**Introdução ao Angular 8**

Angular é uma plataforma e framework para construção da interface de aplicações usando HTML, CSS e, principalmente, JavaScript, criada pelos desenvolvedores da Google.

Angular nos ajuda a criar Single-Page Applications: Um aplicativo de página única que pode ser uma aplicação web ou site que consiste de uma única página web com o objetivo de fornecer uma experiência do usuário similar a de um aplicativo desktop.

O Angular atua no front-end e faz a ligação e tráfego dos dados com o back-end sempre utilizando JSON em arquitetura REST.

Requisitos: JavaScript, CSS, HTML, banco de dados e alguma linguagem de back-end como Java, PHP, etc.

O Angular foi desenvolvido em TypeScript que é uma evolução do JavaScript e com todos os conceitos da orientação a objetos, essa linguagem é a mais usada para aplicações Angular e a que mais encontramos documentações e exemplos na internet.

O dinheiro está em resolver o problema e não na tecnologia.

Só se ganha dinheiro se sabe resolver o problema, não importa com qual tecnologia.

O NPM serve para fazer a instalação de recursos dentro do projeto desenvolvido com Angular.

Podemos instalar o Angular com todas as suas dependências usando o gerenciador NPM.

Exemplo: npm install @angular/cli@8.3.19

Para a criação de um projeto em Angular usamos a tag ng, veja o exemplo abaixo:

ng new projeto-com-npm --skip-git

A flag --skip-git serve para ignorar o uso do git na criação do projeto.

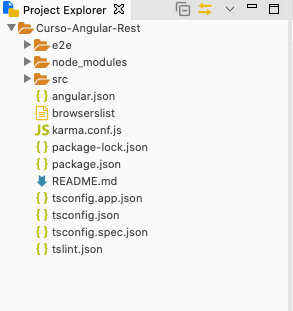
Rota é algo essencial para o funcionamento da aplicação pois serve para navegar entre as páginas do projeto.

Podemos criar projetos em Angular através de recurso visual ou através de linha de comando.

Por fim temos que salvar o projeto, para isso usamos o seguinte comando: npm install --save-dev angular-ide

Por fim temos que importar o projeto para dentro da IDE, para isso basta somente importar a pasta projeto-com-npm que foi criada no momento da criação do projeto com o Angular.

**Entendendo os principais arquivos da estrutura do projeto**



A primeira pasta da estrutura dentro do projeto chamada e2e serve para testes, ou seja, é o diretório para testes.

A segunda pasta da estrutura chamada node\_modules é a pasta do NodeJs onde é salvo todas as bibliotecas necessárias que o Angular necessita dentro do projeto.

A terceira pasta chamada src é a pasta que contém o código fonte que iremos trabalhar, src é a abreviação de source que em inglês significa código.

O arquivo angular.json contém as configurações do Angular para o projeto.

**Arquivos e pastas contidos dentro da pasta src**

A pasta mais importante aonde desenvolvemos o projeto se chama app e fica dentro da página src.

A pasta assets contém os arquivos estáticos também chamados de recursos como imagens (fotografias), vídeos ou qualquer outra mídia que for usada dentro do projeto.

Dentro da pasta environments ficam os arquivos de configuração de ambiente que pode ser um ambiente de produção ou um ambiente de teste.

No arquivo index.html fica o ponto de partida do sistema, normalmente é a página inicial do sistema, faz referência ao elemento principal app-root.

Dentro da pasta src temos o arquivo main.ts que é o ponto de entrada para inicialização de módulos.

Um componente dentro do Angular é a união de arquivos HTML, CSS e JavaScript, uma tela de cadastro de clientes dentro do Angular é um componente.

Um componente é dividido em várias partes, temos que entender como que funciona a estrutura de um componente em Angular.

**Entendendo a estrutura de um componente**

Dentro da pasta app em src ficam os cinco arquivos responsáveis pela criação de um componente, são eles:

* app.component.css
* app.component.html
* app.component.spec.ts
* app.component.ts
* app.module.ts

O Arquivo app.component.css contém toda a parte de CSS do componente, responsável pela aparência da página.

O arquivo app.component.html contém toda a parte de HTML.

O nome muda aonde está escrito app por exemplo se for um cadastro de cliente fica cadastroCliente.component.css

O arquivo app.component.spec.ts serve para realizar teste unitário dentro do componente.

No arquivo app.component.ts é aonde existe a montagem do componente, ele é o componente propriamente dito, é responsável por chamar os arquivos que fazem parte do componente.

O arquivo app.module.ts serve para dar o start no componente, aonde fica a declaração do componente depois de criado para o Angular inicia-lo.

Todos os arquivos se encaixam.

O arquivo angular.json contém a definição do arquivo inicial, e as demais configurações do projeto, é o arquivo principal responsável pela configuração de todo o projeto.

O app-root é responsável por injetar todos os dados dentro da página, é uma tag html que quando chamada faz a inserção do componente.

O app-root é responsável pela saída de dados do componente, definimos tudo o que vai no componente em seu arquivo de montagem.

É o app-root que interage com a página html.

**Instalando BootStrap e Layout de Login**

npm install bootstrap@4.1.3 jquery@3.3.1 popper.js@1.14.3 --save

O BootStrap depende do jQuery e do Popper para poder funcionar no Angular.

Depois de baixado o BootStrap através do NPM temos que referenciar no arquivo angular.json em styles, veja abaixo:

"styles": [

*"node\_modules/bootstrap/dist/css/bootstrap.css"*,

*"src/styles.css"*

],

"scripts": [

*"node\_modules/jquery/dist/jquery.js"*,

*"node\_modules/popper.js/dist/umd/popper.js"*,

*"node\_modules/bootstrap/dist/js/bootstrap.js"*

]

Fazemos a referencia dos arquivos javascript do jQuey, Popper e BootStrap em scripts no arquivo angular.json.

O submit serve para submeter para o lado do servidor, é utilizado em aplicações MVC, em aplicações REST usamos button e fazemos as validações com javascript.

Para capturar dados de um formulário é necessário importar FormsModule e declará-lo em imports no arquivo de declaração de módulos que possuí a extensão .module.ts.

**import** { FormsModule } **from** '@angular/forms';

**import** { NgModule} **from** '@angular/core';

imports: [

BrowserModule,

FormsModule

],

Capturamos os dados dos inputs dos formulários através da classe ngModel, veja abaixo um exemplo:

<input type=*"email"* class=*"form-control"* [(ngModel)] = user.login id=*"user"* name=*"user"* placeholder=*"Informe o username"* required>

Para executarmos algo com a ação de clique fazemos da seguinte forma:

<button type=*"button"* class=*"btn btn-primary"* (click) = *"login()"* >Login</button>

O método login que chamamos acima, está declarado no código abaixo dentro do arquivo app.component.ts.

**export** **class** AppComponent {

title = 'Curso-Angular-Rest';

user = {login: '', password: ''}; // Declaração de objetos.

**public** **login**(){

console.log("Usuário: " + **this**.user.login);

console.log("Senha: " + **this**.user.password);

}

}

x