

PROGRAMAÇÃO WEBII

1

Criando componentes

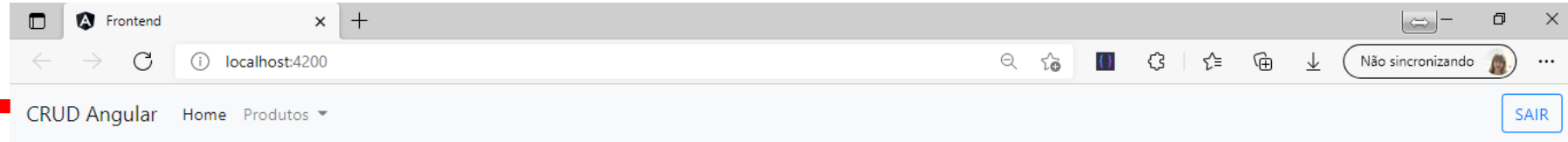
Montando um template

Iniciando...

- Vamos criar o template do nosso projeto utilizando alguns componentes:
 - Vamos navegar até a pasta do nosso projeto
 - Acessar prompt de comando
 - `cd\angular\cursoPWEBII\frontend <ENTER>`
 - `code . <ENTER>`
- Irá ser exibido o último arquivo que trabalhamos na aula anterior.
- Vamos fechar e retornar ao prompt de comando

Objetivo Final – Criação de Componentes

header



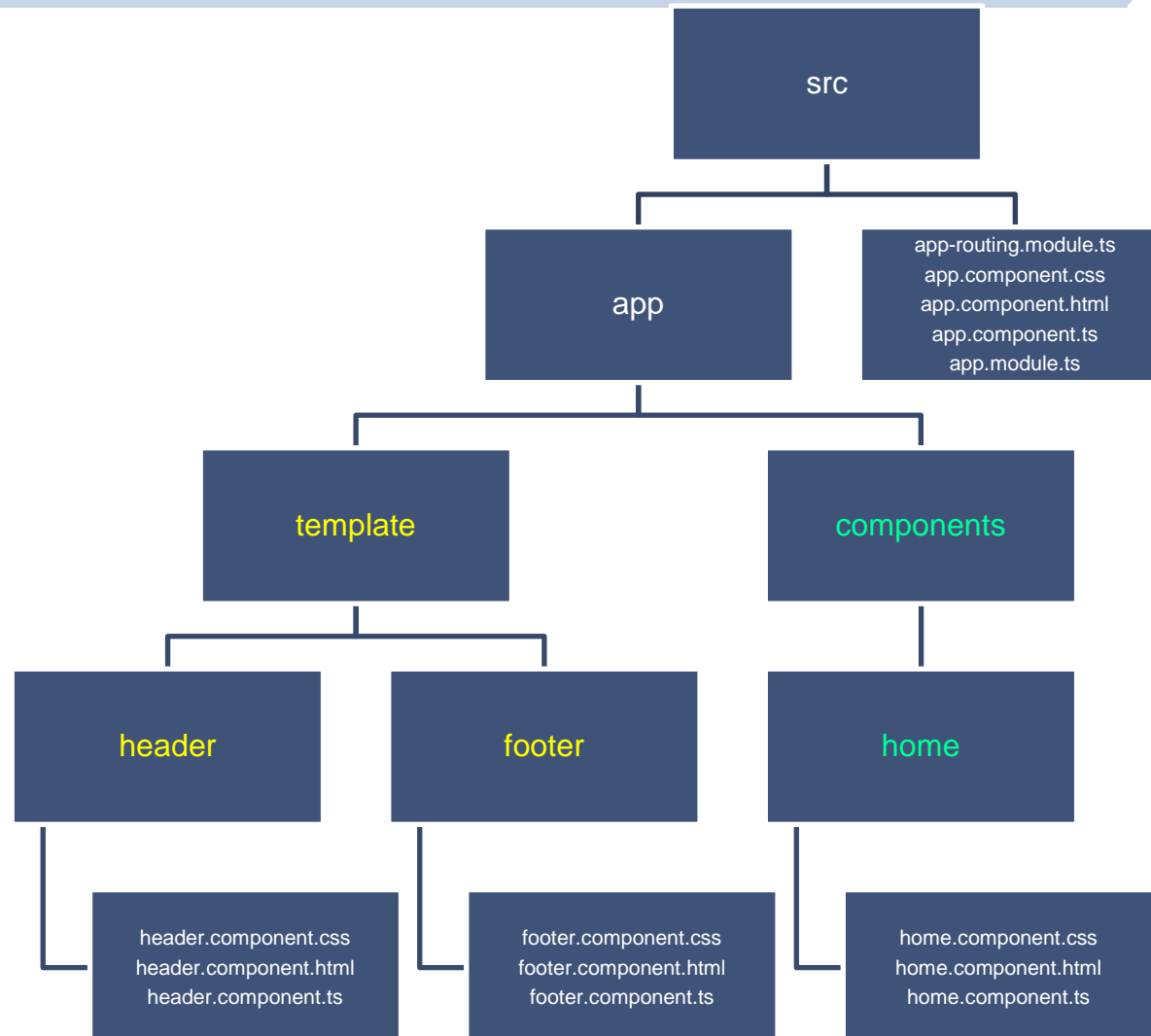
Seja bem vindo ao curso de Angular!



home

footer

Estrutura final



Componentes do bootstrap

Vamos conhecer um pouco sobre os componentes do bootstrap que utilizaremos hoje.
Acesse:

<https://getbootstrap.com/>

Lembrando que precisamos trocar para a documentação 4.6

Components

Alerts

Badge

Breadcrumb

Buttons

Button group

Card

Carousel

Collapse

Dropdowns

Forms

Input group

Jumbotron

List group

Media object

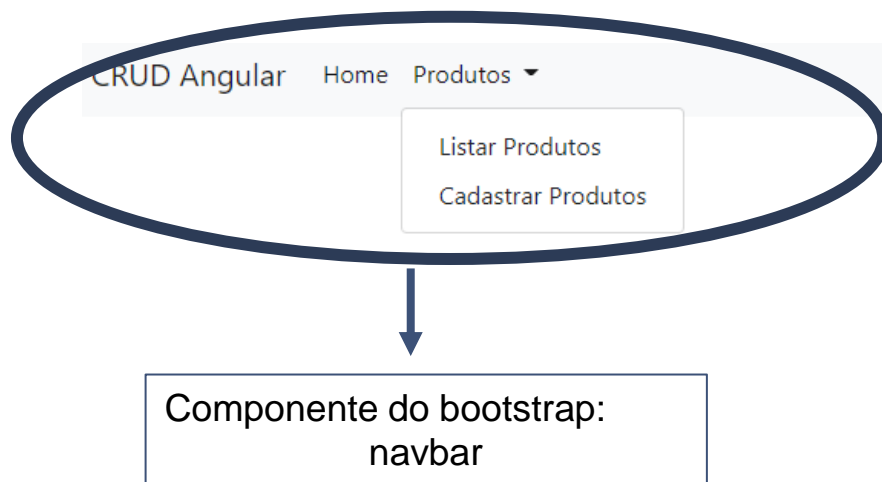
Modal

Navs

Navbar

header.html

Após as alterações, ficará assim:



home.html

Após as alterações, ficará assim:

Seja bem vindo ao curso de Angular!



CREATE

C



READ

R



UPDATE

U



DELETE

D

footer.html

Após as alterações, ficará assim:

Fátima Marques @ Curso Angular



Componentes do bootstrap:
nav
fixed bottom
span class="navbar-text"

app.component.html - container

Finalmente iremos finalizar colocando o conteúdo interno da nossa página dentro de um container.

CRUD Angular Home Produtos ▾

Sair

Seja bem vindo ao curso de Angular!



Criando componentes

- O comando para criar componentes é:
`ng generate component <arquivo>`
- Porém, iremos organizar os componentes do nosso projeto em pastas para facilitar a organização e manutenção do nosso código. Iremos criar a pasta `template` e o arquivo será o `header`:
`ng generate component template/header <ENTER>`

```
>ng g c template/header
```

```
CREATE src/app/template/header/header.component.html (21 bytes)
CREATE src/app/template/header/header.component.ts (275 bytes)
CREATE src/app/template/header/header.component.css (0 bytes)
UPDATE src/app/app.module.ts (484 bytes)
```

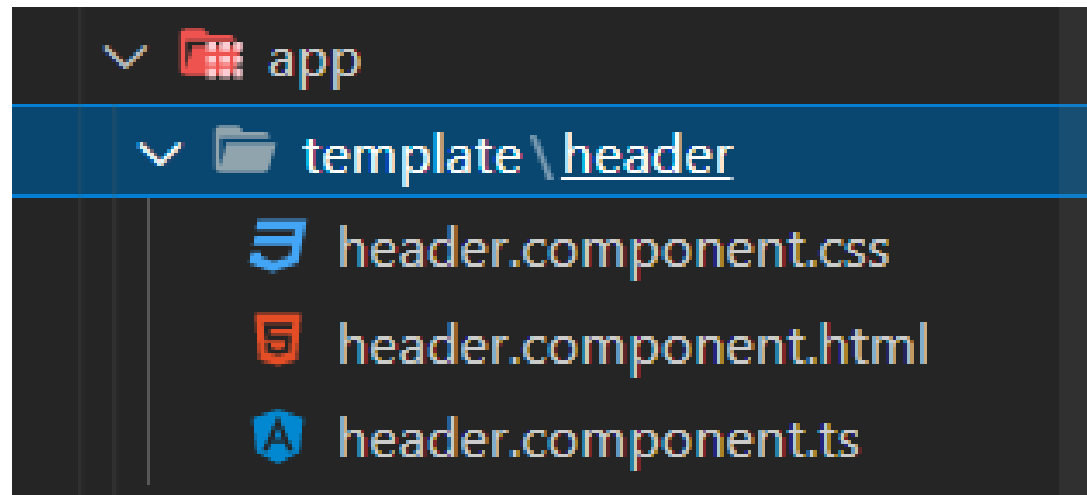
Criando componentes

- Vamos retornar para o VsCode e verificar o arquivo app.module.ts
- Já temos o HeaderComponent importado automaticamente
- Lembram na aula de Module aprendemos que temos que importar dentro de declarations todos os componentes que iremos utilizar na nossa aplicação.
- Aqui temos o HeaderComponent dentro de declarations

```
1  import { NgModule } from '@angular/core';
2  import { BrowserModule } from '@angular/platform-browser';
3
4  import { AppRoutingModule } from './app-routing.module';
5  import { AppComponent } from './app.component';
6  import { HeaderComponent } from './template/header/header.component';
7
8  @NgModule({
9    declarations: [
10     AppComponent,
11     HeaderComponent
12   ],
```

Criando componentes

O próximo passo é ir lá no template e verificarmos o arquivo header (css, html e ts):



Criando componentes

- Agora vamos criar outro elemento.
- Retornamos para o prompt de comando
- Ao selecionar a seta para cima, irá trazer o último comando digitado. Vamos agora aprender outra maneira de gerar os componentes:
- Agora vamos gerar o footer (que será a parte de baixo da nossa página, o rodapé).
ng g c template/footer <ENTER>
- Novamente criou os nossos arquivos e também fez atualização dentro do app.module.ts

```
C:\angular\cursopwebII\frontend>ng g c template/footer
CREATE src/app/template/footer/footer.component.html (21 bytes)
CREATE src/app/template/footer/footer.component.ts (275 bytes)
CREATE src/app/template/footer/footer.component.css (0 bytes)
UPDATE src/app/app.module.ts (575 bytes)
```

Criando componentes

Vamos acessar o arquivo app.module.ts e verificar que já foi atualizado

```
1  import { NgModule } from '@angular/core';
2  import { BrowserModule } from '@angular/platform-browser';
3
4  import { AppRoutingModule } from './app-routing.module';
5  import { AppComponent } from './app.component';
6  import { HeaderComponent } from './template/header/header.component';
7  import { FooterComponent } from './template/footer/footer.component';
8
9  @NgModule({
10   declarations: [
11     AppComponent,
12     HeaderComponent,
13     FooterComponent
14   ],
```

Criando componentes

- Agora vamos criar também a página de home, a página inicial do nosso projeto.
- Retornamos para o prompt de comando
- Ao selecionar a seta para cima, irá trazer o último comando digitado.
- Agora vamos gerar o home (página inicial do nosso projeto). Nós iremos criar uma pasta para criar as páginas do nosso projeto.

ng g c components/home <ENTER>

```
C:\angular\cursopwebII\frontend>ng g c components/home
CREATE src/app/components/home/home.component.html (19 bytes)
CREATE src/app/components/home/home.component.ts (267 bytes)
CREATE src/app/components/home/home.component.css (0 bytes)
UPDATE src/app/app.module.ts (660 bytes)
```

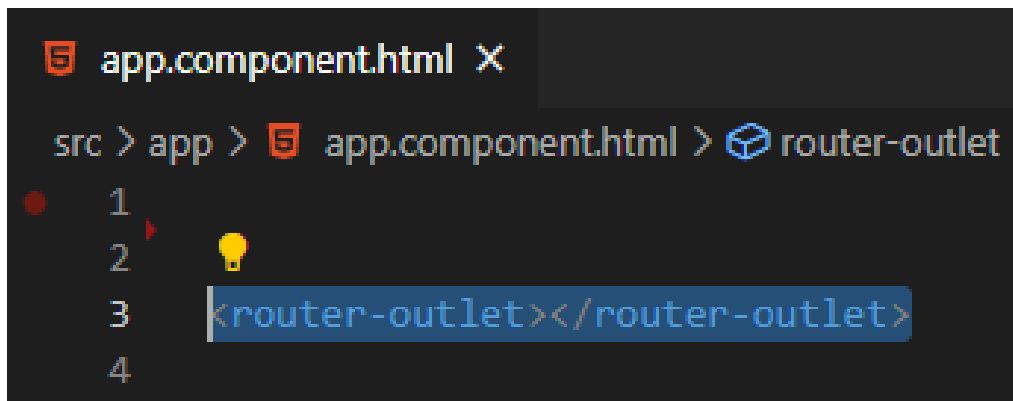

Criando componentes

No app.module.ts teremos agora os 3 componentes que criamos, automaticamente pelo Angular:

```
1  import { NgModule } from '@angular/core';
2  import { BrowserModule } from '@angular/platform-browser';
3
4  import { AppRoutingModule } from './app-routing.module';
5  import { AppComponent } from './app.component';
6  import { HeaderComponent } from './template/header/header.component';
7  import { FooterComponent } from './template/footer/footer.component';
8  import { HomeComponent } from './components/home/home.component';
9
10 @NgModule({
11   declarations: [
12     AppComponent,
13     HeaderComponent,
14     FooterComponent,
15     HomeComponent
16   ],
```

Criando componentes

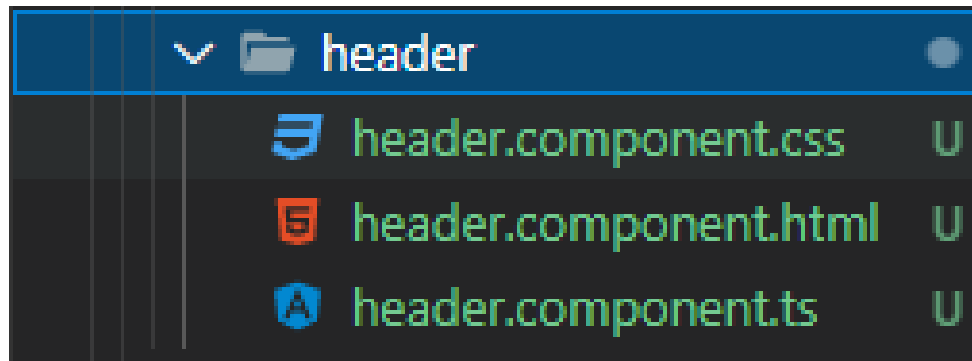
- Agora vamos abrir o arquivo `app.component.html`
- Vamos apagar todo o conteúdo e deixar somente o `router-outlet`
- Vamos salvar o arquivo desse jeito, por enquanto.



```
app.component.html X  
src > app > app.component.html > router-outlet  
1  
2  
3 <router-outlet></router-outlet>  
4
```

Criando componentes

- Agora vamos ver os nossos templates, começando pelo header.
- A estrutura atual da pasta contém:



Criando componentes

- No arquivo header.html contém a estrutura da nossa página. Não precisamos declarar toda a estrutura do nosso html, aqui só vamos declarar o html desse componente, só os pedaços de código desse componente.
- O arquivo css que está vazio.
- O arquivo typescript desse componente.

Criando componentes


■ Analisando o arquivo header.component.ts:

- ▶ Inseriu um decorator @Component
- ▶ Um seletor “app-header” (sempre irá colocar o prefixo do nosso aplicativo, se não alterarmos sempre será app e o nome do nosso componente).
- ▶ Depois teremos o template e o style. O template, como não estamos trabalhando com o template inline, então irá especificar o nome do arquivo: header.component.html e o style igual, irá chamar o arquivo css.
- ▶ Depois faz a exportação da classe implementando OnInit, o OnInit iremos aprender mais para frente como funciona.

```
1 import { Component, OnInit } from '@angular/core';
2
3 @Component({
4   selector: 'app-header',
5   templateUrl: './header.component.html',
6   styleUrls: ['./header.component.css']
7 })
8 export class HeaderComponent implements OnInit {
9
10  constructor() { }
11
12  ngOnInit(): void {
13  }
```

Criando componentes

- Agora vamos colocar o nosso componente para ser exibido na nossa tela.
- Vamos abrir o arquivo `app.component.html`
 - ▶ <CTRL><A> para selecionar todos os comandos desse arquivo
 - ▶ A última linha é o comando `router-outlet` (essa linha deverá permanecer)
 - ▶ Segure a tecla <SHIFT> antes dessa linha, seta para cima e a linha não será selecionada
 - ▶ Tecla para apagar tudo
- Vamos digitar:
 `app-header` <TAB> -> já irá preencher o restante para nós
- Salve o arquivo
- Vamos rodar o servidor para ver nossa página

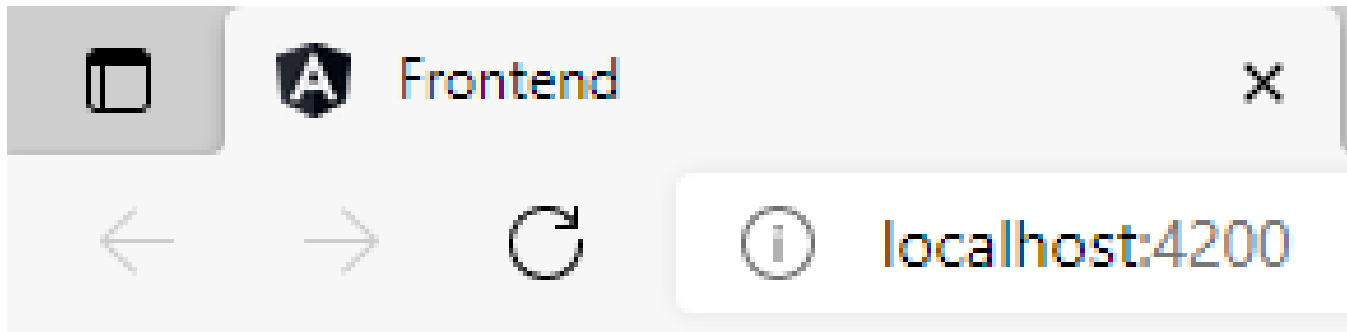
```
src > app >  app.component.html > ...  
1 | <app-header></app-header>  
2 |  
3 | <router-outlet></router-outlet>  
4 |
```

Criando componentes

■ Para que possamos ver nosso projeto rodando, então vamos rodar o servidor Angular:

- ▶ Ir no prompt de comando e digitar o comando abaixo. Aguardar carregar a aplicação

```
ng serve -o <ENTER>
```



header works!

Criando componentes

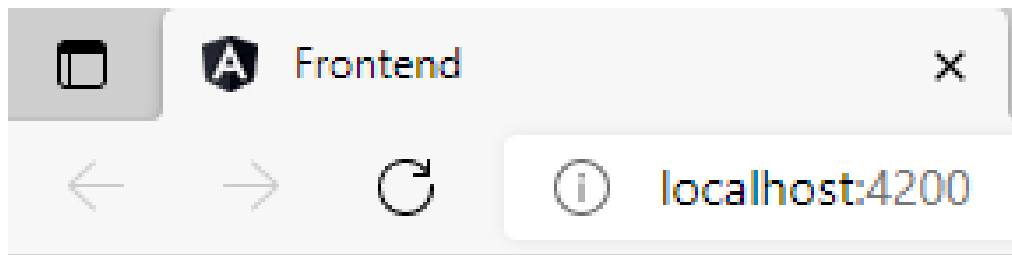
- Vamos adicionar o componente home que foi criado.
- No arquivo `app.component.html` digitar:
 `app-home <TAB>` -> já irá preencher o restante para nós
- De onde buscamos o nome `app-home`. Se você for até o arquivo `header.component.ts` você irá verificar o nome que está descrito como seletor. Esse nome é o que devemos chamar no `app.component.html`

```
app.component.html

src > app > app.component.html > app-home
1 | <app-header></app-header>
2 | <app-home></app-home>
3 |
4 | <router-outlet></router-outlet>
5 |
```


Criando componentes

■ Vamos acessar o browser e verificar como ficou nossa tela:



header works!

home works!

Criando componentes

Agora vamos adicionar o footer também

app-footer <TAB> -> já irá preencher o restante para nós

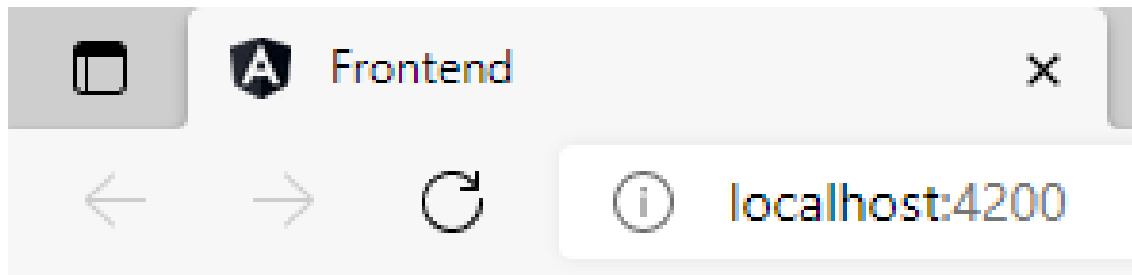
app.component.html

src > app > app.component.html > ...

```
1 <app-header></app-header>
2 <app-home></app-home>
3 <app-footer></app-footer>
4
5 <router-outlet></router-outlet>
6
```

Criando componentes

Vamos acessar o browser e verificar como ficou nossa tela. Já temos o header, home e footer:



header works!

home works!

footer works!

Estilizando nossa página

- Próximo passo seria estilizar nossa página utilizando o bootstrap.
- Vamos acessar a página do bootstrap:
<https://getbootstrap.com/>
- Lembrando que precisamos trocar para a documentação 4.6
- Vamos selecionar componentes

Estilizando nossa página

Vamos iniciar pelo nosso header

- ▶ Vamos até navbar
- ▶ Será exibido um exemplo para nós desse comando e vamos utilizar ele mesmo. Selecione todo o código e copie <CTRL><C>

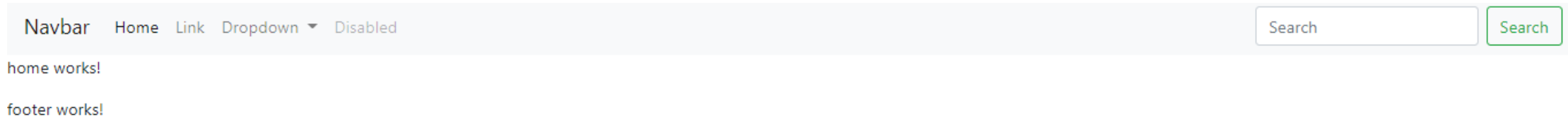
```
Navbar Home Link Dropdown Disabled Search

<nav class="navbar navbar-expand-lg navbar-light bg-light">
  <a class="navbar-brand" href="#">Navbar</a>
  <button class="navbar-toggler" type="button" data-toggle="collapse" data-target="#navbarSupportedContent" aria-controls="navbarSupportedContent" aria-expanded="false" aria-label="Toggle navigation">
    <span class="navbar-toggler-icon"></span>
  </button>

  <div class="collapse navbar-collapse" id="navbarSupportedContent">
    <ul class="navbar-nav mr-auto">
      <li class="nav-item active">
        <a class="nav-link" href="#">Home <span class="sr-only">{current}</span></a>
      </li>
      <li class="nav-item">
        <a class="nav-link" href="#">Link</a>
      </li>
      <li class="nav-item dropdown">
        <a class="nav-link dropdown-toggle" href="#" id="navbarDropdown" role="button" data-toggle="dropdown" aria-haspopup="true" aria-expanded="false">
          Dropdown
        </a>
        <div class="dropdown-menu" aria-labelledby="navbarDropdown">
          <a class="dropdown-item" href="#">Action</a>
          <a class="dropdown-item" href="#">Another action</a>
          <div class="dropdown-divider"></div>
          <a class="dropdown-item" href="#">Something else here</a>
        </div>
      </li>
      <li class="nav-item">
        <a class="nav-link disabled" href="#" tabindex="-1" aria-disabled="true">Disabled</a>
      </li>
    </ul>
    <form class="form-inline my-2 my-lg-0">
      <input class="form-control mr-sm-2" type="search" placeholder="Search" aria-label="Search">
      <button class="btn btn-outline-success my-2 my-sm-0" type="submit">Search</button>
    </form>
  </div>
</nav>
```

Estilizando nossa página

- Acesse o arquivo header.component.html
- Apague o que está escrito
- Cole o código copiado <CTRL><V>
- Salve o arquivo
- Vamos abrir a página e já temos a barra funcionando.



Estilizando nossa página

- Agora vamos personalizar..
- Aonde está nav, vamos alterar para CRUD Angular

```
1 <nav class="navbar navbar-expand-lg navbar-light bg-light">
2   <a class="navbar-brand" href="#">CRUD Angular</a>
3   <button class="navbar-toggler" type="button" data-toggle="collapse" data-target="#navbar" data-
4     <span class="navbar-toggler-icon"></span>
5   </button>
6
```

Estilizando nossa página

- Depois teremos o Home, que continuará
- Logo em seguida temos um link, onde iremos excluir ele. Excluir todo o conteúdo selecionado abaixo:

```
12 <li class="nav-item">  
13 |   <a class="nav-link" href="#">Link</a>  
14 </li>
```


Estilizando nossa página

E vamos substituir a palavra DropDown para Produtos:

```
13 <li class="nav-item dropdown">
14   <a class="nav-link dropdown-toggle" href="#" id="navbarDropdown"
15     Dropdown
16   </a>
```

```
13 <li class="nav-item dropdown">
14   <a class="nav-link dropdown-toggle" href="#" id="navbarDropdown"
15     Produtos
16   </a>
```

Vamos ver como ficou nossa página

CRUD Angular Home Produtos ▼ Disabled

home works!

footer works!

Estilizando nossa página

Agora vamos alterar as ações:

- ▶ Listar Produtos
- ▶ Cadastrar produtos
- ▶ Última ação, iremos apagar.

ANTES:

```
17 <div class="dropdown-menu" aria-labelledby="navbarDropdown">
18   <a class="dropdown-item" href="#">Action</a>
19   <a class="dropdown-item" href="#">Another action</a>
20   <div class="dropdown-divider"></div>
21   <a class="dropdown-item" href="#">Something else here</a>
22 </div>
```

DEPOIS:

```
15   Produtos
16   </a>
17   <div class="dropdown-menu" aria-labelledby="navbarDropdown">
18     <a class="dropdown-item" href="#">Listar Produtos</a>
19     <a class="dropdown-item" href="#">Cadastrar Produtos</a>
20   </div>
```

Estilizando nossa página

Depois teremos mais um item da nossa lista que será desativado. Podemos eliminar:

```
22     <li class="nav-item">
23       <a class="nav-link disabled" href="#" tabindex="-1" aria-disabled="true">
24     </li>
```

Estilizando nossa página

- Por fim, temos um form com o botão de buscar.
- Vamos excluir a linha do comando input

ANTES:

```
24 <form class="form-inline my-2 my-lg-0">
25   <input class="form-control mr-sm-2" type="search" placeholder="Se
26   <button class="btn btn-outline-success my-2 my-sm-0" type="submit"
27 </form>
```

DEPOIS:

```
24 <form class="form-inline my-2 my-lg-0">
25   <button class="btn btn-outline-success my-2 my-sm-0" type="subn
26 </form>
```

Estilizando nossa página

- No botão, alterar success para primary
- E onde está Search, alterar para Sair

```
24 <form class="form-inline my-2 my-lg-0">  
25   <button class="btn btn-outline-success my-2 my-sm-0" type="submit">Search</button>  
26 </form>
```

```
24 <form class="form-inline my-2 my-lg-0">  
25   <button class="btn btn-outline-primary my-2 my-sm-0" type="submit">SAIR</button>  
26 </form>
```

Estilizando nossa página

Vamos ver como ficou nossa aplicação

Se o jquery não estiver instalado ou com problema, o menu não funcionará.

CRUD Angular Home Produtos ▼

SAIR

home works!

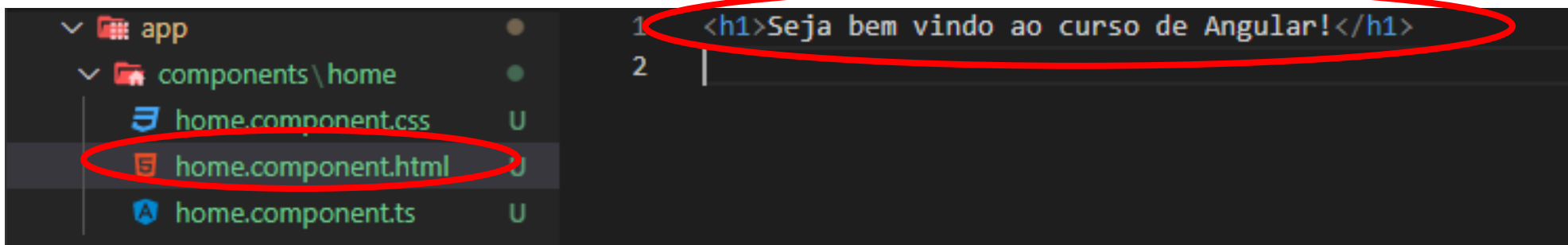
footer works!

Listar Produtos

Cadastrar Produtos

Estilizando nossa página

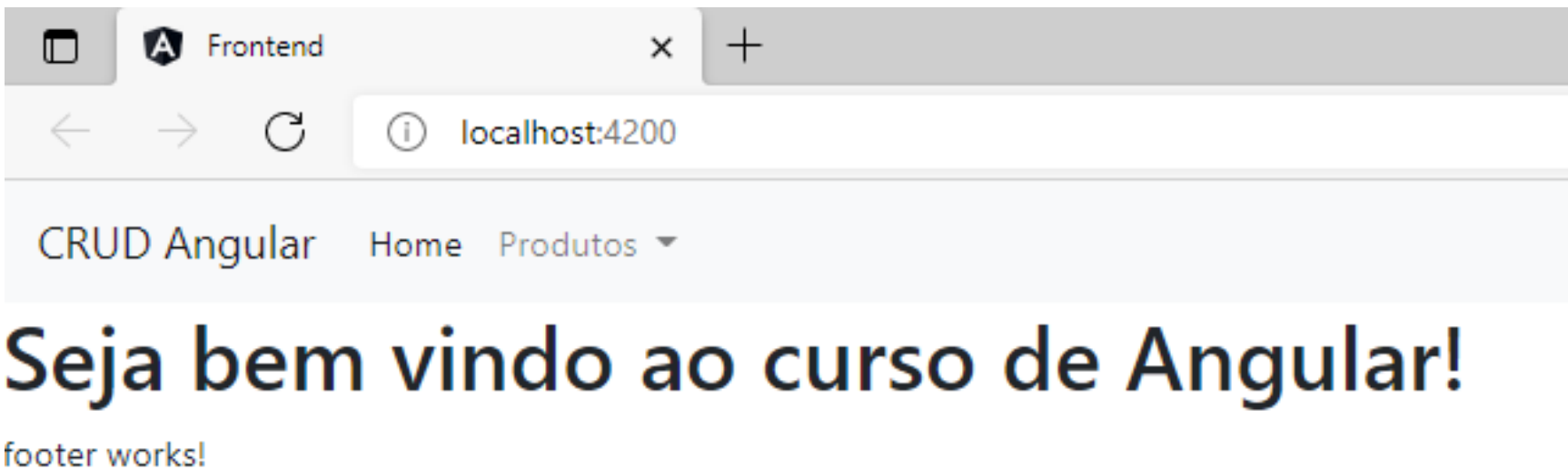
- Nosso header agora está pronto.
- Vamos agora para o nosso home. O que iremos inserir no nosso home?
- Vamos alterar somente o arquivo [home.component.html](#)
- E vamos inserir a tag h1, com o título “Seja bem vindo ao curso de Angular!”
- Vamos salvar e verificar como ficou nossa página.



```
1 <h1>Seja bem vindo ao curso de Angular!</h1>
2
```

Estilizando nossa página

■ Segue nossa página até o momento:



Estilizando nossa página

Continuando no home, agora vamos inserir uma div com uma class row. Para isso, digite:

```
div.row <TAB>
```

Depois uma div com uma class col

```
div.col <TAB>
```

Na coluna iremos importar uma imagem que é a da ETEC, na pasta assets/img

```
img <TAB>
```

Vamos criar outra coluna

```
div.col<TAB>
```

Nessa coluna vamos inserir a imagem de CRUD

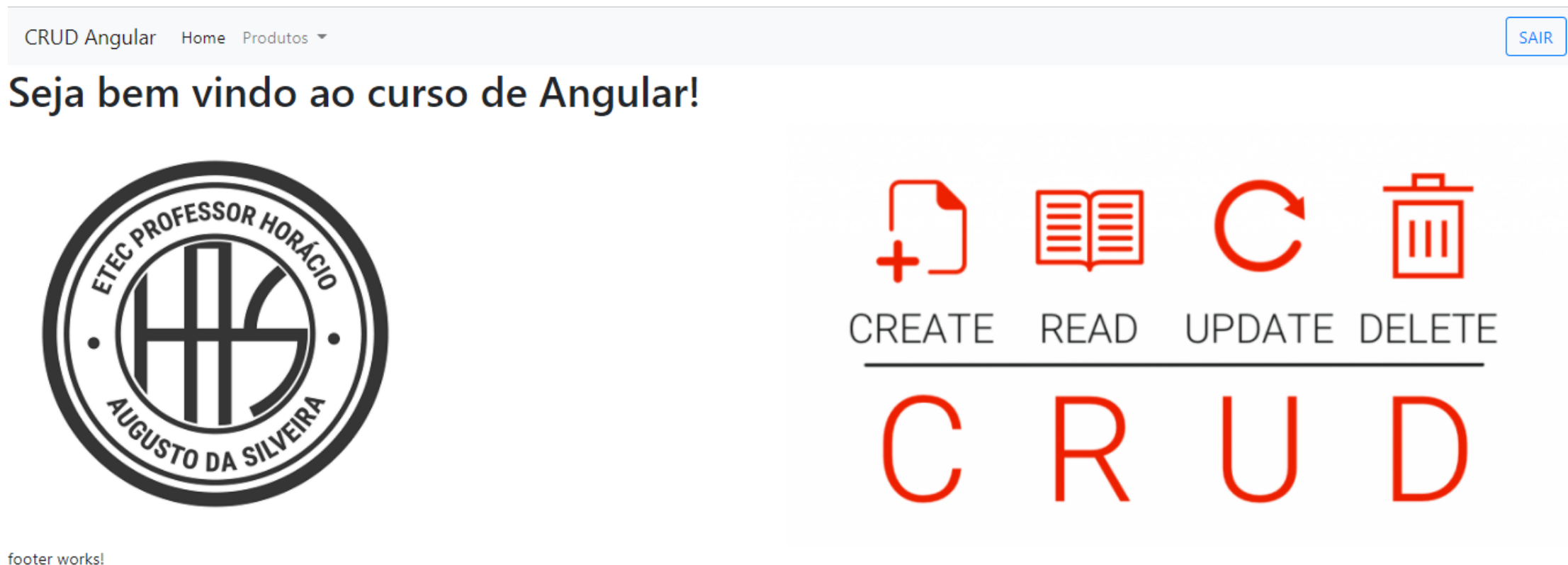
Estilizando nossa página

■ Segue as implementações a serem realizadas:

```
home.component.html X
src > app > components > home > home.component.html > ...
1  <h1>Seja bem vindo ao curso de Angular!</h1>
2  <div class="row">
3    <div class="col">
4      
5    </div>
6    <div class="col">
7      
8    </div>
9  </div>
10
```

Estilizando nossa página

Nossa página home ficará assim:







Estilizando nossa página

- Agora vamos criar nosso footer.
- Para isso, vamos na documentação do bootstrap 4.6 e procurar pelo elemento navs

The screenshot shows the Bootstrap 4.6 documentation page for 'Navs'. The page has a purple header with navigation links: Home, Documentation (active), Examples, Icons, Themes, Expo, and Blog. On the right of the header, it shows 'v4.6', social media icons, and a 'Download' button. A left sidebar contains a search bar and a list of categories: List group, Media object, Modal, Navs (active), Navbar, and Pagination. The main content area features the title 'Navs' and a subtitle 'Documentation and examples for how to use Bootstrap's included navigation components.' Below this is the section 'Base nav'. On the right, there is a 'View on GitHub' button and a list of 'Available styles' including Base nav, Horizontal alignment, Vertical, Tabs, Pills, Fill and justify, and Working with flex.

Home **Documentation** Examples Icons Themes Expo Blog

v4.6     [Download](#)

Search...

List group
Media object
Modal
Navs
Navbar
Pagination

Navs

Documentation and examples for how to use Bootstrap's included navigation components.

[View on GitHub](#)

Base nav

Available styles

- Horizontal alignment
- Vertical
- Tabs
- Pills
- Fill and justify
- Working with flex

Estilizando nossa página

Vamos abrir o código no arquivo footer.component.html

Apagar o conteúdo que existe

Vamos criar um div que possui o elemento navs

div.nav <TAB>

Próximo passo: Vir na documentação do bootstrap e descobrir como ficará fixo aqui em baixo.

Isso fica dentro de Utilities / Position e teremos as diversas posições: fixed top, fixed bottom

```
footer.component.html X
src > app > template > footer >
1  <div class="nav">
2
3
4  </div>
```

Search...

Utilities

Borders
Clearfix
Close icon
Colors
Display
Embed
Flex
Float
Image replacement
Interactions
Overflow
Position

Fixed top

Position an element at the top of the viewport, from edge to edge. Be sure you understand the ramifications of fixed position in your project; you may need to add additional CSS.

```
<div class="fixed-top">...</div>
```

Fixed bottom

Position an element at the bottom of the viewport, from edge to edge. Be sure you understand the ramifications of fixed position in your project; you may need to add additional CSS.

```
<div class="fixed-bottom">...</div>
```

Estilizando nossa página

- Vamos copiar a class de fixo embaixo: fixed bottom
- Vamos inserir no nosso projeto:

```
footer.component.html  
  
src > app > template > footer > footer.component.html  
1  <div class="nav fixed-bottom">  
2  
3  
4  </div>  
5
```

Estilizando nossa página

- Agora vamos inserir umas configurações de cores:
- Vamos inserir que será uma navbar-light e um bg-light (Ao começar a digitar o editor já exibirá as opções)
- Próximo passo é deixar o texto centralizado: justify-content-center

```
footer.component.html •
src > app > template > footer > footer.component.html > ...
1  <div class="nav fixed-bottom navbar-light bg-light justify-content-center">
2
3
4  </div>
5  |
```

Estilizando nossa página

- Dentro da div, vamos criar uma span
- Dentro da span vai ter uma class que será a navbar-text
- Abaixo iremos escrever o nosso nome
- Depois vamos inserir a tag Strong e dentro @ Curso Angular

📄 footer.component.html ✕

src > app > template > footer > 📄 footer.component.html > ...

```
1 <div class="nav fixed-bottom navbar-light bg-light justify-content-center">
2   <span class="navbar-text">
3     Fátima Marques <strong>@ Curso Angular</strong>
4   </span>
5 </div>
```


Estilizando nossa página

Vamos ver como ficou nossa página:

CRUD Angular Home Produtos ▾

SAIR

Seja bem vindo ao curso de Angular!



Estilizando nossa página

- Estamos quase lá... O próximo passo é colocar tudo isso dentro de um container.
- Vamos retornar no nosso código e acessar o app.component.html
- Vamos envolver o app-home dentro de uma div
- Trazer o app-home para dentro da div
- E na div criar uma class denominada container

app.component.html X

src > app > app.component.html > router-outlet

```
1 <app-header></app-header>
2 <div class="container">
3   <app-home></app-home>
4 </div>
5 <app-footer></app-footer>
6 <router-outlet></router-outlet>
```

Estilizando nossa página

- Vamos ver como ficou a página
- As imagens estão quebrando



Seja bem vindo ao curso de Angular!



Estilizando nossa página

- Precisamos criar a classe que deixará nossas imagens responsivas
- Vamos no home.component.html
- Vamos na tag img e inserir a class="img-fluid" que é a classe que deixa a imagem responsiva.

```
home.component.html X
src > app > components > home > home.component.html > ...
1  <h1>Seja bem vindo ao curso de Angular!</h1>
2  <div class="row">
3    <div class="col">
4      
5    </div>
6    <div class="col">
7      
8    </div>
9  </div>
```

Estilizando nossa página

- Agora sim nossa página está ajustada.
- As duas imagens estão centralizadas, o header e footer pegando a página inteira. O conteúdo interno da nossa página ficará dentro de um container.



Seja bem vindo ao curso de Angular!



Tarefa

A tarefa será disponibilizada no Teams

