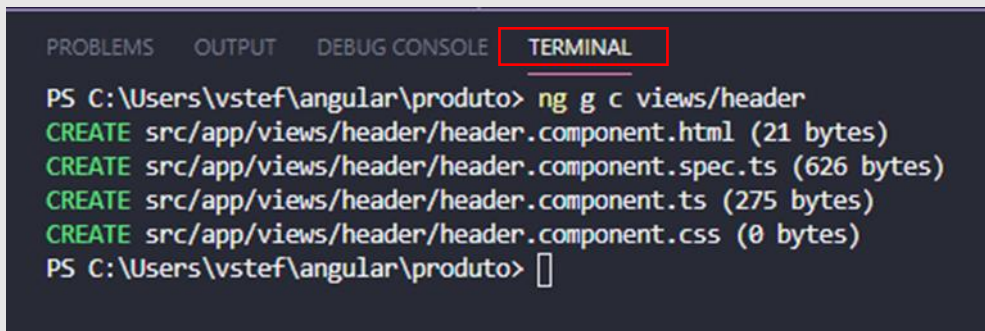


CRIANDO COMPONENTES

Menu (mat-toolbar) e links

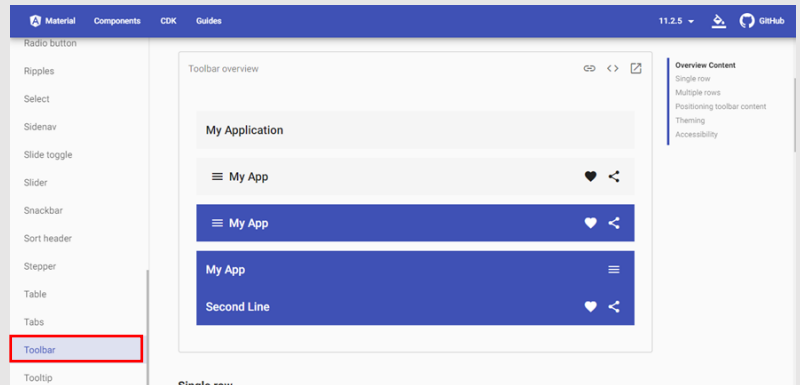
1. Para criar um componente de menu, vamos separar por pastas. Para isso, utilize o comando no **terminal do Visual Studio Code** (**ctrl + shift + `**) ou clique em “**Terminal**” e, depois, em “**New Terminal**”.



```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL
PS C:\Users\vstef\angular\produto> ng g c views/header
CREATE src/app/views/header/header.component.html (21 bytes)
CREATE src/app/views/header/header.component.spec.ts (626 bytes)
CREATE src/app/views/header/header.component.ts (275 bytes)
CREATE src/app/views/header/header.component.css (0 bytes)
PS C:\Users\vstef\angular\produto> █
```

2. Para adicionar um conteúdo visual ao Angular, acesse o site do Material – aquele instalado no seu projeto com (@angular/material), na configuração de ambiente. Será gerado um link (<https://material.angular.io/?theme=indigo-pink>) com o tema escolhido.

3. Acesse o site que foi disponibilizado, clique em “**Components**” e, depois, em “**Toolbar**”.

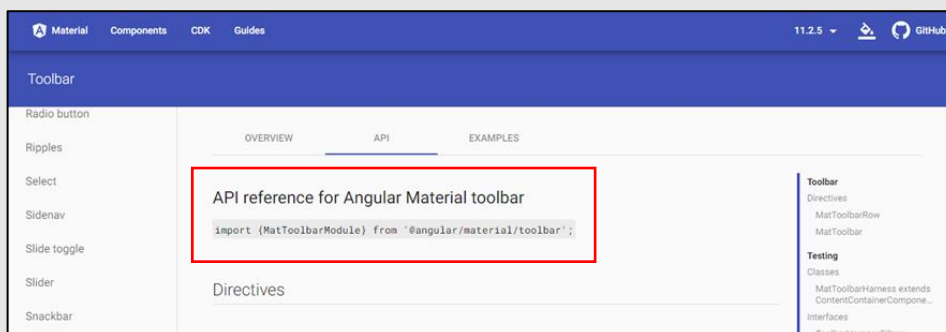


Fonte:

<https://material.angular.io/components/toolbar/overview>.

4. Atente-se ao importar qualquer elemento que irá compor seu HTML no Angular, pois você não pode apenas copiar a estrutura do código pronto e adicioná-la ao seu arquivo de extensão **.html**; caso faça isso, seu projeto apresentará **erro**.

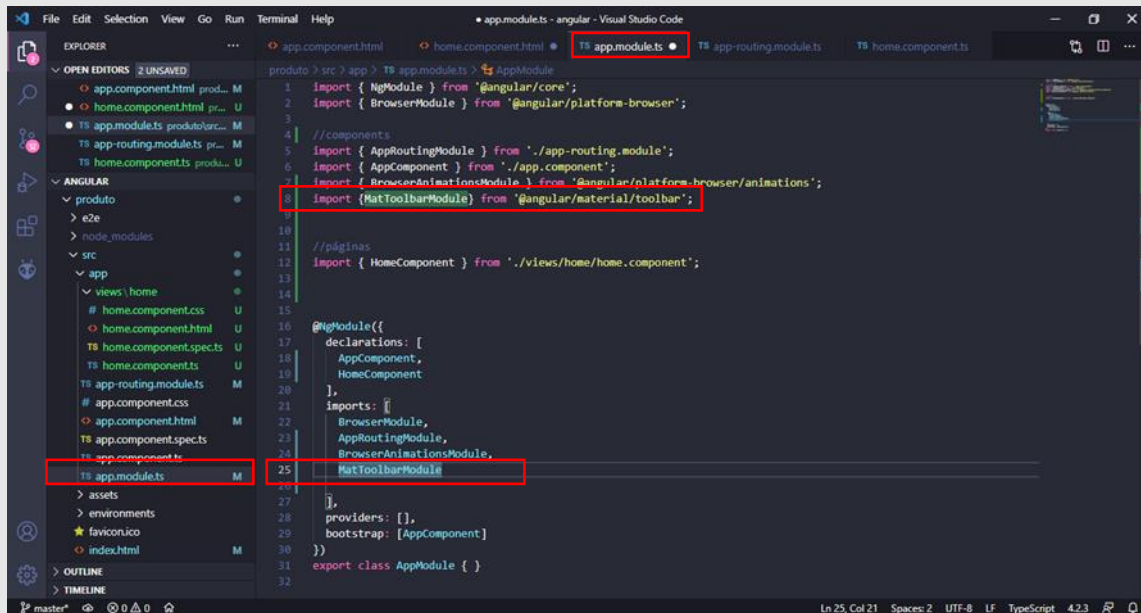
Para que seu projeto reconheça o componente, apresentando-o ao usuário no **browser**, e que não contenha erro, é necessário importar a API, que informará o projeto do componente a ser utilizado. Desse modo, seu projeto reconhecerá o componente como parte integrante dele e você poderá usar o HTML com a estrutura pronta, sem erros. Para isso, clique em **API** e copie a linha de código, conforme mostra a imagem abaixo:



Fonte:

<https://material.angular.io/components/toolbar/overview>.

5. Para fazer a importação, acesse seu projeto, abra o arquivo **app.module.ts** e adicione o **import**, que você copiou do site do Material, à **linha 8 do import da API**, com os demais componentes. Note que dentro das chaves há um nome. É necessário copiar esse nome e colá-lo dentro da tag **<imports>**, conforme consta na **linha 25** da imagem abaixo. Feito isso, salve (**ctrl + s**) as informações. Agora, adicione o HTML ao seu projeto.



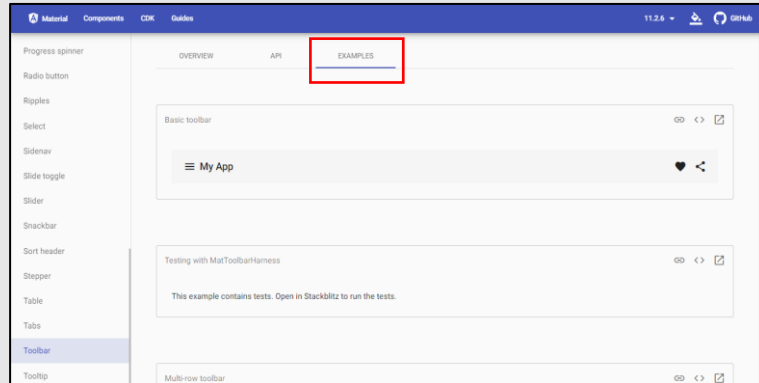
```
1 import { NgModule } from '@angular/core';
2 import { BrowserModule } from '@angular/platform-browser';
3
4 //components
5 import { AppRoutingModule } from './app-routing.module';
6 import { AppComponent } from './app.component';
7 import { BrowserAnimationsModule } from '@angular/platform-browser/animations';
8 import { MatButtonModule } from '@angular/material/toolbar';
9
10
11 //páginas
12 import { HomeComponent } from './views/home/home.component';
13
14
15
16 @NgModule({
17   declarations: [
18     AppComponent,
19     HomeComponent
20   ],
21   imports: [
22     BrowserModule,
23     AppRoutingModule,
24     BrowserAnimationsModule,
25     MatButtonModule
26   ],
27   providers: [],
28   bootstrap: [AppComponent]
29 })
30 export class AppModule { }
```

Importante

Será necessário repetir esse procedimento para todos os componentes do Angular Material.

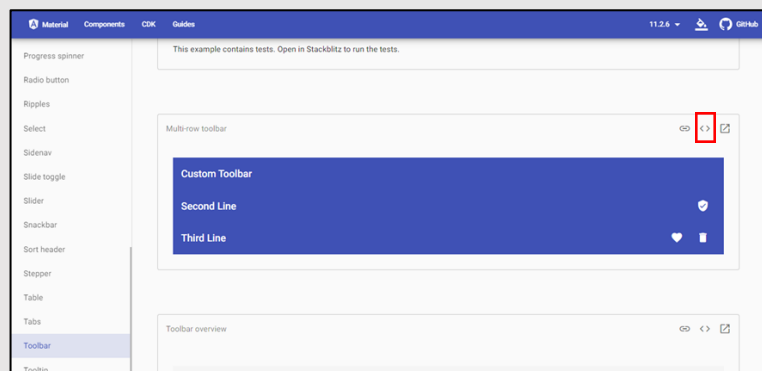


6. Agora que você fez o primeiro passo da importação da API, acesse novamente o site do Angular Material. Clique na aba “**Examples**”, nessa seção são disponibilizadas algumas opções de Front-End com a estrutura já pronta.



Fonte: <https://material.angular.io/components/toolbar/examples>.

7. Para adicionar um desses elementos do *toolbar* (menu) pronto a seu site, você deve clicar no segundo ícone da direita (destacado em vermelho na imagem), que irá disponibilizar o HTML pronto, com as tags do Angular. Vamos utilizar o exemplo de tela **multi-row toolbar**. Ao clicar no ícone, um *accordéon* será aberto, e o código será disponibilizado.



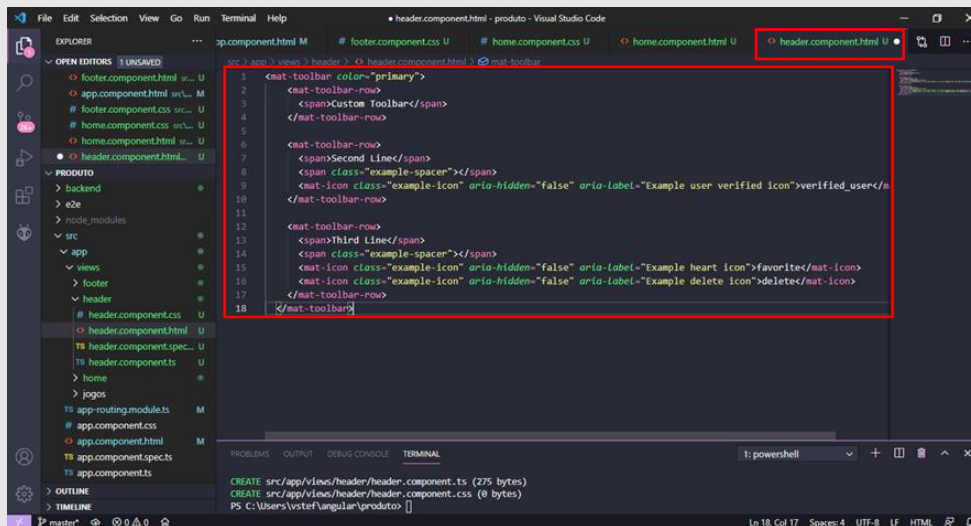
Fonte: <https://material.angular.io/components/toolbar/examples>.

8. Ao clicar no ícone, veja que serão apresentados o HTML, o TS e o CSS:



Fonte: <https://material.angular.io/components/toolbar/examples>.

9. Copie o conteúdo do HTML e cole-o no seu arquivo **header.components.html**.



Dica!



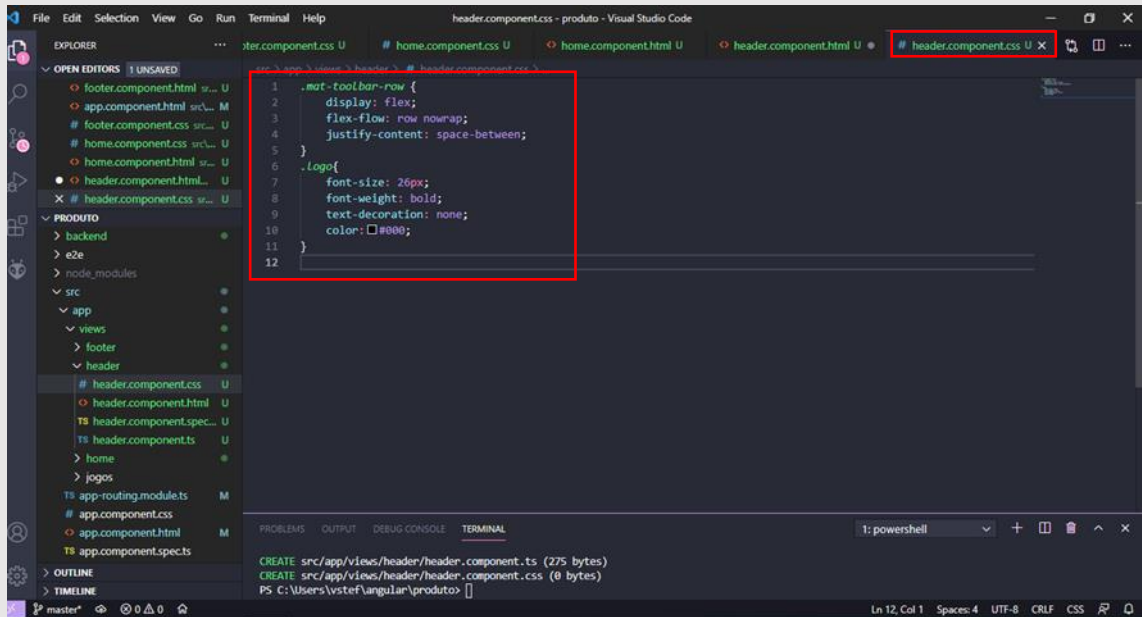
Caso você deseje que o **CSS** seja o mesmo do site do Material, copie o CSS e adicione-o ao seu arquivo **header.components.css**. A aba TS do site do Material não possui conteúdo para adicionar.

10. Você pode utilizar a estrutura das tags **<mat-toolbar>**, conforme adicionado ao nosso arquivo, ou alterá-las para adicionar o conteúdo do site que está sendo desenvolvido. Para isso, você deve alterar os **links, os ícones e o logo (*custom toolbar*)**.

A seguir, analise as duas sugestões, a primeira em HTML e a segunda em CSS.

```
1 <mat-toolbar>
2   <mat-toolbar-row>
3     <a class="logo" routerLink="/">
4       
5     </a>
6     <div class="menu-links">
7       <a mat-button href="#" target="_blank">logos</a>
8       <a mat-button href="#" target="_blank">Portfólio</a>
9       <a mat-button href="#" target="_blank">Contato</a>
10    </div>
11  </mat-toolbar-row>
12 </mat-toolbar>
```

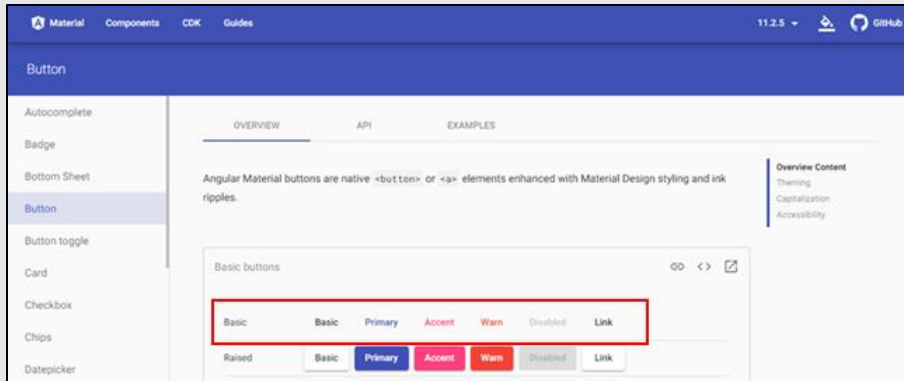
HTML.



CSS.

Fizemos as alterações no conteúdo do código disponibilizado no site do Material, conforme imagens dos itens 8 e 9; porém ***span* não é uma tag de âncora**. Para isso, podemos **adicionar um *button* para substituir o *span***. Acompanhe o processo no item 10.

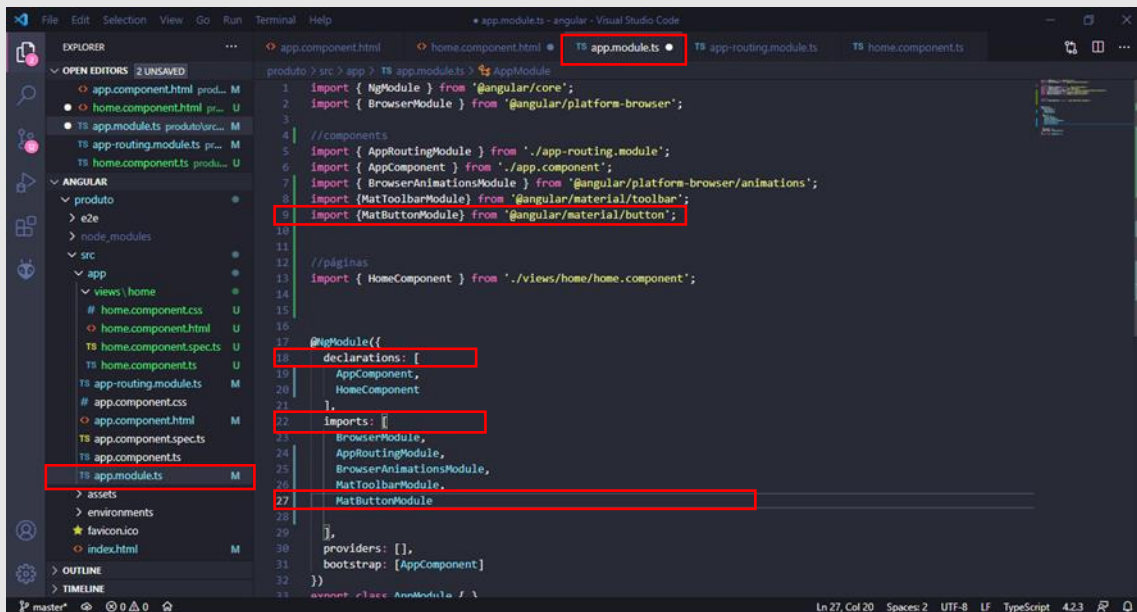
11. Como exemplo, utilize o modelo **basic** da primeira linha.



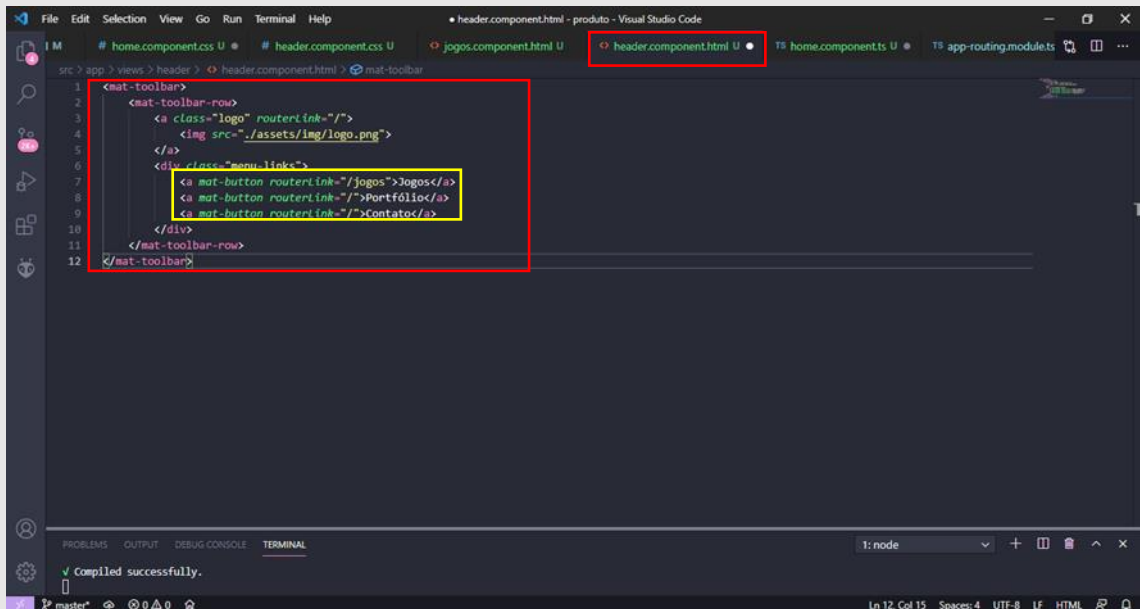
Fonte: <https://material.angular.io/components/button/overview>.

12. Clique em **API** para importar. Copie o **import** e adicione-o ao **app.module.ts** (como feito para o *toolbar*.) Depois, copie o nome entre chaves **{MatButtonModule}** e cole-o em **imports**, na linha seguinte ao **MatToolbarModule**.

O **import** dos arquivos que irão compor seu site (página HTML, por exemplo) será feito em **declarations**. Componentes que irão estruturar seu site (*toolbar*, por exemplo) serão adicionados a **imports**.



13. Após copiar o HTML da opção **basic** (dentro do site do Material), disponibilizado no site do Angular Material, colá-lo no **header.component.html** e fazer a alteração, temos:



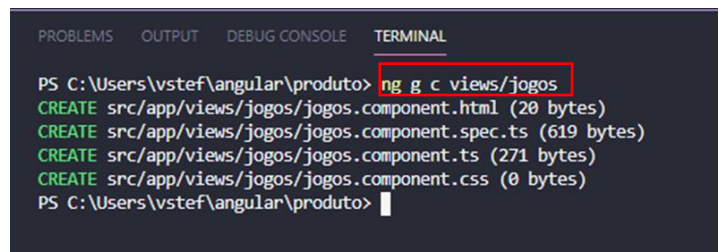
```
1 <mat-toolbar>
2   <mat-toolbar-row>
3     <a class="logo" routerLink="/">
4       
5     </a>
6     <div class="menu-links">
7       <a mat-button routerLink="/jogos">Jogos</a>
8       <a mat-button routerLink="/">Portfólio</a>
9       <a mat-button routerLink="/">Contatos</a>
10    </div>
11  </mat-toolbar-row>
12 </mat-toolbar>
```

Importante

Observe, no item anterior, que adicionamos um novo atributo, chamado **routerLink**. O **routerLink** serve para indicar o atributo do **href**, da âncora e do HTML. No entanto, com ele, não passamos o valor diretamente, por exemplo: **routerLink="jogos.html"**. Para isso, será necessário configurar o arquivo **app-routing.module.ts**.

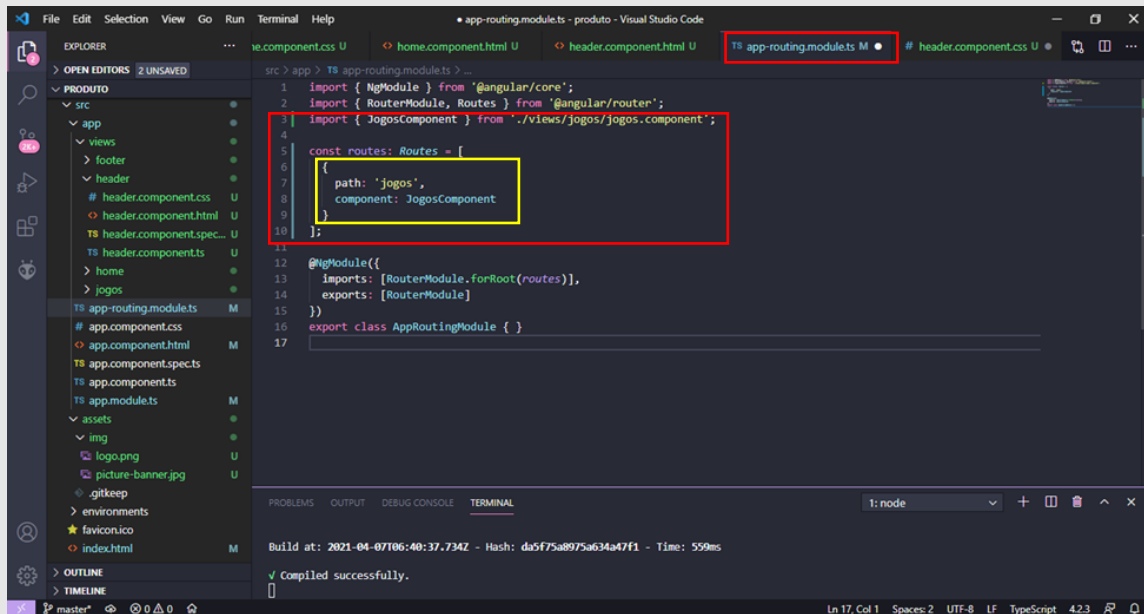


A título de exemplo, a imagem ao lado mostra um arquivo **jogos** que foi criado no Terminal. Para criá-lo, digite: **ng g c views/jogos**.



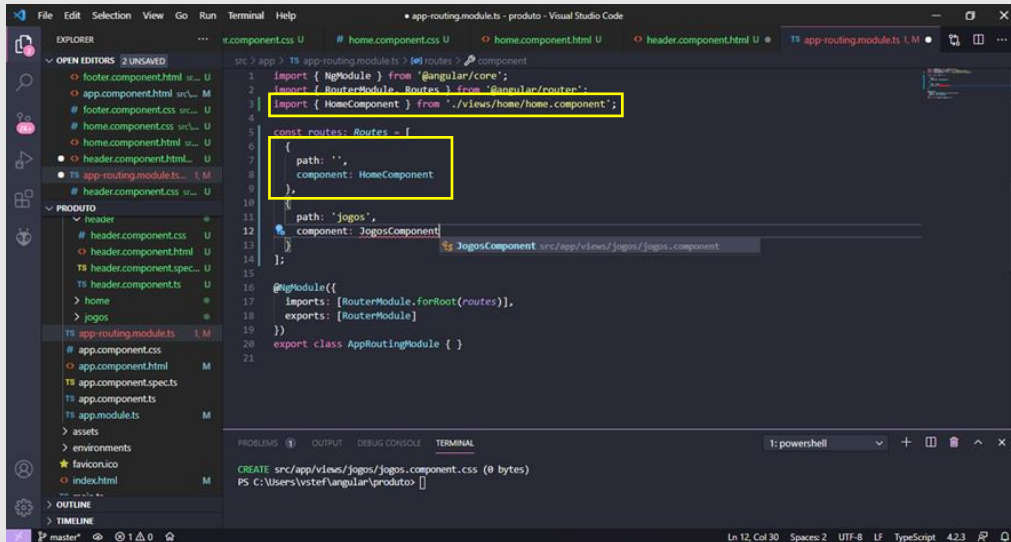
```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL
PS C:\Users\vstef\angular\produto> ng g c views/jogos
CREATE src/app/views/jogos/jogos.component.html (20 bytes)
CREATE src/app/views/jogos/jogos.component.spec.ts (619 bytes)
CREATE src/app/views/jogos/jogos.component.ts (271 bytes)
CREATE src/app/views/jogos/jogos.component.css (0 bytes)
PS C:\Users\vstef\angular\produto>
```

14. Agora, no arquivo **app-routing.module.ts**, faça conforme destacado imagem:



- Observe que o conteúdo foi adicionado a **const routes** (linha 5). Os conteúdos de âncoras devem ser adicionados a essa variável.
- Path** é o caminho do arquivo que adicionaremos. Na **linha 7**, constará o arquivo **jogos**. Na **linha 8**, passamos o caminho **JogosComponent**. Lembre-se de fazer isso para os futuros arquivos que você adicionar. Observe que estamos importando o arquivo (linha 3); como atalho para importar, você pode **selecionar a palavra e segurar Ctrl + espaço** após adicionar o nome **“JogosComponent”**.
- As **rotas de navegabilidade** configuradas nesse documento servem para todo o projeto, o que significa que você pode utilizá-las em todos os seus arquivos.
- Em seu arquivo com a **extensão .html**, adicione o atributo **“routerLink=”** e o nome definido em **path**.

15. Como vimos anteriormente, em `path` você define o nome (que aparecerá em URL) e em componentes passa o caminho do arquivo.



```
1 import { NgModule } from '@angular/core';
2 import { RouterModule, Routes } from '@angular/router';
3 import { HomeComponent } from '../views/home/home.component';
4
5 const routes: Routes = [
6   {
7     path: '',
8     component: HomeComponent
9   },
10   {
11     path: 'jogos',
12     component: JogosComponent
13   }
14 ];
15
16 @NgModule({
17   imports: [RouterModule.forRoot(routes)],
18   exports: [RouterModule]
19 })
20 export class AppRoutingModule { }
21
```

- a) Perceba, na **linha 7**, que estamos passando **aspas vazias** em **path** e o **arquivo HomeComponent** em **components**, resultado da importação (**linha 3**). Isso significa que toda vez que a página for carregada, ela será direcionada para o **arquivo default**, ou seja, **HomeComponents**, que é seu **index.html**.
- b) Note que fizemos uma alteração e, agora, teremos que importar o **JogosComponents**.
- c) Veja que o próprio **IDE** sugere o **arquivo de importação**. Basta selecionar, que ele fará a importação automaticamente.

Dica!

Para importar automaticamente, você pode digitar o **nome do component** e, no final da palavra, segurar **ctrl + espaço**. A importação será sugerida e, ao clicar na sugestão, ela será feita automaticamente, corrigindo o erro.



16. Salve (**ctrl + s**) e, no Terminal, digite novamente: **“ng serve --port 333 -o”** para visualizar o resultado:

