UNIVERSIDADE PRESBITERIANA MACKENZIE
Faculdade de Computação e Informática

Tutorial do projeto de Web mobile - Pichu

1. GRUPO

Nome: Gean Ferreira da Silva
RA: 10438717

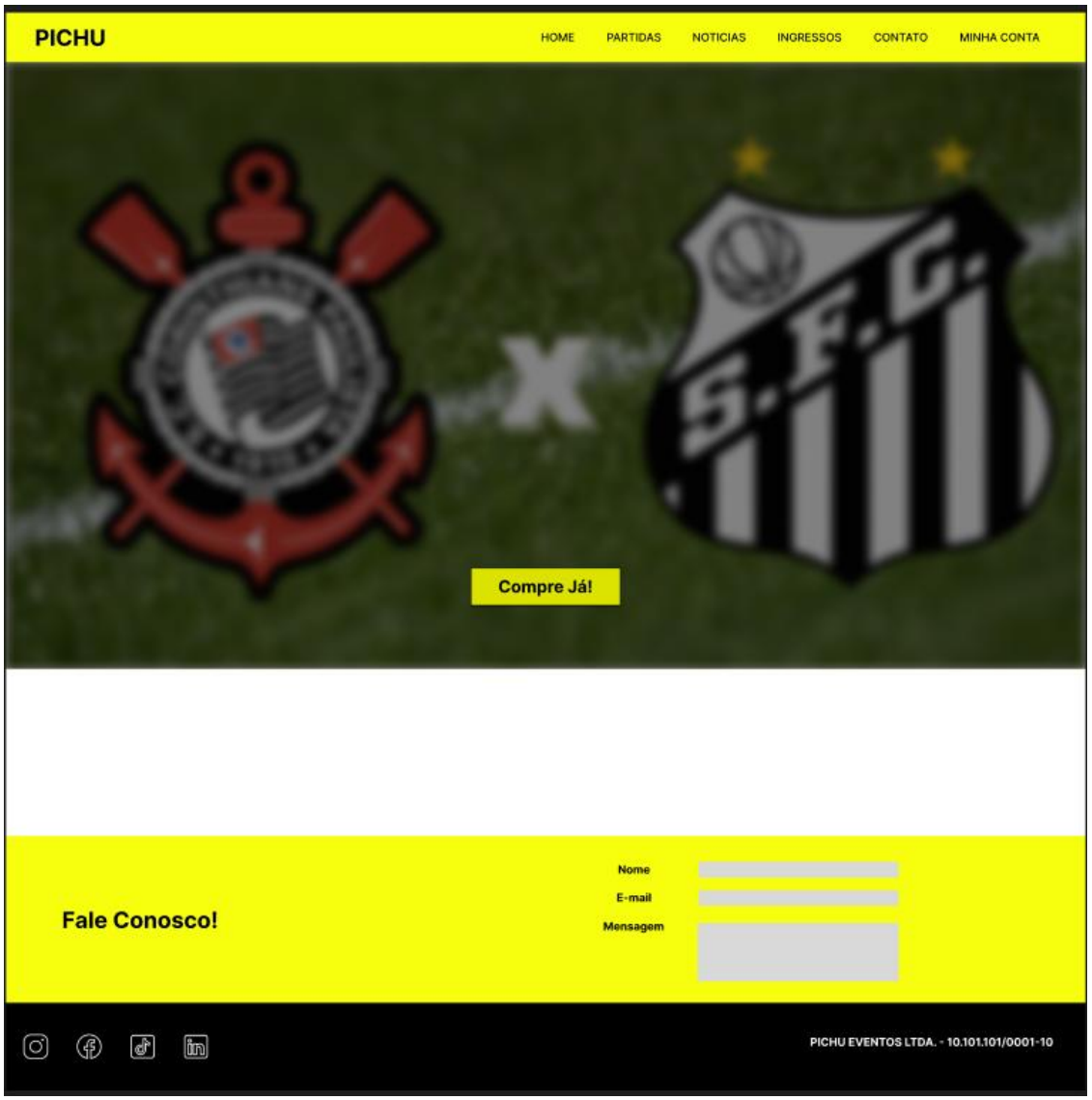
Nome: Moisés Santos
RA: 10419955

Nome: Pedro Moreno Campos
RA:10390807

Professora: Paula Leite

Introdução

Essa ideia nasceu do amor que todos no grupo têm pelo futebol, com cada um torcendo por times diferentes. A partir dessa paixão em comum, surgiu a inspiração para criar um site dedicado à venda de ingressos para jogos de futebol.



O cabeçalho da página contará com o logo da empresa e o menu de navegação horizontal, facilitando o acesso às principais seções da landing page. A primeira seção apresentará um carrossel com imagens dos próximos jogos, acompanhado de um botão de acesso direto à página de pagamento.

Logo abaixo, haverá uma lista das próximas partidas e as principais notícias relacionadas. A página também incluirá um formulário de contato, permitindo que os visitantes entrem em comunicação facilmente. O rodapé contará com links para as redes sociais da empresa, proporcionando uma maneira prática de interação.

Primeiros Passos

O primeiro passo para a criação deste site é a criação de um arquivo HTML que chamaremos de "index.html". O padrão atual é o HTML5, e é o que utilizaremos. O arquivo HTML é dividido em tags, com algumas essenciais para a estrutura básica de um site.

Começamos pelas tags de configuração, como:

- *<DOCTYPE html>* - Define o tipo do documento como HTML5;
- *<html lang="pt-br">* - Define a linguagem do site como português do Brasil, o que ajuda os mecanismos de busca e leitores de tela;
- *<head>* - Responsável por informações importantes para o navegador, como título da página, codificação de caracteres e links para estilos e scripts.

Dentro da tag <head>, incluiremos elementos essenciais como:

- *<meta charset="UTF-8">* -Define a codificação de caracteres como UTF-8, garantindo suporte a acentos e caracteres especiais;
- *<meta name="viewport" content="width=device-width, initial-scale=1.0">* - Garante que o site seja responsivo, ajustando a largura ao dispositivo do usuário;
- *<title> Pichu - Ingressos para jogos </title>* - Define o título que aparece na aba do navegador.

```
<!DOCTYPE html>
<html lang="pt-br">

<head>
   <meta charset="UTF-8">
      <meta   name="viewport"   content="width=device-width,   initial-
scale=1.0">
   <title> Pichu - Ingressos para jogos </title>
   <link rel="stylesheet" href="style.css">
</head>
```

Após o <head>, temos a tag <body>, que contém o conteúdo visível da página, como textos, imagens, links e botões que serão abordados mais à frente.

A partir disso, temos uma estrutura básica para a criação de um site. Nos próximos passos, detalharemos como cada seção do site foi pensada e desenvolvida, abordando conceitos de HTML e CSS para estilização, flexbox, e media queries para garantir a responsividade em diferentes dispositivos, como smartphones e tablets.

Criando o cabeçalho (header)

Com a estrutura básica do site pronta, vamos começar a desenvolver o cabeçalho da página, onde colocaremos o nome do site e um menu de navegação.

HTML do <header>

O código abaixo cria um cabeçalho com o título "Pichu" e um menu de navegação com cinco links:

```
<header>
    <h1>Pichu</h1>
    <nav>
        <ul class="menu">
            <li><a href="#">Home</a></li>
            <li><a href="#">Partidas</a></li>
            <li><a href="#">Comprar Ingressos</a></li>
            <li><a href="#">Contato</a></li>
            <li><a href="#">Minha Conta</a></li>
        </ul>
    </nav>
</header>
```

- A tag **<header>** representa o cabeçalho da página.
- O **<h1>** define o título principal do site.
- A tag **<nav>** é usada para agrupar os links de navegação.
- A lista **<ul>** contém os itens de menu, que são definidos dentro das tags **<li>**.

Estilizando o cabeçalho com CSS

Agora vamos aplicar estilos para organizar o layout do cabeçalho e estilizar o menu de navegação:

```
header {
    display: flex;
    justify-content: space-between;
    align-items: center;
    background-color: #f7ff0b;
    padding: 10px 20px;
}

nav ul {
    list-style: none;
    margin: 0;
```

```
    padding: 0;
    display: flex;
}

nav ul li {
    margin: 0 15px;
}

nav ul li a {
    color: #000000;
    text-decoration: none;
    font-size: 18px;
    padding: 10px 15px;
}

nav ul li a:hover {
    color: #ffffff;
    background-color: #000000;
    border-radius: 5px;
}
```

- **`display: flex`** no header organiza os elementos (o título e o menu) lado a lado.
- **`justify-content: space-between`** garante que o título fique à esquerda e o menu à direita.
- **`align-items: center`** alinha verticalmente os itens no centro do cabeçalho.
- **`background-color`** define a cor de fundo do cabeçalho.
- A lista de navegação é transformada em um menu horizontal com **`display: flex.`**
- Os links ganham espaçamento interno (padding) e efeitos visuais ao passar o mouse com **`:hover`**.

Criando um Slider de Imagens (Carrossel)

Nesta etapa, vamos adicionar um carrossel de imagens (também chamado de *slider*) ao nosso site. Essa funcionalidade permite exibir várias imagens em sequência, melhorando a apresentação visual e destacando conteúdos importantes, como partidas ou eventos.

Estrutura HTML do Slider

A seguir, temos o código HTML responsável por exibir quatro imagens diferentes com botões de navegação manual e automática:

```
<section class="slider">
    <div class="slides">
        <input type="radio" name="radio-btn" id="radio1">
        <input type="radio" name="radio-btn" id="radio2">
        <input type="radio" name="radio-btn" id="radio3">
        <input type="radio" name="radio-btn" id="radio4">
```

```
        <img class="slide first" src="img/carrossel1.png" alt="Imagem de um
jogo de futebol Santos x Corinthians">
        <img class="slide" src="img/carrossel2.jpeg" alt="Imagem de um jogo
de futebol São Paulo x Palmeiras">
        <img class="slide" src="img/carrossel3.png" alt="Imagem de um jogo
de futebol Flamengo x Fluminense">
        <img class="slide" src="img/carrossel4.jpg_large" alt="Imagem de um
jogo de futebol Grêmio x Internacional">

        <div class="navigation-auto">
            <div class="auto-btn1"></div>
            <div class="auto-btn2"></div>
            <div class="auto-btn3"></div>
            <div class="auto-btn4"></div>
        </div>
    </div>

    <nav class="manual-navigation" aria-label="Navegação do carrossel">
        <label for="radio1" class="manual-btn"></label>
        <label for="radio2" class="manual-btn"></label>
        <label for="radio3" class="manual-btn"></label>
        <label for="radio4" class="manual-btn"></label>
    </nav>
</section>
```

- Os inputs `type="radio"` funcionam como controladores para alternar entre as imagens.
- As tags `<img>` exibem as imagens do carrossel.
- A `<div class="navigation-auto">` serve para a navegação automática.
- A `<nav class="manual-navigation">` contém os botões de navegação manual, estilizados como círculos.

Estilizando o Slider com CSS

O estilo a seguir garante que as imagens ocupem toda a largura da tela e que o carrossel funcione corretamente:

```
.slider {
    margin: 0 auto;
    width: 100%;
    max-width: 100vw;
    height: auto;
    overflow: hidden;
    position: relative;
}

.slides {
```

```
    display: flex;
    transition: transform 0.5s ease;
    width: 100%;
    height: auto;
}

.slides img {
    width: 100%;
    height: auto;
    flex: 0 0 100%;
    object-fit: cover;
}

.slides input {
    display: none;
}

.slide {
    position: relative;
    transition: 2s;
}
```

- `.slider` define a área do carrossel.
- `.slides` utiliza `display: flex` para colocar todas as imagens lado a lado.
- Cada `.slide` ocupa 100% da largura da tela.

A seguir, os botões de navegação manual e automática são estilizados para ficarem centralizados e com formato circular:

```
.manual-navigation {
    display: flex;
    justify-content: center; /* Garante que os botões fiquem centralizados
*/
    position: absolute;
    bottom: 1.25rem;
    left: 50%;
    transform: translateX(-50%);
    z-index: 1;
    width: 100%; /* Faz com que a navegação ocupe a largura total da
section */
    padding: 0 10px; /* Espaçamento interno para evitar que os botões saiam
da tela */
}

.manual-btn{
    border: 2px solid #000;
```

```
    padding: 5px;
    border-radius: 50%;
    cursor: pointer;
    margin: 0 10px;
    transition: 1s ease;
    max-width: 30px; /* Define um tamanho máximo para os botões */
    flex-shrink: 0; /* Evita que os botões encolham */
}

.manual-btn:hover {
    background-color: #000;
}
```

Por fim, controlamos a mudança das imagens com CSS puro, dependendo do botão selecionado:

```
#radio1:checked ~ .first {
    margin-left: 0;
}
#radio2:checked ~ .first {
    margin-left: -100%;
}
#radio3:checked ~ .first {
    margin-left: -200%;
}
#radio4:checked ~ .first {
    margin-left: -300%;
}
```

Automatizando o carrossel com JavaScript

Para deixar o carrossel de imagens mais dinâmico, podemos automatizar a transição entre as imagens utilizando JavaScript. Isso permite que as imagens mudem automaticamente a cada 5 segundos, proporcionando uma experiência mais fluida para quem acessa o site.

Abaixo está o código JavaScript que realiza essa automação:

```
let count = 1;
document.getElementById("radio1").checked = true;

setInterval( function() {
    nextImage();
}, 5000)

function nextImage(){
    count++;
    if(count>4){
        count = 1;
```

```
    }

    document.getElementById("radio" + count).checked = true;
}
```

- `let count = 1`; define uma variável para controlar qual imagem está sendo exibida.
- `document.getElementById("radio1").checked = true;` seleciona a primeira imagem como padrão ao carregar a página.
- `setInterval(...)` chama a função `nextImage()` a cada 5000 milissegundos (5 segundos).
- A função `nextImage()` incrementa o número da imagem atual. Quando chega na última (radio4), volta para a primeira (radio1), criando um ciclo infinito.
- A linha `document.getElementById("radio" + count).checked = true;` marca o botão correspondente à imagem atual, fazendo com que ela seja exibida.

Com isso, o carrossel alterna automaticamente entre as imagens, mesmo sem a interação do usuário, tornando a apresentação mais dinâmica.

Seção de partidas

Objetivo do HTML

O objetivo do HTML é estruturar e organizar as informações sobre os próximos jogos de futebol de forma clara e acessível. A utilização de elementos semânticos, como **<section>**, **<h2>**, **<ul>**, **<li>**, e **<article>**, permite que o conteúdo seja facilmente compreendido tanto por usuários quanto por mecanismos de busca. A estrutura hierárquica facilita a leitura e a navegação, proporcionando uma experiência de usuário mais agradável.

```html
<section class="games-container">
    <h2>Próximos Jogos de Futebol</h2>
    <ul id="games-list">
        <li>
            <article class="game-item">
                <p class="game-team">Flamengo vs Vasco</p>
                <p class="game-time">16:00</p>
                <p class="game-date">15/03/2025</p>
            </article>
        </li>

        <li>
            <article class="game-item">
                <p class="game-team">Palmeiras vs São Paulo</p>
                <p class="game-time">18:30</p>
                <p class="game-date">16/03/2025</p>
            </article>
        </li>

        <li>
            <article class="game-item">
                <p class="game-team">Grêmio vs Internacional</p>
                <p class="game-time">20:00</p>
                <p class="game-date">17/03/2025</p>
            </article>
        </li>

        <li>
            <article class="game-item">
                <p class="game-team">Corinthians vs Santos</p>
                <p class="game-time">21:00</p>
                <p class="game-date">18/03/2025</p>
            </article>
        </li>
    </ul>
</section>
```

- **<section class="games-container">**: Define uma seção que contém todos os jogos de futebol. A classe **games-container** é usada para aplicar estilos específicos a esta seção.
- **<h2>**: Título da seção, que informa ao usuário que a lista a seguir contém os próximos jogos de futebol.
- **<ul id="games-list">**: Uma lista não ordenada que contém os jogos. O ID **games-list** permite a aplicação de estilos específicos e manipulação via JavaScript, se necessário.
- **<li>**: Cada item da lista representa um jogo de futebol.
- **<article class="game-item">**: Um elemento que encapsula as informações de cada

jogo, permitindo que cada jogo seja tratado como uma unidade independente.

- **<p class="game-team">**: Exibe os times que estão jogando.
- **<p class="game-time">**: Exibe o horário do jogo.
- **<p class="game-date">**: Exibe a data do jogo.

Explicação do CSS

O objetivo do CSS é estilizar a seção de jogos de futebol, melhorando a apresentação visual e a usabilidade do conteúdo. As regras de estilo aplicadas garantem que a seção seja responsiva, atraente e fácil de ler. O uso de flexbox para a lista de jogos permite que os itens se ajustem de forma dinâmica, enquanto as transições e efeitos de hover melhoram a interatividade.

```css
.games-container {
    max-width: 1200px;
    margin: 0 auto;
    padding: 20px;
    background-color: #fff;
}

h2 {
    text-align: center;
    margin-bottom: 20px;
    color: #333;      Pedro-Moreno, 7 days ago • Substit
    text-decoration: none;
    font-size: 18px;
    padding: 10px 15px;

}

#games-list {
    display: flex;
    list-style: none;
    flex-wrap: wrap;
    gap: 15px;
    justify-content: center;
    padding: 0;
}


.game-item {
    background-color: #f0f0f0;
    padding: 15px;
    border-radius: 8px;
    box-shadow: 0 2px 5px rgba(0, 0, 0, 0.1);
    width: 250px;
    text-align: center;
    transition: transform 0.2s;
}
```

```css
.game-item {
    background-color: #f0f0f0;
    padding: 15px;
    border-radius: 8px;
    box-shadow: 0 2px 5px rgba(0, 0, 0, 0.1);
    width: 250px;
    text-align: center;
    transition: transform 0.2s;
}


.game-item:hover {
    transform: scale(1.05);
}

.game-team {
    font-size: 18px;
    font-weight: bold;
    color: #333;
}

.game-time {
    margin-top: 10px;
    font-size: 16px;
    color: #777;
}

.game-date {
    font-size: 14px;
    color: #888;
}
```

- **.games-container**: Define a largura máxima da seção, centraliza o conteúdo e aplica um fundo branco e um espaçamento interno.
- **h2**: Estiliza o título da seção, centralizando-o, definindo margens, cor, tamanho da fonte e preenchimento.
- **#games-list**: Estiliza a lista de jogos, utilizando flexbox para dispor os itens em linha, permitindo que eles se ajustem em várias linhas, com espaçamento entre eles.
- **.game-item**: Define o estilo de cada item do jogo, incluindo cor de fundo, preenchimento, bordas arredondadas, sombra e largura fixa. Também aplica uma transição para um efeito de escala ao passar o mouse.
- **.game-item:hover**: Aplica um efeito de transformação ao item do jogo quando o mouse passa sobre ele, aumentando ligeiramente seu tamanho para indicar interatividade.
- **.game-team**: Estiliza o nome dos times, definindo tamanho da fonte, peso e cor.
- **.game-time**: Estiliza o horário do jogo, definindo margens, tamanho da fonte e cor.
- **.game-date**: Estiliza a data do jogo, definindo tamanho da fonte e cor.

Seção de compra

```html
<section class="compra">
    <h2>Selecione a Categoria</h2>      You, 5 days ago • Uncommitted changes
    <div class="checkbox-group">
        <label>
            <input type="checkbox" name="categoria" value="minhaVida"> Minha Vida
        </label>
        <label>
            <input type="checkbox" name="categoria" value="estudante"> Estudante
        </label>
    </div>

    <h2>Quantidade de Ingressos</h2>
    <div class="quantity">
        <button onclick="alterarQuantidade(-1)">-</button>
        <span id="quantidade">1</span>
        <button onclick="alterarQuantidade(1)">+</button>
    </div>

    <h2>Opções de Pagamento</h2>
    <div class="payment-options">
        <label><input type="radio" name="pagamento" value="credito"> Cartão de Crédito</label>
        <label><input type="radio" name="pagamento" value="debito"> Cartão de Débito</label>
        <label><input type="radio" name="pagamento" value="pix"> Pix</label>
        <label><input type="radio" name="pagamento" value="boleto"> Boleto</label>
    </div>

    <button id="finalizar">Finalizar Compra</button>
</section>
```

Explicação do Código HTML

- **<h2>**: Define os títulos das seções.
- **Checkboxes**: Permitem selecionar a categoria do ingresso.
- **Quantidade de ingressos**: Usa botões para aumentar ou diminuir o número de ingressos.
- **Opções de pagamento**: Usa inputs do tipo `radio` para selecionar a forma de pagamento.
- **Botão "Finalizar Compra"**: Um botão para concluir a compra.

```css
.compra {
    margin: auto;
    background: #000;
    padding: 10px 20px;
    border-radius: 10px;
    box-shadow: 0 0 10px rgba(255, 255, 255, 0.2);
    text-align: center;
}

.compra h2 {
    margin-bottom: 20px;       You, 31 minutes ago •
    font-family: Arial, sans-serif;
    background-color: #111;
    color: #f7ff0b;
    text-align: center;
}

.checkbox-group {
    display: flex;
    justify-content: center;
    margin-bottom: 20px;
}

.checkbox-group label {
    background: #f7ff0b;
    padding: 10px;
    margin: 0 10px;
    border-radius: 5px;
    cursor: pointer;
}
```

```css
.quantity {
    display: flex;
    align-items: center;
    justify-content: center;
    gap: 10px;
    margin-bottom: 20px;
}

.quantity button {
    background: gold;
    border: none;
    padding: 10px 20px;
    cursor: pointer;
    border-radius: 5px;
    font-size: 16px;
}

.quantity span {
    font-size: 20px;
    font-weight: bold;
    color: #ffffff;
}

.payment-options {
    text-align: left;
    margin-bottom: 20px;
}
.payment-options label {
    display: block;
    margin: 5px 0;
    color: white;
}
```

```css
#finalizar{
    background: green;
    color: white;
    font-weight: bold;
}
```

Explicação do Código CSS

- `.compra`: Define o estilo da seção, incluindo cor de fundo e bordas arredondadas.
- `.checkbox-group`: Define a organização dos checkboxes.
- `.quantity`: Define a aparência da seleção de quantidade de ingressos.
- `.payment-options`: Define o espaçamento entre as opções de pagamento.
- `#finalizar`: Estiliza o botão de finalização da compra.

```javascript
function alterarQuantidade(valor) {
    let spanQuantidade = document.getElementById("quantidade");
    let quantidade = parseInt(spanQuantidade.innerText);

    if (quantidade + valor > 0) {
        spanQuantidade.innerText = quantidade + valor;
    }
}
```

Explicação do Código JavaScript:

- **`alterarQuantidade(valor)`**: A função recebe um valor como parâmetro, que pode ser `-1` ou `1`, dependendo se o usuário clica no botão de diminuição ou aumento da quantidade de ingressos.
- **`let spanQuantidade = document.getElementById("quantidade");`**: A função obtém o elemento `<span>` com o ID "quantidade", que exibe o número de ingressos selecionados.
- **`let quantidade = parseInt(spanQuantidade.innerText);`**: A quantidade atual de ingressos é lida do conteúdo do `<span>` e convertida de texto para número.
- **`if (quantidade + valor > 0)`**: A condição verifica se a nova quantidade após a soma do `valor` (1 ou -1) é maior que 0. Isso evita que a quantidade de ingressos seja negativa.
- **`spanQuantidade.innerText = quantidade + valor;`**: Se a condição for verdadeira, a quantidade no `<span>` é atualizada com a nova quantidade.

Como Funciona o Fluxo Completo:

1. O usuário seleciona uma categoria (ex: "Minha Vida" ou "Estudante").
2. O usuário clica no botão + para aumentar a quantidade de ingressos ou no botão - para diminuir.
3. A função JavaScript é chamada, ajustando a quantidade exibida no `<span id="quantidade">`.
4. O usuário escolhe uma opção de pagamento (cartão de crédito, débito, Pix, boleto).
5. O usuário pode clicar no botão "Finalizar Compra", mas o comportamento desse botão não está definido no código fornecido (seria necessário adicionar uma função para finalizar a compra).

Seção de Contato

Objetivo

A seção de contato é uma parte fundamental de muitos sites, permitindo que os visitantes se comuniquem com os administradores ou proprietários do site. Este trecho de código HTML cria um formulário de contato que coleta informações do usuário, como nome, e-mail e mensagem. O objetivo é facilitar a comunicação entre os usuários e a equipe do site.

Estrutura do Código

```html
<section id="contato">
    <h2>Fale Conosco!</h2>
    <p>Preencha o formulário abaixo para entrar em contato conosco. Responderemos o mais breve possível.</p>

    <form>
        <label for="nome">Nome:</label>
        <input type="text" id="nome" name="nome" required placeholder="Seu nome">
        <label for="email"> E-mail: </label>
        <input type="email" id="email" name="email" required placeholder="exemplo@pichu.com">
        <label for="mensagem">Mensagem:</label>
        <textarea id="mensagem" name="mensagem" rows="4" required placeholder="Sua mensagem"></textarea>
        <input type="submit" value="Enviar">
    </form>
</section>
```

Explicação Detalhada

1. **<section id="contato">**:
   a. Define uma seção do documento HTML com o identificador "contato". Isso permite que a seção seja facilmente referenciada por CSS ou JavaScript, além de melhorar a acessibilidade e a estrutura do documento.
2. **<h2>Fale Conosco!</h2>**:
   a. Um cabeçalho de nível 2 que serve como título da seção. Ele informa ao usuário que esta parte do site é dedicada ao contato.
3. **<p>**:
   a. Um parágrafo que fornece uma breve descrição do que o usuário deve fazer. Neste caso, ele orienta o visitante a preencher o formulário para entrar em contato.
4. **<form>**:
   a. Inicia um formulário HTML. Este elemento é crucial para coletar dados do usuário. O formulário pode ser enviado para um servidor para processamento posterior.
5. **<label for="nome">Nome:</label>**:
   a. Um rótulo associado ao campo de entrada de texto para o nome. O atributo **for** vincula o rótulo ao campo de entrada correspondente, melhorando a acessibilidade.
6. **<input type="text" id="nome" name="nome" required placeholder="Seu nome">**:
   a. Um campo de entrada de texto onde o usuário pode digitar seu nome.
   b. **id** e **name** são atributos que identificam o campo.
   c. **required** indica que este campo deve ser preenchido antes do envio do formulário.
   d. **placeholder** fornece um texto de exemplo que desaparece quando o usuário começa a digitar.
7. **<label for="email"> E-mail: </label>**:
   a. Um rótulo para o campo de entrada de e-mail, semelhante ao rótulo do nome.

8. **<input type="email" id="email" name="email" required placeholder="exemplo@pichu.com">**:
    a. Um campo de entrada específico para e-mails, que valida automaticamente se o formato do e-mail está correto.
    b. Os atributos funcionam da mesma forma que no campo de nome.
9. **<label for="mensagem">Mensagem:</label>**:
    a. Um rótulo para o campo de texto onde o usuário pode digitar sua mensagem.
10. **<textarea id="mensagem" name="mensagem" rows="4" required placeholder="Sua mensagem"></textarea>**:
    a. Um campo de texto multilinha que permite ao usuário escrever uma mensagem mais longa.
    b. **rows** define a altura do campo.
    c. **required** e **placeholder** funcionam da mesma forma que nos campos anteriores.
11. **<input type="submit" value="Enviar">**:
    a. Um botão de envio que, quando clicado, envia os dados do formulário para o servidor. O valor "Enviar" é o texto exibido no botão.

Alerta de Boas-Vindas

Objetivo

Este código tem como objetivo exibir uma mensagem de boas-vindas aos visitantes de um site dedicado ao futebol. Ao carregar a página, um alerta é mostrado, criando uma interação imediata com o usuário e reforçando o tema do site.

Estrutura do Código

```
<body onload="mostrarAlerta()">
...

function mostrarAlerta(){
    alert("Você é fanatico por futebol? Então é muito bem vindo
aqui!");
}
```

1. **<body onload="mostrarAlerta()">**:
    a. O atributo **onload** chama a função **mostrarAlerta()** assim que a página é carregada, garantindo que o alerta seja exibido imediatamente.
2. **function mostrarAlerta()**:
    a. Esta função, definida em JavaScript, utiliza **alert()** para exibir uma mensagem ao usuário, reforçando a temática do site e criando uma interação amigável.

Footer(rodapé)

Explicação código HTML

```
<footer>
    <nav aria-label="Redes sociais">
        <ul class="social-media">
            <li>
                <a href="https://www.facebook.com" target="_blank">
                    <img src="img/facebook-icon.png" alt="Facebook">
                </a>
            </li>
            <li>
                <a href="https://www.linkedin.com" target="_blank">
                    <img src="img/linkedin-icon.png" alt="LinkedIn">
                </a>
            </li>
            <li>
                <a href="https://www.instagram.com" target="_blank">
                    <img src="img/instagram-icon.png" alt="Instagram">
                </a>
            </li>
            <li>
                <a href="https://www.tiktok.com" target="_blank">
                    <img src="img/tiktok-icon.png" alt="TikTok">
                </a>
            </li>
        </ul>
    </nav>
    <p> &copy; 2025 PICHU EVENTOS LTDA. - Todos os direitos reservados </p>
</footer>
```

- A **<nav>** contém uma lista não ordenada **<ul>**, com links (**<a>**) para cada rede social.
- Dentro de cada link, há uma imagem (**<img>**) que representa o ícone de cada rede social, e o **atributo target="_blank"** faz com que a página seja aberta em uma nova aba do navegador.
- O **<p>** no final contém o texto de direitos autorais.

Explicação código CSS

```css
footer {
    display: flex;
    justify-content: space-between;
    align-items: center;
    background-color: #000000;
    padding: 20px 20px;
    text-align: center;
}

footer p {
    color: #ffffff;
    margin: 0;
}

footer .social-media {
    display: flex;
    list-style: none;
}
```

- O **footer** é estilizado com **flexbox**, o que significa que seus elementos internos (como a lista de redes sociais e o texto de copyright) são alinhados de maneira flexível.
- **justify-content: space-between;** coloca o conteúdo do **footer** (a lista e o texto de copyright) nas extremidades opostas da tela, enquanto **align-items: center;** os alinha verticalmente ao centro.
- O **fundo** do rodapé é preto (**background-color: #000000;**), e o texto do **copyright** tem a cor branca (**color: #ffffff;**).
- A classe **.social-media** tem **display: flex;**, o que coloca os ícones das redes sociais em linha, lado a lado.

Responsividade (Media Queries)

```css
@media (max-width: 768px) {
    footer {
        flex-direction: column;
        text-align: center;
    }

    footer .social-media {
        display: flex;
        justify-content: center;
        margin-bottom: 10px;
    }

    footer .social-media a {
        margin: 0 0;
    }
}
```

- Quando a tela tem **largura máxima de 768px** (como em dispositivos móveis), o **footer** muda seu layout para **direção de coluna** (**flex-direction: column;**) para empilhar os elementos verticalmente. Isso melhora a acessibilidade e a navegação em telas pequenas.
- A lista de **social-media** também é centralizada, e a margem entre os ícones é ajustada para garantir que o layout fique adequado para diferentes tamanhos de tela.

Conclusão

Ao longo deste tutorial, foi possível compreender e aplicar na prática os principais fundamentos do desenvolvimento web utilizando HTML, CSS e JavaScript. Começamos estruturando a base do site com HTML5, organizando o conteúdo de forma semântica e acessível. Em seguida, aplicamos estilos com CSS para garantir uma identidade visual atrativa e uma navegação intuitiva, explorando técnicas como Flexbox e media queries para tornar o layout responsivo em diferentes dispositivos.

Também aprendemos a criar elementos interativos, como o carrossel de imagens, e a automatizá-lo com JavaScript puro, o que tornou a experiência do usuário mais dinâmica e moderna. Durante o desenvolvimento, exercitamos habilidades essenciais como separação de responsabilidades entre estrutura, estilo e comportamento, além de boas práticas de organização de código.

Mais do que construir um site de venda de ingressos, o projeto proporcionou uma experiência rica de aprendizado, onde transformamos uma paixão em comum — o futebol — em uma aplicação funcional. Esperamos que este guia sirva como base para a criação de outros projetos e incentive o aprofundamento no universo do desenvolvimento web.