SPRING BOOT + ANGULAR +STRIPE

# PAGOS CON STRIPE EN SPRING BOOT Y ANGULAR 8. CAPÍTULO 5: CREACIÓN Y CONFIGURACIÓN DEL FRONT END

Y llega la hora de crear el proyecto Angular. En la consola tecleamos:

ng new tutorial-stripeFRONT

Nos pregunta si queremos añadir Angular routing (esto es a partir de la versión 7), le decimos que sí Tecla “y”, enter)  
También pregunta por el tipo de formato de tabla de estilos queremos, seleccionamos CSS, que es la opción por defecto.  
Esperamos a que se genere el proyecto…esto puede llevar varios meses o años.

Una vez creado nos situamos en el directorio

cd tutorial-stripeFRONT

Y lo abrimos con el editor (en mi caso Visual Studio Code).

code .

Una vez abierto ctrl+ñ para mostrar un nuevo terminal. Vamos a instalar ng-bootstrap:

npm install --save @ng-bootstrap/ng-bootstrap

En el archivo app.module.ts tenemos que hacer las importaciones. Si vamos a la [documentación](https://ng-bootstrap.github.io/#/getting-started)de ng-bootstrap vemos que se puede importar el módulo principal o bien los componentes específicos que vayamos a utilizar. Como sólo vamos a necesitar el modal seguimos el segundo método:

import { NgbModalModule} from '@ng-bootstrap/ng-bootstrap';

Y no os olvidéis de poner NgbModalModule y en los imports.

Aprovechando que estamos en app.module.ts vamos también a importar el httpclientmodule y los formularios:

import { HttpClientModule } from '@angular/common/http';

import { FormsModule, ReactiveFormsModule } from '@angular/forms';

Y por supuesto, también ponerlos en los imports.

El siguiente paso es instalar ngx-stripe:

npm install ngx-stripe

Lo importamos en app.module.ts:

import { NgxStripeModule } from 'ngx-stripe';

Y lo añadimos a los imports. En este caso, hay que poner la clave pública de nuestra cuenta de Stripe:

NgxStripeModule.forRoot('mi\_clave'),

Nos queda instalar ngx-toastr para mostrar las notificaciones, en la [documentación](https://www.npmjs.com/package/ngx-toastr/v/8.8.0)también podéis ver el proceso de instalación Tecleamos:

npm install ngx-toastr --save

Instalamos las animaciones:

npm install @angular/animations --save

Modificamos el archivo angular.json añadiendo una línea en styles:

"styles": [

"node\_modules/ngx-toastr/toastr.css",

"src/styles.css"

],

Y por último, importamos toastr y animations en el app.module.ts:

import { BrowserAnimationsModule } from '@angular/platform-browser/animations';

import { ToastrModule } from 'ngx-toastr';

El app.module.ts quedará así (de momento):

|  |  |
| --- | --- |
|  | import { BrowserModule } from '@angular/platform-browser'; |
|  | import { NgModule } from '@angular/core'; |
|  |  |
|  | import { AppRoutingModule } from './app-routing.module'; |
|  | import { AppComponent } from './app.component'; |
|  |  |
|  | // ng-bootstrap, toastr, httpclient, forms y ngx-stripe |
|  | import { NgbModalModule} from '@ng-bootstrap/ng-bootstrap'; |
|  | import { HttpClientModule } from '@angular/common/http'; |
|  | import { FormsModule, ReactiveFormsModule } from '@angular/forms'; |
|  | import { NgxStripeModule } from 'ngx-stripe'; |
|  | import { BrowserAnimationsModule } from '@angular/platform-browser/animations'; |
|  | import { ToastrModule } from 'ngx-toastr'; |
|  |  |
|  | @NgModule({ |
|  | declarations: [ |
|  | AppComponent |
|  | ], |
|  | imports: [ |
|  | BrowserModule, |
|  | NgbModalModule, |
|  | HttpClientModule, |
|  | FormsModule, |
|  | ReactiveFormsModule, |
|  | ToastrModule.forRoot(), |
|  | BrowserAnimationsModule, |
|  | NgxStripeModule.forRoot('mi\_clave\_publica'), |
|  | AppRoutingModule |
|  | ], |
|  | providers: [], |
|  | bootstrap: [AppComponent] |
|  | }) |
|  | export class AppModule { } |

[**view raw**](https://gist.github.com/cavanosa/9e8571306ee9248a34435550a8117127/raw/01f03a5adbd4ac1798824e32805233b2a6fb5cc0/app.module.ts)[**app.module.ts**](https://gist.github.com/cavanosa/9e8571306ee9248a34435550a8117127#file-app-module-ts) hosted with  by [**GitHub**](https://github.com/)

Por último, en index.html vamos a añadir el cdn de bootstrap, justo encima del cierre de la etiqueta head:

|  |  |
| --- | --- |
|  | <!doctype html> |
|  | <html lang="es"> |
|  | <head> |
|  | <meta charset="utf-8"> |
|  | <title>TutorialStripeFRONT</title> |
|  | <base href="/"> |
|  |  |
|  | <meta name="viewport" content="width=device-width, initial-scale=1"> |
|  | <link rel="icon" type="image/x-icon" href="favicon.ico"> |
|  | <!-- CDN de Bootstrap --> |
|  | <link rel="stylesheet" href="https://stackpath.bootstrapcdn.com/bootstrap/4.3.1/css/bootstrap.min.css" integrity="sha384-ggOyR0iXCbMQv3Xipma34MD+dH/1fQ784/j6cY/iJTQUOhcWr7x9JvoRxT2MZw1T" crossorigin="anonymous"> |
|  | </head> |
|  | <body> |
|  | <app-root></app-root> |
|  | </body> |
|  | </html> |

[**view raw**](https://gist.github.com/cavanosa/e539fd954c645df753f1f4cec2e1c77e/raw/872d19edcc0baf3bf505f6a84933c7c96ce4f6c8/index.html)[**index.html**](https://gist.github.com/cavanosa/e539fd954c645df753f1f4cec2e1c77e#file-index-html) hosted with  by [**GitHub**](https://github.com/)

**Archivos y carpetas:**

Vamos ahora a crear componentes y servicios. Comenzamos con el menú:

ng g c menu --skipTests

El modal:

ng g c modal --skipTests

El payment:

ng g c payment --skipTests

El servicio de payment:

ng g s payment/payment --skipTests

Y los componentes de lista de artículos y el detalle (irán los dos en la misma carpeta):

ng g c articulo/lista-articulo --skipTests --flat

ng g c articulo/detalle-articulo --skipTests --flat

En esta carpeta también necesitamos un servicio:

ng g s articulo/articulo --skipTests

Luego crearemos la interfaz articulo en la carpeta model:

ng g interface model/articulo

|  |  |
| --- | --- |
|  | export interface Articulo { |
|  | id?: number; |
|  | nombre: string; |
|  | descripcion: string; |
|  | precio: number; |
|  | imagenURL: string; |
|  | } |

[**view raw**](https://gist.github.com/cavanosa/4e759ccf57f86791bb1da22f6d9f3a0c/raw/3969e73af6986227f512b4310df8b12118398911/articulo.ts)[**articulo.ts**](https://gist.github.com/cavanosa/4e759ccf57f86791bb1da22f6d9f3a0c#file-articulo-ts) hosted with  by [**GitHub**](https://github.com/)

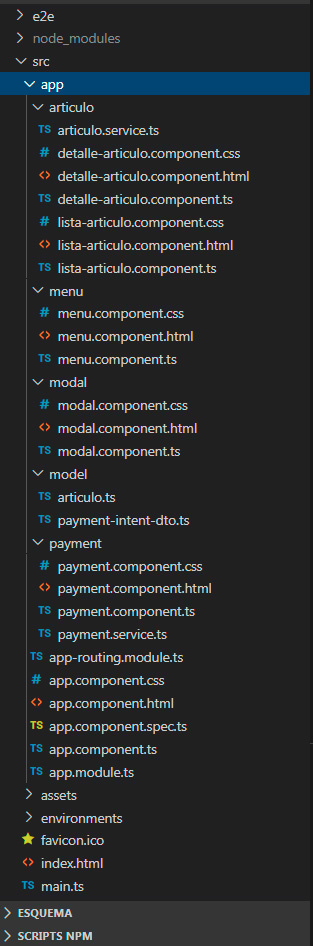
Y la interfaz PaymentIntentDto:

ng g interface model/payment-intent-dto

|  |  |
| --- | --- |
|  | export interface PaymentIntentDto { |
|  | token: string; |
|  | description: string; |
|  | amount: number; |
|  | currency: string; |
|  | } |

[**view raw**](https://gist.github.com/cavanosa/1976b419f223e59b25e8eb8528ff3aa8/raw/6ae712390bb408de7b915d04326482d3a66bb1db/PaymentIntentDto.ts)[**PaymentIntentDto.ts**](https://gist.github.com/cavanosa/1976b419f223e59b25e8eb8528ff3aa8#file-paymentintentdto-ts) hosted with  by [**GitHub**](https://github.com/)

Finalmente, tendremos este árbol de archivos:



**Rutas:**

Dejamos el app-routing.module.ts como sigue:

|  |  |
| --- | --- |
|  | import { NgModule } from '@angular/core'; |
|  | import { Routes, RouterModule } from '@angular/router'; |
|  | import { ListaArticuloComponent } from './articulo/lista-articulo/lista-articulo.component'; |
|  | import { DetalleArticuloComponent } from './articulo/detalle-articulo/detalle-articulo.component'; |
|  |  |
|  | const routes: Routes = [ |
|  | {path: '', component: ListaArticuloComponent}, |
|  | {path: 'lista', component: ListaArticuloComponent}, |
|  | {path: 'detalle/:id', component: DetalleArticuloComponent}, |
|  | {path: '\*\*', redirectTo: 'lista', pathMatch: 'full'} |
|  | ]; |
|  |  |
|  | @NgModule({ |
|  | imports: [RouterModule.forRoot(routes)], |
|  | exports: [RouterModule] |
|  | }) |
|  | export class AppRoutingModule { } |

[**view raw**](https://gist.github.com/cavanosa/fe0193196c1dd36d67969a909f3960e1/raw/b82634f50eec56631002669bb40ea8cf6474326f/app-routing.module.ts)[**app-routing.module.ts**](https://gist.github.com/cavanosa/fe0193196c1dd36d67969a909f3960e1#file-app-routing-module-ts) hosted with  by [**GitHub**](https://github.com/)

La vista principal será la lista de artículos.

**app.module.ts:**

Como vamos a utilizar un modal tenemos que añadir un entryComponent, donde irá el ModalComponent. Dejamos el app.module.ts así:

|  |  |
| --- | --- |
|  | import { BrowserModule } from '@angular/platform-browser'; |
|  | import { NgModule } from '@angular/core'; |
|  |  |
|  | import { AppRoutingModule } from './app-routing.module'; |
|  | import { AppComponent } from './app.component'; |
|  |  |
|  | import { MenuComponent } from './menu/menu.component'; |
|  | import { ModalComponent } from './modal/modal.component'; |
|  | import { ListaArticuloComponent } from './articulo/lista-articulo.component'; |
|  | import { DetalleArticuloComponent } from './articulo/detalle-articulo.component'; |
|  | import { PaymentComponent } from './payment/payment.component'; |
|  |  |
|  | // ng-bootstrap, toastr, httpclient, forms y ngx-stripe |
|  | import { NgbModalModule } from '@ng-bootstrap/ng-bootstrap'; |
|  | import { HttpClientModule } from '@angular/common/http'; |
|  | import { FormsModule, ReactiveFormsModule } from '@angular/forms'; |
|  | import { NgxStripeModule } from 'ngx-stripe'; |
|  | import { BrowserAnimationsModule } from '@angular/platform-browser/animations'; |
|  | import { ToastrModule } from 'ngx-toastr'; |
|  |  |
|  | @NgModule({ |
|  | declarations: [ |
|  | AppComponent, |
|  | MenuComponent, |
|  | ModalComponent, |
|  | ListaArticuloComponent, |
|  | DetalleArticuloComponent, |
|  | PaymentComponent |
|  | ], |
|  | imports: [ |
|  | BrowserModule, |
|  | NgbModalModule, |
|  | HttpClientModule, |
|  | FormsModule, |
|  | ReactiveFormsModule, |
|  | ToastrModule.forRoot(), |
|  | BrowserAnimationsModule, |
|  | NgxStripeModule.forRoot('mi\_clave\_publica'), |
|  | AppRoutingModule |
|  | ], |
|  | entryComponents: [ModalComponent], |
|  | providers: [], |
|  | bootstrap: [AppComponent] |
|  | }) |
|  | export class AppModule { } |

[**view raw**](https://gist.github.com/cavanosa/0c1c138029393b6ab1b5b2f75e93dfe1/raw/fcf17ad1b319ca89977cfe6473d96b460c972260/app.module.ts)[**app.module.ts**](https://gist.github.com/cavanosa/0c1c138029393b6ab1b5b2f75e93dfe1#file-app-module-ts) hosted with  by [**GitHub**](https://github.com/)

**Menu:**

Editamos el archivo menu.component.html para crear un navbar:

|  |  |
| --- | --- |
|  | <nav class="navbar navbar-expand-lg navbar-dark bg-dark"> |
|  | <a class="navbar-brand" href="#">Stripe con Angular</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" routerLink="/lista">Inicio</a> |
|  | </li> |
|  | </ul> |
|  | <ul class="navbar-nav ml-auto"> |
|  | <li class="nav-item "> |
|  | <a class="nav-link" href="#">Blog</a> |
|  | </li> |
|  | </ul> |
|  | </div> |
|  | </nav> |

[**view raw**](https://gist.github.com/cavanosa/0fb3b28a27ce289d5da3ef9478b58e59/raw/8d573cbc18221d569c3d8ec6f2d3a079aaf1e440/menu.component.html)[**menu.component.html**](https://gist.github.com/cavanosa/0fb3b28a27ce289d5da3ef9478b58e59#file-menu-component-html) hosted with  by [**GitHub**](https://github.com/)

**app.component.html:**

Ya para finalizar editamos el html del componente principal dejándolo como se muestra para que contenga sólo el menú:

|  |  |
| --- | --- |
|  | <app-menu></app-menu> |
|  |  |
|  | <router-outlet></router-outlet> |

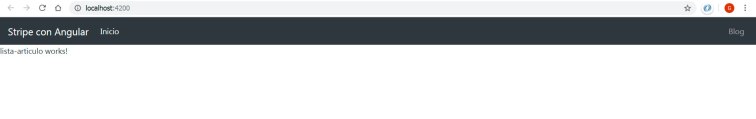
[**view raw**](https://gist.github.com/cavanosa/b49622bb45142bc7944c97f0ffc81c5b/raw/c4d3b87cad8f21bfea28eecbeded0873a303846a/app.component.html)[**app.component.html**](https://gist.github.com/cavanosa/b49622bb45142bc7944c97f0ffc81c5b#file-app-component-html) hosted with  by [**GitHub**](https://github.com/)

**Ejecutar el proyecto:**

En la consola tecleamos:

ng serve -o

Si está todo correcto esto es lo que deberíamos ver:



Y hasta aquí el post. En el siguiente crearemos la lista de artículos y accederemos a los detalles de cada uno.