

PROGRAMAÇÃO WEBII

1

Criando o projeto do curso

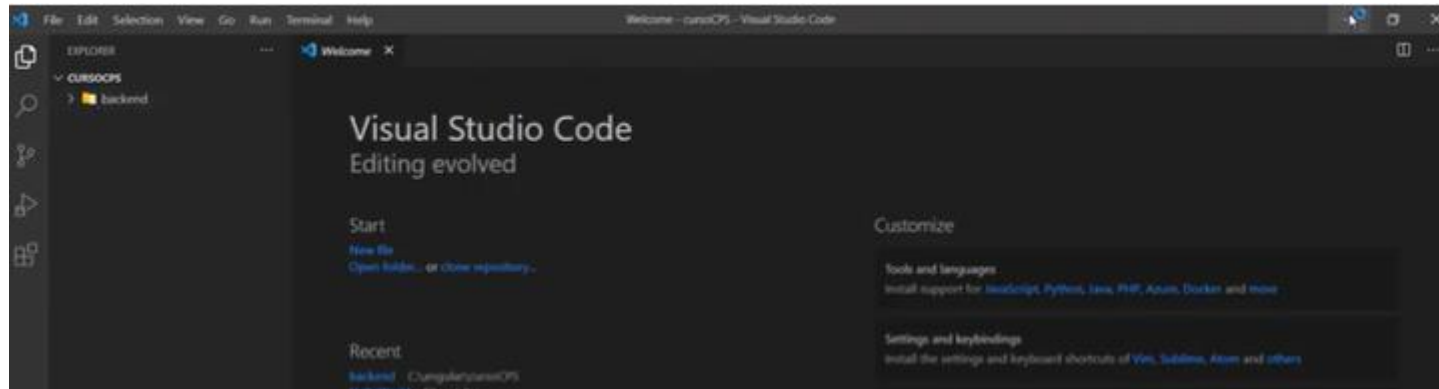
Conhecendo a estrutura de projetos
Angular

Acessando a pasta do projeto

- Vamos acessar a pasta do projeto
 - ▷ Ir para o prompt de comando
 - ▷ Ir para a pasta do projeto
 - ▷ `cd\angular\cursoPWEBII <ENTER>`
- Abrir o Visual Code
 - ▷ `code . <ENTER>`

Acessando a pasta do projeto

- Por enquanto só temos a pasta do backend



- Agora vamos criar nossa pasta do frontend que é o nosso projeto do Angular

Criando o projeto Frontend

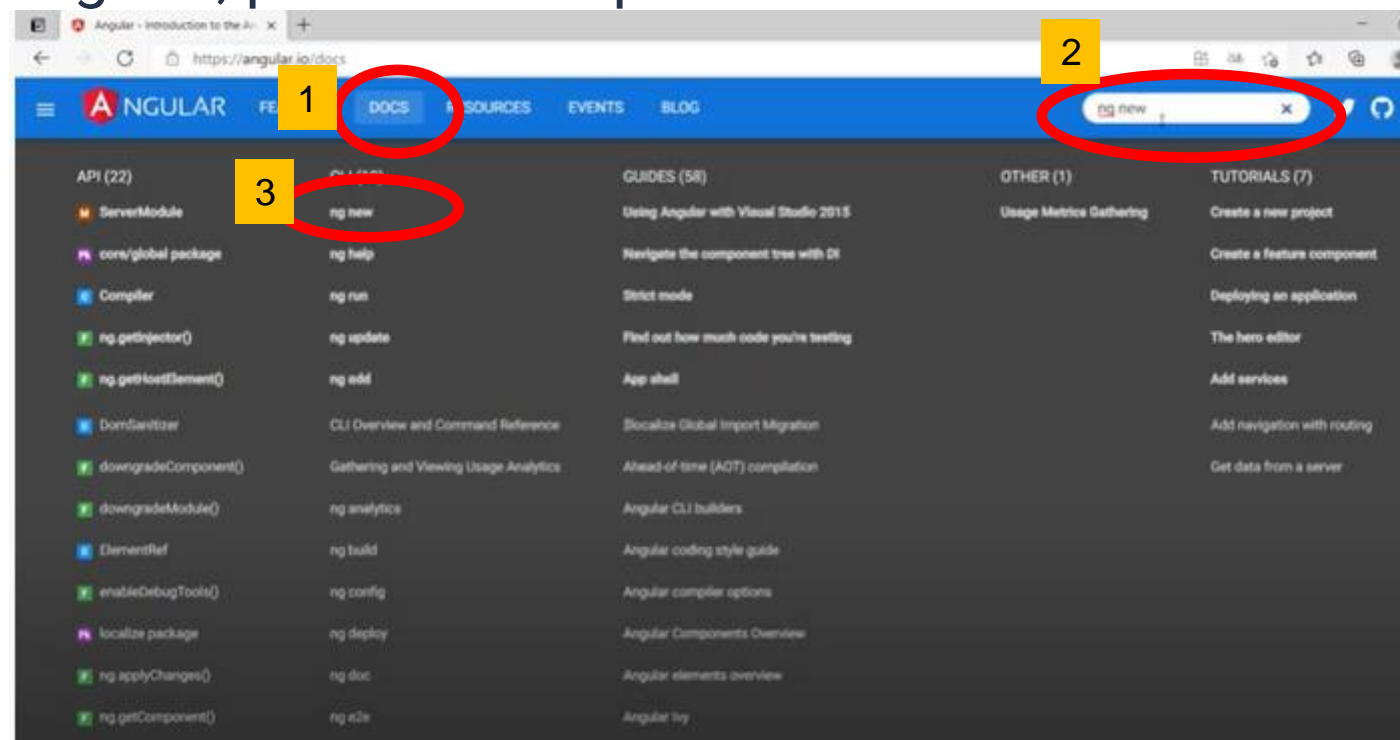
Vamos começar abrindo o site do Angular. Todo o nosso curso será baseado na documentação do Angular, por isso a importância em acessar o site.

► <https://angular.io/>

Selecionar a opção DOCS

Pesquisar por ng new

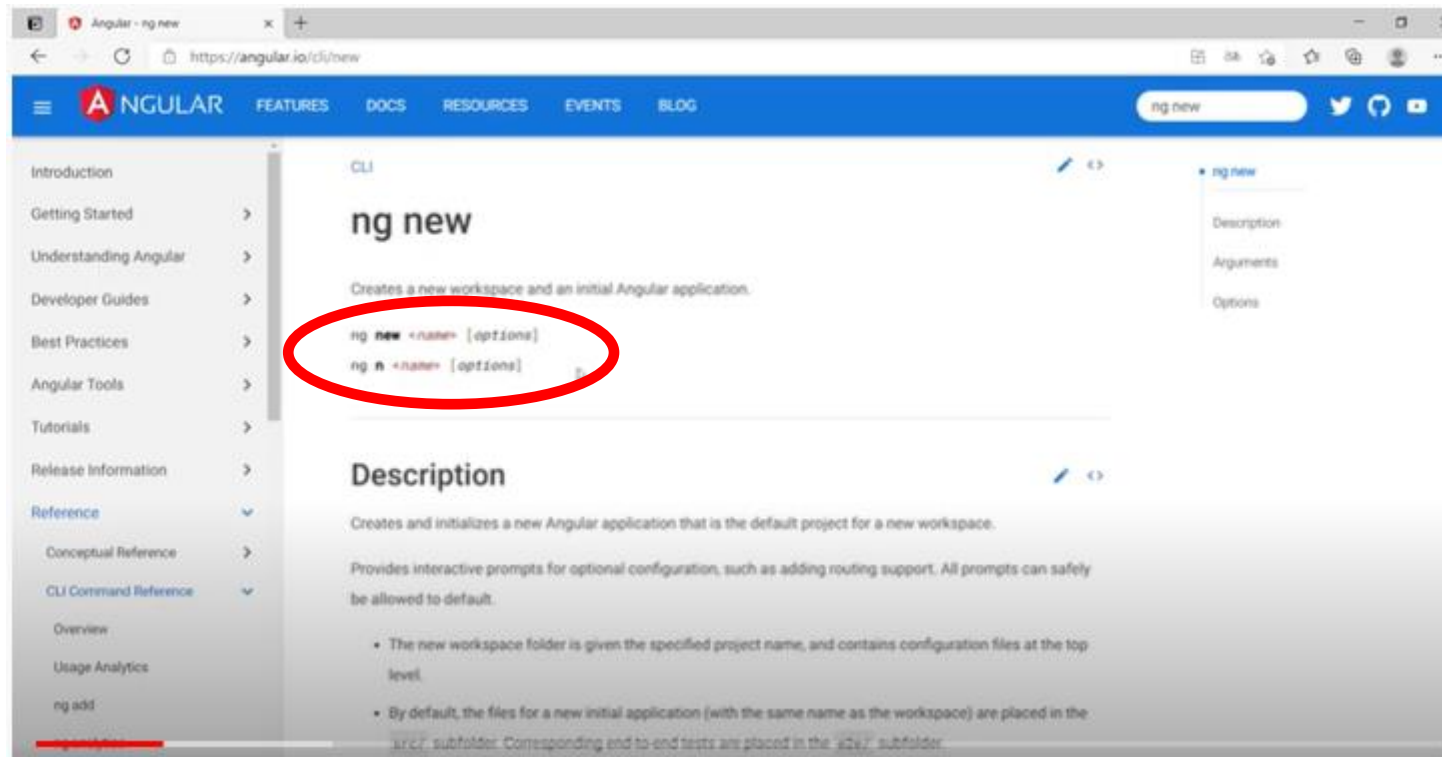
Clicar na opção ng new



Criando o projeto Frontend

■ Será exibida a tela abaixo, onde:

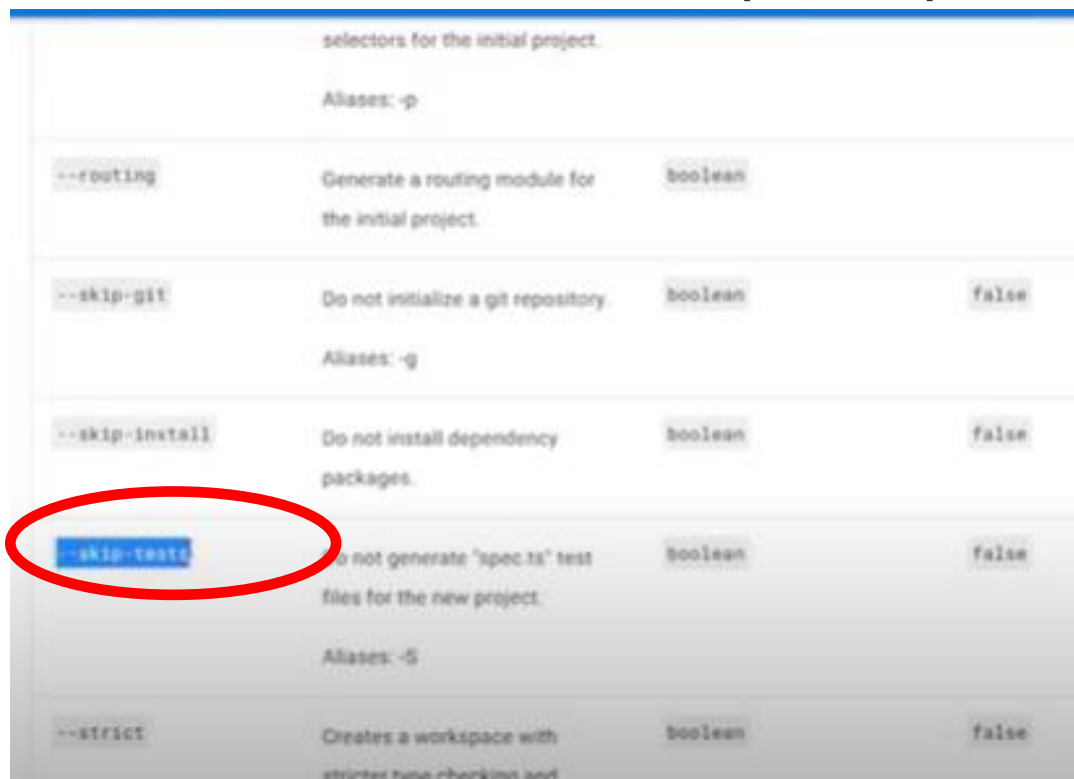
- ▶ `ng new` -> criamos um projeto
- ▶ `ng new options` -> criamos um projeto de acordo com as opções:



Criando o projeto Frontend

■ A opção que iremos utilizar no curso é a `--skip-tests`

- ▶ O foco do curso não é trabalhar com testes e o angular por padrão já cria automaticamente um arquivo `spec.ts` e fica pesado e poluído.



A screenshot of the Angular CLI help text for the `ng new` command. The table lists various options for the initial project. The option `--skip-tests` is highlighted with a red circle.

selectors for the initial project.			
Aliases: -p			
<code>--routing</code>	Generate a routing module for the initial project.	boolean	
<code>--skip-git</code>	Do not initialize a git repository.	boolean	false
Aliases: -g			
<code>--skip-install</code>	Do not install dependency packages.	boolean	false
<code>--skip-tests</code>	Do not generate "spec.ts" test files for the new project.	boolean	false
Aliases: -S			
<code>--strict</code>	Creates a workspace with stricter type checking and	boolean	false

Criando o projeto Frontend

- Sendo assim, voltemos ao prompt de comando e vamos digitar:
 - `ng new frontend --skip-tests <ENTER>`
- Seguir os passos abaixo:

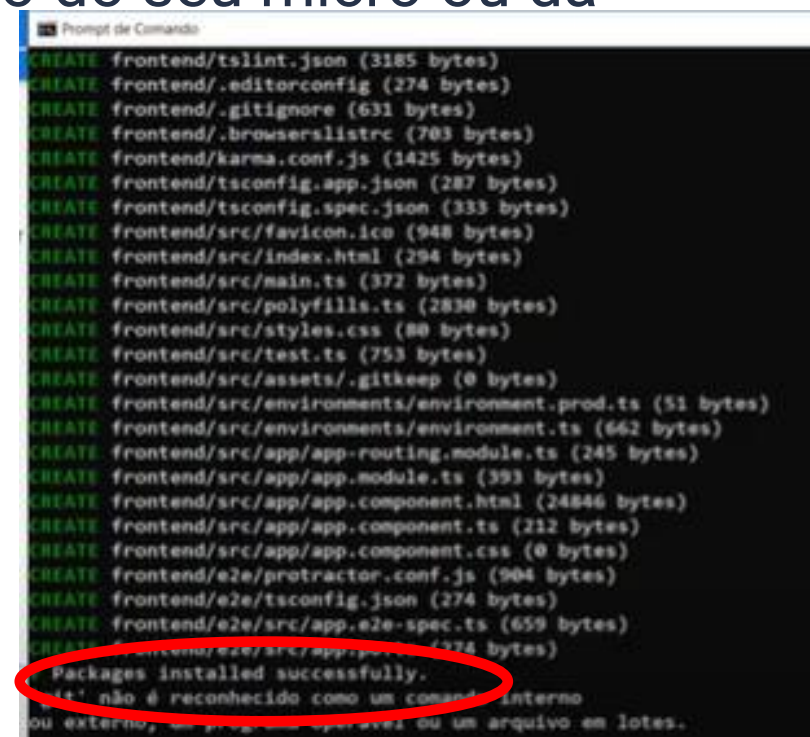
```
Do you want to enforce stricter type checking and stricter bundle budgets in the workspace?  
This setting helps improve maintainability and catch bugs ahead of time.  
For more information, see https://angular.io/strict No  
Would you like to add Angular routing? Yes  
Which stylesheet format would you like to use? (Use arrow keys)  
CSS  
SCSS [ https://sass-lang.com/documentation/syntax#scss ]  
Sass [ https://sass-lang.com/documentation/syntax#the-indented-syntax ]  
Less [ http://lesscss.org ]  
Stylus [ https://stylus-lang.com ]
```

Aguardar a criação
do projeto.
Lembrando... Não
clique na tela senão
o processo irá
pausar...

Vamos selecionar css, pois iremos trabalhar com o bootstrap.

Criando o projeto Frontend

- Assim que a criação do projeto for finalizada, será exibida uma mensagem.
- A criação pode demorar dependendo da configuração do seu micro ou da sua internet.
- E.. será realizado o download de vários arquivos.



```
Prompt de Comando
CREATE frontend/tslint.json (3185 bytes)
CREATE frontend/.editorconfig (274 bytes)
CREATE frontend/.gitignore (631 bytes)
CREATE frontend/.browserslistrc (703 bytes)
CREATE frontend/karma.conf.js (1425 bytes)
CREATE frontend/tsconfig.app.json (287 bytes)
CREATE frontend/tsconfig.spec.json (333 bytes)
CREATE frontend/src/favicon.ico (948 bytes)
CREATE frontend/src/index.html (294 bytes)
CREATE frontend/src/main.ts (372 bytes)
CREATE frontend/src/polyfills.ts (2830 bytes)
CREATE frontend/src/styles.css (80 bytes)
CREATE frontend/src/test.ts (753 bytes)
CREATE frontend/src/assets/.gitkeep (0 bytes)
CREATE frontend/src/environments/environment.prod.ts (51 bytes)
CREATE frontend/src/environments/environment.ts (662 bytes)
CREATE frontend/src/app/app-routing.module.ts (245 bytes)
CREATE frontend/src/app/app.module.ts (393 bytes)
CREATE frontend/src/app/app.component.html (24846 bytes)
CREATE frontend/src/app/app.component.ts (212 bytes)
CREATE frontend/src/app/app.component.css (0 bytes)
CREATE frontend/e2e/protractor.conf.js (904 bytes)
CREATE frontend/e2e/tsconfig.json (274 bytes)
CREATE frontend/e2e/src/app.e2e-spec.ts (659 bytes)
CREATE frontend/e2e/src/app.po.ts (374 bytes)
Packages installed successfully.
ts' não é reconhecido como um comando interno
ou externo, um programa operável ou um arquivo em lotes.
```

Verificando o conteúdo do projeto Frontend

- Vamos verificar os arquivos criados...
- Estando no prompt de comando, digite:
 - ▷ `explorer . <ENTER>`
- Serão exibidas as duas pastas:

Nome	Data de modificação	Tipo	Tamanho
backend	22/03/2021 00:35	Pasta de arquivos	
frontend	25/03/2021 02:53	Pasta de arquivos	

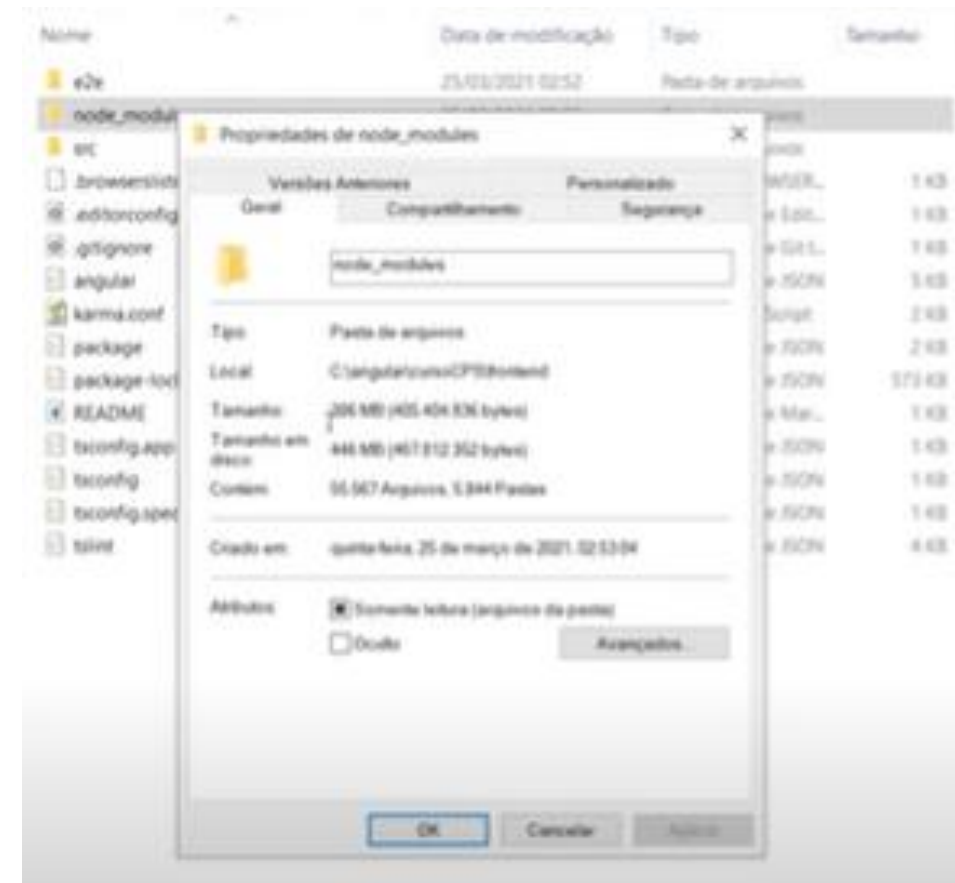
- Acessando a pasta frontend, verificamos os arquivos:

Nome	Data de modificação	Tipo	Tamanho
node_modules	25/03/2021 02:53	Pasta de arquivos	
src	25/03/2021 02:52	Pasta de arquivos	
src	25/03/2021 02:52	Pasta de arquivos	1 KB
editorconfig	25/03/2021 02:52	Arquivo Fonte .SRC	1 KB
gitignore	25/03/2021 02:52	Arquivo Fonte .GTL	1 KB
angular	25/03/2021 02:52	Arquivo Fonte .SON	3 KB
karma.conf	25/03/2021 02:52	Arquivo JavaScript	2 KB
package	25/03/2021 02:52	Arquivo Fonte .SON	2 KB
package-lock	25/03/2021 02:53	Arquivo Fonte .SON	372 KB
README	25/03/2021 02:52	Arquivo Fonte .Mae	1 KB
tsconfig.app	25/03/2021 02:52	Arquivo Fonte .SON	1 KB
tsconfig	25/03/2021 02:52	Arquivo Fonte .SON	1 KB
tsconfig.spec	25/03/2021 02:52	Arquivo Fonte .SON	1 KB
tslint	25/03/2021 02:52	Arquivo Fonte .SON	4 KB

Principal pasta é a node-modules

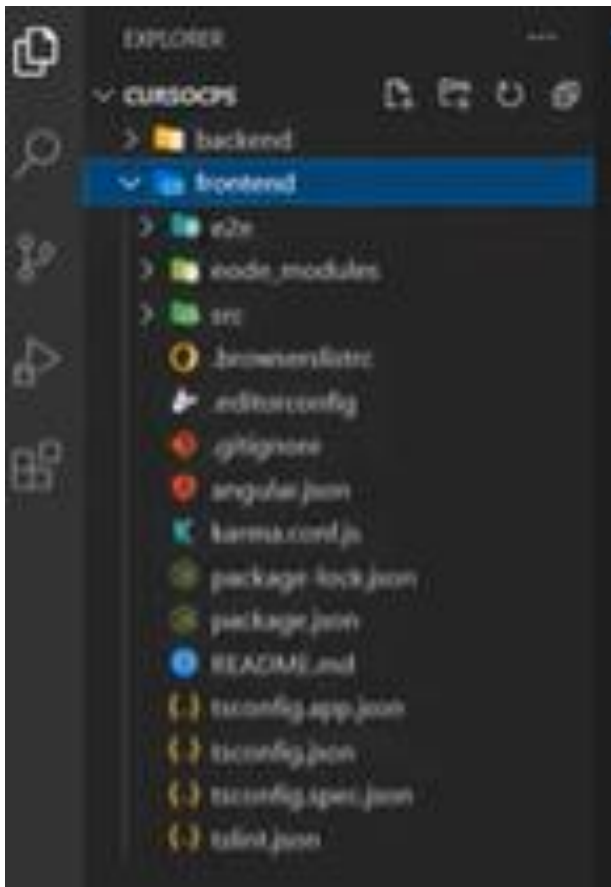
Verificando o conteúdo do projeto Frontend

- Foi feito um download de 500 Mb de arquivo (clique com o botão direito na pasta, propriedades:
- Por isso a demora na criação do projeto.



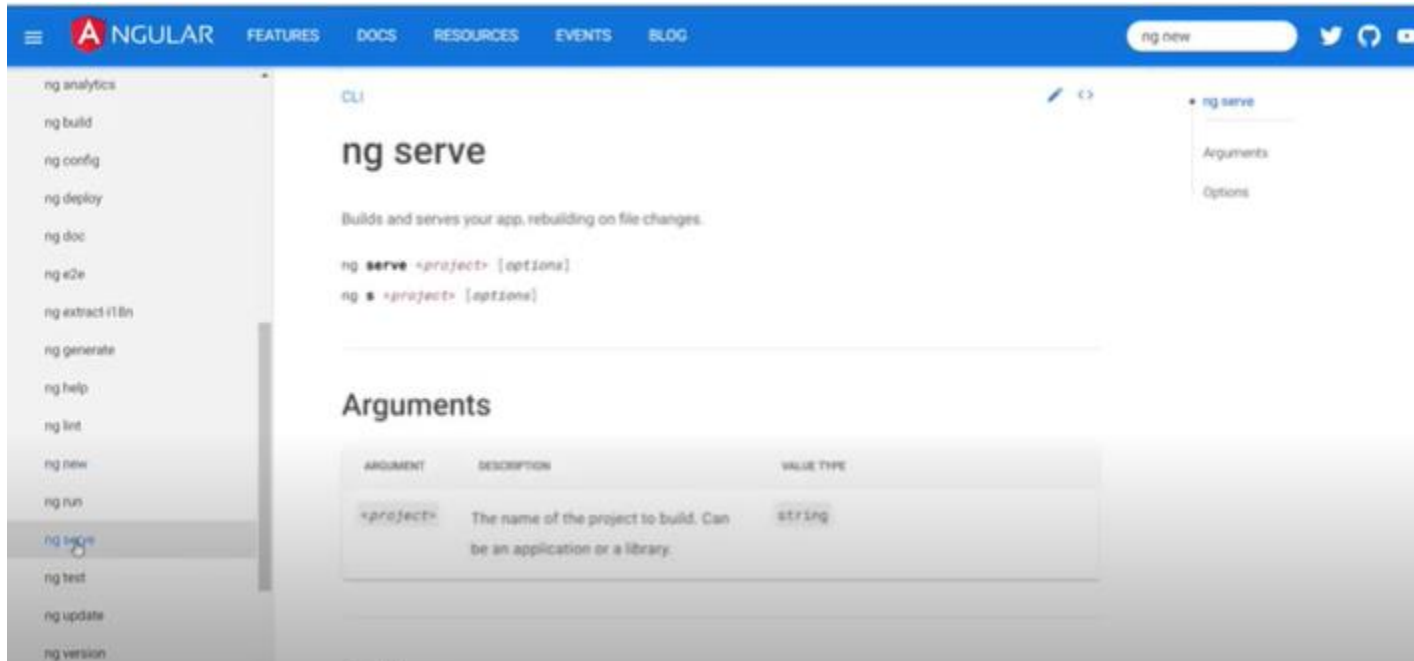
Verificando o conteúdo do projeto Frontend

Voltando ao Visual StudioCode, clique na pasta Frontend e teremos todos os arquivos criados referente ao nosso projeto



Subindo o servidor

- Vamos na nossa documentação do angular e verificar o comando para subir o servidor para ver nossa página criada.
- O comando para subir é o ng serve. Procure na lista e selecione:



Podemos digitar ng serve ou
ng serve options

Vamos ver as opções que temos....

Subindo o servidor

- Procure pela opção `--open` que já abrirá a página automaticamente.

<code>--hmr</code>	Enable hot module replacement.	boolean	<div>ng serve</div> <div>Arguments</div> <div>• Options</div>
<code>--hmr-warning</code>	Deprecated: No longer has an effect. Show a warning when the <code>--hmr</code> option is enabled.	boolean	
<code>--host</code>	Host to listen on.	string	
<code>--live-reload</code>	Whether to reload the page on change, using live-reload.	boolean	
<code>--open</code>	Opens the url in default browser. Aliases: <code>-o</code>	boolean	

Subindo o servidor

- Vamos acessar a pasta frontend (Atenção!!! Precisamos estar dentro da pasta...). Ir para o prompt e comando e digitar:
 - ▷ `cd frontend <ENTER>`
- Digitar o comando:
 - ▷ `ng serve -o <ENTER>`

`--open`

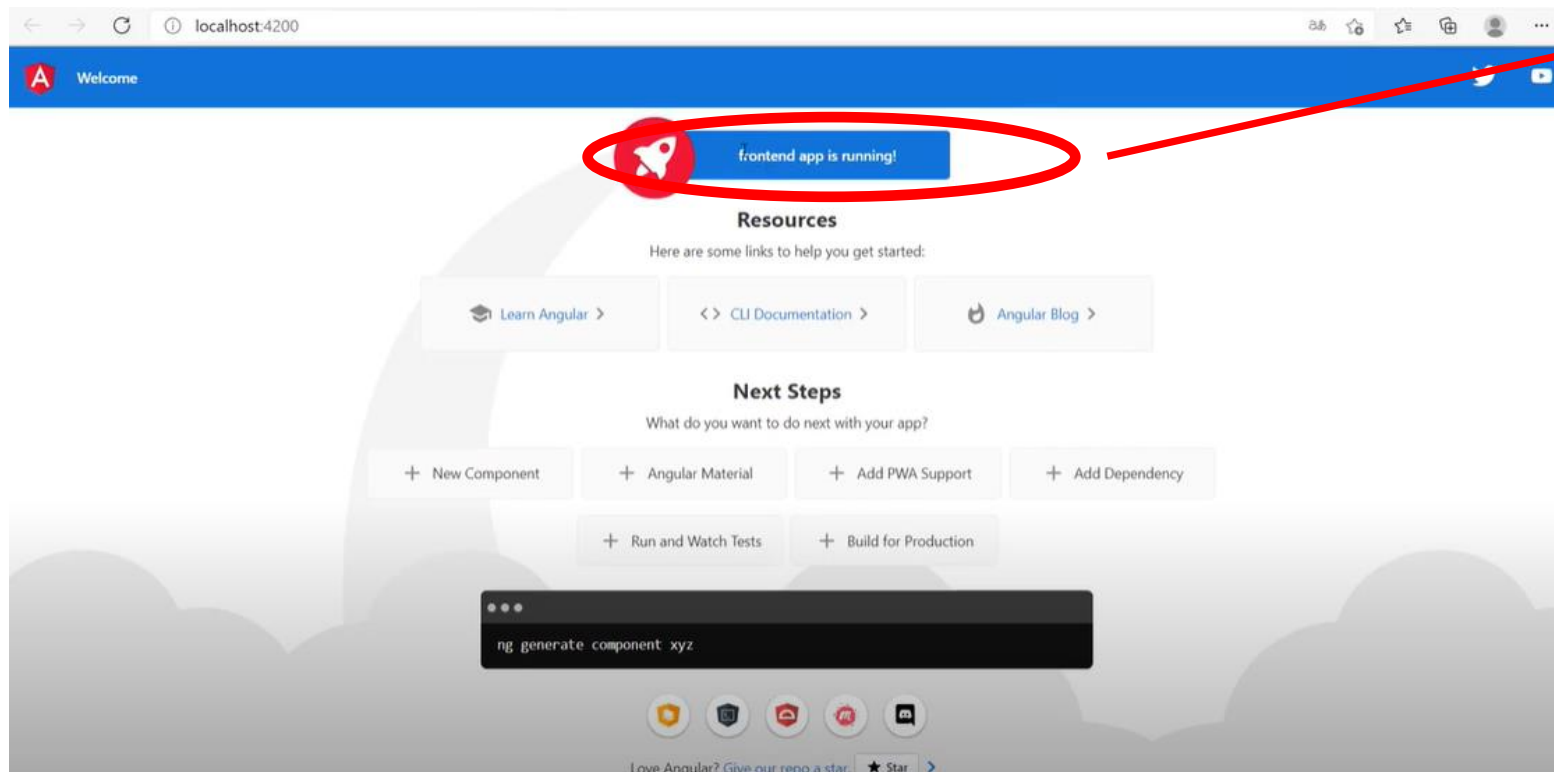
Opens the url in default browser.

Aliases: -o

```
C:\angular\cursopwebII\frontend>ng serve -o
? Would you like to share anonymous usage data about this project with the Angular Team at
Google under Google's Privacy Policy at https://policies.google.com/privacy? For more
details and how to change this setting, see https://angular.io/analytics. No
- Generating browser application bundles (phase: setup)...
```

Subindo o servidor

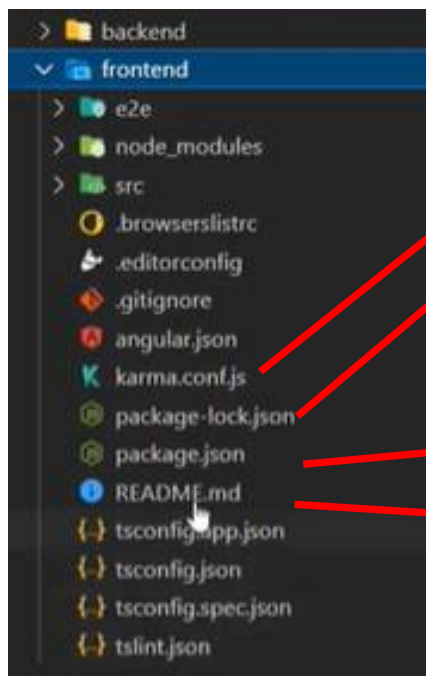
- Vamos aguardar...
- Podemos retornar para o VS Visual Code
- Daqui a pouco o navegador irá exibir a página...



Nome do projeto

Vamos entender a estrutura dos projetos Angular

Quando criamos um projeto no Angular, não importa qual das estruturas escolhemos, terá uma estrutura padrão que estará presente em todos os projetos.



É um arquivo de configuração de testes.

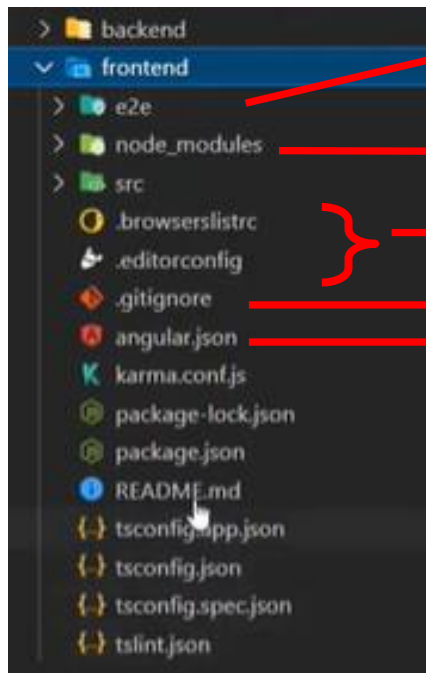
É um arquivo para travar as versões para que você não tenha problema de incompatibilidade.

Arquivos muito importante para o nosso projeto. É nele onde estão todas as configurações do node (nome do nosso projeto, versão, script (se digitarmos npm start será executado o ng serve). Depois temos os pacotes, as dependências que foram baixadas e instaladas para criar o nosso projeto. Note que exibe a versão do Angular que estamos utilizando. Aqui também teremos as dependências de desenvolvimento: devDependencies. Elas só são baixadas no nosso computador quando estamos desenvolvendo.

Arquivos para o github. Quando subir para o github você já terá uma descrição inicial do seu projeto, onde explica os comandos que são necessários para executar.

Arquivos de Configuração do TypeScript. Dificilmente vocês irão alterar esses arquivos

Vamos entender a estrutura dos projetos Angular



Pasta e2e onde contém os arquivos de teste para realizarmos testes unitários

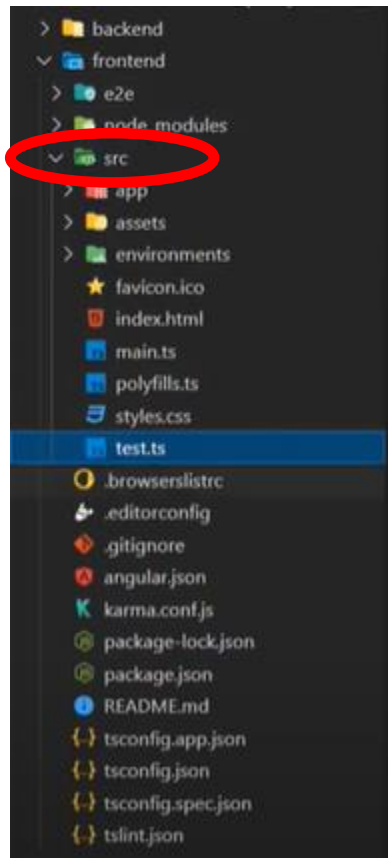
Onde ficam todos os pacotes e dependências que o nosso projeto precisa.

Configurações do editor e configurações do browser. Normalmente não modificamos eles.

Caso enviar o projeto para o github irá ignorar algumas pastas, dentre elas node_modules.

É um arquivo principal que contém todas as configurações do Angular. Nesse arquivo iremos trabalhar nele e fazer algumas alterações. Note que quando criamos o projeto e colocamos `--skip teste`, note que já está definido que não é para criar teste para nenhum dos componentes do angular ao pedir para gerar os componentes.

Vamos entender a estrutura dos projetos Angular



E a principal pasta que temos no nosso projeto que é a src. Dentro dela nós temos:

Test.ts -> arquivo de teste

Style.css -> Arquivo principal de estilo do nosso projeto.

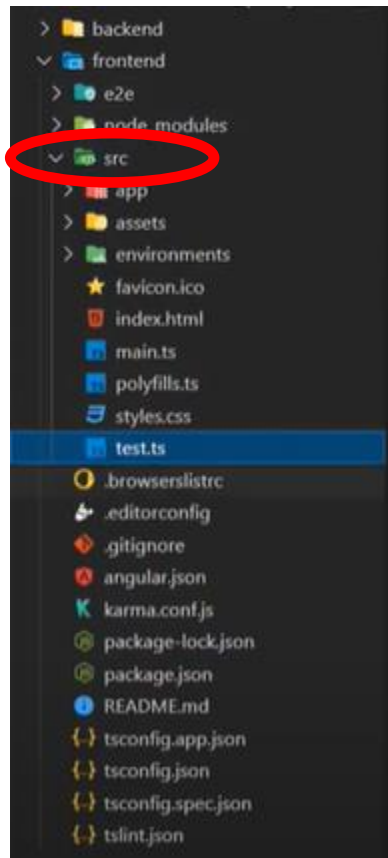
Polyfills.ts -> Arquivo responsável por fazer algumas configurações de compatibilidade para que o nosso projeto fique compatível com todos os navegadores.

Main.ts -> Arquivo muito importante: arquivo inicial, de start do nosso projeto. Nas aplicações java e C Sharp vocês terão dentro da sua classe principal o método main. O nosso arquivo main.ts se equivale a esse método main. E aqui nós teremos qual o módulo que será inicializado no nosso projeto por padrão, o primeiro módulo que irá subir e conter toda nossa aplicação. O Angular é baseado em módulos. Nós teremos inclusive as configurações Environments que são as configurações se nosso projeto está em produção ou não e se estiver em produção já irá fazer uma série de configurações automáticas para nós.

Index.html -> Esse arquivo é a única página html que nós criamos inicialmente que é a página que é carregada inicialmente. Lembrando que o Angular é um framework javascript SPA, é a única página completa que será carregada no nosso projeto. Todo o restante do código html, CSS, javascript que precisaremos carregar virá dentro do app-root. Depois iremos entender um pouco mais sobre o app-root.

```
<!doctype html>
<html lang="en">
<head>
  <meta charset="utf-8">
  <title>Frontend</title>
  <base href="/">
  <meta name="viewport" content="width=device-width, initial-scale=1">
  <link rel="icon" type="image/x-icon" href="favicon.ico">
</head>
<body>
  <app-root></app-root>
</body>
</html>
```

Vamos entender a estrutura dos projetos Angular



favicon.ico -> ícone

environments -> Aqui é onde fica nosso arquivo de environment. Todas as configurações de ambiente, tudo que irá envolver nossa aplicação. Temos a configuração para testes, para criação de projeto e produção. Com esses dois arquivos conseguimos separar sem precisar ficar trocando manualmente essas configurações depois de subir nosso projeto no servidor. Temos dois arquivos, por que? Para que não precisamos ficar modificando o arquivo de configuração geral do nosso projeto. Verifiquem que o arquivo environment.ts está como falso e teremos um arquivo de produção.

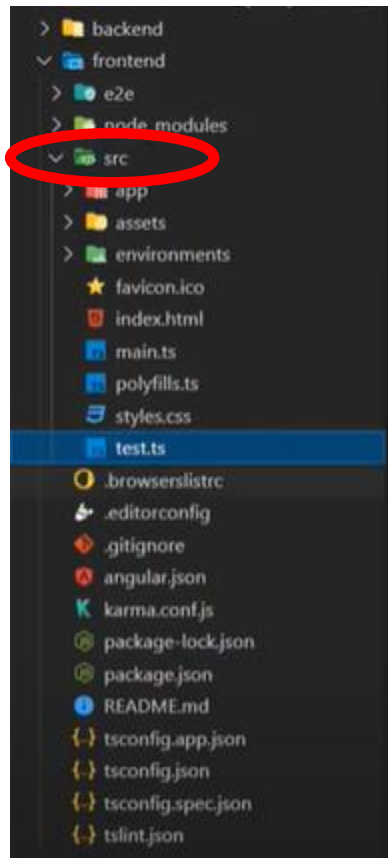
assets -> Conterá todas as nossas imagens que iremos disponibilizar na nossa aplicação e talvez alguns outros arquivos que precisaremos utilizar na nossa aplicação.

app -> pasta principal do nosso projeto. Contém o componente principal, o módulo principal do nosso projeto. Vamos ter:

app.module.ts -> possui as configurações iniciais do nosso projeto. Comentamos quando falamos sobre o main.ts. Módulo principal do nosso projeto, tudo é a partir dele. O que é importante ressaltarmos nesse arquivo. É o único que possui a propriedade bootstrap – é o que define por onde começa nossa aplicação, nesse caso é o AppComponent. Lembrando que o Angular foi feito para que você comece a trabalhar com componentes. O Angular JS não tinha essa componentização e agora com o Angular 2 ou Angular 12 que estamos utilizando tudo é componente e o componente inicial da nossa aplicação é o app.component.

```
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
7 @NgModule({
8   declarations: [
9     AppComponent
10  ],
11   imports: [
12     BrowserModule,
13     AppRoutingModule
14  ],
15   providers: [],
16   bootstrap: [AppComponent]
17 })
18 export class AppModule { }
```

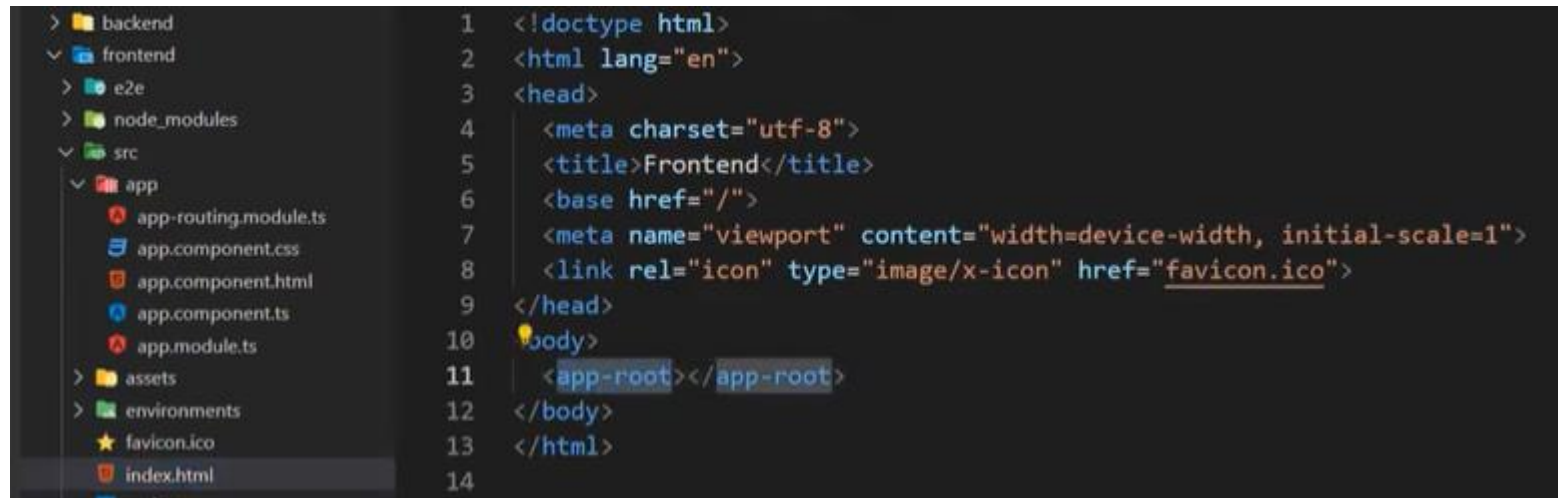
Vamos entender a estrutura dos projetos Angular



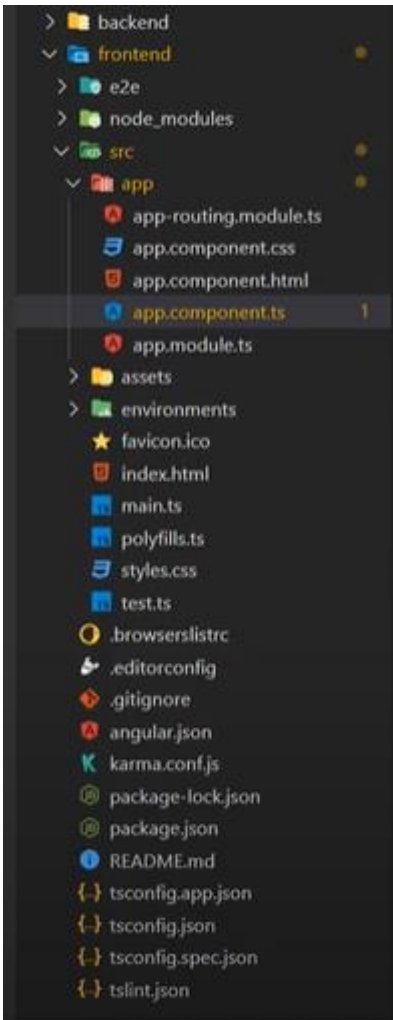
app.component.ts -> Aqui dentro tem um selector e chama: app-root. Aqui é o nome da tag html que iremos colocar quando estivermos desenvolvendo a aplicação, que fica no index.html. Toda vez que quisermos reindexar na nossa aplicação aquele componente que estaremos criando, depois iremos aprender o que são componentes. Por padrão cria como prefixo app, podendo modificar quando você está criando o seu projeto.

templateUrl -> Exibe qual o arquivo html que será o template, a parte estrutural do seu componente.

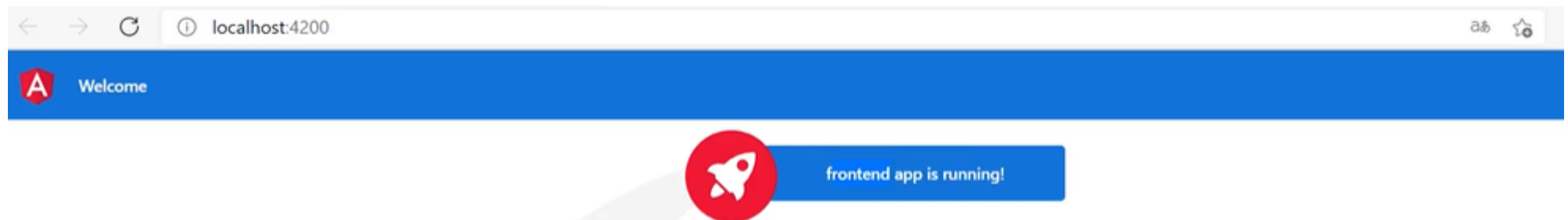
stylesUrls -> o arquivo css, sua folha de estilo que você estará utilizando para estilizar esse componente.



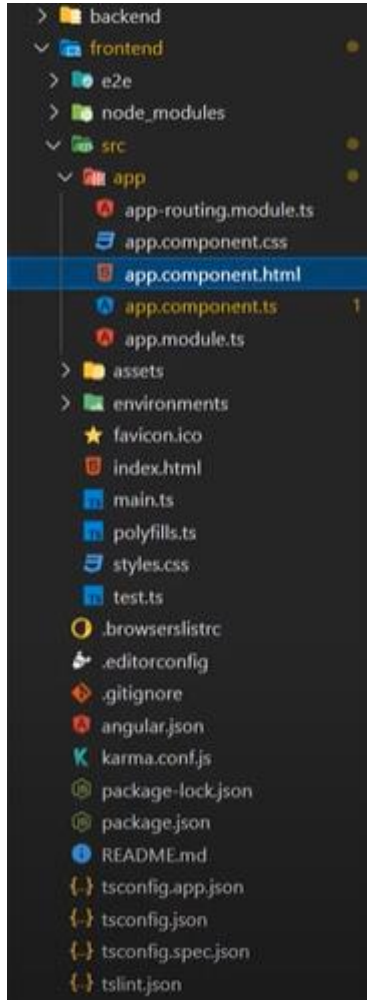
Vamos entender a estrutura dos projetos Angular



No caso, nós temos a nossa classe AppComponent com a variável title, que é a variável que está sendo utilizada para exibir na página principal o nome do título.
Vamos mudar o nome frontend para Curso ETEC, nossa página é renderizada e já aparece o novo título na página, como nome principal da nossa aplicação.



Vamos entender a estrutura dos projetos Angular



Temos o arquivo `app.component.html`. Esse arquivo html é bem grande, não se assustem mas não iremos trabalhar com componentes gigantes desse tamanho, inclusive a intenção de utilizar componentes aqui é que não tenhamos muito código dentro de um único arquivo, facilitando a nossa manutenção.

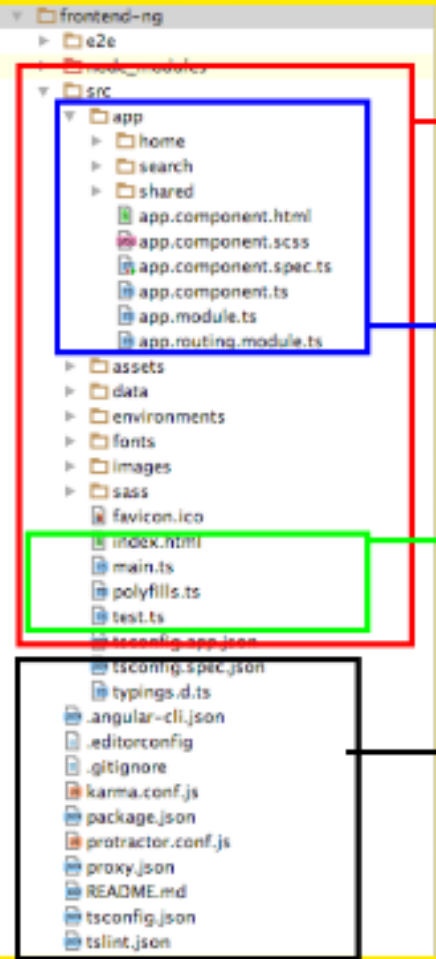
`App.component.css` -> é o arquivo onde iremos estilizar somente o componente aqui do `app.component`.

`App.routing.module` -> Quando criamos nosso projeto, definimos que iríamos utilizar as rotas e é aqui que iremos definir as rotas do nosso projeto.

E com isso teremos aqui todo o nosso projeto. Já conheceram toda a estrutura do nosso projeto frontend onde a pasta que mais iremos utilizar é a `src`

Resumo...

Angular App Structure



The diagram shows a file explorer view of an Angular application structure. It is divided into four main categories, each highlighted with a colored box and a label:

- Application Source** (Red box): This category encompasses the entire 'src' directory, which contains the application's source code.
- Angular App** (Blue box): This category highlights the 'app' subdirectory within 'src', which contains the application's modules, components, and routing.
- Angular Bootstrap** (Green box): This category highlights the files at the root of the 'src' directory: 'index.html', 'main.ts', 'polyfills.ts', and 'test.ts', which are responsible for bootstrapping the application.
- Configuration** (Black box): This category highlights the files at the root of the project, including 'tsconfig.spec.json', 'tsconfig.json', 'typings.d.ts', '.angular-cli.json', '.editorconfig', '.gitignore', 'karma.conf.js', 'package.json', 'protractor.conf.js', 'proxy.json', 'README.md', 'tsconfig.json', and 'tslint.json', which are used for configuring the development environment.

Our application is divided in two parts. -Application itself & Application bootstrap

Our modules, routes, components & services live here.

We bootstrap our app depending on what platform were developing for, and add any required polyfills & vendors.

Typescript config, npm packages, scripts, webpack config, express server, and so on...

Conclusão

- E com isso teremos aqui todo o nosso projeto.
- Já conheceram toda a estrutura do nosso projeto frontend onde a pasta que mais iremos utilizar é a src.

