**Arquivo package.json**

**Comandos que vc pode executar com o próprio angular:**

    "ng": "ng",

    "start": "ng serve",

    "build": "ng build",

    "test": "ng test",

    "lint": "ng lint",

    "e2e": "ng e2e"

**Rodar o projeto a primeira vez: no Terminal na pasta do projeto comando: ng serve --open**

Usamos este comando por que já temos a pasta no esqueleto do projeto que é a node\_modules – que tem todas as bibliotecas que ele baixa com o package json

Se o projeto não tivesse essa pasta primeiro seria necessário o comando: **npm i** pra ele baixar as dependências necessárias

Rodar o projeto normalmente: **ng serve**

Fazer build: **ng build**

**Entendendo a estrutura de pastas do projeto:**

**package.json:** gerenciar as dependências do projeto (similar ao pom.xml do maven)

**package-lock.json:** grava o último npm i que você fez (Não é obrigatório) não é legal ser versionado, então colocar no gitignore ou apagar

**tsconfig.json:** serve pra dizer que o TypeScript que vamos escrever não é interpretado pelo navegador e sim convertido para JavaScript e esse arquivo convertido sim, é o que vai para o navegador, este arquivo que vai para o navegador é criado ao fazer o build do projeto dentro da pasta dist que é criado na hora do build também.

**tslint.json:** Pra ele achar alguns erro importantes no seu código, alguma coisa duplicada, alguma coisa do angular ou variável que possa dar erro

**pasta src:** Onde vc vai trabalhar de fato, tem os seguintes itens:

index.html: página inicial padrão, a princípio não mexemos nela

pasta app:

app.module.ts: serve para carregar todos os módulos do Angular, devido ao formato do Angular de single page aplication – que é gerenciar o sistema inteiro com apenas uma página, carregando os componentes dele, que são classes que você escreve em TypeScript e que você pode criar vários componentes para renderizar no html

app.component.html: html do componente inicial

app.component.spec.ts: teste do componente pra saber se ele existe

app.component.css: css do componente

app.component.ts: componente em si

**Criar componente:** ng generate componente **nomeDoComponente**

Ao rodar esse comando ele cria dentro da pasta app, uma pasta com o nome do componente e os 4 arquivos padrão

Property Binding – propriedade do componente que passa para o html

Event Binding – Você passa uma chamada do html para o seu componente para ele ter algum comportamento

**Imagens:** Para usar imagens é necessário inserir na pasta assets

**Criar módulo de rotas:** **ng generate module app-routing --flat --module=app**

No comando a cima nós estamos gerando um novo módulo chamado AppModuleRouting e passando os parâmetros: -flat para que ele seja criado dentro da pasta src/app e o -module=app para registrarmos ele no nosso arquivo AppModule.

Após criar o módulo de rotas deixar arquivo conforme abaixo:

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

import { CommonModule } from '@angular/common';

import { Routes } from '@angular/router';

import { HomeComponent } from './home/home.component';

import { UsuarioComponent } from './usuario/usuario.

\*\*IMPORT DO COMPONENTE

const routes: Routes = [

  { path: **'nomePathComponent'**, component: **NomeDoComponente** },

{ path: 'home', component: HomeComponent },

  { path: 'usuario', component: UsuarioComponent },

  { path: '', pathMatch:'full', redirectTo:'/home' },

];

@NgModule({

  declarations: [],

  imports: [

    RouterModule.forRoot(routes) \*\*Import das suas rotas

  ],

exports: [RouterModule]

})

export class AppRoutingModule { }

detalhes: <https://angular.io/tutorial/toh-pt5>