PET.COMP

FUNDAMENTOS DE DESENVOLVIMENTO WEB

Centro Federal de Educação Tecnológica de Minas Gerais - CEFET-MG

Diretora-Geral
Carla Simone Chamon

Vice-Diretor
Conrado de Souza Rodrigues

Diretor de Graduação
Moacir Felizardo de França Filho

Diretor de Extensão e Desenvolvimento Comunitário
Patterson Patrício de Souza

Diretor do Campus Leopoldina
José Geraldo Ribeiro Júnior

Diretor Adjunto do Campus Leopoldina
Douglas Martins Vieira da Silva

Chefe do Departamento de Computação
Alexander Corrêa dos Santos

Subchefe do Departamento de Computação
Samuel da Costa Alves Basílio

Coordenador do Curso Técnico em Informática
Luan Soares Oliveira

Subcoordenadora do Curso Técnico em Informática
Tatiana Barbosa de Azevedo

Coordenador do Curso de Engenharia de Computação
Gustavo Montes Novaes

Subcoordenador do Curso de Engenharia de Computação
Fabiano Pereira Bhering

Centro Federal de Educação Tecnológica de Minas Gerais - CEFET-MG
Rua José Peres, 558 - Centro - Leopoldina - MG
Telefone: (32)3449-2346
www.leopoldina.cefetmg.br

PET.COMP

Fundamentos de Desenvolvimento Web

Créditos

Design

Capa: João Victor Domingos e Souza

Projeto Gráfico: João Victor Domingos e Souza

Equipe Editorial

Arthur Mendes Rocha Alves
Gabriella Castro Barbosa Costa Dalpra
João Victor Domingos e Souza
Lorenzo Jordani Bertozzi Luz
Lukas Julius Wolf
Luan Soares Oliveira
Luís Augusto Mattos Mendes
Marcela Gomes Pinheiro
Mateus Pereira Fernandes
Victor Borges Loures de Paula

Revisão

Gabriella Castro Barbosa Costa Dalpra
Luan Soares Oliveira
Luís Augusto Mattos Mendes

Prefácio

É com muito orgulho que apresentamos este material em formato de guia que foi elaborado com muita dedicação e carinho pela equipe do Programa de Educação Tutorial do Curso de Engenharia de Computação do CEFET-MG (Centro Federal de Educação Tecnológica de Minas Gerais), campus Leopoldina. Almejamos oferecer uma experiência de aprendizado agradável e didática, utilizando o Visual Studio Code para a criação de páginas web. A estrutura do conteúdo está organizada de forma a facilitar o aprendizado, dividindo-se em módulos os quais abordam os assuntos introdutórios de HTML, CSS e Bootstrap. Esperamos que este material seja um recurso valioso para você e que ao final deste curso, você se sinta confiante para aprofundar seus conhecimentos em desenvolvimento de sites.

Boa sorte e bom design!

Lukas Julius Wolf
Membro PET.COMP

Objetivos do Guia

- Dominar os conceitos básicos de HTML e CSS;
- Aprender a utilizar o framework Bootstrap[1] para criar layouts responsivos;
- Desenvolver uma página web simples e funcional utilizando as ferramentas ensinadas no curso.

Materiais Necessários

- Computador ou notebook com acesso à internet;
- Editor de código (Recomendação: VS Code);
- Imagens autorais ou disponíveis na Web (Visite sites como: Unsplash, Pexels e Pixabay para encontrar imagens de alta qualidade e gratuitas) para utilização para a criação das páginas.

1 https://getbootstrap.com/

Sumário

MÓDULO I

CONCEITOS FUNDAMENTAIS

Este módulo é uma introdução essencial às técnicas e tecnologias fundamentais usadas na criação de *websites* e aplicações *web*. Através deste guia, exploraremos os componentes do *front-end* — o aspecto do desenvolvimento web que lida com a interação direta do usuário — e ofereceremos uma visão geral do *back-end*, que gerencia a lógica e os dados dos bastidores.

Compreender o desenvolvimento web é crucial em um mundo digitalizado, onde uma forte presença online é vital para o sucesso empresarial e comunicação eficaz. Este módulo é projetado para proporcionar o conhecimento necessário para iniciar ou aprimorar sua capacidade de criação digital.

CONCEITOS FUNDAMENTAIS

1.1 A Evolução da Web

A *web* tem passado por várias fases de desenvolvimento desde a sua criação por Tim Berners-Lee em 1989. Cada fase trouxe novas tecnologias, padrões e práticas que moldaram a forma como interagimos com informações e uns com os outros online. Vamos explorar as principais eras da evolução da web:

1. Web 1.0 (A Web Estática):

- **Período:** final dos anos 1980 até meados dos anos 2000.
- **Características:** a Web 1.0 é frequentemente referida como a "*web* estática". Era basicamente um meio de disseminação de informação, onde as páginas eram codificadas em HTML básico sem interatividade além de *hiperlinks*, apresentando um formato muito similar a um jornal ou revista online.
- **Tecnologias:** HTML básico, imagens em formatos *GIF* e *JPEG*, e a infraestrutura inicial do HTTP.

2. Web 2.0 (A Web Interativa):

- **Período:** meados dos anos 2000 até o início dos anos 2010.
- **Características:** a Web 2.0 é marcada pela interativde e pela participação do usuário. Ela introduziu conceitos de redes sociais, *blogs* e *wikis*, permitindo aos usuários não apenas consumir conteúdo, mas também criar e compartilhar o seu próprio.

- **Tecnologias:** CSS, JavaScript mais avançado, *frameworks* de *front-end* como jQuery, e a emergência de *APIs* REST para serviços *web*.

3. Web 3.0 (A Web Semântica):

- **Período:** início dos anos 2010 até o presente (continua evoluindo).

- **Características:** A Web 3.0 é focada em tornar a *web* mais "inteligente", com máquinas capazes de entender e processar informações de forma semelhante aos humanos, através do uso de dados estruturados e tecnologias semânticas. Ela visa criar uma experiência mais personalizada e automatizada.

- **Tecnologias:** HTML5, CSS3, microformatos, RDF, OWL, SPARQL, e *frameworks* mais sofisticados como *Angular*, *React* e Vue.js.

4. Web 4.0 (A Web Conectada):

- **Período:** visão futurista ainda em desenvolvimento.

- **Características:** Embora ainda seja um conceito em evolução, a Web 4.0 é muitas vezes associada à "Internet das Coisas" (IoT), onde dispositivos conectados à internet comunicam-se e operam juntos de forma inteligente e autônoma.

- **Tecnologias:** tecnologias emergentes em IoT, AI (Inteligência Artificial), integração avançada de dados em tempo real, e redes neurais.

Cada etapa da evolução da web não apenas mudou como acessamos e interagimos com o conteúdo online, mas também como desenvolvemos e projetamos experiências digitais. Compreender esta evolução ajuda os desenvolvedores a antecipar tendências e adaptar-se às novas tecnologias e metodologias que continuam a emergir.

Figura 1: Linha do tempo da evolução da web.

1.2 Definição e Usos

Desenvolvimento *web* refere-se ao processo de criação e manutenção de websites e aplicações web. Ele abrange uma variedade de disciplinas e tecnologias que possibilitam a construção de interfaces interativas, gestão de dados e integração com outros serviços online. Este campo envolve tanto o desenvolvimento do front-end, a parte do website com a qual os usuários interagem diretamente, quanto o back-end, que é a infraestrutura que atua por trás das cortinas, processando as interações, armazenando dados e gerenciando a lógica de negócios. Neste guia, abordaremos a parte introdutória referente ao front-end.

O desenvolvimento web é crucial na era digital atual, pois é responsável por criar interfaces que possibilitam a divulgação visual e textual de quaisquer conteúdos, bem como aplicações responsáveis por criar uma conexão entre fornecedor e cliente nas mais diversas áreas como comunicação, comércio, educação, entretenimento e diversas outras. Aqui está disposta um rol, não taxativo, de pontos que destacam sua importância:

- **Comunicação global:** websites oferecem uma plataforma para que indivíduos e organizações se comuniquem com uma audiência global, compartilhando informações, ideias e produtos.

- **Economia digital:** a web facilita uma parte significativa do comércio mundial, desde pequenos empreendedores vendendo produtos artesanais até grandes corporações oferecendo uma ampla gama de serviços.

- **Economia digital:** a web facilita uma parte significativa do comércio mundial, desde pequenos empreendedores vendendo produtos artesanais até grandes corporações oferecendo uma ampla gama de serviços.

- **Acesso à educação:** plataformas educacionais online permitem o acesso a cursos e materiais de aprendizagem de qualquer lugar do mundo, democratizando a educação e oferecendo oportunidades de aprendizado contínuo.

- **Inovação em serviços:** o desenvolvimento web permite a criação de novos serviços e a transformação de serviços existentes, utilizando tecnologias emergentes como inteligência artificial, big data e computação em nuvem para oferecer soluções mais eficientes e personalizadas.

- **Engajamento e interatividade:** websites interativos e aplicações web engajam os usuários de maneira mais efetiva, proporcionando uma experiência mais rica e personalizada através de interfaces intuitivas e recursos dinâmicos.

Em resumo, o desenvolvimento web não apenas capacita empresas e indivíduos a atingirem seus objetivos de comunicação e comerciais, mas também desempenha um papel vital em várias áreas da sociedade moderna, influenciando como vivemos, trabalhamos e interagimos.

1.3 Estrutura de um Website

Um website é composto por várias camadas que trabalham em conjunto para oferecer uma experiência de navegação coesa e funcional. Essas camadas incluem aspectos de front-end e back-end, além da infraestrutura de hospedagem e domínio. A seguir exploraremos cada um desses componentes:

1. Front-end (Interface do Usuário): O front-end de um website é tudo o que os usuários visualizam e interagem diretamente no navegador. Ele é construído usando três principais tecnologias:

- **HTML (Hypertext Markup Language):** define a estrutura e o conteúdo do site. O HTML é usado para organizar textos, links, imagens e outros conteúdos.

- **CSS (Cascading Style Sheets):** responsável pelo design visual e layout do site. O CSS é usado para estilizar elementos HTML, definindo cores, fontes, espaçamentos e responsividade.

- **JavaScript:** diferentemente do HTML, marcador textual, e CSS, estilizador dos componentes web, o Javascript adiciona interatividade ao site, criando páginas dinâmicas que respondem a interações do usuário, como cliques, rolagens e entradas de teclado.

2. Back-end (Lógica de Servidor): O back-end é a parte do site que os usuários não veem. Ele é responsável pelo armazenamento, processamento e segurança dos dados do site. O back-end geralmente consiste em:

- **Linguagens de programação:** Linguagens como Python, PHP, Ruby, Java e diversas outras são utilizadas para desenvolver a lógica de negócios do site.

- **Servidor web:** um software que aceita solicitações via Internet e entrega os dados ao usuário, como Apache ou Nginx.

- **Sistema Gerenciador de Banco de Dados:** utilizado para armazenar e gerenciar dados, como MySQL, PostgreSQL e MongoDB. O primeiro utiliza armazenamento relacional de dados e o último possibilita o armazenamento de objetos. Já o segundo permite ambos os tipos de armazenamento.

3. Hospedagem e Domínio:

- **Serviços de hospedagem:** são plataformas que armazenam os arquivos do site e os tornam acessíveis na web. Exemplos incluem serviços como AWS (Amazon Web Services), Bluehost e HostGator.

    - **Domínio:** é o endereço do site na web, como www.exemplo.com. O domínio é fundamental para que os usuários possam encontrar facilmente o site.

4. Protocolos de Comunicação: Como qualquer tipo de transferência de dados utilizando redes de computadores, são necessários padrões para envio de dados a fim de estabelecer uma conexão estável e coerente entre os diversos tipos de aparelhos eletrônicos conectados à rede, ou seja, a interoperabilidade. Para atingir este objetivo na comunicação web um protocolo foi definido para realizar essa função, o HTTP. Posteriormente foi necessária a criação de uma camada de segurança neste protocolo, surgindo assim o HTTPS. Eles podem ser definidos como:

- **HTTP (*Hypertext Transfer Protocol*):** é o protocolo usado para a transferência de informações entre o navegador do usuário e o servidor.

- **HTTPS (Secure HTTP):** o HTTPS funciona da mesma forma que o HTTP, porém inclui uma camada de segurança para proteger as informações.

Cada um desses componentes trabalha em conjunto para criar uma experiência de usuário suave e funcional. A boa prática no desenvolvimento web envolve a compreensão e a habilidade para integrar eficazmente estas tecnologias e conceitos.

1.4 Protocolos e Padrões da Web

Os protocolos e padrões da web são acordos técnicos que permitem a interoperabilidade e a comunicação eficaz entre diferentes sistemas de computação na internet. Esses protocolos e padrões são fundamentais para o funcionamento da web global. Aqui estão alguns dos mais importantes:

1.4.1 Protocolos de Comunicação: Como qualquer tipo de transferência de dados utilizando redes de computadores, são necessários padrões para envio de dados a fim de estabelecer uma conexão estável e coerente entre os diversos tipos de aparelhos eletrônicos conectados à rede, ou seja, a interoperabilidade. Para atingir este objetivo na comunicação web um protocolo foi definido para realizar essa função, o HTTP. Posteriormente foi necessária a criação de uma camada de segurança neste protocolo, surgindo assim o HTTPS.

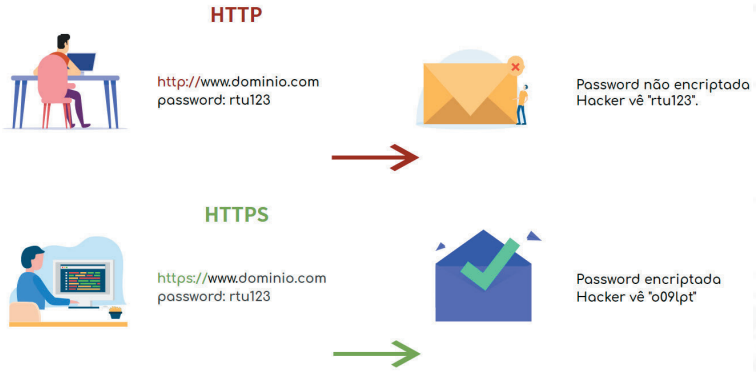


Figura 2: Funcionamento da Criptografia de senhas.

1.4.2 Padrões Web:

- **HTML (Hypertext Markup Language):** como já mencionado, é a linguagem de marcação usada para criar e estruturar páginas na internet.

- **CSS (Cascading Style Sheets):** usado para controlar a apresentação visual das páginas web, permitindo a separação entre conteúdo e design.

- **JavaScript:** uma linguagem de programação que é usada para criar interações dinâmicas nas páginas web.

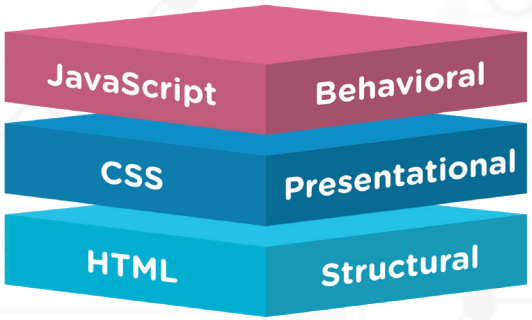


Figura 3: Camadas do Desenvolvimento Web - Estrutura, Apresentação e Comportamento.

1.4.3 Organizações de Padrões:

- **W3C (World Wide Web Consortium):** é a principal organização de padrões para a web, responsável por manter e publicar os padrões de HTML, CSS, entre outros.

- **ETF (Internet Engineering Task Force):** Desenvolve e promove padrões voluntários, principalmente em relação aos protocolos usados na internet.**1.4.4 Acessibilidade e Internacionalização:**

- **WCAG (Web Content Accessibility Guidelines):** Diretrizes desenvolvidas pelo W3C que especificam como tornar o conteúdo da web acessível a pessoas com deficiências.

- **Unicode:** Um padrão para codificação de caracteres, fundamental para suportar texto em todos os idiomas escritos para processamento em computadores.

Esses protocolos e padrões são projetados para garantir que a web seja um ambiente seguro, eficiente e interoperável. Compreender e aplicar esses padrões é crucial para qualquer desenvolvedor web, pois eles afetam diretamente a funcionalidade, a acessibilidade e a segurança

1.5 Ferramentas Essenciais

Existem diversas ferramentas que auxiliam o programador a desenvolver projetos web. Neste guia, focaremos em ferramentas mais básicas para quem está iniciando no desenvolvimento web.

1.5.1. Editores de Código e Ambientes de Desenvolvimento Integrado (IDEs - *Integrated Development Environment*).

- Visual Studio Code (VS Code): um editor de código

de linguagens de programação e tecnologias web. Ele oferece funcionalidades como destaque de sintaxe, autocompletar de código, gestão de versões integrada e extensões personalizáveis.

- Sublime Text: conhecido por sua velocidade e eficiência, é outro editor popular entre os desenvolvedores web. Oferece recursos como múltiplas seleções, modo de distração zero e uma API robusta para extensões.

- Atom: editor de código de fonte aberta que é altamente personalizável, mas um pouco mais pesado que o Sublime Text. Também suporta plug-ins escritos em Node.js e integra-se bem com Git.

- IntelliJ IDEA: um IDE mais robusto, especialmente popular entre os desenvolvedores que trabalham com Java e outras linguagens compiladas. Possui licença gratuita para estudantes.

1.5.2. Frameworks e Bibliotecas:

- Bootstrap: um *framework* que utiliza HTML, CSS e JS (Javascript) para desenvolver websites responsivos, garantindo que o site funcione bem em todos os dispositivos e resoluções de tela.

Para quem tem interesse em aprofundar nesse conteúdo existem outras duas ferramentas muito utilizadas:

- React: uma biblioteca JavaScript para construir interfaces de usuário interativas. É mantida pelo Facebook e é amplamente usada para construir aplicativos web de página única.

- Angular: um framework de desenvolvimento robusto para a construção de aplicativos web dinâmicos, mantido pelo Google.

1.5.3. Ferramentas de Desenvolvimento Front-end.

Browsersync e Live Server: ferramentas que permitem a visualização em tempo real de alterações no código em um ambiente de desenvolvimento local. Live Server é particularmente popular entre usuários de VS Code, pois recarrega automaticamente a página quando arquivos são modificados, facilitando o processo de teste e desenvolvimento.

- Sass: um pré-processador CSS que permite escrever código CSS mais limpo e reutilizável com a ajuda de variáveis.

- Webpack: um empacotador de módulos para aplicações JavaScript modernas, que ajuda a gerenciar dependências, plugins e transformações.

MÓDULO II

INTRODUÇÃO AO HTML

Neste módulo, exploraremos a estrutura básica do HTML, começando pelo essencial `<!DOCTYPE html>` até componentes mais complexos de uma página. Abordaremos as primeiras tags e como elas contribuem para a definição do documento, seguido por uma imersão nas tags de conteúdo que delineiam cabeçalhos, parágrafos, e outras formas de mídia. Através de exemplos práticos e explicações detalhadas, você aprenderá a criar documentos HTML robustos e acessíveis, preparando o terreno para uma aprendizagem mais aprofundada em estilização e interatividade com CSS e JavaScript.

Prepare-se para mergulhar na linguagem que é a espinha dorsal da web, onde cada tag e atributo tem seu papel no vasto mundo do desenvolvimento web. Este módulo irá equipá-lo com os conhecimentos fundamentais necessários para iniciar sua jornada em criar páginas web atrativas e eficientes.

INTRODUÇÃO AO HTML

HTML (HyperText Markup Language) é a linguagem fundamental utilizada para criar e estruturar páginas na web. Ela permite que desenvolvedores definam a organização e o layout do conteúdo, incluindo texto, imagens, links e outros elementos multimídia. Utilizando uma série de tags, como <html>, <head>, <body>, e <p>, HTML forma a base de qualquer página web, estruturando o conteúdo de maneira semântica e hierárquica. Essa linguagem é essencial para a construção de sites, pois permite que os navegadores interpretem e exibam o conteúdo de forma adequada aos usuários, criando a experiência de navegação que conhecemos hoje.

2.1 Estrutura Básica do HTML

O documento HTML sempre inicia com o que chamamos de estrutura básica. Esta estrutura é quase que imutável e sempre começará seu HTML por esse código. Geralmente os editores como o VS Code já tem atalhos para iniciar os documentos HTMLs com essa estrutura, logo, você não precisa se preocupar em decorá-la. Para realizar essa ação no VS Code basta escrever html:5, podemos ver abaixo como ela se inicia:

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
   <meta name="viewport" content="width=de-
vice-width, initial-scale=1.0">
    <title>Document</title>
</head>
<body>

</body>
```

2.2 Primeiras Tags e de Conteúdo

É possível compreender o documento em HTML de uma maneira muito simples, através de uma divisão de blocos das tags essenciais, conforme a seguinte estrutura:

- **Doctype - Definindo o documento**

Uma coisa importante: **SEMPRE** deve existir o doctype, sendo definido pela marcação **<!DOCTYPE html>**.

O **DOCTYPE** não é uma tag **HTML**, mas uma instrução para indexação, dessa forma qualquer navegador e outros programas podem ler o site, dessa forma é possível informar que os dados encontrados ali é um código **HTML**. Consequentemente, torna-se possível saber o que fazer para mostrar seu site da forma correta. Lembre-se: o doctype é **OBRIGATÓRIO** e deve ser sempre a **PRIMEIRA LINHA** do seu documento.

- **Head**: os dados fornecidos dentro desta tag carregam informações para serem indexadas por máquinas. Normalmente estes dados são implícitos para uso e controle do documento: vinculação com outros arquivos, aplicação de lógica de programação de scripts e metadados. Todo o conteúdo do cabeçalho fica delimitado entre a abertura e fechamento da tag **<head>**.

```
<!DOCTYPE html>
<html lang="en">
   <head></head>
</html>
```


Obs.: uma boa prática de programação é sempre manter o código identado — ou seja, mantê-lo organizado visualmente — para realizar essa ação basta pressionar a tecla "*Tab*" ao abrir uma tag no interior de outra, criando dessa forma uma distância entre as tags e impedindo que todo o código fique "reto" dificultando a leitura do arquivo. Após a abertura e o fechamento da head pressionamos a tecla "*Enter*". Dentro do espaço criado da <head> escreveremos as meta informações da nossa página. A seguir estão algumas tags essenciais em todo documento **HTML:**

`<title>`: Entre a tag de abertura e fechamento é possível definir o título da página web.

`<meta charset="UTF-8">`: O UTF-8 é a codificação dos caracteres mais usada no mundo, pois consegue trazer todos os acentos que comentamos, além de interpretar corretamente as outras línguas.

É Unicode, como se fosse um grande dicionário de caracteres que usaremos na aplicação

`<meta http-equiv="X-UA-Compatible" content="IE=edge">`: É uma configuração para sempre utilizarmos sua versão mais recente neste navegador especificamente, no caso, Microsoft Edge.

`<meta name="viewport" content="width=device-width, initial-scale=1.0">:` Esta tag carrega as definições de como a página web irá compreender e renderizar as informações. A partir da definição "***name=viewport***" fica especificado que as definições da tela do aparelho serão utilizadas para renderizar a página, obten-

do a largura e altura da tela do dispositivo. Já no ***content*** é definido que será utilizado a largura de tela do dispositivo pelo comando "***width=devicewidth***". E por fim "***initial-scale*=1.0"** define que a escala será de 1.0 referente à tela do dispositivo.

- **Body**: Trata-se do documento em si, ou seja, a informação legível para o usuário/leitor do documento. É todo e qualquer texto que se deseja apresentar, assim como toda e qualquer forma de mídia de saída (imagens, sons, miniaplicativos embutidos, conteúdo multimídia, etc). Além disso, toda a apresentação de entrada de dados (formulários) também se aplica nesta seção do documento. Na prática, o corpo do documento é delimitado pelo par de tags `<body>` e `</body>`. Este é o preceito básico que deve estar muito bem claro para você: onde as marcações se aplicam e quais são os resultados deste modelo. Por exemplo: se você deseja informar conteúdo textual para saída legível ao usuário do seu sistema web, esta marcação deverá obrigatoriamente estar no bloco do corpo da página. Já para definir qual o tipo de codificação da página (uma meta informação do documento), esta deve obrigatoriamente estar marcada no cabeçalho do mesmo documento, como dito anteriormente no tópico **2.2.2**. Dentro do elemento ***body*** sua estrutura de página terá os elementos semânticos da construção da sua página, onde serão declarados e identificados cabeçalhos, rodapé, conteúdo principal, etc.

Obs.: se estiver utilizando o *Visual Studio Code*, ao digitar '!' + TAB o editor de código cria a sintaxe básica do

documento tipo html.

A importância de se utilizar um editor de código, como o *Visual Studio Code, por* exemplo, é que ele facilita na escrita completando automaticamente a tag que abrimos com a tag posterior, ou seja, de fechamento. O que seria a tag posterior? Temos algumas tags que, para serem implementadas, precisam de uma abertura e de um fechamento. Veremos algumas delas ao longo de todo guia. Em suma, toda tag de fechamento possui uma barra (/) após o sinal '<', como pode ser visto na imagem a seguir:

- **Outras Tags:**

1. **Definindo títulos e subtítulos:** Existe uma outra tag denominada heading, a qual declaramos somente com `<h>`, o qual é acompanhado de um número que designa o peso daquele título. Podemos utilizá-la tanto para títulos quanto para subtítulos. Vamos envolver o nosso título com a heading no tamanho 1 da seguinte forma: `<h1>Exemplo de título</h1>`.

Obs.: Vale ressaltar que a tag h1 deve ser utilizada apenas uma única vez na página. Considera-se este tipo de ação boa prática de programação e, inclusive, contribui para a acessibilidade da sua página web pois programas de acessibilidade que realizaram leituras das páginas utilizam essas tags para navegar por toda ela, pulando de tag h em tag h, portanto o correto aninhamento das tags fornecem uma melhor acessibilidade.

2. **Definindo parágrafos:** Quaisquer textos devem utilizar a tag `<p>`.

3. **Cabeçalho:** Para definir um cabeçalho introdutório ou um conjunto de links deve-se utilizar a tag `<header>`. Normalmente contém:

    - Um ou mais elementos de cabeçalho (`<h1> - <h6>`);
    - Logotipo ou ícone;
    - Informações de autoria;

4. **O conteúdo principal:** toda informação principal do conteúdo deve ser inserido na tag `<main>`. O conteúdo dentro do elemento deve ser exclusivo do documento. Ele não deve conter nenhum conteúdo repetido em documentos, como barras laterais, links de navegação, informações de direitos autorais, logotipos de sites e formulários de pesquisa.

5. **Rodapé**: Como o próprio nome diz, a seguinte tag é responsável por englobar todos os elementos que pertencem ao rodapé do documento: <footer>.

```
<!DOCTYPE html>
<html lang="pt-br">
<head>
   <meta charset="UTF-8">
 <meta http-equiv="X-UA-Compatible" content="IE=edge">
   <meta name="viewport" content="width=device-width,
initial-scale=1.0">
   <title>Portfolio</title>
</head>
<body>
<header></header>
<main></main>
<footer></footer>
</body>
</html>
```


2.3 Comentários e Citações

- **Comentários:** Como a maioria dos tipos de documentos, comentários podem ser inseridos ao longo dele a fim de fornecer informações pertinentes a ele. Esses comentários não são executados e nem exibidos pelo navegador. Para realizar tal ação basta utilizar a seguinte formatação:

```
<!-- Comente aqui seu texto -->
```

- **Citações**: Caso seja necessário fazer alguma citação, dois tipos de tags são fornecidas, elas são: `<q>` e `<blockquote>`. A utilização delas se dá respectivamente pelos seguintes motivos: citação curta e seção.

2.4 Tags Semânticas

Tags semânticas são *tags* que possuem um significado, elas dão sentido a informação de texto ao navegador e buscadores. Por exemplo: para definir um painel de navegação, cabeçalhos, conteúdos laterais, seções de conteúdos, entre outros. É uma boa prática **sempre** utilizar essas tags semânticas com o intuito de melhorar o entendimento do código. Ademais, esse tipo de prática contribui muito no SEO do site (Otimização para motores de busca, responsável por rankear melhor sua aplicação web nos motores de buscas como o Google).

Como demonstrado pela figura abaixo, algumas dessas tags são: *nav, header, section, aside e footer*. Duas delas já foram explicadas no tópico **2.2.4**, *header e footer*.

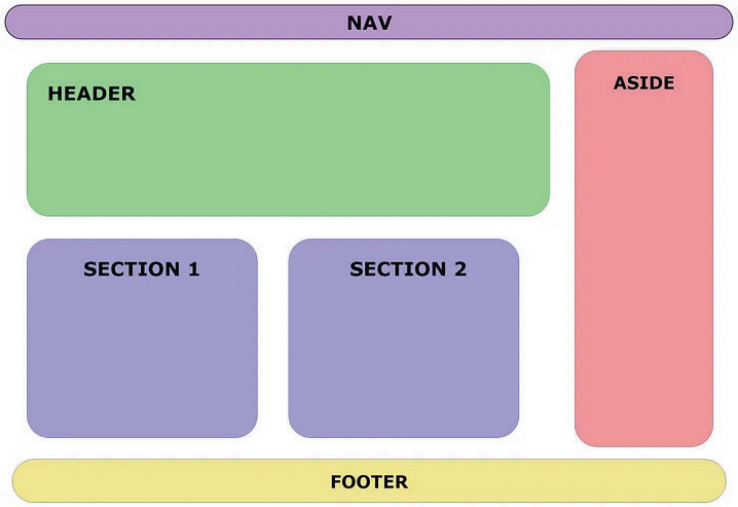


Figura 3: Camadas do Desenvolvimento Web - Estrutura, Apresentação e Comportamento.

- **Painel de navegação**: A *tag* `<nav>` é responsável por encapsular todos os elementos pertencentes ao painel de navegação.

```
<nav>
    <a href="/html/">HTML</a>
    <a href="/css/">CSS</a>
    <a href="/js/">JAVASCRIPT</a>
    <a href="/jquery/">jQuery</a>
 </nav>
```

- **Painel lateral**: Responsável por posicionar conteúdos que são importantes na página web mas que não pertencem às informações principais. Esses conteúdos são posicionados nas laterais e são definidos pela tag `<aside>`.

```
<p>O <b>HTML</b> é uma linguagem de
   marcação utilizada na construção de
         páginas na Web.
</p>
 <aside>
    <h2>Você sabia ?</h2>
    <dl>
        <dt>Linguagem de marcação:</dt>
        <dd>é uma linguagem de marcação
  utilizada na construção
                de páginas na Web.</dd>
    </dl>
</aside>
```

- **Seções**: Quando queremos separar o conteúdo da página através de seções utilizamos a tag `<section>`. Normalmente essas seções definem as margens entre os diferentes conteúdos da página e são utilizados para a navegação interna da própria página através de links. Esse método de navegação interna pode ser visto com mais detalhes no Módulo XX.

```
<section>
   <h1>Cabeçalho 1</h1>
   <p>Um monte de conteúdo incrível
       </p>
</section>
<section>
   <h1>Cabeçalho 2</h1>
  <p>Outro monte de conteúdo incrível</p>
 </section>
```

2.5 Tags sem semântica e estilos de formatação

Diferentemente dos tópicos anteriores, em que as tags possuem sentido semântico com os conteúdos apresentados, aqui serão tratadas tags que não possuem este sentido, não definindo um significado ao texto. Sua utilização se dá principalmente para separação genérica e estilização.

- **Divisão:** a fim de separar os diversos conteúdos de uma página de forma genérica e com uma estiliza-

ção específica usa-se a tag .

- **Quebra de texto:** quando se tem o interesse em realizar uma quebra de linha obrigatória ou gerar um espaçamento entre parágrafos utiliza-se a tag
.

- **Marcações textuais:** existem diversas tags que são utilizadas para estilizar um parágrafo, ou parte dele, de forma simples sem utilizar CSS. Essas tags são normalmente utilizadas aninhadas entre si e dentro de uma tag <p>. O rol taxativo a seguir exemplifica as opções de estilização:

`<i>`: Texto em itálico
`<b>`: Texto em negrito
`<strong>`: Texto importante
`<em>`: Texto enfatizado
`<mark>`: Texto marcado
`<small>`: Texto menor
`<del>`: Texto deletado
`<ins>`: Texto inserido
`<sub>`: texto subscrito
`<sup>`: texto sobrescrito

2.6 Inserir Imagens

Para inserir imagens em uma página web, você usa a *tag* `<img>`. Esta *tag* é autocontida (ou seja, não possui uma *tag* de fechamento) e possui vários atributos que você pode definir. Os dois atributos mais importantes são `src` (en. *source; pt.br. "fonte"*) e `alt` (en. "*alternative text"; pt.br.* "texto alternativo").

Aqui está a estrutura básica para inserir uma imagem:


```
<img src="caminho/para/a/imagem.jpg"
alt="Descrição da imagem">
```

Obs.: O atributo `alt` é responsável por fornecer um conteúdo escrito para substituir a imagem caso ela não seja exibida na página e também funcionará como um texto descritivo da imagem para questões de acessibilidade.

2.7 Tags de Lista e Hiperlink

1. **Lista:** Em **HTML**, você pode criar dois tipos principais de listas: listas não ordenadas (en. *unordered lists*) e listas ordenadas (en. *ordered lists*). Cada tipo de lista possui suas próprias tags específicas. Aqui estão as tags principais e como usá-las:

   a. **Lista Não Ordenada (`<ul>`):** Uma lista não ordenada exibe itens de lista com marcadores (en. *bullet points*).
      - **Tag de abertura e fechamento:** `<ul>` e `</ul>`
      - **Itens de lista:** `<li>` e `</li>`

```
<ul>
<li>Coffee</li>
<li>Tea</li>
<li>Milk</li>
</ul>
```

   b. **Lista Ordenada** (`<ol>`): Uma lista ordenada exibe itens de lista com números ou letras, indicando a ordem dos itens.
      - **Tag de abertura e fechamento:** `<ol>` e `</ol>`

- **Itens de lista:** `<li>` e `</li>`

```
<ol>
<li>Coffee</li>
<li>Tea</li>
<li>Milk</li>
</ol>
```

c. **Lista de Definições:** Uma lista de definições é usada para listas de termos e suas descrições.

- **Tag de abertura e fechamento:** `<dl>` e `</dl>`
- **Termo de definição:** `<dt>` e `</dt>`
- **Descrição do termo:** `<dd>` e `</dd>`

```
<dl>
      <dt>Coffee</dt>
      <dd>- black hot drink</dd>
      <dt>Milk</dt>
<dd>- white cold drink</dd>
</dl>
```

4. **Hiperlink:** A navegação da página se dá pelo uso da tag `<a>`, sendo ela responsável por gerar URLs clicáveis. Existem dois atributos importantes: `href` e `target`. As ações respectivas proporcionadas por esses atributos são: **direcionar o navegador do usuário até a url desejada** e **especificar onde esse novo direcionamento será aberto**, como por exemplo, possibilitando abrir essa url em nova aba (`_blank`) no navegador. Ainda vale ressaltar que as navegações podem ocorrer de forma **interna** e **externa**:

a. **Interna:** ocorre a partir das referências dentro da própria página com a utilização do atributo `href` da tag `<a>` e pelo `id` (identificador) dos elementos dispostos no documento. Abaixo está o exemplo deste tipo de navegação, nele a página irá deslizar até o elemento que possui o `id` titulo.

```
<a href="#titulo">Título</a>
```

b. **Externa:** com o atributo `href` é possível criar um link externo ao da página. O exemplo abaixo gera um item clicável que direciona o navegador para o website do PET.COMP utilizando uma nova aba no próprio navegador:

```
<a  href="https://www.petcomp.cefet-
mg.br/" target="_blank">PET.COMP
</a>
```

MÓDULO III

INTRODUÇÃO AO CSS

CSS, ou Folhas de Estilo em Cascata, é uma linguagem de estilo essencial para definir a apresentação visual de documentos HTML. Neste módulo, exploraremos como o CSS permite aos desenvolvedores separar o conteúdo da apresentação, oferecendo controle sobre layout, design e interatividade das páginas *web*. Você aprenderá a estrutura básica do CSS, incluindo seletores, declarações e a organização das regras.

Daremos um foco especial a conceitos como o modelo de caixa (*Box Model*), diferentes métodos de posicionamento, unidades de medida e a importância da especificidade e herança em CSS. Este conhecimento é fundamental para criar sites visualmente atraentes e responsivos, aptos para funcionar em diversos dispositivos e tamanhos de tela. Prepare-se para dominar o CSS e transformar suas páginas HTML em experiências ricas e envolventes.

INTRODUÇÃO AO CSS

CSS (en. *Cascading Style Sheets; pt.br. Folhas de Estilo em Cascata*) é uma linguagem de estilo utilizada para descrever a apresentação de documentos **HTML**. Ele permite que desenvolvedores web separem o conteúdo da estrutura visual, oferecendo maior flexibilidade e controle sobre o layout e o design das páginas. Desta forma é possível definir estilos para textos, ajustar espaçamentos, posicionar elementos, e criar animações, tornando a experiência do usuário mais agradável e interativa.

3.1 Sintaxe Básica

A sintaxe básica do **CSS** é composta por seletores e declarações. Um seletor indica qual elemento **HTML** será estilizado, enquanto a declaração é composta por uma propriedade e um valor, separados por dois pontos e encerrados por ponto e vírgula. Por exemplo:

```
p {
    color: blue;
    font-size: 16px;
}
```

Nesse exemplo, o seletor **p** aplica às declarações de cor azul e tamanho de fonte de 16 pixels a todos os elementos **<p>** da página.

3.2 Estrutura

A estrutura do **CSS** consiste em regras que são aplicadas aos elementos **HTML**. Cada regra é composta por um seletor e um bloco de declarações. As declarações, por sua vez, são compostas por propriedades CSS e seus respectivos valores. Além disso, é possível agrupar múltiplos seletores em uma única regra, separando-os por vírgulas, para

aplicar o mesmo conjunto de estilos a diferentes elementos.

3.3 Seletores

Os seletores são uma parte fundamental do **CSS**, permitindo que os desenvolvedores escolham quais elementos **HTML** estilizar. Existem vários tipos de seletores, incluindo seletores de **tipo**, **classe**, **ID**, **atributo**, entre outros. Cada tipo de seletor tem uma sintaxe específica e serve para diferentes propósitos, proporcionando uma forma eficiente de aplicar estilos a elementos específicos ou a grupos de elementos.

3.4 Pseudo-classes

As pseudo-classes são usadas para definir um estado especial de um elemento. Elas permitem estilizar elementos com base em informações dinâmicas, como o estado do *link* (**:hover, :visited**), a posição do elemento (**:first-child, :last-child**), entre outros. Por exemplo:

```
a:hover {
    color: red;
}
```

Nesse exemplo, a cor do texto dos links muda para **vermelho** quando o usuário **passa o cursor sobre eles**.

3.5 Combinação de Seletores

A combinação de seletores permite criar regras mais específicas, aplicando estilos a elementos baseados em

suas relações uns com os outros. Existem diferentes formas de combinar seletores, como o **seletor de descendente** (**div p**), **seletor de filho** (**div > p**), **seletor adjacente** (**h1 + p**), entre outros. Essas combinações permitem uma maior precisão ao aplicar estilos.

3.6 Prioridades

A prioridade (ou especificidade) determina qual regra será aplicada quando várias regras conflitantes se aplicam ao mesmo elemento. A especificidade é calculada com base na **origem do estilo** (en. *inline; pt.br. em linha*, interno ou externo), **tipo de seletor** (ID, classe, tipo), e **a ordem das regras no CSS**. Regras *inline* têm a maior prioridade, seguidas por seletores de ID, classes, e finalmente, seletores de tipo.

3.7 Box Model

O **modelo de caixa** (*Box Model*) é um conceito fundamental no **CSS** que descreve como os elementos são representados na página. Cada elemento é considerado uma caixa retangular composta por quatro partes: **conteúdo**, ***padding*** (preenchimento), **borda** e **margem**. Compreender o *Box Model* é essencial para controlar o layout e o espaçamento dos elementos na página.

3.8 Tipos de Unidades

Existem várias unidades de medida para definir tamanhos e espaçamentos, incluindo unidades **absolutas** (pixels, centímetros) e **relativas** (em, rem, porcentagem). Unidades absolutas são fixas e não mudam com o contex-

to, enquanto unidades relativas são baseadas em outras medidas, permitindo designs mais flexíveis e responsivos.

3.9 Posições Estáticas, Absolutas e Relativas

Permite controlar o posicionamento dos elementos na página através das propriedades de posição. A posição **estática** é o valor padrão, onde os elementos são posicionados conforme aparecem no fluxo do documento. A posição **relativa** permite deslocar um elemento em relação à sua posição original. A posição **absoluta** remove o elemento do fluxo do documento e o posiciona em relação ao seu elemento pai mais próximo com posição relativa ou absoluta.

3.10 Importações

A regra **@import** permite importar folhas de estilo externas dentro de uma folha de estilo **CSS**. Isso é útil para organizar e modularizar os diversos arquivos **CSS**, especialmente em projetos grandes. A sintaxe básica é:

```
@import url('styles.css');
```

Essa regra deve ser colocada no início da folha de estilo, antes de quaisquer outras regras **CSS**.

3.11 Frameworks

Frameworks **CSS**, como **Bootstrap**, **Foundation** e **Bulma**, são bibliotecas de estilos pré-desenvolvidas que facilitam a criação de layouts responsivos e modernos. Eles

fornecem uma variedade de componentes, como *grids (grades)*, botões, formulários, e outras interfaces de usuário, que podem ser facilmente integrados e personalizados. A utilização de frameworks pode acelerar o desenvolvimento e garantir consistência visual em um projeto.

MÓDULO IV

BOOTSTRAP

O Bootstrap é um framework de *front-end* criado pelo antigo *Twitter* que é essencial para o desenvolvimento de websites e aplicações web responsivas. Bootstrap oferece uma vasta coleção de componentes prontos para uso, como *grids*, botões, formulários e modais, que facilitam o design e garantem consistência visual. Este framework combina HTML, CSS e JavaScript para permitir que desenvolvedores construam layouts modernos de maneira eficiente.

Ao utilizar o Bootstrap, é necessário incluir os arquivos CSS e JavaScript do *framework*, que podem ser baixados ou incluídos via CDN. Ele disponibiliza um sistema de *grid* flexível e várias classes pré-definidas que facilitam a criação de designs responsivos adaptáveis a diversos tamanhos de tela. Além disso, o Bootstrap oferece uma variedade de modelos prontos que podem ser utilizados como ponto de partida para projetos rápidos e eficazes, permitindo personalizações conforme as necessidades do projeto.

4.1 Utilização

Para utilizar em um projeto, é necessário incluir seus arquivos **CSS** e **JavaScript**, que podem ser baixados ou referenciados via Rede de Distribuição de Conteúdo (CDN, Content Delivery Network). Com o **Bootstrap**, os desenvolvedores podem aplicar classes pré-definidas aos elementos **HTML** para estilizar componentes e layout da página. A biblioteca oferece um sistema de *grid* flexível, que facilita a criação de layouts responsivos, adaptando-se automaticamente a diferentes tamanhos de tela. Além disso, o Bootstrap inclui diversos *plugins* **JavaScript** que adicionam funcionalidades interativas aos componentes.

4.2 Modelos prontos

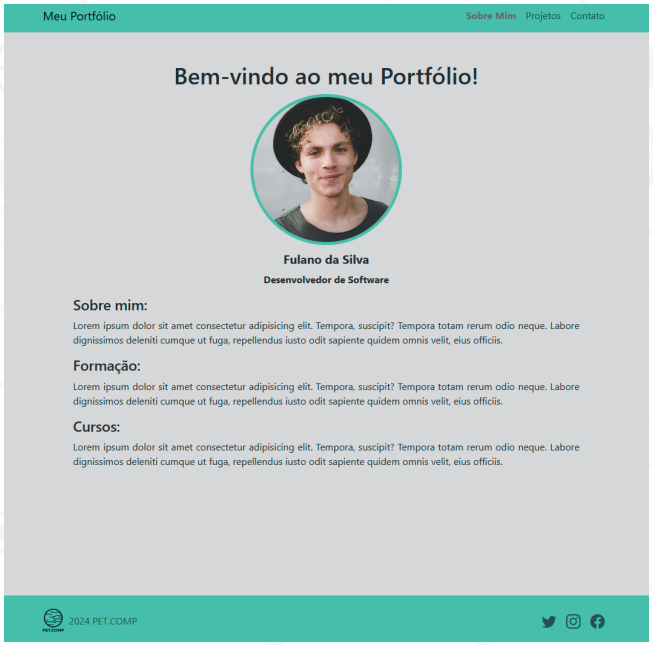
Fornece uma variedade de modelos prontos (templates) que servem como ponto de partida para o desenvolvimento de sites e aplicações. Esses templates cobrem diversas necessidades, desde páginas iniciais simples até *dashboards (painéis)* complexos e sites de comércio. Utilizando esses modelos, os desenvolvedores podem economizar tempo, garantindo ao mesmo tempo que os designs sigam as melhores práticas de usabilidade e responsividade. Esses modelos podem ser personalizados facilmente, permitindo a adaptação às especificidades de cada projeto.

Você pode instalar o Bootstrap e conferir a sua documentação a partir do site: getbootstrap.com

MÓDULO V

ESTUDO DE CASO

Coloque em prática o conteúdo passado nos módulos anteriores com um estudo de caso concreto. Este módulo proporciona a oportunidade de desenvolver um website para exibir o seu portfólio. Acompanhe um passo a passo para a criação de cada elemento do site com html e css, aplicando técnicas de design e navegação. Finalize o módulo publicando seu site e veja como os conceitos aprendidos se transformam em uma presença online efetiva e atraente.

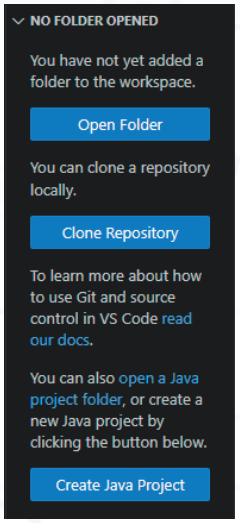


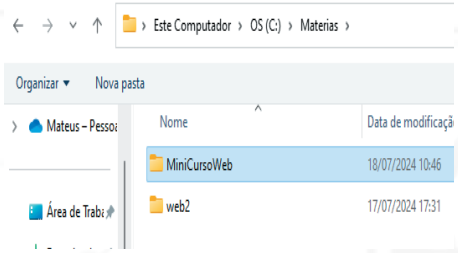
ESTUDO DE CASO

Após abrir a IDE escolhida para desenvolver o site (a escolhida para o estudo de caso foi o Visual Studio Code), siga os passo a seguir:

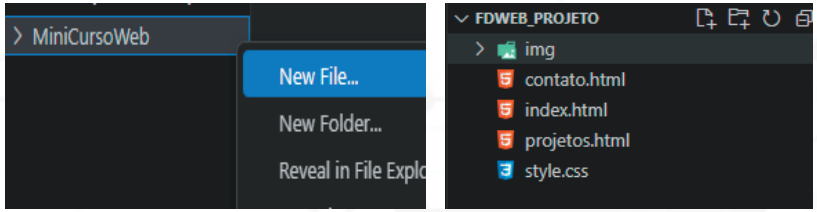
4.1 Preparando o VS Code:

Abra o VS Code e clique em '*Open Folder*'. Selecione a pasta onde deseja armazenar os arquivos de código do site.





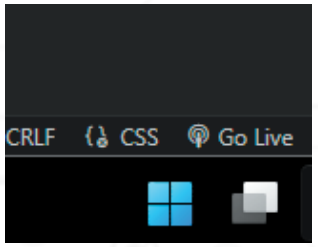
Na pasta selecionada, clique com o botão direito do mouse e selecione *'New File'*. Repita esse processo para criar quatro arquivos com os seguintes nomes: *'index.html', projetos.html', 'contato.html'* e *'style.css'*. Após criar os arquivos, clique em '*New Folder*' e nomeie-a como img.



Após esses passos iniciais, certifique que o *auto save* está ativo no VS Code indo em '*files -> auto save*'. Caso não esteja habilitado, habilite-o.

Com '*Ctrl + Shift + X*' abra a aba de extensões e procure por *Live Server*. O live server, como explicado em módulos anteriores é uma extensão que inicia um servidor local de desenvolvimento que atualiza automaticamente as modificações dos arquivos do projeto.

No canto inferior direito do VS Code, estará a opção de iniciar o servidor local.



Feito esses procedimentos, você estará pronto para criar sua página web.

- **HEAD:**
  No arquivo `index.html`, digite uma exclamação e aperte *Enter* para gerar uma predefinição do html. Dentro desse código, substitua "Document" pelo

nome que deseja que apareça em seu navegador.

Navegue até o site do bootstrap e procure pela opção “Include via CDN”. Copie as duas tags e insira abaixo da tag ‘`<title>`’. Além disso, Adicione o link para o Bootstrap Icons, um conjunto de ícones prontos para serem usados.

Após a inserção das tags do bootstrap, adicione a tag `` `<link rel=”stylesheet” href=”style.css”>` `` para que possa ser carregado o arquivo CSS do projeto. É importante que a tag do css fique após as tags do bootstrap para que funcione corretamente.

Ao final, o seu head estará semelhante a esse:

```
<!DOCTYPE html>
<html lang=”pt-br”>
<head>
    <meta charset=”UTF-8”>
    <meta name=”viewport” content=”width=device-width, ini-
tial-scale=1.0”>
    <title>Sobre Mim</title>

    <!--Bootstrap-->
   <link href=”https://cdn.jsdelivr.net/npm/bootstrap@5.3.3/
dist/css/bootstrap.min.css” rel=”stylesheet” integrity=”-
sha384-QWTKZyjpPEjISv5WaRU9OFeRpok6YctnYmDr5pNlyT2bRjXh0JM-
hjY6hW+ALEwIH” crossorigin=”anonymous”>
  <script src=”https://cdn.jsdelivr.net/npm/bootstrap@5.3.3/
dist/js/bootstrap.bundle.min.js” integrity=”sha384-YvpcrY-
f0tY3lHB60NNkmXc5s9fDVZLESaAA55NDzOxhy9GkcIdslK1eN7N6jIeHz”
crossorigin=”anonymous”></script>

    <!--Bootstrap Icons-->
     <link rel=”stylesheet” href=”https://cdn.jsdelivr.net/
npm/bootstrap-icons@1.11.3/font/bootstrap-icons.min.css”>

    <!--CSS-->
    <link rel=”stylesheet” href=”style.css”>
</head>
```


4.2 Página principal - index (Sobre Mim)

Para estruturar um site simples e responsivo com HTML, CSS e Bootstrap, você pode começar definindo um esqueleto básico da página que inclui seções fundamentais: o corpo (`body`), o conteúdo principal (`main`) e o rodapé (`footer`). Utilizar o Bootstrap facilita o desenvolvimento de layouts responsivos e estilos atraentes sem a necessidade de criar muitas classes personalizadas. Neste guia, vamos explorar o passo a passo para criar a estrutura inicial do corpo da página e detalhar as funções e significados das classes utilizadas.

- **Estrutura Inicial:**
  A estrutura de `<body>` contém três elementos principais: o cabeçalho (`header`), o conteúdo principal (`main`) e o rodapé (`footer`). Essa organização não apenas melhora a legibilidade do código como também contribui para um design coerente e responsivo. Vamos entender cada parte dessa estrutura:

  1. **Tag `<body>` e Classes Bootstrap:** A tag `<body>` deve conter classes que ajudam a definir o layout global da página:

     - `d-flex`: Ativa o Flexbox no `body`, permitindo que seus elementos internos (o `header`, `main`, e `footer`) sejam organizados como itens flexíveis, que podem ser dimensionados e posicionados com facilidade.

     - `flex-column`: Define que o layout do `body` será em coluna, posicionando os elementos

de cima para baixo, ou seja, com o `header` no topo, seguido pelo `main` e, por fim, o `footer`.

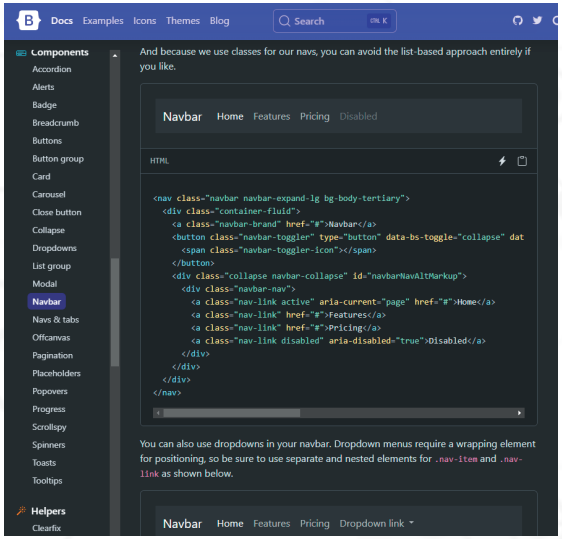
- `min-vh-100`: Garante que a altura mínima do `body` ocupe 100% da altura da viewport, o que ajuda a manter o `footer` na parte inferior da tela, mesmo que o conteúdo seja curto.

Com essa estrutura, o site permanece responsivo e o layout adapta-se automaticamente ao tamanho da tela do usuário.

2. **Cabeçalho (`<header>`):** A seção `<header>` é onde geralmente colocamos a barra de navegação, o logotipo ou outras informações iniciais. Este estudo de caso inclui um menu de navegação com links para as seções “Sobre Mim”, “Projetos” e “Contato”. Vamos explorar o papel de cada linha e as classes Bootstrap utilizadas para configurar essa navegação.

   - **Estrutura Básica do `<header>` e `<nav>`**

Começamos criando a estrutura do cabeçalho e da barra de navegação. Você pode verificar a documentação e exemplos do bootstrap e achar um exemplo que queira utilizar como base:



Pegamos esse `header` de exemplo e fizemos algumas alterações. Ao final o `header` ficará dessa forma:

```
<body class="d-flex flex-column min-vh-100">
    <header>
        <nav class="navbar navbar-expand-lg">
            <div class="container-fluid">
             <a class="navbar-brand" href="#">Meu Portfólio</a>
                    <button class="navbar-toggler" type="but-
                    ton" data-bs-toggle="collapse" da-
                    ta-bs-target="#navbarNavAltMarkup"
                    aria-controls="navbarNavAltMarkup"
                    aria-expanded="false" aria-label="Toggle
                    navigation">
                     <span class="navbar-toggler-icon"></span>
                     </button>
                <div class="collapse navbar-collapse" id="nav
                   baNavAltMarkup">
                    <div class="navbar-nav">
                          <a class="nav-link active" aria-cur
                          rent="page" href="#">Sobre Mim</a>
                          <a class="nav-link" href="projetos.
                          html">Projetos</a>
                          <a class="nav-link" href="contato.
```

```
                    html">Contato</a>
                </div>
            </div>
            </div>
        </nav>
    </header>
```

3. **Conteúdo Principal (`<main>`)**: O `<main>` é onde o conteúdo central da página é exibido. Aqui, é adicionado um recurso importante de Bootstrap:

    - `flex-grow-1`: Essa classe faz com que o `main` ocupe todo o espaço restante na tela verticalmente. Como o `body` possui uma altura mínima de 100% da viewport, isso faz com que o conteúdo principal expanda-se, empurrando o `footer` para o final da página, mesmo em dispositivos maiores.

Dessa forma, o `main` adapta-se à quantidade de conteúdo e mantém o layout equilibrado.

Dentro do `main` adicionaremos uma nova `div` com a classe '`container mt-5`', onde:

- `container`: Centraliza o conteúdo e limita sua largura, garantindo que ele fique bem alinhado e ajustado na tela.

- `mt-5`: Adiciona uma margem superior (`margin-top`) ao contêiner para afastá-lo do cabeçalho.

Dentro do `<div class="container mt-5">`, começamos com o título principal:

```
<main class="flex-grow-1">


  <!-- Conteúdo Principal -->
    <div class="container mt-5">
        <h1 class="text-center">Bem-vindo ao meu Port
            fólio!</h1>

    </div>
</main>
```

- `text-center`: Centraliza o texto horizontalmente na tela.

Em seguida, criamos uma seção para exibir a foto, o nome e a profissão:

```
<div class="text-center">
    <img src="img/sua-foto.jpg" alt="MinhaFoto"
        class="rounded-circle img-fluid">
    <h2><strong class="nome">Fulano da Silva
        </strong></h2>
    <p><strong>Desenvolvedor de Software</strong></p>
</div>
```

- `text-center:` Centraliza todo o conteúdo dentro do contêiner.

- `img-fluid`: Torna a imagem responsiva, ajustando seu tamanho conforme o tamanho da tela.

- `rounded-circle:` Aplica bordas arredondadas à imagem, transformando-a em um círculo.

- `<h2><strong class="nome">Fulano da Silva</strong></h2>`: Exibe o nome em negrito, e a classe `nome` pode ser estilizada no CSS para personalizar o tamanho e o estilo do nome.

- `<p><strong>Desenvolvedor de Software</strong></p>`: Exibe a profissão logo abaixo do nome, também em negrito para destaque.

Essa estrutura fornece uma apresentação visual com uma imagem circular para um toque profissional. Mais para frente faremos modificações com CSS para melhorarmos o design do portfólio.

A próximas seções detalham um pouco sobre você:

```
<section>
        <h4>Sobre mim:</h4>
        <p>Lorem ipsum dolor sit amet consectetur ing elit.
              Tempora, suscipit?Tempora totam rerum odio
              neque. Labore dignissimos deleniti cum que
              ut fuga, repellendus iusto odit sapiente
              quidem omnis velit, eius officiis.
        </p>
</section>

<section>
        <h4>Formação:</h4>
        <p>Lorem ipsum dolor sit amet consectetur adipis
              ic                    ing elit. Tempora,
              suscipit? Tempora totam rerum odio neque.
              Labore dignissimos deleniti cumque ut fuga,
              repellendus iusto odit sapiente quidem omnis
              velit, eius officiis.
        </p>
</section>

<section>
        <h4>Cursos:</h4>
```



```
        <p>Lorem ipsum dolor sit amet consectetur adipis
              ic                    ing elit. Tempora,
              suscipit? Tempora totam rerum odio neque.
              Labore dignissimos deleniti cumque ut fuga,
              repellendus iusto odit sapiente quidem omnis
              velit, eius officiis.
        </p>
</section>
```

Ao final o seu `main` estará semelhante a esse:

```
<main class="flex-grow-1">
      <!-- Conteúdo Principal -->
        <div class="container mt-5">
            <h1 class="text-center">Bem-vindo ao meu Port
               fólio!</h1>
            <div class="text-center">
                <img src="img/sua-foto.jpg" alt="Minha
                      Foto" class="rounded-circle img-flu
                      id">
                <h2><strong class="nome">Fulano da Silva</
                strong></h2>
                <p><strong>Desenvolvedor de Software
                </strong></p>
            </div>

            <section>
              <h4>Sobre mim:</h4>
              <p>Lorem ipsum dolor sit amet consectetur
              ing elit.
              Tempora, suscipit?Tempora totam rerum odio
              neque. Labore dignissimos deleniti cum que
              ut fuga, repellendus iusto odit sapiente
              quidem omnis velit, eius officiis.
              </p>
            </section>

            <section>
              <h4>Formação:</h4>
              <p>Lorem ipsum dolor sit amet consectetur
              ing elit.
              Tempora, suscipit?Tempora totam rerum odio
              neque. Labore dignissimos deleniti cum que
              ut fuga, repellendus iusto odit sapiente
              quidem omnis velit, eius officiis.
              </p>
            </section>
```

```
        <section>
                <h4>Cursos:</h4>
                <p>Lorem ipsum dolor sit amet consectetur
                ing elit.
                Tempora, suscipit?Tempora totam rerum odio
                neque. Labore dignissimos deleniti cum que
                ut fuga, repellendus iusto odit sapiente
                quidem omnis velit, eius officiis.
                </p>
        </section>
    <div>
</main>
```

4. **Rodapé (`<footer>`):** O `<footer>` é a última seção da página e normalmente contém informações de contato, links para redes sociais, ou direitos autorais. Utilizamos algumas classes do Bootstrap para garantir que o `footer` também seja responsivo e organizado:

    - `d-flex`: Ativa o Flexbox, permitindo um controle detalhado sobre a disposição dos elementos internos.
    - `flex-wrap`: Permite que o conteúdo do `footer` quebre para uma nova linha em telas pequenas, garantindo que ele não ultrapasse a largura da tela.
    - `justify-content-between`: Posiciona os itens do `footer` de forma espaçada, com o primeiro item à esquerda e o último à direita.
    - `align-items-center`: Centraliza os itens do `footer` verticalmente, deixando-os alinhados ao meio do rodapé.

    - `py-3`: Adiciona espaçamento vertical (`padding-y`) de 1rem (16px geralmente), deixando o rodapé com uma margem superior e inferior agradável.

Essas classes garantem que o `footer` tenha uma apresentação consistente, centralizada e espaçada adequadamente.

Dentro do `<footer>`, usamos um contêiner interno para centralizar e organizar os elementos. A primeira parte do rodapé inclui algum logotipo (ou ícone) e o nome que deseja inserir

```
<div class="col-md-4 d-flex align-items-center">
      <a href="/" class="mb-3 me-2 mb-md-0 text-muted
text-decoration-none lh-1">
            <img class="bi" src="img/Component 3 (4).
                  png" alt="Logo PET.COMP" width="35"
                  height="40">
      </a>
      <span class="mb-3 mb-md-0 text-muted">2024 PET.
         COMP</span>
</div>
```

- `col-md-4`: Em telas médias ou maiores, esta classe limita a largura da coluna a um quarto da tela, organizando melhor o conteúdo.
- `d-flex` e `align-items-center`: Organizam o ícone e o nome em linha e os centralizam verticalmente.
- `a` (link): link para encaminhar para uma certa página.

- `text-muted:` Aplica uma cor cinza ao link, combinando com o design do rodapé.

- `text-decoration-none`: Remove o sublinhado do link.

- `lh-1`: Define uma linha mais estreita, ajustando o espaçamento entre o ícone e o texto.

No lado direito do rodapé, uma lista de ícones de redes sociais é exibida:

```
<ul class="nav col-md-4 justify-content-end list-
unstyled d-flex m-2">
    <li clasAs="ms-3"><a class="text-muted" href="#">
      <i class="bi bi-twitter"
         style="font-size: 1.5rem;">
       </i></a>
    </li>

    <li class="ms-3"><a class="text-muted" href="#">
       <i class="bi bi-instagram"
         style="font-size: 1.5rem;">
       </i></a>
    </li>

    <li class="ms-3"><a class="text-muted" href="#">
       <i class="bi bi-facebook"
         style="font-size: 1.5rem;">
       </i></a>
    </li>
</ul>
```

- `nav`: Estiliza o contêiner como uma barra de navegação.

- `col-md-4`: Limita a largura em telas médias e maiores, ajudando na organização do conteúdo.

- `justify-content-end`: Alinha a lista de ícones à direita do rodapé.

- `list-unstyled`: Remove a formatação padrão de listas, como marcadores.

- `d-flex`: Alinha os ícones horizontalmente em linha.

- `m-2`: Adiciona uma margem geral aos ícones para espaçamento.

- Para os ícones de redes sociais:

- `ms-3`: Adiciona margem à esquerda de cada ícone, espaçando-os de maneira uniforme.

- `text-muted`: Aplica uma cor cinza suave, harmonizando com o rodapé.

- `font-size: 1.5rem;`: Aumenta o tamanho dos ícones para 1.5 vezes o tamanho da fonte base, melhorando a visibilidade e atraindo a atenção do usuário.

Esses ícones, usando Bootstrap Icons, permitem acesso rápido às redes sociais e mantêm um visual limpo e profissional.

4.3 Página de Projetos

Na criação da página "Projetos" do portfólio, o objetivo é apresentar uma lista organizada e atraente dos trabalhos realizados. Utilizando HTML, CSS e Bootstrap, você poderá exibir os projetos de forma responsiva e visualmente agradável, permitindo que visitantes explorem seu portfólio com facilidade. Cada projeto pode ser representado com uma breve descrição, imagem, e links para mais detalhes ou para o próprio site do projeto.

- Copie e cole a mesma estrutura inicial feita para a pagina index

Altere no header o *nav* ativo:

```
<header>

    <nav class="navbar navbar-expand-lg">
        <div class="container-fluid">
     <a class="navbar-brand" href="#">Meu Portfólio</a>
            <button class="navbar-toggler" type="but-
            ton" data-bs-toggle="collapse" da-
            ta-bs-target="#navbarNavAltMarkup"
            aria-controls="navbarNavAltMarkup"
            aria-expanded="false" aria-label="Toggle
            navigation">
             <span class="navbar-toggler-icon"></span>
             </button>
        <div class="collapse navbar-collapse" id="nav
           baNavAltMarkup">
            <div class="navbar-nav">
                 <a class="nav-link" aria-cur
           rent="page" href="#">Sobre Mim</a>
                 <a class="nav-link active"
                 href="projetos.html">Projetos</a>
                 <a class="nav-link" href="contato.
                 html">Contato</a>
            </div>
        </div>
        </div>
    </nav>
</header>
```


Mantenha a mesma estruturação do main e footer:

```
<main class="flex-grow-1">
      <!-- Conteúdo Principal -->
        <div class="container mt-5">
            <h1 class="text-center">Bem-vindo ao meu Port
               fólio!</h1>
            <div class="text-center">
                <img src="img/sua-foto.jpg" alt="Minha
                      Foto" class="rounded-circle img-flu
                      id">
                <h2><strong class="nome">Fulano da Silva</
               strong></h2>
                <p><strong>Desenvolvedor de Software
               </strong></p>
            </div>
        </div>
</main>
<footer class="d-flex flex-wrap justify-content-between
       align-items-center py-3">
    <div class="container d-flex align-items-center justi
       fy-content-between">

        <div class="col-md-4 d-flex align-items-center">
            <a href="/" class="mb-3 me-2 mb-md-0 text-muted
               text-decoration-none lh-1">
                <img class="bi" src="img/Component 3 (4).
                      png" alt="Logo PET.COMP" width="35"
                      height="40">
            </a>
            <span class="mb-3 mb-md-0 text-muted">2024 PET.
               COMP</span>
        </div>
    </div>
</footer>
```

Para essa página, iremos utilizar *card* do bootstrap . Essa estrutura permite organizar projetos visualmente em uma grade de cards responsiva, que exibe imagem, título, descrição e um botão "Ver Mais" para cada projeto. Futuramente você pode referenciar esses card com seus projetos do *github* por exemplo.

Comece com um título para a seção, que serve para introduzir a área de projetos:

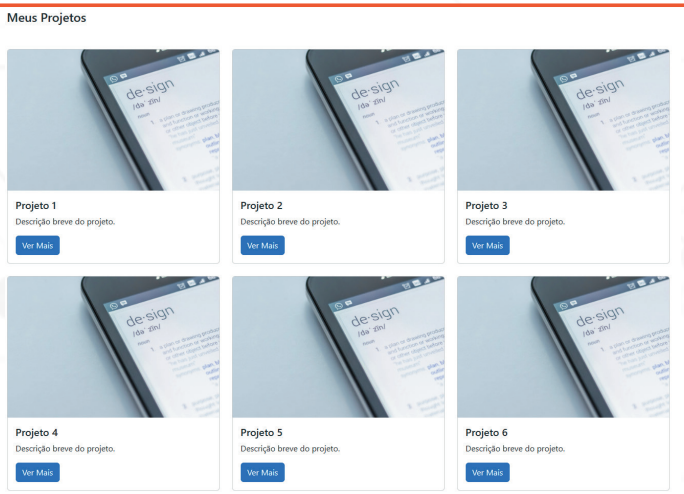
```
<h4>Meus Projetos</h4>
```

Em seguida, crie um contêiner de `row` para abrigar os cards de projeto e garantir uma estrutura organizada e alinhada. A classe `row` do Bootstrap permite criar uma grade flexível.

Procure nos exemplos do bootstrap algum card de sua preferência. Cada card de projeto estará dentro de uma coluna (`col-md-4`), que organiza os cards em três colunas para dispositivos médios e maiores. Dentro de cada coluna, crie o card usando a classe `card` do Bootstrap ou utilize um dos exemplos prontos que você pode encontar no site do bootstrap. Ao final, o card ficará da seguinte forma:

```
<div class="col-md-4 mt-4">
    <div class="card">
            <img class="card-img-top" src="img/pro
                jeto1.jpg" alt="Projeto 1">
            <div class="card-body">
               <h5 class="card-title">Projeto 1</h5>
               <p class="card-text">Descrição breve
                       do projeto.</p>
               <a href="#" class="btn btn-prima
                       ry">Ver Mais</a>
            </div>
    </div>
</div>
```

Repita a estrutura acima para cada projeto. No nosso estudo de caso, fizemos 6 cards.



4.4 Página de Contato

A página de "Contato" é essencial para fornecer aos visitantes um meio fácil e direto de se comunicar com você. Nesta seção, iremos construir uma página de contato onde você poderá incluir informações como e-mail, telefone, links para redes sociais e um formulário de contato simples para o envio de mensagens.

Assim como na página de projetos, mantenha a mesma estrutura para a pagina de contatos, lembrando de alterar a classe do *nav* ativo do *header* para 'Contato'.

No *main*, após o container que descreve sobre você, adicionaremos alguns ícones do bootstrap de redes sociais centralizados no topo da seção:

```
<ul class="nav justify-content-center mt-4">
                <li class="ms-3"><a class="text-mut
```

```
                ed" href="#">
                    <i class="bi bi-github" style="
                        font-size: 2.5rem;"></i></a>
            </li>
            <li class="ms-3"><a class="text-muted"
                href="#">
                    <i class="bi bi-instagram"
                        style="font-size: 2.5rem;">
                    </i></a>
            </li>
            <li class="ms-3"><a class="text-muted"
                href="#">
                    <i class="bi bi-facebook"
                        style="font-size: 2.5rem;">
                    </i></a>
            </li>
</ul>
```

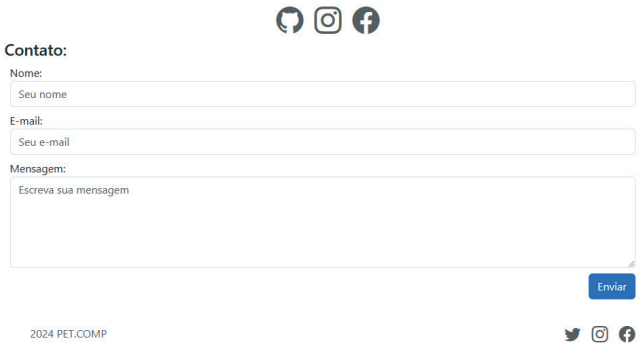
- `nav`: Classe que estiliza a lista como uma barra de navegação.
- `justify-content-center`: Centraliza a lista horizontalmente.
- `mt-4`: Adiciona margem superior, separando os ícones do conteúdo acima.
- `ms-3`: Classe que aplica margem à esquerda de cada ícone, criando um espaçamento uniforme.
- `text-muted`: Aplica uma cor cinza suave aos ícones, harmonizando com o layout.
- `font-size: 2.5rem;`: Ajusta o tamanho dos ícones para 2.5 vezes o tamanho da fonte base, destacando-os visualmente.

Em seguida, adicione um título para a seção de contato seguido do formulário de contato. O formulário é composto por três campos: Nome, E-mail e Mensagem, e um botão de envio. Cada campo é colocado dentro de uma `form-group` para garantir uma organização uniforme.

```
<h4>Contato</h4>

    <form>
        <div class="form-group m-2">
            <label for="nome">Nome:</label>
            <input type="text" class="form-control"
             id="nome" placeholder="Seu nome">
        </div>
        <div class="form-group m-2">
            <label for="email">E-mail:</label>
            <input type="email" class="form-control"
             id="email" placeholder="Seu e-mail">
        </div>
        <div class="form-group m-2">
            <label for="mensagem">Mensagem:</label>
            <textarea class="form-control" id="mensagem"
             rows="5" placeholder="Escreva sua mensagem">
             </textarea>
        </div>
        <div class="form-group text-end m-2">
             <button type="submit" class="btn btn-prima
             ry">Enviar
             </button>
        </div>
    </form>
```

Com esses elementos, você cria uma seção de contato completa, que inclui links para redes sociais e um formulário de contato funcional. A estrutura é organizada e responsiva, permitindo uma navegação clara e uma interação facilitada para os visitantes que desejam se conectar com você. A combinação das classes Bootstrap ajuda a manter o layout limpo, com espaçamentos e alinhamentos apropriados. Após o aprofundamento em seus estudos em programação, você poderá tornar esse formulário funcional.



4.5 Estilos adicionais com CSS

Se você observar, nosso portfólio apresenta alguns problemas como falta de diferenciação do que é header, main e footer, além da nossa imagem de apresentção estar ocupando muito espaço na página. Esses estilos adicionais visam aprimorar a aparência e a experiência de navegação da página, aplicando cores de fundo, espaçamentos, alinhamentos e efeitos visuais que contribuem para um layout coeso e profissional.

1. **Estilização do Corpo da Página** (`body`)

   - `background-color: #dfdddd;`: Define uma cor de fundo cinza claro para o corpo da página. Esse tom neutro serve como pano de fundo e ajuda a destacar os elementos do conteúdo em primeiro plano.

2. **Estilização do Cabeçalho** (`header`)

   - `background-color: #45c4B0;`: Aplica uma cor de fundo azul esverdeado ao cabeçalho, destacando-o visualmente do restante da página e criando um ponto de referência claro para os visitantes.

3. **Estilização da Imagem de Perfil** (`.text-center img`)

   - `width: 250px;` e `height: 250px;`: Define dimensões fixas de 250 pixels de largura e altura para a imagem de perfil, garantindo que ela mantenha um tamanho uniforme.

   - `border-radius: 50%;`: Aplica bordas arredondadas com 50% de raio, transformando a imagem em um círculo, um estilo moderno e elegante.

   - `border: 5px solid #45c4B0;`: Adiciona uma borda de 5 pixels na cor do cabeçalho, criando uma moldura que harmoniza com o layout.

   - `object-fit: cover;`: Garante que a imagem ocupe todo o espaço designado, centralizando o foco e evitando distorções.

4. **Estilização do Link Ativo no Menu de Navegação** (`.nav-link.active`)

   - `color: rgb(104, 104, 104) !important;`: Define uma cor cinza escuro para o link ativo, indicando visualmente a seção atual e propor-

cionando contraste.

- `font-weight: bold;`: Aplica negrito ao link ativo, ajudando a destacá-lo dos outros links.

5. **Estilização do Nome (`.nome`)**

   - `font-size: 20px;`: Aumenta o tamanho da fonte para 20 pixels, destacando o nome como um elemento de identidade e introdução.

6. **Margem nas Seções do Conteúdo Principal (`main section`)**

   - `margin-left: 50px;` e `margin-right: 50px;`: Adiciona uma margem de 50 pixels nas laterais de cada seção dentro do `<main>`, criando espaçamento e melhorando a legibilidade.

7. **Justificação dos Parágrafos nas Seções (`main section p`)**

   - `text-align: justify;`: Justifica o texto, criando um alinhamento simétrico nas margens esquerda e direita, o que melhora a apresentação visual e facilita a leitura de blocos maiores de texto.

8. **Efeitos de Transição para Imagens e Cards (`.text-center img, .card`)**

   - `transition: transform 0.3s ease;`: Configura uma transição suave para as transformações aplicadas, como o efeito de elevação ao passar o mouse. O tempo de 0.3 segundos e o efeito `ease` garantem uma animação agradável e gradual.

9. **Efeito Hover de Elevação para Imagens e Cards (`.text-center img:hover, .card:hover`)**

   - `transform: translateY(-5px);`: Move o elemento 5 pixels para cima quando o usuário passa o mouse sobre ele, simulando um efeito de "flutuação" que indica interatividade e atrai atenção.

10. **Estilização do Rodapé (`footer`)**

    - `background-color: #45c4B0;`: Aplica a mesma cor de fundo do cabeçalho ao rodapé, criando consistência visual entre as áreas superior e inferior da página.

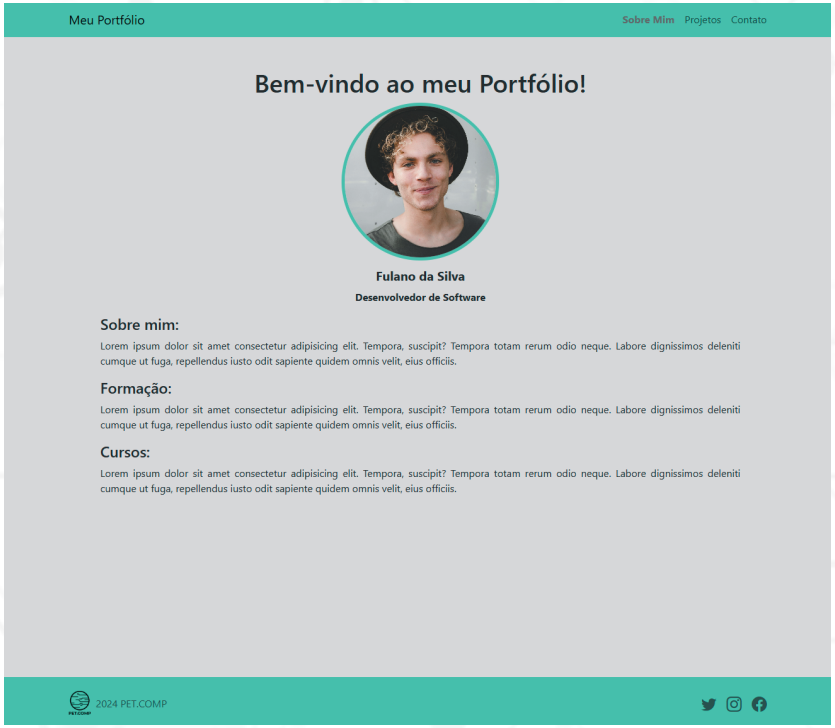
11. **Estilo Responsivo para a Navegação (`@media (min-width: 900px)`)**

    - `@media (min-width: 900px)`: Define uma regra de estilo específica para telas com largura mínima de 900 pixels, ajustando a navegação para dispositivos maiores.

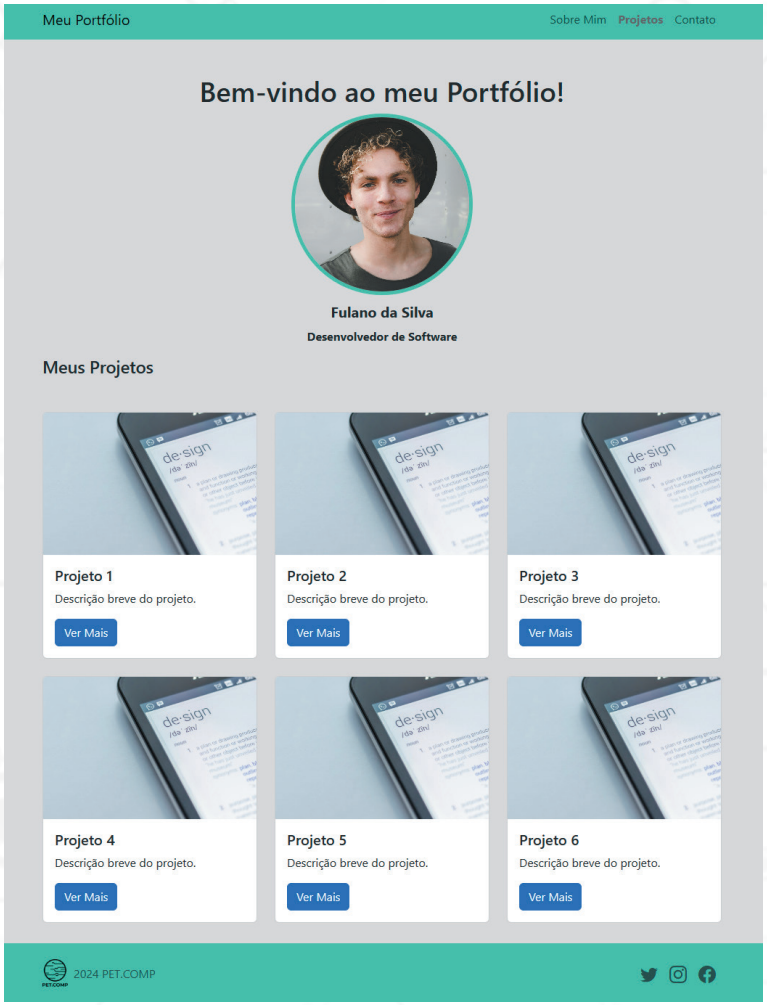
    - `flex-grow: 0;`: Impede que o contêiner de navegação se expanda em telas grandes, mantendo um layout compacto e organizado.

Ao final, nosso páginas do portfólio ficam assim:

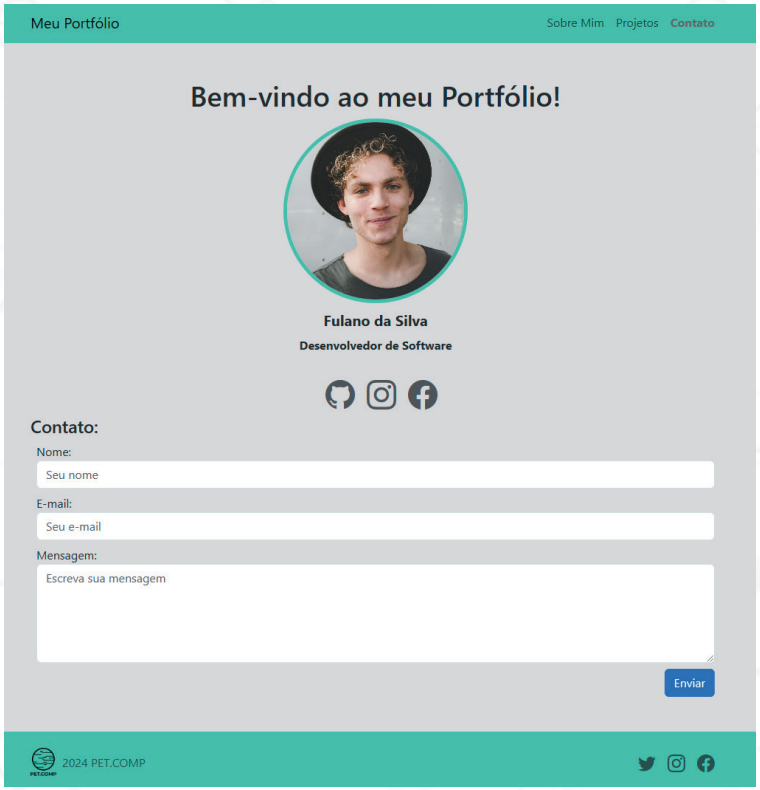
Sobre Mim (index):



Projetos:



Contato:



Você pode acessar o código fonte dessa caso de uso pelo link: XXXX

Glossário

Página 9

Design gráfico: A arte de combinar textos e imagens de maneira criativa para produzir anúncios, revistas, livros e outros materiais visuais.

Posts: Publicações feitas em redes sociais ou blogs, que podem incluir textos, imagens ou vídeos para engajar um público específico.

Slides: Páginas individuais de uma apresentação digital, usadas para expor ideias de forma sequencial e visual.

Posters: Peças gráficas de grande formato destinadas a serem fixadas em superfícies, usadas para anunciar eventos ou comunicar mensagens visuais.

Flyers: Folhetos leves e simples, usados geralmente para promoção rápida de eventos, produtos ou serviços.

Templates: Modelos pré-desenhados que servem como ponto de partida para a criação de documentos, apresentações, sites, entre outros.

Ícones: Representações gráficas simplificadas usadas para facilitar a compreensão visual de funções ou categorias.

Fontes: Variações de estilo de texto, como tipos, tamanhos e formatos, que ajudam a definir a apresentação e o tom de escrita.

Feedback: Resposta ou informação dada por usuários sobre sua experiência com um produto, serviço ou desempenho.

Aplicação web: Programa que é executado em um servidor web e pode ser acessado através de um navegador de internet.

iOS: Sistema operacional móvel desenvolvido pela Apple, usado em iPhones e iPads.

Android: Sistema operacional baseado em Linux, desenvolvido pelo Google, usado em uma variedade de dispositivos móveis.

Página 10

Pop-up: Janela que aparece automaticamente na tela dentro de uma interface digital, geralmente oferecendo informações adicionais, alertas ou ações interativas.

Página 11

Elementos: Componentes visuais como imagens, gráficos, formas e textos que podem ser adicionados e manipulados em um design.

Upload: Ato de enviar dados ou arquivos de um dispositivo local para um servidor ou sistema na nuvem.

Página 13:

Download: Processo de baixar dados, arquivos, ou programas de um servidor ou internet para o computador ou dispositivo local do usuário.

GOSTOU DESSE CONTEÚDO?

ACESSE NOSSAS REDES E FIQUE POR DENTRO DO TRABALHO DO PET.COMP!

petcomp.cefetmg.br

pet.comp.cefetmg@gmail.com

@pet.comp.cefetmg

PET.COMP

REFERÊNCIAS

TRANSFERO SWISS. O que é Web 3.0? Panorama Crypto, 2024. Disponível em: https://panoramacrypto.transfero.com/o-que-e-web-3-0/. Acesso em: 18 jul. 2024.

GROWUNDER. O que é HTTPS, como funciona e quais as vantagens? Growunder, 2024. Disponível em: https://www.growunder.com/pt/blog/dicas/128-o-que-e-https-como-funciona-e-quais-as-vantagens. Acesso em: 18 jul. 2024.

1TRAINING. HTML, CSS & JavaScript Web Development. 1Training, 2024. Disponível em: https://www.1training.org/blog/html-css-javascript-web-development/. Acesso em: 18 jul. 2024.

MARTINS, S. Semântica em HTML5. Medium, 2024. Disponível em: https://medium.com/reprogramabr/semanticahtml5-5252b4937f0a. Acesso em: 18 jul. 2024.


PET.COMP