos de alto desempenho e sejam amplamente utilizadas na indústria, a Godot Engine mostrou-se a opção mais apropriada para este projeto, conciliando facilidade de uso, eficiência em jogos 2D e ausência de barreiras financeiras. Essa escolha permite manter o foco nos objetivos educacionais do sistema, sem comprometer a qualidade técnica do desenvolvimento.

Quadro 2: Comparação entre Games Engines

Critério	Unity ⁴	Unreal ⁵	Cry Engine ⁶	Godot ⁷
Licença	Gratuita até certo faturamento; planos pagos para uso comercial	Gratuita com royalties sobre receita	Gratuita com royalties sobre receita	100% gratuita e open source
Foco Principal	Jogos 2D e 3D, multiplataforma	Jogos AAA em 3D com alto realismo	Jogos AAA em 3D com foco em gráficos realistas	Jogos 2D e 3D, foco em projetos leves e independentes
Curva de Aprendizado	Moderada, ampla documentação	Íngreme, exige mais experiência técnica	Elevada, documentação menos acessível	Suave, interface intuitiva e linguagem própria (GDScript)
Comunidade	Muito ampla e consolidada	Muito ampla e consolidada	Restrita em relação às demais	Em crescimento constante, ativa e colaborativa
Recursos Gráficos	Boa qualidade gráfica, mas limitada frente à Unreal e CryEngine	Qualidade gráfica avançada, referência em realismo	Gráficos de alta qualidade, realismo avançado	Suficiente para 2D e 3D, mas não foca em realismo extremo
Desempenho	Estável em 2D e 3D, boa otimização	Exige hardware robusto	Exige hardware robusto	Leve, eficiente em 2D e moderado em 3D
Custo-Benefício	Bom, mas limitado por licenciamento	Bom apenas para grandes produções	Restrito a projetos de grande porte	Excelente para projetos acadêmicos e independentes

Fonte: Autoria Própria

⁴ Disponível em: <https://unity.com/pt>

⁵ Disponível em: <https://www.unrealengine.com/pt-BR>

⁶ Disponível em: <https://www.cryengine.com>

⁷ Disponível em: godotengine.org

3.4 Requisitos de Software

Com base nos objetivos do projeto e na análise das necessidades dos usuários, foram definidos os requisitos de software. A seguir, detalham-se os requisitos funcionais e não funcionais da aplicação.

3.4.1 Requisitos Funcionais

RF01 – Cadastro

Permite ao usuário realizar o cadastro no sistema.

RF02 – Login

Permite ao usuário realizar o login no sistema, utilizando o SendGrid para o envio de e-mails de verificação e confirmação, garantindo a autenticação do acesso de forma integrada e segura.

RF03 – Logout

Permite ao usuário sair do sistema.

RF04 – Editar Perfil

Permite ao usuário alterar suas informações, como foto de perfil, nome e senha.

RF05 – Consultar Pontuação

Permite ao usuário verificar as maiores pontuações efetuadas em cada jogo, por meio de um ranking online.

RF06 – Alterar Configurações

Permite ao usuário alterar o volume do sistema.

RF07 – Selecionar Jogo

Permite ao usuário selecionar um jogo.

RF08 – Selecionar Fase

Permite ao usuário escolher uma fase do jogo, seja aquela em que parou na última vez que jogou ou uma já concluída anteriormente.

3.4.2 Requisitos Não Funcionais

RNF01 – Usabilidade

A interface deve ser intuitiva e acessível, garantindo uma navegação fluida e uma experiência de aprendizado eficiente.

As interações dos minigames devem ser responsivas e conter feedback visual ou sonoro. O sistema deve, inclusive, permitir ajustes de volume da música e efeitos sonoros.

RNF02 – Desempenho e Eficiência

O sistema será otimizado para hardware moderno. As consultas ao banco de dados serão indexadas, garantindo tempos de resposta adequados ao escopo proposto.

RNF03 – Escalabilidade

A aplicação será desenvolvida com suporte ao crescimento orgânico de usuários (≥ 100). Inicialmente, não será testada para cenários de carga massiva.

RNF04 – Segurança

Será implementada autenticação baseada em JWT (JSON Web Token) e utilização de hash SHA-256 para o armazenamento seguro das senhas.

RNF05 – Responsividade

A interface será projetada para se adaptar a diferentes tamanhos de tela, priorizando a compatibilidade com resoluções comuns em ambiente desktop.

RNF06 – Manutenibilidade

O código será estruturado de forma modular, com separação lógica entre os componentes. Cada tela do jogo possuirá código independente, favorecendo futuras manutenções e expansão do sistema.

RNF07 – Confiabilidade

Serão implementadas validações básicas, sistema funcional de save/load e tratamento de erros, garantindo maior estabilidade durante a execução.

RNF08 – Portabilidade

O sistema será desenvolvido com suporte multiplataforma, visando compatibilidade com Windows, Linux e macOS.

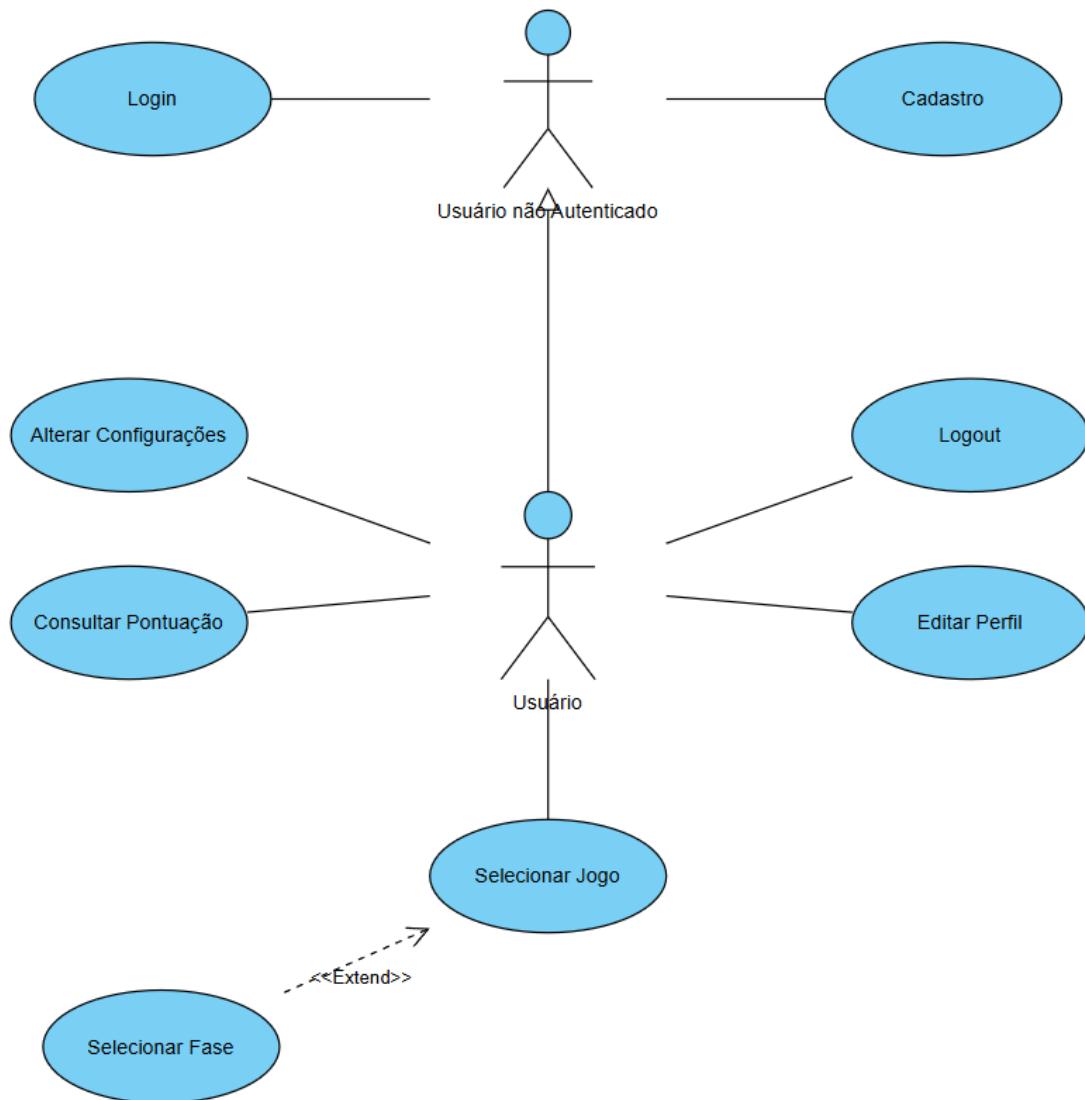
RNF09 – Legislativo

A aplicação deverá obedecer à Lei nº 13.709/2018 – Lei Geral de Proteção de Dados Pessoais (LGPD), em especial aos artigos 7º e 46, que tratam do consentimento para tratamento de dados pessoais e da obrigação de adotar medidas de segurança para proteção das informações dos usuários.

3.4.3 Diagrama de Casos de Uso e Descrição dos Casos de Uso

A seguir (figura 23) é apresentado o diagrama de casos de uso do sistema.

Figura 23 – Diagrama de Caso Uso



Fonte: Autoria própria

Nos Quadros de 3 a 10 são apresentadas as descrições narrativas dos casos de uso, também chamados de casos de uso de baixo nível.

Quadro 3. Caso de uso – Cadastro

Caso de Uso	RF01: CADASTRO
Ator Principal	Usuário não autenticado
Ator Secundário	
Pré-Condição	O usuário não está cadastrado no sistema
Pós-Condição	Usuário Cadastrado
Ações do Ator	Ações do Sistema
1 – O usuário acessa a página de cadastro.	
	2 – O sistema retorna o formulário de cadastro.
3 – O usuário informa os dados obrigatórios (Email e senha) e finaliza cadastro.	
	4 – O sistema valida os dados informados e salva a conta cadastrada no banco de dados, exibindo uma mensagem de sucesso ou de erro.

Fonte: Autoria Própria

Quadro 4. Caso de uso – Login

Caso de Uso	RF02: LOGIN
Ator Principal	Usuário não autenticado
Ator Secundário	
Pré-Condição	O usuário já possui uma conta cadastrada no sistema
Pós-Condição	Usuário logado no sistema
Ações do Ator	Ações do Sistema
1 – O usuário acessa a página de login.	
	2 – O sistema retorna o formulário de login.
3 – O usuário preenche o e-mail e a senha para efetuar o login.	
	4 – O sistema valida as credenciais informadas e exibe uma mensagem de sucesso ou de erro.

Fonte: Autoria Própria

Quadro 5. Caso de uso – Logout

Caso de Uso	RF03: LOGOUT
Ator Principal	Usuário
Ator Secundário	
Pré-Condição	O usuário está autenticado no sistema

Pós-Condição	A sessão do usuário é encerrada
Ações do Ator	Ações do Sistema
1 – O usuário escolhe a opção de 'Sair'.	
	2 – O sistema encerra a sessão e redireciona para a tela de login.

Fonte: Autoria Própria

Quadro 6. Caso de uso – Editar Perfil

Caso de Uso	RF04: EDITAR PERFIL
Ator Principal	Usuário autenticado
Ator Secundário	
Pré-Condição	O usuário deve estar autenticado
Pós-Condição	As informações e os dados do usuário são atualizados
Ações do Ator	Ações do Sistema
1 – O usuário acessa a área do perfil.	
	2 – O sistema retorna a tela do perfil, para que o usuário possa editar os dados.
3 – O usuário altera os dados (usuário, foto de perfil e senha) conforme desejar.	
	4 – O sistema atualiza e valida os dados e no fim, exibe uma mensagem de sucesso ou de erro.

Fonte: Autoria Própria

Quadro 7. Caso de uso – Consultar Pontuação

Caso de Uso	RF05: CONSULTAR PONTUAÇÃO
Ator Principal	Usuário autenticado
Ator Secundário	
Pré-Condição	O usuário deve estar autenticado
Pós-Condição	O usuário visualiza suas maiores pontuações realizadas
Ações do Ator	Ações do Sistema
1 – O usuário acessa a seção de pontuação.	
	2 – O sistema exibe os dados de pontuação conforme o banco de dados.

Fonte: Autoria Própria

Quadro 8. Caso de uso – Alterar Configurações

Caso de Uso	RF06: ALTERAR CONFIGURAÇÕES
Ator Principal	Usuário autenticado
Ator Secundário	
Pré-Condição	O usuário deve estar autenticado
Pós-Condição	As novas configurações de volume são salvas
Ações do Ator	Ações do Sistema

1 – O usuário acessa a tela de configurações.	
	2 – O sistema mostra as opções de volumes que podem ser configuradas.
3 – O usuário ajusta o volume master, ou o volume dos efeitos sonoros, ou se quer tela cheia.	
	4 – O sistema aplica as mudanças e salva as configurações.

Fonte: Autoria Própria

Quadro 9. Caso de uso – Selecionar Jogo

Caso de Uso	RF07: SELECCIONAR JOGO
Ator Principal	Usuário autenticado
Ator Secundário	
Pré-Condição	O usuário deve estar autenticado
Pós-Condição	O sistema carrega o jogo
Ações do Ator	
1 – O usuário acessa a seção de jogos.	
	2 – O sistema retorna a tela de escolha dos jogos.
3 – O usuário escolhe a um dos jogos disponíveis. Se desejar pode escolher uma fase (RF08).	
	4 – O sistema carrega os recursos do jogo escolhido.

Fonte: Autoria Própria

Quadro 10. Caso de uso – Selecionar Fase

Caso de Uso	RF08: SELECCIONAR FASE
Ator Principal	Usuário autenticado
Ator Secundário	
Pré-Condição	O usuário deve estar autenticado, O jogo já deve ter sido iniciado anteriormente
Pós-Condição	O sistema carrega a fase selecionada do jogo, a partir do progresso salvo
Ações do Ator	
1 – O usuário acessa a seção de fases de um jogo já iniciado.	
	2 – O sistema exibe a lista de fases disponíveis de acordo com o progresso do usuário.
3 – O usuário seleciona a fase desejada (já concluída ou em andamento).	
	4 – O sistema carrega os recursos e dados salvos referentes à fase escolhida indo para o RF09.

Fonte: Autoria Própria

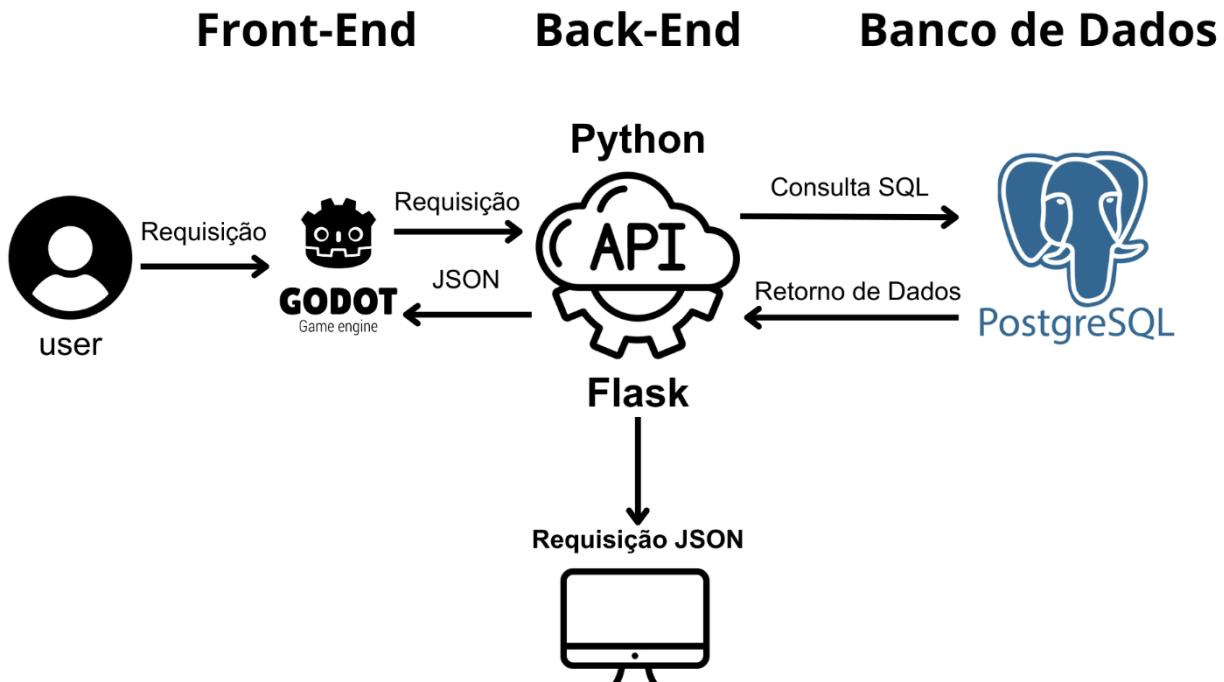
4. Projeto Detalhado do Software

Nos tópicos a seguir, é apresentada a estrutura técnica do Logic Shuffle, descrevendo sua arquitetura, tecnologias empregadas, modelo de dados e principais diagramas de apoio ao desenvolvimento. A modelagem do sistema foi elaborada com base em boas práticas de engenharia de software, priorizando organização modular, facilidade de manutenção e possibilidade de eventual expansão para novos minigames. Além disso, são apresentadas as interfaces do aplicativo, desenvolvidas no Godot Engine, com foco na interatividade, clareza visual e experiência do usuário durante o processo de aprendizagem.

4.1 Arquitetura da aplicação

A arquitetura adotada (figura 24) para o desenvolvimento da aplicação, segue um modelo em camadas, no qual há uma clara separação entre a interface do usuário, a lógica de controle e o armazenamento de dados. O cliente, desenvolvido no Godot Engine utilizando GDScript, é responsável pela execução dos minigames e pela interação com o usuário. A comunicação com o servidor ocorre por meio de requisições HTTP no formato JSON, processadas por um back-end em Python (Flask). Este, por sua vez, realiza o gerenciamento das informações de login e ranking, armazenadas em um banco de dados PostgreSQL.

Figura 24 - Arquitetura do Software



Fonte: Autoria própria

4.2 Tecnologias utilizadas e APIs

O desenvolvimento do Logic Shuffle baseou-se em um conjunto de tecnologias que garantiram desempenho, escalabilidade e integração entre os módulos da aplicação. As ferramentas selecionadas possibilitaram a implementação dos minijogos, a gestão de dados e a comunicação entre as camadas do sistema, assegurando uma arquitetura sólida e eficiente. As principais tecnologias empregadas estão descritas a seguir.

4.2.1 Godot Engine

O Godot Engine⁸ (GDScript / C#) permite a criação de interfaces responsivas e interativas. Suporte a animações e personalização de UI. Integração com scripts em

⁸ Fonte: <https://godotengine.org/pt-br>

GDScript ou C# para manipulação dos minigames.

4.2.2 Python + Flask

A linguagem de programação Python + Flask⁹ é um *framework* leve para o desenvolvimento de aplicações web escrito na linguagem de programação Python. Neste projeto foi responsável pelo gerenciamento de usuários, autenticação e ranking global. Utiliza uma API REST para comunicação com o jogo.

4.2.3 PostgreSQL

O PostgreSQL¹⁰ é um sistema de gestão de base de dados relacional objeto de código aberto (*Open Source Object-Relational Database System - ORDBMS*) que é amplamente utilizado.

4.2.4 SendGrid

O SendGrid¹¹ é um serviço de envio de e-mails em nuvem (*cloud-based email service*), amplamente usado por desenvolvedores e empresas para automatizar o envio de mensagens transacionais e de marketing. Oferece integração via API, autenticação segura e monitoramento de entregas, garantindo confiabilidade nas comunicações do sistema.

⁹ Fonte: <https://flask.palletsprojects.com/en/stable>

¹⁰ Fonte: <https://www.postgresql.org>

¹¹ <https://sendgrid.com/en-us>

4.2.5 SQLAlchemy (ORM)

O SQLAlchemy¹² é um *Object Relational Mapper (ORM)* para a linguagem Python, que facilita a comunicação entre o sistema e o banco de dados relacional. Ele abstrai comandos SQL em classes e objetos Python, simplificando a manipulação de dados. No projeto, o SQLAlchemy será utilizado para realizar consultas, inserções e atualizações de forma estruturada, promovendo maior organização e segurança no acesso ao banco de dados.

4.2.6 Flask-JWT-Extended

O Flask-JWT-Extended¹³ é uma extensão do framework Flask que adiciona suporte à autenticação baseada em *JSON Web Tokens (JWT)*. Essa biblioteca permitirá a implementação de sessões seguras, garantindo que apenas usuários autenticados possam acessar rotas protegidas da API. Será utilizada para gerar, validar e renovar tokens de acesso.

4.2.7 Passlib

A biblioteca Passlib¹⁴ fornece algoritmos de hash seguros para senhas. Ela será utilizada para armazenar credenciais de forma criptografada, impedindo o acesso direto a informações sensíveis. O Passlib oferece suporte a algoritmos modernos, como bcrypt e SHA-256, e é amplamente utilizado em sistemas de autenticação Python.

¹² Disponível em: <https://docs.sqlalchemy.org/en/20/intro.html>

¹³ Disponível em: <https://flask-jwt-extended.readthedocs.io/en/stable>

¹⁴ Disponível em: <https://passlib.readthedocs.io/en/stable>

4.2.8 Python-dotenv

O Python-dotenv¹⁵ é uma biblioteca que permite carregar variáveis de ambiente a partir de um arquivo .env. No contexto do projeto, será responsável por armazenar informações sensíveis, como chaves de API, tokens e credenciais de banco de dados, sem que fiquem expostas no código-fonte. Essa prática aumenta a segurança e facilita a configuração em diferentes ambientes de execução.

4.2.9 UUID

O módulo UUID¹⁶ faz parte da biblioteca padrão do Python e é utilizado para gerar identificadores únicos universais. No projeto, será aplicado para criar *IDs* exclusivos em entidades como usuários e registros de pontuação, evitando colisões e garantindo integridade nas operações de banco de dados.

4.2.10 Hashlib

O Hashlib¹⁷ é um módulo nativo do Python que oferece funções de hash seguras, como SHA-256 e SHA-512. Será empregado para a geração de tokens e validações criptográficas, reforçando a segurança dos processos de autenticação e armazenamento de dados sensíveis.

4.2.11 Godot ConfigFile

O recurso ConfigFile¹⁸ do Godot Engine permite criar e manipular arquivos de configuração em formato .cfg. Ele será utilizado para armazenar informações locais do

¹⁵ Disponível em: <https://pypi.org/project/python-dotenv>

¹⁶ Disponível em: <https://docs.python.org/3/library/uuid.html>

¹⁷ Disponível em: <https://docs.python.org/3/library/hashlib.html>

¹⁸ Disponível em: https://docs.godotengine.org/en/4.4/classes/class_configfile.html

jogo, como progresso do jogador e preferências, sem depender de conexões externas. Essa abordagem garante persistência simples e eficiente dentro do ambiente da Godot.

4.2.12 Godot Audio Engine

O Godot Audio Engine¹⁹ é o sistema de áudio nativo da Godot, responsável por reproduzir sons e trilhas durante a execução do jogo. Ele possibilita o controle de volume, mixagem, efeitos e reprodução espacial, permitindo maior imersão na experiência do usuário. No projeto, será usado para gerenciar efeitos sonoros e música ambiente nas diferentes telas e fases.

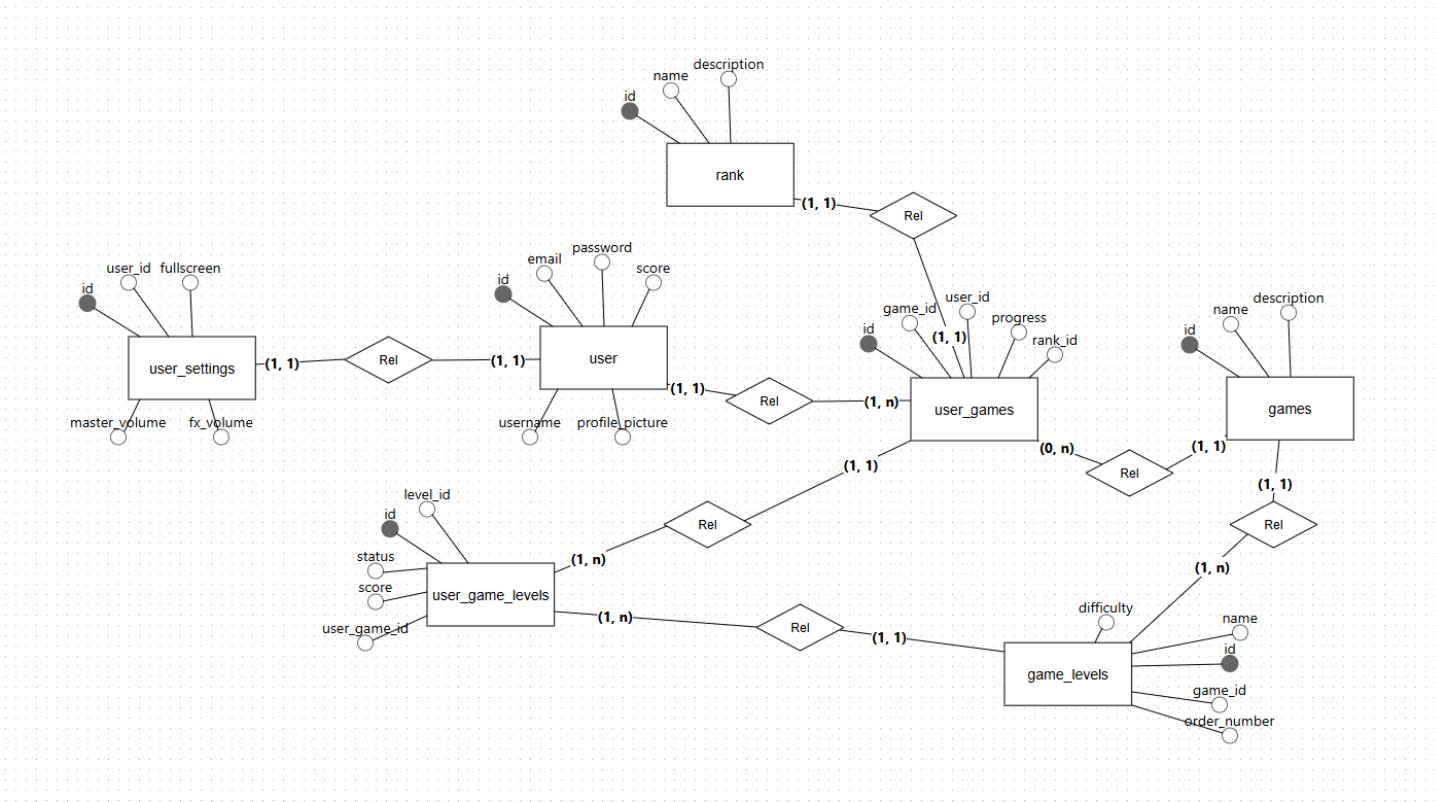
¹⁹ Disponível em: <https://docs.godotengine.org/en/stable/tutorials/audio/index.html>

4.3 Modelo de dados

4.3.1 Modelo Conceitual

O modelo conceitual de banco de dados da aplicação é mostrado na figura 25.

Figura 25 – Modelo Conceitual

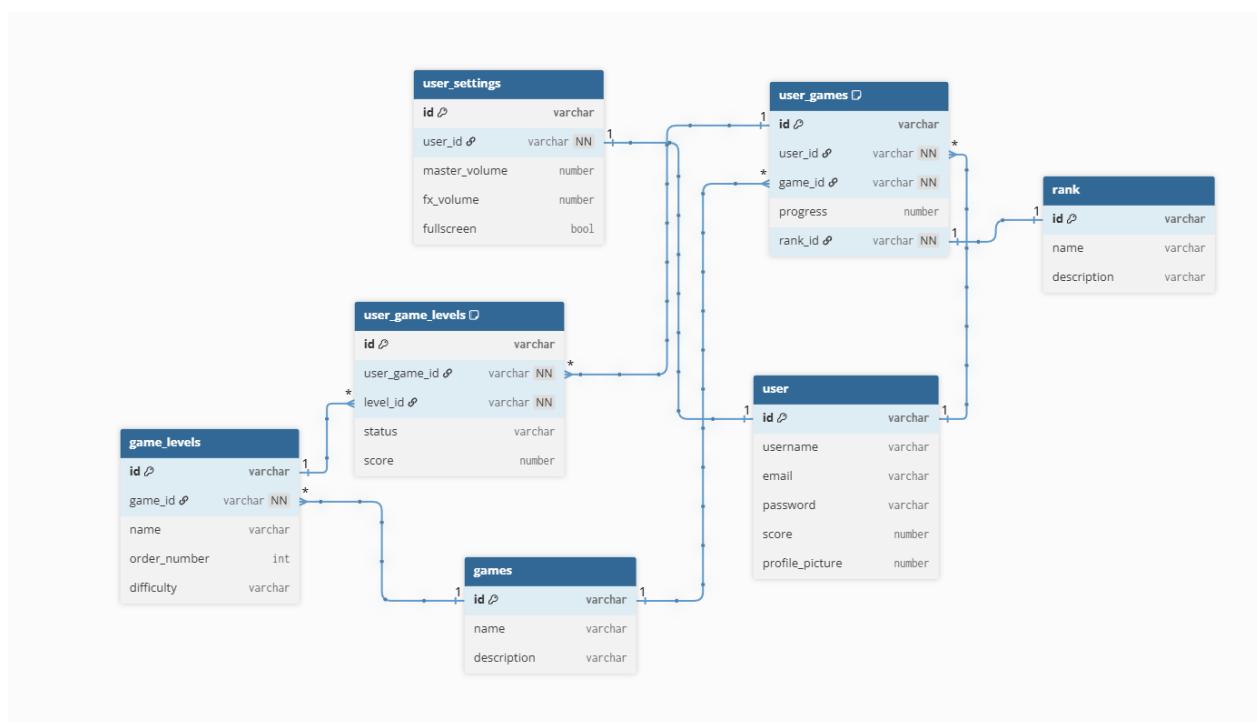


Fonte: Autoria Própria, elaborado com auxílio da ferramenta brmodeloweb

4.3.1 Modelo Lógico

O modelo lógico de banco de dados da aplicação é mostrado na figura 26.

Figura 26 – Modelo Lógico



Fonte: Autoria Própria, elaborado com auxílio da ferramenta dbdiagram.io

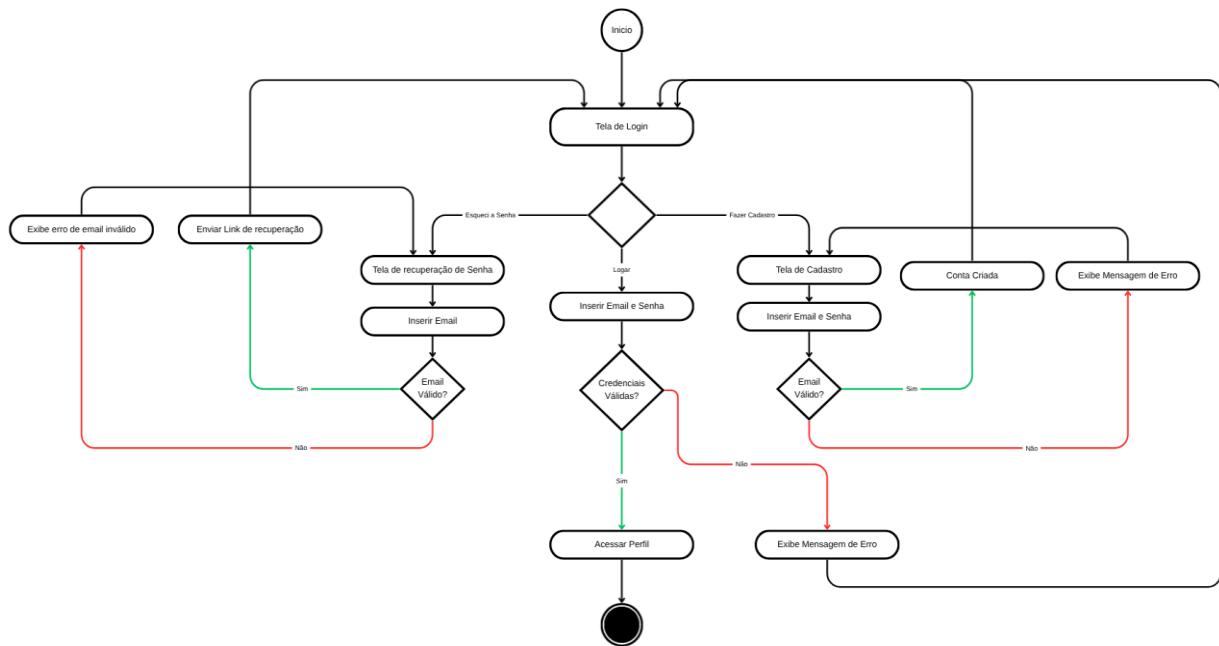
4.4 Diagrama de Atividades

Para representar o comportamento do sistema durante o processo de autenticação e o fluxo posterior de uso da aplicação, foram elaborados dois diagramas de atividades. Esses diagramas descrevem, de forma sequencial, as ações realizadas pelo usuário e pelo sistema, facilitando a compreensão da lógica interna da aplicação e suas transições de estados.

A Figura 27 apresenta o diagrama de atividades referente ao processo de autenticação do usuário, abrangendo as etapas de login, cadastro e recuperação de senha. Já a Figura 28 representa o fluxo pós-autenticação, que contempla a navegação

do usuário pelas funcionalidades principais do sistema, como seleção de minijogos, acesso ao perfil, consulta ao ranking e execução das fases.

Figura 27 – Diagrama de atividades do processo de autenticação do usuário

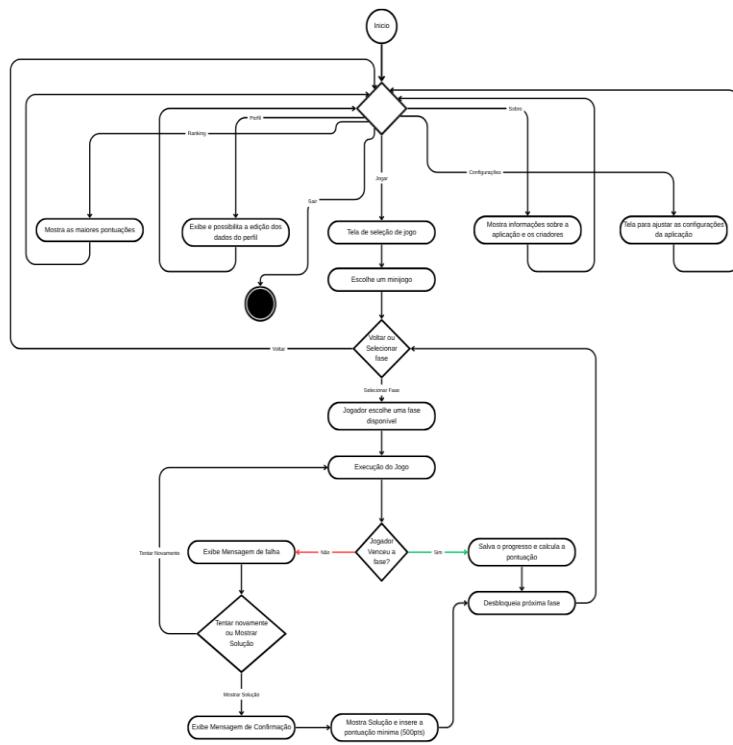


Fonte: Autoria Própria, elaborado com auxílio da ferramenta Canva²⁰

²⁰ Disponível para melhor visualização:

https://www.canva.com/design/DAG4Xe8Bc3o/P18tsd62bCqyluaN6PnruQ/edit?utm_content=DAG4Xe8Bc3o&utm_campaign=designshare&utm_medium=link2&utm_source=sharebutton

Figura 28 – Diagrama de atividades do fluxo pós-autenticação



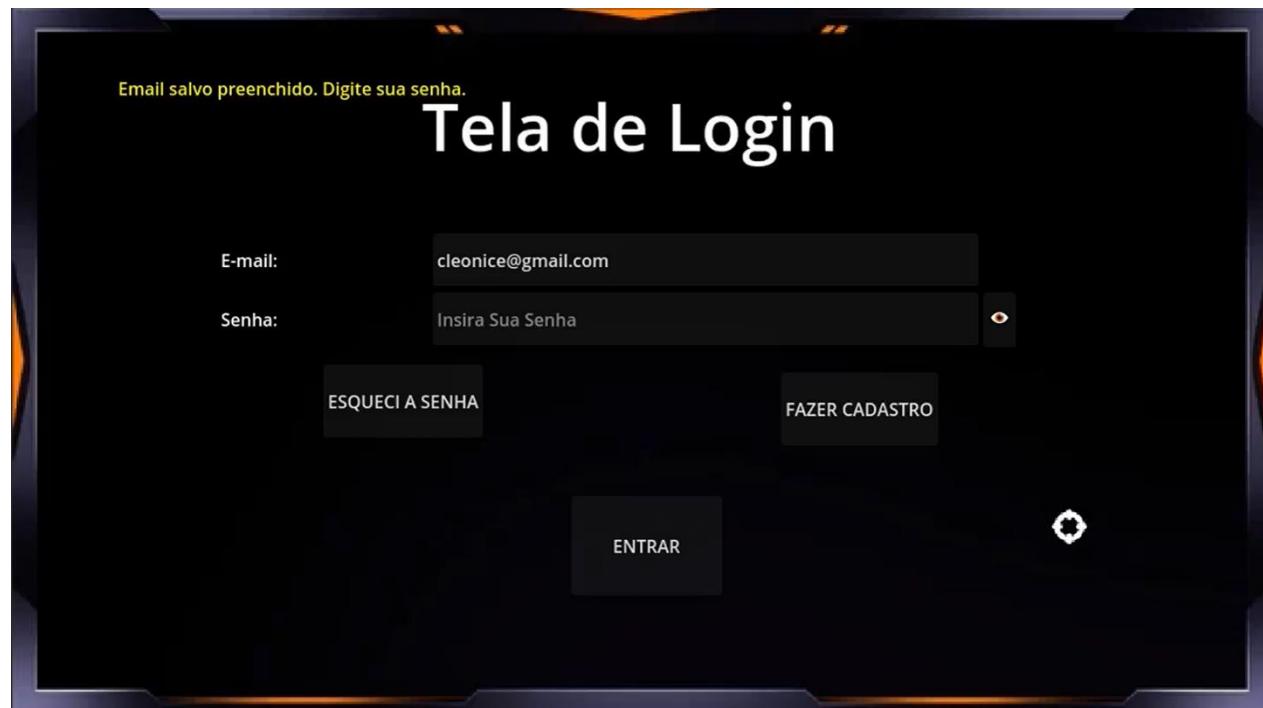
Fonte: Autoria Própria, elaborado com auxílio da ferramenta Canva

4.5 Interfaces com o usuário

Nesse item são apresentadas as interfaces da aplicação.

A Figura 29 apresenta a tela de autenticação do sistema, que serve como principal porta de entrada para a aplicação. Nela, o usuário deve inserir suas credenciais (E-mail e Senha) e pressionar "ENTRAR" para acessar sua conta. A interface também oferece dois fluxos de navegação alternativos: o botão "FAZER CADASTRO" direciona o usuário para a tela de criação de uma nova conta, e a opção "ESQUECI A SENHA" inicia o processo de recuperação de acesso.

Figura 29 – Tela de Login



Fonte: Autoria Própria

Dando sequência ao fluxo de autenticação, a Figura 30 ilustra a "Tela de Cadastro", acessada ao selecionar a opção "FAZER CADASTRO" na tela de login (Figura 29). Nesta interface, o novo usuário deve preencher os campos obrigatórios para a criação de sua conta: "Nome do usuário", "E-mail", "Senha" e a confirmação da senha. A tela também dispõe de um ícone de "voltar" no canto superior esquerdo, permitindo ao usuário abandonar o cadastro e retornar à tela anterior. Após o preenchimento dos dados, o usuário finaliza o processo clicando no botão "CRIAR CONTA".

Figura 30 – Tela de Cadastro



Fonte: Autoria Própria

A Figura 31 exibe a tela de "Recuperação de Senha", que é apresentada ao usuário caso ele selecione a opção "ESQUECI A SENHA" na tela de login (Figura 29). Para iniciar o processo de redefinição, o usuário deve informar o "E-mail" associado à sua conta no campo designado e, em seguida, clicar no botão "RECUPERAR CONTA", assim será enviado um e-mail para continuar com o processo de recuperação. Similarmente à tela de cadastro, um ícone de "voltar" está posicionado no canto superior esquerdo, permitindo ao usuário cancelar a operação e retornar à tela de login.

Figura 31 – Tela de Recuperação de Senha



Fonte: Autoria Própria

Após o usuário se autenticar com sucesso no sistema (Figura 29), ele é direcionado para a tela principal, apresentada na Figura 32. Esta interface centraliza as principais funcionalidades da aplicação. O usuário tem acesso à navegação principal através dos botões "JOGAR", "CONFIGURAÇÕES", "SOBRE O JOGO" e "SAIR", que permite encerrar a aplicação. Adicionalmente, a tela exibe atalhos visuais para "Editar Perfil", representado pelo ícone de avatar, e "Ver Ranking", representado pelo ícone de troféu, permitindo um acesso rápido a essas seções.

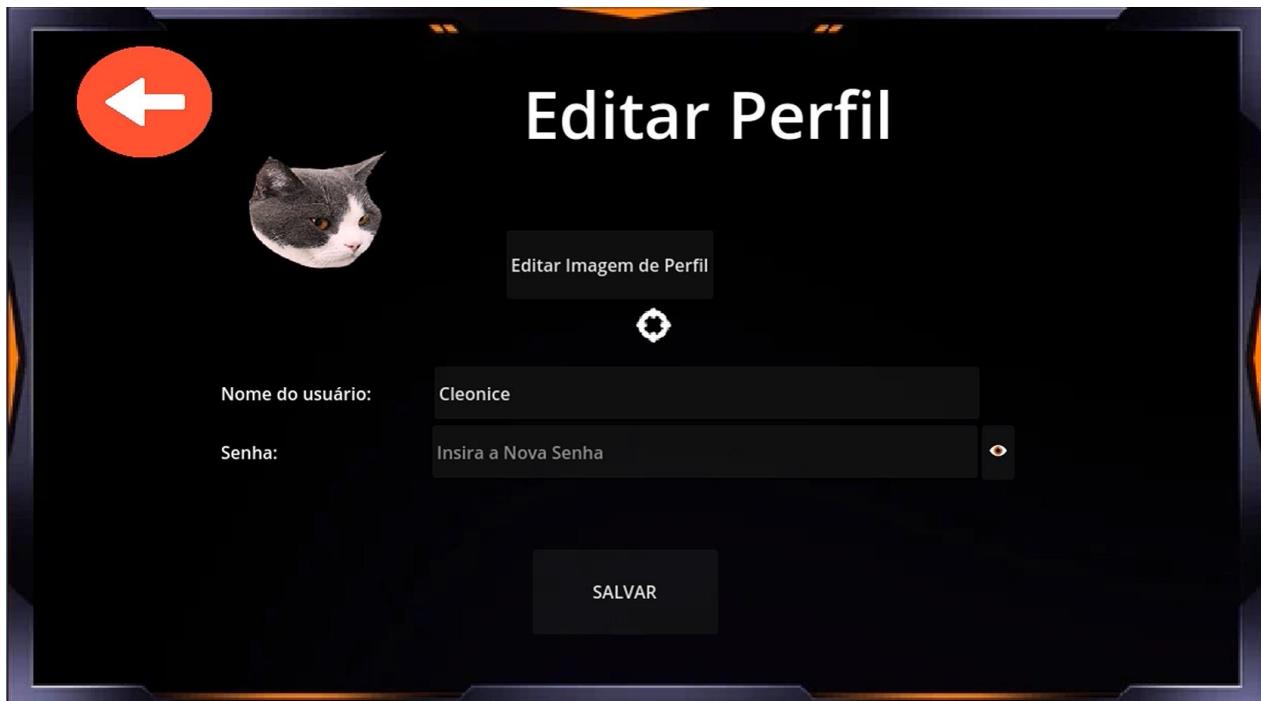
Figura 32 – Tela Inicial



Fonte: Autoria Própria

A Figura 33 demonstra a tela "Editar Perfil", acessada a partir do atalho correspondente na tela principal (Figura 32). Nesta interface, o usuário pode gerenciar as informações da sua conta, permitindo alterar a imagem de avatar ao clicar em "Editar Imagem de Perfil", modificar seu "Nome do usuário" e redefinir sua senha através do campo "Insira a Nova Senha". Após realizar as alterações desejadas, o usuário deve clicar no botão "SALVAR" para que as modificações sejam registradas. O ícone de "voltar" no canto superior esquerdo permite cancelar a operação e retornar à tela principal.

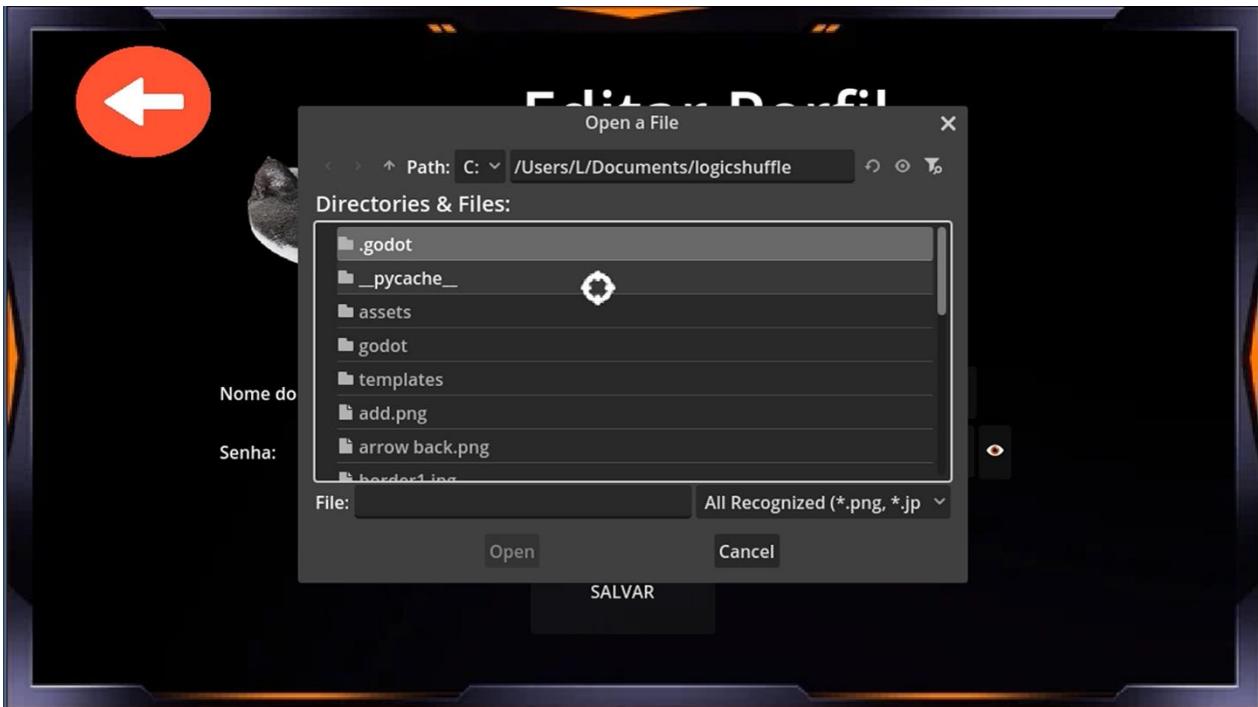
Figura 33 – Tela de Edição de Perfil



Fonte: Autoria Própria

A Figura 34 exibe a interface interna do Godot para a seleção de imagem de perfil, que é acionada quando o usuário clica em "Editar Imagem de Perfil" (Figura 33). Diferente de um seletor de arquivos padrão do sistema operacional, esta tela apresenta ao usuário uma lista de avatares pré-definidos (*.png, *.jpg) que foram incluídos na aplicação. O usuário pode, então, escolher uma das imagens disponíveis na lista e clicar em "Open" para confirmar a alteração do seu avatar, ou "Cancel" para retornar à tela anterior sem fazer mudanças.

Figura 34 – Seletor de Avatar



Fonte: Autoria Própria

A Figura 35 exibe a tela de "Ranking Global", acessada pelo atalho de troféu ("Ver Ranking") na tela principal (Figura 32). A interface apresenta uma lista classificatória com os cinco melhores jogadores do sistema, exibindo a "Posição", o avatar e nome do "Jogador", a "Pontuação" total e o número de "Níveis Completados". O usuário só aparecerá nesta lista após registrar sua primeira pontuação, garantindo que o ranking seja composto apenas por jogadores ativos. Uma barra de rolagem à direita permite a navegação vertical. Para sair desta tela e retornar ao menu principal, o usuário utiliza o ícone de "voltar" no canto superior esquerdo.

Figura 35 – Tela de Ranking Global

The screenshot shows a dark-themed mobile game interface titled "Ranking Global". In the top left corner is a red circular button with a white arrow pointing left. The table lists four players:

Posição	Jogador	Pontuação	Níveis Completados
1° 🥇	Camila	12500	20
2° 🥈	Eros	4000	5
3° 🥉	Gabriel	2000	2
4º	Lucas	2000	2

Fonte: Autoria Própria

A Figura 36 mostra a tela de "Configurações de Jogo", acessada ao clicar no botão "CONFIGURAÇÕES" na tela principal (Figura 32). Esta interface permite ao usuário ajustar sua experiência, oferecendo controles deslizantes para o "Volume da música" e o "Volume dos efeitos sonoros". Além disso, é possível alterar o "Estilo de tela" através da opção "Tela Cheia". O usuário pode confirmar suas escolhas clicando em "Salvar Configurações" ou reverter para os valores originais selecionando "Voltar para Configuração Padrão". O ícone de "voltar" no canto superior esquerdo permite o retorno à tela principal.

Figura 36 – Tela de Configurações



Fonte: Autoria Própria

A Figura 37 exibe a tela "Sobre o Jogo", acessada ao selecionar a opção "SOBRE O JOGO" no menu principal (Figura 32). Esta interface apresenta informações contextuais sobre o projeto, descrevendo-o como uma "Jornada Educativa nas Estruturas de Dados" e seu objetivo de transformar o aprendizado em uma experiência interativa. Logo abaixo, uma seção dedicada aos "Criadores" apresenta os desenvolvedores do projeto. Uma barra de rolagem vertical à direita indica que há mais conteúdo informativo, e o ícone de "voltar" no canto superior esquerdo permite o retorno à tela principal.

Figura 37 – Tela Sobre o Jogo



Fonte: Autoria Própria

Retornando à tela principal (Figura 32) e selecionando a opção "JOGAR", o usuário é direcionado para a tela de seleção de jogos, conforme ilustrado na Figura 38. Esta interface apresenta os minijogos disponíveis no Logic Shuffle, com as opções "Dungeon Cave Quest" e "Pyramid Maker" apresentadas como botões de seleção. O usuário pode iniciar um dos jogos clicando em seu respectivo nome ou retornar ao menu principal através do ícone de "voltar".

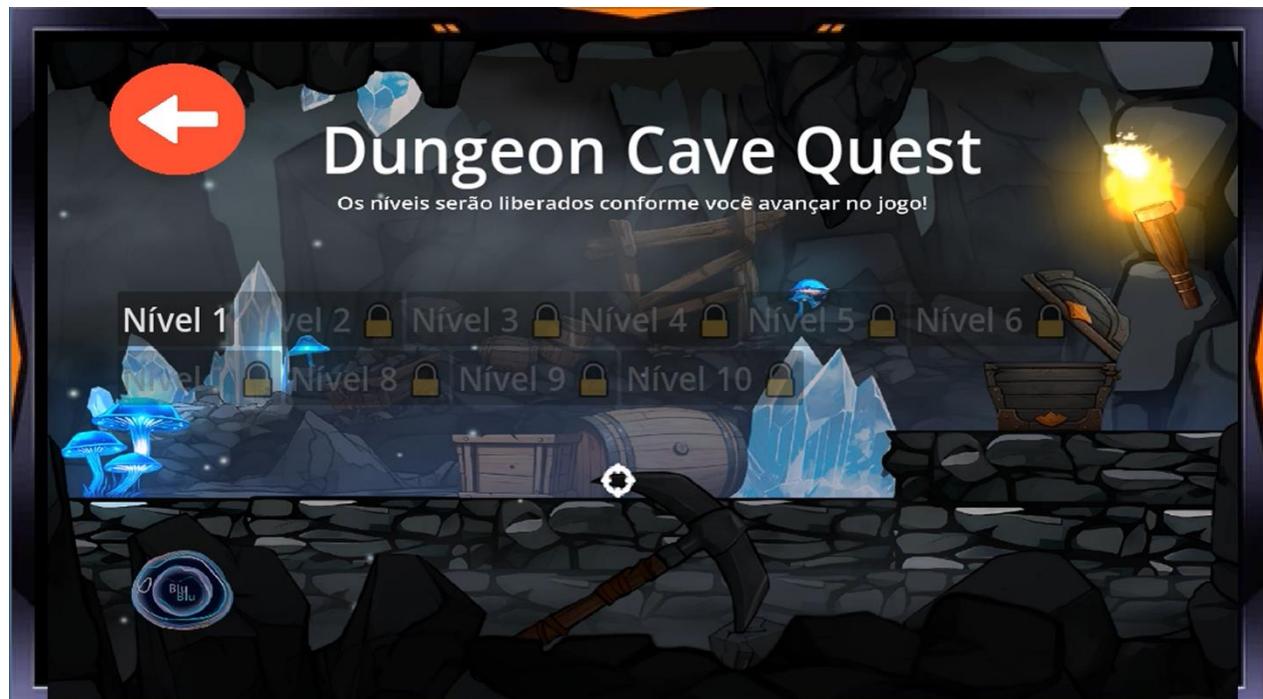
Figura 38 – Tela de Seleção de Jogo



Fonte: Autoria Própria

Ao selecionar o minijogo "Dungeon Cave Quest" na tela anterior (Figura 38), o usuário é apresentado à tela de seleção de fases, mostrada na Figura 39. Esta interface exibe os dez níveis disponíveis para este jogo. O "Nível 1" está liberado para seleção, enquanto os níveis subsequentes (Nível 2, Nível 3, etc.) são representados por um ícone de cadeado, indicando que ainda estão bloqueados. A tela informa ao jogador que "Os níveis serão liberados conforme você avançar no jogo!", estabelecendo a regra de progressão. O usuário pode iniciar o jogo clicando em um nível disponível ou retornar à tela de seleção de jogo (Figura 38) através do ícone de "voltar".

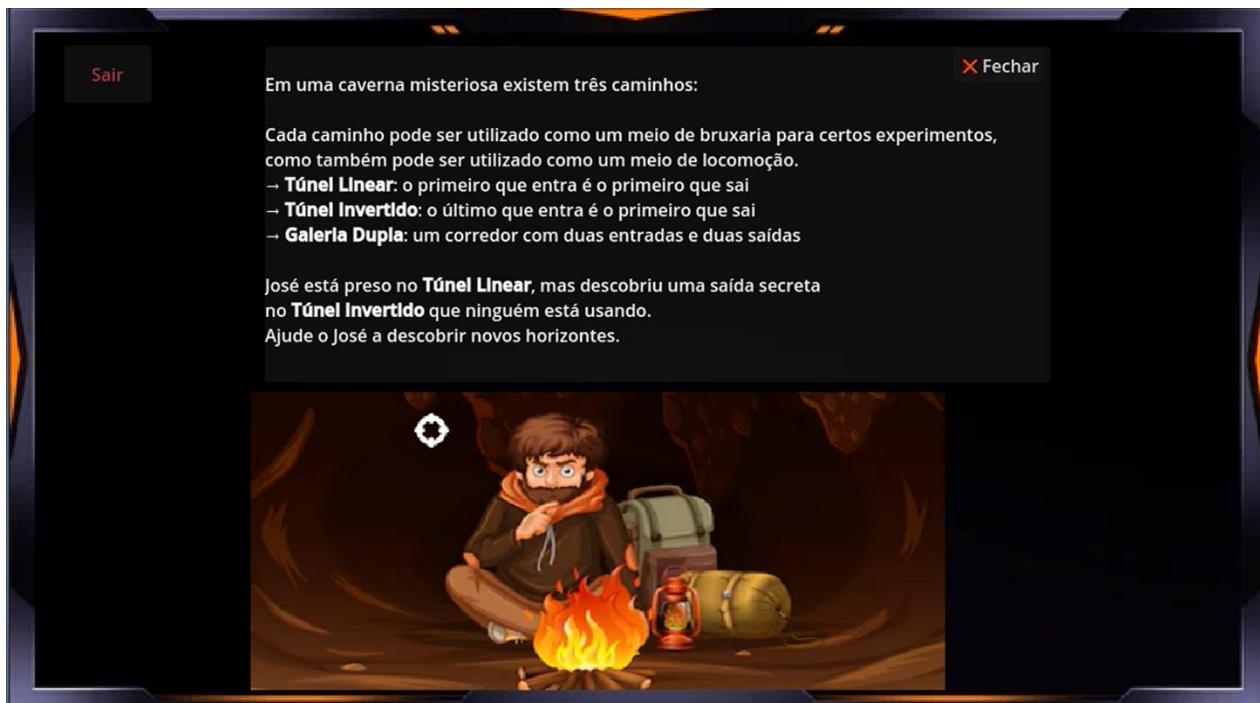
Figura 39 – Tela de Seleção de Fases (Dungeon Cave Quest)



Fonte: Autoria Própria

Após selecionar o "Nível 1" na tela anterior (Figura 39), o usuário é apresentado a uma tela de introdução, conforme mostra a Figura 40. Este padrão se repete no início de cada fase, exibindo uma caixa de texto que apresenta a narrativa e os conceitos específicos daquele nível. O usuário pode clicar em "Fechar" para iniciar a fase ou em "Sair" para retornar à seleção de fases. As telas de introdução dos níveis subsequentes seguem este mesmo formato e encontram-se no Apêndice B.

Figura 40 – Tela de Introdução (Nível 1)



Fonte: Autoria Própria

Ao fechar a tela de introdução (Figura 40), o usuário inicia o Nível 1, cuja interface de jogo é apresentada na Figura 41. Esta interface serve como modelo para a jogabilidade principal, seguindo um padrão que se repete nos demais níveis. A tela é dividida em áreas que representam as estruturas de dados introduzidas na história: a "Galeria Dupla (G.D)" (Deque), o "Túnel Linear (T.L)" (Fila) e o "Túnel Invertido (T.I)" (Pilha). O objetivo do jogador é manipular os itens (como gemas, moedas e o personagem) entre as estruturas de origem, usando os botões de adicionar (+) e mover (->), para replicar a sequência exata de itens mostrada na estrutura de destino (T.I). A interface exibe o "Tempo" e a "Pontuação" atuais, e o jogador pode submeter sua resposta clicando em "Enviar Solução" ou abandonar a fase através do botão "Sair". Há também um botão no canto superior direito com um símbolo de aviso, que ao ser clicado reexibe a história do nível (figura 40). As telas dos níveis subsequentes, que seguem esta mesma estrutura, encontram-se no Apêndice B.

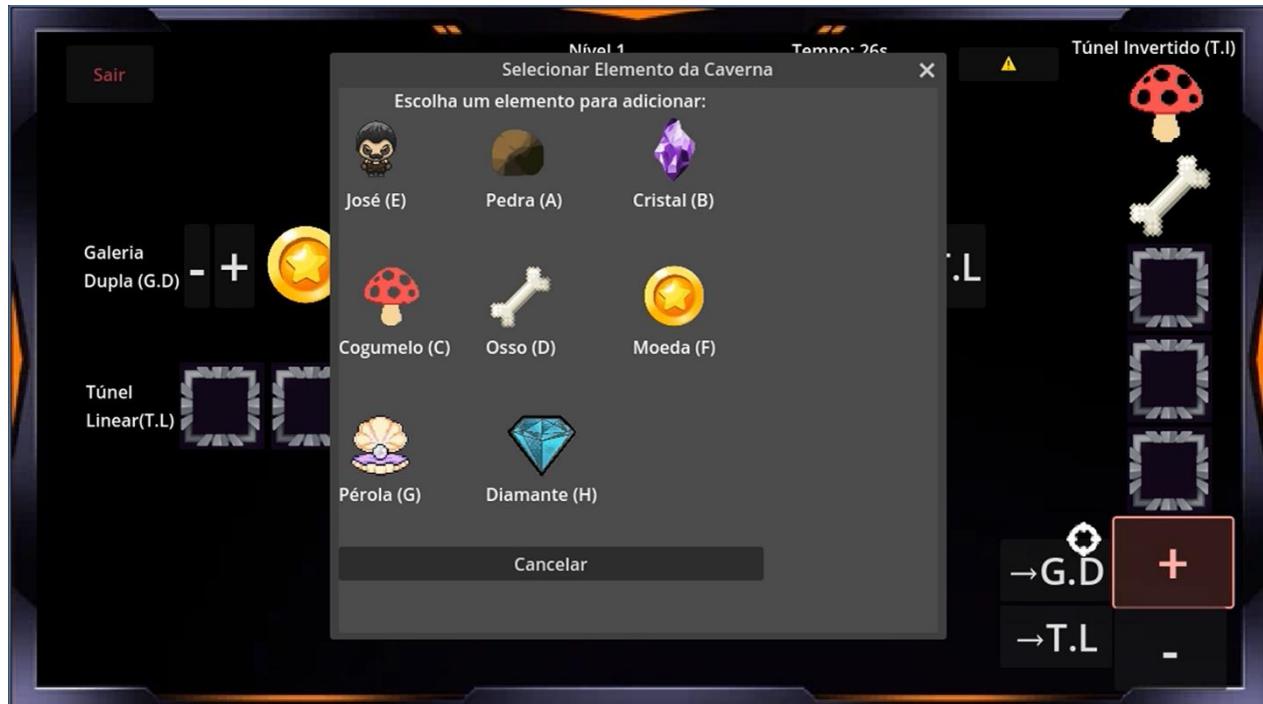
Figura 41 – Interface de Jogo (Nível 1)



Fonte: Autoria Própria

A Figura 42 detalha a janela pop-up "Selecionar Elemento da Caverna". Esta tela é exibida quando o jogador, durante a partida (Figura 41), clica em um dos botões de adicionar ("+"). A interface apresenta um painel com todos os itens que podem ser manipulados no nível, como "José (E)", "Pedra (A)", "Cristal (B)", "Cogumelo (C)", "Osso (D)", entre outros. O jogador deve então selecionar um desses elementos para adicioná-lo à respectiva estrutura de dados. Caso deseje desistir da ação, o usuário pode clicar no botão "Cancelar" ou no ícone 'x' no canto superior direito da janela para fechá-la.

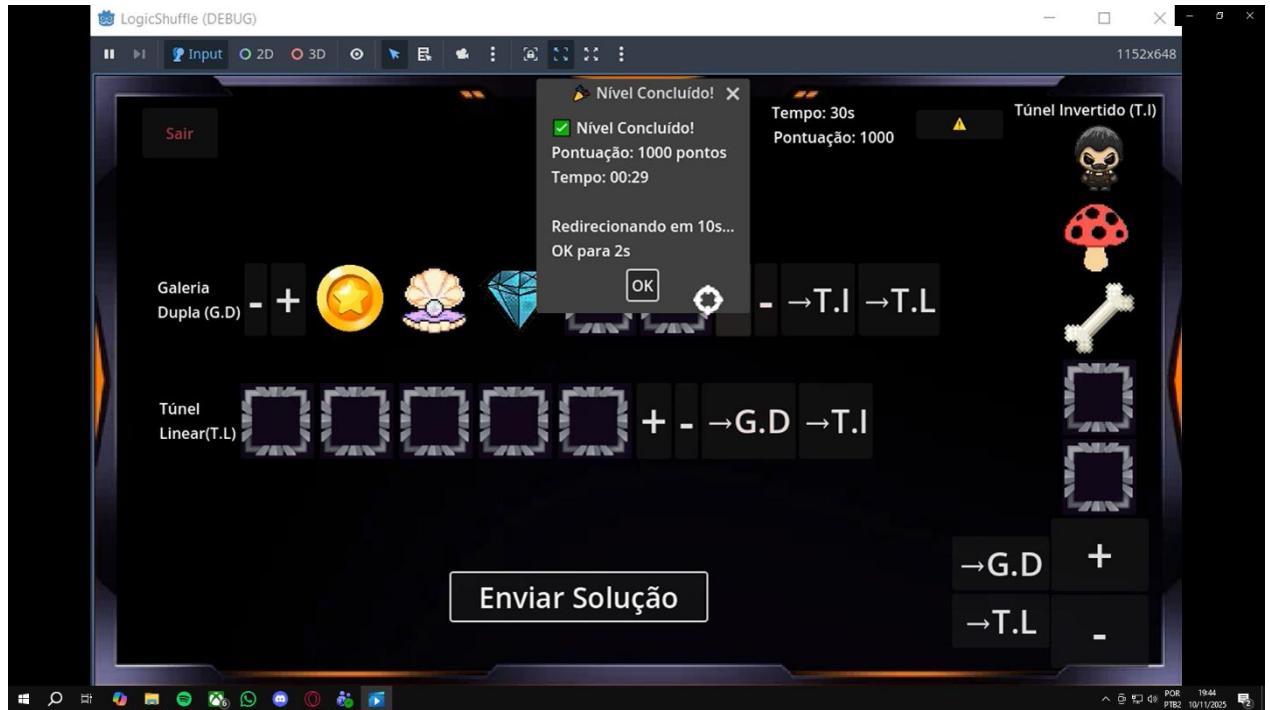
Figura 42 – Janela de Seleção de Elementos



Fonte: Autoria Própria

A Figura 43 mostra a janela pop-up "Nível Concluído!" que é exibida após o jogador submeter uma solução correta através do botão "Enviar Solução" (Figura 41). Esta janela fornece um feedback imediato de sucesso, informando a pontuação e o tempo registrados para a conclusão da fase. A mensagem também indica um redirecionamento automático para a tela de seleção de fase (figura 39) após um contador de 10 segundos, ou o jogador pode clicar em "OK" para avançar imediatamente.

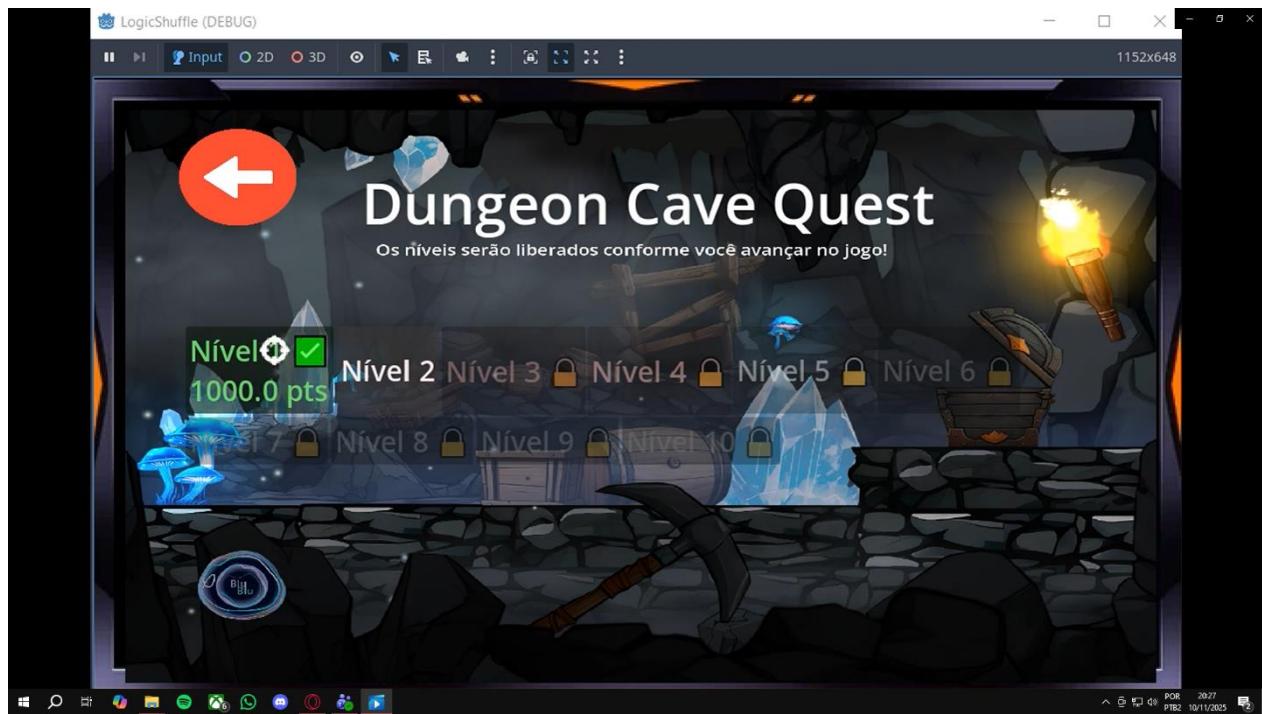
Figura 43 – Janela de Nível Concluído



Fonte: Autoria Própria

A Figura 44 ilustra a tela de seleção de fases atualizada, exibida após a conclusão bem-sucedida do Nível 1 (Figura 43). A interface fornece um feedback visual claro da progressão do jogador: o "Nível 1" agora exibe um ícone de verificação (\checkmark) e a pontuação obtida. Seguindo a regra de progressão do jogo, o "Nível 2", que anteriormente estava bloqueado (Figura 39), encontra-se agora desbloqueado e disponível para seleção. O jogador pode, então, optar por avançar para o próximo desafio ou repetir a fase já concluída.

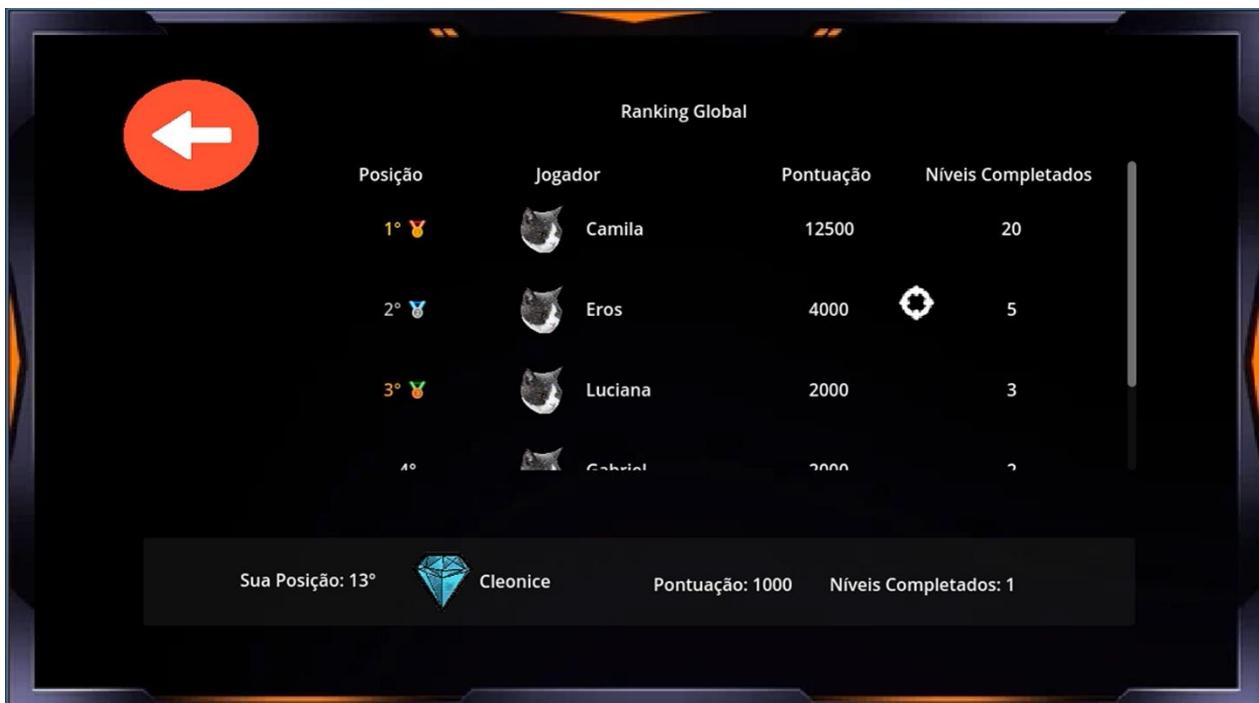
Figura 44 – Tela de Progressão de Fases



Fonte: Autoria Própria

A Figura 45 ilustra a tela de "Ranking Global" após o jogador ter concluído pelo menos um nível e registrado sua primeira pontuação. A interface, acessada pela tela principal (Figura 32), continua a exibir a lista dos cinco melhores jogadores no topo. Adicionalmente, ela agora apresenta uma nova seção na parte inferior, que destaca a classificação pessoal do usuário logado. Esta seção informa a "Sua Posição" no ranking geral, o nome do "Jogador", a "Pontuação" acumulada e o número de "Níveis Completados", permitindo que o jogador acompanhe seu próprio progresso em relação aos demais.

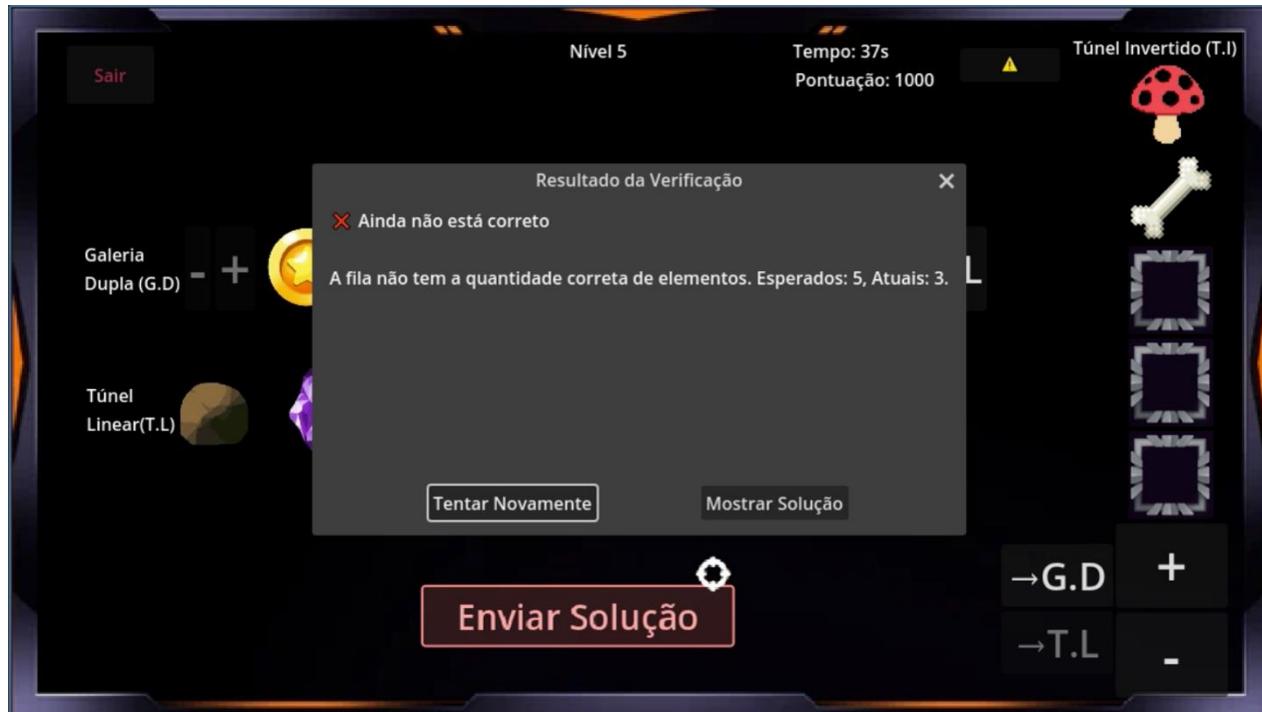
Figura 45 – Ranking Global com Posição do Usuário



Fonte: Autoria Própria

A Figura 46 apresenta a janela "Resultado da Verificação", que é exibida caso o jogador clique em "Enviar Solução" (Figura 41) e a resposta esteja incorreta. O pop-up informa que "Ainda não está correto" e fornece um feedback específico sobre o erro, como por exemplo, "A fila não tem a quantidade correta de elementos. Esperados: 5. Atuais: 3.". A interface oferece ao jogador duas opções: "Tentar Novamente", que fecha a janela e permite ao usuário continuar a manipulação dos itens para corrigir sua solução, ou "Mostrar Solução", que revela a resposta correta para o nível.

Figura 46 – Janela de Verificação de Erro



Fonte: Autoria Própria

Caso o jogador opte por "Mostrar Solução" na janela de erro (Figura 46), o sistema apresenta uma tela de confirmação, como visto na Figura 47. Esta janela, intitulada "Confirmar Mostrar Solução", alerta o usuário sobre a consequência de sua escolha, informando que "Ver a solução reduzirá sua pontuação máxima para 500 pontos.". O usuário deve então decidir se aceita a penalidade clicando em "Sim, Mostrar Solução", o que revelará a resposta correta e definirá sua pontuação para este valor mínimo, ou pode clicar em "Cancelar" para fechar a janela e retornar ao desafio.

Figura 47 – Janela de Confirmação para Mostrar Solução



Fonte: Autoria Própria

A Figura 48 demonstra o estado final da tela de seleção de fases do "Dungeon Cave Quest", que é exibida após o jogador ter concluído todos os desafios disponíveis. Similar às telas de progressão anteriores (Figuras 39 e 44), esta interface agora exibe todos os níveis, do 1 ao 10, como desbloqueados. Cada nível concluído é marcado com um ícone de verificação (✓) e exibe a pontuação final registrada pelo jogador. Neste estado, o jogador pode optar por revisitar e jogar novamente qualquer um dos níveis ou retornar à tela de seleção de jogo (Figura 38) através do ícone de "voltar".

Figura 48 – Tela de Seleção de Fases (Todos os Níveis Concluídos)



Fonte: Autoria Própria

A Figura 49 exibe a tela de seleção de fases do segundo minijogo, "Pyramid Maker", que é acessada ao selecioná-lo na tela de jogos (Figura 38). De forma análoga ao "Dungeon Cave Quest", esta interface apresenta os dez níveis do desafio, dispostos sobre uma imagem temática de pirâmides e uma esfinge. O "Nível 1" está disponível para o jogador iniciar, enquanto os níveis subsequentes são marcados com um ícone de cadeado, indicando que estão bloqueados. A mesma regra de progressão é informada ao usuário: "Os níveis serão liberados conforme você avançar no jogo!". O ícone de "voltar" permite retornar à tela de seleção de jogo.

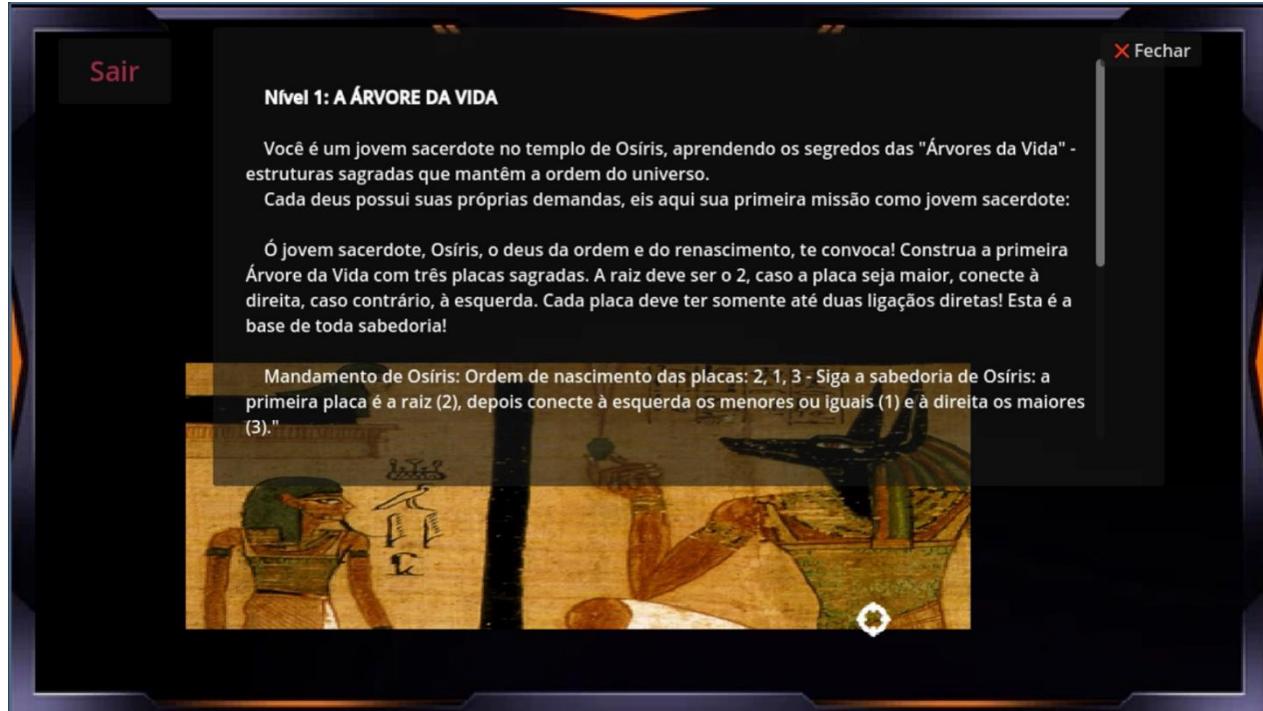
Figura 49 – Tela de Seleção de Fases (Pyramid Maker)



Fonte: Autoria Própria

Ao selecionar o "Nível 1" na tela anterior (Figura 49), o usuário é apresentado à tela de introdução do minijogo, conforme mostra a Figura 50. Seguindo o padrão de introdução narrativa, esta tela apresenta a premissa do desafio, intitulado "A Árvore da Vida". O texto introduz ludicamente o conceito de Árvores Binárias de Busca, instruindo o jogador, na pele de um "jovem sacerdote", a construir uma estrutura onde a raiz é o número 2. O usuário pode clicar em "Fechar" para iniciar a fase ou em "Sair" para retornar à seleção de níveis. As telas de introdução dos níveis subsequentes deste jogo seguem esta mesma estrutura e encontram-se no Apêndice C.

Figura 50 – Tela de Introdução (Pyramid Maker Nível 1)



Fonte: Autoria Própria

Ao fechar a tela de introdução (Figura 50), o usuário acessa a interface de jogo do "Pyramid Maker", ilustrada na Figura 51. Esta tela apresenta uma grande área de construção em cinza, onde o jogador deve montar a "Árvore da Vida" conforme as regras do nível. No topo, uma "Barra de Ferramentas" disponibiliza as ações necessárias para a manipulação da estrutura, incluindo "Add" (Adicionar placa), "Conectar" (ligar placas), "Trocar" (alterar valor), "Del" (Deletar) e "Clear" (Limpar tela). Uma mensagem de instrução ("Clique na área para adicionar uma placa...") guia a ação inicial. A interface exibe o "Tempo" e "Pontuação" atuais, e o jogador submete sua estrutura final através do botão "Enviar Solução". O botão "Sair" permite abandonar a fase. As telas de jogo dos níveis subsequentes deste minijogo seguem este mesmo padrão e encontram-se no Apêndice C.

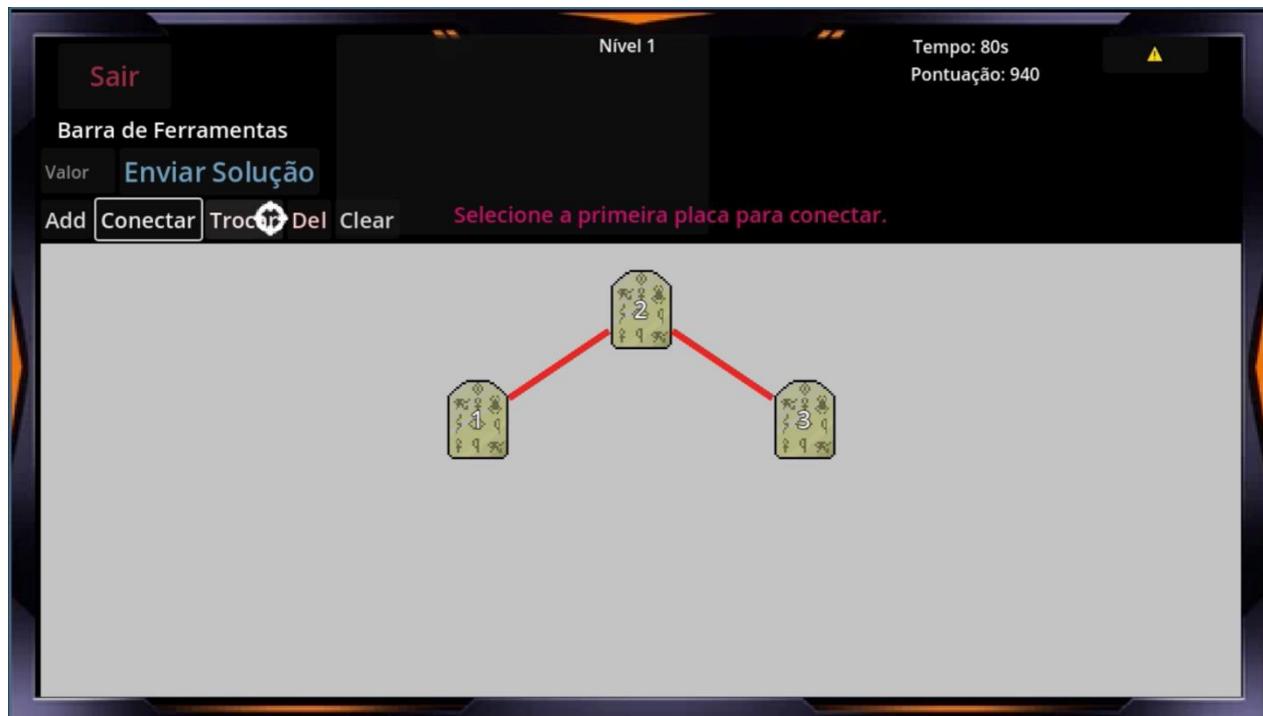
Figura 51 – Interface de Jogo (Pyramid Maker Nível 1)



Fonte: Autoria Própria

A Figura 52 demonstra a mecânica de jogo do "Pyramid Maker" em ação. Utilizando as ferramentas da barra superior (Figura 51), o jogador adicionou ("Add") e conectou ("Conectar") as placas com os valores "2", "1" e "3" na área de construção. A imagem exemplifica a criação da estrutura de árvore binária, com a placa "2" servindo como raiz, a placa "1" (menor) conectada à esquerda e a placa "3" (maior) conectada à direita, conforme as regras estabelecidas na introdução do nível (Figura 50).

Figura 52 – Exemplo de Construção da Árvore (Pyramid Maker)



Fonte: Autoria Própria

A Figura 53 ilustra a tela de seleção de fases do "Pyramid Maker" após a conclusão bem-sucedida do primeiro nível. Assim como no minijogo anterior, a interface fornece um feedback visual de progressão: o "Nível 1" exibe um ícone de verificação (✓) e a pontuação obtida. Consequentemente, o "Nível 2" foi desbloqueado, e seu ícone de cadeado removido, permitindo ao jogador continuar sua jornada.

Figura 53 – Tela de Progressão de Fases (Pyramid Maker)



Fonte: Autoria Própria

A Figura 54 apresenta a janela "Resultado da Verificação", exibida quando o jogador submete uma solução incorreta. De forma análoga ao minijogo "Dungeon Cave Quest" (Figura 46), o pop-up informa que "Ainda não está correto" e fornece um feedback contextual sobre o erro (ex: "Thoth exige exatamente 3 placas sagradas! Você tem 0 placas."). O jogador pode então optar por "Tentar Novamente" ou "Mostrar Solução".

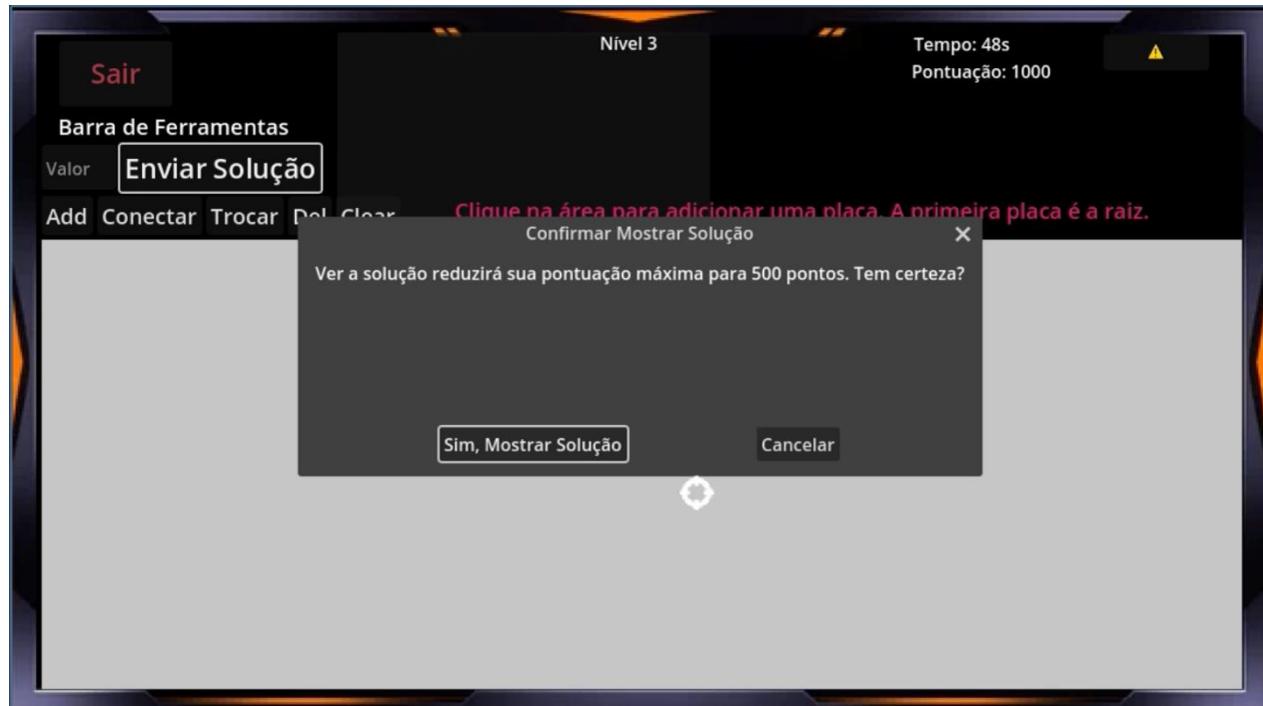
Figura 54 – Janela de Verificação de Erro (Pyramid Maker)



Fonte: Autoria Própria

Caso o jogador clique em "Mostrar Solução" na janela de erro (Figura 54), o sistema exibe a tela de confirmação vista na Figura 55. Esta janela, intitulada "Confirmar Mostrar Solução", alerta o usuário sobre a penalidade, informando que "Ver a solução reduzirá sua pontuação máxima para 500 pontos.". O jogador deve confirmar a ação clicando em "Sim, Mostrar Solução", aceitando a pontuação mínima, ou "Cancelar" para retornar ao desafio.

Figura 55 – Janela de Confirmação para Mostrar Solução (Pyramid Maker)



Fonte: Autoria Própria

5. Implantação

Nesta etapa são apresentados os procedimentos necessários para disponibilizar o Logic Shuffle em ambiente de execução, permitindo que o usuário final possa interagir com a aplicação de maneira estável e funcional. Esta fase apresenta a configuração dos ambientes, a definição dos métodos de instalação, a estimativa de custos e a realização de testes de verificação e validação do software.

O sistema foi preparado para execução em dois formatos distintos: uma versão offline, destinada a usuários finais e demonstrações locais, e uma versão online, voltada para desenvolvimento, testes completos e integração entre os módulos front-end e back-end

Durante o processo de desenvolvimento, estava previsto que o banco de dados fosse implantado em ambiente de nuvem, a fim de permitir acesso remoto e centralizado às informações da aplicação. Entretanto, devido à limitação de tempo disponível para a finalização do projeto, essa etapa não pôde ser concluída. Assim, a versão atual opera com o banco de dados hospedado localmente, o que não compromete as funcionalidades do sistema, mas restringe temporariamente seu uso simultâneo em diferentes dispositivos.

5.1 Instalação e repositórios

O código-fonte final do projeto encontra-se hospedado no repositório público do GitHub, disponível em: <https://github.com/filiwili/LogicShuffle>

Na versão offline, recomendada para uso final, a execução ocorre de forma independente, sem necessidade de instalação de dependências externas ou acesso à internet. Essa abordagem visa facilitar a distribuição e o uso da aplicação em ambientes acadêmicos e de demonstração.

O processo de utilização é simples e pode ser realizado em qualquer computador com sistema operacional Windows 7 ou superior. Para iniciar a aplicação, o usuário deve

acessar a pasta LogicShuffleOffline, executar o arquivo LogicShuffle.exe e aguardar o carregamento automático do jogo. Essa versão apresenta como principais vantagens o funcionamento completo em modo local e a ausência de necessidade de instalação, tornando-se ideal para apresentações rápidas ou utilização em locais sem conexão de rede.

A versão online, por sua vez, foi projetada para o ambiente de desenvolvimento e testes, permitindo a integração total entre o banco de dados, o servidor e a interface gráfica. Essa configuração possibilita a execução de funcionalidades que envolvem autenticação de usuários, ranking global e persistência de pontuação.

Para sua implantação, são necessários os seguintes pré-requisitos: Python 3.8 ou superior, PostgreSQL 12 ou superior e Godot Engine 4.0 ou superior (este último apenas para modificações no front-end). O processo inicia-se pela configuração do banco de dados, realizada através da criação da base logicshuffle no PostgreSQL, conforme os comandos abaixo:

```
CREATE DATABASE logicshuffle;  
\c logicshuffle;
```

Em seguida, procede-se à configuração do back-end. O desenvolvedor deve acessar a pasta LogicShuffleOnline, instalar as dependências listadas no arquivo *requirements.txt* e definir as variáveis de ambiente no arquivo *.env*, conforme o exemplo:

```
DATABASE_URL=postgresql://usuario:senha@localhost:5432/logicshuffle  
SECRET_KEY=chave_secreta_aqui  
JWT_SECRET_KEY=chave_jwt_secreta_aqui
```

Após essa etapa, a aplicação pode ser executada com o comando `python app.py`, iniciando o servidor Flask e possibilitando a comunicação com o banco de dados. O front-end, desenvolvido na Godot Engine, pode ser executado diretamente no ambiente da

engine, permitindo depuração e testes de integração. Essa estrutura assegura a modularidade da aplicação e facilita o processo de manutenção e evolução futura do sistema.

5.2 Custos

Para a disponibilização da aplicação em ambiente web e sua operação contínua, foram considerados alguns serviços essenciais relacionados à infraestrutura, domínio e comunicação. A tabela 1 apresenta a estimativa dos custos mensais.

Tabela 1: Custos de Infraestrutura

Item	Descrição	Preço(USD/mês)	Referência
Hospedagem Back-end	VPS básico (1 GB RAM, 1vCPU)	US\$ 5,00	DigitalOcean ²¹
Banco de Dados	PostgreSQL gerenciado (10 GB)	US\$ 7,00	Supabase ²²
Serviço de E-mail	SendGrid(100,000 emails/mês)	US\$ 19,95	SendGrid ²³
Domínio	Registro “.com”	US\$ 1,17	Namecheap ²⁴
Total Estimado	Infraestrutura completa	US\$ 33,12/mês	-

Fonte: Autoria Própria

Também foram realizadas estimativas dos custos relacionados ao desenvolvimento da aplicação, considerando as horas de trabalho por função e as médias de valores de mercado obtidas em plataformas de prestação de serviços internacionais. A Tabela 2 apresenta a estimativa total de custos de desenvolvimento do Logic Shuffle.

²¹ Disponível em: <https://www.digitalocean.com>

²² Disponível em: <https://supabase.com>

²³ Disponível em: <https://sendgrid.com/en-us>

²⁴ Disponível em: <https://www.namecheap.com>

Tabela 2: Custos de Desenvolvimento

Descrição	Total Horas	Valor/Hora (USD)	Total	Referência
Desenvolvimento Back-End	160 horas	US\$ 25,00	US\$ 4 000,00	Upwork ²⁵ – Python Rates
Desenvolvimento Godot	220 horas	US\$ 30,00	US\$ 6 600,00	Upwork – Godot Rates
Design de Interface	100 horas	US\$ 25,00	US\$ 2 500,00	Upwork – UI/UX Rates
Testes	80 horas	US\$ 18,00	US\$ 1 440,00	Upwork – QA Rates
Total	560 horas	-	US\$ 14 540,00	-

Fonte: Autoria Própria

Essas estimativas reforçam a viabilidade econômica do projeto, considerando que os custos mensais de infraestrutura são reduzidos, permitindo a manutenção contínua do sistema com baixo investimento. Além disso, o desenvolvimento baseou-se em ferramentas gratuitas e de código aberto, o que contribuiu significativamente para a redução dos custos gerais do projeto.

5.3 Testes e Validação da Aplicação

Os testes realizados tiveram como objetivo principal verificar a funcionalidade básica e a estabilidade do sistema Logic Shuffle, assegurando que as principais operações descritas nos requisitos funcionais fossem executadas corretamente.

Foram adotados testes de funcionalidade e de usabilidade em pequena escala, devido às limitações de tempo e disponibilidade de participantes. O processo contemplou a verificação manual das funcionalidades de cadastro, login, seleção de jogo e progressão de fases, bem como a persistência dos dados de pontuação e autenticação entre sessões.

Além dos testes internos realizados pelos desenvolvedores, foram aplicadas avaliações informais com um grupo reduzido de usuários — composto por colegas do

²⁵ Disponível em: <https://www.upwork.com>

curso e docentes — a fim de observar a interação com a interface e identificar possíveis melhorias na navegação e na experiência visual dos minijogos.

De modo geral, os resultados indicaram que a aplicação está funcional e estável, atendendo aos objetivos propostos. As principais observações relatadas pelos participantes concentraram-se em aspectos visuais e de responsividade, que poderão ser aperfeiçoados em versões futuras.

6. Conclusão

O desenvolvimento do Logic Shuffle teve como objetivo principal criar uma ferramenta educacional interativa voltada ao ensino de Estruturas de Dados, utilizando a gamificação como meio para tornar o aprendizado mais acessível, dinâmico e motivador. A aplicação integrou conceitos teóricos com elementos lúdicos, visando proporcionar aos estudantes uma experiência prática de visualização e manipulação das estruturas fundamentais da computação, como listas, pilhas, filas e árvores.

O uso do Godot Engine mostrou-se especialmente adequado para o desenvolvimento dos minijogos, por ser uma ferramenta gratuita, de código aberto (*open source*) e de fácil aprendizado, características que viabilizaram a implementação do projeto sem custos adicionais e com alta flexibilidade de adaptação. Essa escolha também reforça o compromisso do trabalho com o uso de tecnologias acessíveis e sustentáveis para fins educacionais.

Os resultados obtidos demonstraram que o Logic Shuffle cumpre seu propósito de auxiliar alunos no processo de aprendizagem de Estruturas de Dados, oferecendo uma abordagem visual e intuitiva que complementa os métodos tradicionais de ensino. O projeto mostrou-se viável tanto do ponto de vista técnico quanto pedagógico, evidenciando o potencial da gamificação como estratégia de apoio ao ensino de conteúdos abstratos.

Embora não tenha sido possível realizar uma testagem com um número maior de usuários ou aplicar métricas quantitativas de desempenho, o sistema demonstrou viabilidade técnica e usabilidade satisfatória em ambiente de teste controlado.

Como trabalhos futuros, recomenda-se banco de dados em nuvem, a adição de novos minijogos abordando outras estruturas e conceitos computacionais, a ampliação do processo de validação com um número maior de usuários e a aplicação de métricas quantitativas de usabilidade, como o *System Usability Scale (SUS)*. Além disso, sugere-se a coleta de dados sobre o impacto pedagógico do Logic Shuffle no aprendizado e

engajamento dos estudantes, visando aperfeiçoar continuamente sua eficácia educacional.

Referências

BARBOSA, Huminig Schmiedt; SILVA, Felipe Fernandes da; CAMPANO JUNIOR, Maurilio Martins; AYLON, Linnyer Beatrys Ruiz. Jogo educativo no ensino de estrutura de dados: aliando Educação 5.0, gamificação e storytelling. In: TRILHA DE EDUCAÇÃO – ARTIGOS COMPLETOS - SIMPÓSIO BRASILEIRO DE JOGOS E ENTRETENIMENTO DIGITAL (SBGAMES), 22., 2023, Rio Grande/RS. *Anais* [...]. Porto Alegre: Sociedade Brasileira de Computação, 2023. p. 792-803. DOI: https://doi.org/10.5753/sbgames_estendido.2023.234099.

BATTISTELLA, Paulo Eduardo; WANGENHEIM, Aldo von; WANGENHEIM, Christiane Gresse von. *SORTIA – Um Jogo para Ensino de Algoritmo de Ordenação: Estudo de caso na disciplina de Estrutura de Dados*. In: SIMPÓSIO BRASILEIRO DE INFORMÁTICA NA EDUCAÇÃO (SBIE), 23., 2012, Rio de Janeiro. *Anais* [...]. Porto Alegre: Sociedade Brasileira de Computação, 2012. p. 1–10.

BRASIL. Lei nº 9.394, de 20 de dezembro de 1996. Estabelece as diretrizes e bases da educação nacional. *Diário Oficial da União*, Brasília, 1996. Disponível em: https://www.planalto.gov.br/ccivil_03/leis/l9394.htm. Acesso em 18 set 2025.

BRUNET, João Arthur. Tabelas Hash. Disponível em: <https://joaoarthurbm.github.io/eda/posts/hashtable>. Acesso em: 18 set. 2025.

BUSINESS OF ESPORTS. Number of games made with Unity grew 93 % in 2021. Disponível em: <https://thebusinessofesports.com/2022/03/28/number-of-games-made-with-unity-grew-93-in-2021/>. Acesso em: 22 out. 2025.

CASTELLS, Manuel. *A sociedade em rede*. 8. ed. São Paulo: Paz e Terra, 1999.

COOK, H.; APPS, T.; BECKMAN, K.; BENNETT, S. Digital competence for emergency remote teaching in higher education: understanding the present and anticipating the future. *Educational Technology Research and Development*, v. 71, p. 7–32, 2023. DOI: 10.1007/s11423-023-10194-4. Disponível em: <https://doi.org/10.1007/s11423-023-10194-4>. Acesso em: 4 nov. 2025.

CORMEN, Thomas H.; LEISERSON, Charles E.; RIVEST, Ronald L.; STEIN, Clifford. *Introduction to Algorithms*. 3. ed. Cambridge, MA: MIT Press, 2009.

CORTÉS, Mariela Inés. *Estrutura de Dados*. 3. ed. Fortaleza: EdUECE, 2015.

Disponível em: <https://www.uece.br/cct/wp-content/uploads/sites/28/2021/07/Estrutura-de-Dados-2014.pdf>. Acesso em: 30 out. 2025.

CRYTEK. *CRYENGINE: The complete solution for next generation game development*.

Disponível em: <https://www.cryengine.com>. Acesso em: 22 out. 2025.

DIGITALOCEAN. *Pricing – Cloud computing services and costs*. Disponível em:

<https://www.digitalocean.com/pricing>. Acesso em: 4 nov. 2025.

FLASK COMMUNITY. *Flask-JWT-Extended Documentation*. [S.I.]: Read the Docs,

[s.d.]. Disponível em: <https://flask-jwt-extended.readthedocs.io/>. Acesso em: 03 nov. 2025.

GODOT ENGINE. *Godot Engine – Free and open source game engine*. Disponível em:

<https://godotengine.org>. Acesso em: 22 out. 2025.

GODOT ENGINE. *Godot Engine Official Documentation*. Versão 4.x. [S.I.]: Godot

Foundation, [s.d.]. Disponível em: <https://docs.godotengine.org/>. Acesso em: 03 nov. 2025.

GRAN CURSOS ONLINE. *Estrutura de dados – listas*. [s.d.]. Disponível em:

<https://blog.grancursosonline.com.br/estrutura-de-dados-listas>. Acesso em: 18 set. 2025.

HARPOONER, Brass. *Making Cyberglads 1: Choosing a game engine: Unity, Unreal Engine & Godot*. Cyberglads, [s.l.], 2018-2019. Disponível em:

<https://cyberglads.com/making-cyberglads-1-choosing-a-game-engine.html>. Acesso em: 22 out. 2025.

KNUTH, Donald E. *The Art of Computer Programming. Volume 1: Fundamental Algorithms*. 3. ed. Reading, MA: Addison-Wesley, 1997. ISBN 978-0-201-89683-1.

KOHLER, Luciana P. de Araújo; REIS, Dalton Solano dos; LOPES, Mauricio Capobianco; CARVALHO, Gabriel Jorge Utyama de; LEONETTI, Umberto Oliveira de Araújo Neto; SILA, Leonardo Linhares; ARAUJO, Felipe Augusto de Carvalho de; BIZON, Artur Ricardo. Minigames para o desenvolvimento do pensamento computacional. *Revista Novas Tecnologias na Educação (RENOTE)*, Porto Alegre, v. 21, n. 1, p. 1–10, 2023. LOOTCODE. *LootCode: a fantasy coding game to practice data structures and algorithms*. Disponível em: <https://www.lootcode.dev>. Acesso em: 22 out. 2025.

MANZANO, Guilherme. *Tudo o que você precisa saber sobre Algoritmos e Estruturas de Dados*. Disponível em: <https://www.dio.me/articles/tudo-o-que-voce-precisa-saber-sobre-algoritmos-e-estrutura-de-dados>. Acesso em: 18 set. 2025.

NAMECHEAP. *Domain registration and pricing*. Disponível em: <https://www.namecheap.com>. Acesso em: 4 nov. 2025.

PALLETT, A.; COLABORADORES DO SQLALCHEMY. *SQLAlchemy ORM Documentation*. [S.I.]: SQLAlchemy, [s.d.]. Disponível em: <https://docs.sqlalchemy.org/>. Acesso em: 03 nov. 2025.

PARREIRA JÚNIOR, Paulo Afonso. Estruturas de dados genéricas. *Revista PROGRAMAR*, 24 dez. 2014. Disponível em: <https://www.revista-programar.info/artigos/estruturas-de-dados-genericas/>. Acesso em: 18 set. 2025.

PASSLIB DEVELOPERS. *Passlib Documentation*. Versão 1.7.4. [S.I.]: Read the Docs, [s.d.].

PRENSKY, Marc. *Aprendizagem baseada em jogos digitais*. São Paulo: SENAC, 2012.

PYTHON COMMUNITY. *python-dotenv Documentation*. [S.I.]: PyPI, [s.d.].

PYTHON SOFTWARE FOUNDATION. *Documentação oficial do Python 3*. Disponível em: <https://docs.python.org/3/>. Acesso em: 03 nov. 2025.

SENDGRID. *Email delivery service – features and pricing.* Disponível em: <https://sendgrid.com/en-us>. Acesso em: 4 nov. 2025.

SOUZA, André P. de; HONDA, Fabrizio; JUNIOR, Osvaldo; PESSOA, Marcela; PIRES, Fernanda. SpaceCode: um jogo educacional para auxiliar na aprendizagem de Algoritmos e Estrutura de Dados. In: *SIMPÓSIO BRASILEIRO DE INFORMÁTICA NA EDUCAÇÃO (SBIE)*, 35., 2024, Rio de Janeiro/RJ. *Anais [...]*. Porto Alegre: Sociedade Brasileira de Computação, 2024. p. 2663-2672. DOI: <https://doi.org/10.5753/sbie.2024.244116>.

SUPABASE. *Pricing – Database and authentication services.* Disponível em: <https://supabase.com/pricing>. Acesso em: 4 nov. 2025.

UNITY. *Unity Real-Time Development Platform.* Disponível em: <https://unity.com/pt>. Acesso em: 22 out. 2025.

UNREAL ENGINE. *Unreal Engine: The most powerful real-time 3D creation tool.* Disponível em: <https://www.unrealengine.com/pt-BR>. Acesso em: 22 out. 2025.

UPWORK. *Find freelancers, independent talent, and agencies for any project.* Disponível em: <https://www.upwork.com>. Acesso em: 4 nov. 2025.

ZINGARO, Daniel; TAYLOR, Cynthia; PORTER, Leo; CLANCY, Michael; LEE, Cynthia; LIAO, Soohyun Nam; WEBB, Kevin C. Identifying student difficulties with basic data structures. In: *International Computing Education Research Conference (ICER 2018)*, Espoo, Finland, 13–15 ago. 2018. *Proceedings...* New York: Association for Computing Machinery, 2018. p. 169–177. DOI: 10.1145/3230977.3231005.

Glossário

API (*Application Programming Interface*) – Conjunto de rotinas e padrões que permitem a comunicação entre sistemas.

ÁRVORE (*Tree*) – Estrutura hierárquica composta por nós interligados, com um nó raiz e nós filhos.

.ENV (*Environment File*) – Arquivo usado para guardar configurações sensíveis, como senhas e chaves de acesso, separadas do código do programa.

BACK-END – Parte do sistema responsável pela lógica de negócio, processamento e acesso a dados.

Criptografia – Técnica usada para proteger informações, tornando-as legíveis apenas para quem possui autorização.

Deque – Estrutura que permite inserção e remoção de elementos nas duas extremidades

Engine de Jogos – Plataforma que fornece ferramentas para desenvolvimento, simulação e renderização de jogos digitais.

Feedback Imediato – Resposta visual, sonora ou textual oferecida ao usuário logo após uma ação, indicando acertos ou erros.

Fila (*Queue*) – Estrutura FIFO (*First In, First Out*), onde o primeiro elemento inserido é o primeiro a ser removido.

Front-end – Camada do sistema com a qual o usuário interage diretamente.

Gamificação (*Gamification*) – Uso de elementos e mecânicas de jogos (pontuação, ranking, níveis) em contextos não lúdicos para aumentar engajamento e motivação.

GDScript – Linguagem de script usada na Godot Engine, similar ao Python.

Hash Table (Tabela de Espalhamento) – Estrutura que associa chaves a valores por meio de uma função de hash para acesso rápido.

Hashing – Técnica de criptografia que transforma dados em códigos fixos e irreversíveis para garantir segurança.

Interface do Usuário (UI) – Conjunto de elementos visuais que permitem a interação entre usuário e sistema.

Lista Encadeada – Estrutura linear em que cada elemento (nó) aponta para o próximo, formando uma cadeia dinâmica.

Minigame – Pequeno jogo ou módulo que faz parte de uma aplicação maior, usado para treinar ou avaliar habilidades específicas.

Nível (Level) – Etapa ou fase de dificuldade progressiva em um jogo.

Open Source (Código Aberto) – Software cujo código-fonte é público e pode ser modificado livremente.

ORM (Object Relational Mapper) – Técnica que permite trabalhar com bancos de dados usando objetos e classes, sem escrever comandos SQL diretamente.

Pilha (Stack) – Estrutura LIFO (Last In, First Out), onde o último elemento inserido é o primeiro a ser removido.

Ranking – Sistema de classificação que organiza usuários com base em pontuações ou desempenho.

Recursividade – Técnica de programação em que uma função chama a si mesma para resolver subproblemas menores.

REST (Representational State Transfer) – Estilo de arquitetura de comunicação entre sistemas distribuídos via HTTP.

SHA-256 – Algoritmo de criptografia usado para gerar *hashes* seguros, amplamente utilizado para proteger senhas e tokens.

Storytelling – Técnica narrativa utilizada para criar envolvimento emocional e contextualizar atividades dentro de um jogo.

Token JWT (JSON Web Token) – Padrão para autenticação e troca segura de informações entre sistemas.

UML (Unified Modeling Language) – Linguagem padrão para modelagem visual de sistemas de software.

Apêndice A – Cópia do Questionário

Neste item é apresentado uma cópia do questionário utilizado na pesquisa com os possíveis usuários do sistema Logic Shuffle. O objetivo desse instrumento foi coletar percepções, opiniões e sugestões relacionadas à usabilidade, interface e funcionalidades da aplicação, de modo a subsidiar a análise de aceitação e a validação dos requisitos do sistema.

1) Você é formado, cursa algum curso ou trabalha na área de Tecnologia de Informação?

- () Sim
() Não

2) Caso seja formado ou esteja cursando, qual é o curso:

Resposta aberta.

3) Já cursou a disciplina de Estrutura de Dados?

- () Sim
() Não

4) Qual parte da disciplina de Estrutura de Dados você achou mais difícil?

- () Pilhas, Filas, Deque
() Alocação dinâmica
() Recursividade
() Listas encadeadas

() Tabelas de espalhamento

() Árvores

() Outro: Resposta Aberta.

5) Em uma escala de 1 a 5, qual foi o seu nível de dificuldade geral na disciplina?

(1) Muito fácil

(2) Fácil

(3) Mediano

(4) Difícil

(5) Muito difícil

6) Você considera que os métodos tradicionais (aula + exercícios) foram suficientes para o aprendizado?

() Sim

() Parcialmente

() Não

7) Qual é a maior dificuldade ao aprender Estruturas de Dados?

() Visualizar o funcionamento

() Entender a lógica

() Implementar em código

() Memorizar a teoria

() Outro: Resposta aberta.

8) Você já utilizou algum jogo ou aplicativo para estudar conteúdos acadêmicos?

- () Sim
- () Não

9) Em sua opinião uma aplicação no formato de minijogos poderia facilitar o aprendizado de Estruturas de Dados?

- (1) Concordo Totalmente
- (2) Concordo Parcialmente
- (3) Indiferente
- (4) Concordo Parcialmente
- (5) Concordo Totalmente

10) Que tipo de recurso interativo ajudaria mais no seu aprendizado?

- [] Visualizações animadas
- [] Exercícios práticos gamificados
- [] Desafios em grupo
- [] Sistema de pontuação/ranking
- [] Feedback imediato de erros
- [] Outro: Resposta Aberta

11) Qual dispositivo você mais usaria para jogar/aprender?

- () Computador/Notebook
- () Tablet
- () Celular

12) Você preferiria minijogos mais curtos e diretos ou desafios mais longos e completos?

- () Minijogos curtos e diretos
- () Desafios longos e completos
- () Indiferente

13) Qual estrutura de dados você acha que mais precisa de visualização prática para aprender?

- [] Pilhas, Filas, Deque
- [] Alocação dinâmica
- [] Recursividade
- [] Listas encadeadas
- [] Tabelas de espalhamento
- [] Árvores
- [] Outro: Resposta Aberta.

Feedback aberto

Deixe aqui algum comentário ou sugestão para a aplicação.

Resposta aberta.

Apêndice B – Níveis subsequentes Dungeon Cave Quest

Figura B.1 - Tela de Introdução (Nível 2)



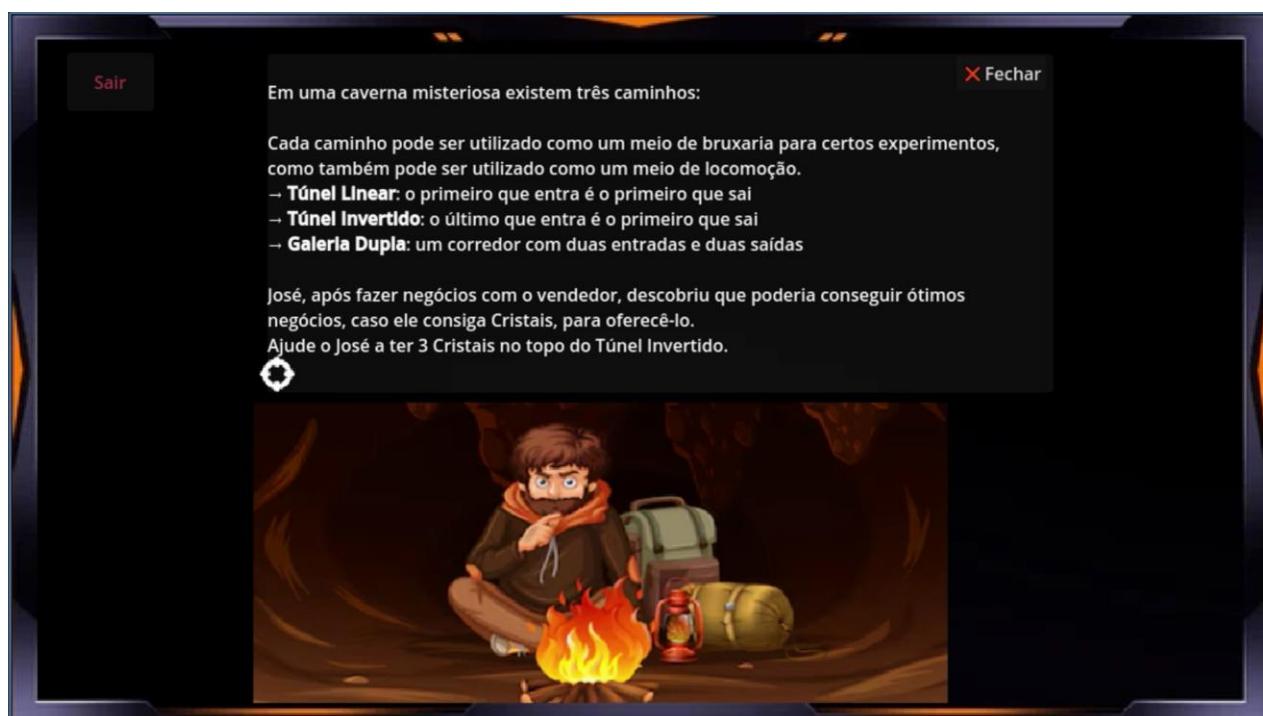
Fonte: Autoria Própria

Figura B.2 - Interface de jogo (Nível 2)



Fonte: Autoria Própria

Figura B.3 - Tela de Introdução (Nível 3)



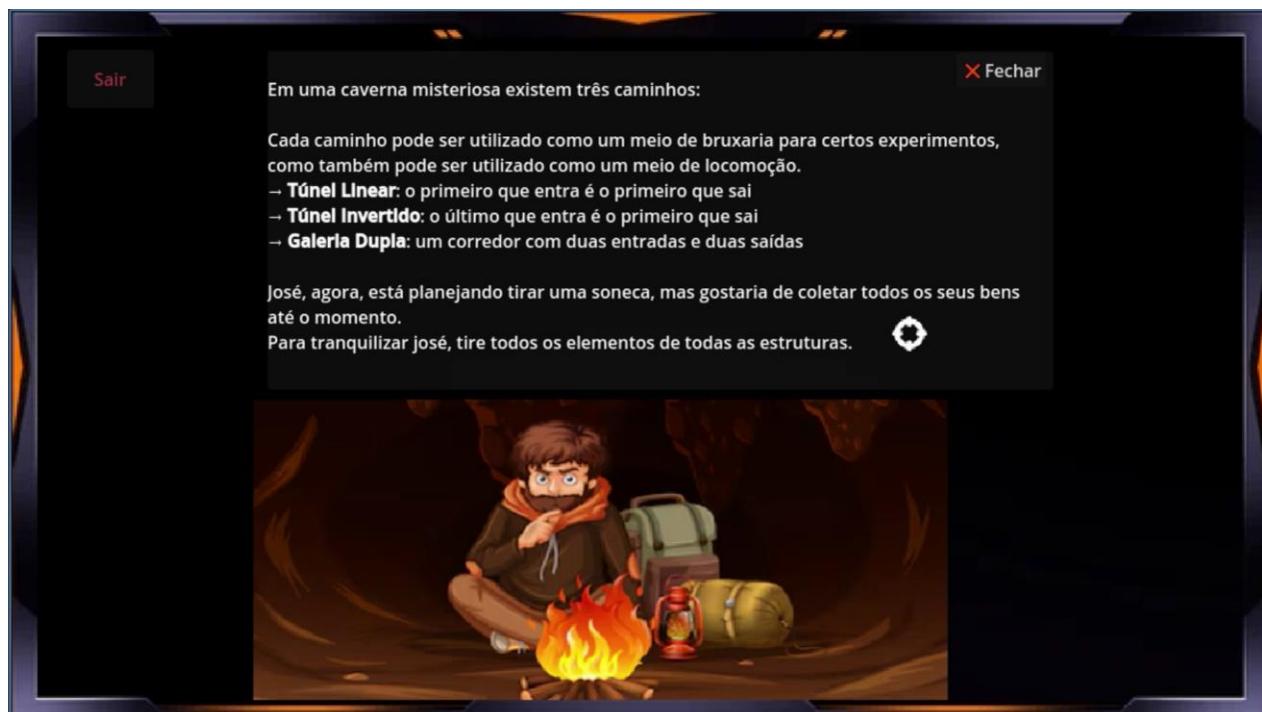
Fonte: Autoria Própria

Figura B.4 - Interface de jogo (Nível 3)



Fonte: Autoria Própria

Figura B.5 - Tela de Introdução (Nível 4)



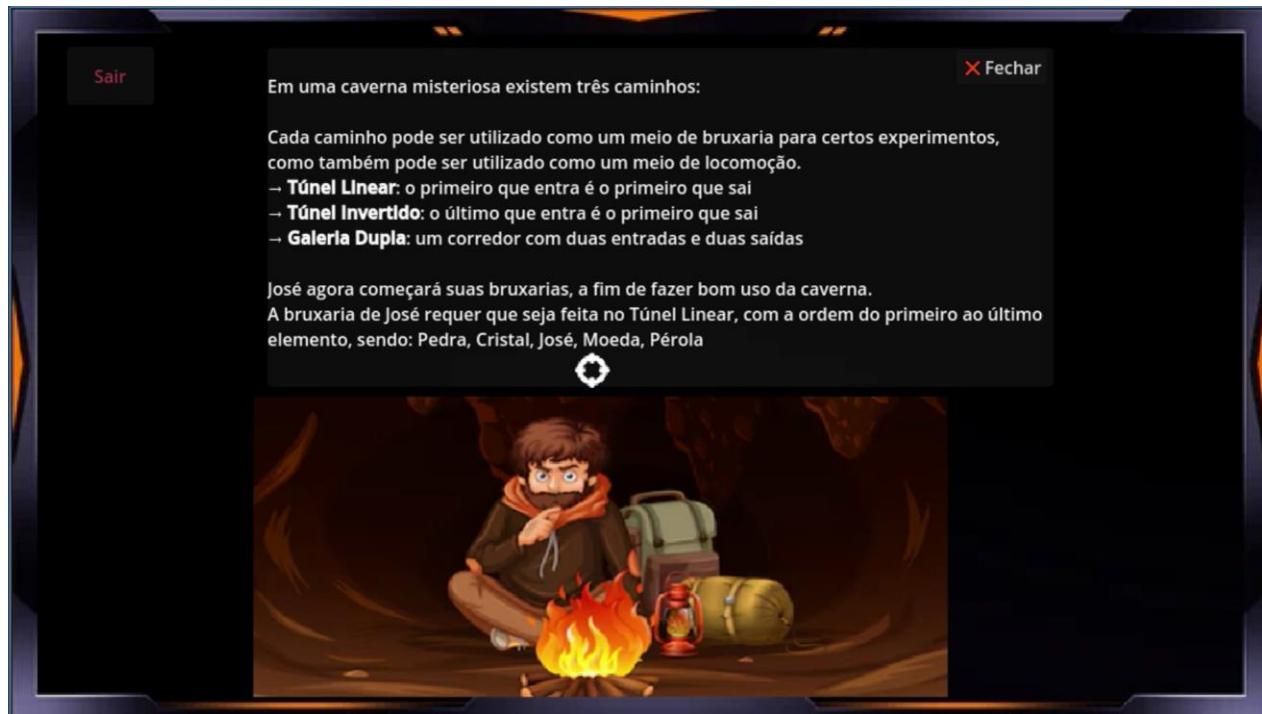
Fonte: Autoria Própria

Figura B.6 - Interface de jogo (Nível 4)



Fonte: Autoria Própria

Figura B.7 - Tela de Introdução (Nível 5)



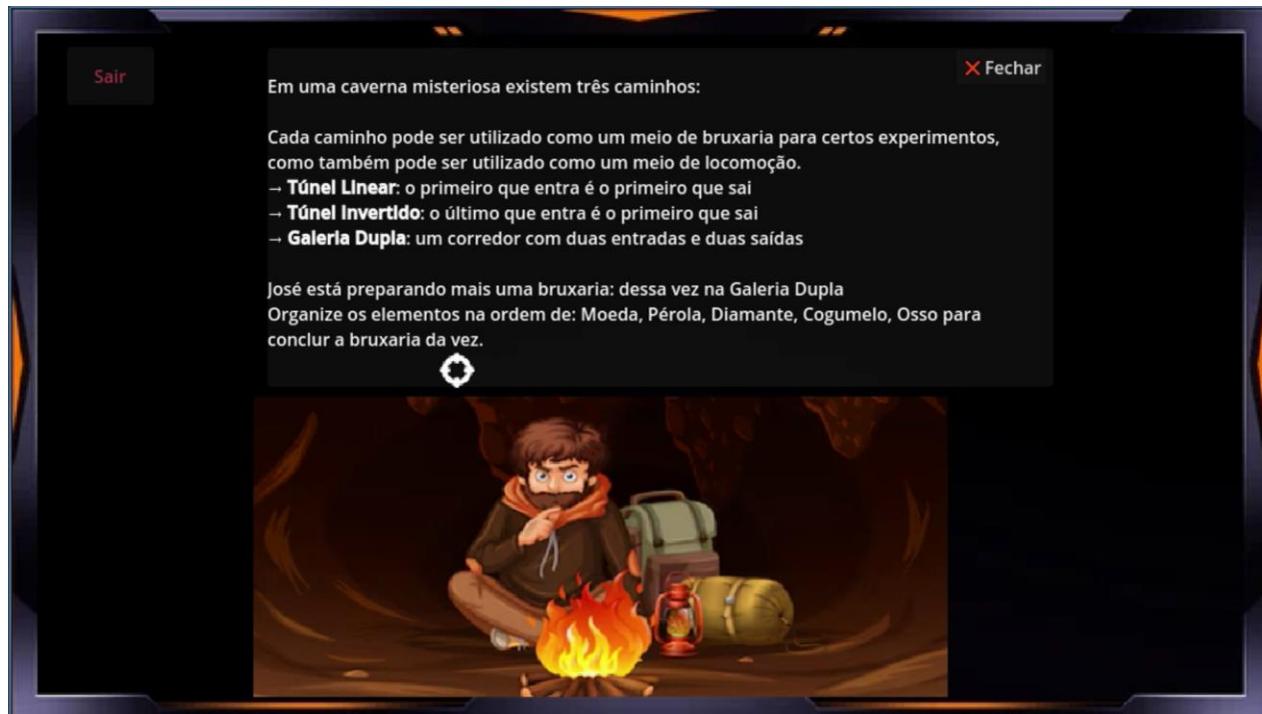
Fonte: Autoria Própria

Figura B.8 - Interface de jogo (Nível 5)



Fonte: Autoria Própria

Figura B.9 - Tela de Introdução (Nível 6)



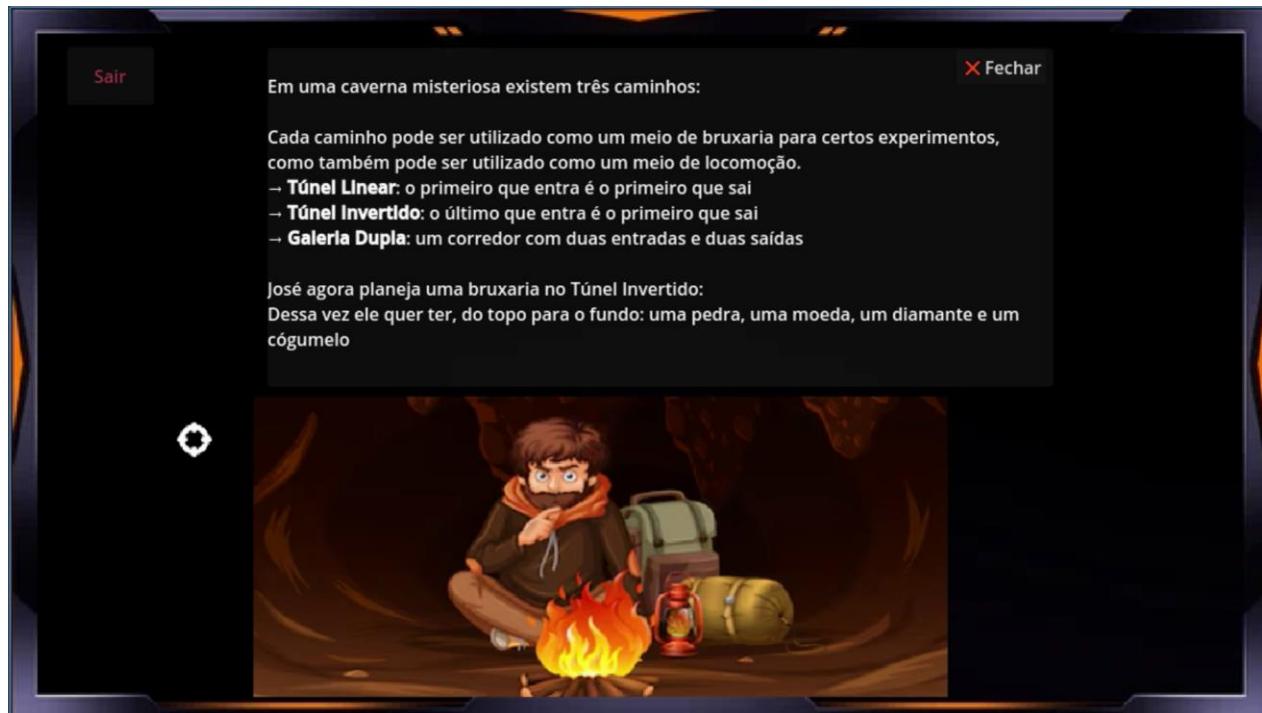
Fonte: Autoria Própria

Figura B.10 - Interface de jogo (Nível 6)



Fonte: Autoria Própria

Figura B.11 - Tela de Introdução (Nível 7)



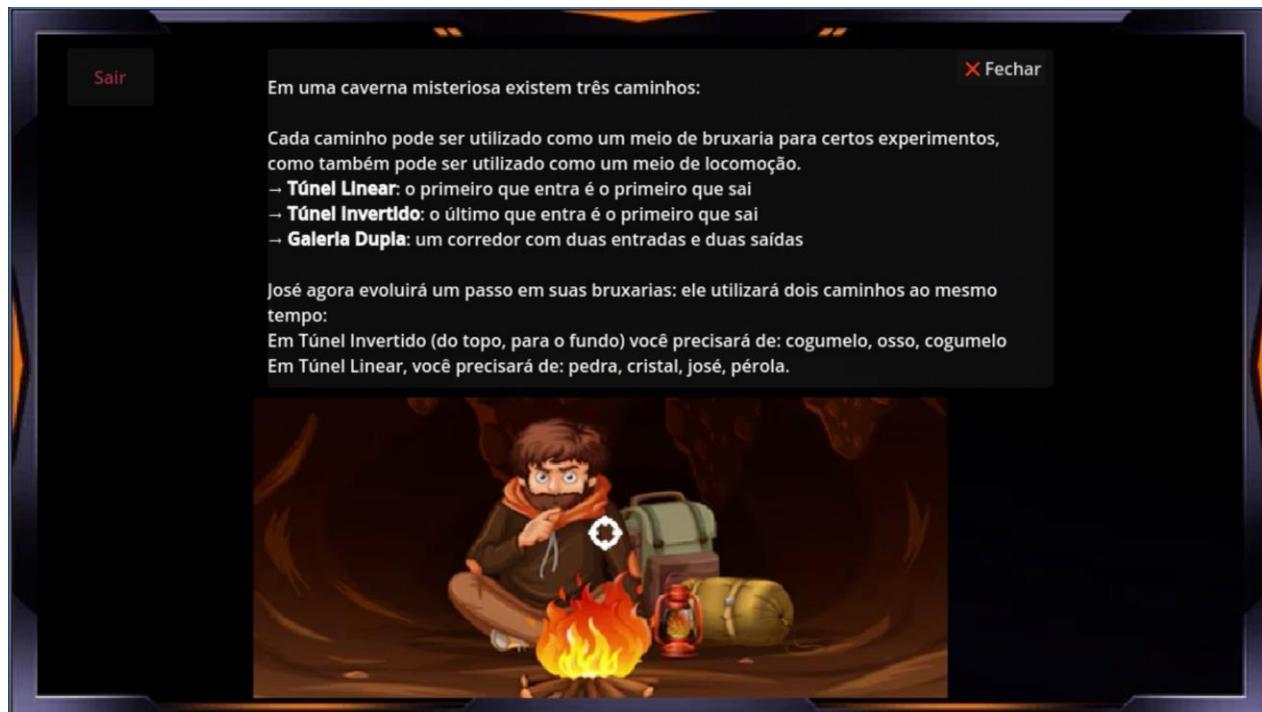
Fonte: Autoria Própria

Figura B.12 - Interface de jogo (Nível 7)



Fonte: Autoria Própria

Figura B.13 - Tela de Introdução (Nível 8)



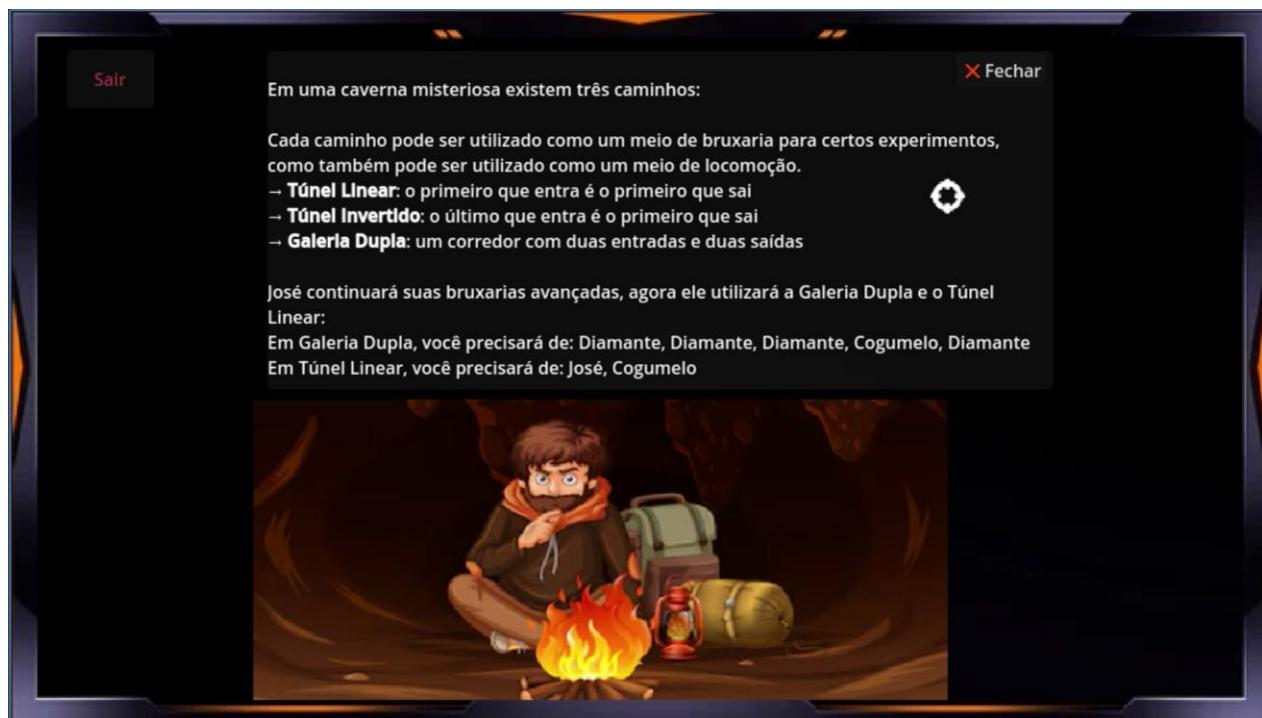
Fonte: Autoria Própria

Figura B.14 - Interface de jogo (Nível 8)



Fonte: Autoria Própria

Figura B.15 - Tela de Introdução (Nível 9)



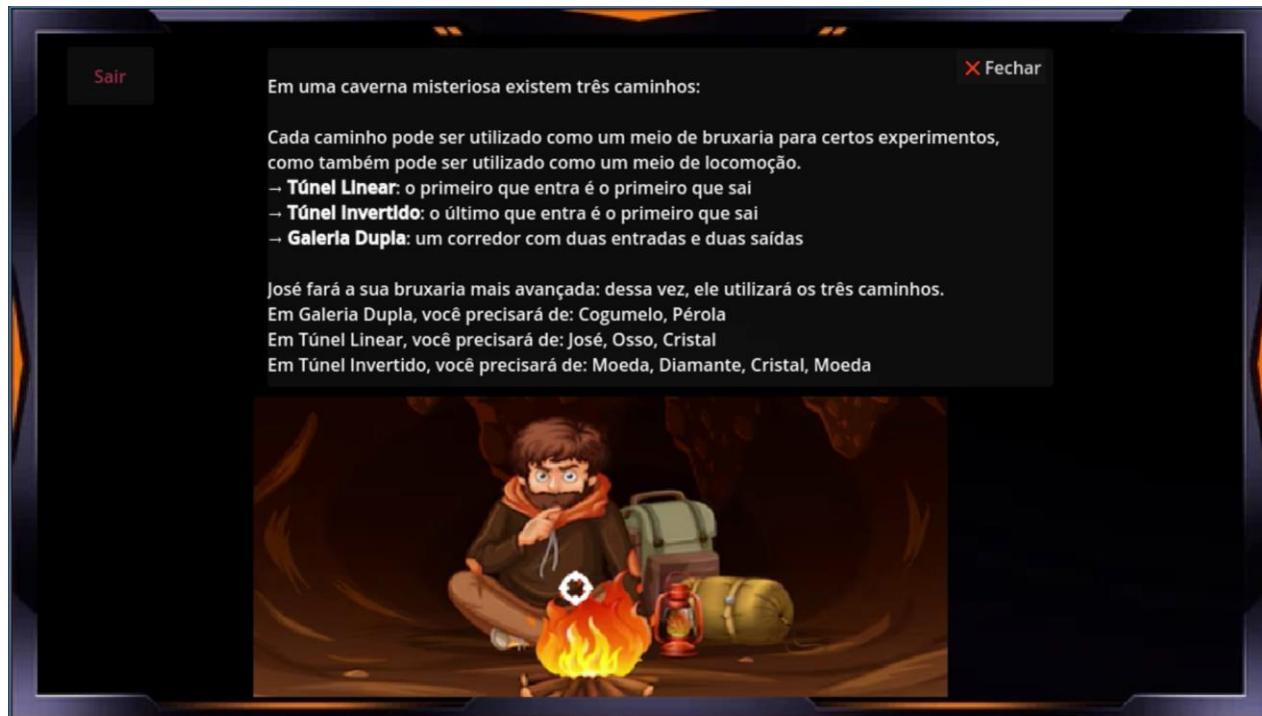
Fonte: Autoria Própria

Figura B.16 - Interface de jogo (Nível 9)



Fonte: Autoria Própria

Figura B.17 - Tela de Introdução (Nível 10)



Fonte: Autoria Própria

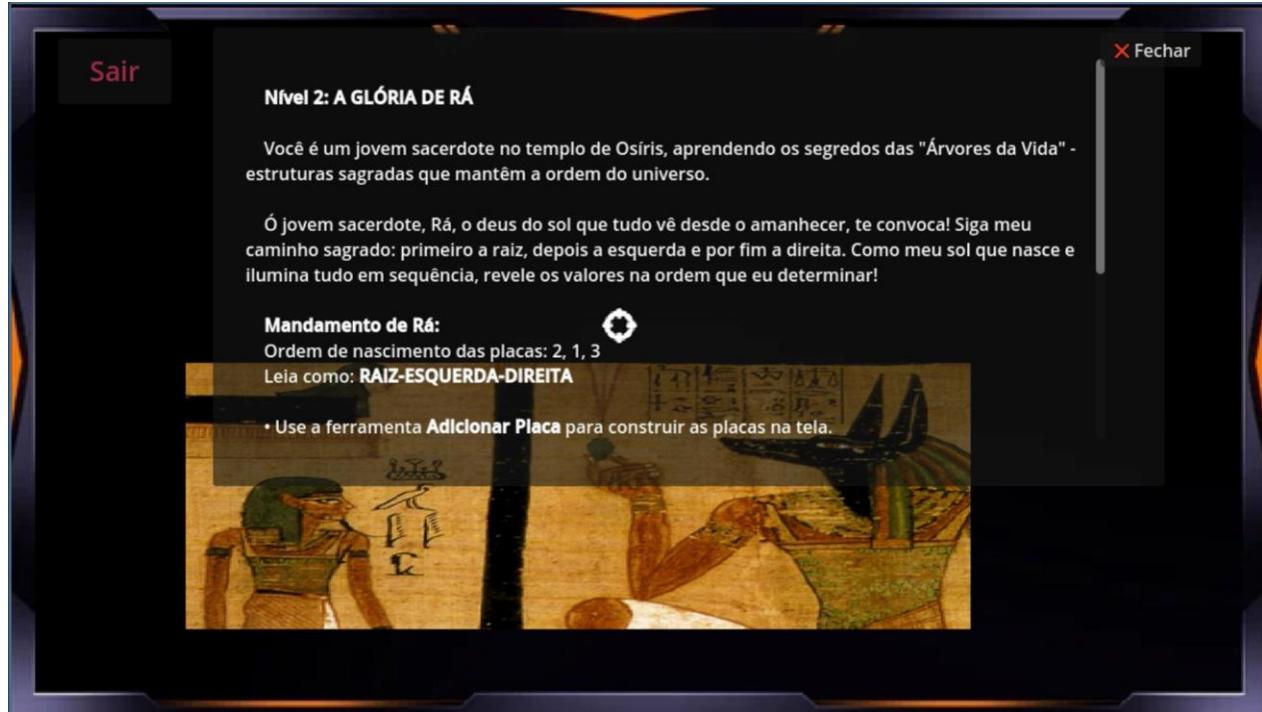
Figura B.18 - Interface de jogo (Nível 10)



Fonte: Autoria Própria

Apêndice C – Níveis subsequentes Pyramid Maker

Figura C.1 - Tela de Introdução (Nível 2)



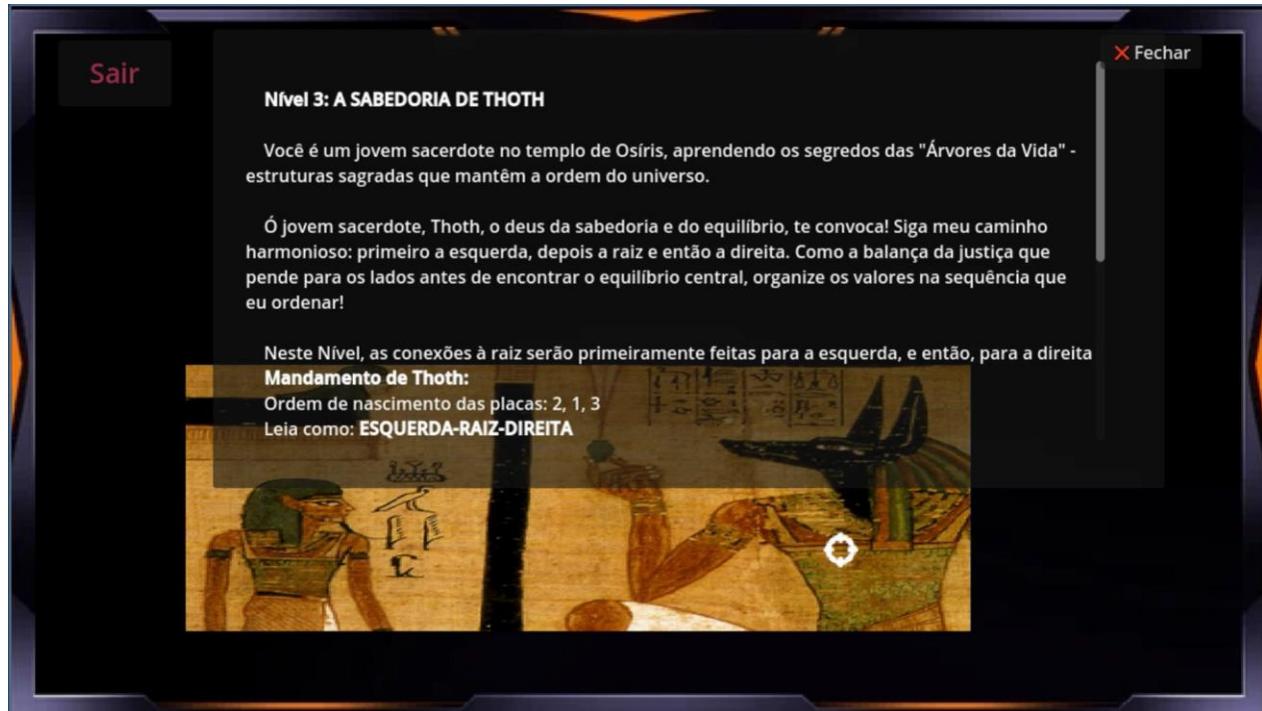
Fonte: Autoria Própria

Figura C.2 - Interface de jogo (Nível 2)



Fonte: Autoria Própria

Figura C.3 - Tela de Introdução (Nível 3)



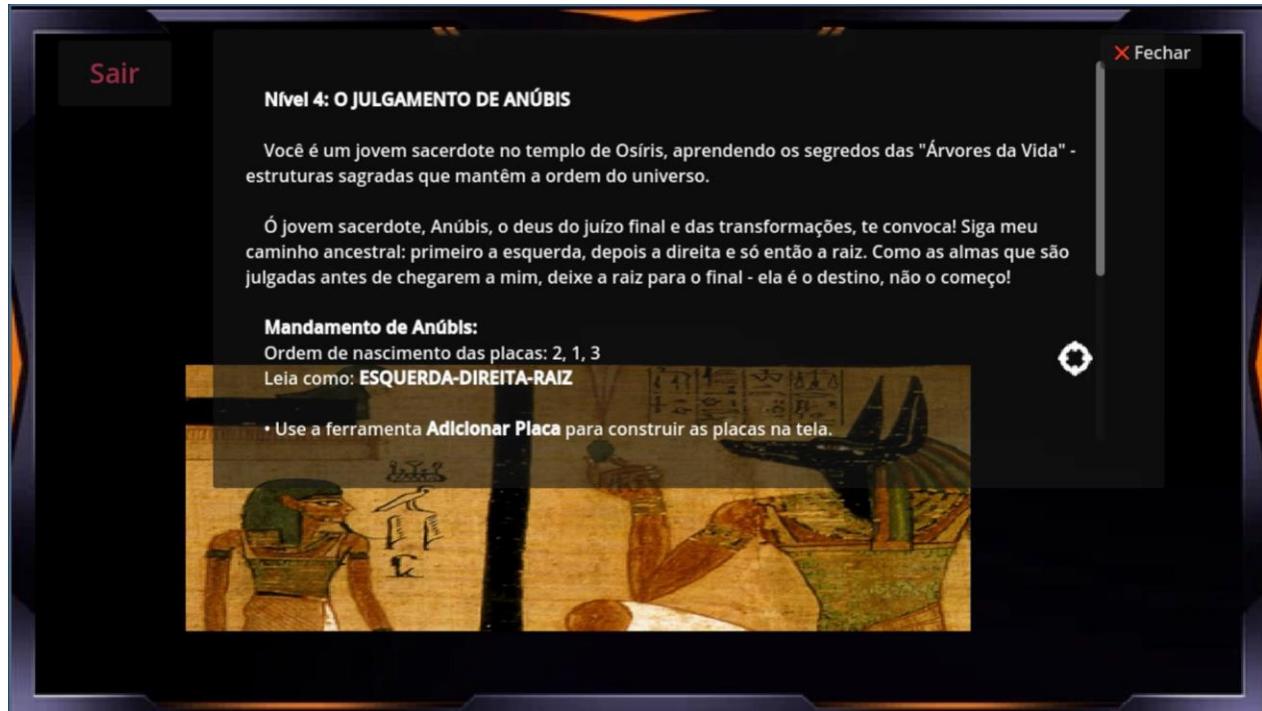
Fonte: Autoria Própria

Figura C.4 - Interface de jogo (Nível 3)



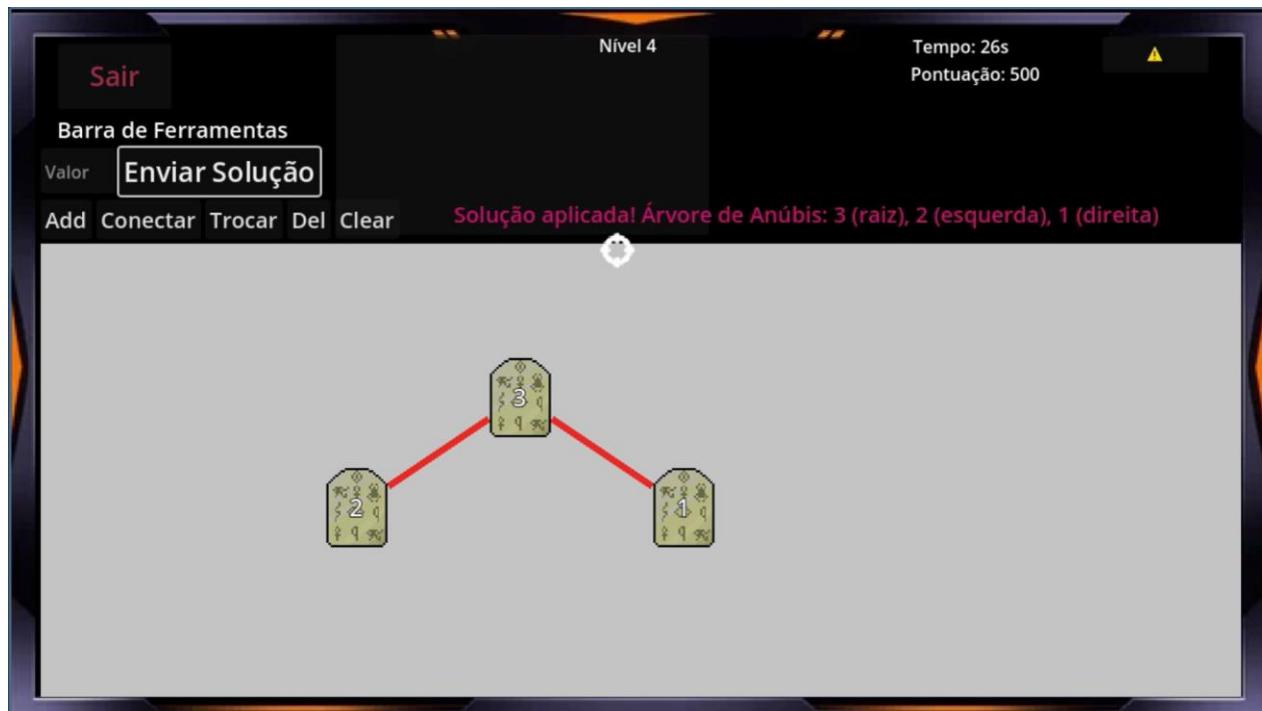
Fonte: Autoria Própria

Figura C.5 - Tela de Introdução (Nível 4)



Fonte: Autoria Própria

Figura C.6 - Interface de jogo (Nível 4 Resolvido)



Fonte: Autoria Própria

Figura C.7 - Tela de Introdução (Nível 5)



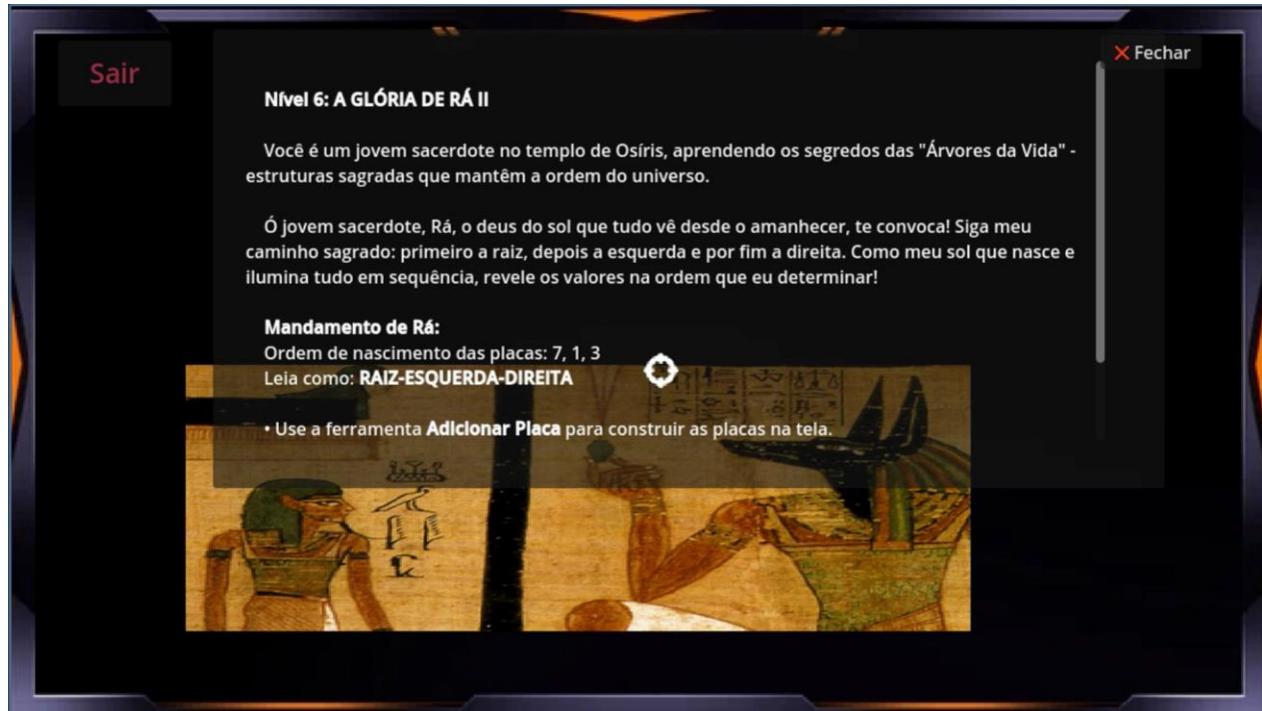
Fonte: Autoria Própria

Figura C.8 - Interface de jogo (Nível 5)



Fonte: Autoria Própria

Figura C.9 - Tela de Introdução (Nível 6)



Fonte: Autoria Própria

Figura C.10 - Interface de jogo (Nível 6)



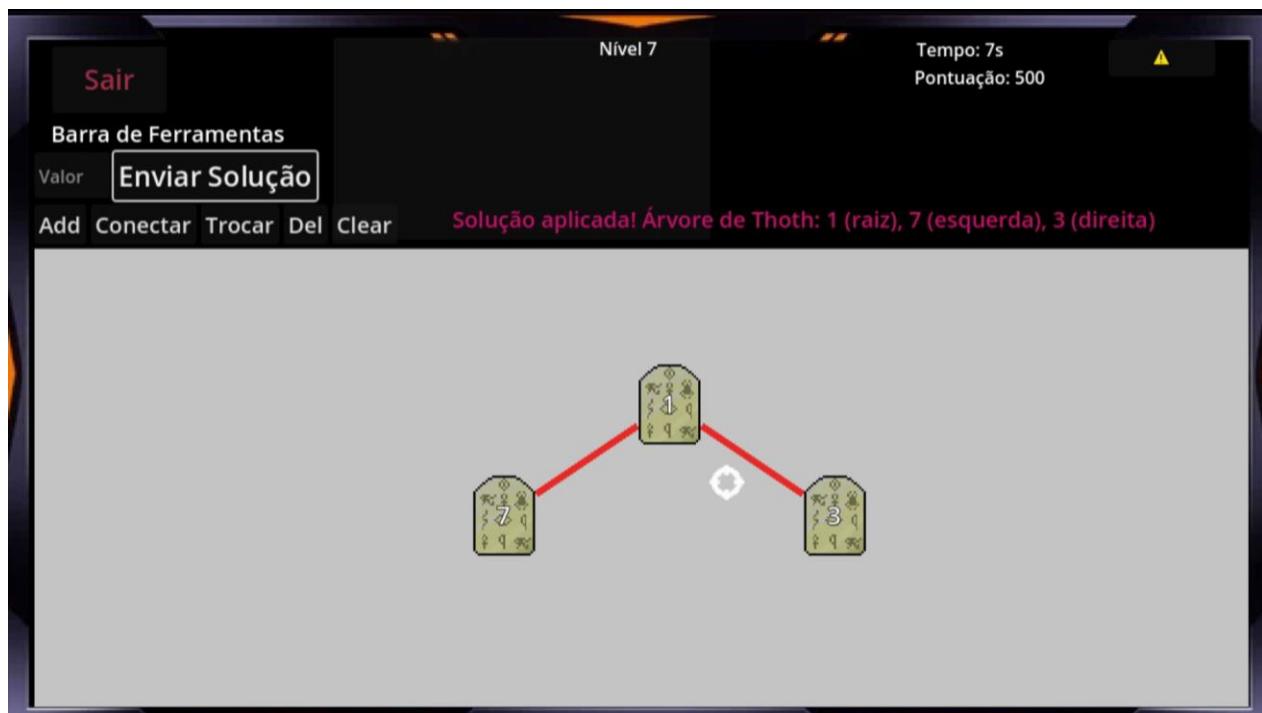
Fonte: Autoria Própria

Figura C.11 - Tela de Introdução (Nível 7)



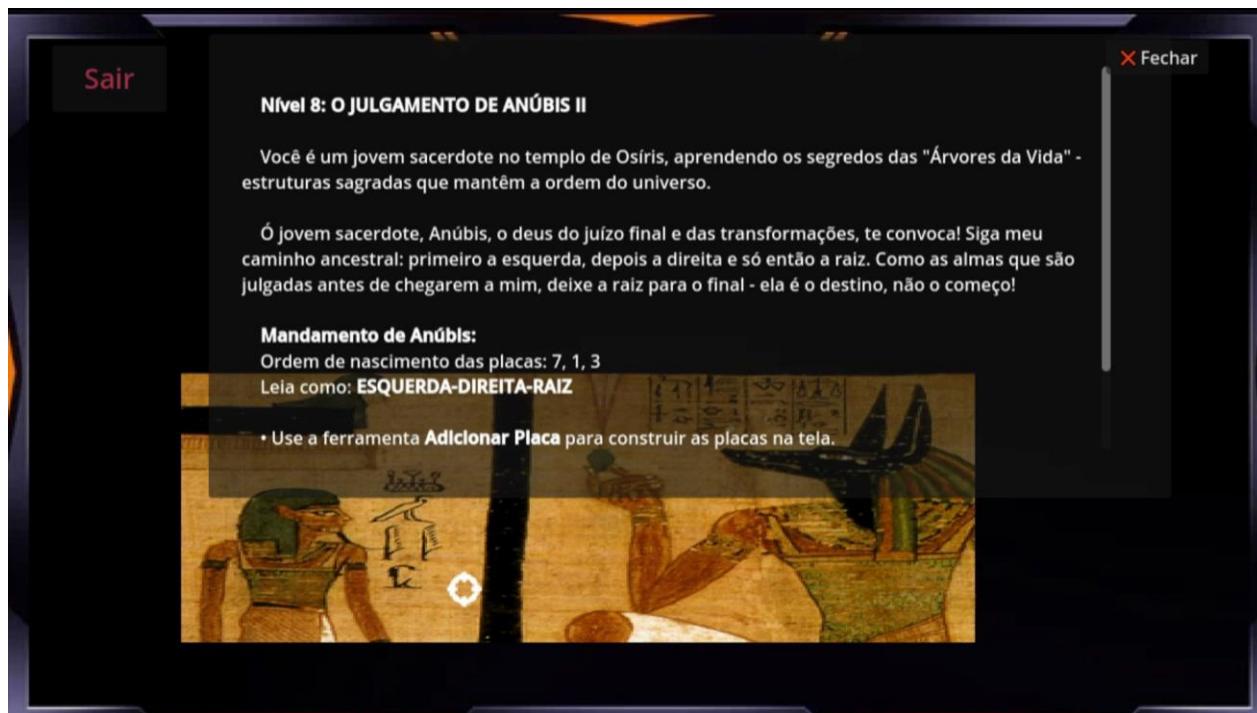
Fonte: Autoria Própria

Figura C.12 - Interface de jogo (Nível 7 Resolvido)



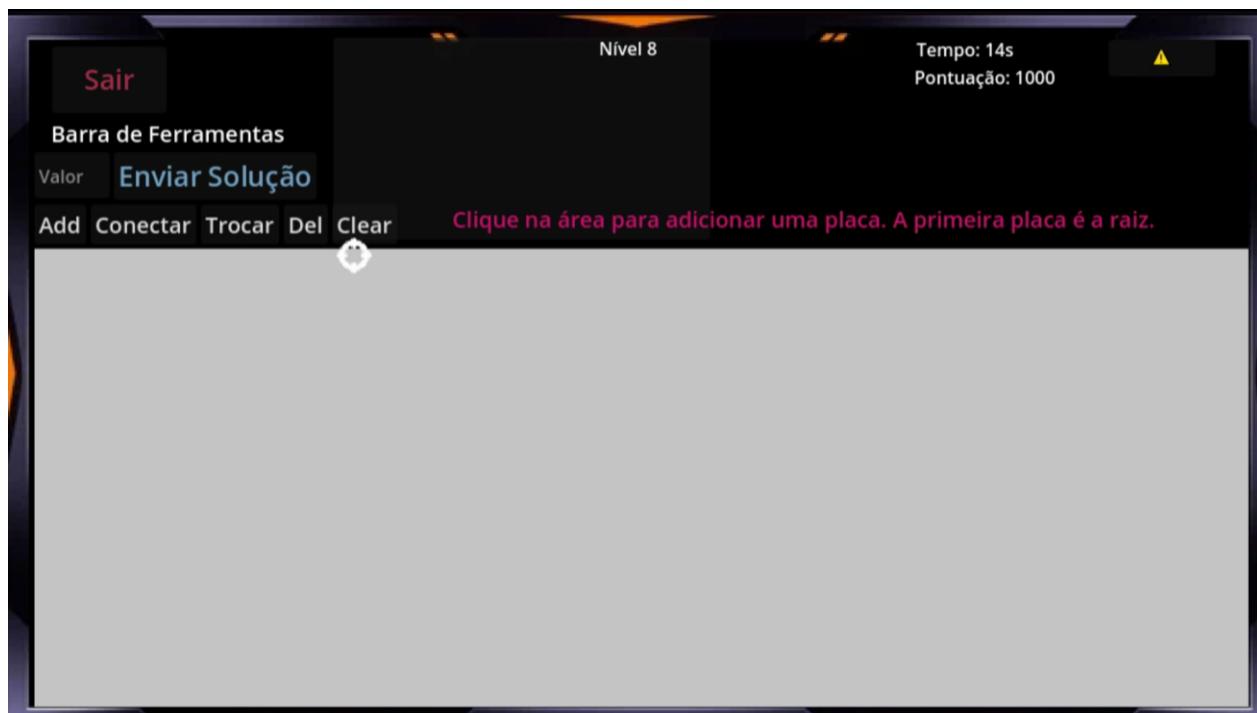
Fonte: Autoria Própria

Figura C.13 - Tela de Introdução (Nível 8)



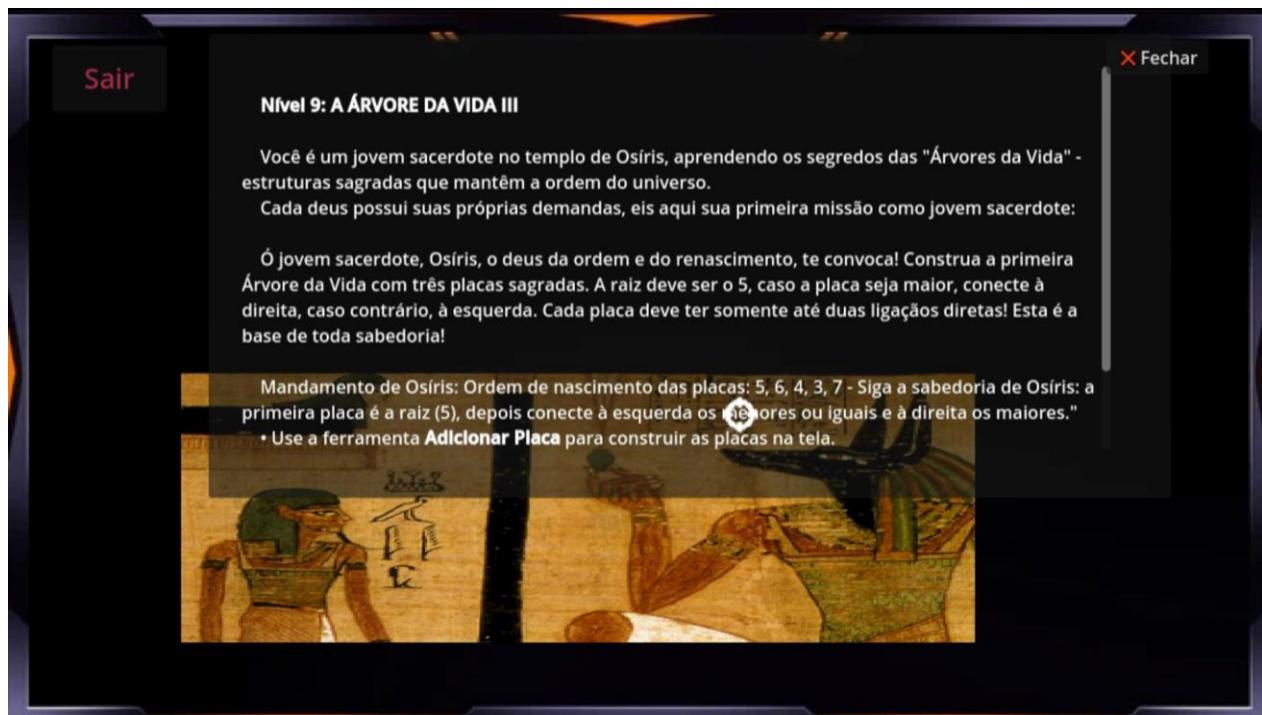
Fonte: Autoria Própria

Figura C.14 - Interface de jogo (Nível 8)



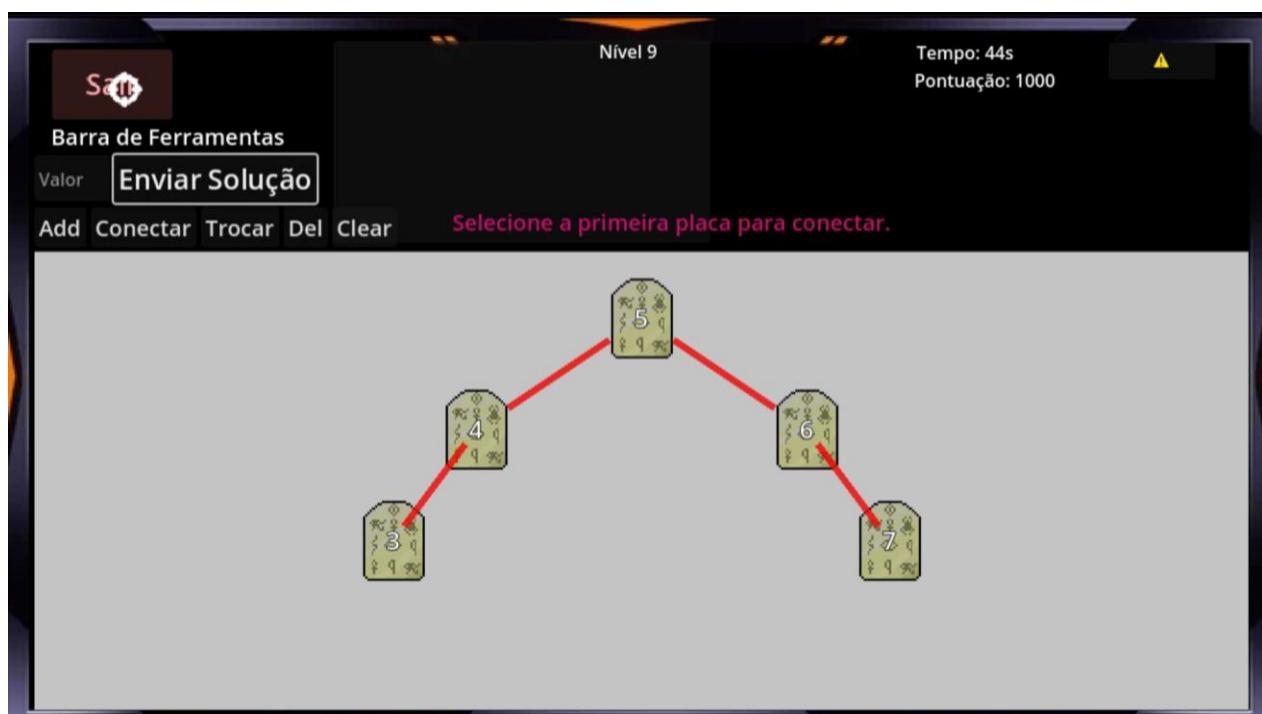
Fonte: Autoria Própria

Figura C.15 - Tela de Introdução (Nível 9)



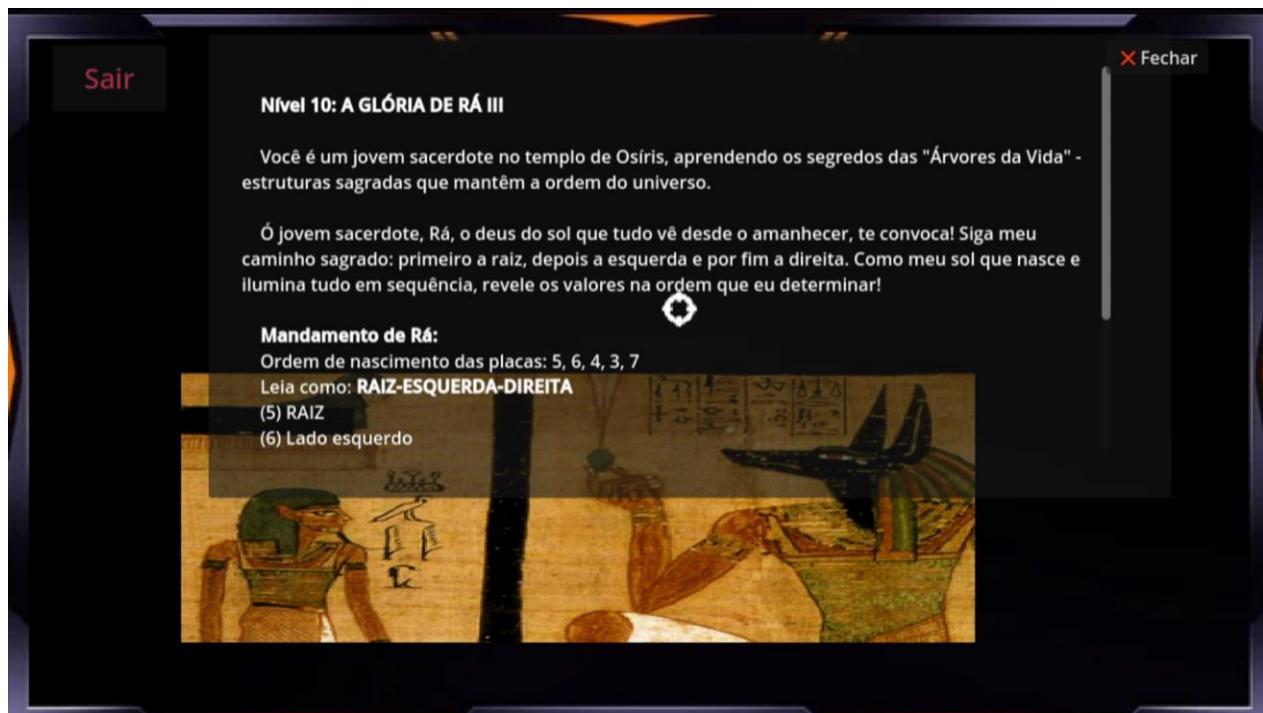
Fonte: Autoria Própria

Figura C.16 - Interface de jogo (Nível 9 Resolvido)



Fonte: Autoria Própria

Figura C.17 - Tela de Introdução (Nível 10)



Fonte: Autoria Própria

Figura C.18 - Interface de jogo (Nível 10)



Fonte: Autoria Própria