CENTRO FEDERAL DE EDUCAÇÃO TECNOLÓGICA DE MINAS GERAIS
CAMPUS VII - UNIDADE TIMÓTEO
Curso: Engenharia de Computação
Disciplina: Desenvolvimento de Sistemas
Professor: Odilon Corrêa da Silva

GUIA 02 - CSS: Grid Layout

Objetivo

- Desenvolver layouts utilizando CSS Grid layout

Grid layout

Grid Layout ou CSS Grid Layout é um sistema para criação de layout baseado em uma grade bidimensional e otimizado para design de interfaces. Nesse modelo de grade, os elementos-filhos do container que define a grade podem ser posicionados livremente (Figura 01).

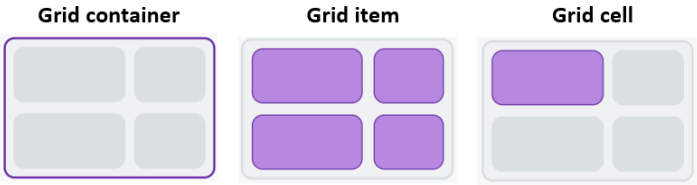


Figura 01: Grid container e grid item

O CSS Grid Layout possui diversas propriedades para trabalhar com o **grid container** (elemento pai) e com os **grid items** (elementos filhos), permitindo alinhar o conteúdo dentro das **grid cell** (células do grid). As propriedades de alinhamento são divididas em dois grupos:

1. **Alinhamento do Conteúdo dentro das Células**

   A propriedade **justify-items** alinha o conteúdo horizontalmente dentro da célula (Figura 02):

   - **stretch** (padrão): estica o conteúdo para preencher a célula.
   - **start**: alinha o conteúdo no início da célula.
   - **end**: alinha o conteúdo no final da célula.
   - **center**: alinha o conteúdo ao centro da célula.

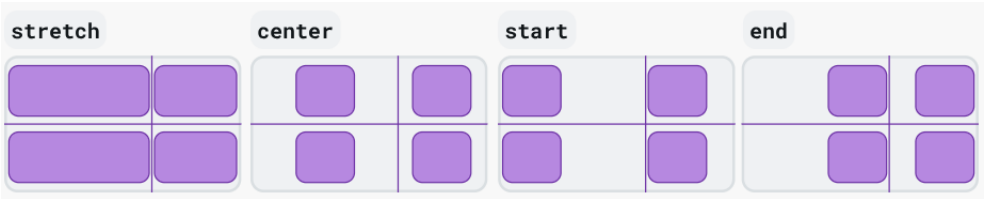


Figura 02: Opções da propriedade justify-items

A propriedade **align-items** alinha o conteúdo verticalmente dentro da célula (Figura 03):

- **stretch** (padrão): estica o conteúdo para preencher a célula.
- **start**: alinha o conteúdo na parte superior da célula.
- **end**: alinha o conteúdo na parte inferior da célula.
- **center**: alinha o conteúdo ao centro da célula.

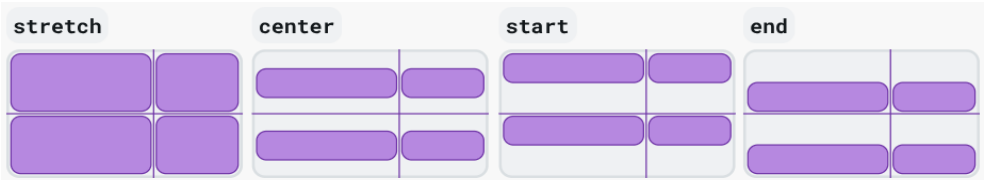


Figura 03: Opções da propriedade align-items

2. **Alinhamento dos Grid Items dentro do Container**

A propriedade **justify-content** alinha os **grid items** horizontalmente dentro do **grid container** (Figura 04):

- **stretch**: expande os itens para preencherem a largura do container.
- **center**: alinha os itens no centro do container.
- **start**: alinha os itens no início do container.
- **end**: alinha os itens no final do container.
- **space-between**: distribui os itens com espaço entre eles.
- **space-around**: distribui os itens com espaço ao redor deles.
- **space-evenly**: distribui os espaços igualmente entre os itens.

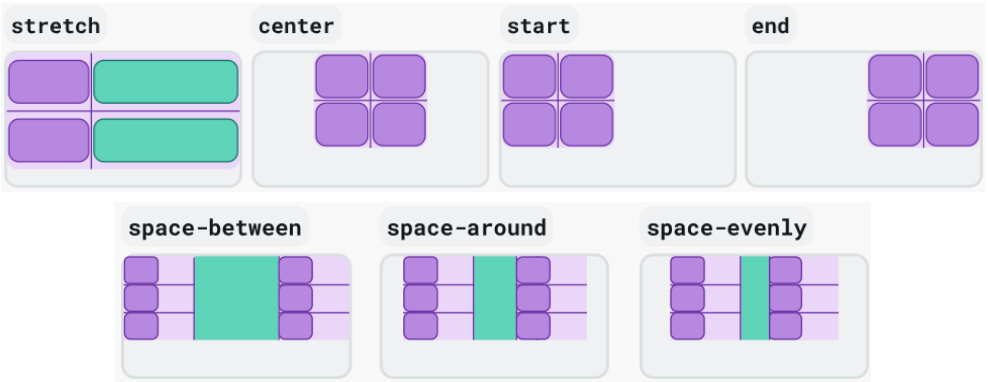


Figura 04: Opções da propriedade o justify-content

A propriedade **align-content** alinha os **grid items** verticalmente dentro do **grid container** (Figura 05):

- **stretch**: expande os itens para preencherem a altura do container.
- **start**: alinha os itens no topo do container.
- **end**: alinha os itens na base do container.
- **center**: alinha os itens no centro do container.
- **space-around**: distribui os itens com espaço ao redor deles.
- **space-between**: distribui os itens com espaço entre eles.
- **space-evenly**: distribui os espaços igualmente entre os itens.

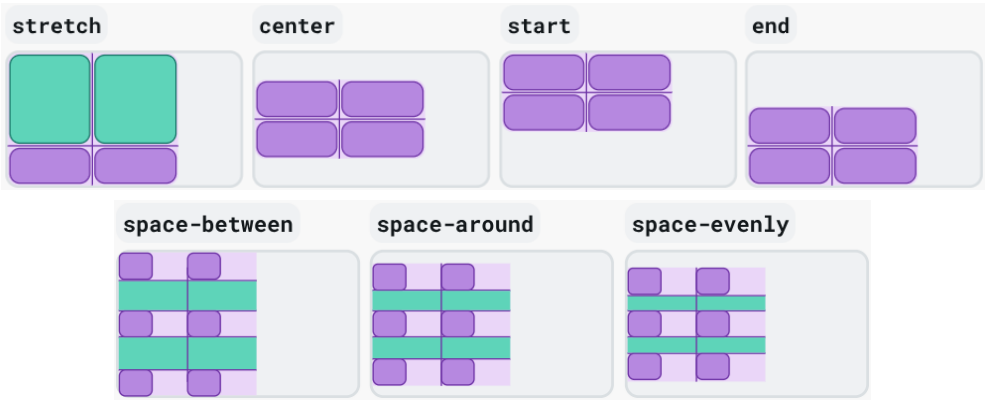


Figura 05: Opções da propriedade o align-content

No CSS Grid Layout, o posicionamento com áreas nomeadas permite definir um layout de forma mais intuitiva.

- **display** (no **grid container**): Define o elemento como um grid container, ativando o modelo de layout baseado em grade.
- **grid-template-areas** (no **grid container**): Especifica áreas nomeadas dentro do grid, facilitando a estruturação do layout.
- **grid-template-columns** (no **grid container**): Define o número e largura das colunas do grid.
- **grid-template-rows** (no **grid container**): Define a quantidade e altura das linhas do grid.

- **grid-area** (no **grid item**): Associa um item do grid a uma das áreas nomeadas definidas em **grid-template-areas**.

Ao combinar **grid-template-areas** com **grid-template-columns** e **grid-template-rows**, é possível estruturar layouts flexíveis e organizados. Os **grid items** são então posicionados com **grid-area**, garantindo um controle mais claro e semântico sobre o design.

Exemplo

- A Figura 06 ilustra um layout simples utilizando CSS Grid Layout. Esse layout representa a estrutura básica de uma página que conterá imagens e textos.

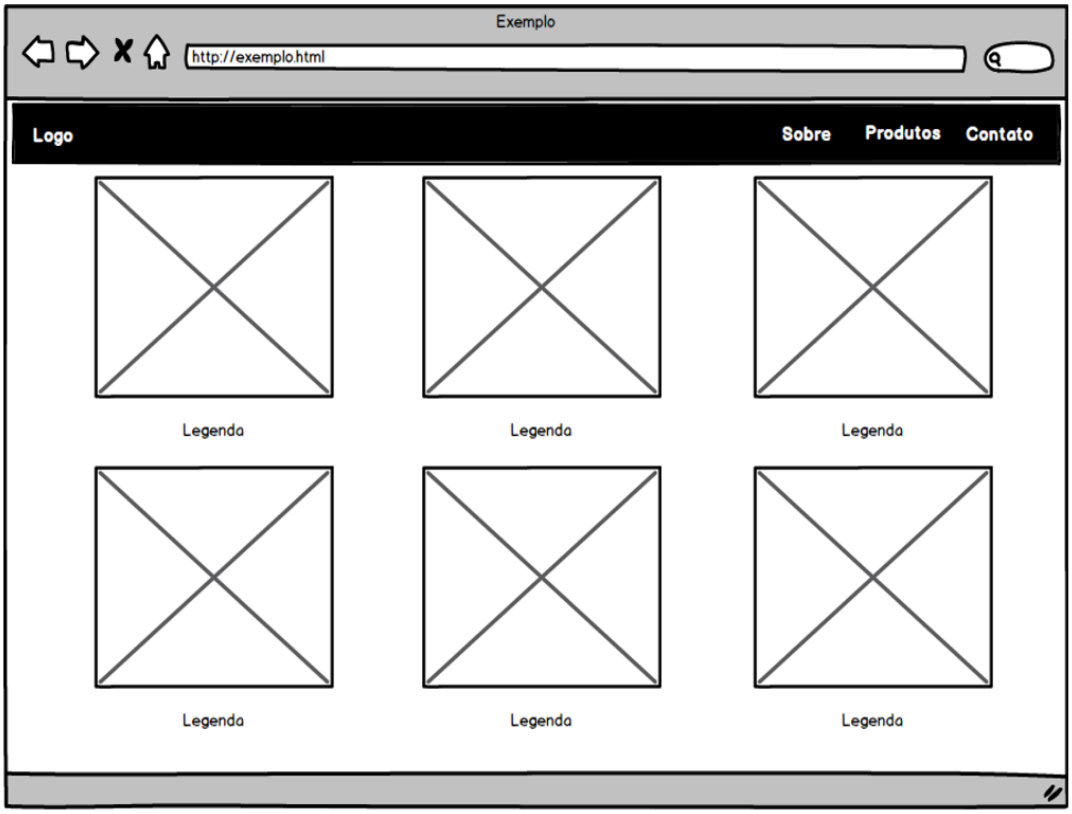


Figura 06: Protótipo do exemplo

- Passo 1: Estrutura Básica

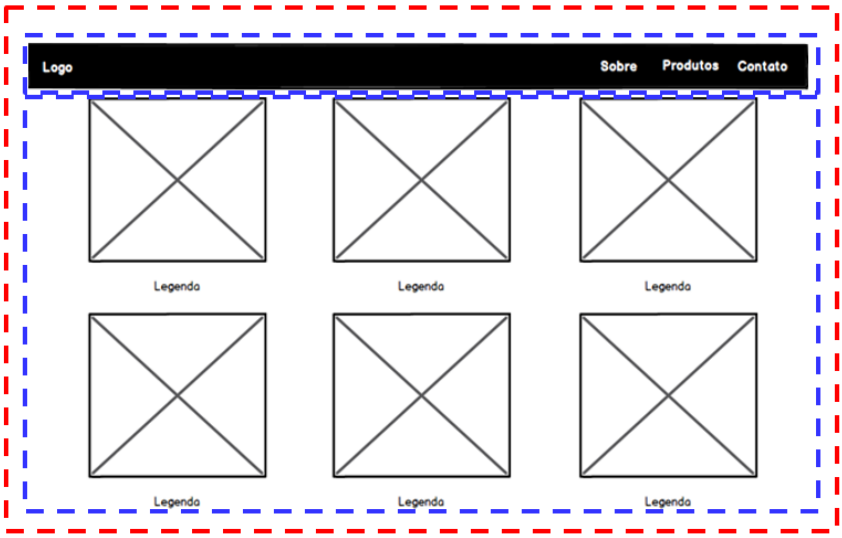


Figura 07: Abstração da estrutura básica

```html
<div id="container">
    <div id="cabecalho">
    </div>
    <div id="galeria">
    </div>
</div>
```

Trecho do código HTML do exemplo

```css
#container{
    display: grid;
    grid-template-areas: "cabecalho"
                         "galeria";
}

#cabecalho{
    grid-area: cabecalho;
}

#galeria{
    grid-area: galeria;
}
```

Trecho do código CSS do exemplo

- Passo 2: Estrutura do Cabeçalho

Figura 08: Abstração do cabeçalho

```html
<div id="cabecalho">
    <div id="logo">
        <a href="#">Logo</a>
    </div>
    <div id="opcoes">
        <a href="#">Sobre</a>
        <a href="#">Produtos</a>
        <a href="#">Contatos</a>
    </div>
</div>
```

Trecho do código HTML do exemplo

```css
#cabecalho{
    grid-area: cabecalho;
    background-color: black;

    /*sub-grid do #container*/
    display: grid;
    grid-template-areas: "logo opcoes";
    justify-content: space-between;
    padding: 15px;
}

#logo{
    grid-area: logo;

}

#opcoes{
    grid-area: opcoes;

}

a{
    color: white;
    text-decoration: none;
}

#logo a{
    padding-right: 10px;
}

#opcoes a{
    padding-left: 10px;
}
```

Trecho do código CSS do exemplo

- Passo 3: Estrutura da Galeria

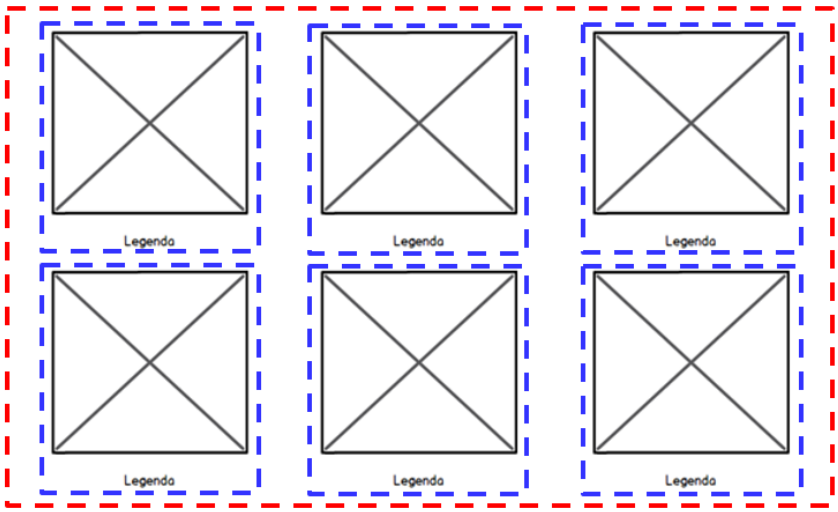


Figura 09: Abstração da galeria

```html
<div id="galeria">
    <div>
        <img src="imagens/produto01.jpg">
        <p>Legenda</p>
    </div>
    <div>
        <img src="imagens/produto02.jpg">
        <p>Legenda</p>
    </div>
    <div>
        <img src="imagens/produto03.jpg">
        <p>Legenda</p>
    </div>
    <div>
        <img src="imagens/produto04.jpg">
        <p>Legenda</p>
    </div>
    <div>
        <img src="imagens/produto05.jpg">
        <p>Legenda</p>
    </div>
    <div>
        <img src="imagens/produto06.jpg">
        <p>Legenda</p>
    </div>
</div>
```

Trecho do código HTML do exemplo

```css
#galeria{
    grid-area: galeria;
    justify-content: space-evenly;
    padding-top: 10px;

    /*sub-grid do container galeria*/
    display: grid;
    grid-template-columns: 30vw 30vw 30vw;
}

#galeria div p{
    text-align: center;
    padding-top: 5px;
    padding-bottom: 10px;
}

#galeria div img{
    width: 80%;
    margin-left: 10%;
    height: 30vh;
}
```

Trecho do código CSS do exemplo

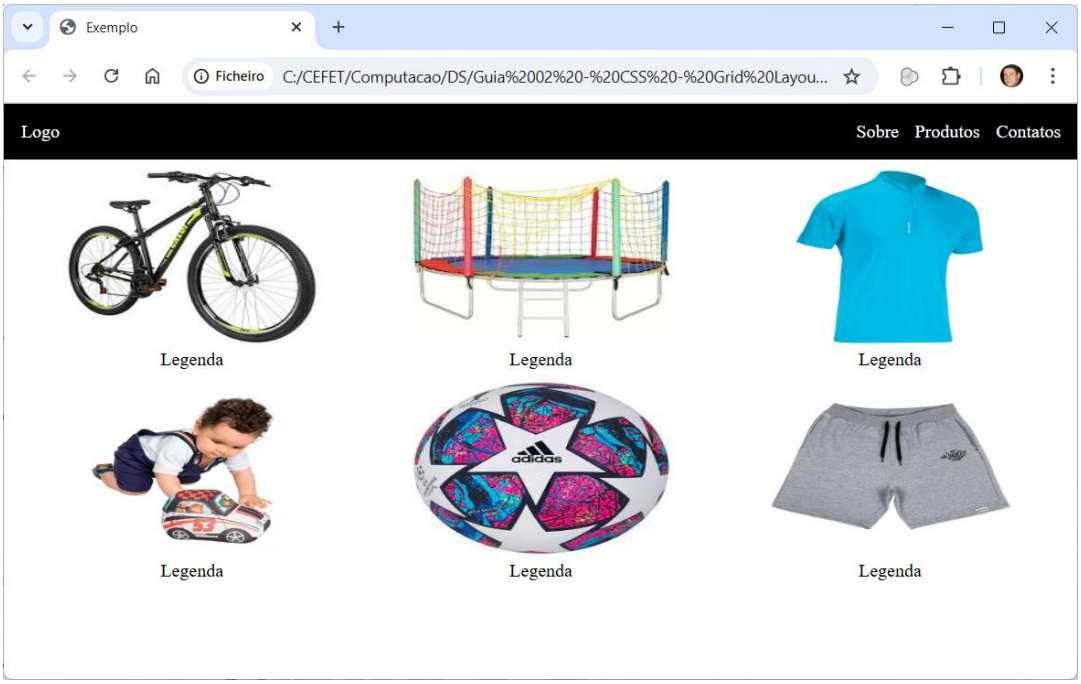


Figura 10: Página com conteúdo

Nota HTML

- O elemento é uma divisão genérica para conteúdo de fluxo. Ele é utilizado para agrupar elementos, facilitando a organização do layout e a aplicação de estilos.

Nota CSS

- As unidades relativas vw (*viewport width*) e vh (*viewport height*) correspondem, respectivamente, à largura e à altura da área visível da página (*viewport*).

Exercícios

1. Utilizando as propriedades que julgar necessárias do CSS Grid Layout, crie uma página HTML que tenha um layout semelhante ao protótipo abaixo:

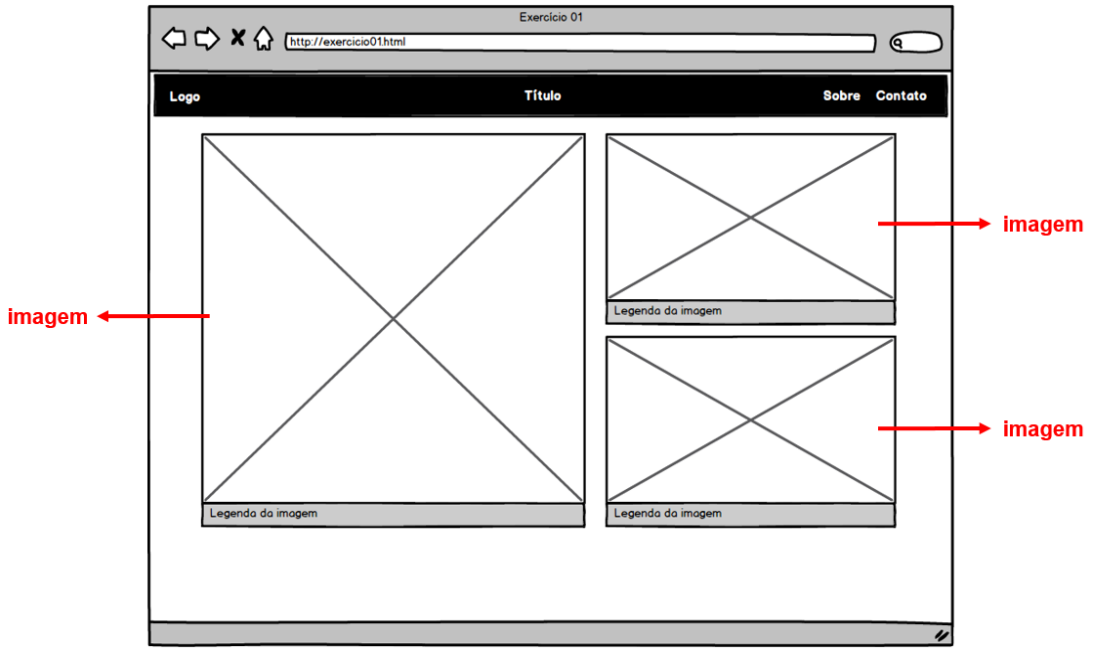


Figura 11: Protótipo do exercício 01

2. Utilizando as propriedades que julgar necessárias do CSS Grid Layout, crie uma página HTML que tenha um layout semelhante ao protótipo abaixo:

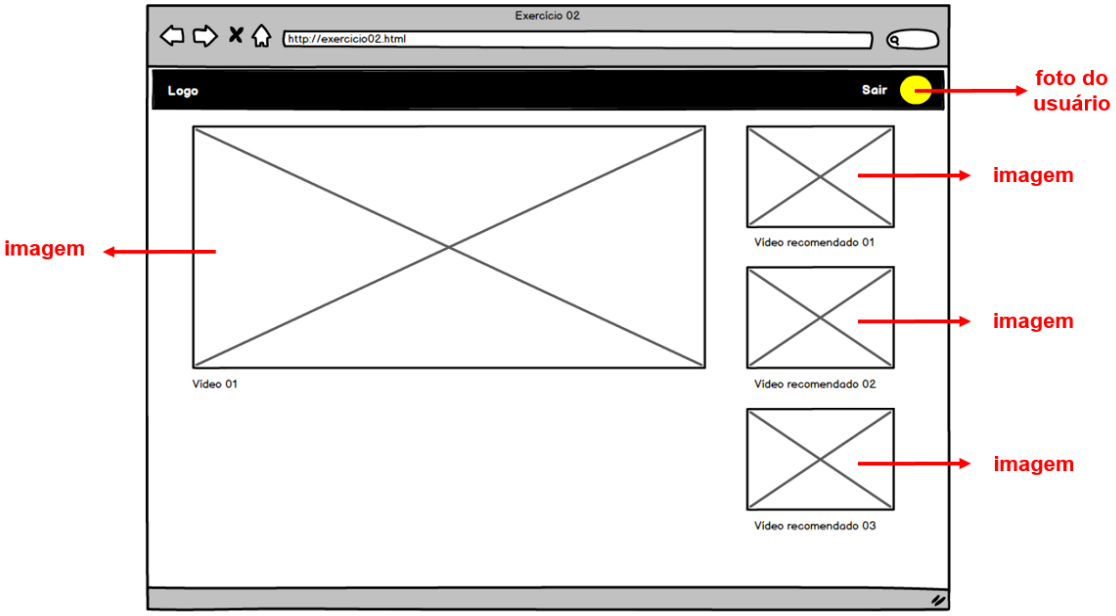
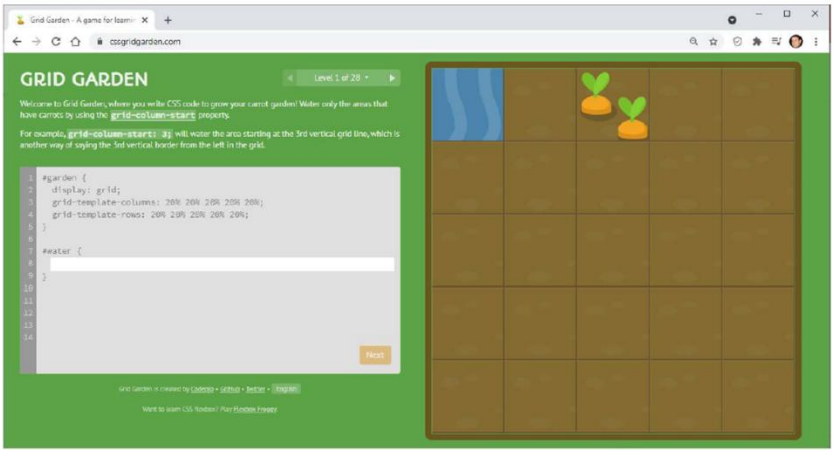


Figura 12: Protótipo do exercício 02

3. Teste seus conhecimentos sobre as propriedades do CSS Grid Layout com o jogo "Grid Garden".



https://cssgridgarden.com

Referências:

Este guia foi baseado nas postagens "CSS Grid: Alinhando Elementos" e "CSS Grid – Um Guia Interativo", escritas por Hugo de Oliveira e Akira Hanashiro, respectivamente. Os artigos abordam diversas técnicas para alinhar elementos dentro de um grid container. Os textos completos estão disponíveis em:

- CSS Grid – Um Guia Interativo (Parte 1 – Containers)
  https://www.treinaweb.com.br/blog/css-grid-um-guia-interativo-parte-1-containers
- CSS Grid: alinhando elementos
  https://triangulo.dev/posts/css-grid-alinhando-elementos-na-pratica/#alinhando-os-itens-do-grid-com-align-items-e-justify-items

Material complementar:

- Todos os recursos e propriedades do CSS Grid Layout podem ser consultados na documentação a seguir:
  https://www.origamid.com/projetos/css-grid-layout-guia-completo
- Aprenda CSS Grid em 30 Minutos
  https://www.youtube.com/watch?v=j_fzKbfH7W4
- Unidades CSS relativas: VW, VH, VMAX, VMIN (CSS3)
  https://www.youtube.com/watch?v=g__c-7M9Xzk