

# Práctica Angular

La dirección del repositorio es <https://github.com/MaiolCarmona/angular-practica.git>

# Mock de backend

Se han guardado los diferentes conjuntos de datos (banners, categorías, productos) en cuatro data buckets, que se sirven usando los templates y helpers de Mockoon desde los distintos endpoints. Hay definidos cinco endpoints, a través de templates y helpers de Mockoon dos de ellos sirven distintos datos de forma dinámica en función de parámetros de url.

Se ha guardado una copia del entorno de Mockoon utilizado en el repositorio, en la carpeta raíz del proyecto con el nombre “mock-api-rest-practica-angular.json”.

## Data Buckets:

## Endpoints:

The screenshot shows the Mock API Rest Practice application interface. The top navigation bar includes tabs for Routes (5), Data (4), Headers (2), Callbacks, Logs (50), Proxy, and Settings. The left sidebar displays the URL `localhost:3000/` and the title "Mock api rest practice ...". The main content area lists several routes:

- `/categories` (GET)
- `/products/category` (GET)
- `/products/category...` (GET) - This route is currently selected, as indicated by the blue background and the active tab. It has a documentation box: "Add documentation for this route" and a response section: "Response 1 (200)". The "Status & Body" tab is selected, showing the status "200 - OK" and a placeholder "Add documentation for this response". Below this is a code editor with the following content:

```
1 {{data (urlParam 'category') (concat '$[?(@.id==' (urlParam 'id') ')]')}}
```

 and buttons for Body, Inline, File, and Data.
- `/panoramicBanners` (GET)
- `/regularBanners` (GET)

The right sidebar contains sections for Filter, Headers, Callbacks, and Logs, with the Logs section showing 50 entries.

## Aplicación de Angular

Se han añadido Bootstrap y el tema “lux” de Bootswatch (<https://bootswatch.com>), que sobreescribe ciertos estilos de Bootstrap y suponía un punto de partida más cercano en estilos al diseño propuesto. También se ha añadido el módulo de FontAwesome para utilizar sus iconos en diversos puntos.

En cuanto a la estructura del proyecto se ha dividido en ocho componentes, **nav** y **footer** que están presentes en todo momento, **home** para la página principal, **gallery** para ver el conjunto de productos de una categoría y **detail** para mostrar un único producto, y un componente **about**, cargando estos tres con router-outlet a través de la navegación. El componente **product-card** se utiliza como un subcomponente dentro del componente **gallery**.

Se han creado dos servicios BannersService para recuperar del backend los banners y ShopService las categorías y los productos (todos los de una categoría, o uno por id).

Las diferentes entidades (banners, categorías, productos) se han modelizado a través de interfaces.

Se expone a continuación una breve explicación de los diferentes componentes. Gran del responsiveness del diseño viene dado por Bootstrap, en lo sucesivo solo detallaré en cuanto al responsiveness lo que no derive directamente de estos estilos que se aplican al usar Bootstrap.

### Nav

Utiliza el servicio shop.service para recuperar las categorías y establecer dinámicamente los enlace de un dropdown menú, si bien al solo haber por el momento productos en una categoría el routerLink de estos enlaces se ha mantenido estático apuntando a la categoría “dresses”.



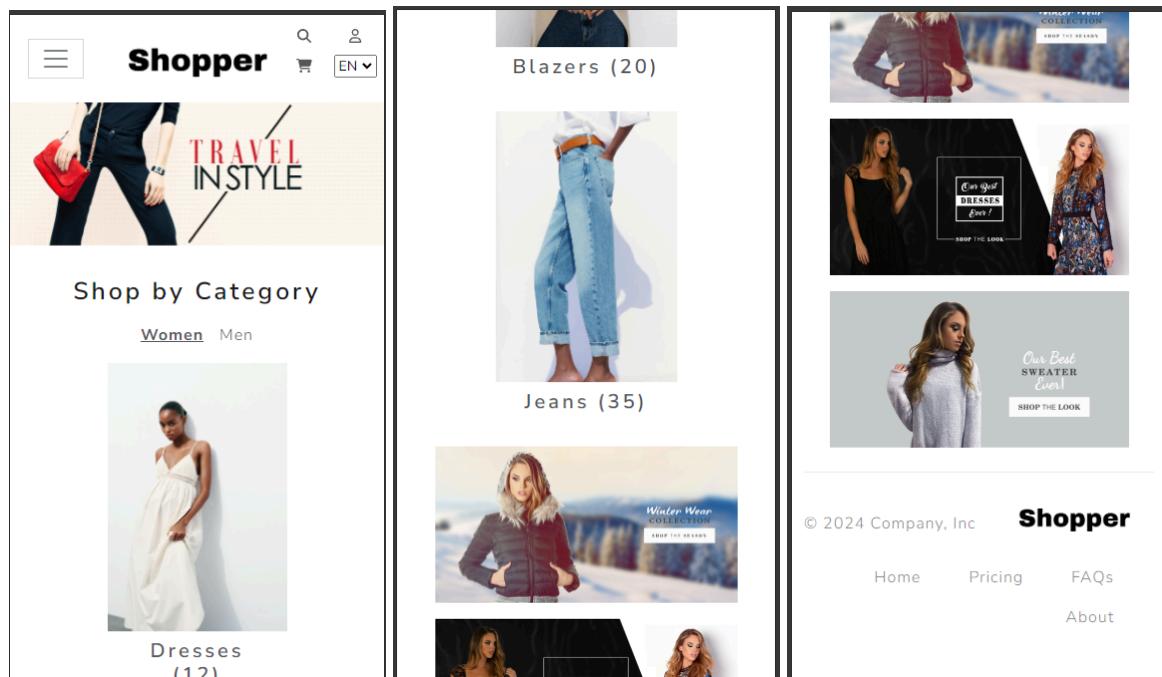
## Home

Utiliza el servicio BannersService para recuperar los banners panorámicos y los banners “normales”. Los primeros se integran en un carousel de Bootstrap y los normales aparecen en una sección a continuación.

A través del servicio ShopService se recuperan las categorías con las que se hacen las cards se la sección “shop by category”. El renderizado de las cards está condicionado a que el género de la categoría coincida con el seleccionado que el usuario puede cambiar a través de unos enlaces (no hay categorías para “men” por lo que de seleccionarlo no se renderiza nada).

Por debajo de cierto tamaño del viewport y usando la propiedad css order se cambia el orden de la sección de las categorías con la de los banners para que la zona de interacción del usuario no quede tan abajo. También se cambia la disposición de los banner a una vertical. La sección de categoría tiene responsive a través del uso de display:flex y flex-wrap:wrap y ciertos widths con tamaños relativos.

Móvil (375 x 667):



Tablet vertical (820 x 1180)

**Shopper**

MAKE THE MOST WITH OUR COLLECTIONS

Shop by Category

Women Men

Dresses (12) Skirts (18) Jackets (12)

© 2024 Company, Inc.

**Shop by Category**

Women Men

Dresses (12) Skirts (18) Jackets (12)

Blazers (20) Jeans (35)

Home Pricing FAQs About

Monitor:

HOME CATEGORIES ABOUT US

**Shopper**

TRAVEL IN STYLE

Shop by Category

Women Men

Dresses (12) Skirts (18) Jackets (12) Blazers (20) Jeans (35)

© 2024 Company, Inc.

**Shopper**

Home Pricing FAQs About

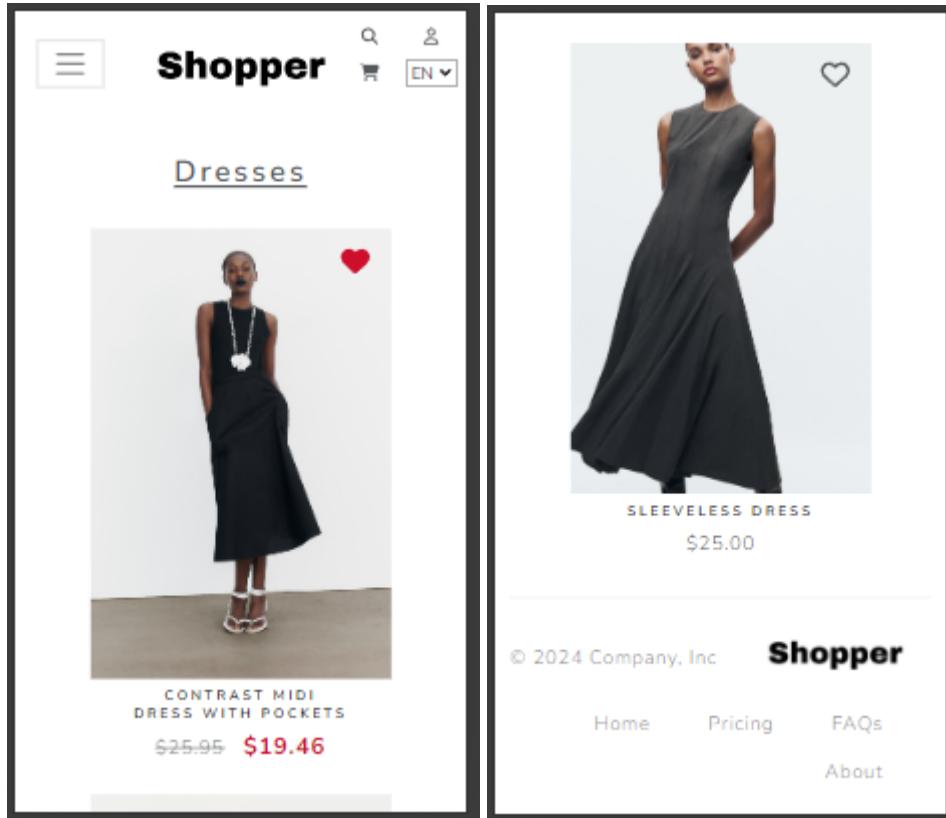
## Gallery

Se accede a este componente a través de los routerLink de las cards de categorías de Home o de los enlaces del dropdown “categorías” del nav (todos dirigen a /gallery/dresses porque solo hay productos para esa categoría, en caso de haber más se establecerían también de forma dinámica los routerLink). El path que corresponde al componente es /gallery/:category.

El componente a través de ActivatedRoute recupera de ruta el parámetro “category” y lo pasa como argumento a una función del ShopService que recupera del backend los productos de una categoría. Por cada uno de estos productos se carga el subcomponente product-card que recibe como input el propio producto. En este componente product-card se condiciona la aparición de cierta información (precio con descuento, ícono de favorito) y ciertos estilos al valor de ciertos campos de los productos.

Las cards están dispuestas utilizando flex:grid y el uso de la función repeat con el argumento auto-fill en la propiedad grid-template-columns hace que el layout sea responsive variando el número de elementos por fila en función del ancho del viewport.

Móvil (375 x 667):



Monitor:

The screenshot shows a shopping website with a navigation bar at the top. The main content area displays a grid of dresses. Each item has a small heart icon in the top right corner. Below each image, the product name, original price, and discounted price are listed.

Product	Original Price	Discounted Price
CONTRAST MIDI DRESS WITH POCKETS	\$25.95	\$19.46
TRF FITTED DENIM DRESS	\$25.95	\$25.95
ZW COLLECTION SHIRT DRESS WITH BELT	\$25.95	\$25.95
STRAPPY MIDI DRESS	\$25.00	\$18.75
CREASED DRESS WITH BELT	\$25.00	\$25.00
TULLE PRINT DRESS WITH RUFFLES	\$25.00	\$25.00
EMBROIDERED SHORT DRESS	\$25.00	\$25.00
PRINTED TULLE DRESS	\$25.00	\$18.75
SATIN ANIMAL PRINT DRESS	\$25.00	\$25.00
SHORT TIERED DRESS	\$25.00	\$25.00
FRINGED DRESS	\$25.00	\$25.00
SLEEVELESS DRESS	\$25.00	\$25.00

At the bottom of the page, there is a footer with links to Home, Pricing, FAQs, and About.

## Detail

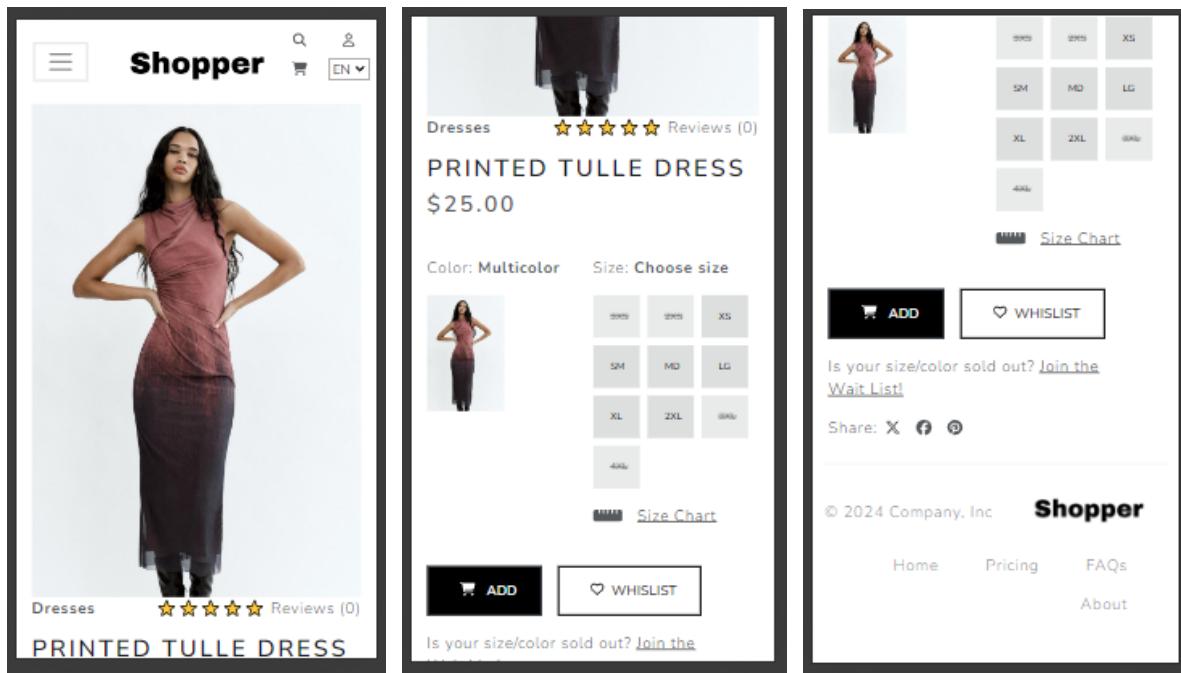
El path que corresponde a este componente es /gallery/:category/:id . Las cards del componente anterior tienen el routerLink:  
[routerLink]="" "/gallery/" + categoryName +'/' + product.id" (categoryName es el atributo de la clase correspondiente al componente Gallery, al que se le asignó el valor recuperado de los parámetros de url).

El componente a través de ActivatedRoute recupera de ruta los parámetros "category" y "id" y los pasa como argumento a un método del ShopService que trae

del backend un producto por categoría y id. Los valores de los campos de este producto son renderizados en las diferentes secciones del template, y se condicionan los estilos o la aparición de algunos elementos (mensaje de “out of stock”, tachado de las tallas sin stock, botones de add to cart y waitlist...) a ciertos valores de esos campos.

Se ha utilizado en esta ocasión display:grid con grid-template-areas y algunas media queries para mostrar tres configuraciones distintas del template, si bien la configuración intermedia quizá no se utilice en demasiados dispositivos, resultaba más equilibrada en ciertos anchos y aspect-ratios.

Móvil (375 x 667):



Surface Duo (540 x 720):



**Dresses** ★★★★★ Reviews (0)

### PRINTED TULLE DRESS

\$25.00

**ADD** **WHISLIST**

Is your size/color sold out?  
[Join the Wait List!](#)

Share:

**Color:** Multicolor **Size:** Choose size

Sizes	Sizes	XS	SM
MD	LG	XL	2XL
SM	4XL		



**ADD** **WHISLIST**

Is your size/color sold out?  
[Join the Wait List!](#)

Share:

**Color:** Multicolor **Size:** Choose size

Sizes	Sizes	XS	SM
MD	LG	XL	2XL
SM	4XL		

[Size Chart](#)

© 2024 Company, Inc

**Shopper**

[Home](#) [Pricing](#) [FAQs](#) [About](#)

Monitor:

---

[HOME](#) [CATEGORIES ▾](#) [ABOUT US](#)



**OUT OF STOCK**

**Dresses** ★★★★★ Reviews (0)

### EMBROIDERED SHORT DRESS

\$25.00 (Out of Stock)

**Color:** Multicolor



**Size:** Choose size

Sizes	Sizes	XS	SM	MD	L
XS	SM	XS	SM	MD	L

[Size Chart](#)

**WAIT LIST** **WHISLIST**

Is your size/color sold out? [Join the Wait List!](#)

Share:



### EMBROIDERED SHORT DRESS

\$25.00 (Out of Stock)

Color: Multicolor



Size: Choose size

SMS	SMG	XS	SX	MD	LG
XS	SML	SML	SML	LG	

[Size Chart](#)

[WAIT LIST](#)

[WHISHLIST](#)

Is your size/color sold out? [Join the Wait List!](#)

Share: [X](#) [F](#) [D](#)

© 2024 Company, Inc

**Shopper**

[Home](#)

[Pricing](#)

[FAQs](#)

[About](#)

## About

Se accede a este componente desde enlaces en el nav y en el footer. El contenido de este componente es estático. Simplemente he incluido texto organizado en una o dos columnas en función de ancho del viewport y he aprovechado para probar una técnica que ví en un video de Kevin Powell (<https://www.youtube.com/@KevinPowell>) en el que se hace el font-size relativo al ancho del viewport y haciendo que el ancho del contenedor y los márgenes y padding sean relativos a este font-size varía el tamaño de los párrafos manteniendo las proporciones de la estructura, el número de palabras por linea.