Plano de aula para o curso de desenvolvimento Web

Objetivo do curso:

Desenvolver habilidades de criação de sites com o uso das ferramentas HTML, CSS e JavaScript.

O curso será dividido em dois módulos:

Módulo 1: Conceitos de HTML e CSS

1. O que é Desenvolvimento Web?
   1. Front end, e back end
   2. Banco de dados
2. O que é HTML?
3. Apresentação da EDITOR DE TEXTO usada no curso.
   1. O que é VSCode?
   2. Configurações básicas
4. O que são Git e Github?
   1. O que é versionamento
   2. Como criar uma conta no Github?
   3. Como clonar um repositório?
   4. Como baixar o Github Desktop?
   5. O que é commit?
   6. Comandos básicos no Git
5. Estrutura básica de um HTML.
   1. Como definir o idioma de uma página?
6. O que são tags?
   1. O que são atributos de uma tag?
7. O que é “head” e o que devemos colocar nela?
   1. O que é charset?
   2. Definindo um título para a página
   3. Criando comentários no HTML
8. O que é “body”?
9. Apresentando o site modelo para a aula.
   1. Como desenhar template de site
10. Planejando a estrutura do site
    1. Definindo a index
11. O que é HTML semântico?
12. Definindo CSS
    1. Como aplicar CSS no HTML
       1. CSS dentro do HTML
       2. CSS na linha da tag
       3. CSS como arquivo separado
13. Criando o topo do site
    1. O que é “header”?
    2. Definindo DIV.
    3. Definindo altura e largura
    4. Trabalhando com cores
    5. Inserindo imagens
    6. Definindo padding, margim e border.
14. Criando a navegação para outras páginas do site
    1. Definido links
    2. Formatando a aparência dos links
       1. Mudando cor da fonte
       2. Tipo de fonte
       3. Tamanho de fonte
       4. Removendo o sublinhado da fonte
       5. Posicionando os links
    3. Criando o efeito de passar o mouse
15. Trabalhando com a section.
    1. Definindo display
    2. Dividindo espaços na section usando o flex.
16. Criando o rodapé do site
17. Trabalhando responsividade
    1. O que é site responsivo
    2. Como analisar os dispositivos
    3. Tipos de responsividade
18. Considerações finais

Sumário

[Introdução 4](#_Toc180571430)

[Desenvolvimento Web 5](#_Toc180571431)

[O que é HTML? 5](#_Toc180571432)

[Visual Studio Code 6](#_Toc180571433)

[Trabalhando com Git e Github 7](#_Toc180571434)

[Mas vamos entender primeiro o Git. 7](#_Toc180571435)

[Como instalar o Git? 7](#_Toc180571436)

[Por que apareceu entre parênteses a palavra máster? 9](#_Toc180571437)

[Criando uma pasta para guardar imagens 11](#_Toc180571438)

[Clonando um repositório remoto em um repositório local 15](#_Toc180571439)

[Vamos agora começar a estudar o HTML 18](#_Toc180571440)

[Mas o que são tags? 19](#_Toc180571441)

[Vamos começar a desenvolver a nossa primeira página? 21](#_Toc180571442)

[O que é CSS? 25](#_Toc180571443)

[Mas como aplicar CSS na página? 25](#_Toc180571444)

[Vamos começar a criar nossos estilos. 26](#_Toc180571445)

Curso de Desenvolvimento Web – Front End

# Introdução

# Desenvolvimento Web

Usamos esse termo para representar a criação de sites para a internet ou extranet. O profissional que atua nessa área precisa ter conhecimento, pelo menos, em HTML, CSS e JavaScript.

Os profissionais nessa área costumam ser divididos em três grupos:

* Front end: desenvolvedores com conhecimento em HTML, CSS, JavaScript e alguns frameworks.
* Back end: desenvolvedores com conhecimentos das ferramentas acima e com conhecimento em ferramentas para a criação e manipulação de banco de dados.

De uma forma mais clara, podemos entender que front end é a pessoa que cria a parte do site que fica visível para o usuário, enquanto o back end criar a parte que o usuário não vê, mas que faz toda a diferença em um site, por exemplo:

“Em um site de compras, o usuário quer enxergar o produto, mas ele não enxerga o banco de dados que cadastrou esse produto”.

Por falar em banco de dados, todas as informações sobre determinado item têm que ser guardadas. Imagine que você tenha uma coleção de carrinhos, vida de vários lugares e conseguida por vários meios, alguns deles engraçados. Em sua mente é guardado o nome de cada modelo e qual foi a forma que esse modelo foi adquirido. Isso é uma espécie de banco de dados onde são guardadas todas as informações relevantes para você sobre essa coleção de carrinhos.

# O que é HTML?

Antes de responder essa pergunta, vamos entender o que é programação.

Programar é dizer, com detalhes, o que o computador precisa fazer para resolver um problema. Para dizer a solução, usamos algoritmo.

Algoritmo é uma sequência de instruções que dizem exatamente o que o computador tem que fazer.

Quando desenvolvemos uma página para a internet, não estamos falando em programação, mas em dizer para o navegador onde cada coisa deve ficar. Os programas que usamos para criar algoritmos recebem o nome de linguagem de programação, como exemplo temos o Python, já o HTML é uma linguagem de marcação, ou seja, ela define qual o tipo de conteúdo será mostrado e onde ele será mostrado.

Desenvolver páginas para a internet não é o mesmo que programar computadores, porém existem situações em que podemos usar linguagem de programação para definir um comportamento em relação a atitude de um usuário, como por exemplo dizer ao usuário que ele está clicando em um local errado ou até mesmo permitir que o usuário decida qual cor ele quer que a página seja exibida, isso são apenas alguns dos exemplos.

Para isso usamos linguagem de programação e a mais comum é o JavaScript.

Concluindo, a linguagem HTML é usada como linguagem de marcação, definindo lugares onde cada coisa será apresentada.

HTML é a sigla de Hypertext Markup Language, que traduzindo seria, Linguagem de marcação de texto.

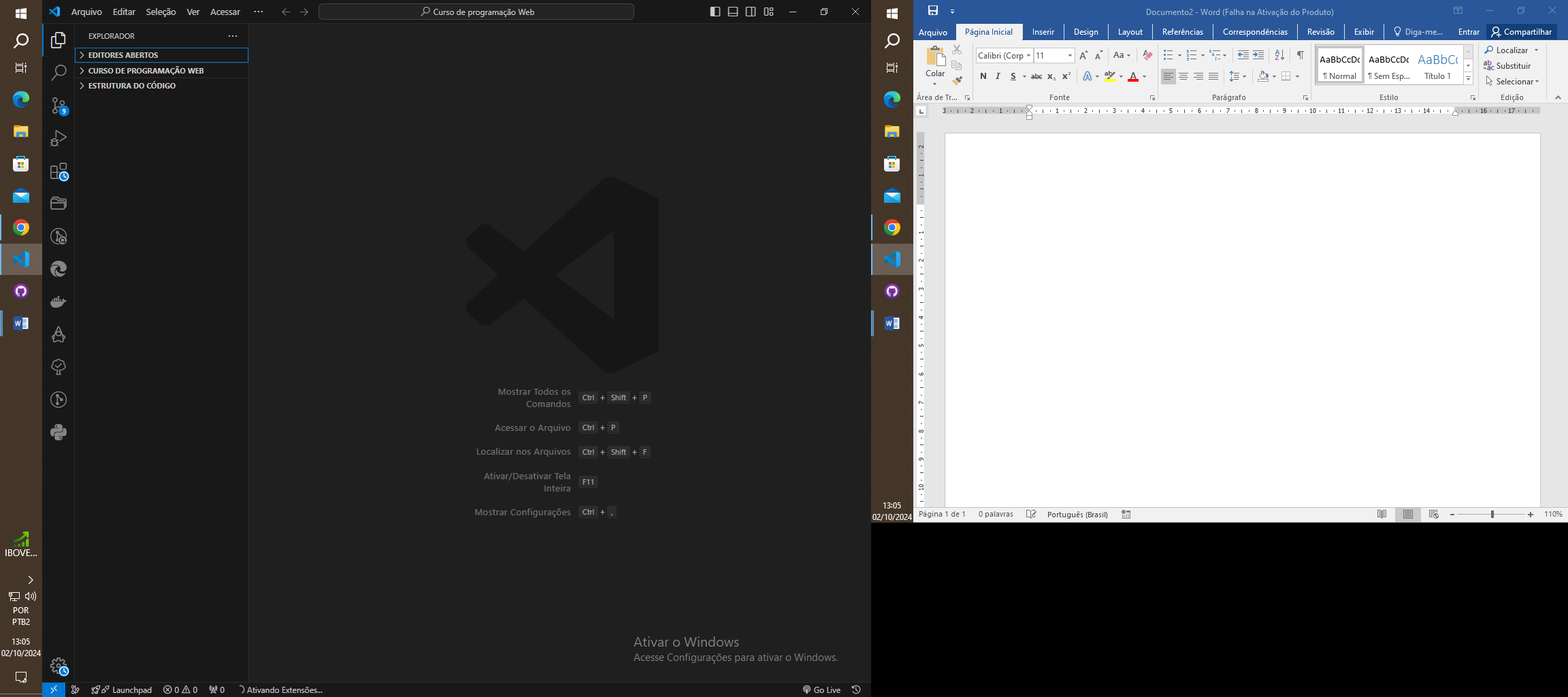
# Visual Studio Code

Para facilitar a vida dos desenvolvedores foram criados programas capazes de servir de base para as linguagens de programação e marcação, os chamados Editores de texto, eles têm como objetivo facilitar a vida dos desenvolvedores, seja completando partes do código (termo usado para as linhas de comando do algoritmo ou do HTML), seja apresentando possíveis falhas antes mesmo de finalizar o trabalho. O Visual Studio Code é um desses editores.

Com ele algumas tarefas se tornam bem simples. Em nosso curso usaremos ele por dois motivos o primeiro, ele é gratuito, o segundo, ele tem integração com ferramentas que usaremos durante esse módulo e em módulos posteriores.

Para baixar o VSCode (Visual Studio Code) basta apenas fazer uma pesquisa simples no navegador ou se preferir acesse: <https://code.visualstudio.com/> , escolha uma versão que se adeque a você.

Não ensinarei os detalhes desse editor, esse não é o objetivo do curso, porém se quiser aproveitar todas as possibilidades dele tem vários tutoriais na internet.



# Trabalhando com Git e Github

Uma das ferramentas que usaremos, junto ao VSCode, é o famoso Git. Ele é um programa que guarda as alterações feitas em um trabalho e nos possibilita acessar todas as versões, alteradas ou antes das alterações.

O interessante é que podemos decidir, depois de um tempo, se determinada alteração foi benéfica ou não, e podemos optar por mantê-las ou exclui-las.

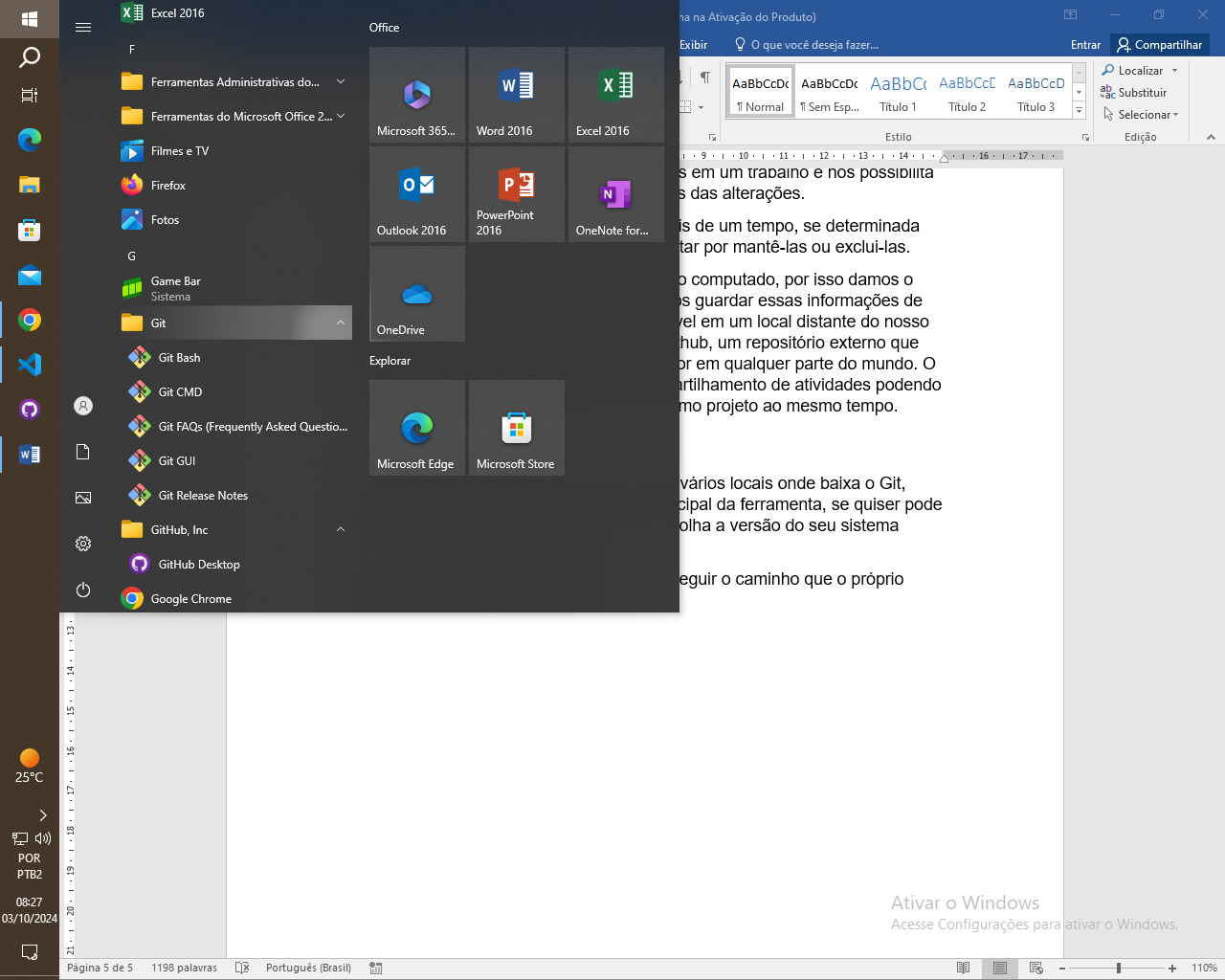
O Git salva essas alterações dentro do nosso computado, por isso damos o chamamos de repositório local, mas podemos guardar essas informações de forma remota, ou seja, deixando ela disponível em um local distante do nosso computador pessoal, para isso usamos o Github, um repositório externo que poderá ser acessado de qualquer computador em qualquer parte do mundo. O mais interessante é que ele permite o compartilhamento de atividades podendo trabalhar duas ou mais pessoas em um mesmo projeto ao mesmo tempo.

## Mas vamos entender primeiro o Git.

### Como instalar o Git?

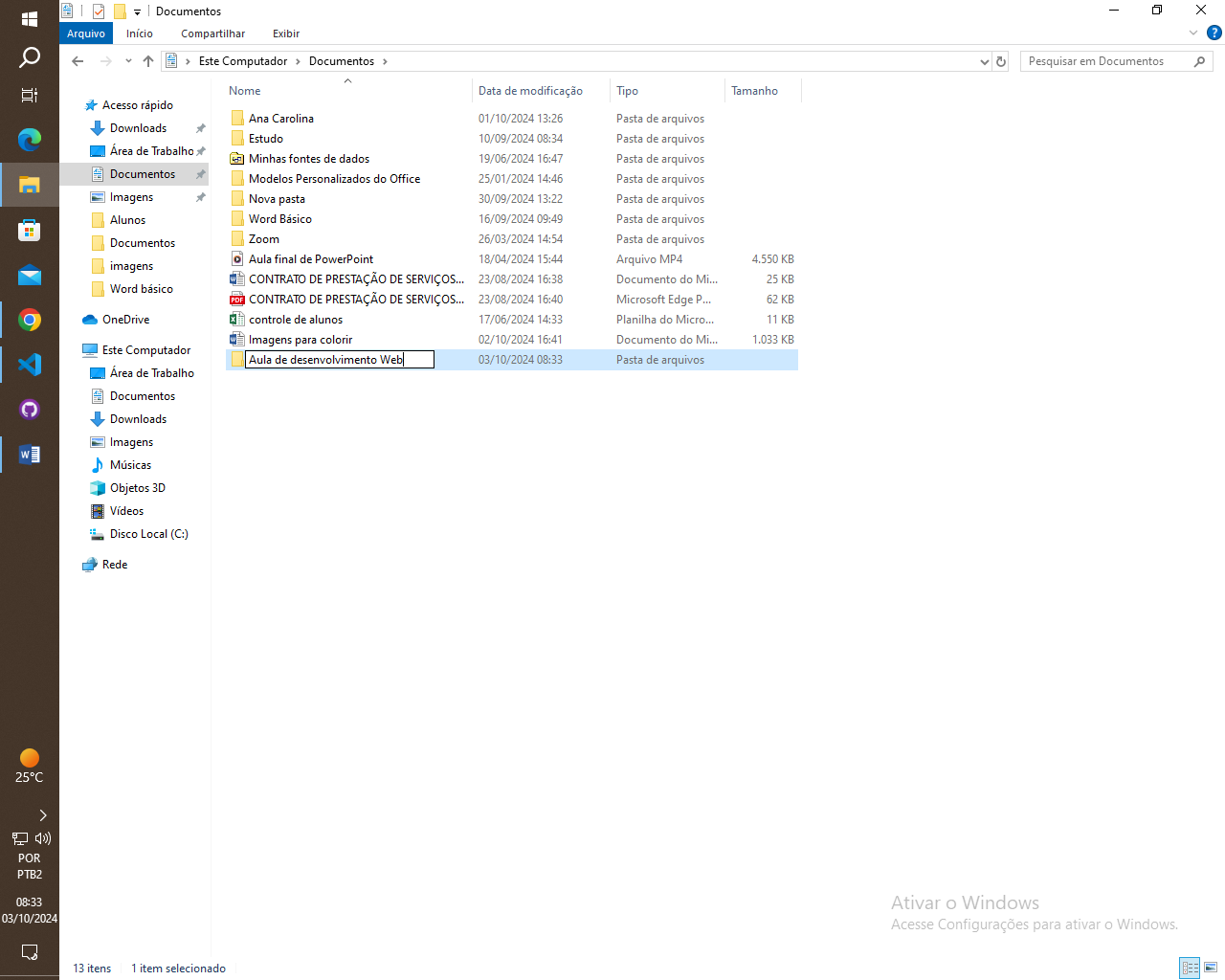
Fazendo uma pesquisa rápida encontramos vários locais onde baixa o Git, porém recomento baixar sempre do site principal da ferramenta, se quiser pode acessar: <https://git-scm.com/downloads>. Escolha a versão do seu sistema operacional e comece o download.

Na hora de instalar, não tem segredo, é só seguir o caminho que o próprio instalador mostra. Será instalado as seguintes opções, conforme a imagem:



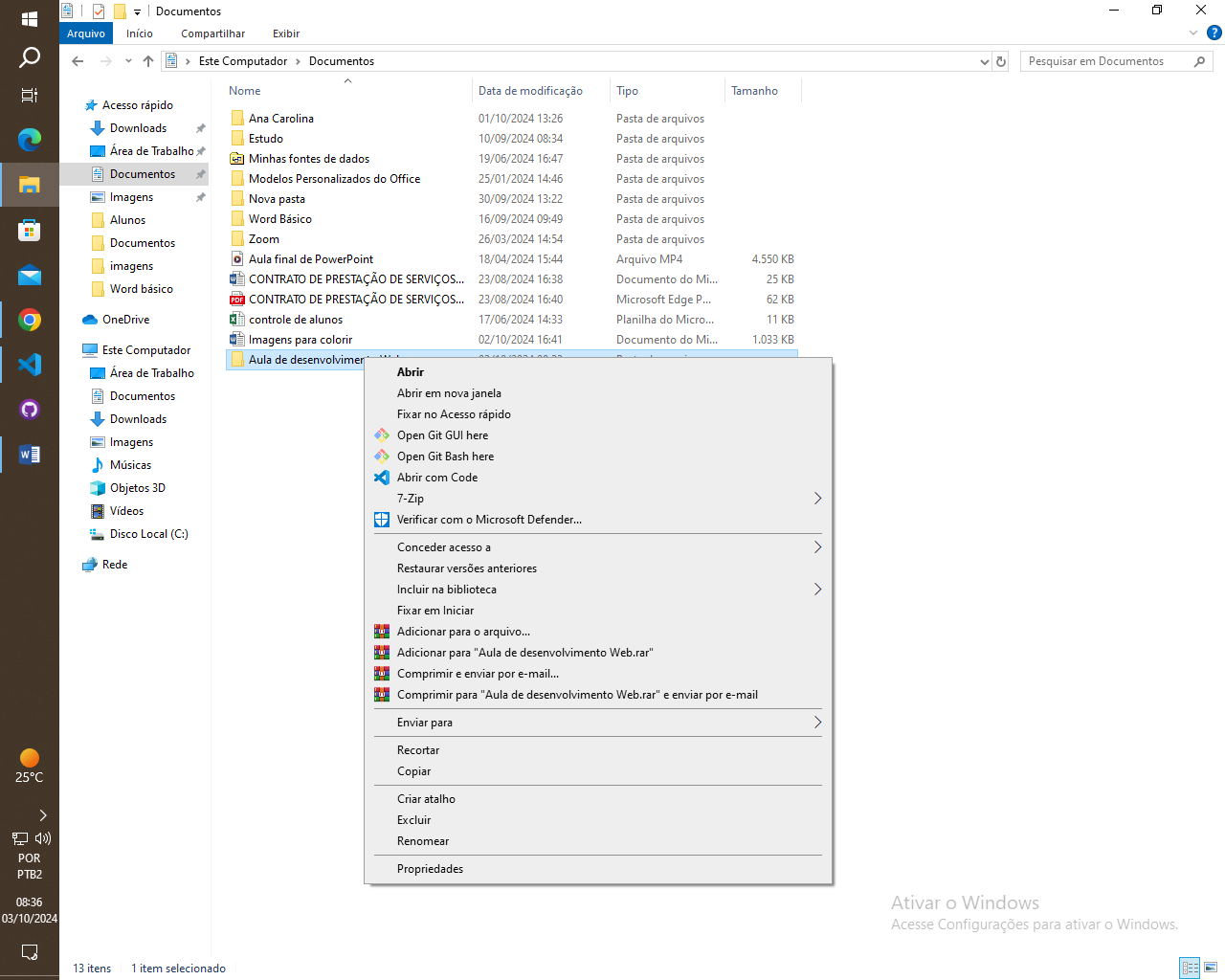
É fundamental que a instalação seja do site oficial, tente não instalar de terceiros.

Com o Git instalado, vamos criar uma pasta chamada “Aula de desenvolvimento Web”, poderá ser criada em qualquer lugar do computador, mas recomendo que seja dentro da pasta “Documentos”.

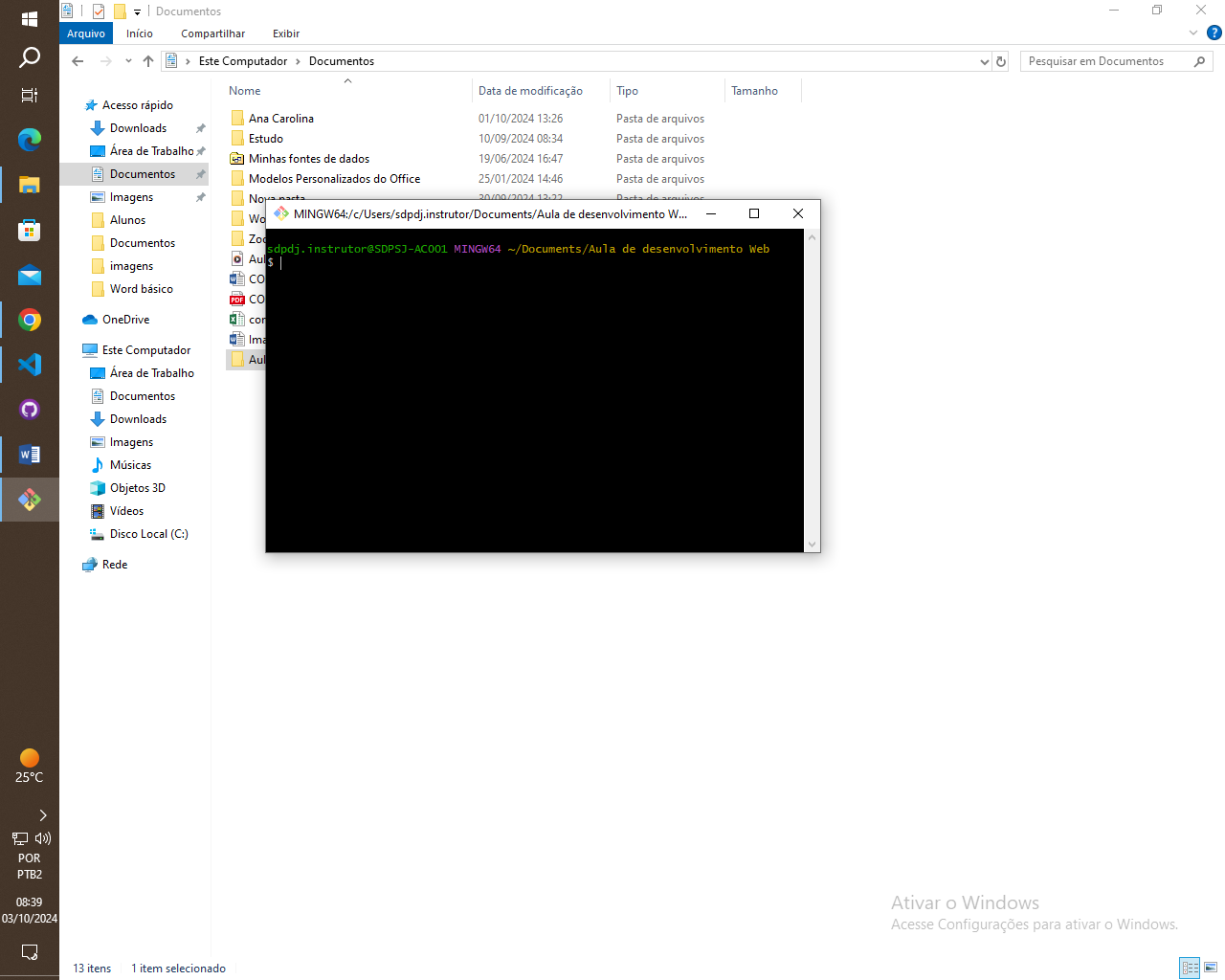


Com a pasta criada iniciaremos o nosso repositório local.

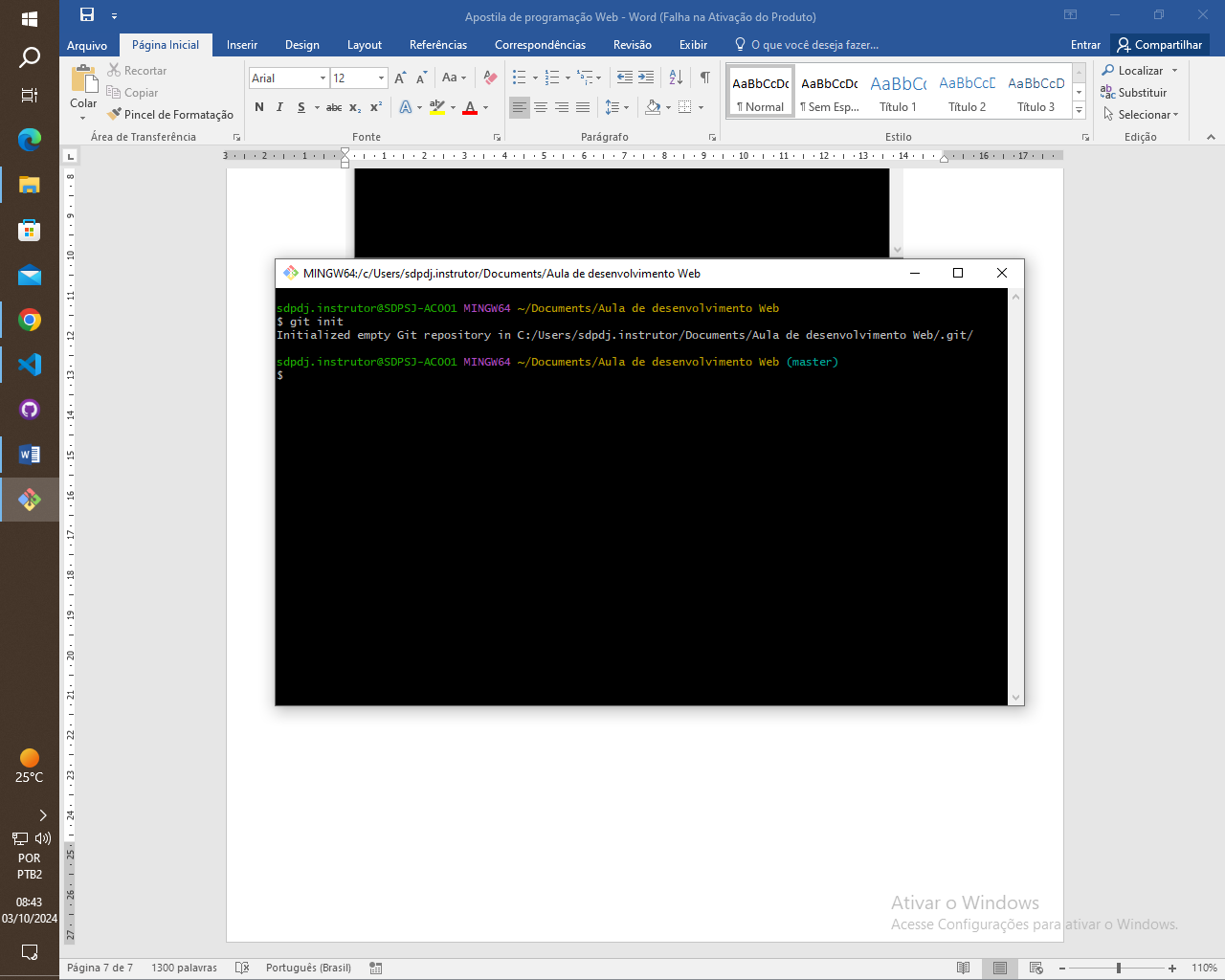
1. Clique com o botão direito do mouse sobre a pasta.



1. Escolha opção: Open Git Bash here.



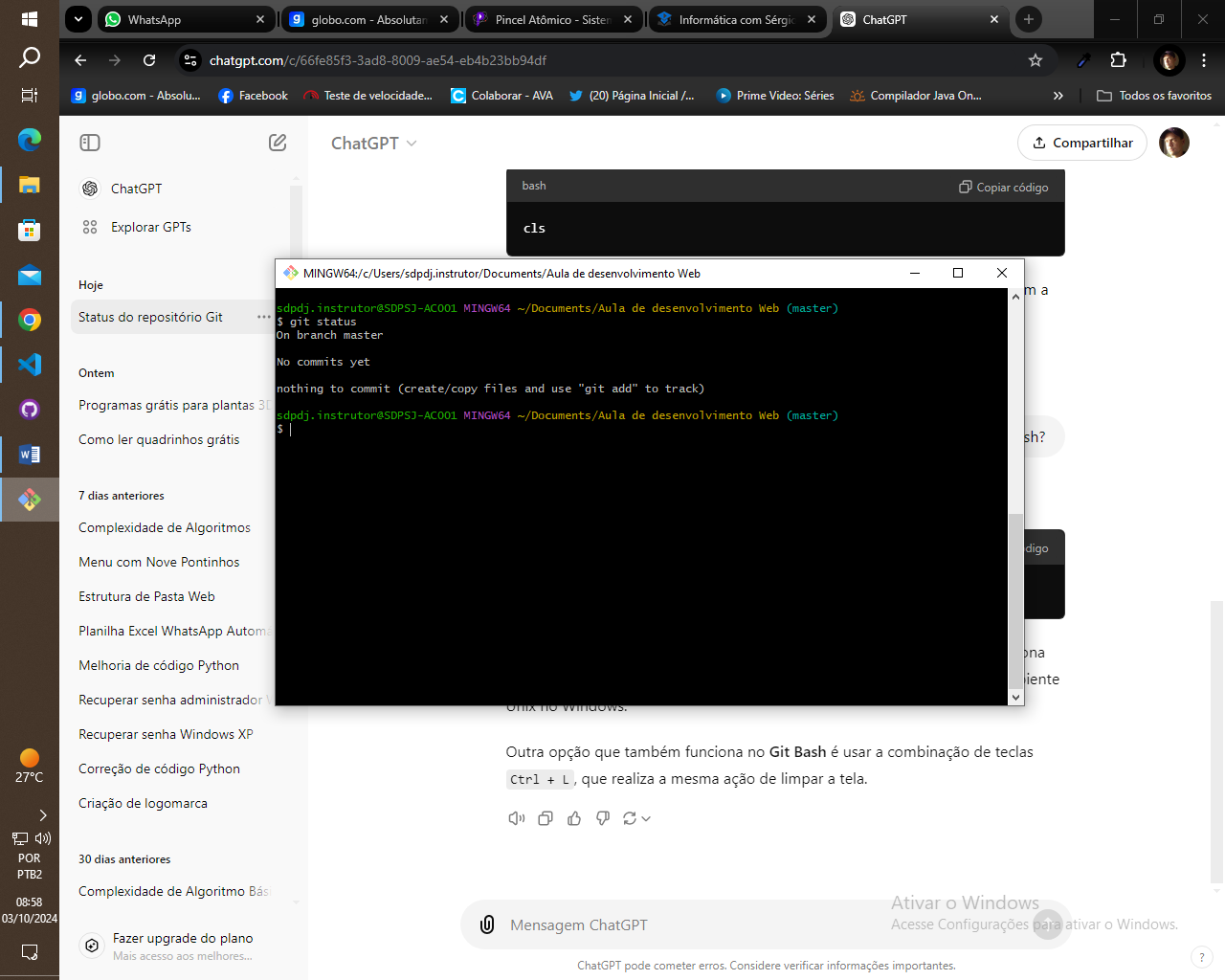
Pronto! Será aberto uma janela como essa.

Nesse momento o Git ainda não foi iniciado, para isso basta digitar o comando: git init e pressionar Enter.

### Por que apareceu entre parênteses a palavra máster?

O Git trabalha com ramificações, ou seja, ele cria partes de versões do trabalho e todas elas podem ser finalizadas dentro da “máster”.

Imagina que você e um amigo estão trabalhando ao mesmo tempo em um projeto. Ele cria uma atualização e você uma outra atualização, seria interessante que cada um estivesse guardando em um local diferente o seu trabalho. Depois é só comparar os dois e decidir se irão usar as versões. Caso a resposta seja positiva, essas versões se juntarão a ramificação principal e passarão a fazer parte do arquivo principal.

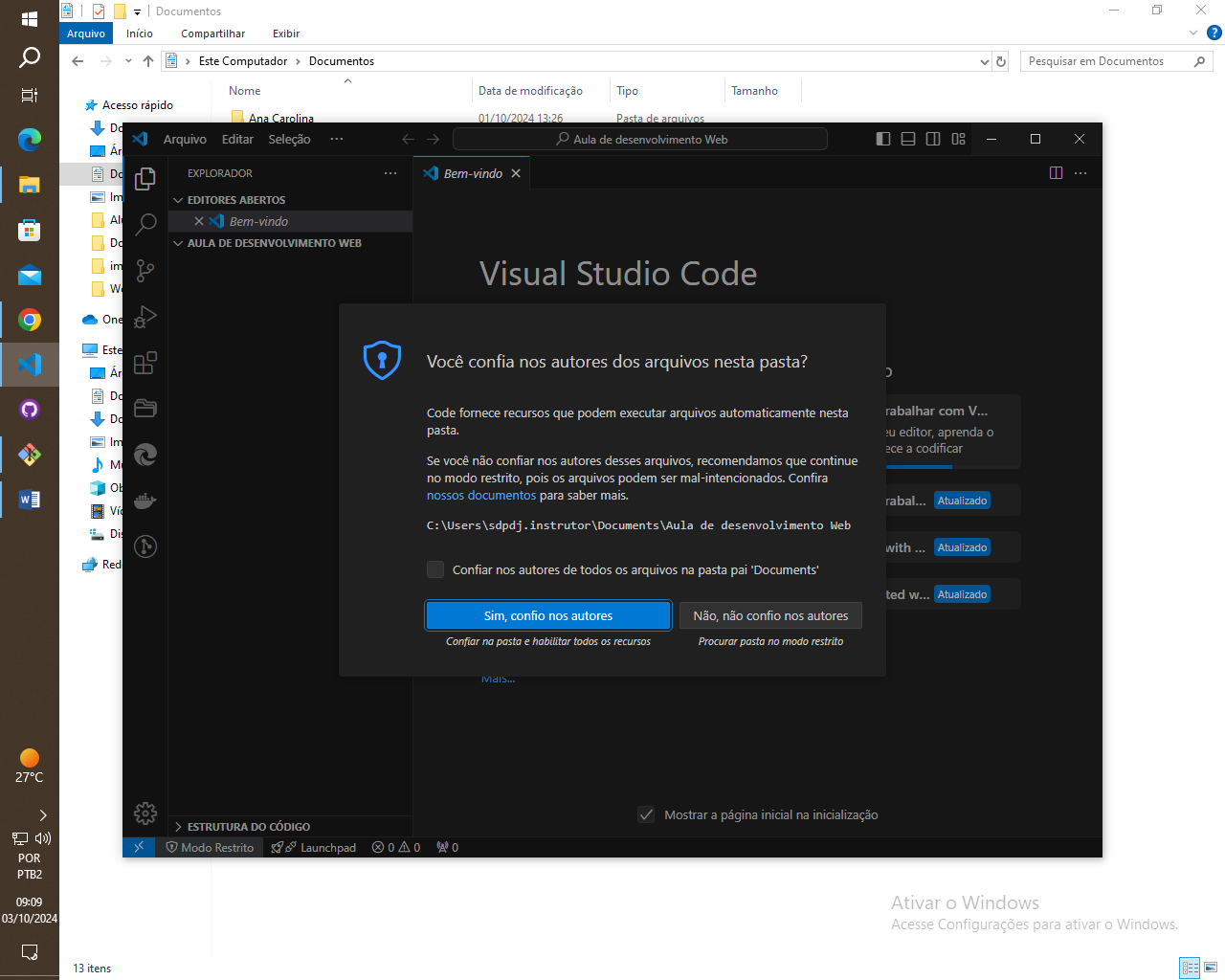
Para saber se está tudo certo com sua pasta, basta digitar o comando git status.

Vamos entender essas informações:

1. É mostrado uma mensagem mostrando que estamos dentro da branch master, ou seja, a ramificação principal. É importante destacar que alguns sistemas estão adotando chamar essa ramificação de main.
2. Os restantes das informações irão indicar que você não tem nenhum arquivo dentro dessa pasta é que não foram criadas modificações no seu trabalho. Entenderemos com detalhes mais tarde.

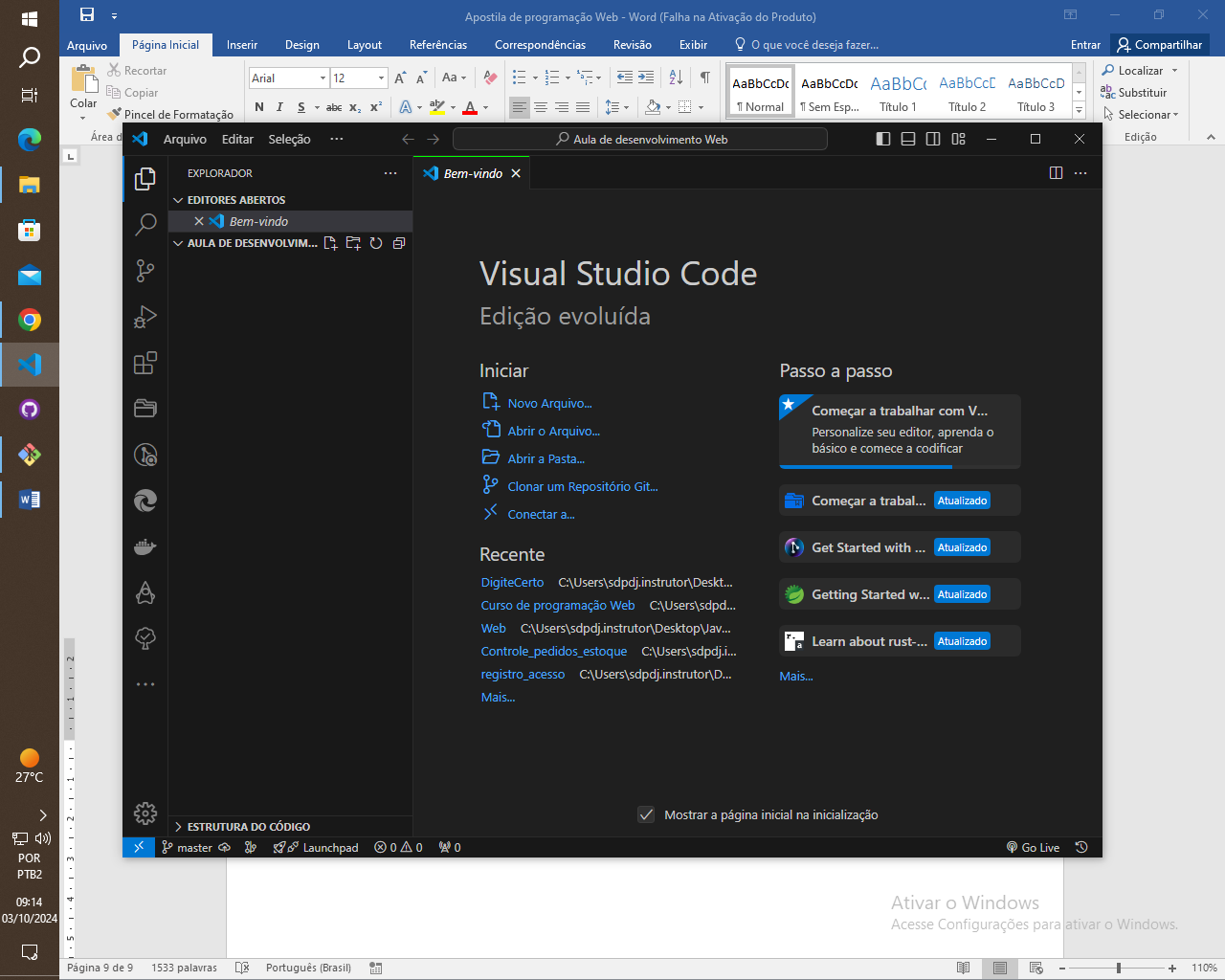
Vamos agora abrir essa pasta dentro do VSCode, para isso basta apenas clicar com o botão direito do mouse sobre ela e escolher a opção “Abrir com Code”.

Será perguntado se você confia nos autores da pasta, é só confirmar.



### Criando uma pasta para guardar imagens

Observe a faixa lateral esquerda da janela.

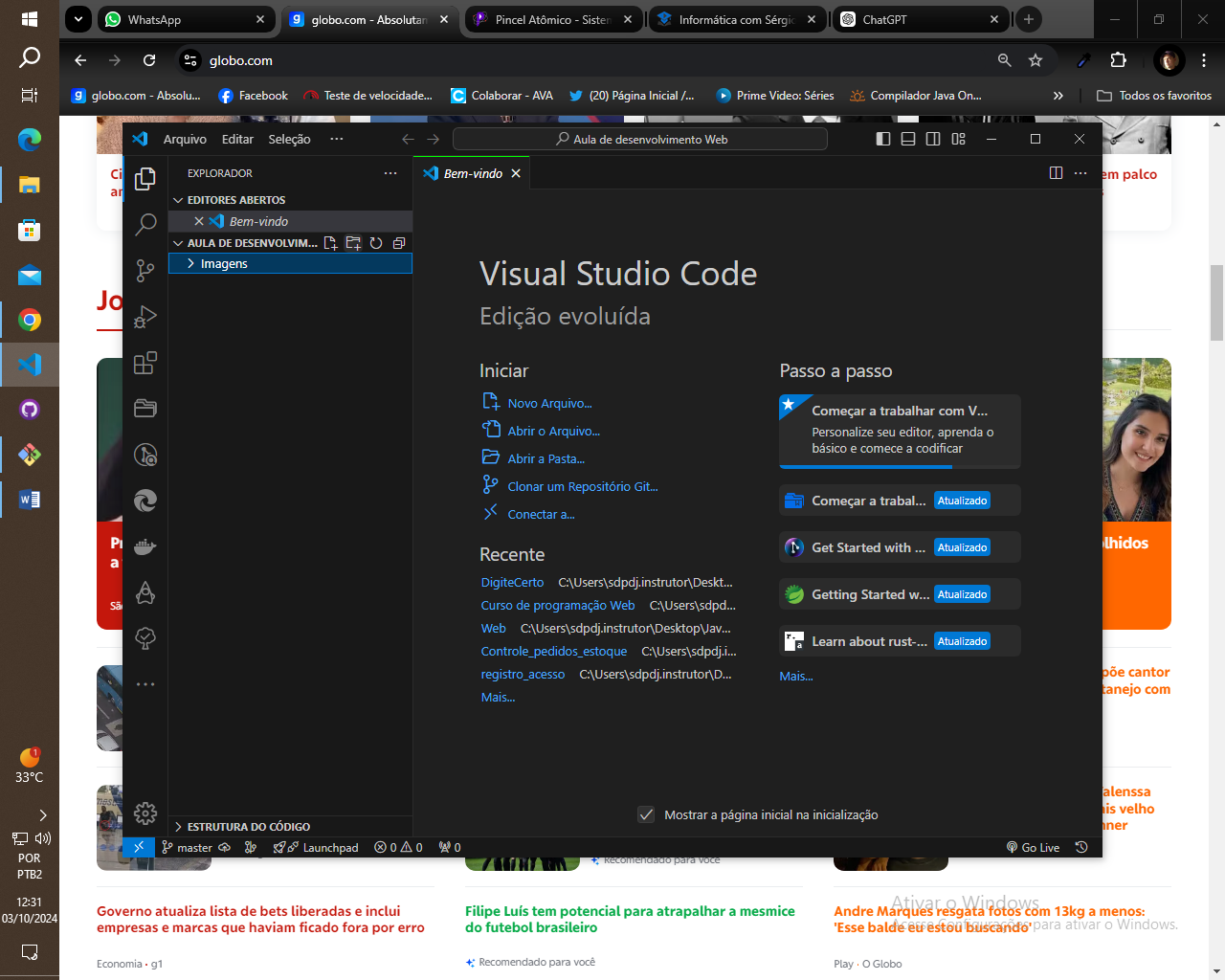
Muita atenção com as informações mostradas aqui.

É mostrado o nome da pasta principal e quatro ferramentas ao lado.

A primeira é para a criação de um novo arquivo, a segunda para a criação de uma nova pasta.

Usaremos essa segunda ferramenta para criar uma pasta chamada “imagens”, nela guardaremos todas as imagens que serão utilizadas durante as aulas desse módulo.

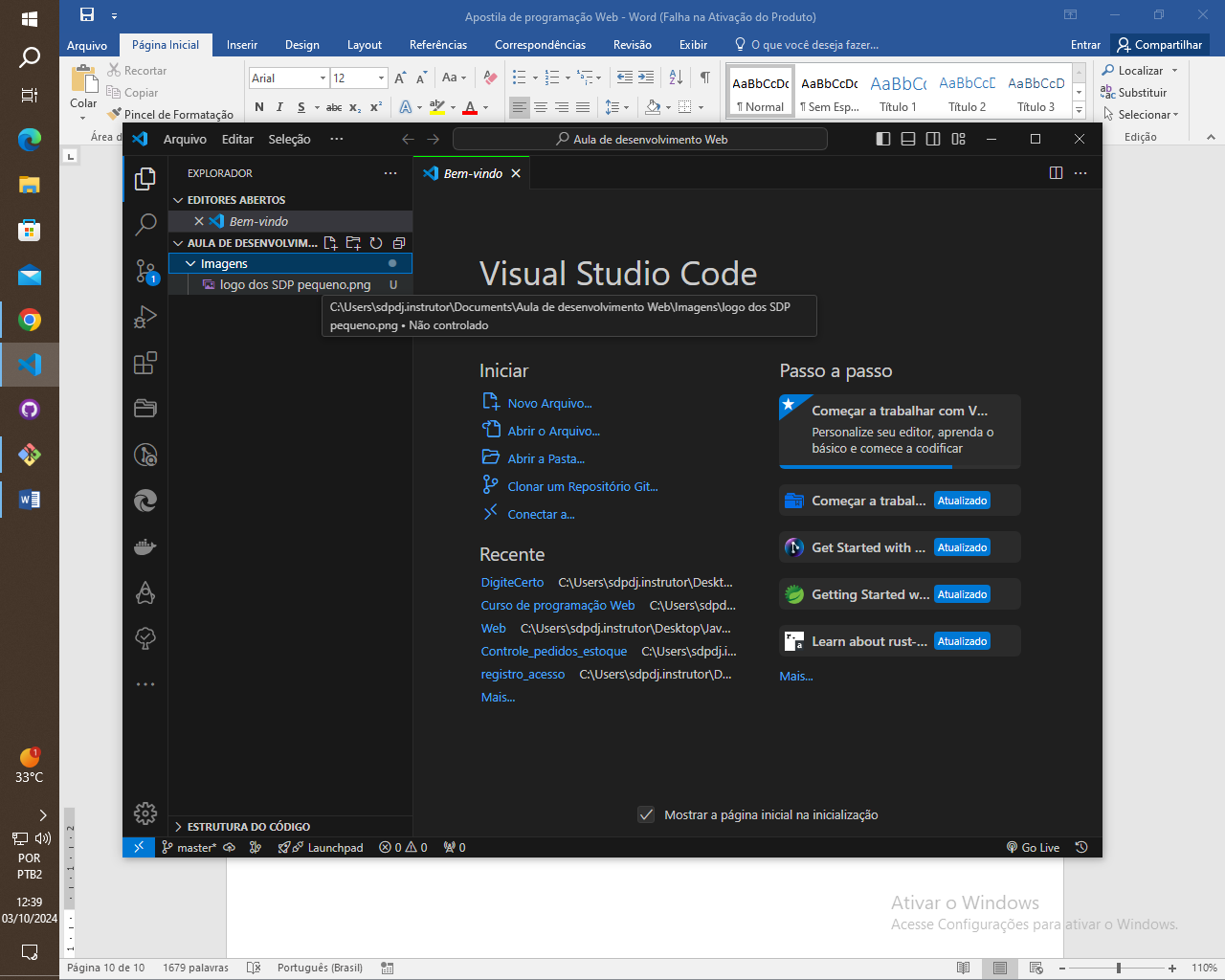
Mais à frente teremos uma aula mostrado o padrão de criação de pastas adotado pelos desenvolvedores, é importante conhece-los e adota-los, a isso dar-se o nome de “Boas Práticas”.

 Após a criação da pasta, o seu professor irá te ajuda a colocar a logo do Sistema Divina Providência dentro dela.

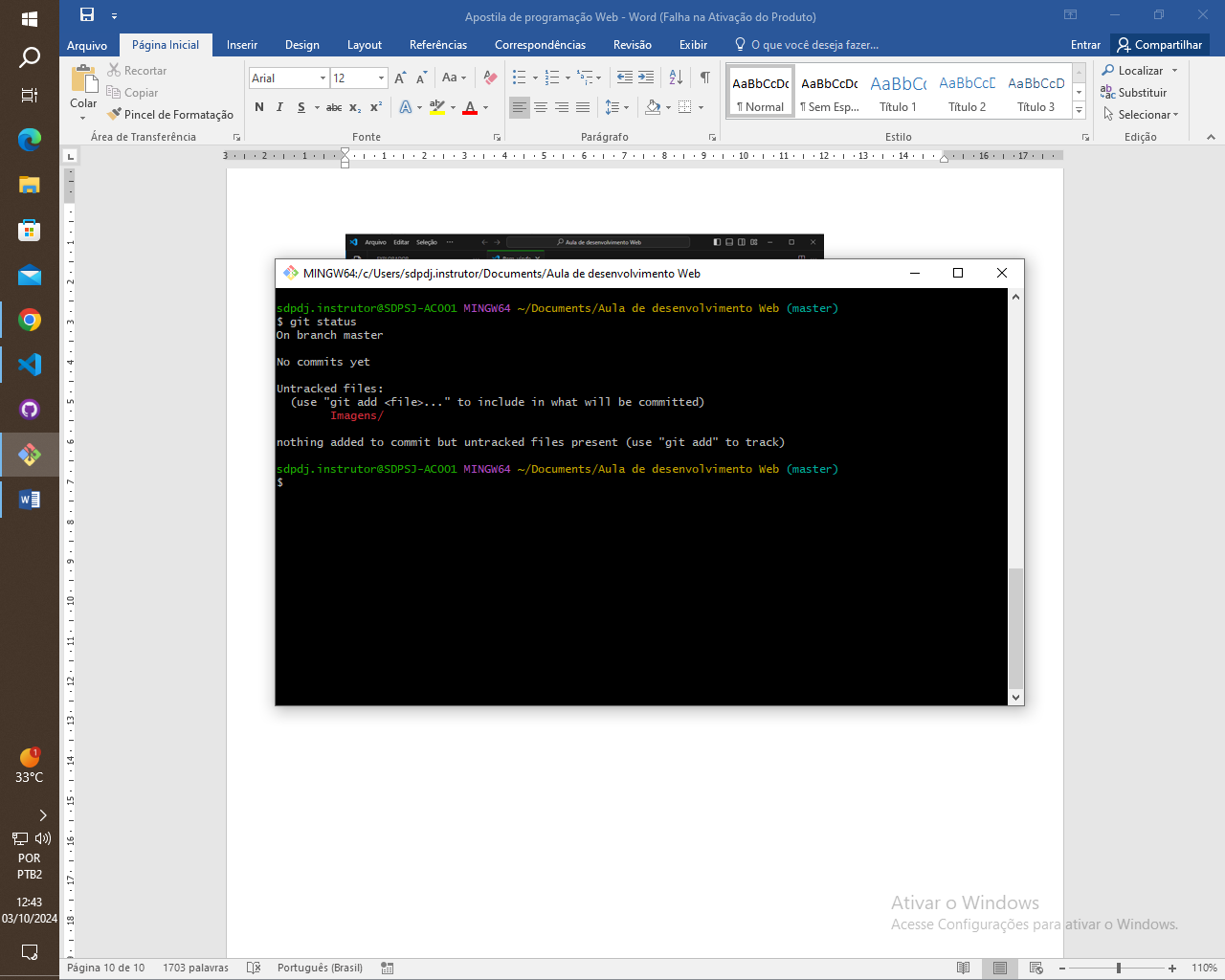
Para isso, talvez será preciso acessar o Google e pesquisar por logo do Sistema Divina Providência.

Você fará isso da mesma que que fez em seu curso de “Digitação e Internet”.

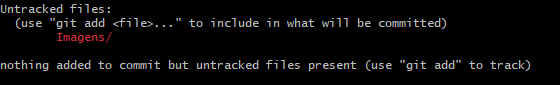
Depois confira o VSCode.



Agora nosso repositório está com um arquivo guardado e vamos visualizar isso no Git. Acesse o Git e digite novamente o comando git status.



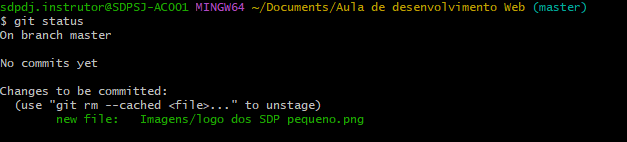
Note que agora, além de termos a indicação de uma branch master, temos também a informação de que há um novo arquivo dentro da pasta.



O arquivo está dentro da pasta, mas não está dentro do Git, para resolvermos isso vamos usar o comando git add seguido do nome da pasta que contém o arquivo.



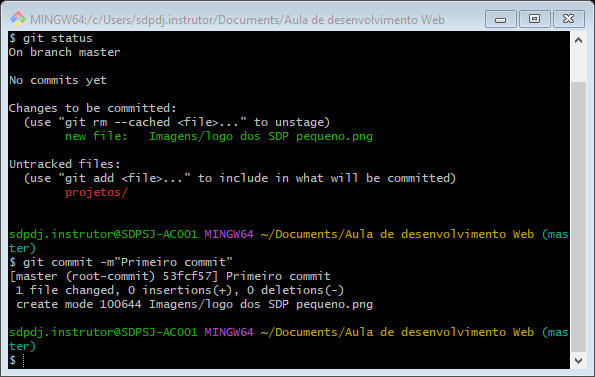
Agora, para saber se tudo está correto, basta digitar o comando git status.



Vimos que está tudo certo, porém é solicitado que façamos um commit, mas o que é isso???

Commit é o mesmo que criar um comentário para dizer o que foi feito de mudança, dessa forma fica mais fácil de se lembrar ou de se entender o que é pretendido naquele momento.

Para commitar vamos usar o seguinte comando git commit –m “comentários aqui dentro”.



Nesse momento temos as informações referentes ao nosso diretório.

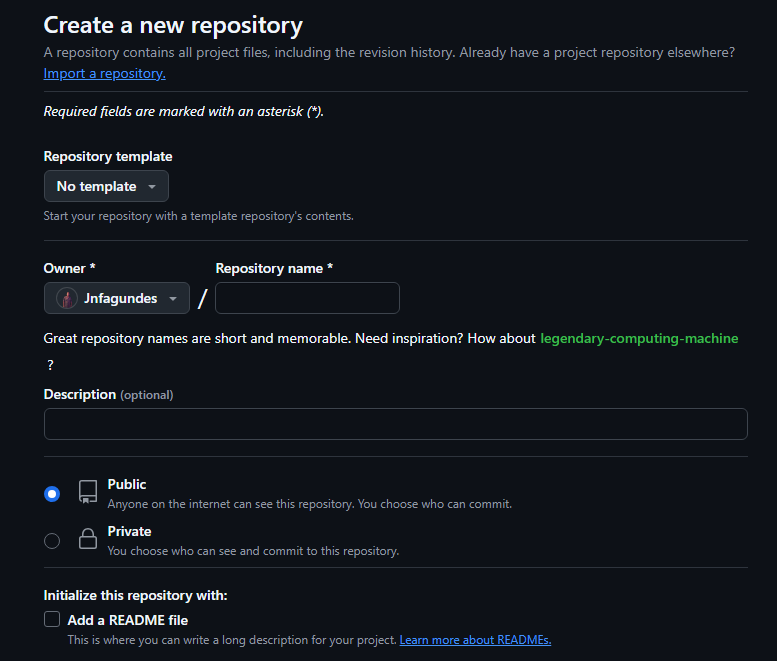
Simples para muitas pessoas e um pouco complicado para algumas, mas não fique preocupado. Além do Git, que é um versionador usado em um repositório local, temos o Github que é repositório remoto.

Para criar uma conta no Github basta acessar: <https://github.com> e fazer o cadastro.

Peça ajuda ao professor em sala acaso não consiga se cadastrar.

Vamos criar um novo repositório.

Clique em New e logo após vamos preencher os campos necessários.

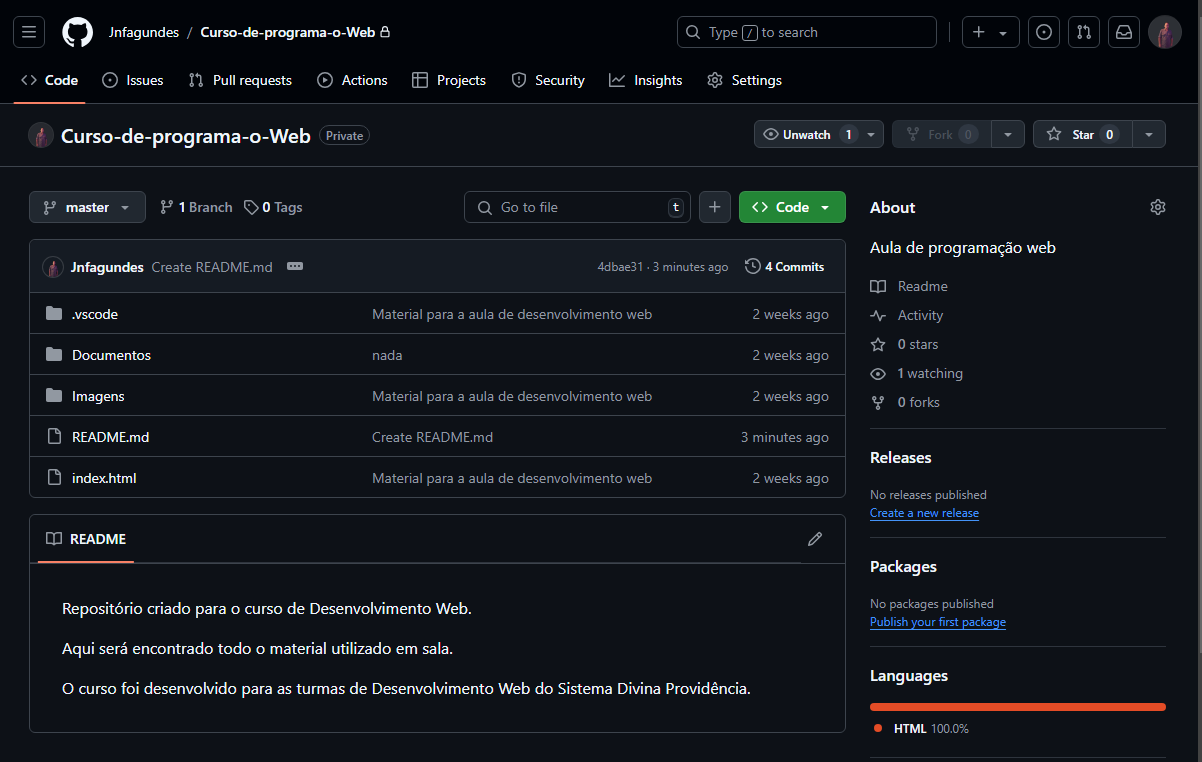


O seu repositório precisa de um nome que faça sentido para os trabalhos, uma descrição coerente do que será armazenado e trabalhado nele. Defina se o repositório será público ou privado e inicialize com um README.

O README é um texto contendo informações gerais sobre o trabalho que está sendo desenvolvido, coisas como quem está trabalhando nele, quem são os responsáveis pelo projeto e qual o objetivo do projeto.

É importante ter coerência na hora de determinar os nomes dos repositórios e na criação do README. Lembre-se que tudo isso é uma forma de documentação e servirá de parâmetro para mudanças futuras.

Seja sempre o mais profissional possível, mesmo que isso seja apenas um curso. Seriedade sempre.



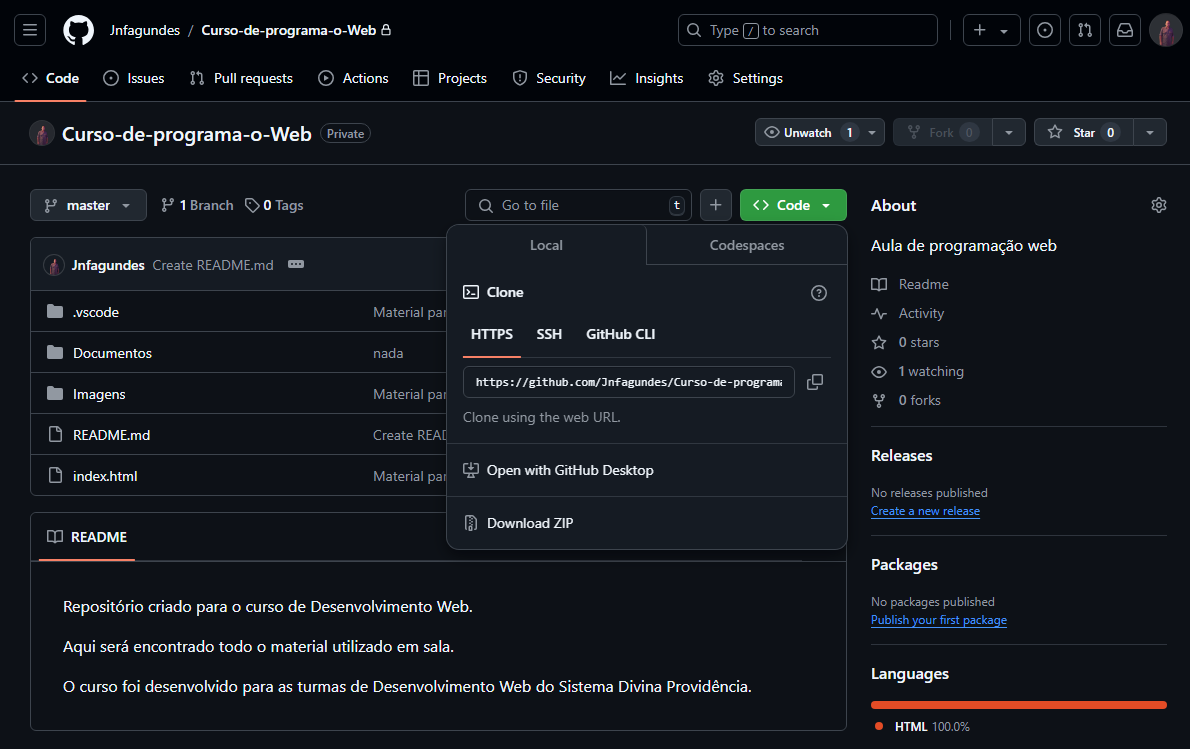
Aqui temos um repositório aberto no Github.

## Clonando um repositório remoto em um repositório local

Para trabalharmos com o projeto, devemos baixa-lo para nossa máquina. Isso será feito de uma forma muito simples, mas para isso baixe o GitHub desktop.

Baixe a ferramenta com a ajude de seu professor de sala, ou use um tutorial da internet.

Com o GitHub baixado e instalado em sua máquina, basta apenas clonar o repositório remoto.

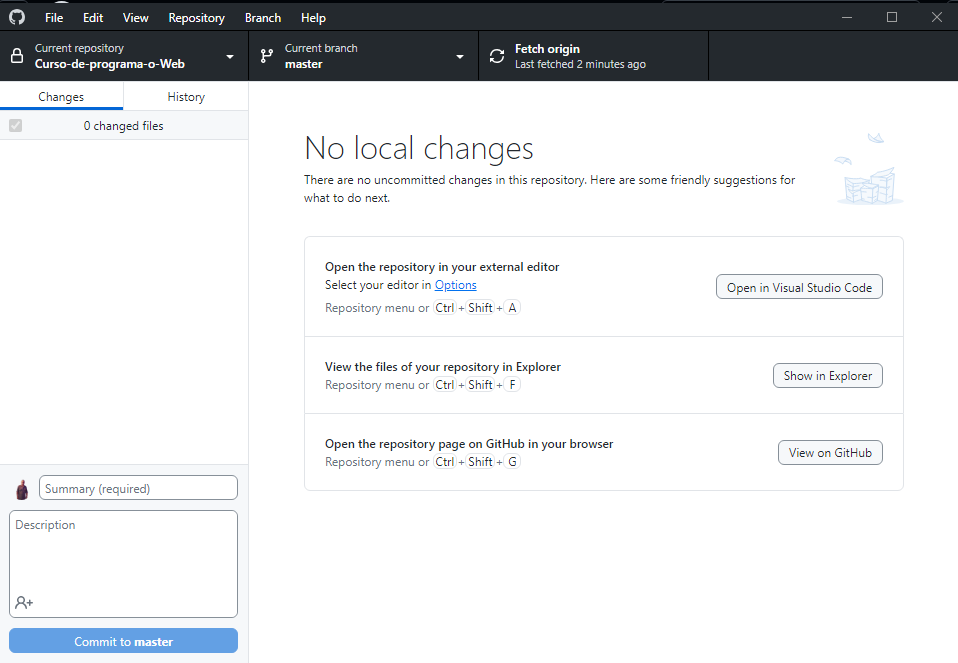


Clique em <>code e logo após clique em “Open with Github desktop”

Ao clonar escolha um local para guardar o arquivo clonado.

Na dúvida, procure o professor.

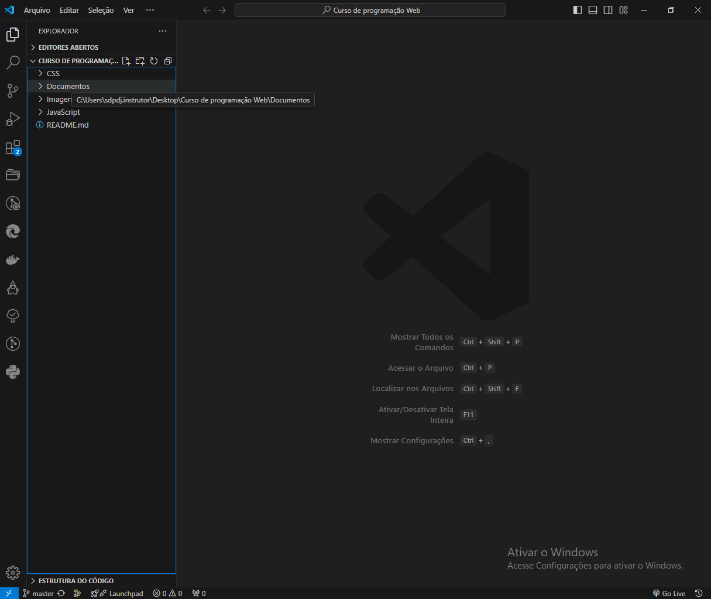
Uma vez clonado teremos a tela do programa:



Perceba que podemos abrir o VSCode diretamente dele e podemos também explorar a pasta que contém nossos arquivos.

Clique em “Open in Visual Studio Code”.

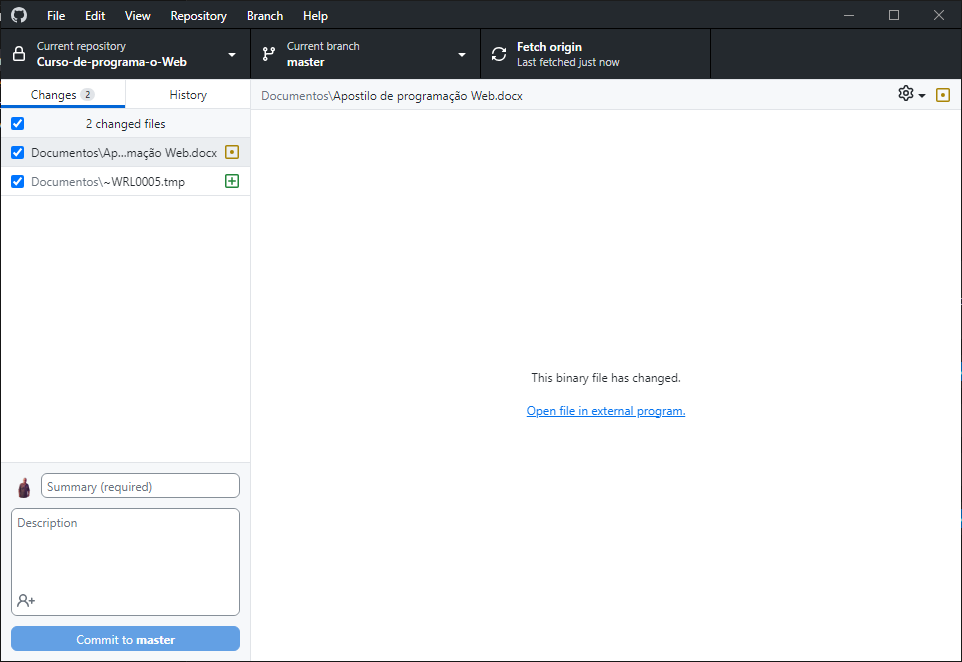
Crie as seguintes pastas conforme a imagem.

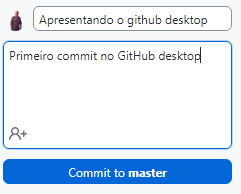
As pastas são:

* CSS
* Imagens
* Documentos
* JavaScript

Logo após cria-las, vá até o github desktop e perceba que ele nos mostra que houve alteração.

Provavelmente a sua imagem estará diferente da minha, mas o princípio é o mesmo.



Na imagem acima foi sublinhado de vermelho a parte responsável pelo commit, já fizemos isso antes, porém aqui será muito mais fácil.

Aqui colocaremos um pequeno título e na caixa de baixo colocaremos uma pequena descrição daquilo que foi feito.

Logo após clicaremos na opção de Commit to master, ou seja, envie para o master, e por fim clique em Push origin para enviar as mudanças para o repositório remoto.

A princípio isso é tudo que devemos saber, durante o curso surgirá novas dicas.

Importante: Usaremos bastante os repositórios locais e o remoto.

# Vamos agora começar a estudar o HTML

Lembra de quando falamos de HTML?

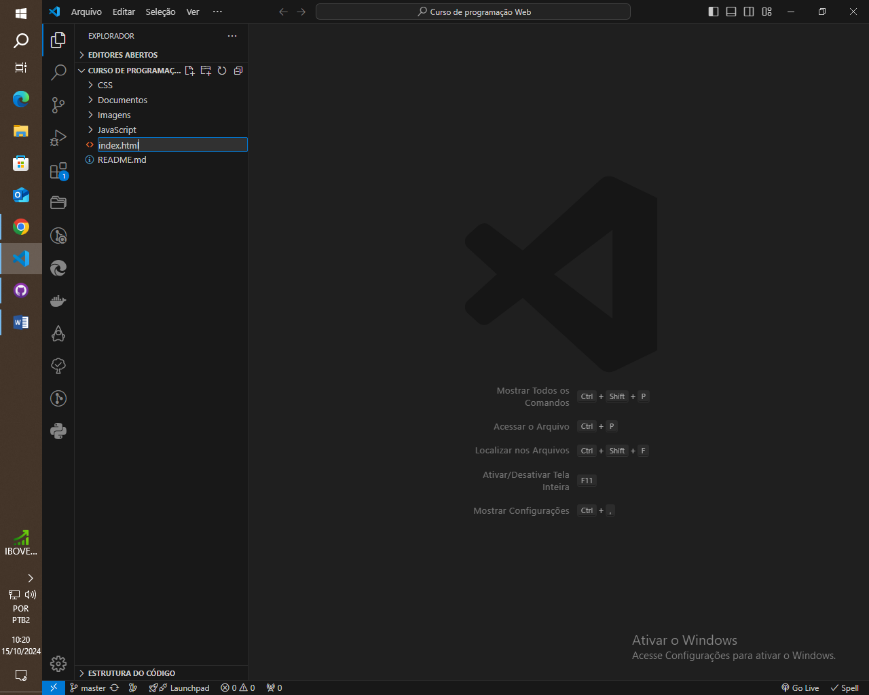
Definimos ele como uma linguagem de marcação, portanto é fundamental entender que ele não é usado como linguagem de programação.

O HTML serve para posicionar o texto da sua página, definindo a função de cada um dos elementos desse texto.

De uma forma mais clara: ele define quem é o título principal e os subtítulos, define quem é o cabeçalho da e o rodapé, assim por diante.

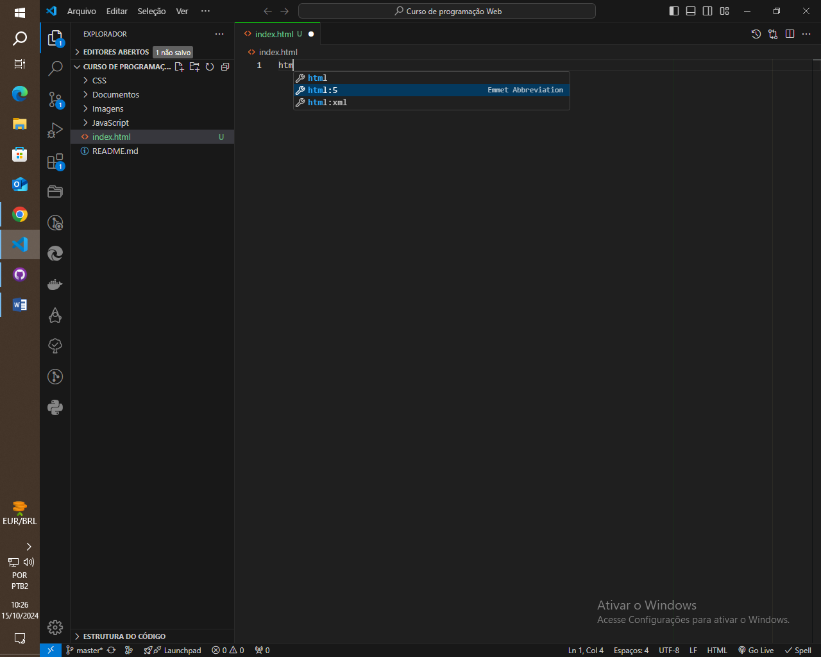
Vamos ver a estrutura básica do HTML, mas para isso vamos até o VSCode criar um arquivo com o nome de index.html.

Na sua aula de informática básica foi ensinado extensão de arquivo, por isso não entrarei em detalhes, qualquer dúvida consulte seu professor.

Logo após a criação do arquivo index, o VSCode abrirá uma guia.

Observação importante:

* O primeiro botão cria um arquivo
* Não entre dentro de nenhuma pasta, caso contrário criará o arquivo dentro dela. Cuidado com isso.
* Todos os arquivos terão a sua extensão mostrada, quando isso não acontecer, estaremos vendo uma pasta e não um arquivo.

Digite html na barra da nova guia e escolha a opção “html: 5”.

Será criado uma estrutura básica, vamos analisar cada item dela.

1º <!DOCTYPE html>

2º <html lang="pt">

3º <head>

1. <meta charset="UTF-8">
2. <meta name="viewport" content="width=device-width, initial-scale=1.0">
3. <title>Document</title>

4º </head>

5º <body>

6º </body>

7º </html>

1º... Aqui temos uma conversa interna. O navegador está recebendo a informação de que será aberto um documento do tipo HTML 5, a versão mais recente do html.

2ª... Temos o início do arquivo HTML. Aqui está sendo mostrado que o nosso documento está no idioma pt. Podemos ser mais claros e colocar o Br para indicar que o idioma é o Português do Brasil.

3º... Essa parte guarda informações importantes para o navegador, o que chamamos de metadados.

1. Temos o perfil de caracteres utilizados pelo documento, ou seja, aqui mostra que nosso site usará caracteres do tipo “ç”, acentos etc.
2. Aqui definimos em qual tipo de tela vamos trabalha.
   1. Temos: name= “viewport”; mostra que estamos definindo as configurações da área visível na tela.
   2. Temos: content=”width=device-width; garante que a largura da viewport seja igual à largura da tela onde a viewport será exibida.
   3. Temos: initial-scale=1.0”; indica que a página será exibida sem zoom, apenas com seu tamanho natural.
3. Aqui teremos o título do documento que será exibido na guia do seu navegador.

4ª... Nessa linha temos o fechamento da tag <head>

## Mas o que são tags?

Tags são elementos que dizem ao navegador como interpretar cada item colocado na página, ou seja, temos tags para indicar que determinado item tem que ser entendido como título, outras para indicar que temos um parágrafo etc.

As tags mais comuns são dos seguintes tipos:

* Estruturais: usadas para definir a estrutura de um documento.
* Conteúdos: usada para mostrar qual é o tipo de conteúdo que está sendo colocado ali.
* Auto fechadas; usada sem a necessidade de terem fechamento, diferente da tag <head> que necessita fechamento usando </head>

A diferença da tag aberta para a tag fechada é a aparição da barra / logo após o sinal de menor <.

As tags podem ter atributos, como exemplo temos a tag <html>. Atributo é o mesmo que características aplicadas a essa tag, como vimos o atributo lang=”pt”. Para colocar um atributo em uma tag, usamos a tag de abertura, <html lang=”pt”>.

Na 5ª e 6ª temos a abertura e o fechamento da tag body. Nessa tag colocamos todo o conteúdo que será visto pelos usuários do site, ou seja, é aqui que temos contato com o conteúdo enquanto estamos navegando.

E por fim fechamos o documento usando a tag de fechamento </html>

Para facilitar nossos estudos, usaremos comentários em algumas linhas importantes, para isso basta usar <!--Comentar aqui -->.

<!DOCTYPE html> <!--Indica que o documento será um html5-->

<html lang="pt"> <!--Abertura do html e sua definição de linguagem-->

<head>

    <meta charset="UTF-8">

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

    <title>Document</title>

</head>

<body>

</body>

</html> <!--Encerramento do html-->

Crie o hábito de comentar seus códigos, dessa forma ficará mais fácil o entendimento e evitará possíveis dúvidas futuras.

# Vamos começar a desenvolver a nossa primeira página?

Esse será o nosso modelo:



Ele é bem simples, mas atende todas as nossas necessidades. Teremos aqui a utilização de alguns recursos poderosos e gostaria que você, aluno, tente aplicar tudo que for ensinado aqui em algum projeto pessoal.

Tente montar uma página para um comercio local como padaria, salão etc. Nessa área, só se aprende exercitando, e isso te ajudará a criar um portfólio.

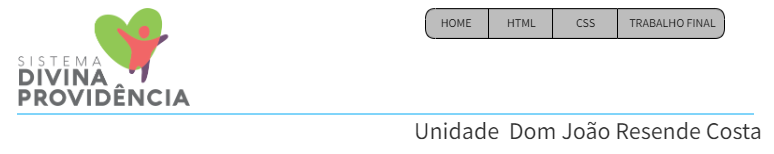
Para a criação desse modelo, foi utilizado o site <https://wireframepro.mockflow.com/> , recomendo que pesquise na internet como usá-lo ou arrisque fazer algo.

Podem ser utilizados editores de imagem, qualquer um. O mais importe é entender que todas as vezes que desenvolvemos um site, devemos ter um planejamento, ou seja, devemos montar um modelo antes, dessa forma podemos alterar o que não ficou legal.

Vamos começar pelas definições básicas do site.

Primeiro devemos definir um título que será apresentado na guia da página.

<title> Estudando desenvolvimento Web </title>



O topo pode ser entendido como cabeça do site e dentro do html temos uma parte reservada para isso <header></header>.

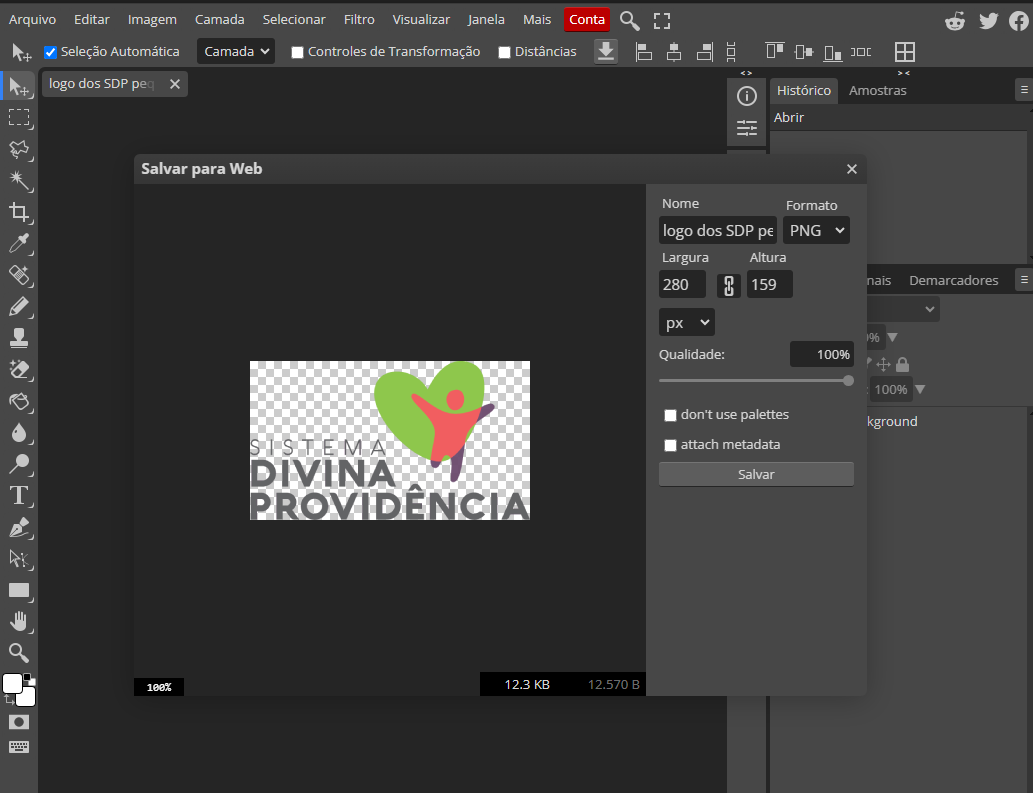
Essa característica do html dá a ele o conceito de semântico, ou seja, cada coisa no html vai ser entendo pelo navegador de acordo com o sentido desse algo.

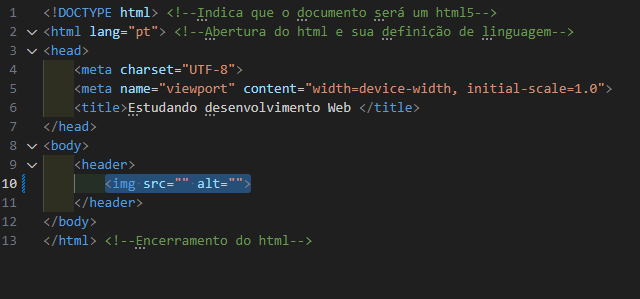
Isso ajuda os sites de busca a entenderem cada trecho de sua página.

Todas as vezes em que pensarmos na parte de cima do site, vamos pensar em header.

No VSCode, basta digitarmos header e apertar enter, automaticamente é criado a tag de abertura e a tag de fechamento, entre elas colocaremos os elementos referentes.

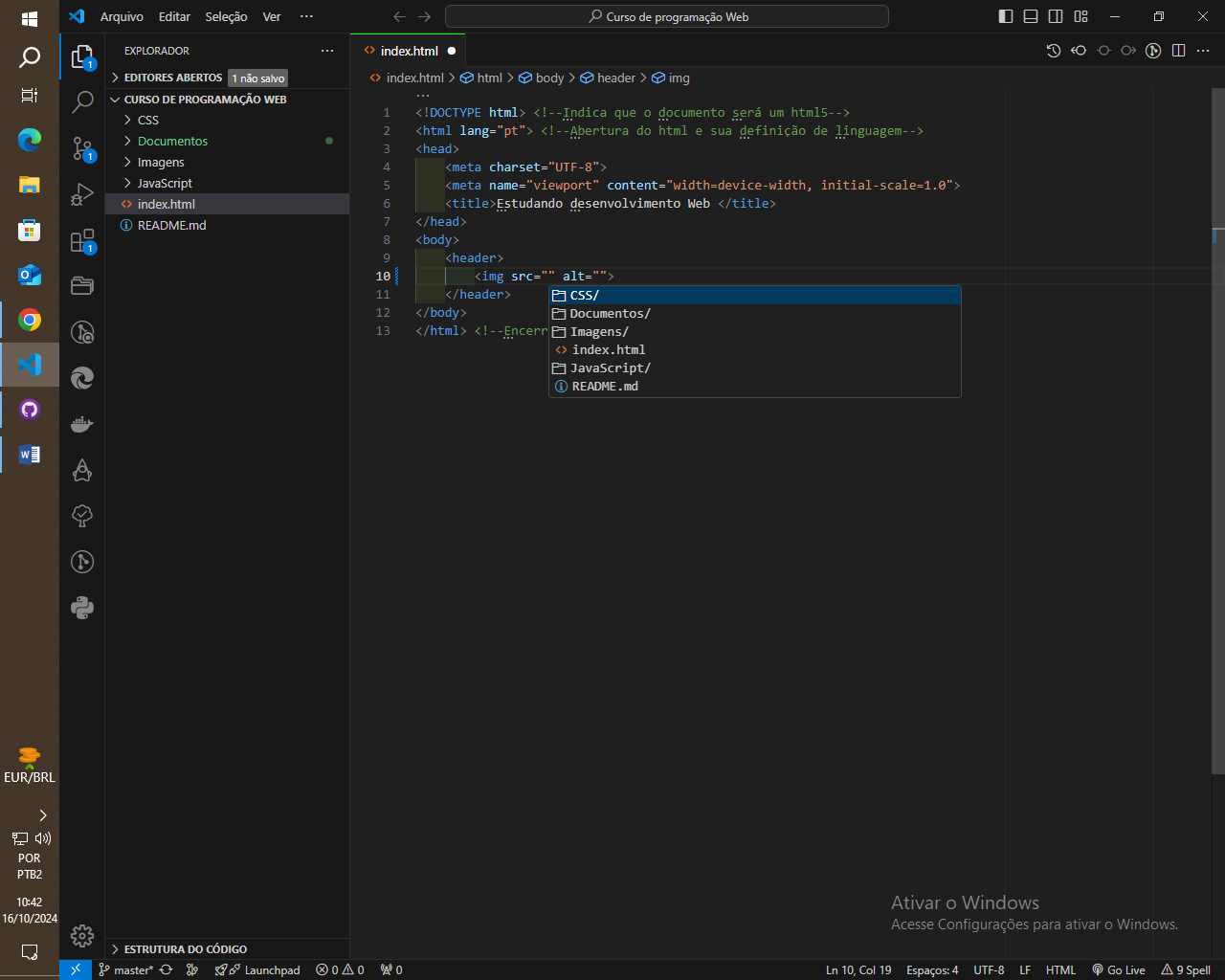
O primeiro elemento que iremos inserir é a logo do Sistema Divina Providência, para baixa basta pesquisa-la no Google. Para edita-la no tamanho correto, basta abrir o site <https://www.photopea.com/>. Esse site é uma versão gratuita do Photoshop, bem intuitivo. Abra a imagem e remova o plano de fundo, depois exporte em formato png com as seguintes dimensões altura: 159 e largura: 280.

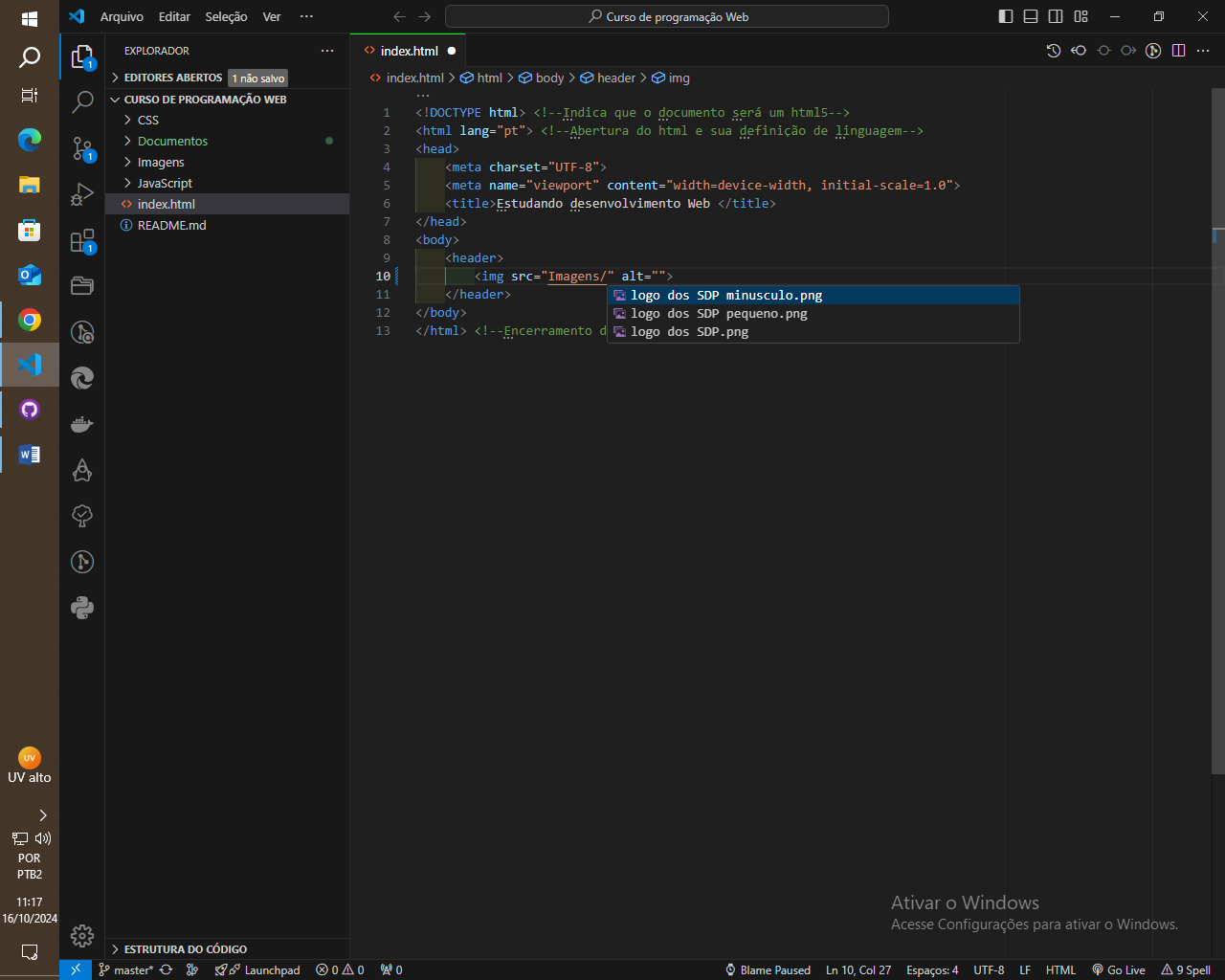


Agora que a imagem já se encontra na pasta imagens, vamos coloca-la dentro da nossa página. Vamos digitar img e pressionar a tecla enter.

Mais uma vez temos a ajuda do VSCode.

Essa tag não tem fechamento, mas ela traz consigo dois itens bem interessantes. O primeiro é src=””, usado para colocarmos o endereço da imagem. Para isso posicione o cursor entre as aspas duplas e pressione as teclas CTRL+ SPACE.

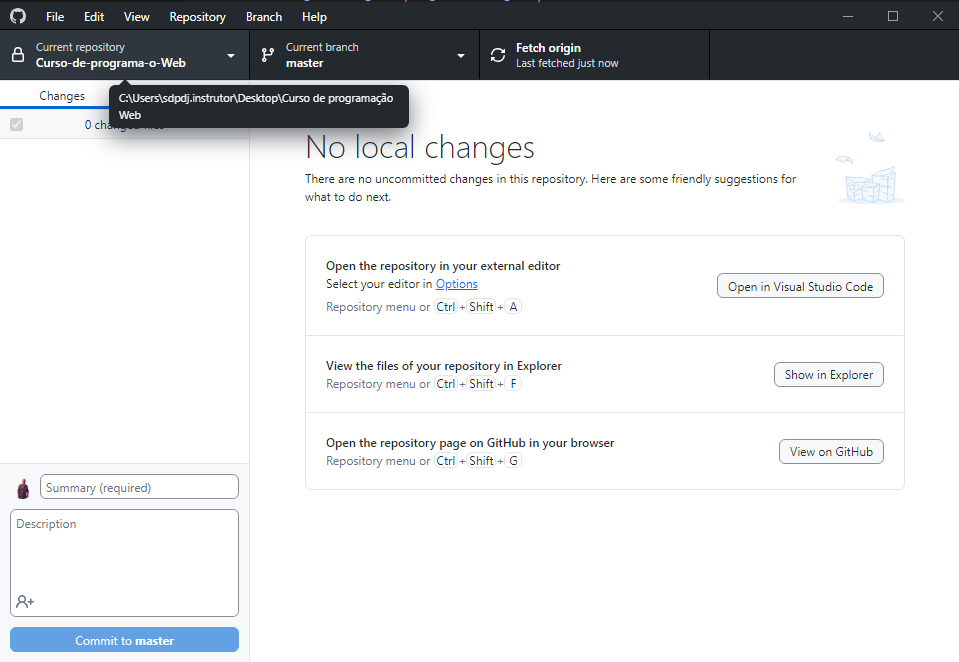
Usando esse atalho, temos uma pequena lista de pastas e arquivos pertencentes ao nosso projeto.

Vá até a pasta imagens e escolha a imagem da logo que acabamos de adicionar.

A outra opção dentro da tag img é a alt=””, entre as aspas colocaremos um texto que descreverá a nossa imagem caso ela não seja carregada na página.

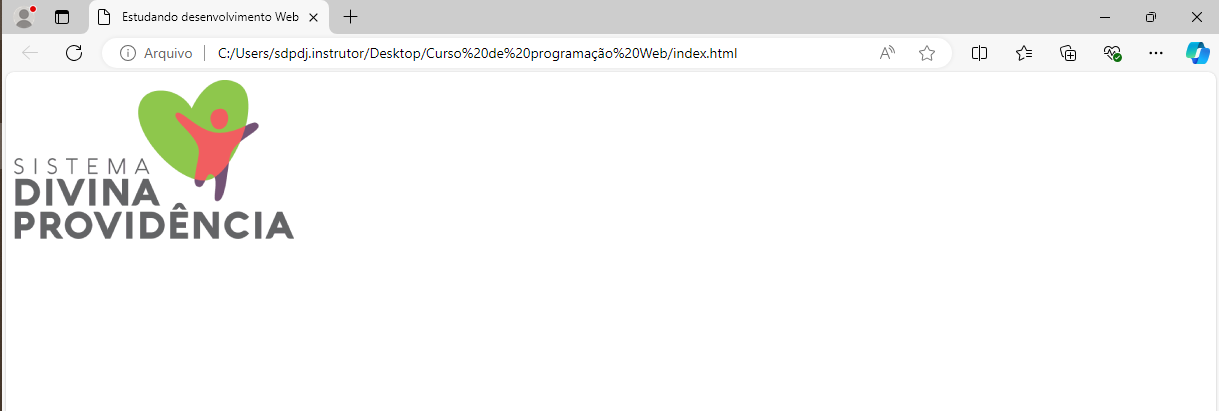
Vamos colocar “Logo do Sistema Divina Providência”.

Use o atalho CTRL+S para salvar as alterações do projeto, logo após acesse o github desktop, faça um commit “Colocando uma imagem no site”, use a seguinte descrição: “Primeira imagem inserida”. E clique em commit to master.

Por último clique em Push para mandar essa alteração para o repositório remoto.

Agora clique em Show in Explorer e abra o seu arquivo index no seu navegador de preferência.

Acredito que o resultado esteja igual a esse:



Na parte reservada ao nome da guia está o título que demos ao nosso projeto “Estudando desenvolvimento Web”, e na parte do endereço do site está o local do nosso arquivo, no nosso caso, está sendo mostrado o local em que temos o nosso repositório local.

Lembre-se que estamos trabalhando no header, parte do HTML responsável pelo topo do site. Isso se chama semântica, cada elemento tem um motivo para existir.

Temos que montar agora as partes do link.

Link é uma palavra, frase ou imagem que ao ser clicada nos leva para outra parte da página ou para outra página. Nesse site temos quatro links que nos levam para quatro páginas dentro do site.

Para criar a área do link, usamos a tag <nav></nav> e dentro dela colocamos a tag <a>, essa tag é usada para ancorar o link a página que queremos acessar, já a tag nav especifica que teremos uma área para colocar as ancoras.

Monte conforme mostrado, observe o “#” sendo usado depois do termo “href”.

Quando a página já está criada, colocamos o local da página no lugar do #. Esse símbolo é usado para que ao clicar não paremos em uma página vazia, sem conteúdo.

<header>

        <img src="Imagens/logo dos SDP pequeno.png" alt="Logo do Sistema Divina Providência">

        <nav>

            <a href="#">HOME</a>

            <a href="#">HTML</a>

            <a href="#">CSS</a>

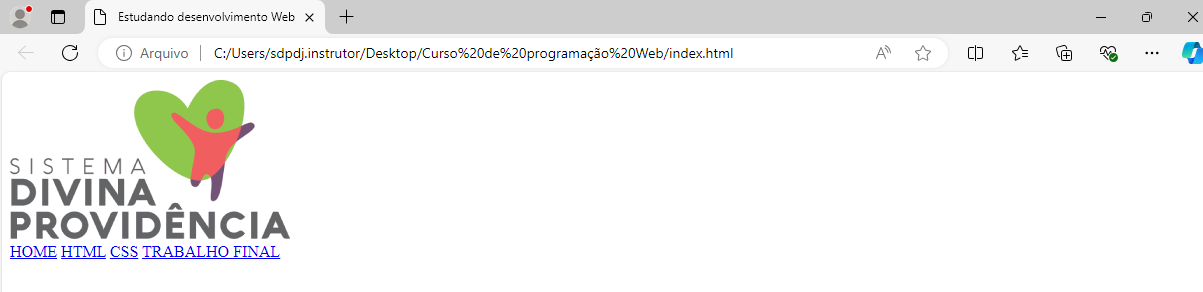
            <a href="#">TRABALHO FINAL</a>

        </nav>

    </header>

Salve a alteração apertando as teclas CRTL+S, vá até o navegador e o atualize apertando a tecla F5.

O resultado será esse:



Nada parecido com o modelo que desenvolvemos! Mas calma, ainda temos muito o que fazer!

# O que é CSS?

Vamos agora abordar um conceito bem interessante no desenvolvimento Web que são as folhas de estilo, o famoso CSS.

Enquanto o HTML é usado para marcar as posições do conteúdo de uma página, o CSS é responsável por dar uma aparência aos elementos da página.

É ele quem diz qual a cor que cada texto tem, onde que devo posicionar cada imagem etc.

## Mas como aplicar CSS na página?

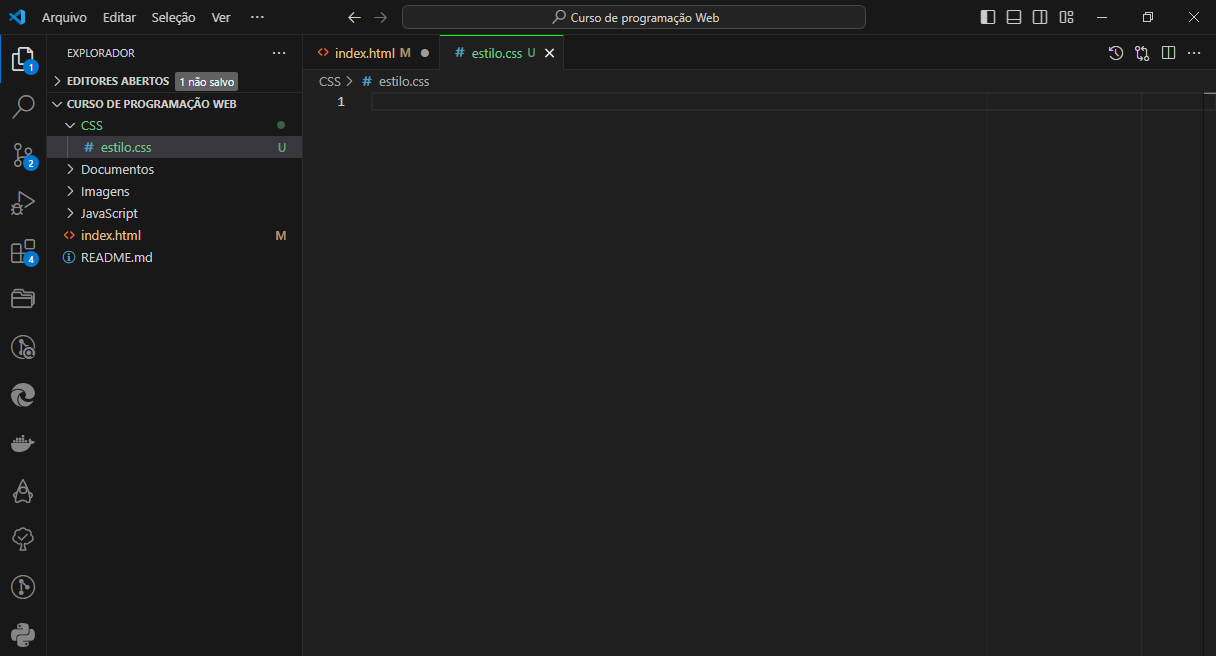
Temos três formas de aplicar CSS nas páginas, vamos a elas:

* CSS interno: Nessa situação o CSS é aplicado diretamente no HTML entre as tags <style> </style>, dentro do head.
* CSS inline: Aqui podemos adicionar o CSS dentro da tag usando o atributo style.
* CSS externo: podemos montar um arquivo separado e organizar o nosso CSS dentro dele (recomendado). Para funcionar usamos a tag <link> dentro da tag head. Ficará assim:
* <link rel="stylesheet" href="style.css">

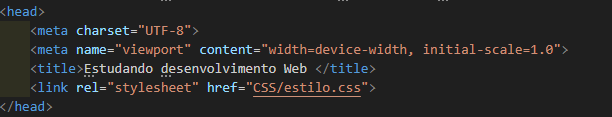
Vamos entender cada parte dessa tag:

* Link; tag usada para ligar elementos externos à nossa página.
* Rel; define o tipo de relação entre o documento HTML e o recurso vinculado.
* Stylesheet; indica que o arquivo externo é uma folha de estilo CSS.
* Href; indica o caminho do arquivo. Quando ele está na mesma pasta do CSS, basta colocar o nome do arquivo.

Então vamos criar o nosso arquivo CSS. Para isso vamos clicar na pasta CSS que criamos lá no VSCode. Logo após clique na opção para criar o arquivo e dê o nome “estilo.css”.



Agora precisamos ligar o CSS a nossa página HTML.

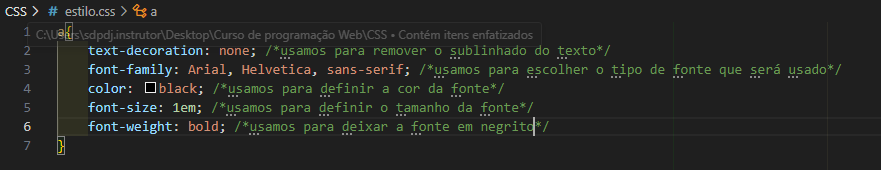


Observe que o estilo.css está dentro da pasta css. Foi usado uma barra para separar pasta de arquivo.

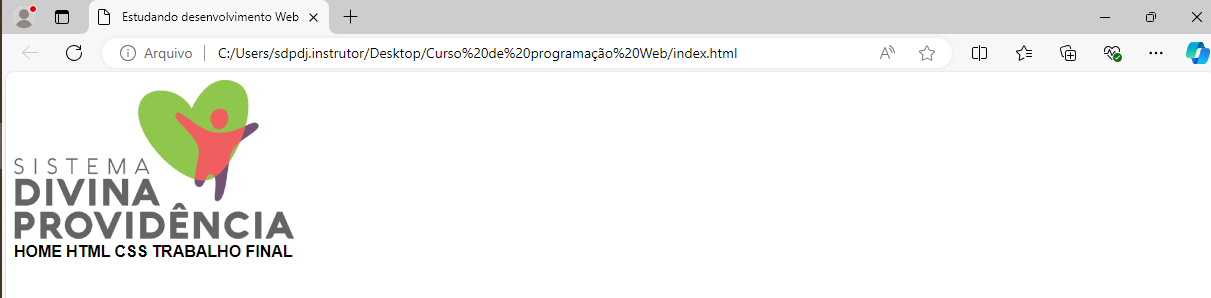
## Vamos começar a criar nossos estilos.

Vamos mostrar para o CSS que queremos mudar a aparência dos elementos <a>. Para isso, basta apenas digitar a{ e apertar enter.

Entre as chaves, vamos colocar tudo que queremos modificar.



Nesse trecho temos os estilos que serão aplicados via CSS externo. Observe que também foi criado um comentário para cada linha de estilo indicando o que cada código faz. O resultado será esse:



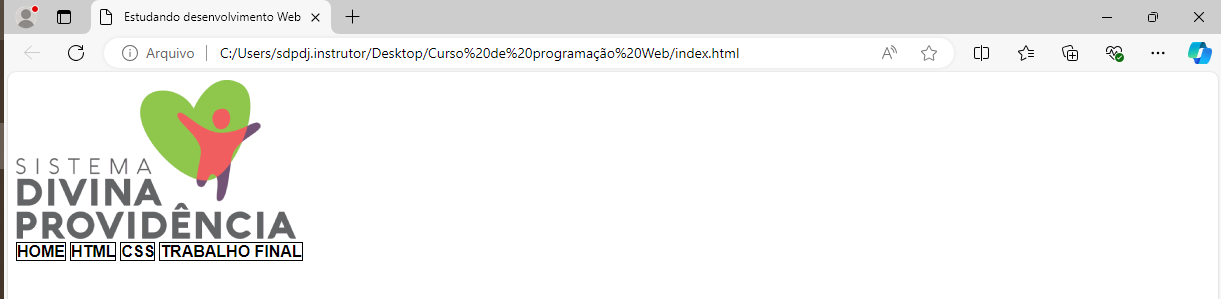
Observação: A nossa aplicação de estilo está agindo diretamente sobre o elemento <a>, não fizemos nada em relação ao elemento <nav>

Vamos aplicar bordas em torno do elemento <a>. Para isso inserir:

, o solid indica que teremos uma borda com um traço constante com a largura de 1 px e cor preta.

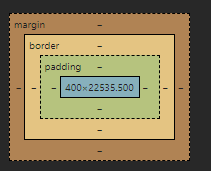
Não vamos nos preocupar com as escalas “em” ou “px”, falaremos adiante.

O resultado será esse:



Ainda não se encontra como queremos.

Vamos entender um item importante antes de continuar:

Essa imagem mostra como o nosso objeto é visto pelo navegador e quais são os elementos desse objeto.

No centro temos o conteúdo da nossa tag, em volta dele temos o padding que é um espaço entre o nosso conteúdo e os limites da tag.

Logo após temos o border, ou seja, uma borda que temos em volta da nossa tag.

Acima temos um margin, ou seja, um espaço entre as tags.

Com base nessas informações vamos analisar o resultado que temos.

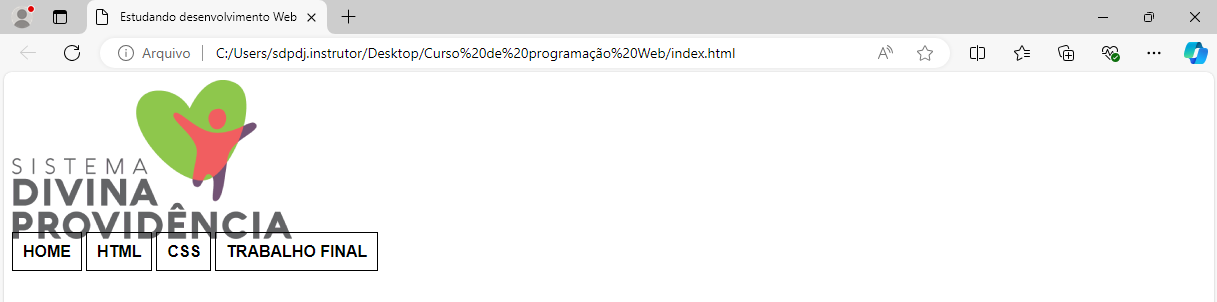


* O conteúdo das nossas tags são os textos dentro delas.
* O espaço que temos entre o texto e a borda, chamamos de padding.

Vamos aumentar esse padding. Para isso digite no seu VSCode:

  padding: 10px;

Veja o resultado disso:



Não se preocupe, ainda, com a nossa tag invadindo o espaço da imagem, isso será corrigido, apenas veja que o espaço entre o conteúdo e a borda foram alterados.

Agora observe que temos um espaço entre uma borda lateral de uma tag a outra. Isso se dá ao fato de termos uma margem (margin) entre eles. Vamos remove-la.