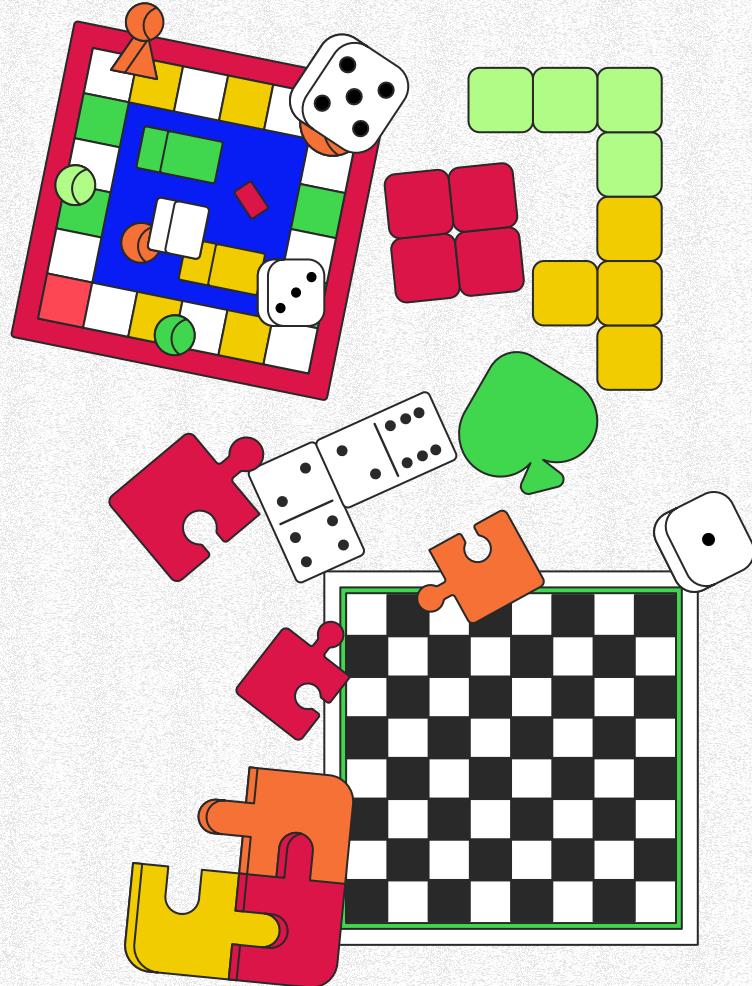
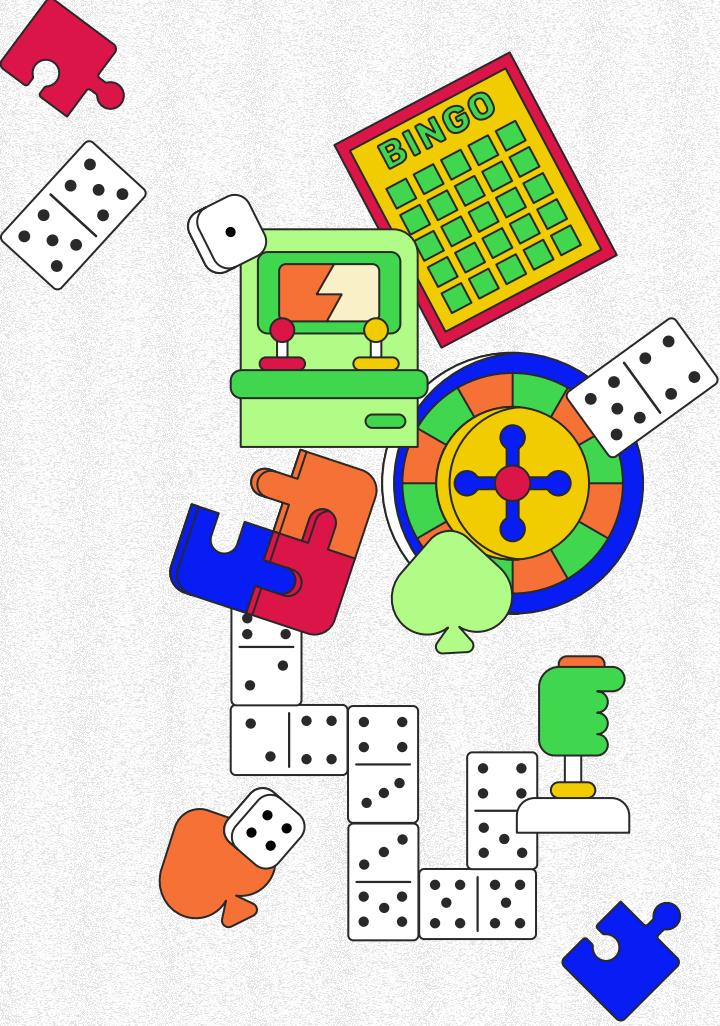


ODSlize

Gamificação dos ODS

Gerência de Configuração e Mudanças





Integrantes



Geisa Moraes



Leonardo Inácio



Tiago Amaro

Índice

01

Descrição

Definição do projeto

04

Requisitos

Lista de requisitos

02

Gerência

Organização e ferramentas

03

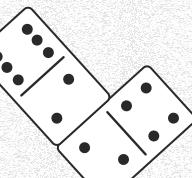
Idealização

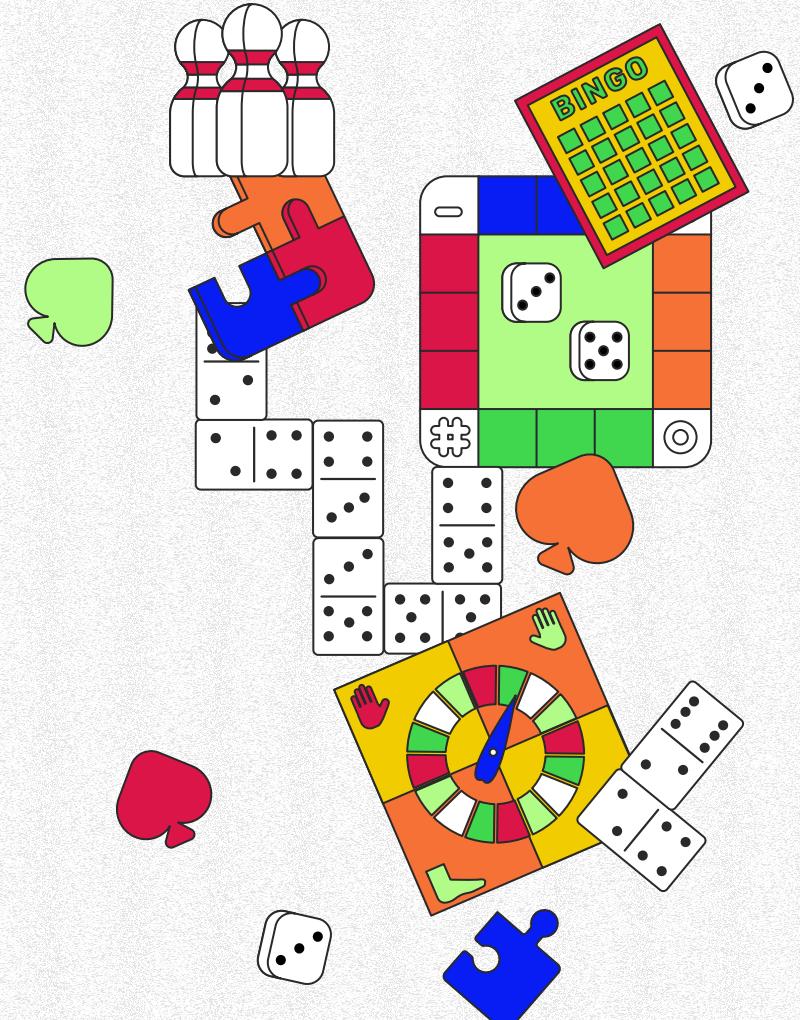
Identidade e arquitetura

06

Plano de teste

Técnicas e ferramentas





Definição

O ODSsize consiste em um jogo do tipo *slide puzzle* com temática educativa voltada para os 17 Objetivos de Desenvolvimento Sustentável (ODS) da ONU.

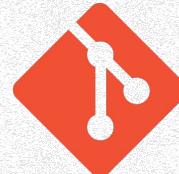
Gerenciamento



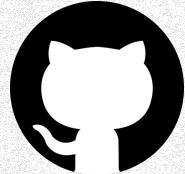
Agenda



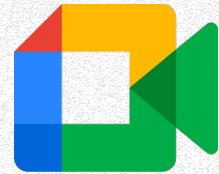
Docs



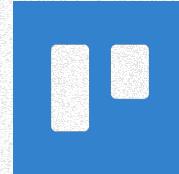
Git



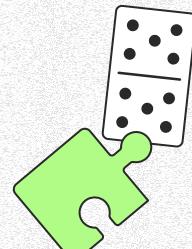
Github



Meet



Trello





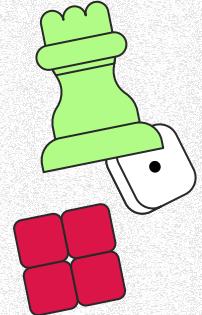
Idealização

Nome

ODS + Deslize = ODSsize

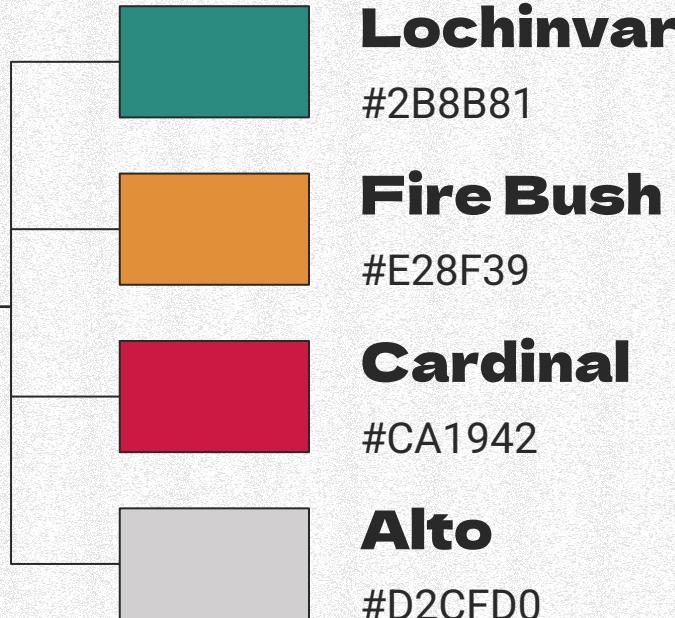
Trocadilho com a palavra *slide* de *slide puzzle*, que lembra o ato de deslizar as peças pelo tabuleiro

Logo



Idealização

Paleta de cores



Arquitetura do sistema



Single Page Application - SPA

- Construída com HTML5, CSS3 e JavaScript



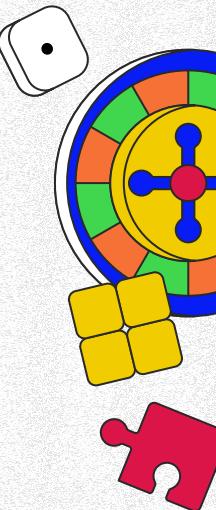
Serviços de Mídia em Nuvem

- Utilizando um Bucket S3 integrada com uma CDN



Controle de Versão (Git com GitHub)

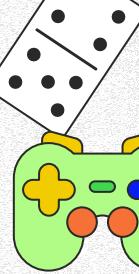
Padrões de Projeto



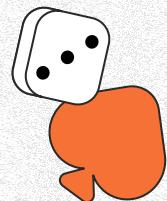
- **State Pattern** - utilizado para gerenciar os diferentes estados do jogo;
- **Strategy Pattern** - define algoritmos específicos para determinados comportamentos variáveis;
- **Factory Method** - são empregados para a criação de objetos complexos;
- **Observer Pattern** - utilizado para notificar componentes da interface do usuário sobre mudanças de estado no jogo;
- **Command Pattern** - aplicado para representar as ações do jogador como objetos.



Requisitos funcionais

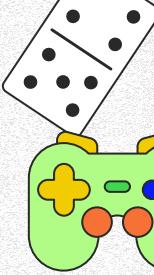
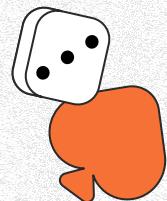


- **[RF001]** Iniciar partida com o tabuleiro embaralhado automaticamente.
- **[RF002]** Movimentar peças para a casa vazia adjacente por meio de um clique ou gesto.
- **[RF003]** Reiniciar partida atual com um novo embaralhamento.
- **[RF004]** Contabilizar tempo e movimentos realizados por rodada.
- **[RF005]** Possuir níveis de dificuldade, permitindo que o jogador avance no jogo.

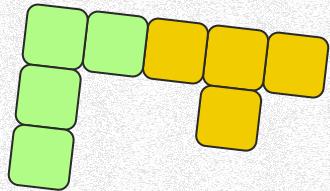


Requisitos funcionais

- **[RF006] Armazenar resultados do jogo** (tempo e quantidade de movimentos do jogador).
- **[RF007] Informar a vitória do jogador** após verificação automática da ordem das peças.
- **[RF008] Explicar o ODS** após o estado de vitória, promovendo conhecimento ao jogador.
- **[RF009] Apresentar resultado esperado** para visualização da organização das peças.



Requisitos não funcionais



- **[RNF001] - Usabilidade (operabilidade)**: a interface do sistema deve ser responsiva, adaptando-se automaticamente a diferentes tamanhos de tela.
- **[RNF002] - Usabilidade (aprendizibilidade)**: o sistema deve informar as instruções de como se joga, facilitando o aprendizado do usuário.
- **[RNF003] - Desempenho (comportamento em relação ao tempo)**: o tempo de resposta para as ações do usuário no jogo deve ser inferior a 1 segundo.

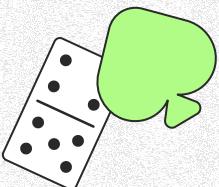


Diagrama de caso de uso

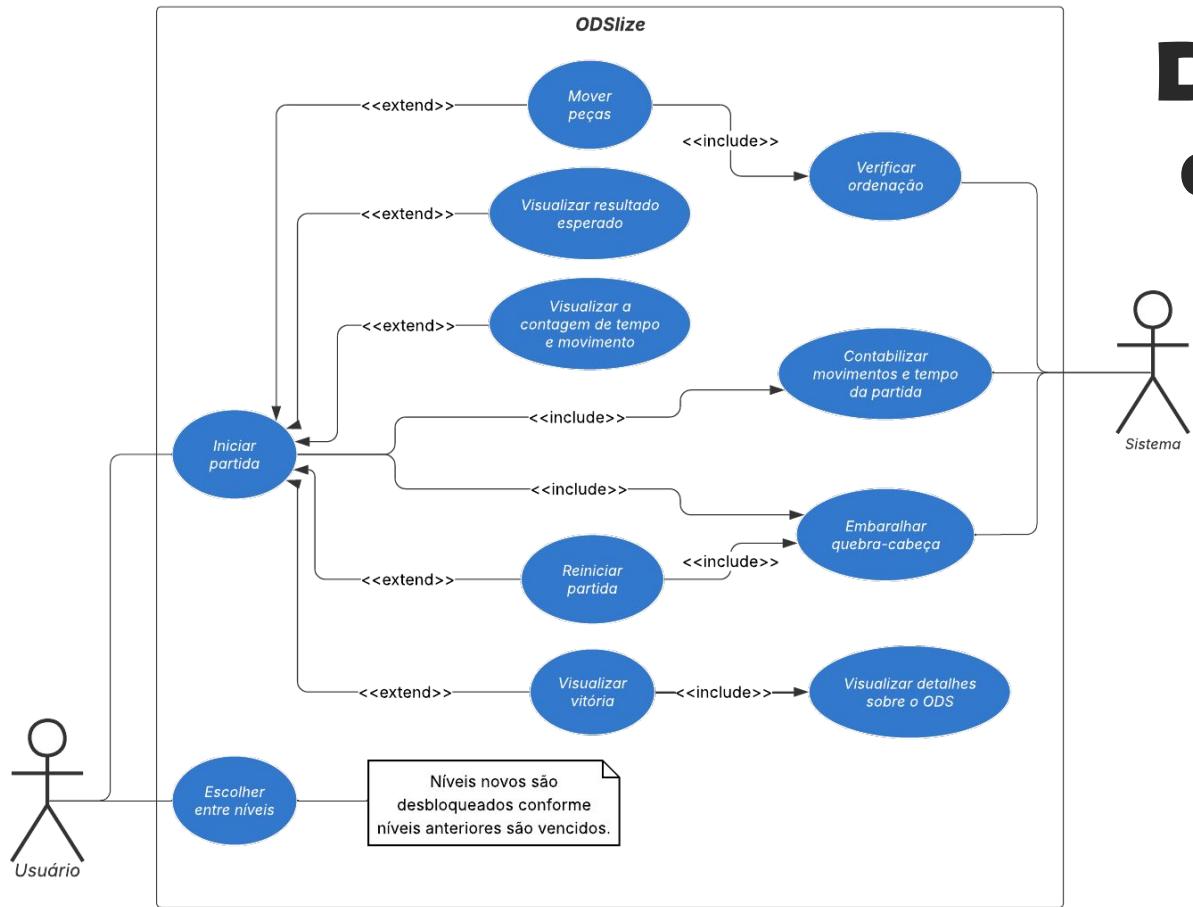
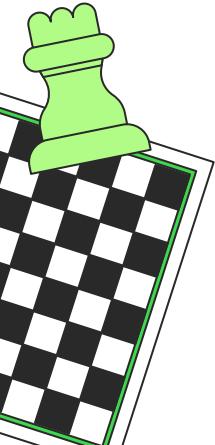
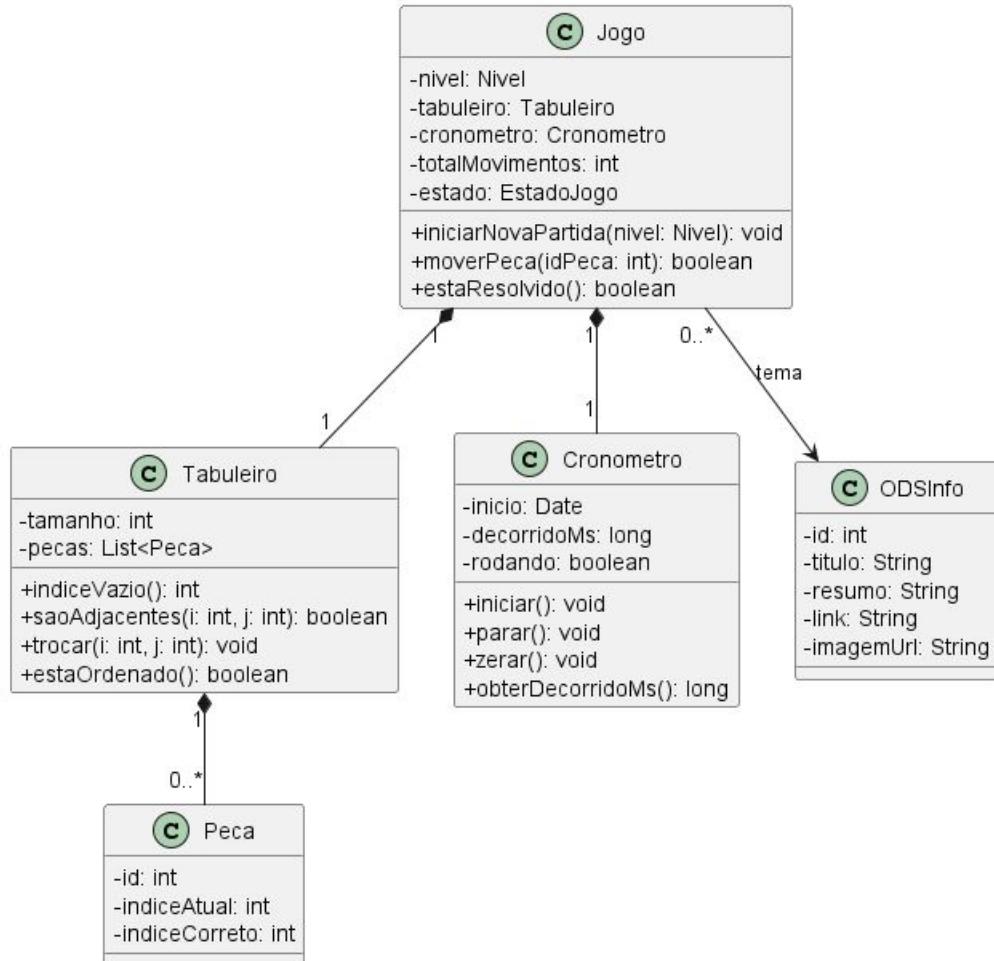
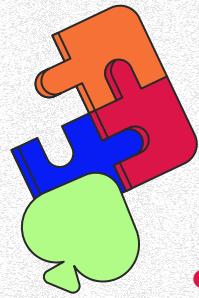


Diagrama de classes



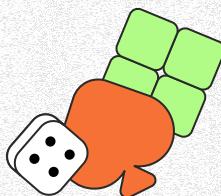
ODSsize - Diagrama de Classes (v1)

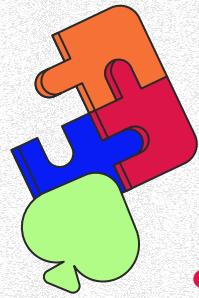




Plano de teste

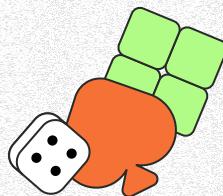
- **Planejamento e realização dos testes**
 - Testes estruturais;
 - Testes de unidade.
 - Testes funcionais;
 - Testes exploratórios;
 - Testes não funcionais.
 - Testes de responsividade, desempenho e qualidade de página.





Plano de teste

- **Abordagens de testes utilizados**
 - **Unidade (estruturais)**: Jest + React Testing Library
 - **Desempenho e Qualidade**: Lighthouse
 - **Código-fonte**: SonarQube
 - **Exploratórios**: manuais baseados na expertise do testador



Obrigado!

geisa.gabriel@alunos.ufersa.edu.br

leonardo.dantas69361@alunos.ufersa.edu.br

tiaqo.nunes@alunos.ufersa.edu.br

CREDITS: This presentation template was created by [Slidesgo](#), and includes icons by [Flaticon](#), and infographics & images by [Freepik](#)

