

INTENSIVÃO DE JAVASCRIPT

Apostila Completa Aula 1

Guia passo a passo para construir seu próprio site
de E-Commerce a partir do zero!



Parte 1

Instalando as ferramentas



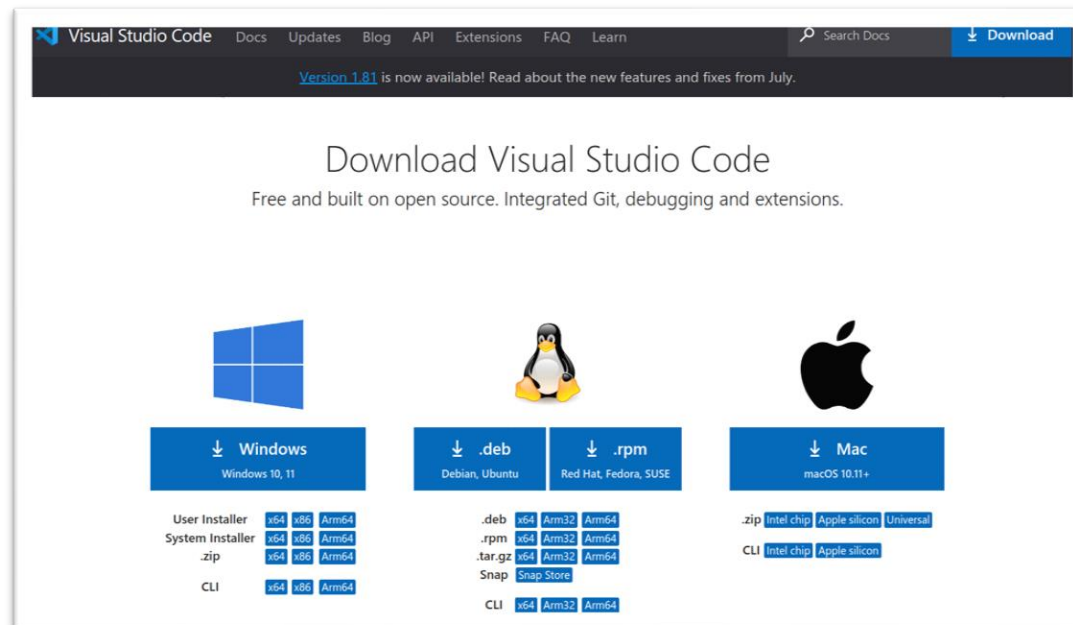


Instalando as ferramentas – VS CODE

Caso você ainda não tenha o Visual Studio Code instalado, basta seguir os procedimentos abaixo. A instalação do VS Code é totalmente gratuita, e você pode usá-lo em seu computador sem precisar pagar nada. O link para fazer o download do programa é mostrado abaixo:

<https://code.visualstudio.com/>

- 1 - Baixe o arquivo de instalação correspondente ao seu sistema operacional (Windows, MacOS ou Linux).
- 2 - Execute o arquivo de instalação e siga as instruções na tela. Em geral, é só continuar clicando em "Próximo".
- 3 - No Windows, durante a instalação, marque a opção "Add to path" para adicionar o VS Code às suas variáveis de ambiente.



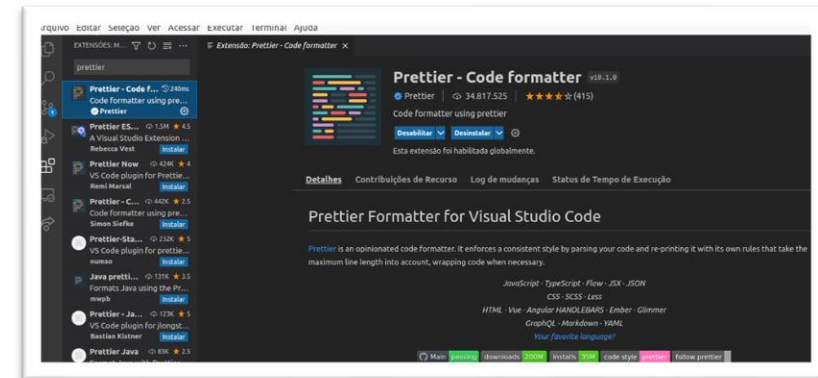


Instalando as ferramentas – VS CODE - EXTENSÕES

As extensões no Visual Studio Code são pequenos programas que adicionam recursos extras ao editor de código. Elas são como "plugins" que podem ser instalados para personalizar e estender as funcionalidades do VS Code.

A extensão Prettier adiciona a capacidade de formatar automaticamente o código, tornando-o mais legível e organizado.

Para instalar a extensão Prettier no Visual Studio Code:



- Clique no ícone de extensões no menu lateral esquerdo (ou use o atalho "Ctrl+Shift+X").
- Na barra de pesquisa, digite "Prettier".
- A extensão "Prettier - Code Formatter" deve aparecer nos resultados da pesquisa. Clique no botão "Instalar" ao lado dela.
- Após a instalação, você pode configurar o Prettier como o formatador padrão para o seu projeto. Para fazer isso, abra as configurações do Visual Studio Code (menu "File" > "Preferences" > "Settings" ou use o atalho "Ctrl+,").
- Na barra de pesquisa das configurações, digite "Default Formatter" e selecione a opção "Editor: Default Formatter".
- No campo de valor, digite "esbenp.prettier-vscode" e pressione Enter para salvar as alterações.
- Agora, o Prettier está instalado e configurado como o formatador padrão no Visual Studio Code. Você pode usar o comando "Format Document" (menu "Edit" > "Format Document" ou atalho "Shift+Alt+F") para formatar o código.



Instalando as ferramentas – VS CODE - EXTENSÕES

Já a extensão ES6 String HTML adiciona suporte para destacar a sintaxe do HTML dentro de strings de várias linhas no código JavaScript.

Para instalar a extensão ES6 String HTML no Visual Studio Code:

- Clique no ícone de extensões no menu lateral esquerdo (ou use o atalho "Ctrl+Shift+X").
- Na barra de pesquisa, digite "ES6 String HTML".
- A extensão "ES6 String HTML" deve aparecer nos resultados da pesquisa. Clique no botão "Instalar" ao lado dela.
- Após a instalação, a extensão irá adicionar suporte de realce de sintaxe para código HTML dentro de strings de várias linhas no ES6.



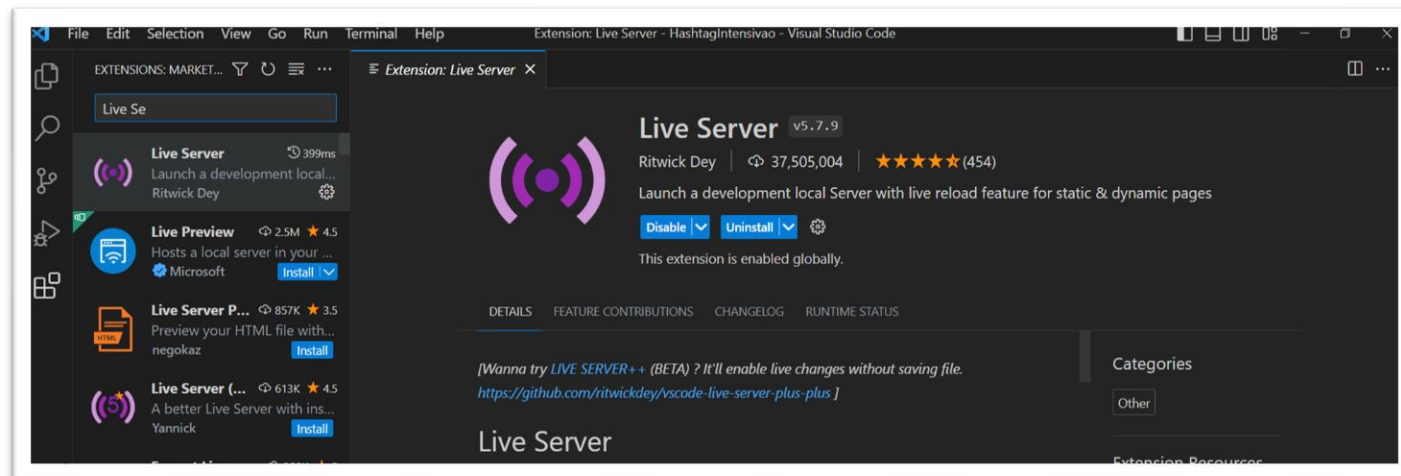


Instalando as ferramentas – VS CODE - EXTENSÕES

O Live Server é uma extensão muito útil para desenvolvimento web, pois facilita a visualização e a atualização instantânea das alterações feitas no código HTML.

Vou te explicar como baixar a extensão "Live Server" no VS Code:

- Abra o Visual Studio Code (VS Code) e vá para a seção de extensões.
- Pesquise por "Live Server" e clique em "Instalar" ao lado da extensão desenvolvida por Ritwick Dey.
- Aguarde a instalação e clique em "Gerenciar" para acessar as configurações da extensão.
- Personalize as configurações, se desejar.
- Abra um arquivo HTML no VS Code, clique com o botão direito do mouse e selecione "Abrir com Live Server".
- O Live Server iniciará um servidor local e abrirá o arquivo HTML no navegador padrão.





Instalando as ferramentas – Node.js

O Node.js é um ambiente de execução de código JavaScript do lado do servidor. Ele permite que você execute código JavaScript fora do navegador, o que significa que você pode criar aplicativos de servidor, scripts de linha de comando e muito mais usando JavaScript. Para instalar o Node.js, você pode seguir os seguintes passos:

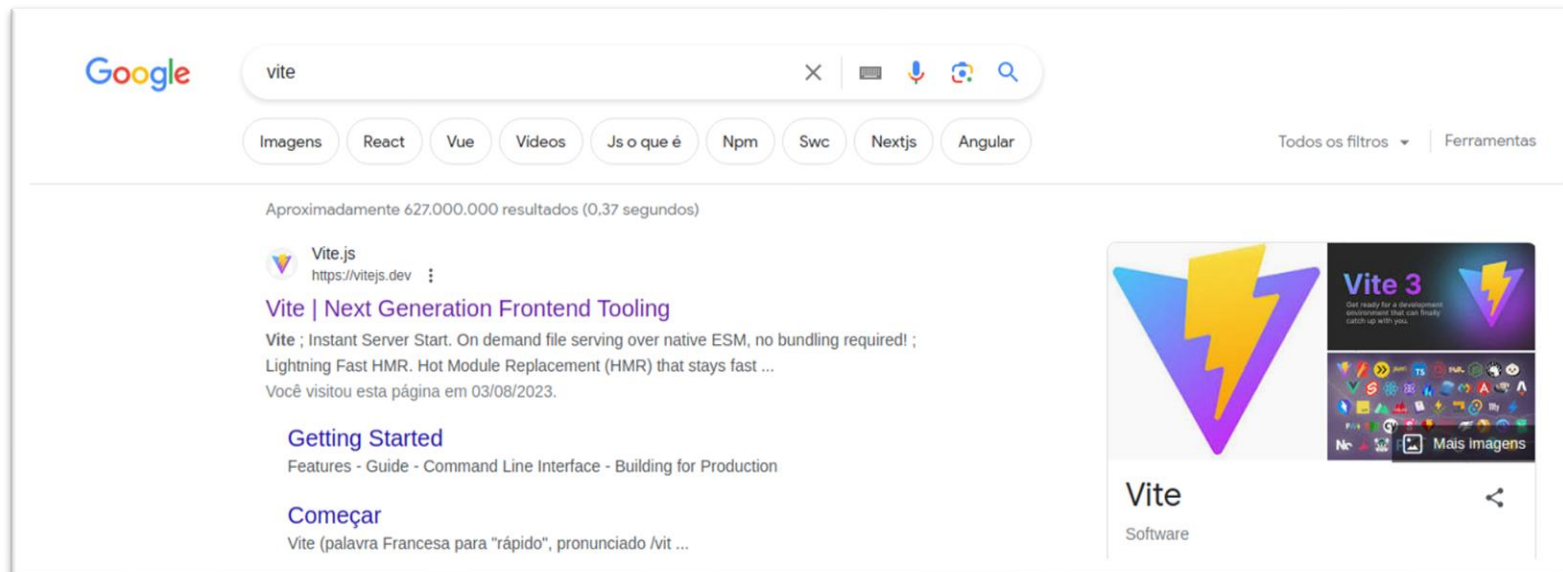


- Acesse o site oficial do Node.js em <https://nodejs.org>.
- Na página inicial, você verá duas versões para download: LTS (Long Term Support) e Current. A versão LTS é recomendada para a maioria dos usuários, pois é mais estável e possui suporte a longo prazo. Selecione a versão LTS ou a versão mais recente, se preferir.
- Após selecionar a versão desejada, você será redirecionado para a página de download. Escolha o instalador adequado para o seu sistema operacional (Windows, macOS ou Linux) e clique no link para iniciar o download.
- Após o download ser concluído, execute o instalador e siga as instruções na tela para concluir a instalação.
- Após a instalação ser concluída, você pode verificar se o Node.js foi instalado corretamente abrindo o terminal ou prompt de comando e digitando o comando `node -v`. Se a versão do Node.js for exibida, significa que a instalação foi bem-sucedida.



Instalando as ferramentas – VITE.JS

O Vite.js é um build tool (ferramenta de construção) e um servidor de desenvolvimento rápido para projetos web. Ele foi projetado para melhorar a experiência de desenvolvimento, oferecendo tempos de compilação instantâneos e recarregamento rápido do navegador. Ele permite que os desenvolvedores escrevam código moderno e aproveitem recursos como o hot module replacement (substituição de módulo em tempo real) para uma experiência de desenvolvimento mais produtiva.





Instalando as ferramentas – VITE.JS



O Vite é uma ferramenta de construção que oferece uma experiência de desenvolvimento mais rápida e leve para projetos web modernos. Ele simula um servidor durante o desenvolvimento para melhorar o desempenho e a produtividade. Aqui estão os passos para entender como o Vite simula um servidor:

- O Vite possui uma opção de configuração chamada `server.port` que permite especificar a porta do servidor^[^0^]. Por padrão, ele usa a porta 3000.
- Durante o desenvolvimento, o Vite utiliza um servidor de desenvolvimento embutido para fornecer funcionalidades adicionais, como substituição de módulo instantânea (HMR) extremamente rápida^[^3^]. O HMR permite atualizar o código no navegador em tempo real, sem a necessidade de recarregar a página.
- Além disso, o Vite oferece uma API de baixo nível chamada `configureServer`, que permite adicionar intermediários de SSR (Server-Side Rendering) ao servidor de desenvolvimento^[^2^]. Os intermediários de SSR pré-interpretam a aplicação para HTML no lado do servidor e, em seguida, a hidratam no cliente.



Instalando as ferramentas – VITE.JS

- O Vite também possui comandos de linha de comando, como `vite dev` e `vite preview`, que podem ser usados para executar aplicações com SSR[^2^]. É possível adicionar intermediários de SSR ao servidor de desenvolvimento com `configureServer` e ao servidor de pré-visualização com `configurePreviewServer`.
- Durante o processo de construção, o Vite utiliza o Rollup para empacotar o código e produzir recursos estáticos altamente otimizados para produção[^3^]. Isso garante um desempenho eficiente e uma melhor experiência para os usuários.

<https://vitejs.dev/>

Neste momento, estamos explicando os conceitos do Vite e como ele funciona internamente. Estamos fornecendo uma visão detalhada de como o Vite opera e como ele trabalha nos bastidores. Mas , quando começarmos nosso projeto, vamos criar um programa usando o Vite para rodar nosso site de E-Commerce. E explicaremos passo a passo como dar os primeiros passos nesse processo.



Parte 2

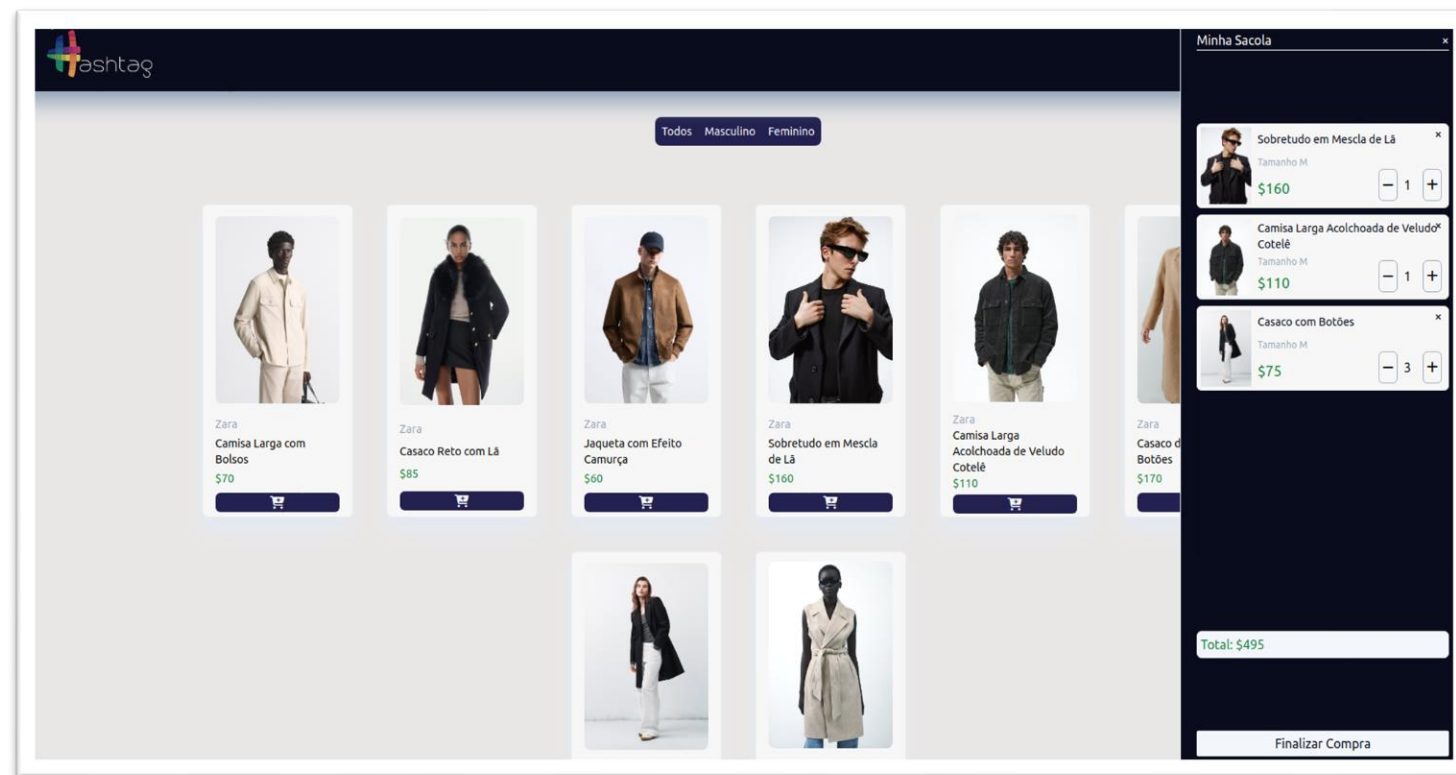
Apresentação do site E-Commerce

Apresentação do Site de E-Commerce

Seja bem vindo à nossa primeira aula do Intensivão de Javascript!

A imagem que está ao lado mostra como ficará o **Site de E-Commerce** que será desenvolvido durante o nosso Intensivão. Nas próximas aulas, você aprenderá tudo, desde o básico do Javascript, como usar ferramentas de construção e um servidor de desenvolvimento, até criar um site totalmente interativo e estiloso, como o da imagem ao lado.

E neste PDF você encontra todo o passo a passo da primeira aula para dar início a sua jornada até o resultado final. O conteúdo tá muito maneiro e tenho certeza que você vai gostar!



Parte 3

O Projeto

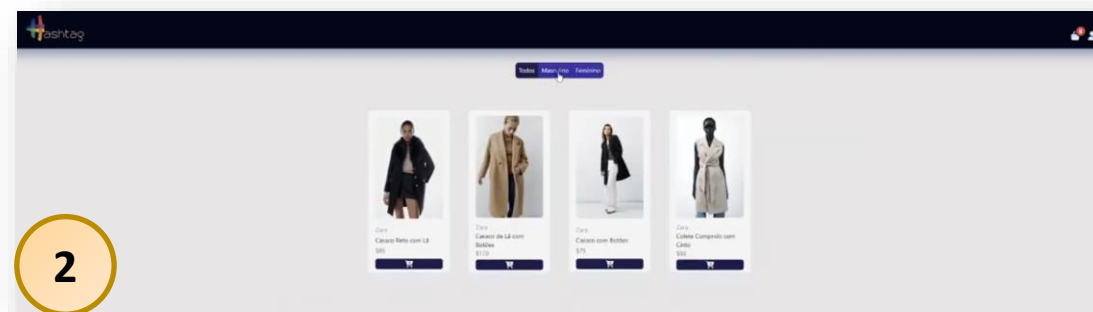
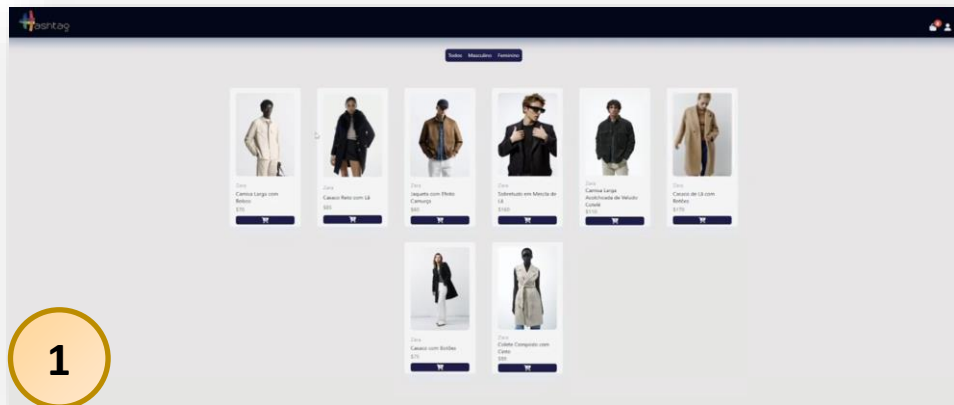
O Projeto

Neste momento, vamos explorar o conceito de um site de E-Commerce, entender como ele funciona e as funcionalidades que iremos desenvolver ao longo do nosso intensivo de Javascript.

Um site de E-Commerce, também conhecido como uma loja virtual, é uma plataforma online onde produtos são vendidos. No nosso caso, estaremos focando em uma loja de roupas.

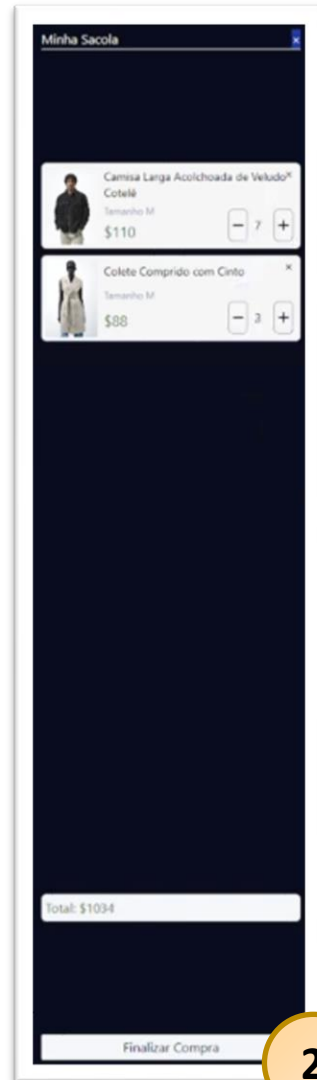
Durante o nosso intensivão de Javascript, vamos implementar várias funcionalidades que são essenciais para um site de E-Commerce. Algumas delas incluem:

- **Catálogo de produtos: (Imagem 1)** – Página onde os clientes podem ver todos os produtos disponíveis para venda. E também adicionaremos a funcionalidade de – Categorias **(Imagem 2)**, no qual os cliente podem filtrar as categorias que o site possui.



O Projeto

1



2

- **Detalhes do produto:** Os clientes podem ver as informações de cada produto, como descrições detalhadas, fotos, tamanhos disponíveis, etc. **(Imagem 1)**.
- **Carrinho de compras:** Uma funcionalidade que permite aos clientes revisar os produtos que selecionaram, alterar quantidades, remover itens e ver o total da compra **(Imagem 2)**.

Parte 3

O Projeto

- **Finalização da compra (checkout):** Uma página onde os clientes podem fornecer suas informações de entrega e pagamento para completar a compra (**Imagem 1**).
- **Histórico de compras:** Uma funcionalidade que permite aos clientes ver todas as suas compras anteriores (**Imagem 2**).

Finalizar Compra

Seus Dados

Nome
John

Sobrenome
Lira

Email
john.lira@hashtag.com

Numero de Contato
(21) 99887-7665

Pagamento e Entrega

Numero do Cartao
1111 2222 3333 4444

CVV
987

Validade
12/99

CEP
234576-789

Endereço
Rua que Sobra e Desce

Numero
123

Complemento
12/99

Resumo da Compra

Camisa Larga Acolchoada de Veludo Cotelê	Tamanho M	\$110	7
Casaco de Lã com Botões	Tamanho M	\$170	3
Colete Comprido com Cinto	Tamanho M	\$88	4

Total: Total: \$1632

Finalizar Compra

Seu histórico de compras

August 5 2023, 16:49:24

	Camisa Larga com Bolsos	Tamanho M	\$70	1
	Casaco Rato com Lã	Tamanho M	\$85	1
	Jaqueta com Efeito Camurça	Tamanho M	\$60	3
	Sobretudo em Mescla de Lã	Tamanho M	\$160	3
	Camisa Larga Acolchoada de Veludo Cotelê	Tamanho M	\$110	2
	Casaco de Lã com Botões	Tamanho M	\$170	2

1

2

Parte 4

Introdução ao Javascript

Introdução ao Javascript

O que é JavaScript?



JavaScript é uma linguagem de programação de alto nível, dinâmica e interpretada, que é usada principalmente para tornar as páginas da web interativas. Com JavaScript, os desenvolvedores podem controlar o comportamento de elementos da página da web, criar animações, processar e manipular dados, entre outras coisas. A linguagem é executada no navegador do cliente, o que significa que o código é executado no dispositivo do usuário, em vez do servidor web, tornando a interação do usuário com a página da web mais rápida e eficiente.

JavaScript foi criado em 1995 por Brendan Eich, enquanto ele trabalhava na Netscape Communications Corporation. Originalmente, foi desenvolvido para funcionar no navegador Netscape Navigator com o objetivo de facilitar a programação de animações e alertas em páginas da web. Em 1996, a Microsoft começou a suportar JavaScript em seu navegador, o que ajudou a consolidar a linguagem como uma das principais linguagens de programação da atualidade.

Introdução ao Javascript

O que é ES6?

ES6, também conhecido como ECMAScript 2015, é uma versão do padrão ECMAScript que define a linguagem JavaScript. ES6 introduziu uma série de novos recursos e melhorias significativas na linguagem JavaScript, incluindo novas declarações de variáveis (`let` e `const`), classes, módulos, promessas, operadores de spread e rest, entre outros. Esses novos recursos tornaram o JavaScript uma linguagem de programação mais poderosa e fácil de usar.

Interface

A interface JavaScript em uma página web é o conjunto de funcionalidades que permite a implementação de elementos interativos na página, proporcionando uma experiência de usuário mais dinâmica e complexa. JavaScript é uma linguagem de programação client-side, o que significa que é executada a partir do cliente, não necessitando acessar servidores.

Isso permite que o desenvolvedor trabalhe a partir dos navegadores tradicionais, sem a necessidade de criar executáveis. A linguagem JavaScript é usada para manipular HTML e CSS para atualizar uma interface do usuário. Assim, os scripts são inseridos em páginas HTML e interpretados pelo navegador cada vez que a página é carregada.

Introdução ao Javascript

A linguagem JavaScript é acionada pelo motor de renderização do navegador após a interpretação do HTML e do CSS. Isso garante que a estrutura da página esteja pronta quando o JavaScript for executado.

Com o JavaScript, é possível controlar os elementos de uma página em tempo real, sem necessariamente ter que receber os dados ou uma resposta do servidor. Por exemplo, é possível atualizar o conteúdo de uma página web sem precisar recarregá-la por completo ao preencher um formulário.

Além disso, o JavaScript permite adicionar comportamento interativo nas páginas da web, permitindo que os usuários interajam com uma página. Isso pode incluir mostrar e esconder informações ao clicar em um botão ou alguma parte do site.

Em resumo, a interface JavaScript para uma página web é a camada final de funcionalidade em sites altamente interativos, adicionando funcionalidade interativa e outros conteúdos dinâmicos da web às páginas web.



Parte 5

Arquitetura da Internet

Arquitetura da internet

A arquitetura da internet é composta por uma série de elementos interligados que trabalham juntos para fornecer a infraestrutura que possibilita a navegação na web.

Em essência, a internet é uma gigantesca rede de computadores que se comunicam entre si. Essa comunicação é possível graças a uma série de protocolos e sistemas, como o Protocolo de Internet (IP), o Sistema de Nomes de Domínio (DNS) e o Protocolo de Transferência de Hipertexto (HTTP).

1 - **Endereços IP e DNS:** Cada computador conectado à internet tem um endereço IP único, que é uma série de números usada para identificar o computador na rede. No entanto, lembrar esses números pode ser difícil para os humanos, então usamos nomes de domínio (como google.com) para facilitar a identificação dos sites. O DNS é o sistema que traduz esses nomes de domínio em endereços IP.

2 - **HTTP e servidores web:** Quando você digita um endereço de site no seu navegador, o navegador envia uma requisição HTTP para o servidor que hospeda o site. Se o servidor aprovar a requisição, ele envia os arquivos do site para o navegador em pequenos pedaços chamados pacotes de dados. O navegador então monta esses pedaços para mostrar o site completo.

3 - **TCP/IP:** O HTTP e todas as outras informações enviadas entre o cliente (seu computador) e o servidor são transmitidas através da sua conexão de internet usando o TCP/IP. O TCP/IP é um conjunto de regras que definem como os dados devem ser enviados e recebidos na internet.

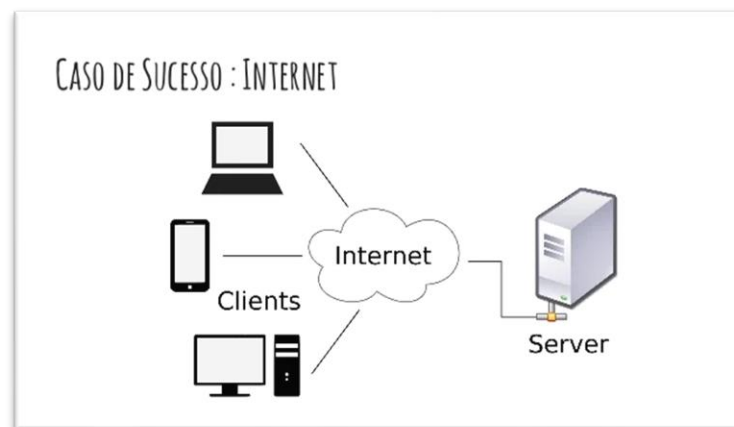
Arquitetura da internet

A arquitetura da web é uma parte importante da arquitetura da internet, pois ela define como as informações são organizadas e apresentadas nos sites. Uma boa arquitetura da web facilita a navegação dos usuários e ajuda a melhorar a experiência do usuário.

Essa é uma visão geral simplificada da arquitetura da internet. Na realidade, há muitos outros componentes e protocolos envolvidos, cada um desempenhando um papel crucial para garantir que a internet funcione de maneira eficiente.

Em uma arquitetura cliente/servidor, o cliente é geralmente um navegador web que solicita informações e o servidor é a máquina que responde a essas solicitações.

Quando você digita um URL no seu navegador (cliente), ele envia uma solicitação ao servidor que hospeda o site. O servidor então responde enviando os arquivos necessários para o navegador exibir o site. Esses arquivos geralmente incluem HTML, CSS, e JavaScript.



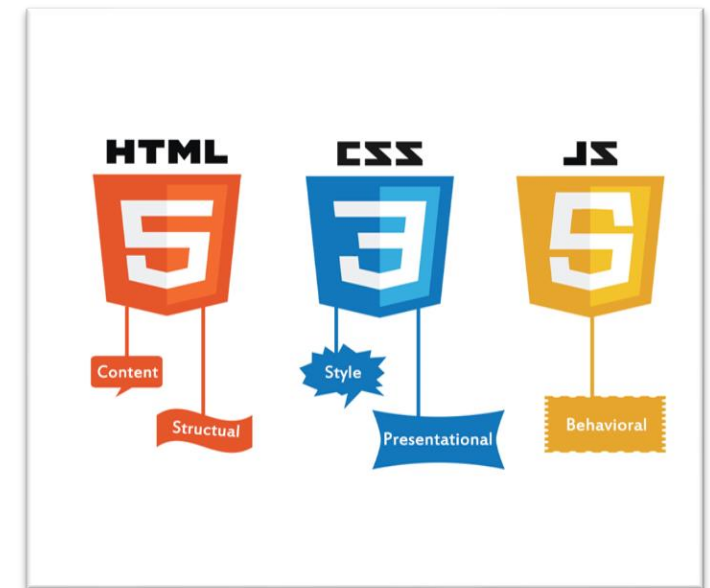
Arquitetura da internet

HTML, CSS e JavaScript são as três tecnologias fundamentais usadas para criar sites.

- **HTML (HyperText Markup Language)** é usado para estruturar o conteúdo de um site. Ele define a estrutura básica do site e o conteúdo como texto, imagens, formulários etc.
- **CSS (Cascading Style Sheets)** é usado para controlar a aparência do site. Ele define a disposição dos elementos HTML, cores, fontes e outros aspectos visuais.
- **JavaScript** é uma linguagem de programação que é usada para tornar os sites interativos. Ele pode manipular o HTML e o CSS para adicionar, remover ou alterar conteúdo, responder a ações do usuário, realizar cálculos e muito mais.

Na arquitetura cliente/servidor, o HTML, CSS e JavaScript são usados juntos para criar a experiência do usuário no lado do cliente.

Quando o servidor recebe uma solicitação do cliente, ele envia os arquivos HTML, CSS e JavaScript para o navegador. O navegador então interpreta o HTML para estruturar a página, aplica o CSS para estilizar a página e executa o JavaScript para adicionar interatividade.

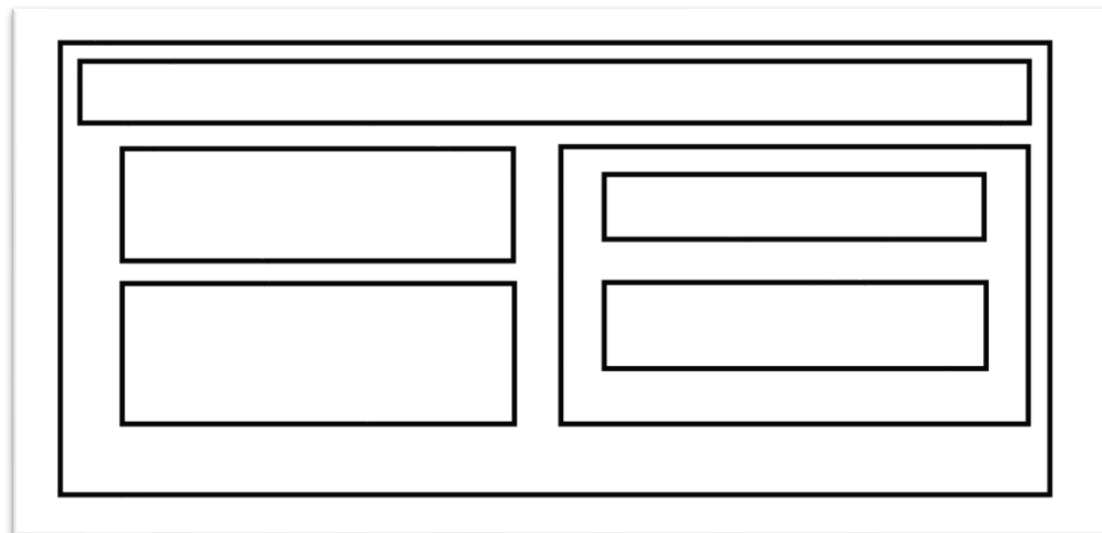


Arquitetura da internet

Um conceito importante para estruturarmos nosso site é fazer um comparativo entre o HTML e as estruturas retangulares.

O HTML define a estrutura e a organização do conteúdo de um site. Ele permite criar cabeçalhos, parágrafos, listas, imagens e outros elementos que compõem a página. Assim como um esqueleto fornece a estrutura básica de um edifício, o HTML fornece a estrutura básica de um site.

As estruturas retangulares, como divs e sections, são elementos do HTML que podem ser usados para criar blocos de conteúdo retangulares em um site. Esses blocos podem ser posicionados, dimensionados e estilizados de várias maneiras para criar layouts complexos com o CSS. Podemos comparar as estruturas retangulares a blocos de construção que são usados para criar a aparência visual de um site.



Parte 6

Iniciando o Projeto - Vite

Iniciando o Projeto - Vite

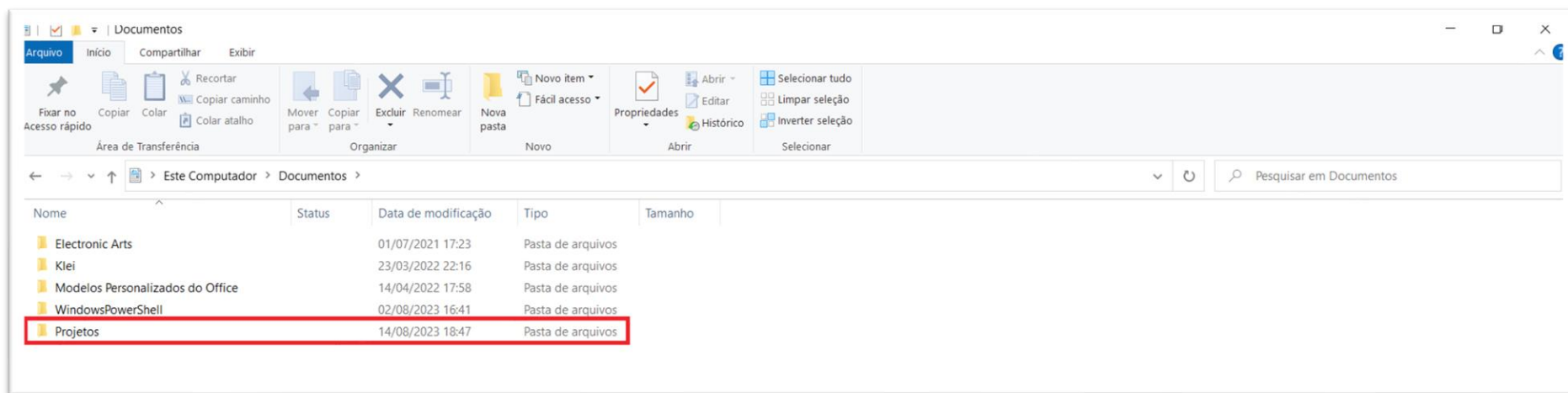
Para iniciar o nosso projeto, vamos criar uma nova pasta com o nome "Projetos". Essa pasta será o local onde iremos armazenar todos os arquivos do nosso site de E-Commerce. Dentro dessa pasta, colocaremos os arquivos HTML, CSS e Javascript, além de alguns arquivos adicionais que serão utilizados para estilização e configuração do ambiente de desenvolvimento, como a criação do nosso programa utilizando o Vite.

Vou te explicar passo a passo como criar uma pasta nova no Windows:

- Primeiro, abra o "Explorador de Arquivos" clicando no ícone da pasta amarela na barra de tarefas ou pressionando a tecla "Windows" + "E" no teclado.
- Navegue até o local onde você deseja criar a nova pasta. Por exemplo, você pode escolher criar a pasta na área de trabalho ou dentro de outra pasta existente.
- Com o local selecionado, clique com o botão direito do mouse em um espaço vazio da janela do Explorador de Arquivos. Um menu de opções será exibido.

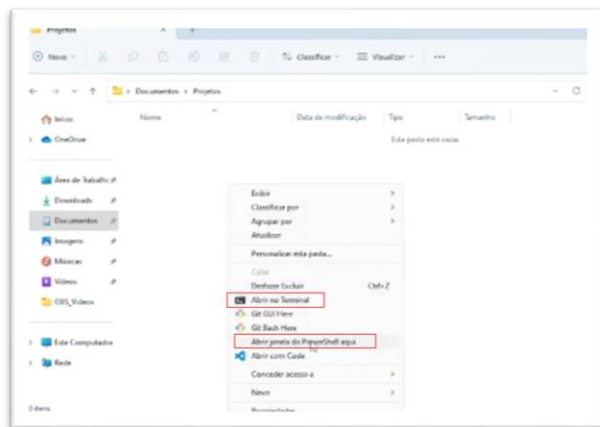
Iniciando o Projeto - Vite

- No menu de opções, passe o cursor sobre a opção "Novo" e, em seguida, clique em "Pasta". Uma nova pasta será criada com o nome "Nova pasta" destacado.
- Digite o nome desejado para a nova pasta. No nosso caso, digite "Projetos" como o nome da pasta.
- Pressione a tecla "Enter" no teclado para confirmar o nome da pasta. A nova pasta será criada no local selecionado.



Iniciando o Projeto - Vite

1 - Dentro da pasta recém criada "Projetos", você irá clicar com o botão direito do mouse e abrir a opção "Terminal" ou "Powershell" para iniciarmos com o Vite.

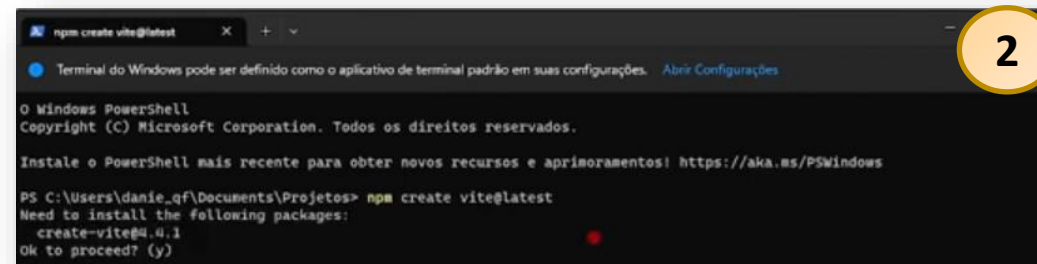
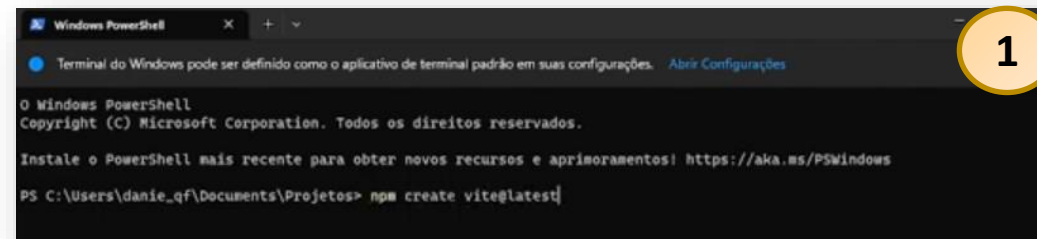


2 - No terminal, digite o comando NPM do Vite (o comando "npm create vite@latest" é usado para criar um novo projeto Vite usando o gerenciador de pacotes npm):

```
$ npm create vite@latest
```

bash

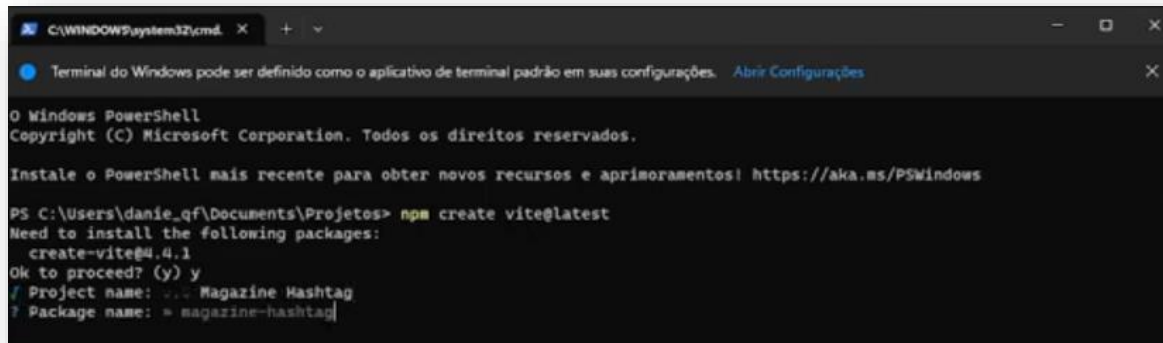
3 - O terminal irá iniciar o processo do Vite, após o comando de NPM (**Imagem 1**), para continuar aperte y + Enter (**Imagem 2**).



Iniciando o Projeto - Vite

4 – O próximo passo é escolher um nome para o seu projeto Vite e especificar o nome do pacote que será usado para publicar e instalar o projeto.

- O **"nome"** se refere ao nome do seu projeto Vite. Por exemplo, você pode dar a ele um nome descritivo, como "Magazine Hashtag".
- O **"pacote"** se refere ao nome do pacote que será usado para publicar e instalar o projeto. Geralmente, é recomendado usar o mesmo nome do projeto como o nome do pacote.



```
C:\WINDOWS\system32\cmd. x + -
Terminal do Windows pode ser definido como o aplicativo de terminal padrão em suas configurações. Abrir Configurações

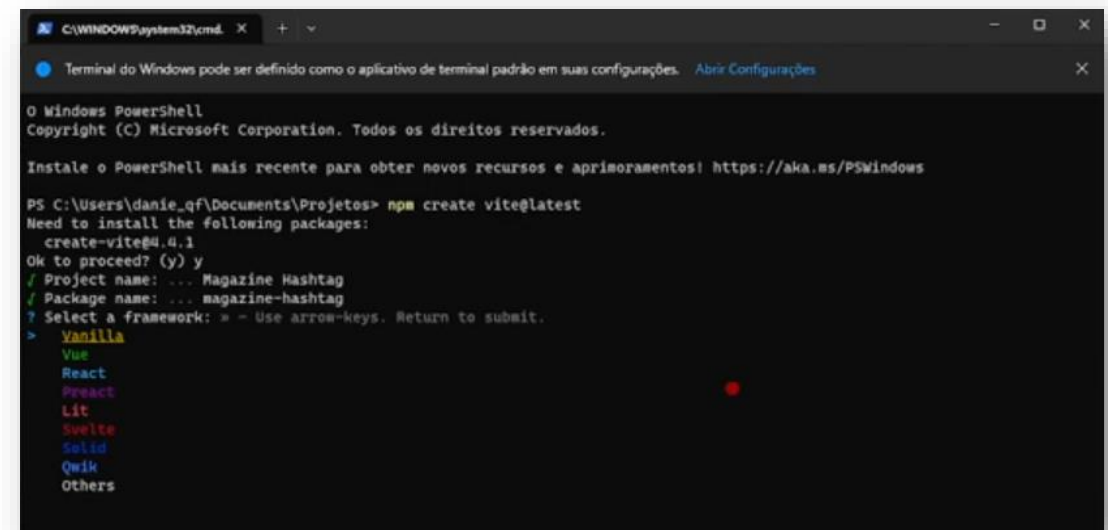
O Windows PowerShell
Copyright (C) Microsoft Corporation. Todos os direitos reservados.

Instale o PowerShell mais recente para obter novos recursos e aprimoramentos! https://aka.ms/PSWindows

PS C:\Users\danie_qf\Documents\Projetos> npm create vite@latest
Need to install the following packages:
  create-vite@4.1
Ok to proceed? (y) y
? Project name: ... Magazine Hashtag
? Package name: ... magazine-hashtag
```

5 – Vamos selecionar o template ou estrutura inicial para o seu projeto Vite, que será o modelo Vanilla.

- **"Vanilla"**: refere-se a um projeto Vite básico que utiliza JavaScript puro, ou seja, sem nenhum framework específico. Nesse caso, você terá uma estrutura inicial mínima para começar a desenvolver seu aplicativo web.



```
C:\WINDOWS\system32\cmd. x + -
Terminal do Windows pode ser definido como o aplicativo de terminal padrão em suas configurações. Abrir Configurações

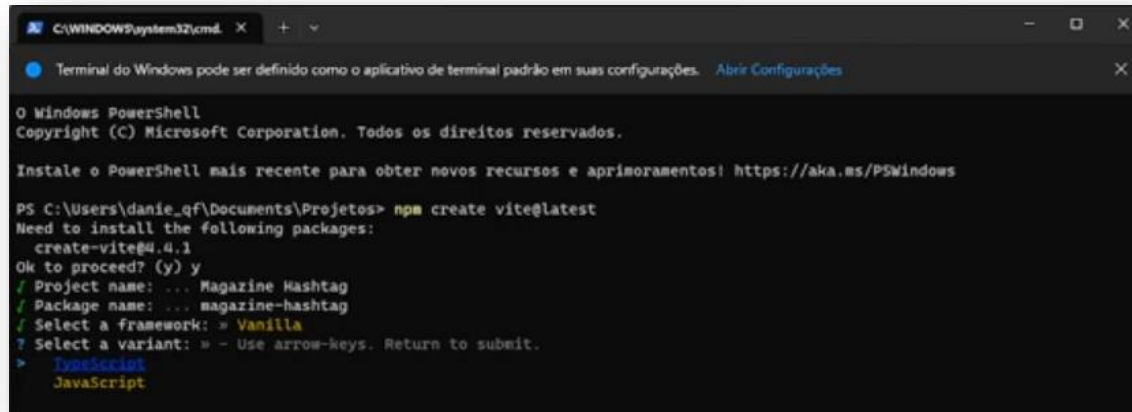
O Windows PowerShell
Copyright (C) Microsoft Corporation. Todos os direitos reservados.

Instale o PowerShell mais recente para obter novos recursos e aprimoramentos! https://aka.ms/PSWindows

PS C:\Users\danie_qf\Documents\Projetos> npm create vite@latest
Need to install the following packages:
  create-vite@4.1
Ok to proceed? (y) y
? Project name: ... Magazine Hashtag
? Package name: ... magazine-hashtag
? Select a framework: » - Use arrow-keys. Return to submit.
> Vanilla
  Vue
  React
  Preact
  Lit
  Svelte
  Solid
  Qwik
  Others
```

Iniciando o Projeto - Vite

6 – Por fim, ao escolher "javascript" após executar o comando "npm create vite@latest", você está selecionando o template de JavaScript puro para o seu projeto Vite.



```
C:\WINDOWS\system32\cmd. X + -
Terminal do Windows pode ser definido como o aplicativo de terminal padrão em suas configurações. Abrir Configurações X

O Windows PowerShell
Copyright (C) Microsoft Corporation. Todos os direitos reservados.

Instale o PowerShell mais recente para obter novos recursos e aprimoramentos! https://aka.ms/PSWindows

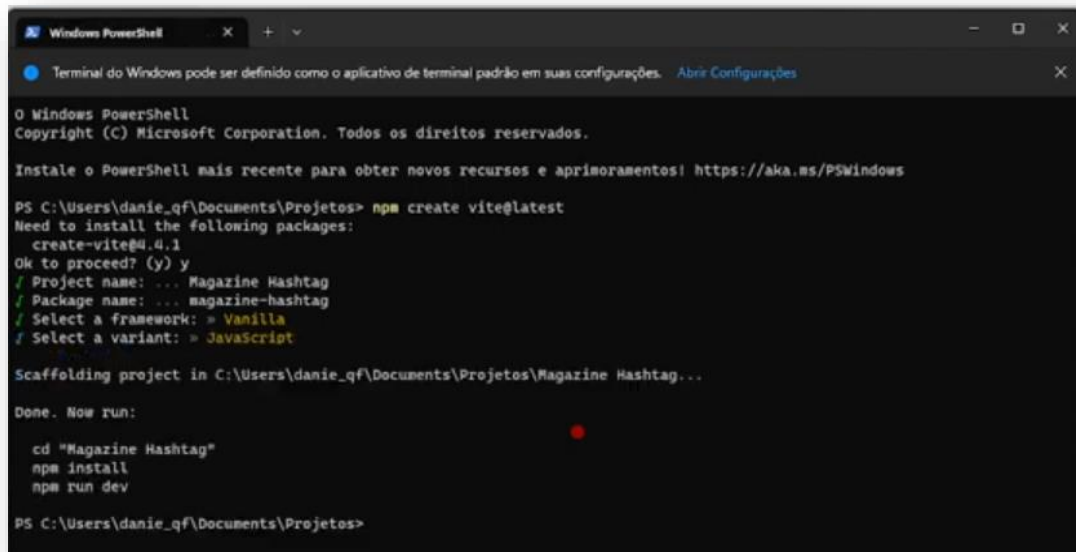
PS C:\Users\danie_qf\Documents\Projetos> npm create vite@latest
Need to install the following packages:
  create-vite@4.1
Ok to proceed? (y) y
✓ Project name: ... Magazine Hashtag
✓ Package name: ... magazine-hashtag
✓ Select a framework: > Vanilla
? Select a variant: > - Use arrow-keys. Return to submit.
  > TypeScript
  JavaScript
```

Após executar o comando "npm create vite@latest" no terminal, você pode esperar ver algumas mensagens de progresso e informações sobre o processo de criação do projeto Vite. Essas mensagens podem incluir:

- Verificação da versão do npm e do Node.js instalados na sua máquina.
- Download e instalação das dependências necessárias para o projeto Vite.
- Criação da estrutura de diretórios e arquivos iniciais do projeto.
- Configuração do ambiente de desenvolvimento, como a definição de scripts de build e start.
- Mensagens de sucesso indicando que o projeto Vite foi criado com êxito.

Iniciando o Projeto - Vite

Você deve se deparar com uma janela parecida com essa, o que quer dizer que as configurações foram feitas com Sucesso.



```
Windows PowerShell
Terminal do Windows pode ser definido como o aplicativo de terminal padrão em suas configurações.  Abrir Configurações

O Windows PowerShell
Copyright (C) Microsoft Corporation. Todos os direitos reservados.

Instale o PowerShell mais recente para obter novos recursos e aprimoramentos! https://aka.ms/PSWindows

PS C:\Users\danie_qf\Documents\Projetos> npm create vite@latest
Need to install the following packages:
create-vite@4.1
Ok to proceed? (y) y
/ Project name: ... Magazine Hashtag
/ Package name: ... magazine-hashtag
/ Select a framework: > Vanilla
/ Select a variant: > JavaScript

Scaffolding project in C:\Users\danie_qf\Documents\Projetos\Magazine Hashtag...

Done. Now run:

  cd "Magazine Hashtag"
  npm install
  npm run dev

PS C:\Users\danie_qf\Documents\Projetos>
```

E agora vamos iniciar a nossa aplicação Vite no navegador, seguindo alguns comandos.

1 – No terminal digite "cd nomeDaPastaProjeto"

```
PS C:\Users\mikag\OneDrive\Documentos\Projetos> cd '..\Magazine Hashtag\'
```

2–Execute NPM install para instalar as dependências de um projeto. Quando você executa o comando "npm install" no terminal, o npm verifica o arquivo "package.json" do seu projeto para identificar as dependências listadas nele.

```
PS C:\Users\mikag\OneDrive\Documentos\Projetos\Magazine Hashtag> npm install
```

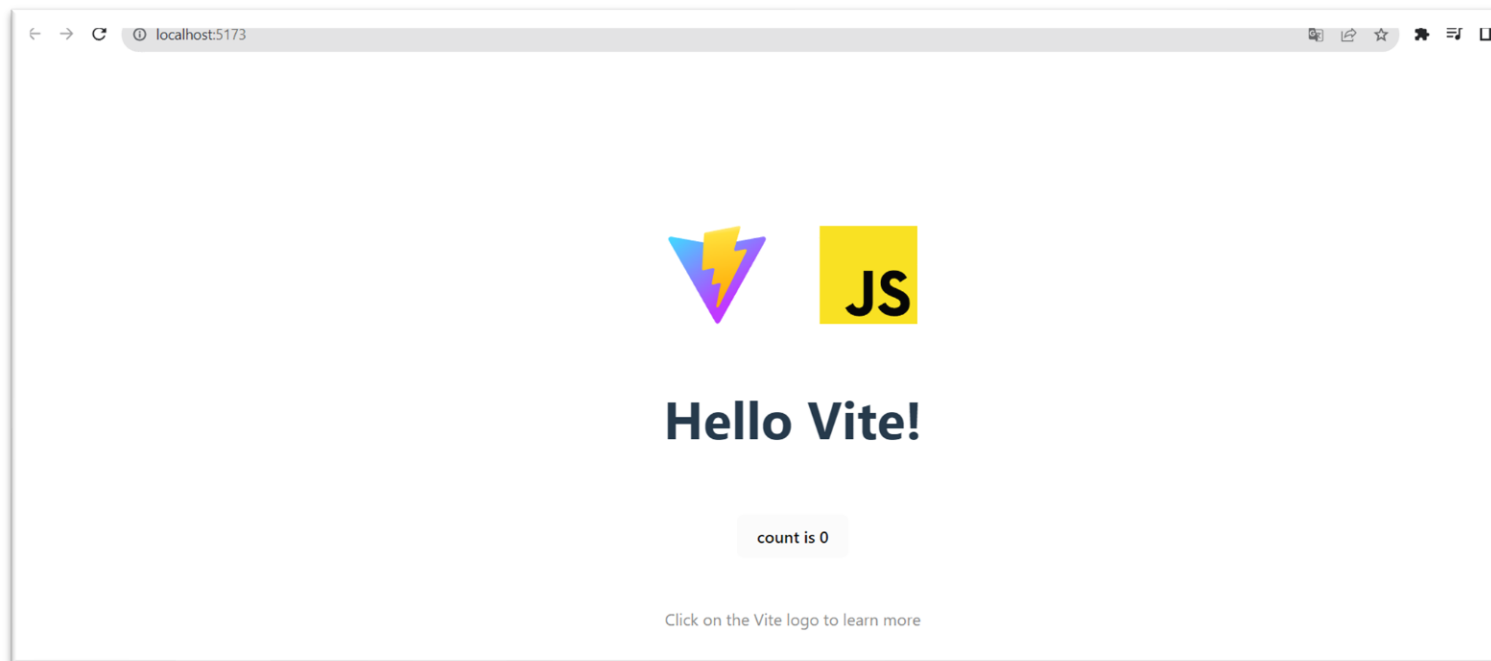
3 – Rode "npm run dev" – para iniciar o servidor de desenvolvimento do Vite no navegador.

```
PS C:\Users\mikag\OneDrive\Documentos\Projetos\Magazine Hashtag> npm run dev

> magazine-hashtag@0.0.0 dev
> vite
```


Iniciando o Projeto - Vite

Pronto! A sua aplicação Vite já está no ar! Você pode abrir o seu navegador e acessar o endereço "<http://localhost:3000>" (ou a porta específica informada no terminal) para visualizar a sua aplicação Vite em tempo real.



Parte 7

Estruturas do Projeto - HTML / CSS

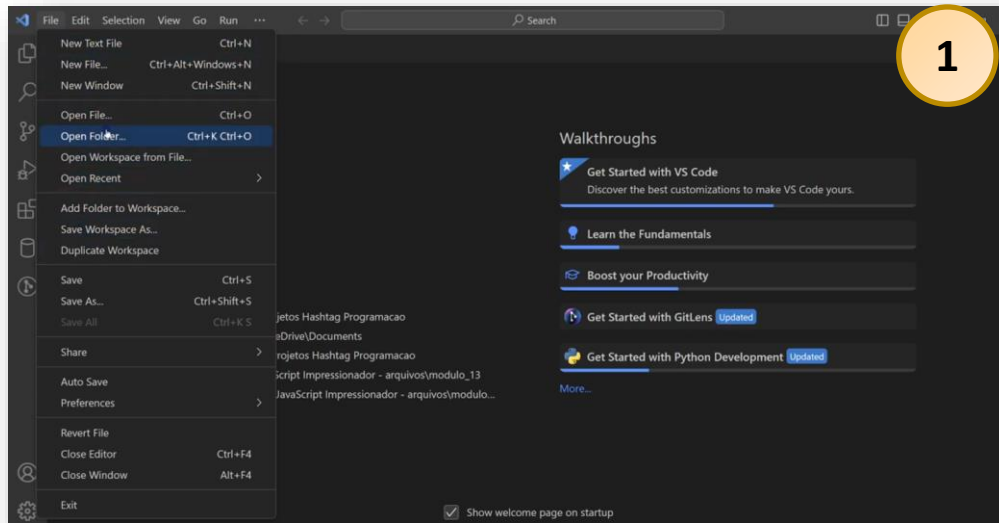
Estruturas do Projeto – HTML / CSS

Agora vamos abrir a nossa pasta no VS Code, que será o programa que utilizaremos para realizarmos a construção do nosso site. Para fazer isso, siga os passos abaixo:

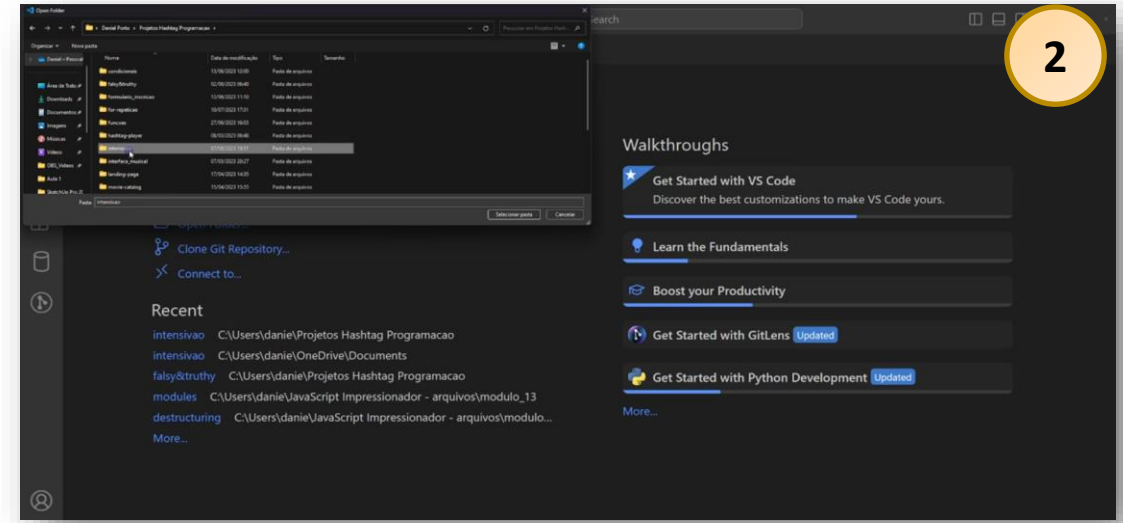
- Certifique-se de ter o Visual Studio Code instalado no seu computador. Se ainda não tiver, você pode baixá-lo gratuitamente no site oficial do VS Code.
- Abra o Visual Studio Code clicando no ícone do programa na área de trabalho ou no menu Iniciar.
- No VS Code, clique em 'Arquivo' no menu superior e, em seguida, selecione 'Abrir Pasta'. Uma janela de seleção de pasta será exibida **(Imagem 1)**.
- Navegue até o local onde você criou a pasta 'Magazine Hashtag'. Por exemplo, se você a criou na área de trabalho, vá até a área de trabalho **(Imagem 2)**.
- Selecione a pasta 'Magazine Hashtag' e clique no botão 'Selecionar Pasta' na parte inferior direita da janela.
- A pasta 'Magazine Hashtag' será aberta no VS Code. Você verá a estrutura de arquivos e pastas do seu projeto no painel lateral esquerdo **(Imagem 3)**.

Estruturas do Projeto- HTML / CSS

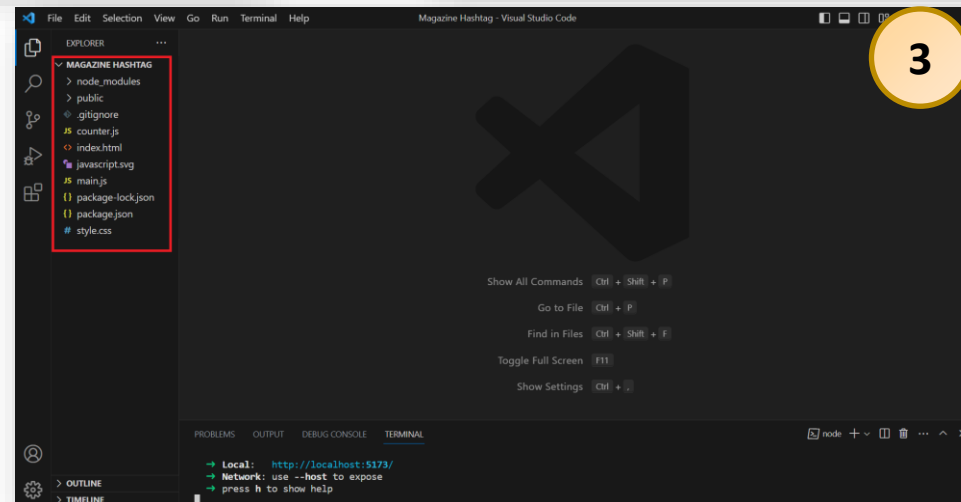
- Abrir Folder/Pasta



- Procurando pasta criada para projeto



- Arquivos e Pastas do projeto



Estruturas do Projeto- HTML / CSS

Com o nosso projeto aberto no VS Code, vamos **remover os arquivos que não serão necessários** para nós. Isso acontece porque o Vite já cria uma estrutura inicial para a aplicação, mas queremos criar nosso próprio site de E-Commerce. Os únicos arquivos que vamos manter são o index.html, o arquivo packages.json, a pasta node_modules, public e o arquivo .gitignore.

Ao remover os arquivos que não precisamos, teremos um projeto limpo e pronto para começar a construir nosso site de E-Commerce do zero. O arquivo index.html é onde iremos escrever o código HTML da nossa página inicial. O arquivo packages.json contém informações sobre as dependências do nosso projeto, como bibliotecas e frameworks que podemos usar. A pasta node_modules é onde as dependências são armazenadas após a instalação. A pasta "public" é um diretório especial no projeto Vite que é usado para armazenar arquivos estáticos, como imagens, fontes e outros recursos que serão acessados diretamente pelo navegador. Esses arquivos não passam pelo processo de compilação do Vite e são servidos diretamente do diretório "public" para o navegador. Manter essa pasta é importante para garantir que nossos arquivos estáticos sejam acessíveis corretamente durante o desenvolvimento e implantação do site.

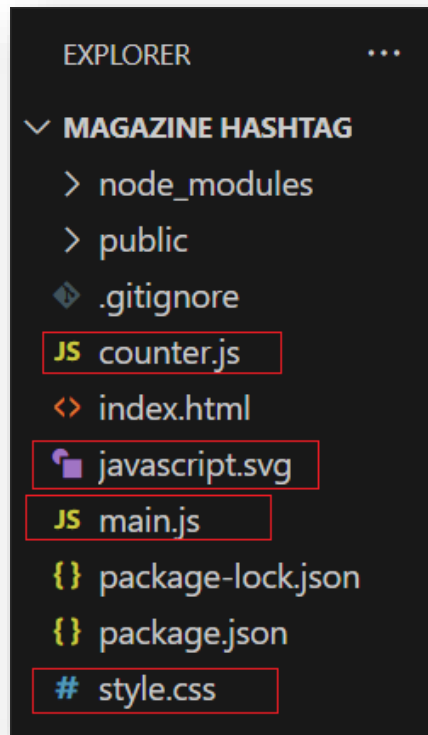
O arquivo ".gitignore" é um arquivo de configuração que especifica quais arquivos e pastas devem ser ignorados pelo sistema de controle de versão Git. Ao manter o arquivo ".gitignore", podemos evitar que arquivos desnecessários ou sensíveis, como arquivos de configuração local ou chaves de API, sejam incluídos no repositório Git. Isso é importante para manter a segurança e a organização do nosso projeto.

Ao manter apenas esses arquivos essenciais, teremos total controle sobre a criação do nosso site de E-Commerce, podendo adicionar nossos próprios arquivos, estilos e funcionalidades personalizadas. Isso nos permite construir um site único e adaptado às nossas necessidades específicas.

Estruturas do Projeto- HTML / CSS

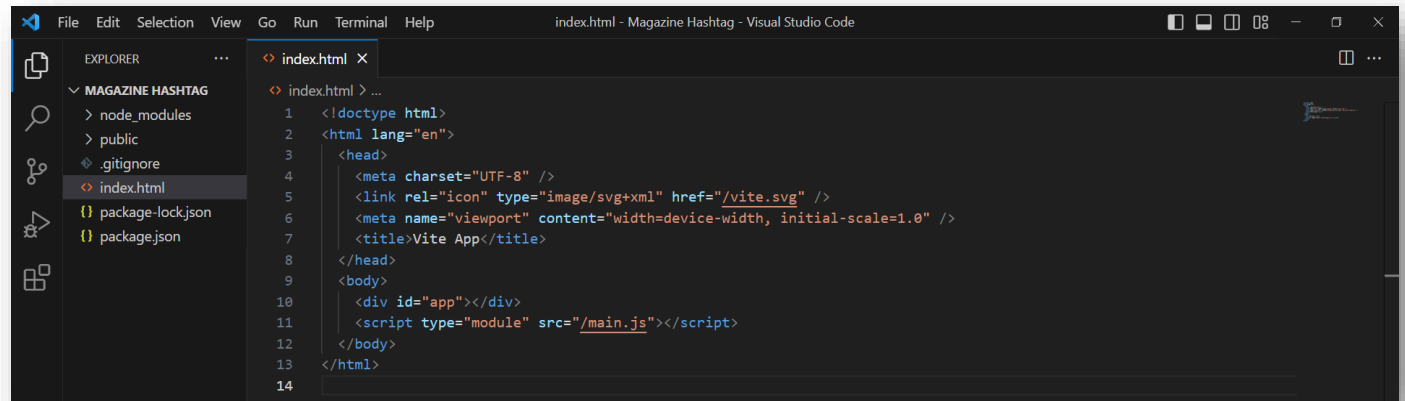
Então iremos deletar os arquivos:

- Counter.js
- Javascript.svg
- Main.js
- Style.css



E nossa estrutura inicial irá manter os arquivos:

- Index.html
- Pasta node_modules
- Pasta public
- .gitignore
- Package-lock.json
- Package.json



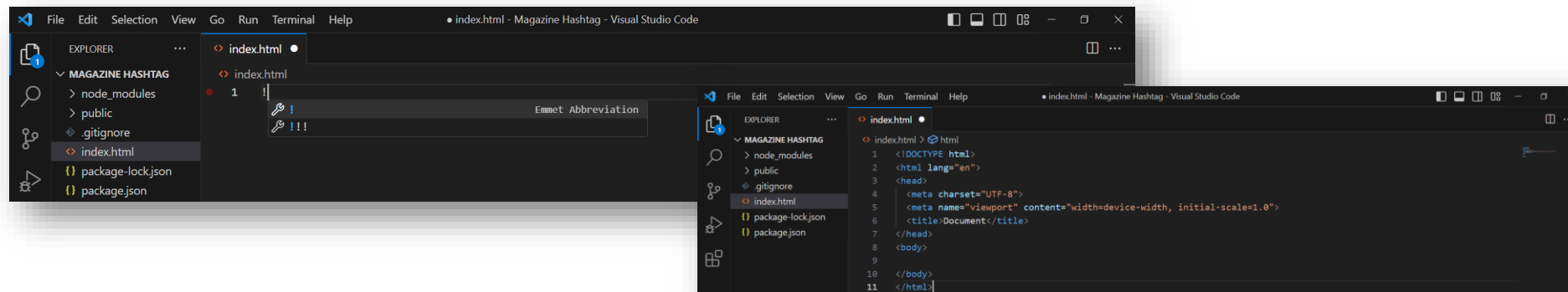
Estruturas do Projeto – HTML / CSS

O arquivo "index.html" é um arquivo HTML que serve como a página inicial do seu site. Ele contém a estrutura básica do seu site, incluindo a marcação HTML, os estilos CSS e os scripts JavaScript. É a partir desse arquivo que você irá construir e desenvolver o conteúdo do seu site, adicionando elementos, estilos e funcionalidades.

Ao abrir o seu site no navegador, o arquivo "index.html" será o primeiro a ser carregado e exibido como a página inicial do seu site. É nele que você irá definir o layout, a estrutura e o conteúdo principal do seu site.

Agora você pode começar a editar o arquivo "index.html" no VS Code, adicionando o código HTML necessário para construir o seu site de E-Commerce.

- Com o arquivo "index.html" aberto no VS Code, apague todo o conteúdo dele e na linha em branco dentro do arquivo digite (!) + tab ou Enter.
- O VS Code irá expandir automaticamente o snippet ! para um modelo básico de arquivo HTML.
- O modelo básico de arquivo HTML incluirá a estrutura básica do HTML, incluindo as tags <html>, <head> e <body>.



Estruturas do Projeto- HTML / CSS

A estrutura básica do HTML gerada pelo VS Code inclui a declaração do tipo de documento (**<!DOCTYPE html>**), a **tag <html>** que envolve todo o conteúdo da página, a **tag <head>** que contém informações sobre a página, como o título exibido na aba do navegador (**tag <title>**), e a **tag <body>** onde você irá adicionar o conteúdo visível do seu site.

Agora você pode começar a adicionar o conteúdo do seu site dentro das **tags <body>...</body>**. Por exemplo, você pode adicionar cabeçalhos, parágrafos, imagens, links e outros elementos HTML.



Em HTML, uma **tag** é um elemento fundamental usado para estruturar e formatar o conteúdo de uma página da web. As **tags** são escritas **entre os símbolos < e >**, e podem conter atributos e/ou texto.

As **tags HTML** são essenciais para estruturar e organizar o conteúdo de uma página da web. Elas permitem que você crie seções, formate o texto, insira imagens, crie links e muito mais.

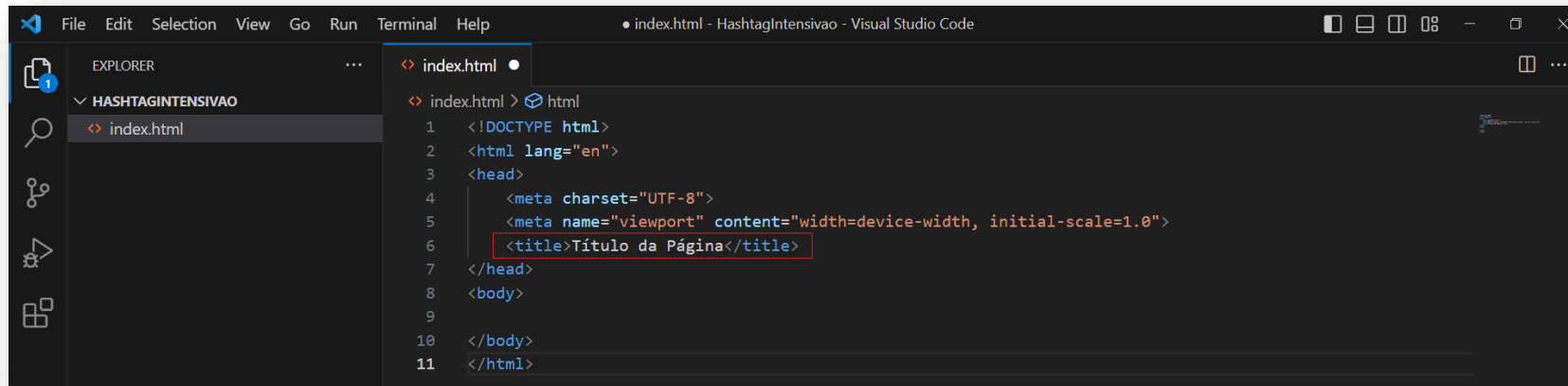
Ao escrever código HTML, é importante usar as **tags** corretas e fechá-las adequadamente para garantir que a página seja exibida corretamente nos navegadores.

Compreender e utilizar as **tags** corretamente é fundamental para criar páginas da web bem estruturadas e funcionais.

Estruturas do Projeto – HTML / CSS

A **tag <title>** é usada dentro da **seção <head>** do seu documento HTML para definir o título da página. O título é exibido na barra de título do navegador e também pode ser exibido em outros lugares, como nos resultados de pesquisa.

É importante notar que o título da página deve ser relevante e descritivo para que os usuários possam entender o conteúdo da página antes mesmo de abri-la. Além disso, os mecanismos de busca também levam em consideração o título da página ao exibir os resultados de pesquisa.



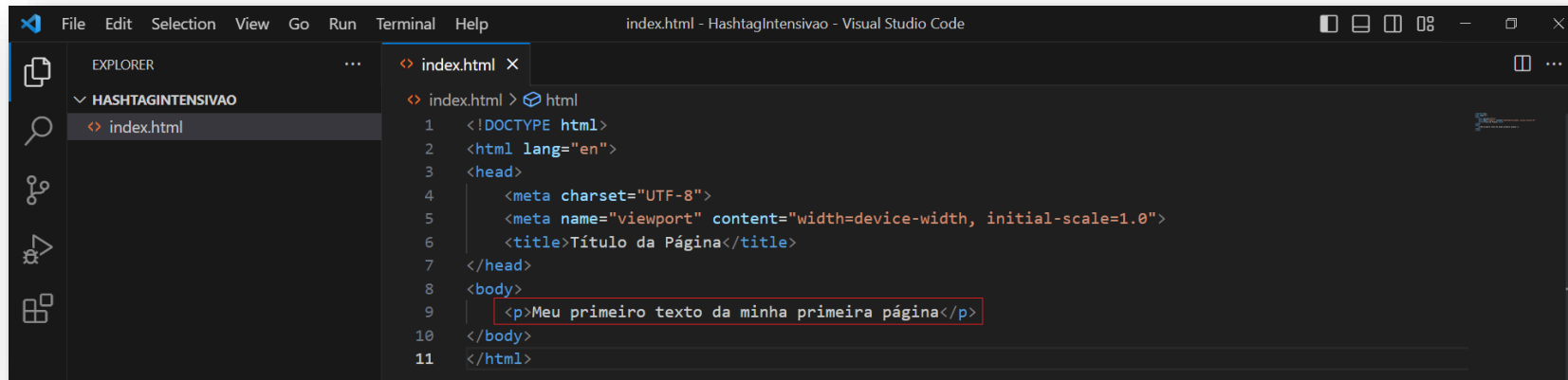
```
1 <!DOCTYPE html>
2 <html lang="en">
3 <head>
4   <meta charset="UTF-8">
5   <meta name="viewport" content="width=device-width, initial-scale=1.0">
6   <title>Título da Página</title>
7 </head>
8 <body>
9
10 </body>
11 </html>
```



Estruturas do Projeto – HTML / CSS

A **tag <p>** é uma tag HTML usada para definir um parágrafo de texto em um documento HTML. A **tag <p>** é usada para separar e estruturar o conteúdo de texto em parágrafos distintos.

A **tag <p>** é útil para organizar e estruturar o conteúdo de texto em um documento HTML. Ela também é importante para a acessibilidade, pois ajuda a tornar o conteúdo mais legível e compreensível para os usuários.



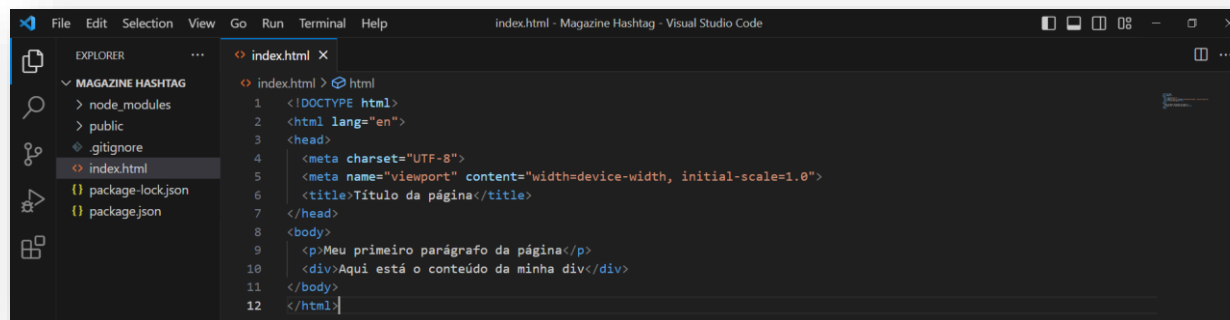
```
index.html - HashtagIntensivao - Visual Studio Code
index.html x
index.html > html
1 <!DOCTYPE html>
2 <html lang="en">
3 <head>
4   <meta charset="UTF-8">
5   <meta name="viewport" content="width=device-width, initial-scale=1.0">
6   <title>Título da Página</title>
7 </head>
8 <body>
9   <p>Meu primeiro texto da minha primeira página</p>
10 </body>
11 </html>
```



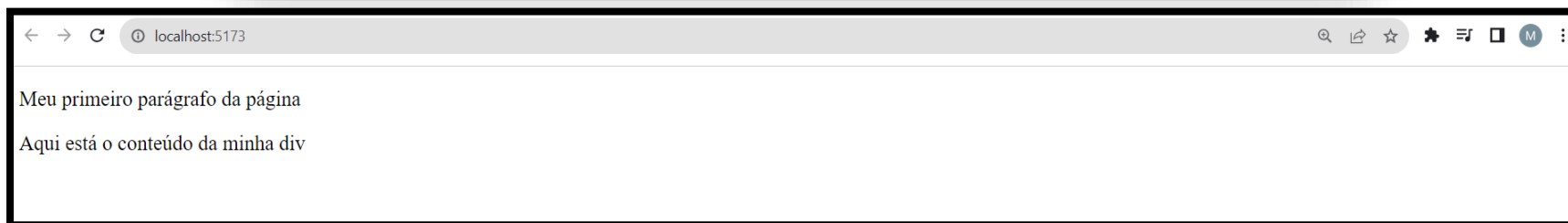
Estruturas do Projeto – HTML / CSS

A **tag <div>** é uma das tags mais conhecidas e utilizadas no HTML. Ela é um elemento de divisão que permite agrupar e organizar o conteúdo de uma página. A **<div>** não possui um significado especial, mas é amplamente utilizada para criar blocos de conteúdo e aplicar estilos CSS a esses blocos.

A sintaxe da **tag <div>** é simples: você abre a tag com **<div>** e fecha com **</div>**. Todo o conteúdo que estiver entre essas tags será considerado parte da divisão. Por padrão, a **<div>** é um elemento de bloco, o que significa que ela gera uma quebra de linha automática. Isso ocorre porque o display padrão da **<div>** é **display: block**.



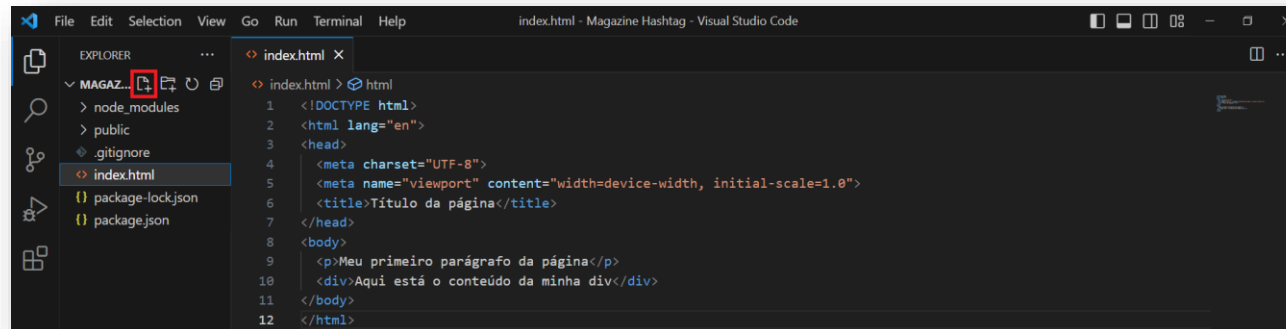
```
1 <!DOCTYPE html>
2 <html lang="en">
3 <head>
4   <meta charset="UTF-8">
5   <meta name="viewport" content="width=device-width, initial-scale=1.0">
6   <title>Título da página</title>
7 </head>
8 <body>
9   <p>Meu primeiro parágrafo da página</p>
10  <div>Aqui está o conteúdo da minha div</div>
11 </body>
12 </html>
```



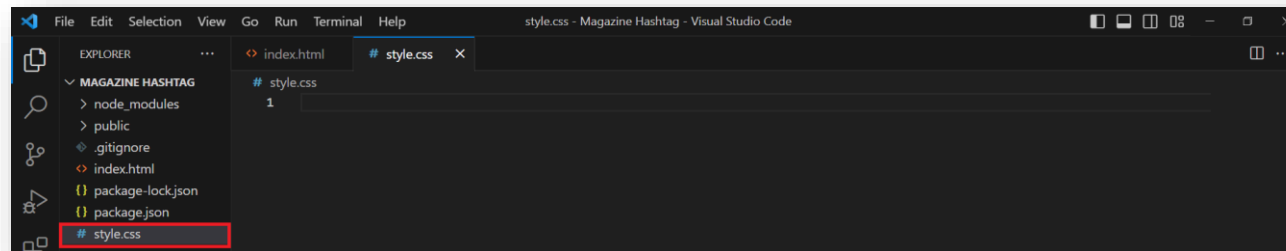
Estruturas do Projeto – HTML / CSS

Nesse momento iremos criar nosso arquivo **style.css**. O arquivo style.css é um arquivo que contém as regras de **estilo em CSS** (Cascading Style Sheets) para um documento HTML. Ele é usado para adicionar estilos e personalizar a aparência dos elementos HTML em uma página da web.

No VS Code clique no ícone adicionar novo arquivo:



- Crie um **arquivo style.css** na mesma pasta onde o documento HTML está localizado. Certifique-se de usar a extensão .css para indicar que é um arquivo CSS.



Estruturas do Projeto – HTML / CSS

Você pode adicionar as regras de estilo no arquivo style.css para estilizar os elementos HTML. Por exemplo, para definir a cor do texto de todos os parágrafos (**<p>**) como vermelho, você pode adicionar a seguinte regra CSS no arquivo style.css:

```
# style.css > ...  
1  p {  
2    color: red;  
3  }
```

No documento HTML, adicione a seguinte linha dentro da **tag <head>** para vincular o arquivo style.css ao documento HTML:

```
<? index.html • # style.css  
<? index.html > html > head  
1  <!DOCTYPE html>  
2  <html lang="en">  
3  <head>  
4    <meta charset="UTF-8">  
5    <meta name="viewport" content="width=device-width, initial-scale=1.0">  
6    <title>Título da página</title>  
7    <link rel="stylesheet" href="style.css">  
8  </head>  
9  <body>  
10   <p>Meu primeiro parágrafo da página</p>  
11   <div>Aqui está o conteúdo da minha div</div>  
12 </body>  
13 </html>
```

Estruturas do Projeto – HTML / CSS

Essa linha de código adiciona um link para o arquivo `style.css` na página HTML, permitindo que as regras de estilo sejam aplicadas. O **`href="style.css"`** é um atributo da **tag `<link>`** que especifica o local do arquivo CSS que será vinculado ao documento HTML.

Depois de adicionar as regras de estilo no arquivo `style.css`, salve-o. As regras de estilo serão aplicadas automaticamente aos elementos HTML correspondentes quando a página for carregada no navegador.



É importante lembrar que você pode adicionar várias regras de estilo no arquivo `style.css` para estilizar diferentes elementos HTML. Você pode utilizar seletores CSS para segmentar elementos específicos, como classes, IDs, tags HTML, etc., e aplicar estilos específicos a esses elementos.

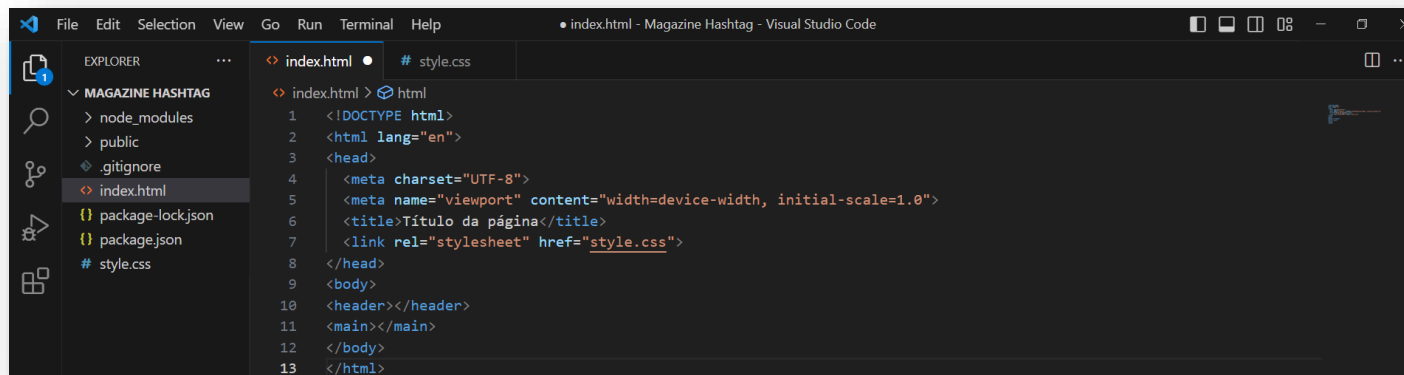
Além disso, você pode utilizar várias propriedades CSS para definir diferentes estilos, como cor de texto, cor de fundo, tamanho da fonte, margens, preenchimento, alinhamento, bordas e muitos outros.

Estruturas do Projeto – HTML / CSS

No projeto utilizaremos o HTML semântico que é uma abordagem de marcação que visa reforçar o significado e a estrutura do conteúdo em uma página da web, em vez de apenas definir sua aparência visual. Ele utiliza tags semânticas que descrevem claramente o propósito e o contexto do conteúdo, tornando-o mais compreensível para os desenvolvedores, mecanismos de busca e leitores de tela.

Ao usar o HTML semântico, você está criando um código mais organizado, de fácil manutenção e acessível. Além disso, os motores de busca tendem a indexar melhor o conteúdo de páginas que utilizam tags semânticas corretamente, o que pode melhorar o ranqueamento nos resultados de pesquisa.

- A **tag <header>** é usada no HTML para representar o cabeçalho de um documento ou de uma seção. Ela geralmente contém elementos como o logotipo, o título da página e a navegação principal.
- A **tag <main>** é usada no HTML para representar o conteúdo principal de um documento ou de uma seção. Ela deve ser usada apenas uma vez em cada página e deve conter o conteúdo central e relevante para os usuários.



```
File Edit Selection View Go Run Terminal Help • index.html - Magazine Hashtag - Visual Studio Code
EXPLORER
MAGAZINE HASHTAG
  > node_modules
  > public
  .gitignore
  index.html
  package-lock.json
  package.json
  # style.css
index.html
1 <!DOCTYPE html>
2 <html lang="en">
3 <head>
4   <meta charset="UTF-8">
5   <meta name="viewport" content="width=device-width, initial-scale=1.0">
6   <title>Título da página</title>
7   <link rel="stylesheet" href="style.css">
8 </head>
9 <body>
10 <header></header>
11 <main></main>
12 </body>
13 </html>
```

Parte 8

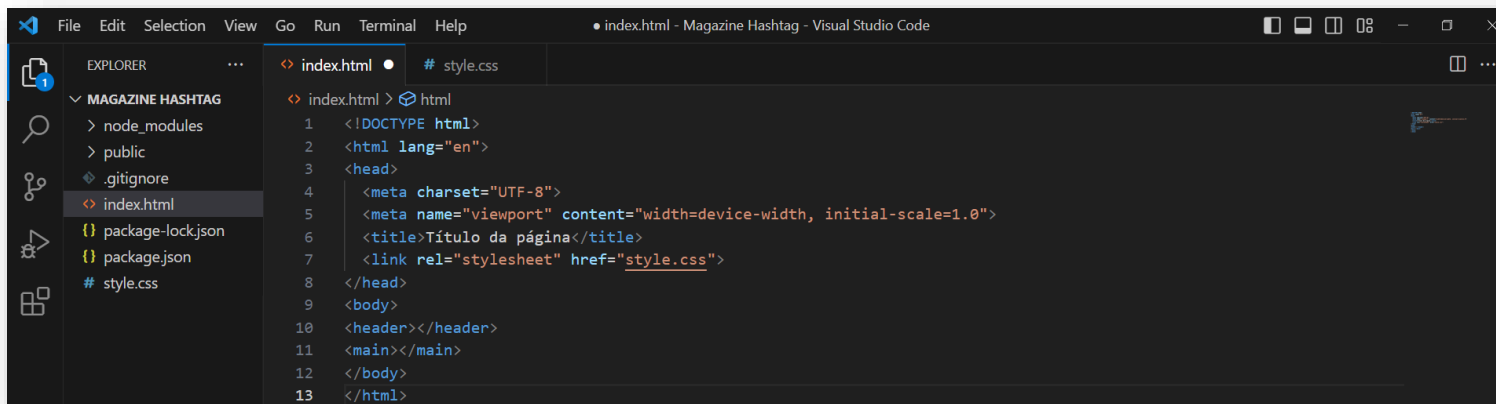
Desenvolvimento do Card de Produtos

Desenvolvimento do Card de Produtos

No projeto utilizaremos o HTML semântico que é uma abordagem de marcação que visa reforçar o significado e a estrutura do conteúdo em uma página da web, em vez de apenas definir sua aparência visual. Ele utiliza tags semânticas que descrevem claramente o propósito e o contexto do conteúdo, tornando-o mais compreensível para os desenvolvedores, mecanismos de busca e leitores de tela.

Ao usar o HTML semântico, você está criando um código mais organizado, de fácil manutenção e acessível. Além disso, os motores de busca tendem a indexar melhor o conteúdo de páginas que utilizam tags semânticas corretamente, o que pode melhorar o ranqueamento nos resultados de pesquisa.

- A **tag <header>** é usada no HTML para representar o cabeçalho de um documento ou de uma seção. Ela geralmente contém elementos como o logotipo, o título da página e a navegação principal.
- A **tag <main>** é usada no HTML para representar o conteúdo principal de um documento ou de uma seção. Ela deve ser usada apenas uma vez em cada página e deve conter o conteúdo central e relevante para os usuários.



```
File Edit Selection View Go Run Terminal Help
• index.html - Magazine Hashtag - Visual Studio Code

EXPLORER
MAGAZINE HASHTAG
  > node_modules
  > public
  .gitignore
  index.html
  package-lock.json
  package.json
  # style.css

index.html
1 <!DOCTYPE html>
2 <html lang="en">
3 <head>
4   <meta charset="UTF-8">
5   <meta name="viewport" content="width=device-width, initial-scale=1.0">
6   <title>Título da página</title>
7   <link rel="stylesheet" href="style.css">
8 </head>
9 <body>
10 <header></header>
11 <main></main>
12 </body>
13 </html>
```

Desenvolvimento do Card de Produtos

Vamos trabalhar na parte principal do nosso projeto, que é desenvolver os Cards de Produtos. Para isso, vamos usar a **tag <section>**. Essa tag é usada para dividir o conteúdo da página em seções lógicas ou áreas temáticas. Ela nos ajuda a organizar e estruturar o conteúdo da página da web, facilitando para os desenvolvedores e usuários entenderem a hierarquia e a organização do conteúdo. No nosso caso, a **tag <section>** será responsável por organizar os nossos Cards de Produtos.

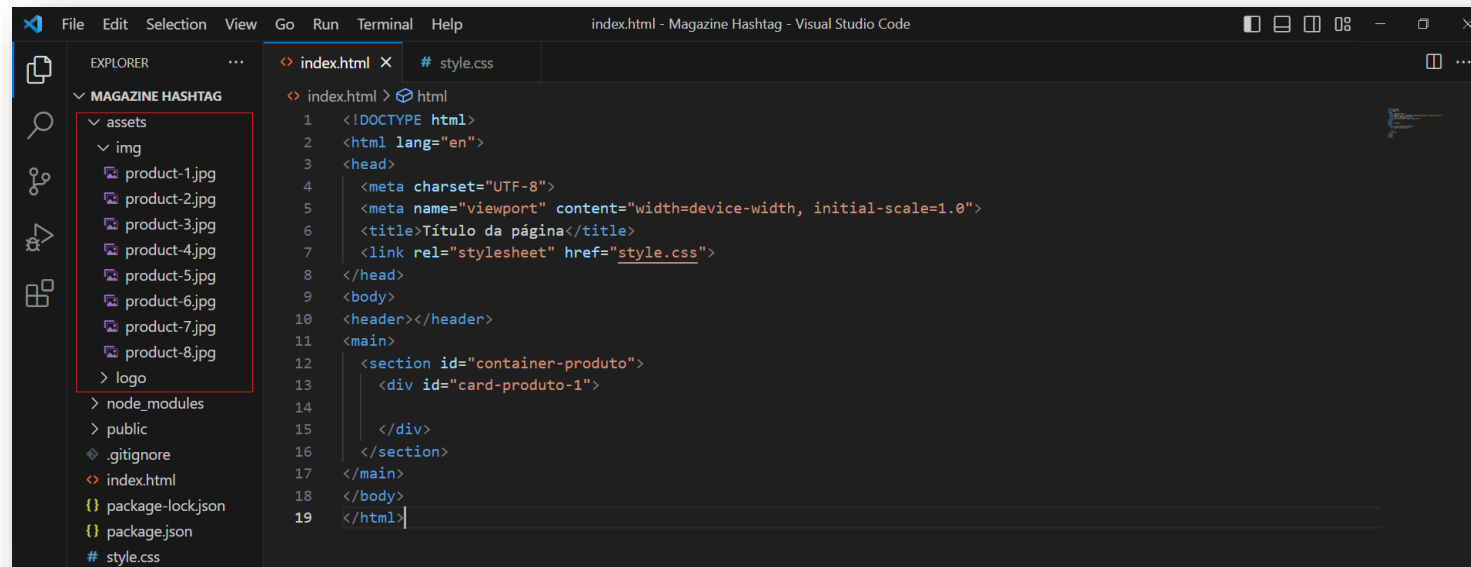
```
<> index.html • # style.css
<> index.html > html
1  <!DOCTYPE html>
2  <html lang="en">
3  <head>
4    <meta charset="UTF-8">
5    <meta name="viewport" content="width=device-width, initial-scale=1.0">
6    <title>Título da página</title>
7    <link rel="stylesheet" href="style.css">
8  </head>
9  <body>
10 <header></header>
11 <main>
12   <section id="container-produto"></section>
13 </main>
14 </body>
15 </html>
```

O **atributo "id"** é um atributo HTML que é usado para identificar exclusivamente um elemento em uma página da web. Ele fornece um identificador único para o elemento, permitindo que seja referenciado por meio de scripts ou estilos CSS. O valor do atributo "id" deve ser único em toda a página, ou seja, nenhum outro elemento deve ter o mesmo valor de **"id"**. O **atributo "id"** é muito útil para selecionar e manipular elementos específicos usando JavaScript ou CSS.

Desenvolvimento do Card de Produtos

Para estruturar nossos cards, vamos adicionar as informações que desejamos dentro deles, como imagem, descrição e preço. Você terá acesso às imagens dos produtos disponíveis para download, mas fique à vontade para usar as imagens que preferir para a construção do seu site de comércio eletrônico. Os cards são elementos visuais que exibem informações sobre os produtos de forma organizada e atraente. Eles são muito utilizados em sites de vendas online para apresentar os produtos aos usuários de maneira clara e atrativa.

- Crie uma pasta chamada **"assets"** dentro da raiz do projeto Magazine Hashtag. Dentro dela crie uma pasta **"img"**, local que você irá adicionar as imagens dos produtos. E uma pasta chamada **"logo"**, que irá conter a imagem do logo do nosso site.



Desenvolvimento do Card de Produtos

A **tag ** é usada para exibir imagens em uma página da web. Ela possui dois atributos principais: **src** e **alt**.

```

```

O **atributo src** é usado para especificar o caminho ou URL da imagem que você deseja exibir. Por exemplo, se a imagem estiver localizada na mesma pasta do arquivo HTML, você pode usar apenas o nome do arquivo. Se a imagem estiver em uma pasta diferente, você precisará especificar o caminho completo até a imagem.

O **atributo alt** é usado para fornecer um texto alternativo para a imagem. Esse texto é exibido caso a imagem não possa ser carregada ou se o usuário estiver usando um leitor de tela. O texto alternativo é importante para acessibilidade e também é útil para mecanismos de busca entenderem o conteúdo da imagem.

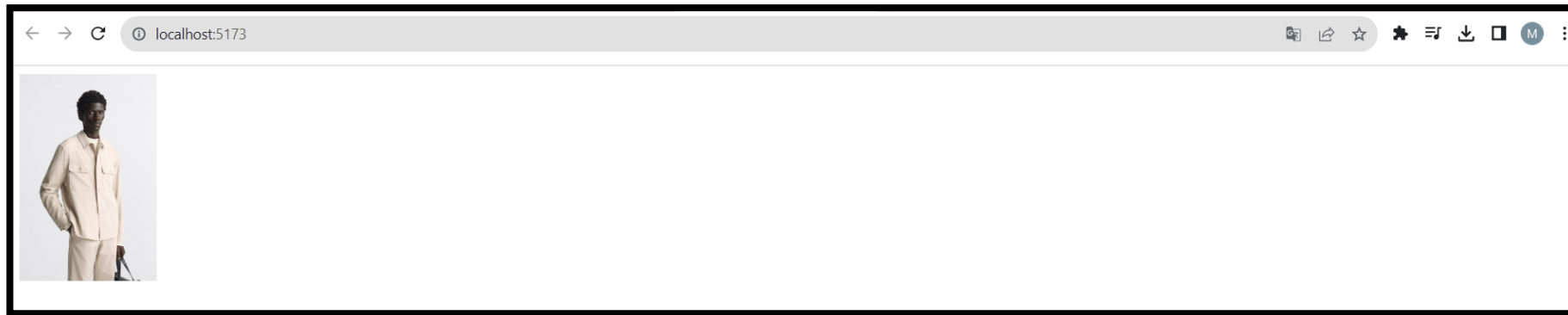
```
11 <main>
12   <section id="container-produto">
13     <div id="card-produto-1">
14       
15     </div>
16   </section>
17 </main>
```

Desenvolvimento do Card de Produtos

Para visualizarmos melhor o nosso card, podemos utilizar uma estilização em linha temporária no nosso elemento de imagem. Isso significa que iremos adicionar estilos diretamente na **tag ** para obter uma visualização temporária do card.

```

```



- O **estilo style="height: 200px;"** define a altura da imagem como 200 pixels. Isso significa que a imagem será exibida com uma altura de 200 pixels. Você pode ajustar esse valor conforme necessário para atender aos requisitos de design do seu site.

Desenvolvimento do Card de Produtos

Vamos incluir os elementos adicionais do nosso Card, como a marca, descrição e preço, além de implementar um botão com funcionalidade inteligente para adicionar o item ao carrinho de compras ao longo do projeto.

A **tag <button>** é uma tag HTML que cria um botão interativo em uma página da web. Ela é usada para criar botões clicáveis que podem executar ações quando são clicados pelo usuário. Os botões podem ser usados para enviar formulários, executar scripts, navegar para outras páginas, entre outras funcionalidades. A **tag <button>** pode conter texto, imagens ou outros elementos HTML como seu conteúdo. É possível adicionar atributos como id, class e onclick para personalizar o comportamento e a aparência do botão.

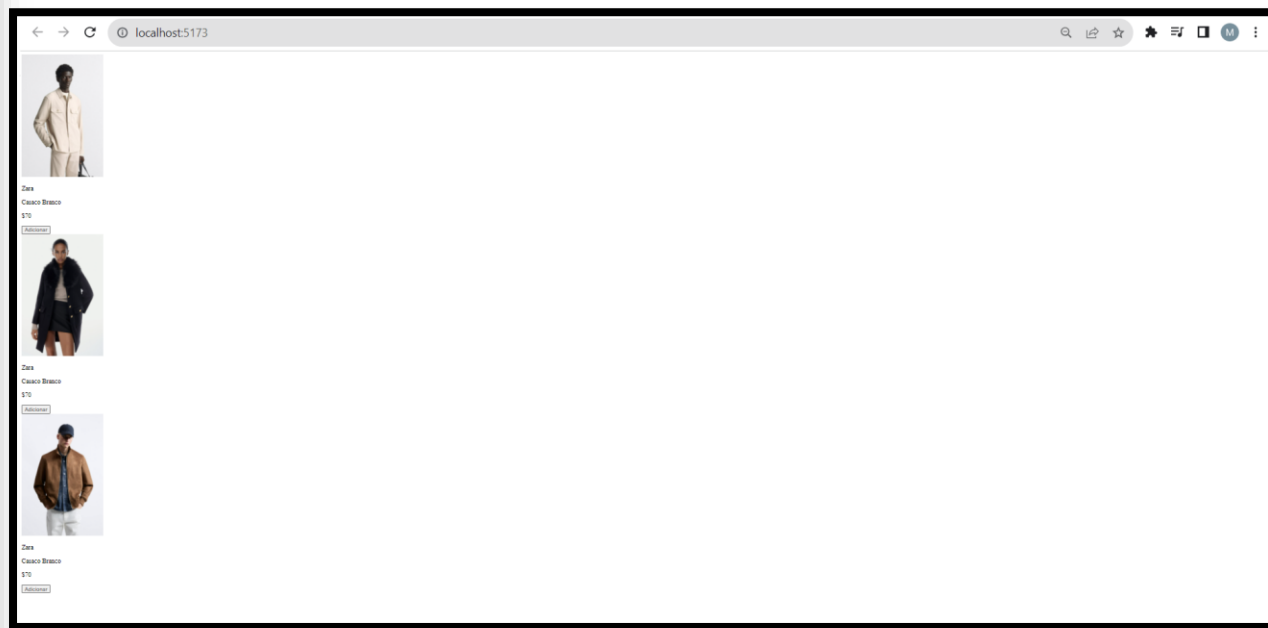
```
11 <main>
12   <section id="container-produto">
13     <div id="card-produto-1">
14       
15       <p>Zara</p>
16       <p>Casaco Branco</p>
17       <p>$70</p>
18       <button>Adicionar</button>
19     </div>
20   </section>
21 </main>
```



Desenvolvimento do Card de Produtos

No contexto específico do HTML, criar vários elementos iguais, como vários elementos de cards, um após o outro, pode resultar em um código HTML extenso e difícil de gerenciar. Isso ocorre porque cada card teria que ser repetido manualmente no código HTML, o que pode levar a erros e dificultar a manutenção do código.

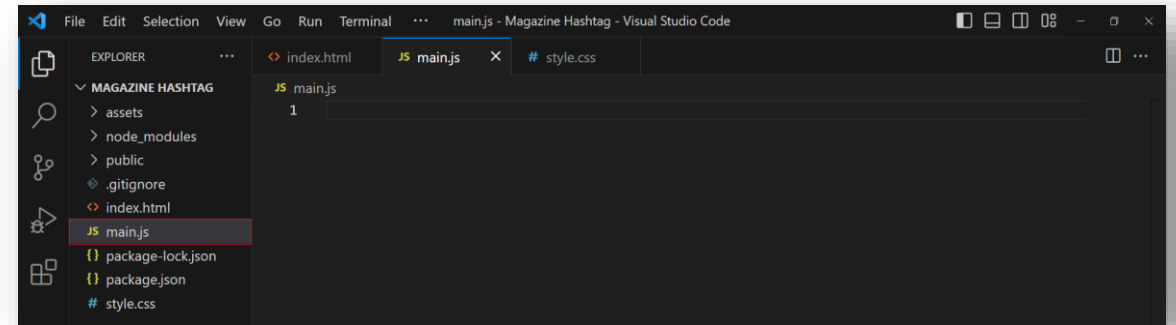
```
index.html X # style.css
index.html > html
5 <meta name="viewport" content="width=device-width, initial-scale=1.0">
6 <title>Título da página</title>
7 <link rel="stylesheet" href="style.css">
8 </head>
9 <body>
10 <header></header>
11 <main>
12 <section id="container-produto">
13 <div id="card-produto-1">
14 
15 <p>Zara</p>
16 <p>Casaco Branco</p>
17 <p>$70</p>
18 <button>Adicionar</button>
19 </div>
20 <div id="card-produto-2">
21 
22 <p>Zara</p>
23 <p>Casaco Branco</p>
24 <p>$70</p>
25 <button>Adicionar</button>
26 </div>
27 <div id="card-produto-1">
28 
29 <p>Zara</p>
30 <p>Casaco Branco</p>
31 <p>$70</p>
32 <button>Adicionar</button>
33 </div>
34 </section>
35 </main>
36 </body>
37 </html>
```



Desenvolvimento do Card de Produtos

Uma maneira de criar vários elementos de cards dinamicamente é usando JavaScript. Então vamos criar um arquivo chamado **"main.js"**.

A extensão de arquivo **".js"** é usada para arquivos que contêm código JavaScript, ou seja, contêm instruções e lógica que são interpretadas e executadas pelo navegador.



Vamos começar a entender como o Javascript irá nos auxiliar para colocar inteligência ao nosso site. Em JavaScript, uma **variável** é um espaço de armazenamento que pode conter um valor. Ela é usada para armazenar e manipular dados durante a execução de um programa. As **variáveis** podem ser pensadas como caixas ou recipientes que guardam informações.

```
JS main.js > ...  
1  const nomeProduto = "Casaco Branco";
```

Declarando uma variável chamada **nomeProduto** utilizando a palavra-chave **const**. A palavra-chave **const** é usada para declarar uma variável com um valor que não pode ser alterado posteriormente. Em outras palavras, uma vez que atribuímos um valor a uma variável declarada com **const**, esse valor não pode ser modificado. No exemplo, estamos atribuindo o **valor "Casaco Branco"** à variável **nomeProduto**. Isso significa que a variável **nomeProduto** sempre terá o valor "Casaco Branco" durante a execução do programa.

Desenvolvimento do Card de Produtos

As variáveis em JavaScript podem armazenar diferentes tipos de dados, como números, strings, booleanos, objetos e até mesmo funções.

```
const idade = 25; // número
const sobrenome = "Silva"; // string
const isEstudante = true; // booleano
const pessoa = { nome: "Maria", idade: 30 }; // objeto
```

Em JavaScript, uma **string** é uma sequência de caracteres que representa **texto**. Ela pode conter letras, números, símbolos e espaços em branco. As strings são usadas para armazenar e manipular texto em um programa.

Existem duas formas de representar uma string em JavaScript: usando **aspas duplas** (") ou **aspas simples** ('). Ambas as formas são válidas e podem ser usadas para criar strings.

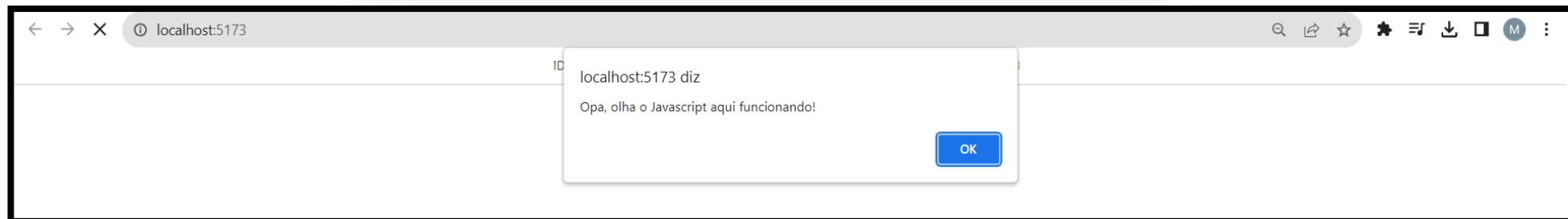
```
JS main.js > ...
1  const nomeProduto = "Casaco Branco";
2  const marca = 'Zara';
```

Desenvolvimento do Card de Produtos

Uma **ação** é uma atividade específica que ocorre em resposta a um evento ou a uma solicitação, e vamos criar uma através de uma **função** chamada **alert()**.

A **função alert()** é usada para exibir uma caixa de diálogo com uma mensagem para o usuário. Essa mensagem pode ser uma string ou um valor que pode ser convertido em uma string. Essa função é útil para fornecer informações ou interagir com o usuário durante a execução de um programa JavaScript em um ambiente de navegador.

```
JS main.js
1  // const nomeProduto = "Casaco Branco";
2  // const marca = 'Zara';
3  // const preco = 70;
4  // const nomeArquivoImagem = "product-1.jpg";
5
6  alert("Opa, olha o Javascript aqui funcionando!");
```



Desenvolvimento do Card de Produtos

No seu arquivo HTML, você pode adicionar o código JavaScript de duas maneiras diferentes:

a. Adicionando o código diretamente no arquivo HTML:

- Abra a **tag <script>** e feche-a com **</script>**.
- Dentro dessas tags, coloque o código JavaScript que você deseja executar.

b. Linkando um arquivo JavaScript externo:

- Utilize o **atributo src** da tag <script> para especificar o caminho para o arquivo JavaScript externo.
- Por exemplo: **<script src="caminho/para/o/arquivo.js"></script>**.
- Certifique-se de substituir "caminho/para/o/arquivo.js" pelo caminho correto do seu arquivo JavaScript.

É recomendado colocar o código JavaScript na seção <head> do arquivo HTML para separá-lo do conteúdo da página. No entanto, se você colocar o código JavaScript na seção <body>, a página poderá carregar mais rapidamente.

```
9 <body>
10 <header></header>
11 <main>
12 <section id="container-produto">
13   <div id="card-produto-1">
14     
15     <p>Zara</p>
16     <p>Casaco Branco</p>
17     <p>$70</p>
18     <button>Adicionar</button>
19   </div>
20 </section>
21 </main>
22 <script src="main.js"></script>
23 </body>
```

Desenvolvimento do Card de Produtos

Para a construção dos nossos produtos vamos utilizar um tipo de dado em Javascript, chamada de **Objeto**.

Um objeto é uma **estrutura de dados** que permite armazenar informações relacionadas em **pares de chave e valor**. Pense em um objeto como uma caixa que pode conter várias propriedades. Cada propriedade é composta por uma chave (um nome) e um valor associado a essa chave.

Os **objetos** em JavaScript são muito flexíveis e poderosos, permitindo que você armazene e manipule dados de forma organizada. Eles são amplamente utilizados para representar entidades do mundo real, como pessoas, produtos, entre outros.

```
JS main.js > ...  
1  const produto1 = {  
2      nome: "Casaco Branco",  
3      marca: "Zara",  
4      preco: 70,  
5      nomeArquivoImagem: "product-1.jpg",  
6  };
```

Desenvolvimento do Card de Produtos

Para adicionar inteligência a uma página da web usando JavaScript, podemos utilizar várias funcionalidades da linguagem.

Uma delas é a **Manipulação do DOM (Document Object Model)**, o JavaScript pode ser usado para interagir com o DOM, que representa a estrutura da página HTML. Isso permite que você altere o conteúdo, estilos e comportamento da página de forma dinâmica.

Utilizando esse recurso, vamos recortar nossos elementos que representam o nosso **Card de Produtos** no arquivo **index.html** e armazenar essas informações dentro de uma variável no nosso arquivo **main.js**. Para isso vamos utilizar o conceito de **String Template (Imagem 1)**.

```
JS main.js > ...
1  const produto1 = {
2    nome: "Casaco Branco",
3    marca: "Zara",
4    preco: 70,
5    nomeArquivoImagem: "product-1.jpg",
6  };
7
8  const cartaoProduto = `<div id="card-produto-1">
9    
14   <p>Zara</p>
15   <p>Casaco Branco</p>
16   <p>$70</p>
17   <button>Adicionar</button>
18   </div>`
```

1

Desenvolvimento do Card de Produtos

String template, também conhecido como template literal, é uma funcionalidade do JavaScript que permite criar strings de forma mais flexível e legível, além de permitir a interpolação de valores e expressões dentro da string. Em vez de usar aspas simples (") ou duplas (") para delimitar uma string, utiliza-se **crases** (``).

- Para inserir uma variável ou expressão dentro de uma string, utiliza-se **`${}`**.
- Com o string template, é possível criar strings que abrangem várias linhas sem a necessidade de concatenação ou caracteres de escape. Basta inserir as quebras de linha diretamente no template.
- Além da interpolação de valores, é possível incluir expressões dentro do template. As expressões são avaliadas e o resultado é convertido em uma string.

Desenvolvimento do Card de Produtos

Vamos deixar nossos Cards dinâmicos para que eles possam exibir as informações corretas de cada produto. Para isso, iremos utilizar as informações armazenadas no **objeto de produto** e exibi-las no nosso elemento HTML chamado "**cartaoProduto**".

```
JS main.js > ...
1  const produto1 = {
2    nome: "Casaco Branco",
3    marca: "Zara",
4    preco: 70,
5    nomeArquivoImagem: "product-1.jpg",
6  };
7
8  const cartaoProduto = `<div id="card-produto-1">
9    
14   <p>${produto1.marca}</p>
15   <p>${produto1.nome}</p>
16   <p>${produto1.preco}</p>
17   <button>Adicionar</button>
18   </div>`
```

Vamos analisar cada parte do código:

- ****:
 - O atributo src está sendo preenchido com o caminho da imagem, que é obtido a partir da propriedade nomeArquivoImagem do objeto produto1.
- **<p>\${produto1.marca}</p>**:
 - Exibe a marca do produto, obtida a partir da propriedade marca do objeto produto1.
- **<p>\${produto1.nome}</p>**:
 - Exibe o nome do produto, obtido a partir da propriedade nome do objeto produto1.
- **<p>\${produto1.preco}</p>**:
 - Exibe o preço do produto, obtido a partir da propriedade preco do objeto produto1.

Desenvolvimento do Card de Produtos

Para renderizar o Card no navegador seguimos na construção do nosso código no arquivo main.js:

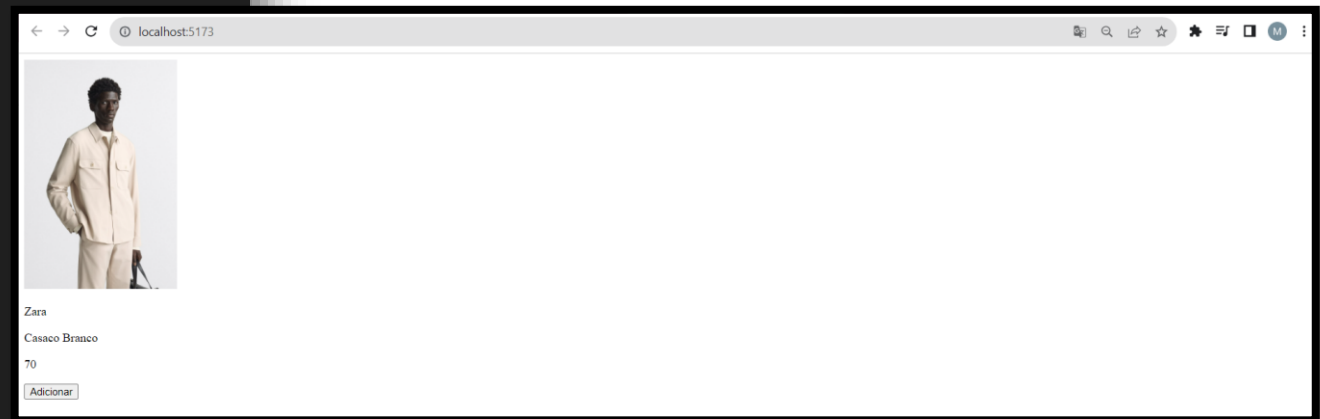
```
20 document.getElementById("container-produto").innerHTML += cartaoProduto;
```

- **document.getElementById("container-produto")**: O `document` representa o documento HTML da página. O **método `getElementById`** é usado para selecionar um elemento HTML com base no seu ID. Nesse caso, estamos procurando um elemento com o ID **"container-produto"**.
- **.innerHTML**: A propriedade `innerHTML` permite acessar ou modificar o conteúdo HTML dentro de um elemento. Ao usar essa propriedade, podemos obter ou alterar o código HTML dentro do elemento selecionado.
- **+=**: O operador `+=` é usado para adicionar algo ao conteúdo HTML existente dentro do elemento. Nesse caso, estamos adicionando o valor da variável `cartaoProduto` ao conteúdo HTML dentro do elemento com o ID "container-produto".
- **cartaoProduto**: Essa é uma variável que contém um código HTML para representar um cartão de produto. O valor dessa variável será adicionado ao conteúdo HTML existente dentro do elemento "container-produto".

Desenvolvimento do Card de Produtos

Em resumo, o **código** `document.getElementById("container-produto").innerHTML += cartaoProduto`; está adicionando o código HTML do cartão de produto (armazenado na variável `cartaoProduto`) ao conteúdo HTML existente dentro do elemento com o ID "container-produto". Isso permite que o cartão de produto seja exibido na página, juntamente com outros elementos que já estão presentes no "container-produto".

```
JS main.js > ...
1  const produto1 = {
2    nome: "Casaco Branco",
3    marca: "Zara",
4    preco: 70,
5    nomeArquivoImagem: "product-1.jpg",
6  };
7
8  const cartaoProduto = `<div id="card-produto-1">
9    
14   <p>${produto1.marca}</p>
15   <p>${produto1.nome}</p>
16   <p>${produto1.preco}</p>
17   <button>Adicionar</button>
18 </div>`
19
20 document.getElementById("container-produto").innerHTML += cartaoProduto;
21
```



Desenvolvimento do Card de Produtos

Nesse momento, vamos construir todos os nossos produtos, ou seja, vamos criar uma variável para cada objeto que irá representar as informações de um Card de Produto. É importante lembrar que, ao trabalhar com variáveis em JavaScript, não podemos usar nomes iguais. Portanto, cada variável precisa ter um nome diferente.

Para criar as variáveis dos produtos, você pode seguir os seguintes passos:

- Escolha um nome único para cada variável que representará um produto. Por exemplo, você pode usar "produto1", "produto2", "produto3" e assim por diante.
- Cada variável deve ser declarada usando a palavra-chave `const` seguida pelo nome escolhido e o sinal de igual (=).
- Atribua um objeto com as informações do produto à variável.
- Vamos adicionar a propriedade "id" que representará o código do produto.

```
JS main.js > [🔗] cartaoProduto
1   const produto1 = {
2       id: 1,
3       nome: "Casaco Branco",
4       marca: "Zara",
5       preco: 70,
6       nomeArquivoImagem: "product-1.jpg",
7   };
8
9   const produto2 = {
10      id: 2,
11      nome: "Sobretudo Azul Marinho",
12      marca: "Zara",
13      preco: 110,
14      nomeArquivoImagem: "product-2.jpg",
15  };
```

Desenvolvimento do Card de Produtos

O próximo passo é criarmos uma variável que irá armazenar nossos produtos e para isso vamos conhecer a estrutura de **listas** do Javascript.

Uma **lista(array)** em JavaScript é uma estrutura de dados que permite **armazenar vários valores em uma única variável**. a lista `catalogo` é criada usando colchetes `[]`. Dentro dos colchetes, temos os elementos da lista, que são os objetos `produto1` e `produto2`. Esses objetos são separados por vírgulas.

```
17  const catalogo = [produto1, produto2];
```

A **lista catalogo** agora contém os **objetos produto1 e produto2**, e podemos acessar esses objetos usando índices. O **índice é a posição do elemento na lista**, começando do zero.

Por exemplo, `catalogo[0]` nos dá o primeiro elemento da lista, que é o objeto `produto1`, e `catalogo[1]` nos dá o segundo elemento da lista, que é o objeto `produto2`.

Podemos usar a lista `catalogo` para percorrer todos os produtos e realizar operações em cada um deles.

Desenvolvimento do Card de Produtos

No momento, temos as variáveis dos nossos produtos criadas separadamente, como `produto1`, `produto2`, etc. Em seguida, essas variáveis são armazenadas no nosso catálogo usando a lista `catalogo`.

No entanto, existe uma maneira alternativa de armazenar os produtos diretamente dentro da lista, sem a necessidade de criar variáveis separadas. Podemos definir a estrutura dos objetos dos produtos dentro da lista `catalogo` diretamente, como no exemplo abaixo:

```
JS main.js > [?] cartaoProduto
1  const catalogo = [
2      {
3          id: 1,
4          marca: 'Zara',
5          nome: 'Camisa Larga com Bolsos',
6          preco: 70,
7          imagem: 'product-1',
8          feminino: false,
9      },
10     {
11         id: 2,
12         marca: 'Zara',
13         nome: 'Casaco Reto com Lã',
14         preco: 85,
15         imagem: 'product-2',
16         feminino: true,
17     },
18 ];
```

Nesse exemplo, em vez de criar variáveis separadas para cada produto, definimos a estrutura dos objetos dos produtos diretamente dentro da lista `catalogo`. Cada objeto representa um produto e contém as propriedades `nome`, `preco`, `marca` e `imagem`.

Dessa forma, podemos acessar os produtos diretamente pela lista `catalogo` usando índices. Por exemplo, `catalogo[0]` nos dá o primeiro produto da lista, e `catalogo[1]` nos dá o segundo produto. Essa maneira alternativa de armazenar os produtos diretamente dentro da lista pode ser útil quando temos muitos produtos e não queremos criar várias variáveis separadas. Além disso, facilita a iteração e manipulação dos produtos usando loops ou outras operações.

Desenvolvimento do Card de Produtos

```
68 const cartaoProduto = `<div id="card-produto-1">
69   
74   <p>${catalogo[0].marca}</p>
75   <p>${catalogo[0].nome}</p>
76   <p>${catalogo[0].preco}</p>
77   <button>Adicionar</button>
78 </div>`
79
80 document.getElementById("container-produto").innerHTML += cartaoProduto;
```

Mas observe que ainda temos nosso Card fixo, não adicionamos o dinamismo para criar todos os Cards de todos os nossos produtos. Para isso, utilizamos o **loop for...of**.

O **loop for...of** é uma estrutura de controle que nos permite percorrer cada elemento de uma coleção, como um array, de forma mais simples e direta.

```
68 for (const produtoCatalogo of catalogo) {
69   const cartaoProduto = `<div id="card-produto-1">
70     
75     <p>${produtoCatalogo.marca}</p>
76     <p>${produtoCatalogo.nome}</p>
77     <p>${produtoCatalogo.preco}</p>
78     <button>Adicionar</button>
79   </div>`;
80
81   document.getElementById("container-produto").innerHTML += cartaoProduto;
82 }
```

Desenvolvimento do Card de Produtos

Finalizamos a primeira parte do nosso projeto com a estrutura do código, utilizando o **loop for...of** para percorrer cada **objeto do array catalogo**. A cada iteração do loop, a variável **produtoCatalogo** recebe um objeto do array catalogo. Em seguida, estamos criando uma string chamada **cartaoProduto** que representa o HTML do card do produto.

Dentro dessa string, temos várias **tags HTML**, como ``, `<p>` e `<button>`, que são preenchidas com os valores das **propriedades do objeto produtoCatalogo**. Por exemplo, `${produtoCatalogo.imagem}` é substituído pelo valor da propriedade imagem do objeto produtoCatalogo.

Essa string **cartaoProduto** representa o HTML completo do card do produto e pode ser utilizada para renderizar o card na página.

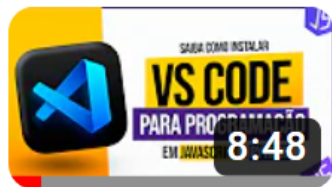


Parte Bônus

Vídeos

Complementares

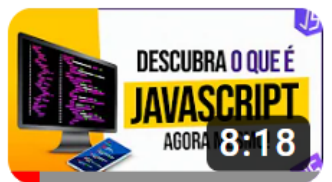
Vídeos Complementares



Instalação do VS Code para Programação em JavaScript e Python

Hashtag Programação

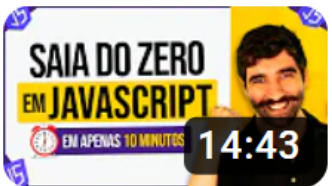
https://www.youtube.com/watch?v=iLraM_NZYfA&list=PLpdAy0tYrnKyLd8e6e3suYCYO3LKzCVzi&index=2



O que é JavaScript?

Hashtag Programação

<https://www.youtube.com/watch?v=oWSxNI8Pg9k&list=PLpdAy0tYrnKyLd8e6e3suYCYO3LKzCVzi>



Aprenda JavaScript em 10 min [Iniciantes]

Hashtag Programação

<https://www.youtube.com/watch?v=MombnkR3oD4&list=PLpdAy0tYrnKyLd8e6e3suYCYO3LKzCVzi&index=18>

Ainda não segue a gente no Instagram e nem é inscrito no nosso canal do Youtube? Então corre lá!



@hashtagtreinamentos



youtube.com/hashtag-treinamentos

