Protótipo

Logo

TRANSFORMANDO O BRASIL PARA A POPULAÇÃO TRANS!

Nossa luta é por um país onde pessoas trans tenham acesso digno à saúde, oportunidades e respeito. Aqui, você encontra informações sobre direitos, atendimento médico, capacitação profissional e inclusão social. Mas não paramos por aí: também expomos dados e estatísticas sobre as violências e desigualdades que essa comunidade enfrenta todos os dias. Queremos abrir os olhos da sociedade e provocar a mudança. Junte-se a nós e faça parte dessa revolução!

INICIATIVAS

A **ANTRA** (antrabrasil.org) fortalece a população trans no Brasil, promovendo acesso à saúde, educação e políticas públicas, além de denunciar a transfobia e desigualdades.

Já a **AzMina** (azmina.com.br) combate a violência de gênero com jornalismo independente, dados e tecnologia, empoderando mulheres e pessoas LGBTQIA+ com informação de qualidade.

Dois portais essenciais para quem busca justiça, igualdade e transformação social. Acesse, se informe e faça parte da mudança!

Antra
Associação Nacional de Travestis e Transexuais

AZMina revista

EMPREGOS

Transforme o futuro com inclusão e oportunidades! O **EmpoderaTrans** conecta talentos trans e travestis ao mercado de trabalho, promovendo empregabilidade e diversidade.

Já o **EducaTransforma** oferece cursos e capacitações para fortalecer carreiras e ampliar horizontes. Acesse, apoie e faça parte da mudança!

EDUCA TRANS FORMA

Empodera Trans

ESTATÍSTICAS

- O Brasil é o país que mais mata pessoas trans, representando cerca de 40% dos homicídios no mundo (Transgender Europe – TGEU).
- 80% das pessoas trans já sofreram violência física, psicológica ou sexual (IBGE e Ministério da Saúde).
- 30% das pessoas trans entrevistadas em 2021 relataram agressões por parte da polícia (Anistia Internacional e Human Rights Watch).
- A expectativa de vida de pessoas trans no Brasil é de aproximadamente 35 anos, comparada à média nacional de 75 anos (UFBA).

FALE CONOSCO

E-mail

Telefone

Nos dê seu feedback, indique iniciativas ou venha fazer parte do nosso time!

ENVIAR CANCELAR

Instagram | Tiktok

Devs

Tutorial

Como fazer o seu próprio site informativo usando HTML5, CSS e JavaScript

Para criar um site informativo o primeiro passo é criar a sua base em HTML5, vamos começar criando uma estrutura base.

Criando a Estrutura Básica do HTML

O primeiro passo é criar a estrutura básica do HTML. Todo documento HTML começa com a declaração do tipo de documento (<!DOCTYPE html>) e a tag <html>, que indica o início do código HTML.

```
<!DOCTYPE html>

<html lang=”pt-br”>
```

Dentro da tag <html>, você define o idioma do conteúdo (neste caso, português) com o atributo lang=”pt-br”.

Lembre-se que toda tag que for aberta deve ser fechada, então eu recomendo que feche ela assim que abrir a tag. Para fechar você vai usar uma / antes do nome da tag, no caso do html fica </html>.

Após esse passo seu código estará assim:

```
<!DOCTYPE html>

<html lang= “pt-br”>

</html>
```

Cabeçalho do Documento (<head>)

O cabeçalho do HTML é onde você inclui meta informações sobre a página, links para fontes, estilos (CSS) e o título da página que aparece na aba do navegador.

```
<head>

    <meta charset=”UTF-8”>

    <meta name=”viewport” content=”width=device-width, initial-scale=1.0”>

     <title>Futuro Trans Brasil</title>

     <link rel= "shortcut icon" type="image/x-icon" href="assets/transgender.ico">
```

```
    <a href="https://www.flaticon.com/free-icons/trans" title="trans icons"></a>
    <link
href="https://cdnjs.cloudflare.com/ajax/libs/font-awesome/5.15.4/css/all.min.css"
rel="stylesheet">
    <link rel="preconnect" href="https://fonts.googleapis.com">
    <link rel="preconnect" href="https://fonts.gstatic.com" crossorigin>
    <link
href="https://fonts.googleapis.com/css2?family=Anaheim:wght@400..800&display=swa
p" rel="stylesheet">
    <link
href="https://fonts.googleapis.com/css2?family=Raleway:wght@700&display=swap"
rel="stylesheet">
    <link
href="https://fonts.googleapis.com/css2?family=Blinker:wght@100;200;300;400;600;70
0;800;900&display=swap" rel="stylesheet">
    <link rel="stylesheet" href="css/style.css">
    <script src="javascript/script.js" defer></script>
</head>
```

Aqui, incluímos:

• O charset UTF-8 para garantir que todos os caracteres especiais sejam corretamente exibidos.

• O meta tag viewport para tornar o site responsivo (adequado para diferentes dispositivos).

• O título: “Futuro Trans Brasil”, que aparecerá na aba do navegador.

• O link para o arquivo CSS e Javascript (vamos trabalhar com eles nas próximas etapas).

• Os links de fontes que serão usadas no CSS para estilizar o site.

Após esse passo seu código estará assim:

```
<!DOCTYPE html>
<html lang="pt-br">
  <head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Futuro Trans Brasil</title>
    <link rel="shortcut icon" type="image/x-icon" href="assets/transgender.ico">
    <a href="https://www.flaticon.com/free-icons/trans" title="trans icons"></a>
```

```
    <link
href="https://cdnjs.cloudflare.com/ajax/libs/font-awesome/5.15.4/css/all.min.css"
rel="stylesheet">

    <link rel="preconnect" href="https://fonts.googleapis.com">

    <link rel="preconnect" href="https://fonts.gstatic.com" crossorigin>

    <link
href="https://fonts.googleapis.com/css2?family=Anaheim:wght@400..800&display=swa
p" rel="stylesheet">

    <link
href="https://fonts.googleapis.com/css2?family=Raleway:wght@700&display=swap"
rel="stylesheet">

   <link
href="https://fonts.googleapis.com/css2?family=Blinker:wght@100;200;300;400;600;70
0;800;900&display=swap" rel="stylesheet">

    <link rel="stylesheet" href="css/style.css">

    <script src="javascript/script.js" defer></script>

  </head>

</html>
```

Lembrete: É importante tabular o código para que a visualização e organização dele fique melhor, assim caso haja possíveis erros, você conseguirá encontrar o erro com mais facilidade. As tabulações são o recuo antes das tags, como ocorre no caso do head e das tags meta.

Corpo do Documento (<body>)

O corpo do documento é onde o conteúdo visível da página será colocado, ou seja, tudo que irá aparecer no nosso site. A estrutura interna pode ser dividida em várias seções usando as tags <section>, <article>, e outras. Vamos ver cada uma delas.

Para começar a montar o body do nosso código você irá abrir e fechar a tag body.

```
<body>

</body>
```

1. Header (Cabeçalho)

Aqui colocamos um cabeçalho simples com a logo do site:

```
<header>

   <img src="assets/logo.png" alt="logo" class="logo">

</header>
```

Para colocar a logo no site, foi usada a tag img, essa tag é utilizada para inserir imagens no nosso site. A tag alt acompanha a imagem para garantir que o site seja acessível. Vale ressaltar que no nome da imagem está indicado também a pasta em que ela está no computador para quando ela for acessada não haja erros, se você deixar o arquivo html e a imagem na mesma pasta não é necessário indicar o caminho.

2. Seção “Transformando”

Essa seção contém uma imagem de protesto e um texto explicativo sobre a missão do site:

```
<section class=”transformando”>
   <img src=”assets/protesto3.jpg” alt=”Manifestação pelos direitos trans”>
   <h2 class="transformandoTitle">TRANSFORMANDO O BRASIL PARA A POPULAÇÃO TRANS!</h2>
    <p> Nossa luta é por um país onde pessoas trans tenham acesso digno à saúde, oportunidades e respeito. Aqui, você encontra informações sobre direitos, atendimento médico, capacitação profissional e inclusão social. Mas não paramos por aí: também expomos dados e estatísticas sobre as violências e desigualdades que essa comunidade enfrenta todos os dias. Queremos abrir os olhos da sociedade e provocar a mudança. Junte-se a nós e faça parte dessa revolução!
   </p>
</section>
```

A tag <img> é usada para inserir uma imagem, enquanto o texto é colocado dentro da tag <p> (parágrafo). O título da seção é definido com a tag <h2>. O elemento “class” adicionado as tags “section” e “h2” será usado quando formos criar o arquivo de CSS.

3. Seção “Iniciativas”

Dividida em dois blocos:

• Bloco esquerdo (bloco_esq1): Apresenta informações sobre a ANTRA e AzMina, com links externos.

• Bloco direito (bloco_dir): Contém imagens representativas dessas iniciativas.

As informações que estão dentro do <!-- → ficam como comentários para que haja uma melhor orientação quando você for ler o código e verificar erros ou alterar informações.

```
<!—Seção de iniciativas -->
<section class=”bloco_pri1”> <!—Bloco Principal –>
        <article class=”bloco_esq1”> <!—Bloco Esquerdo –>
                <h2>INICIATIVAS</h2>
```

 A ANTRA fortalece a população trans no Brasil, promovendo acesso à saúde, educação e políticas públicas, além de denunciar a transfobia e desigualdades.

Já a <a AzMina combate a violência de gênero com jornalismo independente, dados e tecnologia, empoderando mulheres e pessoas LGBTQIA+ com informação de qualidade.

Dois portais essenciais para quem busca justiça, igualdade e transformação social. Acesse, se informe e faça parte da mudança!

</article>

<article class=”bloco_dir”> <!—Bloco Direito –>

<img src=”assets/antra.png” alt=”antra”>

<img src=”assets/azmina.png” alt=”azmina”>

</article>

</section>

Os conceitos usados nesta seção são os mesmos da anterior sendo a tag <img> usada para inserir uma imagem, enquanto o texto é colocado dentro da tag (parágrafo). O título da seção é definido com a tag <h2>. O elemento “class” adicionado as tags “section” e “article” será usado quando formos criar o arquivo de CSS. A novidade desse trecho é a tag article, ela foi utilizada para separar os elementos em dois blocos dentro da section.

4. Seção empregos

Os conceitos usados nesta seção são os mesmos da anterior sendo a tag <img> usada para inserir uma imagem, enquanto o texto é colocado dentro da tag (parágrafo). O título da seção é definido com a tag <h2>. O elemento “class” adicionado as tags “section” e “article” será usado quando formos criar o arquivo de CSS. A tag article foi utilizada para separar os elementos em dois blocos dentro da section o que irá facilitar a aplicação do CSS

<!—Seção de empregos –>

<section class=”bloco_pri2”> <!—Bloco Principal –>

<article class=”bloco_esq2”> <!—Bloco Esquerdo –>

<h2>EMPREGOS</h2>

Transforme o futuro com inclusão e oportunidades! O EmpoderaTrans conecta talentos trans e travestis ao mercado de trabalho, promovendo empregabilidade e diversidade.

Já o EducaTransforma oferece cursos e capacitações para fortalecer carreiras e ampliar horizontes. Acesse, apoie e faça parte da mudança!

```
        </article>
        <article class=”bloco_dir”> <!—Bloco Direito –>
            <img src=”assets/empodera_trans.png” alt=”azmina”>
            <img src=”assets/educa_transforma.png” alt=”antra”>
        </article>
</section>
<section class=”protesto”>
    <img src=”assets/protesto1.jpg” alt=”antra” class=”protesto”>
</section>
```

5. Seção estatística

Utiliza um slider feito com JavaScript para apresentar dados sobre a realidade da população trans no Brasil. Cada slide apresenta diferentes estatísticas, por enquanto será criado apenas o html que iremos adaptar para o slider quando formos criar o JavaScirpt.

```
<!—Seção de estatisticas –>
<section class=”estatisticas”>
        <article class=”slider”>
            <nav class=”slide”>
            <h2 class=”estatisticas-title”>ESTATÍSTICAS</h2>
            <ul>
            <li>O Brasil é o país que mais mata pessoas trans,representando cerca de 40% dos homicídios no mundo (Transgender Europe – TGEU).</li>
            <li>80% das pessoas trans já sofreram violência física, psicológica ou sexual (IBGE e Ministério da Saúde).</li>
            <li>30% das pessoas trans entrevistadas em 2021 relataram agressões por parte da polícia (Anistia Internacional e Human Rights Watch).</li>
            <li>A expectativa de vida de pessoas trans no Brasil é de aproximadamente 35 anos, comparada à média nacional de 75 anos (UFBA).</li>
            </ul>
            </nav>
            <nav class=”slide”>
            <h2 class=”estatisticas-title”>ESTATÍSTICAS</h2>
            <ul>
```

```
                <li>Cerca de 20% dos estudantes trans abandonam a escola devido à discriminação e ao bullying (INEP em 2021)</li>

                <li>Segundo um relatório da UNESCO de 2020, jovens trans e não-binários têm uma taxa de matrícula inferior em comparação a jovens cisgêneros</li>

                <li>Apenas 20% das pessoas trans conseguem empregos formais; 80% trabalham em ocupações informais ou de baixo rendimento, incluindo prostituição (UFBA).</li>

            </ul>

            </nav>

            <nav class="slide">

            <h2 class="estatisticas-title">ESTATÍSTICAS</h2>

            <ul>

                <li>80% das pessoas trans no Brasil relataram sintomas de depressão; 60% relataram ansiedade, frequentemente agravadas por discriminação e violência (Lancet Psychiatry, 2020).</li>

                <li>70% das pessoas trans não têm acesso adequado a serviços de saúde e enfrentam discriminação no atendimento (IBGE e Ministério da Saúde).</li>

                <li>70% das pessoas trans consideraram o suicídio, citando discriminação e violência como fatores de estresse (Conselho Federal de Psicologia).</li>

            </ul>

            </nav>

        </article>

        <article class="dot-container">

            <span class="dot active" onclick="currentSlide(0)"></span>

            <span class="dot" onclick="currentSlide(1)"></span>

            <span class="dot" onclick="currentSlide(2)"></span>

        </article>

</section>
```

Aqui temos:

- Section definindo o bloco total
- Article definindo o slider
- Nav definindo os slides
- H2 definindo o título

• ull iniciando uma lista e li definindo os elementos da lista

• é usado para criar os indicadores de navegação dos slides.

6. Seção contatos e rodapé

Inclui campos para e-mail, telefone e mensagem, com botões para enviar ou cancelar. Contém links para redes sociais e os nomes dos criadores do projeto.

```
<!—Seção de contatos –>
    <form>
      <h2 class=”fale”>FALE CONOSCO</h2>
     <input type=”email” class=”azul” name=”email” placeholder=”E-mail”required=””>
     <input type=”tel” class=”azul” name=”telefone”placeholder=”Telefone”required=””>
    <textarea id=”mensagem”class=”azul”rows=”6placeholder=”Nos dê seu feedback” required=””></textarea>
     <section class=”botao”>
     <input class=”button” id=”enviar” type=”submit” value=”Enviar”>
     <input class=”button” id=”cancelar” type=”reset” value=”Cancelar”>
     </section>
</form>
<hr>
<footer>
      <p>Instagram | Tiktok</p>
      <br>
      <p>Ana, Dayô e Enzo</p>
      <a href=https://github.com/Enssoo/Pichu>Github</a>  Github do projeto 
</footer>
```

Aqui temos:

• form define um formulário para envio de feedbacks, opiniões, etc.

• h2 aplica o título do formulário.

• input type abre caixas de mensagens específicas para a adição de dados como e-mail e telefone. Type garante que o formato do dado estará correto.

• textarea aplica uma caixa de texto para ser digitada uma mensagem.

• input class Button cria um botão.

• hr pula uma linha.

• footer define o rodapé.

• p adiciona os textos ao rodapé.

• a adiciona o link do github onde você pode encontrar os códigos, imagens e links utilizados.

7. Botão de topo da tela

Para finalizar o HTML, vamos criar um botão para voltar ao topo da tela, que será funcional usando JavaScript.

<button id="pgUp"><img id="imgPgUp" src="assets/pgUp.png"></button>

- <button id="pgUp"> → Define o botão e adiciona a ele o ID “pgUp”, que será usado depois para adicionar as funcionalidades em JavaScript.
- <img id="imgPgUp" src="assets/pgUp.png"> → Define que o botão será representado pela imagem pgUp.png. Será a imagem que ficará visível para o usuário. E também define um ID para que essa imagem possa ser estilizada no CSS.

Criando o CSS

O CSS (Cascading Style Sheets) é a linguagem usada para estilizar páginas da web, definindo a aparência e o layout dos elementos HTML. Com ele, é possível modificar cores, fontes, tamanhos, espaçamentos, posicionamentos e tornar o site responsivo para diferentes dispositivos. Sua sintaxe segue a estrutura seletor { propriedade: valor; }, onde o seletor define o elemento a ser estilizado, a propriedade indica o que será modificado (como cor ou tamanho) e o valor especifica a configuração desejada. Por exemplo, p { color: blue; font-size: 16px; } define que todos os parágrafos (<p>) terão o texto na cor azul e tamanho de 16 pixels.

A seguir vou te mostrar os elementos e suas funções para o nosso código:

1. * {

   box-sizing: border-box;

   margin: 0;

   padding: 0;

   }

Aqui temos as definições gerais da página:

- *{} → é o seletor universal, todos os elementos de CSS aplicados a ele irão ser aplicados a todos os elementos da tela.
- box-sizing: border-box → serve para controle do tamanho dos elementos, com ele você pode incluir na altura e largura do elemento, o conteúdo, padding e borda, facilitando a organização do layout.

- margin = 0 → remove espaçamentos externos, garantindo que não haja margens padrão.
- padding = 0 → elimina espaços internos, para garantir que os elementos se adequem ao layout com mais facilidade.

2. body, html {

   margin: 0;

   padding: 0;

   height: 100%;

   }

Aqui temos as definições gerais da página:

- body, html {} → Define regras que serão aplicadas especificamente as tags <body> e <html>, garantindo um controle mais preciso do layout.
- margin: 0; → Remove espaçamentos externos, eliminando as margens padrão que alguns navegadores adicionam. Isso evita que a página tenha um espaço indesejado ao redor.
- padding: 0; → Remove espaços internos, garantindo que o conteúdo ocupe toda a área disponível sem espaçamentos extras.
- height: 100%; → Define que a altura do <html> e do <body> ocupem 100% da tela, o que é útil para layouts que precisam preencher toda a página.

3. header {

   display: flex;

   justify-content: center;

   align-items: center;

   height: 20%;

   background-image: linear-gradient(45deg, #bfdeff, #ffbfed, #ffffff, #ffbfed, #bfdeff);

   width: 100%;

   }

Aqui temos as definições do header, a área do cabeçalho da página:

- display: flex; → Ativa o Flexbox, permitindo um controle mais fácil sobre o alinhamento dos elementos dentro do header.
- justify-content: center; → Centraliza os elementos horizontalmente dentro do cabeçalho.
- align-items: center; → Centraliza os elementos verticalmente, garantindo que fiquem alinhados no meio do header.
- height: 20%; → Define que o header ocupará 20% da altura total da página, ajustando seu tamanho proporcionalmente.

- background-image: linear-gradient(45deg, #bfdeff, #ffbfed, #ffffff, #ffbfed, #bfdeff); → Aplica um degradê diagonal (45°) com cores suaves, criando um efeito visual bonito e moderno.
- width: 100%; → Faz com que o header ocupe toda a largura da página, garantindo um ajuste total no topo do layout.

4. .logo {

   width: 8%;

   }

Aqui temos as definições da classe .logo, que estiliza a logo da página:

- width: 8%; → Define que a largura da logo ocupará 8% da largura total do elemento pai, nesse caso o header. Isso garante que a logo fique proporcional ao tamanho do cabeçalho, ajudando na responsividade e na boa visualização do site.

5. section {

   width: 100%;

   font-family: "Anaheim", serif;

   font-optical-sizing: auto;

   font-style: normal;

   }

Aqui temos as definições da section, que estiliza uma seção da página:

- width: 100%; → Faz com que a seção ocupe toda a largura disponível, garantindo que ela se estenda por toda a página.
- font-family: "Anaheim", serif; → Define a fonte do texto como `"Anaheim"`, uma tipografia específica que traz um estilo diferenciado. Caso essa fonte não esteja disponível, o navegador pode usar uma alternativa serifada.
- font-optical-sizing: auto; → Ajusta automaticamente o tamanho óptico da fonte para melhorar a legibilidade e aparência em diferentes tamanhos.
- font-style: normal; → Mantém a fonte no estilo normal, sem itálico ou outros efeitos.

6. h2 {

   font-family: 'Raleway', sans-serif;

   font-weight: 700;

   font-size: 4.125em;

   color: #91fffc;

   text-shadow: 4px 4px 0px rgba(0, 74, 173, 0.7);

}

Aqui temos as definições do h2, que estiliza títulos de segundo nível na página:

- font-family: 'Raleway', sans-serif; → Define a fonte do h2 como Raleway, uma tipografia moderna e limpa. Caso ela não esteja disponível, será usada uma alternativa sem serifa (`sans-serif`).
- font-weight: 700; → Define o peso da fonte como negrito, tornando o título mais destacado.
- font-size: 4.125em; → Ajusta o tamanho do texto para 4.125 vezes o tamanho base da fonte, garantindo um título grande e chamativo.
- color: #91fffc; → Define a cor do texto como um tom de azul claro, trazendo um efeito visual diferenciado.
- text-shadow: 4px 4px 0px rgba(0, 74, 173, 0.7); → Adiciona uma sombra ao texto, deslocada 4px para a direita e 4px para baixo, com uma leve transparência azulada, criando um efeito tridimensional.

7. a {

   text-decoration: none;

   color: #7176d1;

   font-weight: bold;

   }

Aqui temos as definições para o elemento a, que estiliza os links da página:

- text-decoration: none; remove o sublinhado padrão dos links, deixando o visual mais limpo.
- color: #7176d1; define a cor do texto como um tom de azul arroxeado, dando destaque ao link.
- font-weight: bold; deixa o texto mais espesso, tornando o link mais visível e chamativo.

8. .transformando {

   display: flex;

   flex-direction: column;

   justify-content: center;

   align-items: center;

   height: 110%;

   background-color: #516ad5;

   text-align: justify;

   }

Aqui temos as definições para a classe .transformando, que estiliza uma seção da página:

- display: flex; ativa o Flexbox, permitindo um melhor controle do posicionamento dos elementos dentro da seção.
- flex-direction: column; organiza os elementos em coluna, empilhando-os verticalmente.
- justify-content: center; centraliza os elementos verticalmente dentro da seção.
- align-items: center; centraliza os elementos horizontalmente, garantindo um alinhamento uniforme.
- height: 110%; define a altura da seção como 110% da altura do elemento pai, o que pode fazer com que ela ultrapasse a tela dependendo do layout.
- background-color: #516ad5; aplica um fundo azul arroxeado, trazendo contraste e destaque para a seção.
- text-align: justify; faz com que o texto dentro do elemento fique justificado, deixando as margens alinhadas para um visual mais organizado.

9. .transformando > img {

   display: flex;

   align-items: baseline;

   width: 80%;

   height: 50%;

   }

Aqui temos as definições para .transformando > img, que estiliza as imagens dentro do elemento .transformando:

- - `display: flex;` transforma a imagem em um contêiner flexível, permitindo que outras propriedades do Flexbox sejam aplicadas a ela.
- align-items: baseline; alinha os itens dentro do contêiner base na linha de base do texto, nesse caso ele alinha a imagem que é o elemento filho do contêiner flexível.
- width: 80%; faz com que a imagem ocupe 80% da largura do elemento pai, no caso o contêiner, garantindo que ela se ajuste ao layout.
- height: 50%; define que a altura da imagem será 50% da altura do elemento pai, ajudando no ajuste proporcional dentro da seção.
- Esse código permite que a imagem seja tratada como um elemento filho do contêiner flexível, facilitando o controle sobre seu posicionamento e tamanho no layout.

10. .transformando > p {

    font-size: 1.875em;

    width: 80%;

    font-family: "Blinker", serif;

    font-weight: 400;

    font-style: normal;

```
    color: white;
}
```

Aqui temos as definições para .transformando > p, que estiliza os parágrafos dentro do elemento .transformando:

- font-size: 1.875em; define o tamanho do texto como 1.875 vezes o tamanho padrão da fonte, tornando-o maior e mais legível.
- width: 80%; faz com que o parágrafo ocupe 80% da largura do elemento pai, evitando que o texto fique muito espalhado na tela.
- font-family: "Blinker", serif; define a fonte como "Blinker", e caso ela não esteja disponível, o navegador utilizará uma alternativa serifada.
- font-weight: 400; ajusta a espessura do texto para um peso regular, mantendo uma aparência equilibrada.
- font-style: normal; mantém o texto sem itálico ou outros estilos adicionais.
- color: white; altera a cor do texto para branco, garantindo um bom contraste caso o fundo seja escuro.

11.
```
.transformandoTitle {
    font-size: 3.5em;
    margin: 1.5% 0%;
    display: flex;
    align-items: baseline;
}
```

Aqui temos as definições para .transformandoTitle, que estiliza um título dentro da seção .transformando:

- font-size: 3.5em; define um tamanho grande para o texto, tornando-o mais chamativo e destacado.
- margin: 1.5% 0%; adiciona um espaçamento externo de 1.5% no topo e na base do título, enquanto mantém as laterais sem margens. Isso ajuda a separar o título de outros elementos sem alterar sua largura.
- display: flex; transforma o título em um contêiner flexível, permitindo o uso de propriedades do Flexbox.
- align-items: baseline; alinha os elementos internos do título com base na linha de base do texto, útil se houver outros elementos dentro do título que precisem de alinhamento preciso.

12.
```
.bloco_pri1 { /* Bloco Principal Iniciativas*/
    display: flex;
    flex-direction: row;
    width: 100%;
```

```
    height: 50em;
}
```

Aqui temos as definições para .bloco_pri1, que estiliza o bloco principal de iniciativas:

- display: flex; ativa o Flexbox, permitindo um melhor controle do posicionamento dos elementos dentro do bloco.
- flex-direction: row; organiza os elementos horizontalmente, alinhando-os em uma linha.
- width: 100%; faz com que o bloco ocupe toda a largura disponível do elemento pai.
- height: 50em; define a altura do bloco como 50 vezes o tamanho da fonte padrão, garantindo um espaço amplo para os conteúdos internos.

13.
```
.bloco_pri2 { /* Bloco Principal Empregos*/
    display: flex;
    flex-direction: row;
    width: 100%;
    height: 50em;
}
```

Aqui temos as definições para .bloco_pri2, que estiliza o bloco principal de empregos:

- display: flex; ativa o Flexbox, permitindo um melhor controle do posicionamento dos elementos dentro do bloco.
- flex-direction: row; organiza os elementos horizontalmente, alinhando-os em uma linha.
- width: 100%; faz com que o bloco ocupe toda a largura disponível do elemento pai.
- height: 50em; define a altura do bloco como 50 vezes o tamanho da fonte padrão, garantindo um espaço amplo para os conteúdos internos.

14.
```
.bloco_esq1 { /* Bloco Esquerdo Iniciativas */
    display: flex;
    flex-direction: column;
    flex-grow: 0;
    width: 70%;
    margin: 80px 40px;
}
```

Aqui temos as definições para .bloco_esq1, que estiliza o bloco esquerdo da seção de iniciativas:

- display: flex; ativa o Flexbox, permitindo um melhor controle do posicionamento dos elementos dentro do bloco.
- flex-direction: column; organiza os elementos em coluna, empilhando-os verticalmente.
- flex-grow: 0; impede que o bloco cresça automaticamente para ocupar o espaço disponível.
- width: 70%; define que o bloco ocupará 70% da largura do elemento pai.
- margin: 80px 40px; adiciona um espaçamento externo de 80px no topo e na base, e 40px nas laterais, garantindo um melhor posicionamento no layout.

15. .bloco_esq2 { /* Bloco Esquerdo Empregos */

    display: flex;

    flex-direction: column;

    flex-grow: 0;

    width: 70%;

    margin: 80px 40px;

}

Aqui temos as definições para .bloco_esq2, que estiliza o bloco esquerdo da seção de empregos:

- display: flex; ativa o Flexbox, permitindo um melhor controle do posicionamento dos elementos dentro do bloco.
- flex-direction: column; organiza os elementos em coluna, empilhando-os verticalmente.
- flex-grow: 0; impede que o bloco cresça automaticamente para ocupar o espaço disponível.
- width: 70%; define que o bloco ocupará 70% da largura do elemento pai.
- margin: 80px 40px; adiciona um espaçamento externo de 80px no topo e na base, e 40px nas laterais, garantindo um melhor posicionamento no layout.

16. .bloco_esq1 > p {

    font-size: 2.1875em;

    width: 85%;

    margin: 45px 20px;

}

Aqui temos as definições para .bloco_esq1 > p, que estiliza os parágrafos dentro do bloco esquerdo da seção de iniciativas:

- font-size: 2.1875em; define um tamanho de fonte grande para destacar o texto dentro do bloco.

- width: 85%; faz com que o parágrafo ocupe 85% da largura do elemento pai, evitando que o texto fique muito espalhado na tela.
- margin: 45px 20px; adiciona um espaçamento externo de 45px no topo e na base, e 20px nas laterais, garantindo uma melhor distribuição do conteúdo dentro do bloco.

17. .bloco_esq2 > p {

    font-size: 2.1875em;

    width: 85%;

    margin: 80px 0px 40px 0px;

    }

Aqui temos as definições para .bloco_esq2 > p, que estiliza os parágrafos dentro do bloco esquerdo da seção de empregos:

- font-size: 2.1875em; define um tamanho de fonte grande para destacar o texto dentro do bloco.
- width: 85%; faz com que o parágrafo ocupe 85% da largura do elemento pai, evitando que o texto fique muito espalhado na tela.
- margin: 80px 0px 40px 0px; adiciona um espaçamento externo de 80px no topo, 40px na base e mantém as laterais sem margem, garantindo um bom posicionamento dentro do bloco.

18. .bloco_dir { /* Bloco Direito */

    display: flex;

    flex-direction: column;

    justify-content: space-around;

    width: 25%;

    }

Aqui temos as definições para .bloco_dir, que estiliza o bloco direito da página:

- display: flex; ativa o Flexbox, permitindo um melhor controle do posicionamento dos elementos dentro do bloco.
- flex-direction: column; organiza os elementos em coluna, empilhando-os verticalmente.
- justify-content: space-around; distribui os elementos dentro do bloco, garantindo que eles fiquem espaçados de maneira uniforme.
- width: 25%; define que o bloco ocupará 25% da largura do elemento pai, mantendo um tamanho menor em relação ao bloco esquerdo.

19. .bloco_dir > img {

        width: 90%;

    }

Aqui temos as definições para .bloco_dir > img, que estiliza as imagens dentro do bloco direito:

- width: 90%; faz com que a imagem ocupe 90% da largura do elemento pai, garantindo que ela se ajuste ao layout sem ocupar todo o espaço disponível.

20. .protesto {

        width: 100%;

        height: 600px;

    }

Aqui temos as definições para .protesto, que estiliza um elemento que ocupa uma seção inteira da página:

- width: 100%; faz com que o elemento ocupe toda a largura disponível do elemento pai.
- height: 600px; define a altura fixa de 600 pixels, garantindo um tamanho padronizado para o conteúdo dentro dessa seção.

21. .estatisticas { /* Animação de transição*/

        display: flex;

        flex-direction: column;

        justify-content: center;

        align-items: center;

        height: auto;

        background-color: #ffc5ea;

        width: 100%;

        overflow: hidden;

        position: relative;

        padding: 80px;

    }

Aqui temos as definições para .estatisticas, que estiliza uma seção dedicada às estatísticas da página:

- display: flex; ativa o Flexbox, permitindo um melhor controle do posicionamento dos elementos dentro do bloco.
- flex-direction: column; organiza os elementos em coluna, empilhando-os verticalmente.
- justify-content: center; centraliza os elementos verticalmente dentro da seção.
- align-items: center; centraliza os elementos horizontalmente, garantindo um alinhamento uniforme.
- height: auto; permite que a altura do bloco se ajuste conforme o conteúdo interno.
- background-color: #ffc5ea; define um fundo rosa-claro para a seção, criando um visual diferenciado.
- width: 100%; faz com que o bloco ocupe toda a largura disponível do elemento pai.
- overflow: hidden; impede que qualquer conteúdo que ultrapasse o tamanho do elemento fique visível, útil para animações ou efeitos visuais.
- position: relative; estabelece que o bloco será posicionado em relação ao seu próprio espaço, permitindo posicionamentos absolutos de elementos internos se necessário.
- padding: 80px; adiciona um espaçamento interno uniforme de 80px, garantindo que o conteúdo não fique colado nas bordas.

22. .slider {

    display: flex;

    width: 100%;

    transition: transform 0.5s ease-in-out;

    }

Aqui temos as definições para .slider, que estiliza um elemento deslizante (carrossel ou galeria de imagens):

- display: flex; ativa o Flexbox, permitindo organizar os elementos internos em linha dentro do contêiner.
- width: 100%; faz com que o slider ocupe toda a largura disponível do elemento pai.
- transition: transform 0.5s ease-in-out; aplica uma transição suave ao transformar o elemento, garantindo que os movimentos de deslocamento ocorram em 0.5 segundos com um efeito de aceleração e desaceleração suave.

23. .slide {

    width: 100%;

    max-width: 80%;

    flex: 0 0 100%;

```
    box-sizing: border-box;
    padding: 20px;
    text-align: center;
    word-wrap: break-word;
    overflow-wrap: break-word;
    margin: 0 auto;
}
```

Aqui temos as definições para .slide, que estiliza cada item dentro de um carrossel ou galeria deslizante:

- width: 100%; faz com que o slide ocupe toda a largura disponível dentro do contêiner.
- max-width: 80%; limita a largura máxima do slide a 80% do espaço disponível, impedindo que ele fique muito largo em telas grandes.
- flex: 0 0 100%; garante que cada slide ocupe exatamente 100% da largura do contêiner pai dentro do Flexbox, sem crescer ou encolher automaticamente.
- box-sizing: border-box; inclui o padding e a borda dentro do cálculo da largura e altura do elemento, evitando que ele ultrapasse o tamanho esperado.
- padding: 20px; adiciona um espaçamento interno de 20px para evitar que o conteúdo fique muito próximo das bordas.
- text-align: center; centraliza todo o conteúdo de texto dentro do slide.
- word-wrap: break-word; permite que palavras longas sejam quebradas em várias linhas se necessário.
- overflow-wrap: break-word; faz a mesma função do word-wrap, garantindo que o texto não ultrapasse os limites do slide.
- margin: 0 auto; centraliza horizontalmente o slide dentro do contêiner pai.

24.
```
.estatisticas-title {
    margin: 0;
    font-size: 4.125em;
    padding-bottom: 2%;
}
```

Aqui temos as definições para .estatisticas-title, que estiliza o título da seção de estatísticas:

- margin: 0; remove qualquer espaçamento externo padrão ao redor do título.
- font-size: 4.125em; define um tamanho grande para o texto, tornando-o chamativo e destacando a importância da informação.
- padding-bottom: 2%; adiciona um espaçamento interno abaixo do título, garantindo um melhor espaçamento entre ele e os elementos seguintes.

25. ul {

    list-style-position: inside;

    text-align: justify;

    font-size: 1.5em;

    list-style-type: disc;

    padding-left: 1em;

    margin: 0;

}

Aqui temos as definições para ul, que estiliza listas não ordenadas na página:

- list-style-position: inside; faz com que os marcadores da lista fiquem dentro do bloco de texto, alinhados com o conteúdo.
- text-align: justify; alinha o texto da lista de forma justificada, garantindo um visual mais organizado e uniforme.
- font-size: 1.5em; aumenta o tamanho do texto da lista para melhor legibilidade.
- list-style-type: disc; define que os itens da lista usarão marcadores em formato de círculo sólido.
- padding-left: 1em; adiciona um pequeno espaçamento à esquerda da lista, garantindo uma separação sutil do restante do conteúdo.
- margin: 0; remove qualquer espaçamento externo padrão ao redor da lista, permitindo um melhor controle do layout.

26. li {

    margin: 0;

    padding: 0;

    font-size: 1.5em;

    line-height: 1.5;

}

Aqui temos as definições para li, que estiliza os itens individuais dentro de uma lista:

- margin: 0; remove qualquer espaçamento externo ao redor dos itens da lista.
- padding: 0; elimina qualquer espaçamento interno, garantindo um alinhamento mais uniforme.
- font-size: 1.5em; aumenta o tamanho do texto dos itens da lista para melhorar a legibilidade.
- line-height: 1.5; define um espaçamento entre linhas de 1.5 vezes o tamanho da fonte, tornando o texto mais espaçado e agradável à leitura.

27. .dot-container {

```
    display: flex;
    justify-content: center;
    margin-top: 20px;
}
```

Aqui temos as definições para .dot-container, que estiliza o contêiner dos indicadores de navegação (pontos) de um carrossel ou slider:

- display: flex; ativa o Flexbox, permitindo um melhor controle do posicionamento dos elementos dentro do contêiner.
- justify-content: center; centraliza os elementos horizontalmente dentro do contêiner, garantindo que os pontos fiquem alinhados no meio da tela.
- margin-top: 20px; adiciona um espaçamento superior de 20px, separando os pontos do restante do conteúdo acima.

```
28. .dot {
    height: 15px;
    width: 15px;
    margin: 0 5px;
    background-color: #bbb;
    border-radius: 50%;
    display: inline-block;
    cursor: pointer;
    transition: background-color 0.3s ease;
}
```

Aqui temos as definições para .dot, que estiliza os indicadores de navegação (pontos) de um carrossel ou slider:

- height: 15px; e width: 15px; definem um tamanho fixo de 15x15 pixels para cada ponto.
- margin: 0 5px; adiciona um espaçamento horizontal de 5px entre os pontos, garantindo que não fiquem colados.
- background-color: #bbb; define uma cor cinza clara como fundo padrão dos pontos.
- border-radius: 50%; transforma os pontos em círculos perfeitos.
- display: inline-block; mantém os pontos organizados em linha, sem ocupar um bloco inteiro.
- cursor: pointer; faz com que o cursor mude para o formato de clique ao passar sobre os pontos, indicando que são interativos.
- transition: background-color 0.3s ease; cria um efeito suave de transição na cor de fundo ao interagir com o ponto, melhorando a experiência visual.

Esse código garante que os pontos de navegação do carrossel fiquem bem visíveis, com um design arredondado e um efeito suave ao passar o mouse ou interagir com eles.

```
29. .dot.active {
        background-color: #7176d1;
    }
```

Aqui temos as definições para .dot.active, que estiliza o ponto ativo no carrossel ou slider:

- background-color: #7176d1; altera a cor do ponto ativo para um tom de roxo-azulado, diferenciando-o dos demais e indicando qual slide está sendo exibido no momento.

```
30. .estatisticas-title{
            margin: 0
    }
```

Aqui temos as definições para .estatisticas-title, que estiliza o título da seção de estatísticas:

- margin: 0; remove qualquer espaçamento externo ao redor do título, garantindo que ele fique alinhado sem margens adicionais.

```
31. form {
        display: flex;
        flex-direction: column;
        justify-content: center;
        align-items: center;
        width: 100%;
        margin: 0px 0px 40px 0px;
        padding: 4.5%;
    }
```

Aqui temos as definições para form, que estiliza a estrutura de um formulário na página:

- display: flex; ativa o Flexbox, permitindo um melhor controle do alinhamento dos elementos dentro do formulário.

- flex-direction: column; organiza os elementos do formulário em coluna, empilhando-os verticalmente.
- justify-content: center; centraliza os elementos verticalmente dentro do formulário.
- align-items: center; centraliza os elementos horizontalmente, garantindo um alinhamento uniforme.
- width: 100%; faz com que o formulário ocupe toda a largura disponível.
- margin: 0px 0px 40px 0px; adiciona um espaçamento inferior de 40px, separando o formulário dos elementos abaixo dele.
- padding: 4.5%; adiciona um espaçamento interno ao redor do formulário, garantindo que os campos não fiquem muito próximos das bordas.

Esse código garante que o formulário fique centralizado, bem estruturado e responsivo dentro da página.

```
32. .fale {
        margin: 2% 0%;
        font-size: 4em;
        color: #004aad;
        text-shadow: 4px 4px 0px #97a6e6;
    }
```

Aqui temos as definições para .fale, que estiliza um título ou texto de destaque na seção de contato:

- margin: 2% 0%; adiciona um espaçamento externo de 2% acima e abaixo do texto, mantendo as laterais sem margens.
- font-size: 4em; define um tamanho grande para o texto, tornando-o chamativo e de fácil leitura.
- color: #004aad; aplica uma cor azul escuro ao texto, garantindo contraste e destaque.
- text-shadow: 4px 4px 0px #97a6e6; adiciona uma sombra ao texto com um leve tom azul claro, criando um efeito visual que melhora a legibilidade e o design.

```
33. .azul {
        margin: 10px 0;
        text-align: center;
        padding: 30px;
        border: none;
        border-radius: 50px;
        background-color: #cbe3e7;
```

```
    box-sizing: border-box;
    width: 50%;
    color: #004aad;
    font-size: 1.875em;
}
```

Aqui temos as definições para .azul, que estiliza um botão ou uma caixa de destaque na página:

- margin: 10px 0; adiciona um espaçamento externo de 10px acima e abaixo, mantendo as laterais sem margens.
- text-align: center; centraliza o texto dentro do elemento.
- padding: 30px; adiciona um espaçamento interno de 30px, garantindo que o conteúdo tenha um bom espaçamento das bordas.
- border: none; remove qualquer borda visível do elemento.
- border-radius: 50px; arredonda as bordas, criando um efeito visual mais suave e moderno.
- background-color: #cbe3e7; define um tom azul claro como fundo do elemento.
- box-sizing: border-box; garante que a largura e a altura incluam o padding, evitando distorções no layout.
- width: 50%; faz com que o elemento ocupe 50% da largura disponível.
- color: #004aad; define a cor do texto como azul escuro, garantindo contraste e legibilidade.
- font-size: 1.875em; aumenta o tamanho do texto para melhorar a visibilidade e tornar o elemento mais chamativo.

34.
```
.azul::placeholder { /* Permite alterar elementos do placeholder*/
    color: #004aad;
    opacity: 1;
}
```

Aqui temos as definições para .azul::placeholder, que estiliza o texto de espaço reservado dentro de um campo de entrada (input ou textarea) com a classe .azul:

- color: #004aad; define a cor do placeholder como azul escuro, garantindo que ele fique visível e harmonizado com o design do campo.
- opacity: 1; força o placeholder a manter opacidade total, garantindo que ele não fique semitransparente (o valor padrão costuma ser menor em alguns navegadores).

35.
```
textarea.azul {
    resize: vertical;
    width: 50%;
```

```
    padding: 50px;
}
```

Aqui temos as definições para textarea.azul, que estiliza uma área de texto (<textarea>) com a classe .azul:

- resize: vertical; permite que o usuário redimensione a área de texto apenas verticalmente, impedindo ajustes horizontais que poderiam comprometer o layout.
- width: 50%; define que a largura do campo ocupará 50% do espaço disponível, garantindo que ele não fique muito estreito nem muito largo.
- padding: 50px; adiciona um espaçamento interno generoso, garantindo que o texto dentro do campo tenha espaço adequado e não fique muito próximo das bordas.

Esse código cria uma área de texto bem estruturada, confortável para digitação e adaptável ao conteúdo inserido pelo usuário.

36.
```
.botao {
    margin: 1% 5%;
    width: 49.5%;
    display: flex;
    justify-content: space-between;
}
```

Aqui temos as definições para .botao, que estiliza um contêiner de botões ou elementos interativos na página:

- margin: 1% 5%; adiciona um espaçamento externo de 1% no topo e na base e 5% nas laterais, garantindo um bom distanciamento dos outros elementos.
- width: 49.5%; define a largura do contêiner como 49.5%, permitindo que ele ocupe metade do espaço disponível, possivelmente ao lado de outro elemento semelhante.
- display: flex; ativa o Flexbox, permitindo um alinhamento mais dinâmico dos elementos internos.
- justify-content: space-between; distribui os elementos dentro do contêiner com espaço uniforme entre eles, garantindo que fiquem organizados e bem posicionados.

37.
```
#enviar {
    border: none;
    border-radius: 40px;
    padding: 25px;
```

```
    width: 40%;
    font-size: 1.875em;
    background-color: #004aad;
    color: white;
}
```

Aqui temos as definições para #enviar, que estiliza um botão de envio na página:

- border: none; remove qualquer borda visível do botão, criando um design mais limpo.
- border-radius: 40px; arredonda as bordas do botão, tornando-o mais suave e moderno.
- padding: 25px; adiciona um espaçamento interno para tornar o botão maior e mais confortável de clicar.
- width: 40%; define que o botão ocupará 40% da largura disponível, garantindo um tamanho equilibrado.
- font-size: 1.875em; aumenta o tamanho do texto dentro do botão, tornando-o mais legível e chamativo.
- background-color: #004aad; define um fundo azul escuro, destacando o botão na página.
- color: white; define a cor do texto como branco, garantindo contraste e facilitando a leitura.

```
38. #cancelar {
    border: none;
    border-radius: 40px;
    padding: 25px;
    width: 40%;
    font-size: 1.875em;
    background-color: #c2c2c2;
    color: white;
}
```

Aqui temos as definições para #cancelar, que estiliza um botão de cancelamento na página:

- border: none; remove qualquer borda visível, mantendo um design mais limpo.
- border-radius: 40px; arredonda as bordas do botão, criando um visual suave e moderno.
- padding: 25px; adiciona espaçamento interno, tornando o botão maior e mais confortável para interação.
- width: 40%; define que o botão ocupará 40% da largura disponível, garantindo um tamanho equilibrado.

- font-size: 1.875em; aumenta o tamanho do texto dentro do botão, tornando-o mais legível e chamativo.
- background-color: #c2c2c2; define um fundo cinza claro, diferenciando-o de outros botões e indicando uma ação secundária.
- color: white; define a cor do texto como branco, garantindo contraste e facilitando a leitura.

39. footer {

```
display: flex;

flex-direction: column;

justify-content: center;

align-items: center;

width: 100%;

height: 10%;

font-size: 1.475em;

margin-top: 0.5%;

font-family: "Anaheim", serif;

font-optical-sizing: auto;

font-style: normal;
```

}

Aqui temos as definições para footer, que estiliza o rodapé da página:

- display: flex; ativa o Flexbox, permitindo um melhor controle sobre o posicionamento dos elementos dentro do rodapé.
- flex-direction: column; organiza os elementos verticalmente, empilhando-os um abaixo do outro.
- justify-content: center; alinha os elementos verticalmente ao centro, garantindo que fiquem bem distribuídos dentro do rodapé.
- align-items: center; centraliza os elementos horizontalmente, mantendo um alinhamento uniforme.
- width: 100%; faz com que o rodapé ocupe toda a largura da tela.
- height: 10%; define a altura do rodapé como 10% do tamanho do elemento pai, garantindo que ele não fique muito grande ou pequeno.
- font-size: 1.475em; ajusta o tamanho do texto para torná-lo legível sem ocupar muito espaço.
- margin-top: 0.5%; adiciona um pequeno espaçamento superior, garantindo que o rodapé não fique colado aos elementos anteriores.
- font-family: Anaheim, serif; define a fonte do texto, garantindo um estilo visual consistente com o restante do site.
- font-optical-sizing: auto; ajusta automaticamente o tamanho óptico da fonte para melhor legibilidade.
- font-style: normal; mantém o estilo do texto normal, sem itálico ou variações.

Esse código garante que o rodapé fique bem estruturado, centralizado e com um design harmonioso no layout da página.

40. #pgUp {

```
display: flex;
align-items: flex-end;
position: fixed;
margin: 3%;
bottom: 0;
right: 0;
background-color: transparent;
border: 0;
width: 5%;
```

}

Aqui temos a estilização do botão que volta para o topo da página:

- display: flex; → Torna o botão um container flexível — isso permite alinhar conteúdo dentro dele (tipo uma imagem, ícone, texto etc).
- align-items: flex-end; → Dentro do botão, alinha o conteúdo na parte de baixo (inferior). No caso da nossa imagem ela vai ficar grudada no fundo do botão.
- position: fixed; → Faz com que o botão fique fixo na tela, mesmo quando você rola a página, deixando ele sempre visível como um botão flutuante.
- margin: 3%; → Coloca uma margem externa de 3% em todos os lados (mas principalmente afeta a posição de bottom e right, como veremos a seguir).
- bottom: 0; → Cola o botão no rodapé da janela (parte de baixo).
- right: 0; → Cola o botão na borda direita da tela.
- background-color: transparent; → Deixa o fundo transparente — ou seja, só o conteúdo dentro (como uma imagem de seta) vai aparecer, sem nenhuma caixinha.
- border: 0; → Remove qualquer borda visível do botão (estética mais limpa).
- width: 5%; → O botão vai ocupar 5% da largura da janela. Mantém o botão responsivo. (menor em telas pequenas, maior em telas grandes).

41. #imgPgUp {

```
width: 100%;
```

}

Aqui temos a estilização da imagem atribuida ao botão de page up:

- #imgPgUp → seleciona um elemento com o ID imgPgUp.
- width: 100%; → define que a largura do elemento será 100% da largura do elemento pai.

CSS Responsivo

O CSS responsivo permite que um site se adapte automaticamente a diferentes tamanhos de tela, garantindo uma experiência agradável para usuários de celulares, tablets e desktops. Para isso, utilizamos as Media Queries, um recurso do CSS que aplica estilos específicos com base nas dimensões do dispositivo, como largura da tela, altura, resolução e orientação. As Media Queries são regras do CSS que permitem definir estilos condicionais, ou seja, aplicar diferentes configurações de layout dependendo do tamanho da tela do usuário. Com elas, podemos ajustar fontes, reorganizar elementos, modificar cores e até ocultar partes do site para melhorar a usabilidade. A estrutura básica de uma Media Query é: @media (condição) { seletor { propriedade: valor; } }. Uma das condições mais comuns é o tamanho da largura da tela, usando min-width e max-width, como no exemplo: @media (min-width: 768px) and (max-width: 1024px) { body { background-color: lightblue; } }. Nesse caso, o fundo do site muda para azul claro quando a largura da tela está entre 768px e 1024px. Com essas técnicas, o site se torna mais acessível, bonito e funcional em qualquer dispositivo. O nosso site será composto de 5 Media Queries, a seguir vou te mostrar o Media Query pronto e explicar as suas funções.

1.
```
@media (min-width: 1502px) and (max-width: 1701px){
    .transformandoTitle{
        font-size: 2.65em;
        margin: 1.5% 0%;
    }
    .transformando{
     height: 80%;
    }
    .transformando > img {
        height: 42%;
    }
}
```

Aqui temos um bloco de Media Query, que ajusta o layout da página para telas com largura entre 1502px e 1701px:

- .transformandoTitle
    - font-size: 2.65em; reduz o tamanho do título para se adequar melhor ao espaço disponível.
    - margin: 1.5% 0%; mantém o espaçamento externo do título sem alterar a largura.
- .transformando

    - height: 80%; ajusta a altura da seção .transformando, garantindo que ela não ocupe muito espaço em telas menores dentro desse intervalo de largura.
- .transformando > img

    - height: 42%; reduz a altura da imagem dentro da seção .transformando, garantindo um melhor encaixe no layout ajustado.

Esse código garante que o design da página se adapte corretamente a telas médias, mantendo um layout equilibrado e bem organizado.

2. @media (min-width: 1201px) and (max-width: 1501px) {
   .transformandoTitle{
   font-size: 2.3em;
   margin: 1.5% 0%;
   }
   .protesto{
   height: 450px;
   }
   .transformando{
   height: 85%;
   }
   .transformando > img {
   height: 38%;
   }
   .bloco_esq1 , .bloco_esq2 {
   font-size: 0.8em;
   }
   .bloco_pri2 {
   height: 40em;
   }
   .estatisticas{
   padding: 50px
   }
   li{
   font-size: 1.2em;
   }
   #pgUp {
   width: 7%;
   }
   }

Aqui temos um bloco de Media Query, que ajusta o layout da página para telas com largura entre 1201px e 1501px:

- .transformandoTitle

    - font-size: 2.3em; reduz o tamanho do título para se adequar melhor ao espaço disponível.
    - margin: 1.5% 0%; mantém um espaçamento externo adequado.

- .protesto
    - height: 450px; diminui a altura da seção, ajustando-a para telas menores.
- .transformando
    - height: 85%; ajusta a altura da seção para garantir um encaixe adequado no layout.
- .transformando > img
    - height: 38%; reduz a altura da imagem dentro da seção .transformando, garantindo um equilíbrio visual.
- .bloco_esq1 , .bloco_esq2
    - font-size: 0.8em; reduz o tamanho da fonte dentro desses blocos para melhorar a legibilidade e manter o layout organizado.
- .bloco_pri2
    - height: 40em; ajusta a altura do bloco principal para garantir que os conteúdos fiquem bem distribuídos.
- .estatisticas
    - padding: 50px; reduz o espaçamento interno da seção para manter um layout mais compacto.
- li
    - font-size: 1.2em; diminui o tamanho da fonte dentro das listas, garantindo um melhor aproveitamento do espaço.
- #pgUp
    - width: 7%: define que a largura do elemento será 7% da largura do elemento pai.

Esse código permite que o site se adapte bem a telas médias, garantindo que os elementos fiquem organizados e proporcionais ao tamanho da tela.

3.
```
@media (min-width: 1025px) and (max-width: 1200px){
    .logo{
        width: 10%;
    }
    .transformando{
        height: 75%;
    }
    .transformandoTitle{
        font-size: 2em;
        margin: 1.5% 0%;
    }
    .transformando > p {
        font-size: 1.675em;
        width: 80%;
    }
```

```
    .transformando > img {
        height: 45%;
    }
    .protesto{
        height: 350px;
    }
    .bloco_esq1 , .bloco_esq2 {
        font-size: 0.8em;
    }
    .bloco_pri2 {
        height: 45em;
    }
    .estatisticas{
        padding: 50px
    }
    li{
        font-size: 1em;
    }
    form{
        padding: 3%;
        margin-bottom: 20px;
    }
    .azul{
       padding: 20px;
       font-size: 1.775em;
    }
    #enviar {
        padding: 25px;
        width: 30%;
        font-size: 1.675em;
    }

    #cancelar {
        padding: 10px;
        width: 30%;
        font-size: 1.675em;
    }
    #pgUp {
        width: 8%;
    }

}
```

Aqui temos um Media Query que ajusta o layout da página para telas com largura entre 1025px e 1200px:

- .logo
    - width: 10%; aumenta o tamanho da logo para melhor visibilidade em telas menores.

- .transformando
    - height: 75%; reduz a altura da seção para manter um layout mais compacto.
- .transformandoTitle
    - font-size: 2em; diminui o tamanho do título para que ele se ajuste ao espaço disponível.
    - margin: 1.5% 0%; mantém o espaçamento externo adequado.
- .transformando > p
    - font-size: 1.675em; reduz o tamanho do texto para melhor legibilidade.
    - width: 80%; mantém um alinhamento adequado do texto dentro da seção.
- .transformando > img
    - height: 45%; ajusta a altura da imagem para um melhor encaixe visual.
- .protesto
    - height: 350px; reduz a altura da seção para que ocupe menos espaço na tela.
- .bloco_esq1 , .bloco_esq2
    - font-size: 0.8em; diminui o tamanho da fonte para manter um bom aproveitamento do espaço.
- .bloco_pri2
    - height: 45em; ajusta a altura do bloco principal para manter o equilíbrio no layout.
- .estatisticas
    - padding: 50px; reduz o espaçamento interno da seção para um ajuste melhor na tela.
- li
    - font-size: 1em; reduz o tamanho da fonte dos itens da lista para um melhor aproveitamento do espaço.
- form
    - padding: 3%; diminui o espaçamento interno do formulário para evitar ocupar muito espaço.
    - margin-bottom: 20px; adiciona um espaçamento inferior para melhor separação.
- .azul
    - padding: 20px; ajusta o espaçamento interno dos campos de entrada para uma melhor aparência.
    - font-size: 1.775em; reduz o tamanho da fonte nos campos de entrada para se adaptar melhor ao layout.
- #enviar

    - padding: 25px; ajusta o espaçamento do botão para um melhor clique.
    - width: 30%; reduz a largura do botão para manter um layout equilibrado.
    - font-size: 1.675em; ajusta o tamanho do texto do botão.
- #cancelar

    - padding: 10px; reduz o espaçamento interno para um melhor encaixe no layout.
    - width: 30%; mantém a largura semelhante ao botão de envio.
    - font-size: 1.675em; mantém a proporção do texto dentro do botão.
- #pgUp {
    - width: 8%: define que a largura do elemento será 8% da largura do elemento pai.

Esse código garante que o site se ajuste corretamente para telas menores dentro desse intervalo de largura, mantendo a usabilidade e a estética.

4. @media (min-width: 769px) and (max-width: 1024px){

```
.logo{
    width: 10%;
}
h2{
    font-size: 3.125em;
    text-shadow: 2px 2px 0px rgba(0, 74, 173, 0.7);
}
.transformando{
    height: 75%;
}
.transformandoTitle{
    font-size: 1.5em;
}
.transformando > p {
    font-size: 1.3em;
    width: 80%;
}
.transformando > img {
    height: 40%;
}
.protesto{
    height: 350px;
}
.bloco_esq1 , .bloco_esq2 {
    font-size: 0.7em;
}
.bloco_pri1 {
    margin-bottom: 1em;
}

.bloco_pri2 {
    height: 45em;
}
```

```
    .estatisticas{
        padding: 40px
    }
    li{
        font-size: 1em;
    }
    form{
        padding: 3%;
        margin-bottom: 20px;
    }
    .fale{
        font-size: 3em;
        text-shadow: 3px 3px 0px #97a6e6;
    }
    .azul{
       padding: 10px;
       width: 60%;
       font-size: 1.775em;
    }
    textarea.azul {
        width: 60%;
        padding: 50px;
    }
    #enviar {
        padding: 25px;
        width: 30%;
        font-size: 1em;
    }
    #cancelar {
        padding: 10px;
        width: 30%;
        font-size: 1em;
    }
    #pgUp {
        width: 8.5%;
    }
}
```

Aqui temos um Media Query que ajusta o layout para telas entre 769px e 1024px:

- .logo
    - width: 10%; ajusta o tamanho da logo para manter a proporção adequada.
- h2
    - font-size: 3.125em; aumenta o tamanho dos títulos de segundo nível para maior destaque.
    - text-shadow: 2px 2px 0px rgba(0, 74, 173, 0.7); adiciona uma sombra para melhorar a legibilidade.
- .transformando

    - height: 75%; ajusta a altura da seção.

- .transformandoTitle
    - font-size: 1.5em; reduz o tamanho do título para se adequar ao layout menor.
- .transformando > p
    - font-size: 1.3em; ajusta o tamanho do texto para facilitar a leitura.
    - width: 80%; mantém um alinhamento equilibrado.
- .transformando > img
    - height: 40%; ajusta a altura da imagem.
- .protesto
    - height: 350px; reduz a altura para um melhor ajuste na tela.
- .bloco_esq1 , .bloco_esq2
    - font-size: 0.7em; reduz o tamanho do texto dentro desses blocos.
- .bloco_pri1
    - margin-bottom: 1em; adiciona um espaçamento inferior para melhorar o fluxo visual.
- .bloco_pri2
    - height: 45em; ajusta a altura do bloco principal.
- .estatisticas
    - padding: 40px; reduz o espaçamento interno para otimizar o layout.
- li
    - font-size: 1em; ajusta o tamanho da fonte para manter a leitura confortável.
- form
    - padding: 3%; ajusta o espaçamento interno do formulário.
    - margin-bottom: 20px; adiciona um espaçamento inferior para separar melhor os elementos.
- .fale
    - font-size: 3em; reduz o tamanho do título para manter uma hierarquia visual adequada.
    - text-shadow: 3px 3px 0px #97a6e6; adiciona uma sombra para melhor legibilidade.
- .azul
    - padding: 10px; reduz o espaçamento interno para melhor ajuste.
    - width: 60%; aumenta a largura dos campos de entrada para facilitar o preenchimento.
    - font-size: 1.775em; ajusta o tamanho do texto.
- textarea.azul
    - width: 60%; mantém a proporção do campo de texto.
    - padding: 50px; garante um espaçamento interno adequado.
- #enviar
    - padding: 25px; ajusta o espaçamento interno do botão.
    - width: 30%; mantém uma largura proporcional.
    - font-size: 1em; ajusta o tamanho do texto para manter a usabilidade.
- #cancelar
    - padding: 10px; reduz o espaçamento interno.
    - width: 30%; mantém a consistência visual com o botão de envio.
    - font-size: 1em; ajusta o tamanho da fonte.
- #pgUp {
    - width: 8.5%: define que a largura do elemento será 8.5% da largura do elemento pai.

Esse Media Query adapta o design para telas médias, mantendo a responsividade e garantindo que os elementos permaneçam bem organizados.

5.
```
@media (min-width: 481px) and (max-width: 768px){
    header{
        height: 15%;
    }
    .logo{
        width: 12%;
    }
    .transformando{
        height: 80%;
    }
    .transformandoTitle{
        font-size: 1.8em;
        width: 85%;
        text-align: center;
    }
    .transformando > p {
        font-size: 1.3em;
        width: 80%;
        text-align: justify;
    }
    .transformando > img {
        height: 28%;
    }
    h2{
        font-size: 3.5em;
        text-shadow: 2px 2px 0px rgba(0, 74, 173, 0.7);
    }
    .protesto{
        height: 235px;
    }
    .bloco_pri1 {
        margin-bottom: 1em;
        height: 45em;
    }
    .bloco_pri2 {
        height: 37em;
        margin-bottom: 1em;
    }
    .bloco_pri1, .bloco_pri2{
        flex-direction: column;
    }
    .bloco_esq1 , .bloco_esq2 {
        font-size: 0.57em;
       margin: 50px 20px 40px 0px;
        width: 100%;
        align-items: center;
    }
    .bloco_esq2 > p, .bloco_esq1 > p {
```

```
    font-size: 2.1875em;
    width: 80%;
    margin: 40px 0px 20px 0px;
    text-align: justify;
}
.bloco_dir{
    width: 100%;
    flex-direction: row;
    justify-content: space-around;
}
.bloco_dir > img {
   width: 35%;
}
.estatisticas{
    padding: 50px 10px;
}
.estatisticas-title{
    font-size: 2.5em;
}
li{
    font-size: 0.7em;
}
.slide{
    padding: 0px;
}
form{
    padding: 10%;
    margin-bottom: 20px;
}
.fale{
    font-size: 2.5em;
    text-shadow: 3px 3px 0px #97a6e6;
}
.azul{
   padding: 5px;
   width: 85%;
   font-size: 1.275em;
}
textarea.azul {
    width: 85%;
    padding: 10px;
    border-radius: 20px;
}
.botao{
    display: flex;
    flex-direction: column;
    align-items: center;
   gap: 10px;
}
#enviar {
    padding: 10px;
```

```
        width: 170%;
        font-size: 1em;
    }
    #cancelar {
        padding: 10px;
        width: 170%;
        font-size: 1em;
    }
    #pgUp {
        width: 10%;
    }

}
```

Este Media Query adapta o layout para telas entre 481px e 768px, otimizando o design para dispositivos menores, como tablets e celulares grandes.

- header { height: 15%; } → Reduz a altura do cabeçalho para economizar espaço.
- .logo { width: 12%; } → Ajusta o tamanho da logo para manter a proporção adequada.
- .transformando { height: 80%; } → Mantém uma altura responsiva.
- .transformandoTitle
  - font-size: 1.8em; → Reduz o tamanho do título para caber melhor.
  - width: 85%; text-align: center; → Ajusta a largura e centraliza o texto.
- .transformando > p
  - font-size: 1.3em; width: 80%; text-align: justify; → Ajusta a fonte e alinha o texto.
- .transformando > img { height: 28%; } → Reduz a altura da imagem.
- h2
  - font-size: 3.5em; → Aumenta o título para manter a hierarquia visual.
  - text-shadow: 2px 2px 0px rgba(0, 74, 173, 0.7); → Mantém a sombra para contraste.
- .protesto { height: 235px; } → Reduz a altura da seção para se encaixar melhor.
- .bloco_pri1, .bloco_pri2
  - flex-direction: column; → Muda de linha para coluna, melhorando a visualização em telas menores.
- .bloco_pri1
  - margin-bottom: 1em; height: 45em; → Ajusta altura e espaçamento.
- .bloco_pri2
  - height: 37em; margin-bottom: 1em; → Ajusta para manter harmonia no layout.
- .bloco_esq1, .bloco_esq2
  - font-size: 0.57em; → Diminui um pouco o tamanho do texto.
  - margin: 50px 20px 40px 0px; → Ajusta espaçamentos.
  - width: 100%; align-items: center; → Expande para ocupar toda a largura.
- .bloco_esq2 > p, .bloco_esq1 > p
  - font-size: 2.1875em; → Mantém o destaque do parágrafo.
  - width: 80%; margin: 40px 0px 20px 0px; → Ajusta largura e espaçamentos.

    - text-align: justify; → Melhora a legibilidade.
- .bloco_dir
    - width: 100%; flex-direction: row; justify-content: space-around; → Os elementos ficam lado a lado para melhor aproveitamento do espaço.
- .bloco_dir > img { width: 35%; } → Ajusta o tamanho das imagens.
- .estatisticas { padding: 50px 10px; } → Ajusta o espaçamento interno.
- .estatisticas-title { font-size: 2.5em; } → Reduz o tamanho do título.
- li { font-size: 0.7em; } → Reduz a fonte para caber melhor.
- .slide { padding: 0px; } → Remove espaçamentos extras.
- form { padding: 10%; margin-bottom: 20px; } → Ajusta espaçamento do formulário.
- .fale
    - font-size: 2.5em; → Reduz tamanho do título para encaixar melhor.
    - text-shadow: 3px 3px 0px #97a6e6; → Mantém destaque no texto.
- .azul
    - padding: 5px; width: 85%; font-size: 1.275em; → Ajusta tamanho e largura dos campos.
- textarea.azul
    - width: 85%; padding: 10px; border-radius: 20px; → Ajusta espaçamento e arredondamento.
- .botao
    - display: flex; flex-direction: column; align-items: center; gap: 10px;
        - Empilha os botões e adiciona um espaço entre eles.
- #enviar, #cancelar
    - padding: 10px; width: 170%; font-size: 1em;
        - Aumenta a largura dos botões para facilitar o clique.
- #pgUp {
    - width: 10%: define que a largura do elemento será 10% da largura do elemento pai.

Este Media Query reorganiza os elementos para garantir que o site continue intuitivo, responsivo e acessível em telas menores.

JavaScript

Agora vamos adicionar funcionalidades ao nosso código utilizando JavaScript, uma linguagem de programação essencial para tornar páginas da web interativas e dinâmicas. Diferente do HTML, que estrutura o conteúdo, e do CSS, que estiliza a aparência, o JavaScript permite adicionar comportamentos, como animações, validação de formulários e interações do usuário. Sua sintaxe segue a estrutura variável = valor; e o uso de funções, como function nomeDaFuncao() { // código }, para executar ações específicas. Por exemplo, let mensagem = "Olá!"; console.log(mensagem); declara uma variável e exibe seu valor no console. Vamos começar aplicando o JavaScript no Slider do nosso projeto.

1. Slider:

   document.addEventListener("DOMContentLoaded", function () {

```
    let slides = document.querySelectorAll(".slide");
    let dots = document.querySelectorAll(".dot");
    let currentSlideIndex = 0;
    function showSlide(index) {
        slides.forEach((slide, i) => {
            slide.style.display = i === index ? "block" : "none";
        });
        dots.forEach((dot, i) => {
            dot.classList.toggle("active", i === index);
        });
        currentSlideIndex = index;
    }
    dots.forEach((dot, index) => {
        dot.addEventListener("click", function () {
            showSlide(index);
        });
    });


    showSlide(currentSlideIndex);
});
```

Esse código cria um slider manual, onde o usuário pode navegar pelos slides clicando nos pontos de indicação (dots). A seguir as funções de cada elemento do código:

- O código é executado somente após o carregamento do conteúdo da página, graças ao document.addEventListener("DOMContentLoaded", function () {...});.
- let slides = document.querySelectorAll(".slide"); → Variável que captura todos os elementos com a classe .slide.
- let dots = document.querySelectorAll(".dot"); → Variável que captura todos os elementos com a classe .dot.
- let currentSlideIndex = 0; → Define a variável que rastreia o slide atual.
- showSlide(index) percorre todos os slides e:
    - Exibe o slide correspondente (slide.style.display = "block") e oculta os outros (slide.style.display = "none").

    - Atualiza os "dots" (indicadores), ativando apenas o ponto correspondente ao slide atual (dot.classList.toggle("active", i === index);).

    - Atualiza currentSlideIndex com o índice do slide atual.

- Percorre todos os dots com dots.forEach((dot, index) => {...}) e adiciona um evento de clique em cada um.

- Quando um dot é clicado, ele chama showSlide(index) para exibir o slide correspondente.
- showSlide(currentSlideIndex); garante que, ao carregar a página, o primeiro slide seja exibido corretamente.

2. Verificação se os campos foram preenchidos corretamente:

```
enviar.onclick = function enviarFeedback(){
    var campo = document.getElementsByClassName("azul");
    var enviar = document.getElementById("enviar");
    var preenchido = true;

    for (var i = 0; i < campo.length; i++) {
        if (campo[i].value.trim() === "") {
            preenchido = false;
            break;
        }
    }

    if (!preenchido) {
        alert("Preencha todos os campos corretamente 🥺");
        campo.preventDefault();
    }

    else {
        alert("Feedback enviado com sucesso! 😎");
    }
};
```

Esse código verifica se os campos do formulário para feedbacks foram preenchidos corretamente.

- enviar.onclick = function enviarFeedback() → Define o que deve acontecer quando o botão com ID enviar for clicado, ele atribui a função enviarFeedback ao evento de clique no botão.
- var campo = document.getElementsByClassName("azul") → Pega todos os elementos do HTML que tem a classe azul e retorna eles como um conjunto de elementos (tipo um array ou uma lista).
- var enviar = document.getElementById("enviar") → Pega os elementos com ID enviar e retorna para a função enviarFeedback.
- var preenchido = true → Inicia a variável preenchido para verificar se todos os campos da classe azul foram preenchidos, e retorna True se sim.
- for (var i = 0; i < campo.length; i++) { → Percorre todos os campos com a classe azul e caso encontre algum campo vazio, ele é definido como false, e para o loop de verificação do for.
- campo[i].value → pega o conteúdo digitado.

- .trim() → remove espaços em branco no início e no fim (para evitar que um campo com só espaços conte como preenchido).
- if (!preenchido) { → Recebe a verificação do for e se os campos constarem diferente de preenchido dispara uma alerta para o usuário.
- alert("Preencha todos os campos corretamente 🥺"); → Esse é o alerta disparado para o usuário em caso de erro no preenchimento.
- campo.preventDefault(); → Não deixa o formulário ser enviado com campos vazios.
- else { → Se todos os campos estiverem preenchidos, dispara um alerta de enviado com sucesso.
- alert("Feedback enviado com sucesso! 😎"); →Esse é o alerta disparado para o usuário em caso de envio com sucesso

3. Botão de voltar ao topo da tela

```
document.getElementById("pgUp").onclick = function subirTela() {
    window.scrollTo({ top: 0, behavior: "smooth" });
};
```

Esse código define um botão para voltar ao topo da tela:

- document.getElementById("pgUp").onclick = function subirTela() { → Pega os elementos com o ID pgUP e define que quando clicado nesse elemento será executada a função subir tela.
- window.scrollTo({ top: 0, behavior: "smooth" }); → Função para rolar a tela até uma parte específica, nesse caso, as especificações da função (top: 0, behavior: "smooth") fazem com que a tela seja rolada até o topo da tela e garantem que a rolagem seja feita de forma suave com uma animação, ao invés de pular direto para o topo.