

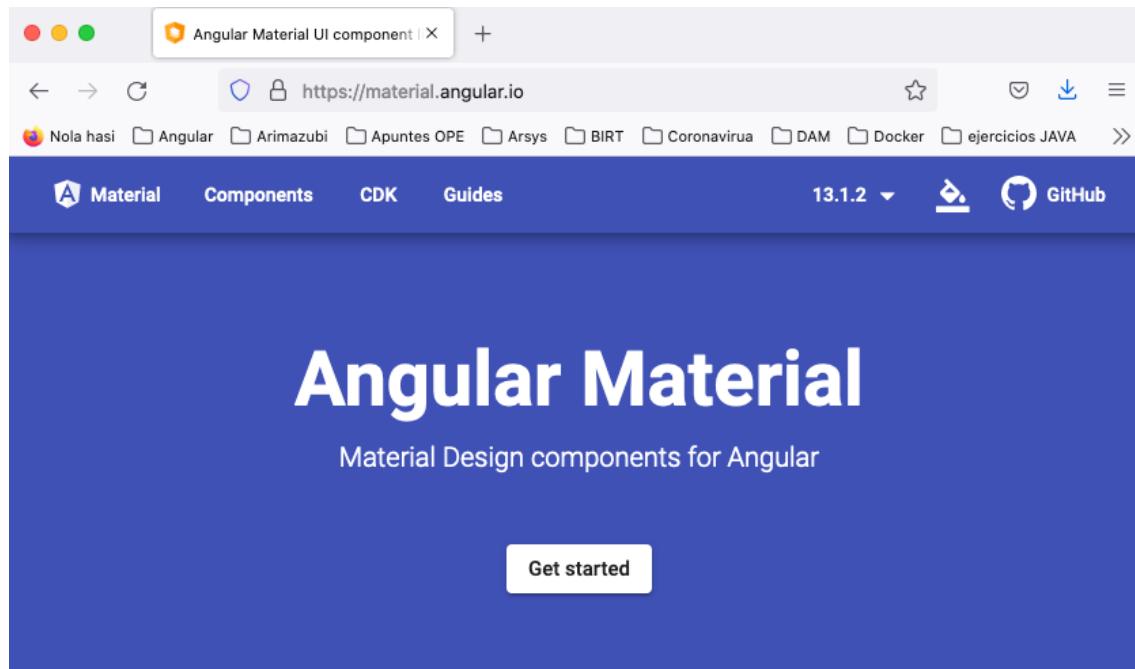
## Mapa localStorage paso a paso

He rehecho el proyecto porque ha habido importantes cambios en la Librería Angular Maps.

### 1. Creación del proyecto:

```
mapasLocalStorage — -zsh — 117x36
Last login: Wed Jan 19 13:22:52 on console
[jsersan@iMac-de-Jose ~ % cd /Users/jsersan/Desktop/Angular/mapasLocalStorage
[jsersan@iMac-de-Jose mapasLocalStorage % ng new appMapalocalStorage
[?] Would you like to add Angular routing? No
[?] Which stylesheet format would you like to use? SCSS [ https://sass-lang.com/documentation/syntax#scss ]
CREATE appMapalocalStorage/README.md (1065 bytes)
CREATE appMapalocalStorage/.editorconfig (274 bytes)
CREATE appMapalocalStorage/.gitignore (620 bytes)
CREATE appMapalocalStorage/angular.json (3287 bytes)
CREATE appMapalocalStorage/package.json (1085 bytes)
CREATE appMapalocalStorage/tsconfig.json (863 bytes)
CREATE appMapalocalStorage/.browserslistrc (600 bytes)
CREATE appMapalocalStorage/karma.conf.js (1438 bytes)
CREATE appMapalocalStorage/tsconfig.app.json (287 bytes)
CREATE appMapalocalStorage/tsconfig.spec.json (333 bytes)
CREATE appMapalocalStorage/.vscode/extensions.json (130 bytes)
CREATE appMapalocalStorage/.vscode/launch.json (474 bytes)
CREATE appMapalocalStorage/.vscode/tasks.json (938 bytes)
CREATE appMapalocalStorage/src/favicon.ico (948 bytes)
CREATE appMapalocalStorage/src/index.html (305 bytes)
CREATE appMapalocalStorage/src/main.ts (372 bytes)
CREATE appMapalocalStorage/src/polyfills.ts (2338 bytes)
CREATE appMapalocalStorage/src/styles.scss (80 bytes)
CREATE appMapalocalStorage/src/test.ts (745 bytes)
CREATE appMapalocalStorage/src/assets/.gitkeep (0 bytes)
CREATE appMapalocalStorage/src/environments/environment.prod.ts (51 bytes)
CREATE appMapalocalStorage/src/environments/environment.ts (658 bytes)
CREATE appMapalocalStorage/src/app/app.module.ts (314 bytes)
CREATE appMapalocalStorage/src/app/app.component.scss (0 bytes)
CREATE appMapalocalStorage/src/app/app.component.html (23332 bytes)
CREATE appMapalocalStorage/src/app/app.component.spec.ts (995 bytes)
CREATE appMapalocalStorage/src/app/app.component.ts (224 bytes)
✓ Packages installed successfully.
  Successfully initialized git.
jsersan@iMac-de-Jose mapasLocalStorage %
```

### 2. Instalamos Angular Material. Es un componente como bootstrap pero para Angular.



Por ejemplo, en Components>

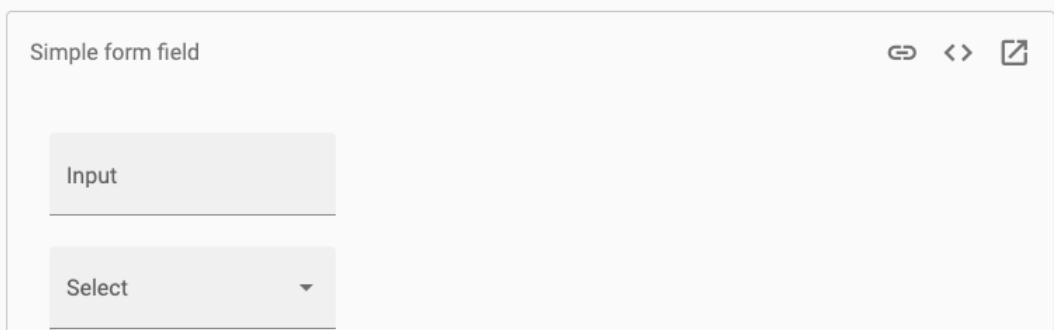
[OVERVIEW](#)    [API](#)    [EXAMPLES](#)

<mat-form-field> is a component used to wrap several Angular Material components and apply common [Text field](#) styles such as the underline, floating label, and hint messages.

In this document, "form field" refers to the wrapper component <mat-form-field> and "form field control" refers to the component that the <mat-form-field> is wrapping (e.g. the input, textarea, select, etc.)

The following Angular Material components are designed to work inside a <mat-form-field>:

- <input matNativeControl> & <textarea matNativeControl>
- <select matNativeControl>
- <mat-select>
- <mat-chip-list>



Entramos en el proyecto de Angular en el punto 1:

```
[jsersan@iMac-de-Jose appMapalocalStorage % ls -l
total 1688
-rw-r--r--  1 jsersan  staff   1065 19 ene 17:40 README.md
-rw-r--r--  1 jsersan  staff    3287 19 ene 17:40 angular.json
-rw-r--r--  1 jsersan  staff   1438 19 ene 17:40 karma.conf.js
drwxr-xr-x  607 jsersan  staff  19424 19 ene 17:41 node_modules
-rw-r--r--  1 jsersan  staff  832913 19 ene 17:41 package-lock.json
-rw-r--r--  1 jsersan  staff   1085 19 ene 17:40 package.json
drwxr-xr-x  11 jsersan  staff    352 19 ene 17:40 src
-rw-r--r--  1 jsersan  staff   287 19 ene 17:40 tsconfig.app.json
-rw-r--r--  1 jsersan  staff   863 19 ene 17:40 tsconfig.json
-rw-r--r--  1 jsersan  staff   333 19 ene 17:40 tsconfig.spec.json
jsersan@iMac-de-Jose appMapalocalStorage % ]
```

## Install Angular Material

Use the Angular CLI's installation [schematic](#) to set up your Angular Material project by running the following command:

```
ng add @angular/material
```

Entonces:

```
jsersan@iMac-de-Jose appMapalocalStorage % ng add @angular/material
  i Using package manager: npm
  ✓ Found compatible package version: @angular/material@13.1.2.
  ✓ Package information loaded.

The package @angular/material@13.1.2 will be installed and executed.
Would you like to proceed? (Y/n) █
```

La instalación completa:

```
The package @angular/material@13.1.2 will be installed and executed.
Would you like to proceed? Yes
✓ Package successfully installed.
? Choose a prebuilt theme name, or "custom" for a custom theme: Deep
r.io?theme=deeppurple-amber ]
? Set up global Angular Material typography styles? No
? Set up browser animations for Angular Material? Yes
UPDATE package.json (1151 bytes)
✓ Packages installed successfully.
UPDATE src/app/app.module.ts (423 bytes)
UPDATE angular.json (3461 bytes)
UPDATE src/index.html (564 bytes)
UPDATE src/styles.scss (181 bytes)
jsersan@iMac-de-Jose appMapalocalStorage % █
```

Ahora hemos instalado Angular Material, con el kit completo de animaciones y desarrollo.

Para que poder utilizar animaciones en nuestro proyecto en la página oficial de angular/material, por ejemplo, para un módulo Slider debemos importar en app.module.ts:

## Display a component

Let's display a slider component in your app and verify that everything works.

You need to import the `MatSliderModule` that you want to display by adding the following lines to your `app.module.ts` file.

```
import { MatSliderModule } from '@angular/material/slider';

@NgModule ({
  imports: [
    MatSliderModule,
  ]
})
class AppModule {}
```

Al haberle dicho que sí a “¿Set up Browser Animations for Angular Material?”, automáticamente nos importa el módulo BrowserAnimationsModule

```
TS app.module.ts M X
appMapalocalStorage > src > app > TS app.module.ts > ...
1  import { NgModule } from '@angular/core';
2  import { BrowserModule } from '@angular/platform-browser';
3
4  import { AppComponent } from './app.component';
5  import { BrowserAnimationsModule } from '@angular/platform-browser/animations';
6
7  @NgModule({
8    declarations: [
9      AppComponent
10    ],
11    imports: [
12      BrowserModule,
13      BrowserAnimationsModule
14    ],
15    providers: [],
16    bootstrap: [AppComponent]
17  })
18  export class AppModule { }
19
```

Instalamos hammer.js que es una librería para detectar gestos “multi-touch”

```
[jsersan@iMac-de-Jose appMapalocalStorage % npm i --save hammerjs
up to date, audited 1018 packages in 3s
91 packages are looking for funding
  run `npm fund` for details
40 vulnerabilities (3 low, 37 moderate)

To address issues that do not require attention, run:
  npm audit fix

To address all issues (including breaking changes), run:
  npm audit fix --force

Run `npm audit` for details.
jsersan@iMac-de-Jose appMapalocalStorage % ]
```

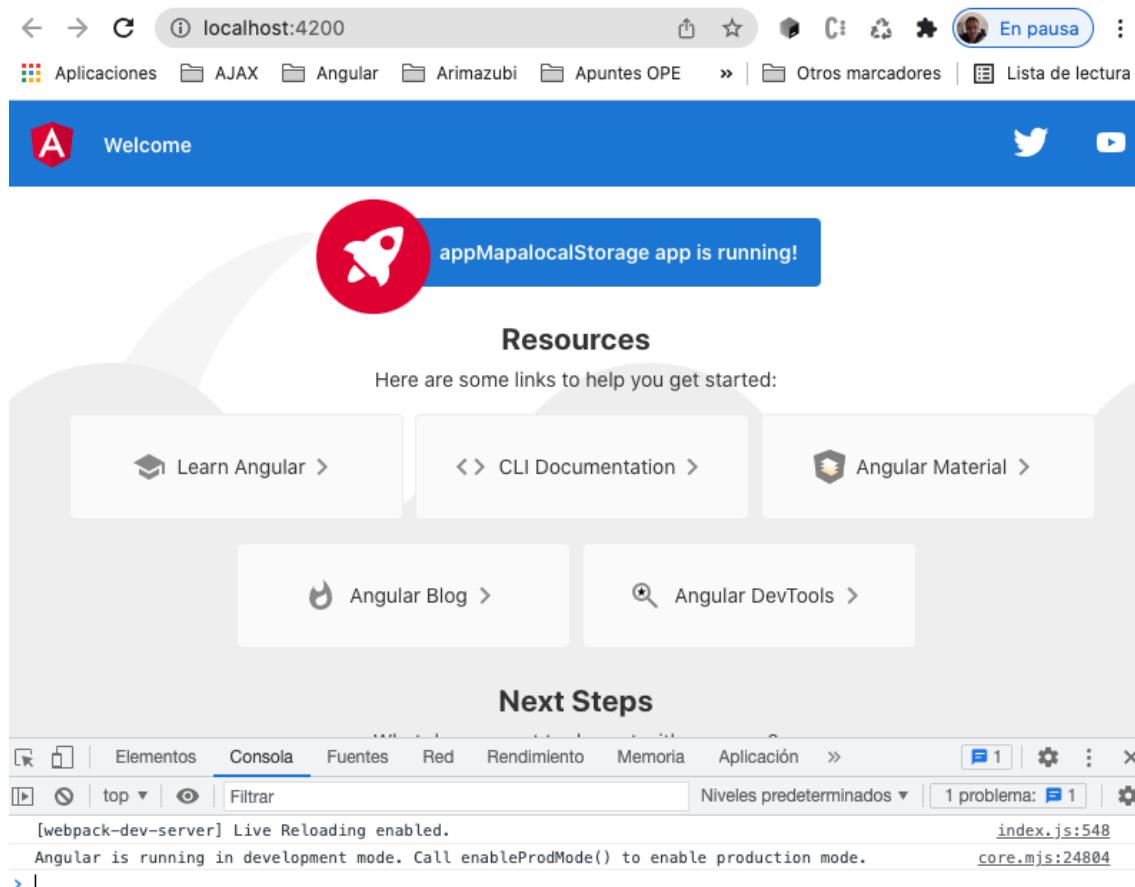
Ahora debemos importarlo en src/main.ts:

```

  styles.scss M   TS main.ts M X
appMapalocalStorage > src > TS main.ts > ...
1  import { enableProdMode } from '@angular/core';
2  import { platformBrowserDynamic } from '@angular/platform-browser-dynamic';
3
4  import { AppModule } from './app/app.module';
5  import { environment } from './environments/environment';
6
7  import 'hammerjs';
8
9  if (environment.production) {
10  |   enableProdMode();
11  }
12
13 platformBrowserDynamic().bootstrapModule(AppModule)
14   | .catch(err => console.error(err));

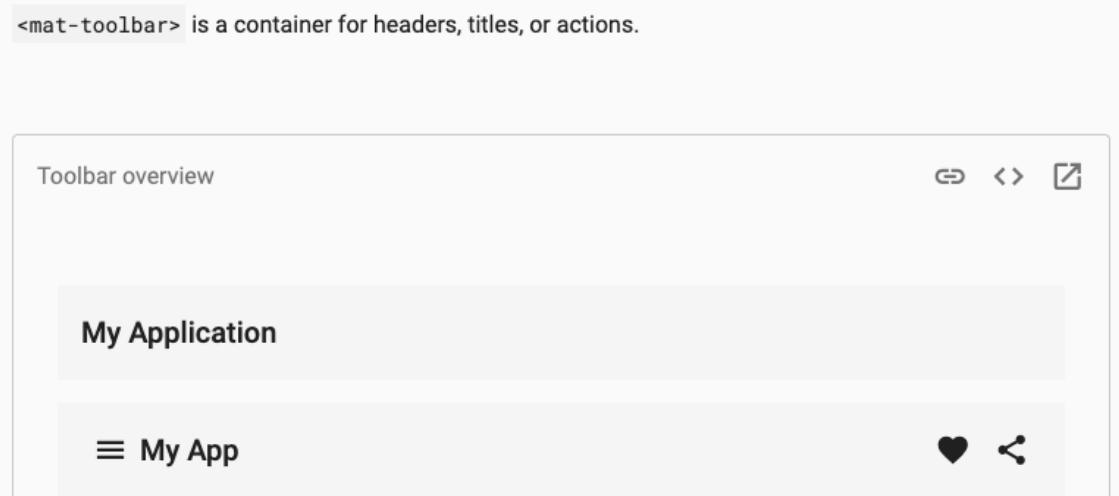
```

Lo probamos:



### 3. Uso de componentes:

Incluimos una barra de herramientas:



Copiamos el contenido y lo pegamos en app.component.html:

```

    styles.scss M      app.component.html M X
  appMapalocalStorage > src > app > app.component
  1 |   <mat-toolbar>
  2 |     <span>Mis mapas</span>
  3 |   </mat-toolbar>

```

Resultado: aparecen los siguientes errores porque no importado los componentes de material.

The screenshot shows the browser's developer tools DevTools open. A message at the top says "El esquema de color preferido por el sistema ha cambiado. Para aplicar este cambio a DevTools, vuelve a cargarlo." Below this, the console tab shows several error messages from the Angular compiler:

- [webpack-dev-server] ERROR index.js:558
 src/app/app.component.html:1:1 - error NG8001: 'mat-toolbar' is not a known element:
 1. If 'mat-toolbar' is an Angular component, then verify that it is part of this module.
 2. If 'mat-toolbar' is a Web Component then add 'CUSTOM\_ELEMENTS\_SCHEMA' to the '@NgModule.schemas' of this component to suppress this message.
- 1 <mat-toolbar>
 ~~~~~
- src/app/app.component.ts:5:16
 5 templateUrl: './app.component.html',
 ~~~~~
 Error occurs in the template of component AppComponent.

La librería de material está segmentada: si quiero utilizar un menú debo importar un menú, si quiero un toolbar, debo importar el toolbar. Esto me ayuda a que la aplicación solo tenga los elementos que necesita.

Así, hacemos click en la parte de API de la documentación:

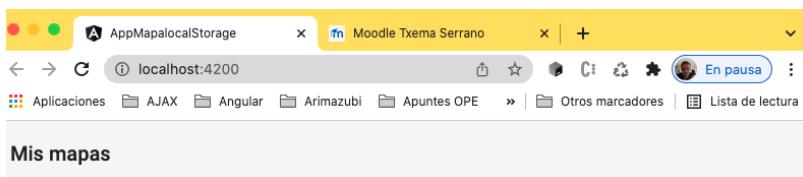
Lo importamos en app.module.ts:

```

1  import { NgModule } from '@angular/core';
2  import { BrowserModule } from '@angular/platform-browser';
3
4  import { AppComponent } from './app.component';
5  import { BrowserAnimationsModule } from '@angular/platform-browser/animations';
6
7  import {MatToolbarModule} from '@angular/material/toolbar';
8
9  @NgModule({
10    declarations: [
11      AppComponent
12    ],
13    imports: [
14      BrowserModule,
15      BrowserAnimationsModule,
16      MatToolbarModule
17    ],

```

Guardo los cambios:



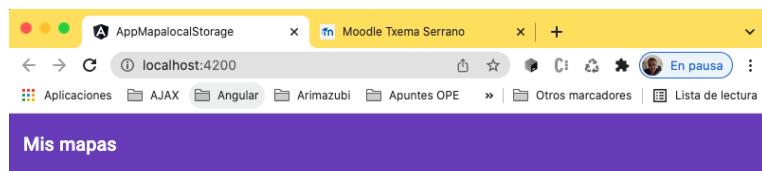
Cambiamos el color:

```

1  <mat-toolbar color="primary">
2    <span>Mis mapas</span>
3  </mat-toolbar>

```

Resultado:



Si el toolBar no cogiera todo el ancho de la página, en styles.css:

```
styles.scss M X app.component.html M app.component.scss TS app.module.ts M
appMapalocalStorage > src > styles.scss > ...
1 /* You can add global styles to this file, and also import other style files */
2
3 html, body { height: 100%; }
4 body { margin: 0; font-family: Roboto, "Helvetica Neue", sans-serif; }
5
6
```

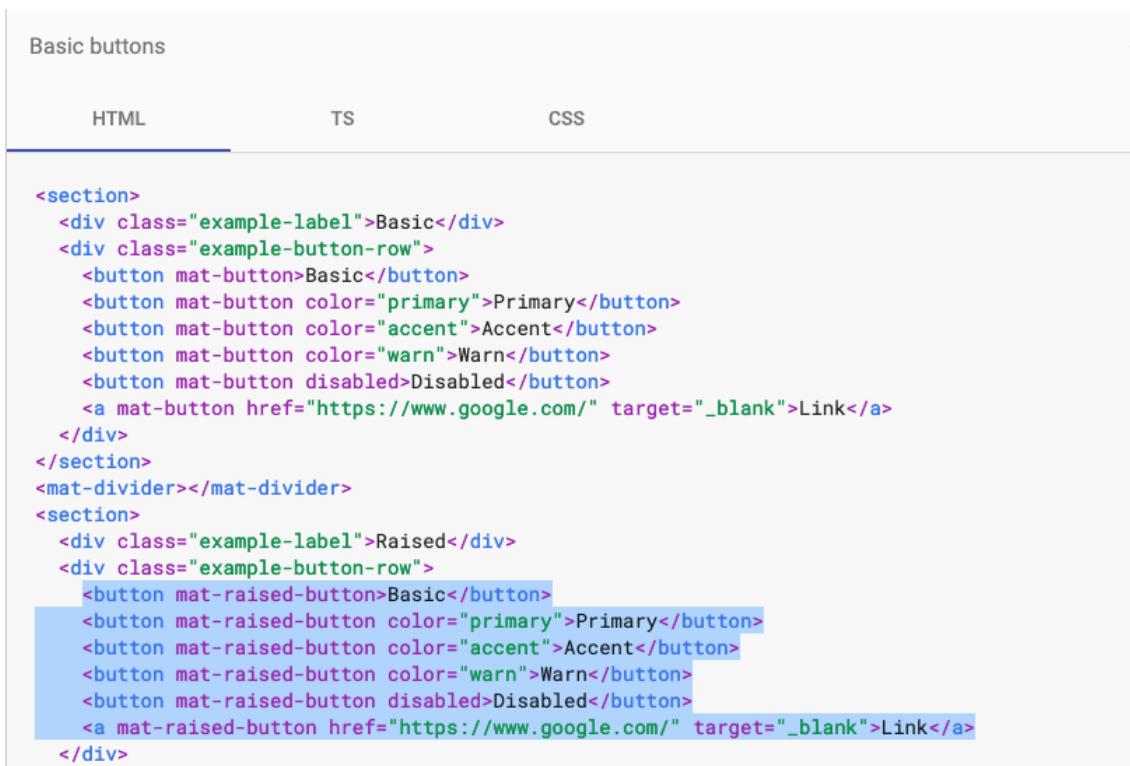
Si no queremos que quede totalmente pegado arriba. Cuando definamos esta clase en un control lo usaremos:

```
styles.scss M X app.component.html M app.component.scss TS app.module.ts M
appMapalocalStorage > src > styles.scss > ...
1 /* You can add global styles to this file, and also import other style files */
2
3 html, body { height: 100%; }
4 body { margin: 0; font-family: Roboto, "Helvetica Neue", sans-serif; }
5
6 .main-container {
7   margin-top: 10px;
8 }
```

En la documentación vamos a ver los ejemplos de los botones:

A screenshot of the Angular Material design system documentation. The top navigation bar says 'Basic buttons'. Below it, there are four rows of buttons labeled 'Basic', 'Raised', 'Stroked', and 'Flat'. Each row contains buttons in various states: 'Basic' (light blue), 'Primary' (dark blue), 'Accent' (pink), 'Warn' (red), 'Disabled' (gray), and 'Link' (blue). The 'Primary' button in each row is highlighted.

El código de todos estos botones los ponemos en app.component.html:



```

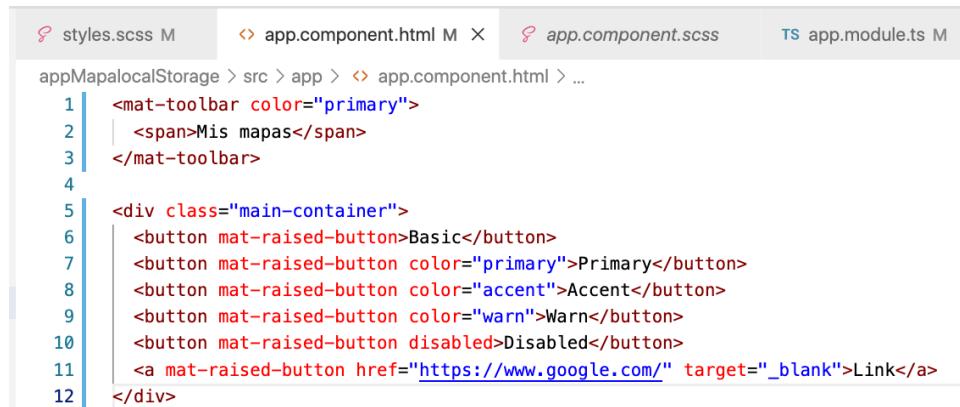
Basic buttons

HTML          TS          CSS

<section>
  <div class="example-label">Basic</div>
  <div class="example-button-row">
    <button mat-button>Basic</button>
    <button mat-button color="primary">Primary</button>
    <button mat-button color="accent">Accent</button>
    <button mat-button color="warn">Warn</button>
    <button mat-button disabled>Disabled</button>
    <a mat-button href="https://www.google.com/" target="_blank">Link</a>
  </div>
</section>
<mat-divider></mat-divider>
<section>
  <div class="example-label">Raised</div>
  <div class="example-button-row">
    <button mat-raised-button>Basic</button>
    <button mat-raised-button color="primary">Primary</button>
    <button mat-raised-button color="accent">Accent</button>
    <button mat-raised-button color="warn">Warn</button>
    <button mat-raised-button disabled>Disabled</button>
    <a mat-raised-button href="https://www.google.com/" target="_blank">Link</a>
  </div>
</section>

```

Así:

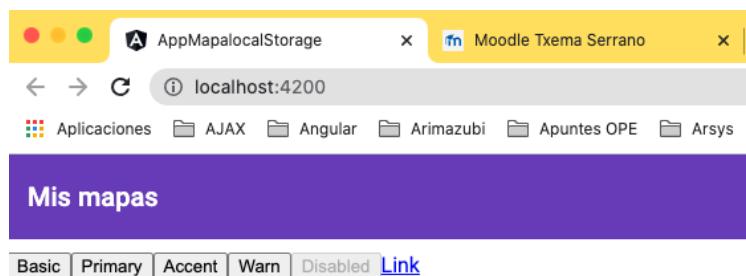


```

styles.scss M      app.component.html M X      app.component.scss      app.module.ts M
appMapalocalStorage > src > app > app.component.html > ...
1  <mat-toolbar color="primary">
2  |   <span>Mis mapas</span>
3  </mat-toolbar>
4
5  <div class="main-container">
6  |   <button mat-raised-button>Basic</button>
7  |   <button mat-raised-button color="primary">Primary</button>
8  |   <button mat-raised-button color="accent">Accent</button>
9  |   <button mat-raised-button color="warn">Warn</button>
10 |   <button mat-raised-button disabled>Disabled</button>
11 |   <a mat-raised-button href="https://www.google.com/" target="_blank">Link</a>
12 </div>

```

Resultado:



No obstante, debemos importar el módulo en app.module.ts:

The screenshot shows the 'API' tab of the Angular Material Button API reference. It displays the code snippet: `import {MatButtonModule} from '@angular/material/button';`

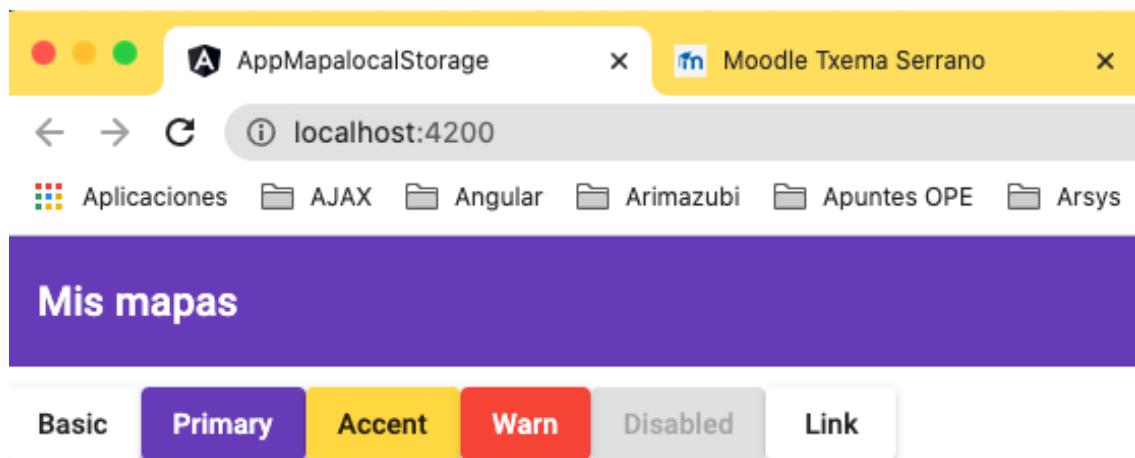
Así en app.module.ts::

```

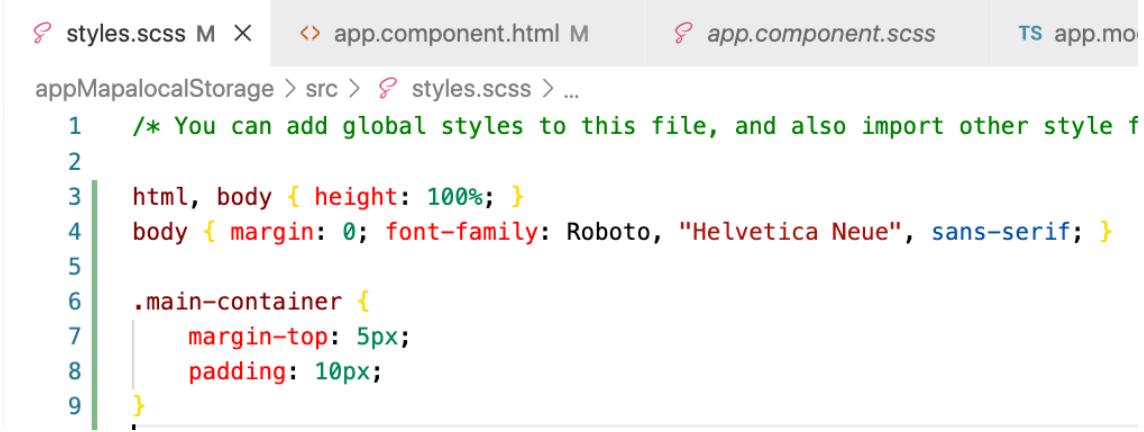
8 | import {MatButtonModule} from '@angular/material/button';
9 |
10 | @NgModule({
11 |   declarations: [
12 |     AppComponent
13 |   ],
14 |   imports: [
15 |     BrowserModule,
16 |     BrowserAnimationsModule,
17 |     MatToolbarModule,
18 |     MatButtonModule
19 |   ],

```

Ahora ya se ven los botones bien:



Para que no se vean pegados a los bordes:

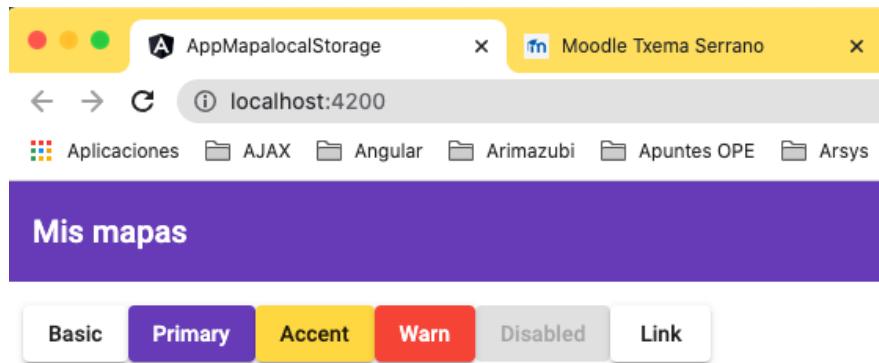


```

1  /* You can add global styles to this file, and also import other style f
2
3  html, body { height: 100%; }
4  body { margin: 0; font-family: Roboto, "Helvetica Neue", sans-serif; }
5
6  .main-container {
7    margin-top: 5px;
8    padding: 10px;
9

```

Resultado:



Si le ponemos también una barra de progreso:



The screenshot shows a code editor with two tabs: 'HTML' and 'TS'. The 'HTML' tab contains the code: <mat-progress-bar mode="indeterminate"></mat-progress-bar>. The 'TS' tab is empty. Below the code editor is a visual representation of an indeterminate progress bar, which is a horizontal bar with a blue segment on the left and a grey segment on the right.

Así:

```

11 |   <a mat-raised-button href="https://www.google.com/" target="_blank">Link</a>
12 |   </div>
13 |
14 |   <mat-progress-bar mode="indeterminate"></mat-progress-bar>

```

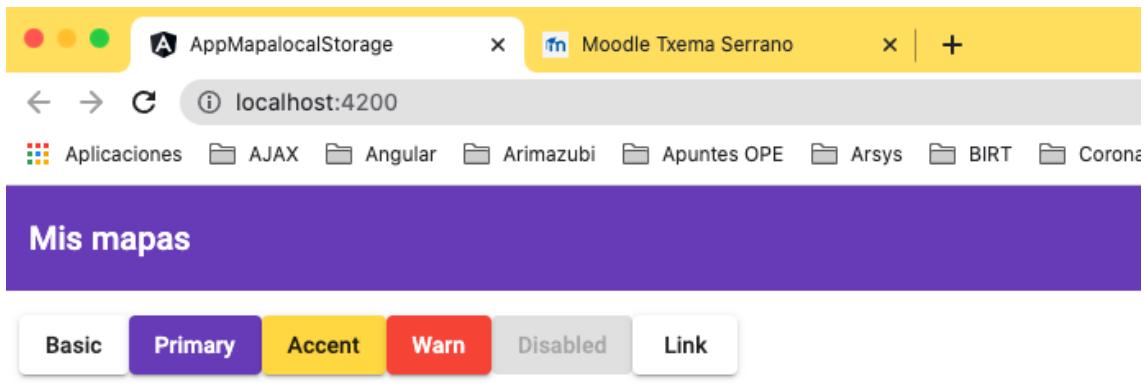
Y debemos importarla.

```

  styles.scss M   app.component.html M   TS app.component.ts   TS app.module.ts M X
appMapalocalStorage > src > app > TS app.module.ts > AppModule
  3
  4   import { AppComponent } from './app.component';
  5   import { BrowserAnimationsModule } from '@angular/platform-browser/animations';
  6
  7   import { MatToolbarModule } from '@angular/material/toolbar';
  8   import {MatButtonModule} from '@angular/material/button';
  9   import {MatProgressBarModule} from '@angular/material/progress-bar';
10
11 @NgModule({
12   declarations: [
13     AppComponent
14   ],
15   imports: [
16     BrowserModule,
17     BrowserAnimationsModule,
18     MatToolbarModule,
19     MatButtonModule,
20     MatProgressBarModule
21   ],

```

Resultado:



#### 4.- Componentes personalizados.

Como app.module.ts está creciendo demasiado a medida que incluyamos componentes de material, creamos un módulo para material. El app.module.ts debe ser lo más sencillo posible ya que aquí es donde importamos las cosas necesarias.

Creamos módulos personalizados. En la terminal:

PROBLEMAS	SALIDA	CONSOLA DE DEPURACIÓN	TERMINAL
			jsersan@iMac-de-Jose appMapalocalStorage % ng g module material --flat CREATE src/app/material.module.ts (194 bytes) jsersan@iMac-de-Jose appMapalocalStorage % █

Así el módulo generado:

The screenshot shows the VS Code interface with the following details:

- EXPLORADOR:** Shows the project structure under "MAPALOCALSTORAGE". It includes a .angular folder, a .vscode folder, a node\_modules folder, and a src folder containing an app folder. Inside the app folder are app.component.html, app.component.scss, app.component.spec.ts, app.component.ts, app.module.ts, and material.module.ts (which is currently selected).
- TERMINAL:** Shows the command: "ng g module material --flat" followed by "CREATE src/app/material.module.ts (194 bytes)".
- CODE EDITOR:** Displays the content of material.module.ts. The code is as follows:

```

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

@NgModule({
  declarations: [],
  imports: [
    CommonModule
  ]
})
export class MaterialModule {}
```

Quitamos de app.module.ts:

The screenshot shows the VS Code interface with the following details:

- EXPLORADOR:** Shows the project structure under "MAPALOCALSTORAGE". It includes a .angular folder, a .vscode folder, a node\_modules folder, and a src folder containing an app folder. Inside the app folder are app.component.html, app.component.scss, app.module.ts, and material.module.ts.
- TERMINAL:** Shows the command: "ng g module material --flat" followed by "CREATE src/app/material.module.ts (194 bytes)".
- CODE EDITOR:** Displays the content of app.module.ts. The code is as follows:

```

import { AppComponent } from './app.component';
import { BrowserAnimationsModule } from '@angular/platform-browser/animations';

import { MatToolbarModule } from '@angular/material/toolbar';
import { MatButtonModule } from '@angular/material/button';
import { MatProgressBarModule } from '@angular/material/progress-bar';
```

Lo ponemos en el módulo generado:

```

TS material.module.ts U X   S styles.scss M   D app.component.html M   TS app.m...
appMapalocalStorage > src > app > TS material.module.ts > ...
1 import { NgModule } from '@angular/core';
2 import { CommonModule } from '@angular/common';
3
4 // Angular Material
5 import {MatToolbarModule} from '@angular/material/toolbar';
6 import {MatButtonModule} from '@angular/material/button';
7 import {MatProgressBarModule} from '@angular/material/progress-bar';
8
9 @NgModule({
10   declarations: [],
11   imports: [
12     CommonModule,
13     MatToolbarModule,
14     MatButtonModule,
15     MatProgressBarModule
16   ]
17 })

```

En app.module.ts debo importar el módulo de material:

```

TS app.module.ts M X   TS material.module.ts U   S styles.scss M   D app.component.html M
appMapalocalStorage > src > app > TS app.module.ts > ...
1 import { NgModule } from '@angular/core';
2 import { BrowserModule } from '@angular/platform-browser';
3
4 import { AppComponent } from './app.component';
5 import { BrowserAnimationsModule } from '@angular/platform-browser/animations';
6
7 import { MaterialModule } from './material.module';
8
9 @NgModule({
10   declarations: [
11     AppComponent
12   ],
13   imports: [
14     BrowserModule,
15     BrowserAnimationsModule,
16     MaterialModule

```

Sigue dando error:

```

✖ ▶ [webpack-dev-server] ERROR
src/app/app.component.html:1:1 - error NG8001: 'mat-toolbar' is not a known element:
1. If 'mat-toolbar' is an Angular component, then verify that it is part of this module.
2. If 'mat-toolbar' is a Web Component then add 'CUSTOM_ELEMENTS_SCHEMA' to the '@NgModule.schema'
1 <mat-toolbar color="primary">
~~~~~

```

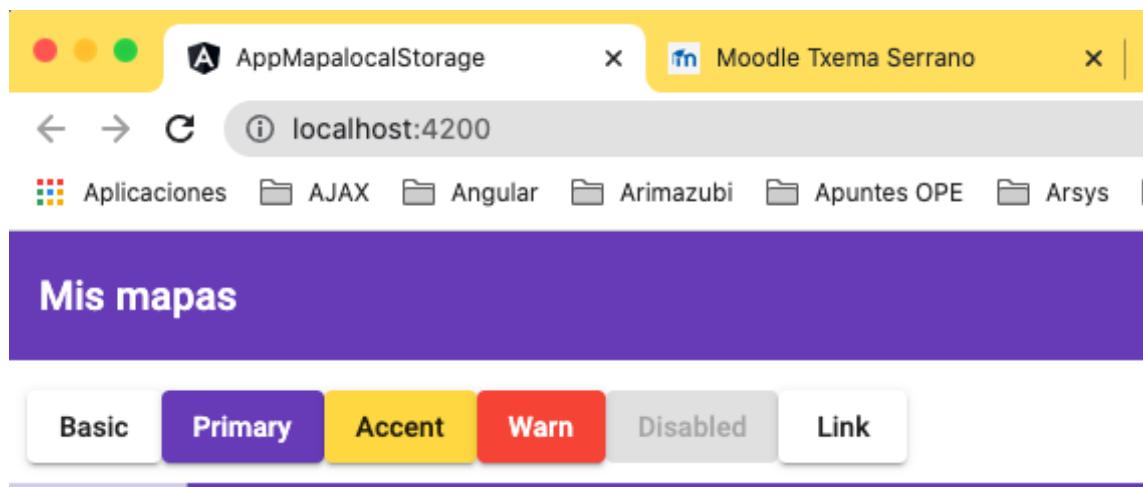
Esto es debido a que desde material no estamos exportando estos módulos para que puedan ser utilizados. Así en material.module.ts:

```

9  @NgModule({
10    declarations: [],
11    imports: [
12      CommonModule,
13      MatToolbarModule,
14      MatButtonModule,
15      MatProgressBarModule
16    ],
17    exports: [
18      MatToolbarModule,
19      MatButtonModule,
20      MatProgressBarModule
21    ]
22 })

```

Guardo los cambios:

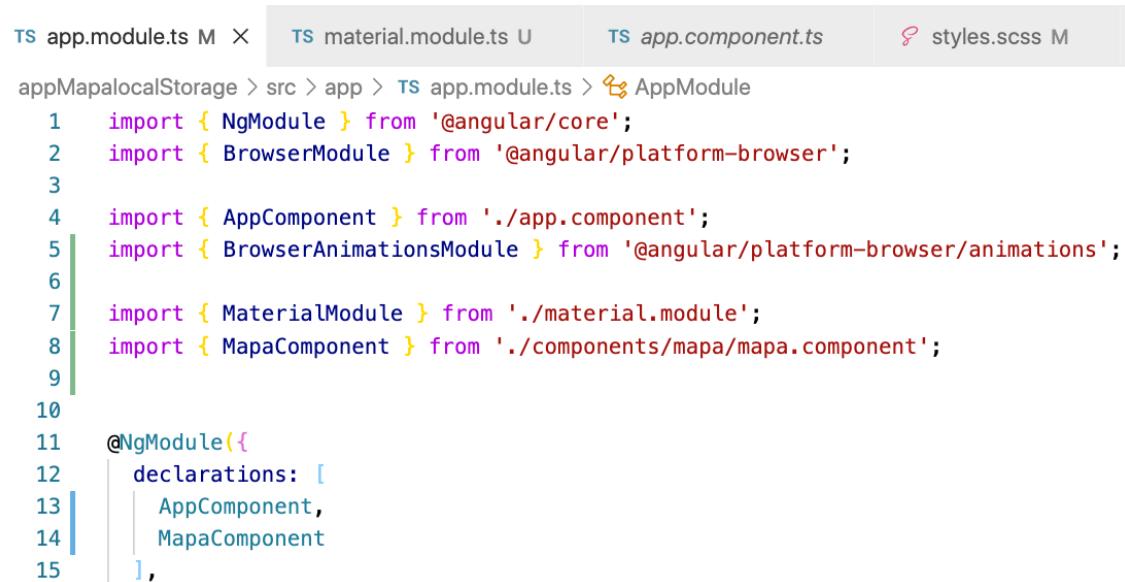


Ahora si necesito hacer importaciones de material, lo haré en material.module.ts

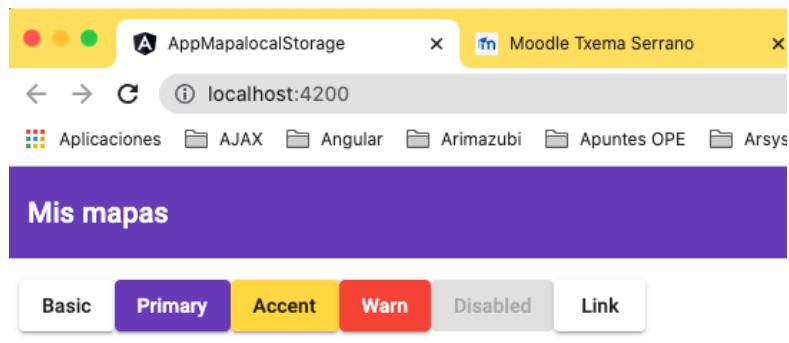
## 5.- Componente del mapa y su diseño.

```
jsersan@iMac-de-Jose appMapalocalStorage % ng g c components/mapa --module=app.module --skip-tests
CREATE src/app/components/mapa/mapa.component.scss (0 bytes)
CREATE src/app/components/mapa/mapa.component.html (19 bytes)
CREATE src/app/components/mapa/mapa.component.ts (268 bytes)
UPDATE src/app/app.module.ts (582 bytes)
```

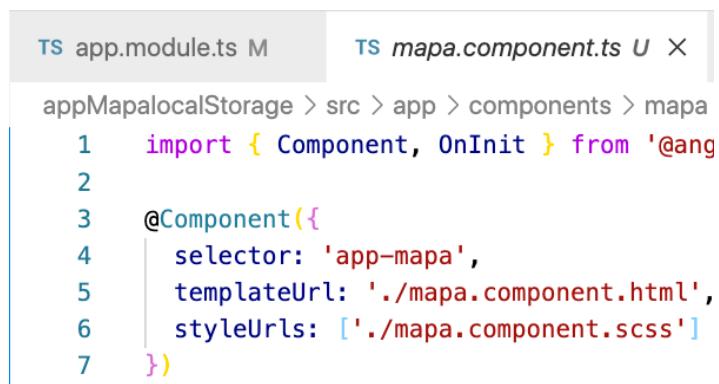
En app.module.ts:



```
TS app.module.ts M X TS material.module.ts U TS app.component.ts styles.scss M
appMapalocalStorage > src > app > TS app.module.ts > AppModule
1 import { NgModule } from '@angular/core';
2 import { BrowserModule } from '@angular/platform-browser';
3
4 import { AppComponent } from './app.component';
5 import { BrowserAnimationsModule } from '@angular/platform-browser/animations';
6
7 import { MaterialModule } from './material.module';
8 import { MapaComponent } from './components/mapa/mapa.component';
9
10
11 @NgModule({
12   declarations: [
13     AppComponent,
14     MapaComponent
15   ],
```



En mapa.component.ts vemos que el selector es app-mapa:



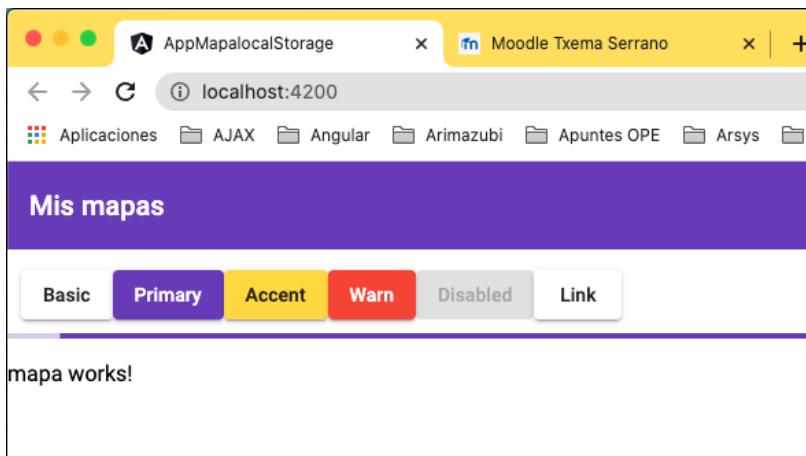
```
TS app.module.ts M TS mapa.component.ts U X
appMapalocalStorage > src > app > components > mapa
1 import { Component, OnInit } from '@angular/core'
2
3 @Component({
4   selector: 'app-mapa',
5   templateUrl: './mapa.component.html',
6   styleUrls: ['./mapa.component.scss']
7 })
```

Así en app.component.html:

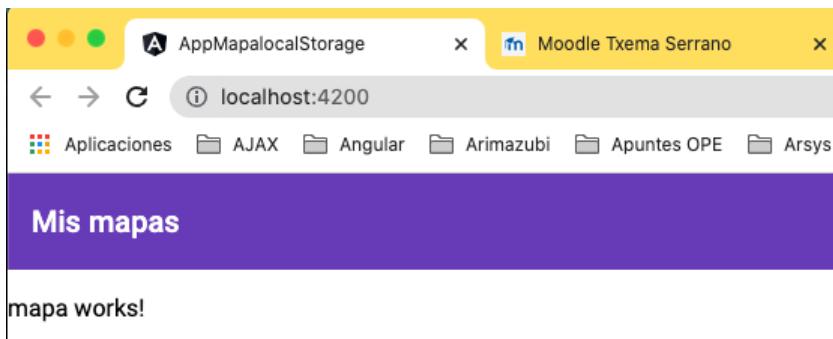
```

1 <mat-toolbar color="primary">
2   <span>Mis mapas</span>
3 </mat-toolbar>
4
5 <div class="main-container">
6   <button mat-raised-button>Basic</button>
7   <button mat-raised-button color="primary">Primary</button>
8   <button mat-raised-button color="accent">Accent</button>
9   <button mat-raised-button color="warn">Warn</button>
10  <button mat-raised-button disabled>Disabled</button>
11  <a mat-raised-button href="https://www.google.com/" target="_blank">Link</a>
12 </div>
13
14 <mat-progress-bar mode="indeterminate"></mat-progress-bar>
15
16 <app-mapa></app-mapa>
--
```

Resultado;



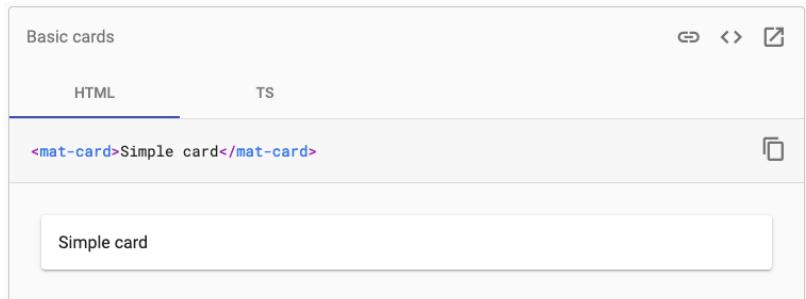
Quitamos los botones:



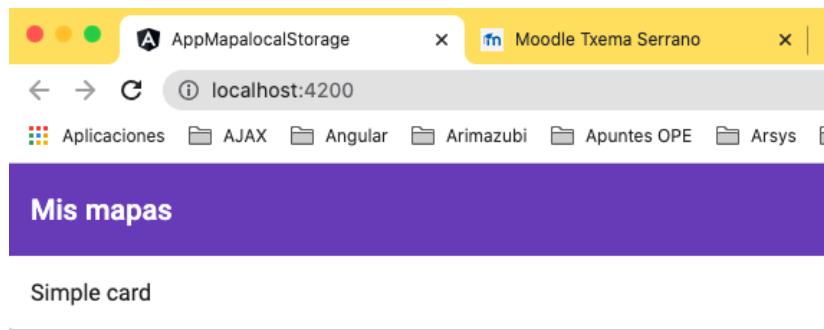
Vamos a insertar una tarjeta simple card. Recuerden que hay que importarla:

```
appMapalocalStorage > src > app > TS material.module.ts > MaterialModule
  1  import { NgModule } from '@angular/core';
  2  import { CommonModule } from '@angular/common';
  3
  4  // Angular Material
  5  import {MatToolbarModule} from '@angular/material/toolbar';
  6  import {MatButtonModule} from '@angular/material/button';
  7  import {MatProgressBarModule} from '@angular/material/progress-bar';
  8  import {MatCardModule} from '@angular/material/card';
  9
 10 @NgModule({
 11   declarations: [],
 12   imports: [
 13     CommonModule,
 14     (alias) class MatProgressBarModule
 15     import MatProgressBarModule
 16     MatProgressBarModule,
 17     MatCardModule
 18   ],
 19   exports: [
 20     MatToolbarModule,
 21     MatButtonModule,
 22     MatProgressBarModule,
 23     MatCardModule
```

Copiamos el código en mapa.component.html:



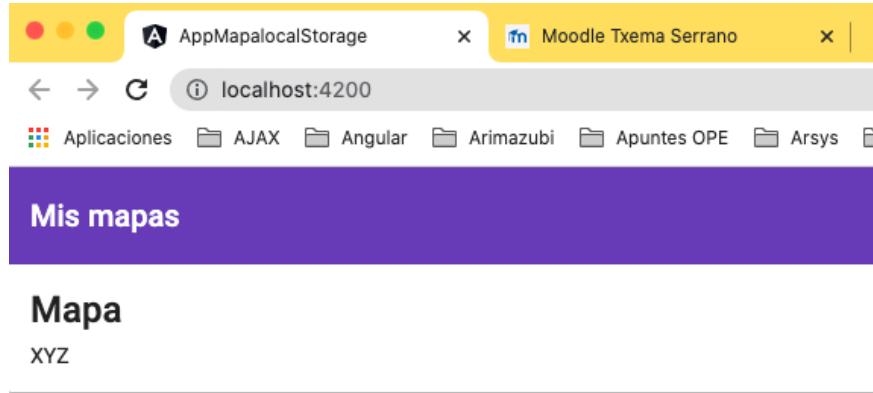
Resultado:



Lo personalizamos:

```
mapa.component.html U  TS material.module.ts U  
appMapalocalStorage > src > app > components > mapa > <br>  
1   <mat-card>  
2  
3       <mat-card-title>Mapa</mat-card-title>  
4  
5       <mat-card-content>  
6           XYZ  
7       </mat-card-content>  
8   </mat-card>
```

Resultado:



## 6.- Mostrando un mapa de Google Maps.



En Getting Started:

## Setting up Angular Google Maps

### Install Angular Google Maps

Angular Google Maps (short name: AGM) gets shipped via the Node Package Manager (NPM). Run the following command to add it to your new project:

```
npm install @agm/core
```

Tras instalarlo:

```
jsersan@iMac-de-Jose mapasLocalStorage % npm install @agm/core
npm WARN deprecated @types/googlemaps@3.43.3: Types for the Google Maps browser API have moved to @types/google.maps. Note: these types are not for the googlemaps npm package, which is a Node API.

added 8 packages, and audited 9 packages in 5s

found 0 vulnerabilities
jsersan@iMac-de-Jose mapasLocalStorage %
```

En app.module.ts:

```
10 import { AgmCoreModule } from '@agm/core';
11
12 @NgModule({
13   declarations: [
14     AppComponent,
15     MapaComponent
16   ],
17   imports: [
18     BrowserModule,
19     BrowserAnimationsModule,
20     MaterialModule,
21     AgmCoreModule
22   ],
23 }
```

En la misma página está el como se utiliza:

```
@NgModule({
  imports: [
    BrowserModule,
    AgmCoreModule.forRoot({
      apiKey: ''
    })
  ],
  providers: [],
  declarations: [ AppComponent ],
  bootstrap: [ AppComponent ]
})
```

Entonces, tras poner una apiKey válida :

```
@NgModule({
  declarations: [
    AppComponent,
    MapaComponent
  ],
  imports: [
    BrowserModule,
    BrowserAnimationsModule,
    MaterialModule,
    AgmCoreModule.forRoot({
      apiKey: 'AIzaSyDY3YQ6aNb7YXoz13RIxq5fVyzdbak7sQ0'
    })
  ],
})
```

En la misma página está la documentación del uso del mapa:

### Extending the app component

Angular CLI already created an app component the we'll now use to create our first google map. Open the file `src/app/app.component.ts` and modify it like below:

```
import { Component } from '@angular/core';

@Component({
  selector: 'app-root',
  templateUrl: 'app.component.html',
  styleUrls: ['app.component.css'],
})
export class AppComponent {
  title = 'My first AGM project';
  lat = 51.678418;
  lng = 7.809007;
}
```

Ahora trabajaremos con el componente mapa. En mapa.component.ts, colocamos la longitud y la latitud:

```

8  export class MapaComponent implements OnInit {
9
10    lat = 51.678418;
11    lng = 7.809007;
12
13    constructor() { }
14
15    ngOnInit(): void {
16    }

```

En mapa.component.html, según la documentación:

## Setup the template

Open the file `src/app/app.component.html` and paste the following content:

```

<h1>{{ title }}</h1>

<!-- this creates a google map on the page with the given lat/lng from -->
<!-- the component as the initial center of the map: -->
<agm-map [latitude]="lat" [longitude]="lng">
  <agm-marker [latitude]="lat" [longitude]="lng"></agm-marker>
</agm-map>

```

Así, en nuestro mapa.component.html lo copio donde pone XYZ:



The screenshot shows a code editor with the following details:

- File Structure:** The sidebar shows the project structure: `src > app > components > mapa > mapa.component.html`.
- Code Editor:** The main area displays the contents of `mapa.component.html`. The copied code from the documentation is pasted into the template section of the mat-card.
- Code Content:**

```

1  <mat-card>
2
3    <mat-card-title>Mapa</mat-card-title>
4
5    <mat-card-content>
6      <agm-map [latitude]="lat" [longitude]="lng">
7        <agm-marker [latitude]="lat" [longitude]="lng"></agm-marker>
8      </agm-map>
9    </mat-card-content>
10   </mat-card>

```

Si alguno les sale este error:

```
Build at: 2022-01-20T11:27:38.658Z - Hash: c78d5bfa4fd66ef - Time: 237ms
Error: node_modules/@agm/core/lib/services/google-maps-api-wrapper.d.ts:51:41 - error TS2314: Generic type 'Ma
51 subscribeToMapEvent<N extends keyof google.maps.MapHandlerMap>(eventName: N): Observable<google.maps.M
Error: node_modules/@agm/core/lib/services/google-maps-api-wrapper.d.ts:51:94 - error TS2314: Generic type 'Ma
51 subscribeToMapEvent<N extends keyof google.maps.MapHandlerMap>(eventName: N): Observable<google.maps.M
Error: node_modules/@agm/core/lib/services/managers.marker-manager.d.ts:26:93 - error TS2694: Namespace 'googl
26 createEventObservable<T extends (google.maps.MouseEvent | void)>(eventName: google.maps.MarkerMouseEver
Error: node_modules/@agm/core/lib/services/managers.marker-manager.d.ts:26:129 - error TS2694: Namespace 'googl
26 createEventObservable<T extends (google.maps.MouseEvent | void)>(eventName: google.maps.MarkerMouseEver
```

Es por la versión de Angular Maps. Debemos volver a una versión anterior:

```
$ npm i @types/googlemaps@3.39.12
```

Resultado:

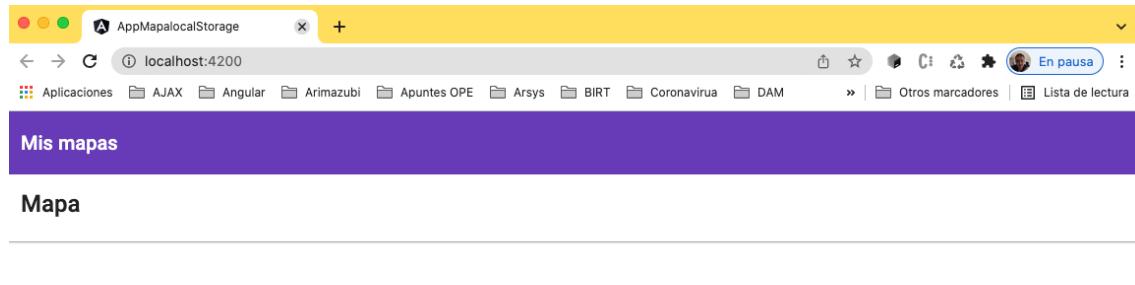
```
jsersan@iMac-de-Jose appMapalocalStorage % npm install @types/googlemaps@3.39.12 --save-dev
npm WARN deprecated @types/googlemaps@3.39.12: Types for the Google Maps browser API have moved to @t
removed 1 package, changed 1 package, and audited 1020 packages in 4s
91 packages are looking for funding
  run `npm fund` for details
40 vulnerabilities (3 low, 37 moderate)

To address issues that do not require attention, run:
  npm audit fix

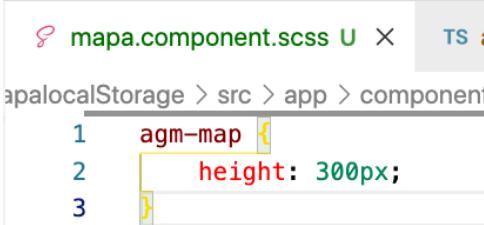
To address all issues (including breaking changes), run:
  npm audit fix --force

Run `npm audit` for details.
```

El proyecto:



Para poder ver el mapa debemos modificar el ancho del mismo en el fichero css:

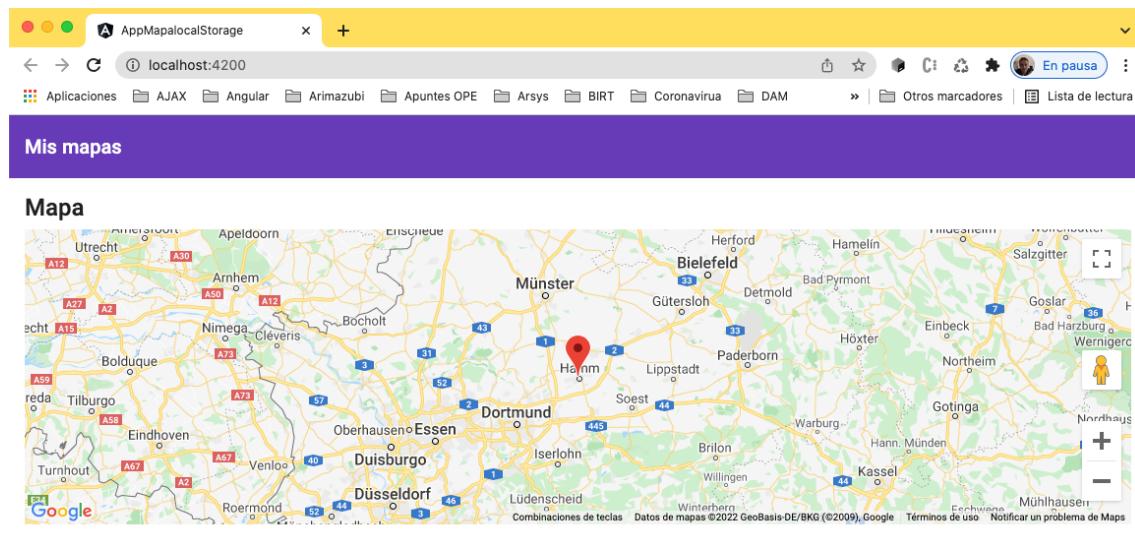


```

1 agm-map {
2   height: 300px;
3 }

```

El resultado es:



Algún punto en Alemania.

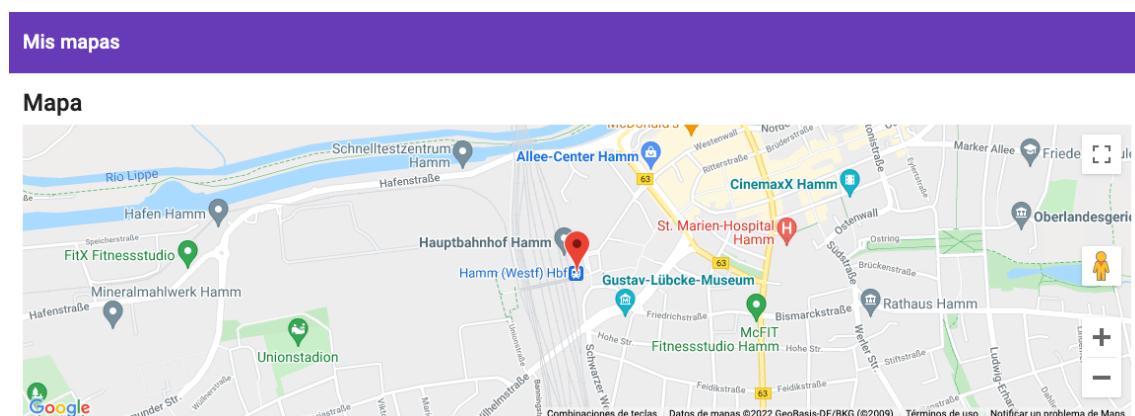
Le especificamos en mapa.component.html un zoom de 15:

```

5 <mat-card-content>
6   <agm-map [latitude]="lat" [longitude]="lng" [zoom]="15">
7     <agm-marker [latitude]="lat" [longitude]="lng"></agm-marker>
8   </agm-map>
9 </mat-card-content>

```

Resultado:



Ahora falta que pongamos unos marcadores por lo que miramos la documentación:

Comprobamos el componente AgmInforWindow:

Colocamos el componente AgmInfoWindow dentro de AgmMarker:

```
<mat-card-content>
  <agm-map [latitude]="lat" [longitude]="lng" [zoom]="15">
    <agm-marker [latitude]="lat" [longitude]="lng">

      <agm-info-window>
        <strong>Título</strong>

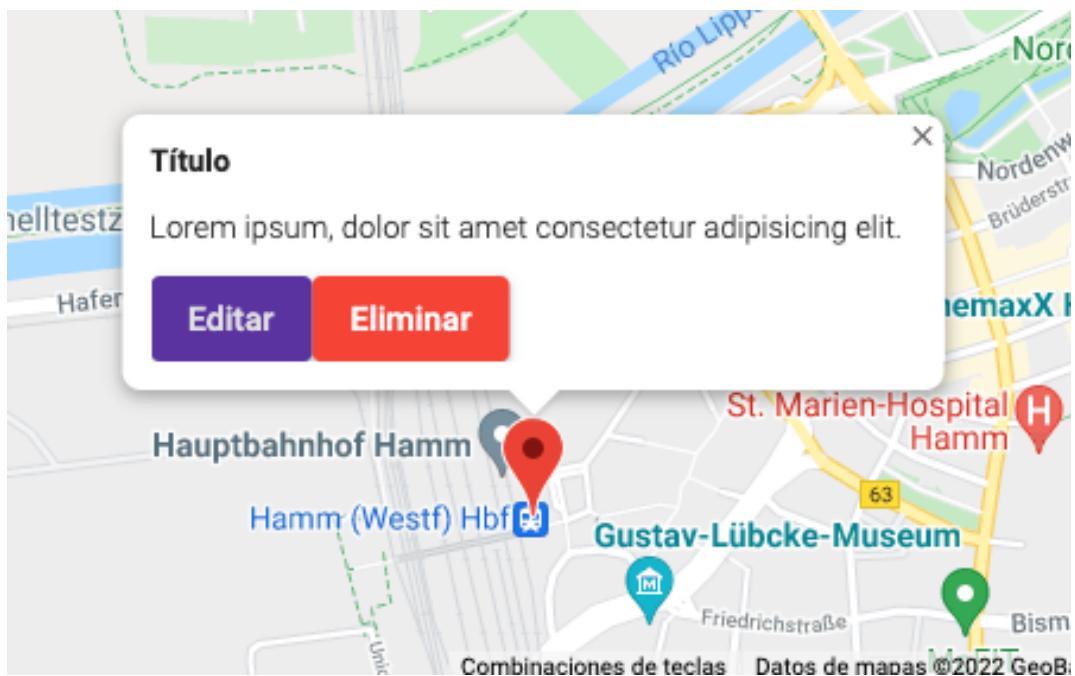
        <p>Lorem ipsum, dolor sit amet consectetur adipisicing elit. Incidu</p>

        <div>
          <button mat-raised-button color="primary">Editar</button>
          <button mat-raised-button color="warn">Eliminar</button>
        </div>

      </agm-info-window>

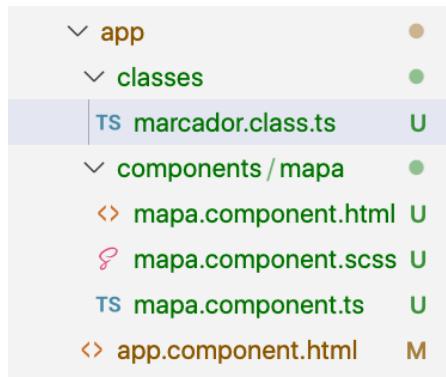
    </agm-marker>
  </agm-map>
</mat-card-content>
```

En funcionamiento:



## 7.- Creando marcadores de forma dinámica.

Para el manejo de los marcadores nos creamos una clase marcador.class.ts.



El contenido de la clase es:

```

3   export class Marcador {
4
5     public lat: number;
6     public lng: number;
7
8     public titulo = "Sin título";
9     public desc = "Sin descripción";
10
11    constructor (lat: number, lng: number){
12      this.lat = lat;
13      this.lng = lng;
14    }
15
16  }

```

Para poder utilizar la clase en otros archivos debe estar exportada. Ahora creamos un array de marcadores en mapa.componrnt.ts:

```

2   import { Marcador } from '../classes/marcador.class';
3
4   @Component({
5     selector: 'app-mapa',
6     templateUrl: './mapa.component.html',
7     styleUrls: ['./mapa.component.scss']
8   })
9   export class MapaComponent implements OnInit {
10
11   marcadores: Marcador[] = [];
12
13   lat = 51.678418;
14   lon = -7.800007;

```

Queremos que en el constructor se cree el primer marcador de forma dinámica:

```

15
16  constructor() {
17
18    const nuevoMarcador = new Marcador(51.678418, 7.809007);
19
20    this.marcadores.push(nuevoMarcador);
21
22 }

```

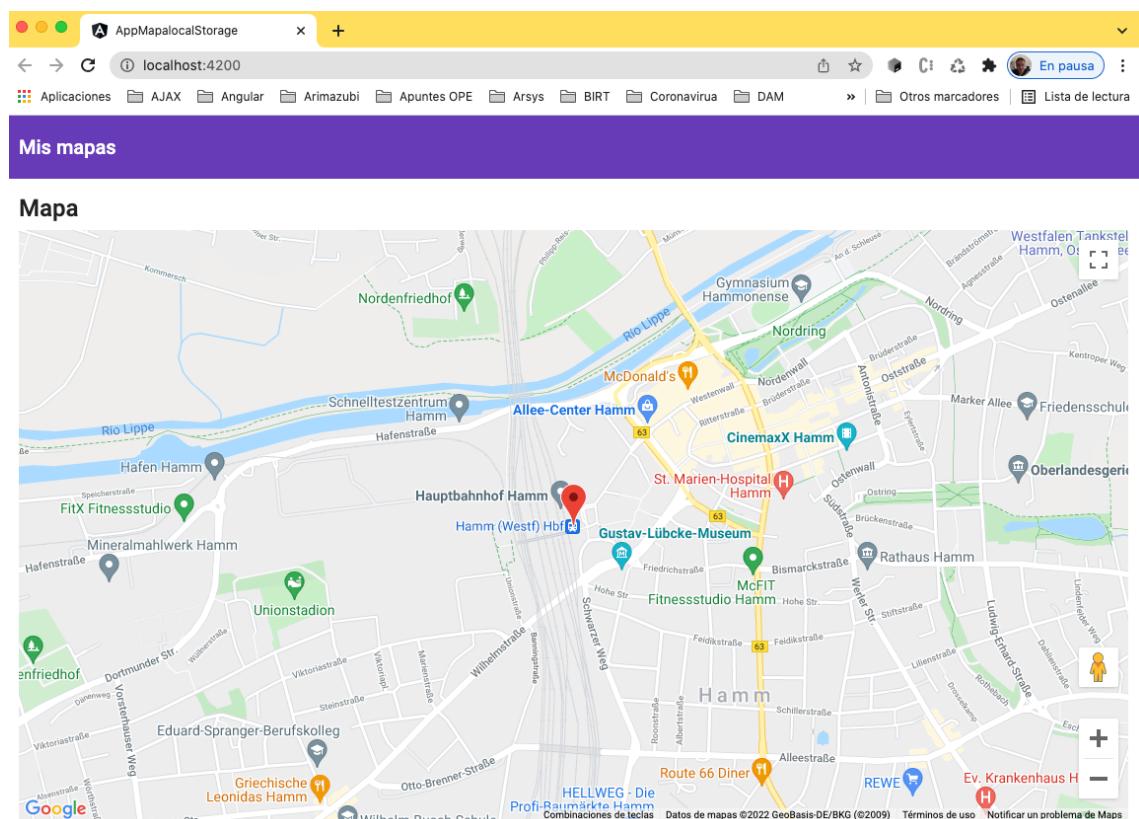
En mapa.component.html:

```

5   <mat-card-content>
6     <agm-map [latitude]="lat" [longitude]="lng" [zoom]="15">
7       <agm-marker *ngFor="let marcador of marcadores"
8         [latitude]="marcador.lat"
9         [longitude]="marcador.lng">
10

```

El resultado es:



El cargador se está cargando de forma dinámica a partir del constructor.

Lo personalizamos:

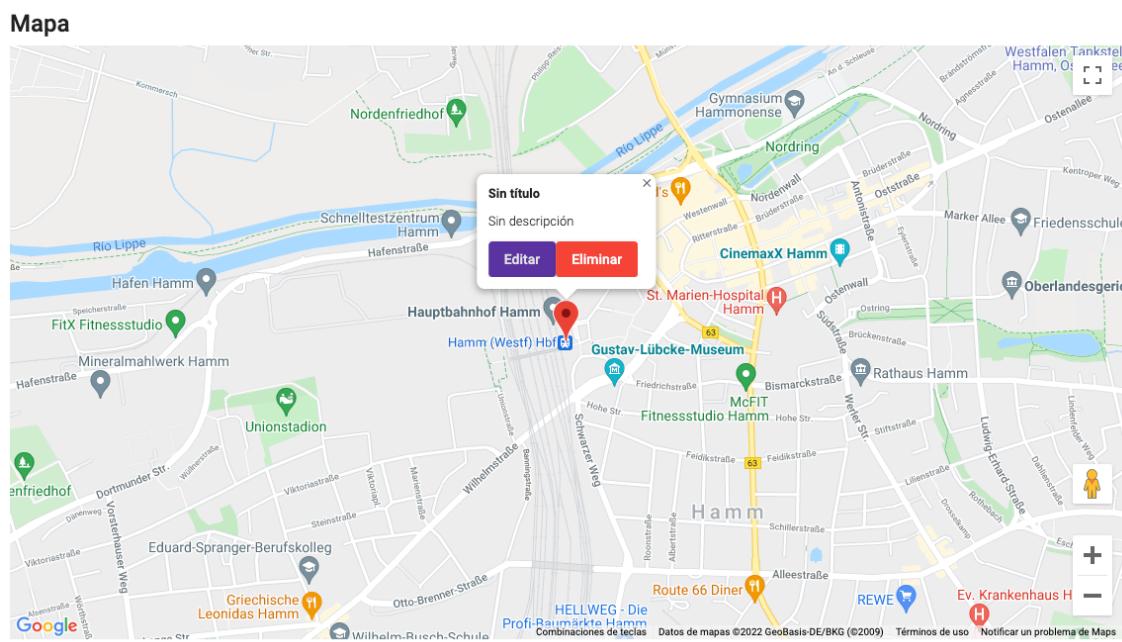
```
<agm-info-window>
  <strong>{{ marcador. titulo }}</strong>

  <p> {{ marcador.desc }}</p>

  <div>
    <button mat-raised-button color="primary">Editar</button>
    <button mat-raised-button color="warn">Eliminar</button>
  </div>

</agm-info-window>
```

Resultado:



Para crear un nuevo evento de forma dinámica. Buscamos en el @agm/core. Trabajamos con el evento mapClick:

## Outputs

boundsChange	mapDoubleClick
centerChange	mapReady
idle	mapRightClick
mapClick	mapTypeidChange

Así en map.component.html agregamos el evento:

```

5   <mat-card-content>
6     <agm-map (mapClick)="agregarMarcador($event)"
7       [latitude]="lat" [longitude]="lng" [zoom]="15">
8       <agm-marker *ngFor="let marcador of marcadores"
9         [latitude]="marcador.lat"
10        [longitude]="marcador.lng">
11
12       <agm-info-window>
13         <strong>{{ marcador. titulo }}</strong>

```

La función agregarMarcador debemos crearla en el mapa.component.ts.

```

24   agregarMarcador(evento:any){
25
26     console.log(evento);
27
28 }

```

En el caso que no se muestre bien en la consola el valor del marcador, debemos actualizar el agm.

```
$ npm uninstall @agm/core
$ npm install @agm/core@1.1.0 --force
```

Así cuando hago click en cualquier punto:

```

▼{coords: {...}, placeId: undefined} ⓘ
  ▼coords:
    lat: 51.67788578257858
    lng: 7.802548240692158
    ► [[Prototype]]: Object
    placeId: undefined
    ► [[Prototype]]: Object

```

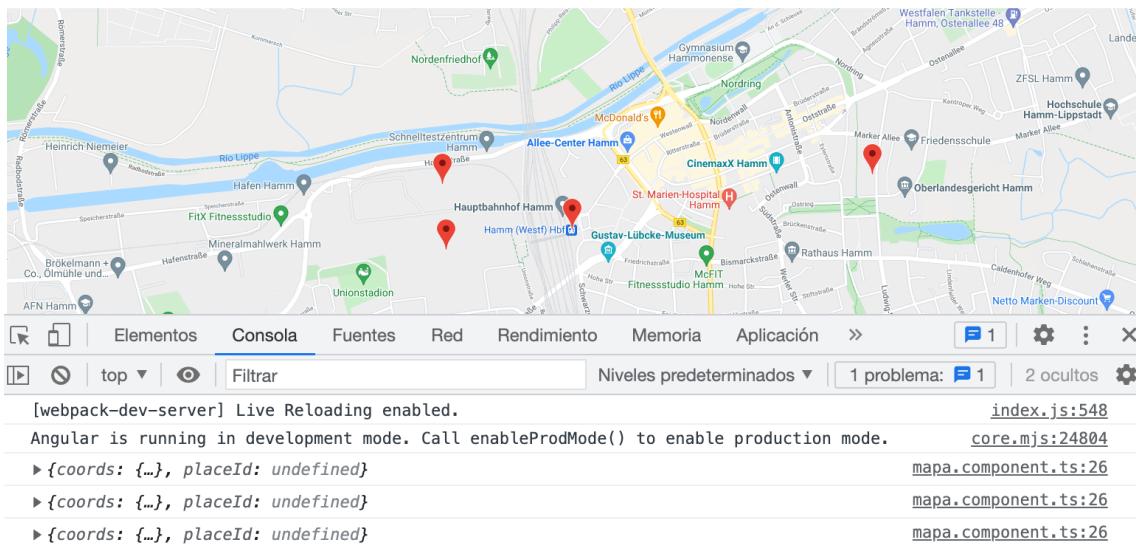
Hay que agregar este punto al array de marcadores en la función agregarMarcador como está en el constructor:

```

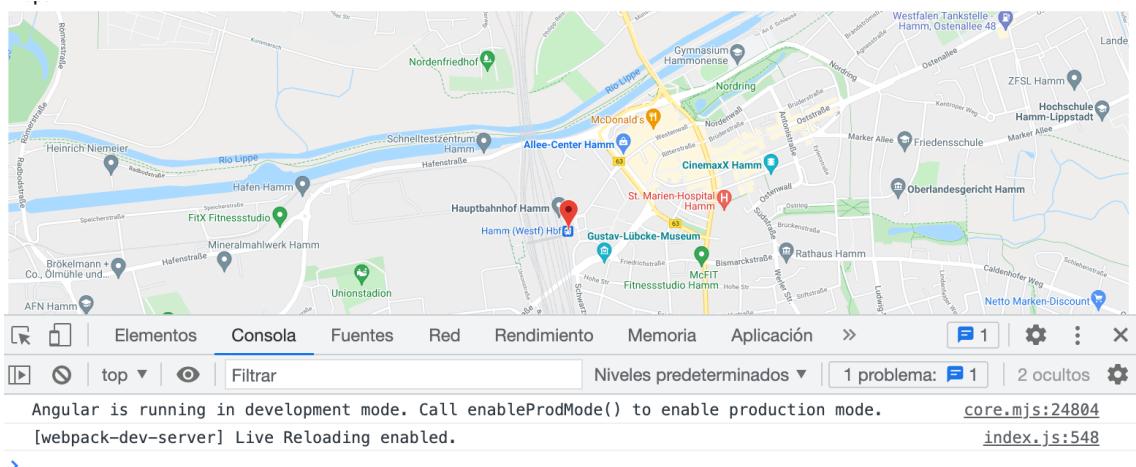
24   agregarMarcador(evento:any){
25
26     console.log(evento);
27
28     const nuevoMarcador = new Marcador(evento.coords.lat, evento.coords.lng);
29
30     this.marcadores.push(nuevoMarcador);
31
32 }

```

El resultado es:



Si refrescamos lo perdemos porque todavía no he utilizado el localStorage.



Agregamos un poco el código:

```
24 agregarMarcador(evento:any){  
25  
26     console.log(evento);  
27  
28     const coords: {lat:number, lng:number} = evento.coords;  
29  
30     const nuevoMarcador = new Marcador(coords.lat, coords.lng);  
31  
32     this.marcadores.push(nuevoMarcador);  
33  
34 }
```

## 8.- Guardando los marcadores en el localStorage.

En el localStorage sólo se pueden guardar strings.

```
36  guardarStorage(){
37
38      localStorage.setItem('marcadores', this.marcadores);
39
40 }
```

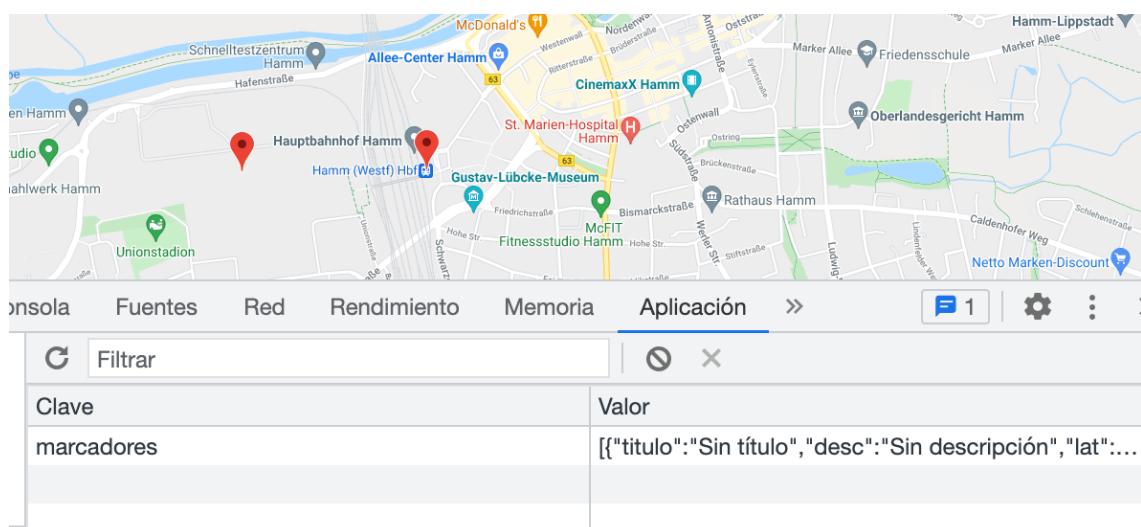
Como vemos, al intentar guardar una array de marcadores da error. Debemos convertir el objeto en una string:

```
36  guardarStorage(){
37
38      localStorage.setItem('marcadores', JSON.stringify(this.marcadores));
39
40 }
```

Ahora debemos llamar a esta función desde agregarMarcador:

```
24  agregarMarcador(evento:any){
25
26      console.log(evento);
27
28      const coords: {lat:number, lng:number} = evento.coords;
29
30      const nuevoMarcador = new Marcador(coords.lat, coords.lng);
31
32      this.marcadores.push(nuevoMarcador);
33
34      this.guardarStorage();
35
36 }
```

Al pulsar en el mapa:



Si lo examinamos de cerca:

Clave	Valor
marcadores	[{"titulo": "Sin título", "desc": "Sin descripción", "lat": ..., "lng": ...}]
▼ [{titulo: "Sin título", desc: "Sin descripción", lat: 51.678418, lng: 7.809007}, ...]	
► 0: {titulo: "Sin título", desc: "Sin descripción", lat: 51.678418, lng: 7.809007}	
► 1: {titulo: "Sin título", desc: "Sin descripción", lat: 51.67836477853932, lng: 7.8007}	
► 2: {titulo: "Sin título", desc: "Sin descripción", lat: 51.67881715896127, lng: 7.8262}	

Como el punto original sí está en el array. Lo podemos quitar del constructor porque ya está en el localStorage que se nos queda vacío:

```

16   constructor() {
17
18
19 }
```

Ahora debemos preguntar si existe el localStorage el arreglo marcadores para cargarlo:

```

16   constructor() {
17
18     if(localStorage.getItem("marcadores")){
19       this.marcadores = JSON.parse(localStorage.getItem("marcadores"));
20     }
21
22
23 }
```

Pero no podemos cargarlo así porque el objeto del localStorage es un objeto JSON que debemos parsear:

```
Error: src/app/components/mapa/mapa.component.ts:19:36 - error TS2345: Argument type 'null' is not assignable to type 'string'.
19   this.marcadores = JSON.parse(localStorage.getItem("marcadores"));
                                         ~~~~~~
```

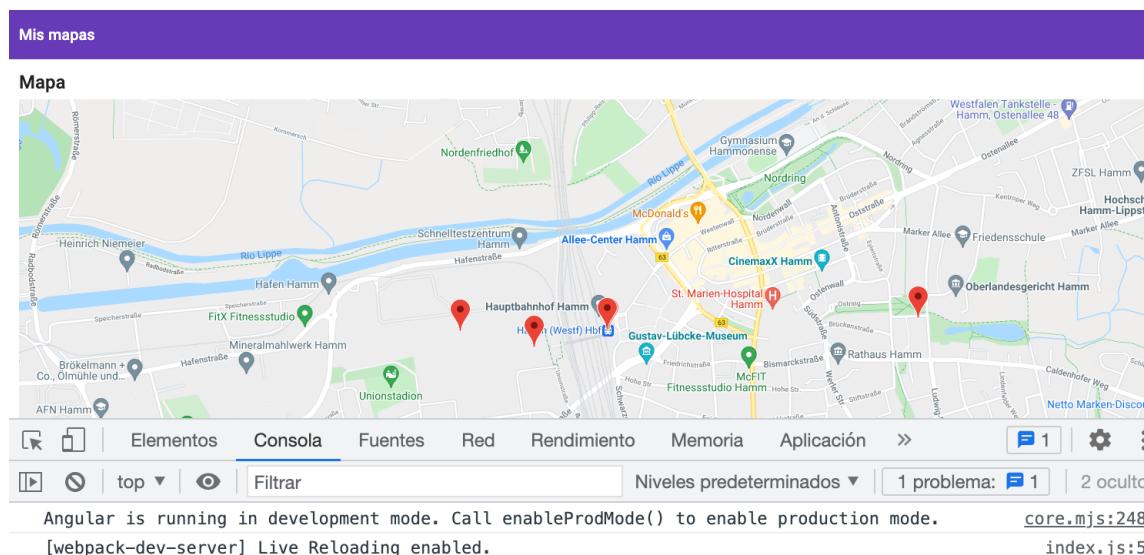
Nos sale este error porque la función getItem puede devolver una string o null (ahí es donde está el problema). Para solucionarlo, en el componente hacemos este cambio:

```

16  constructor() {
17
18    if(localStorage.getItem("marcadores")){
19      this.marcadores = JSON.parse(localStorage.getItem("marcadores") || '{}');
20
21
22 }

```

Resultado:



Otra forma de hacerlo:

```

16  constructor() {
17
18    if(localStorage.getItem("marcadores")){
19      this.marcadores = JSON.parse(localStorage.getItem("marcadores")!);
20
21
22 }

```

## 9.- Borrar marcadores.

Lo primero es agregar el evento al botón borrar en mapa.component.html

```
<agm-marker *ngFor = "let marcador of marcadores; let i = index"
            [latitude]="marcador.lat"
            [longitude]="marcador.lng">

    <agm-info-window>
        <strong>{{ marcador.titulo }}</strong>
        <p>{{ marcador.desc }}</p>

    <div>
        <button mat-raised-button color="primary">Editar</button>
        <button (click)="borrarMarcador(i)">
```

Asociamos a cada marcador un identificador de esta forma:

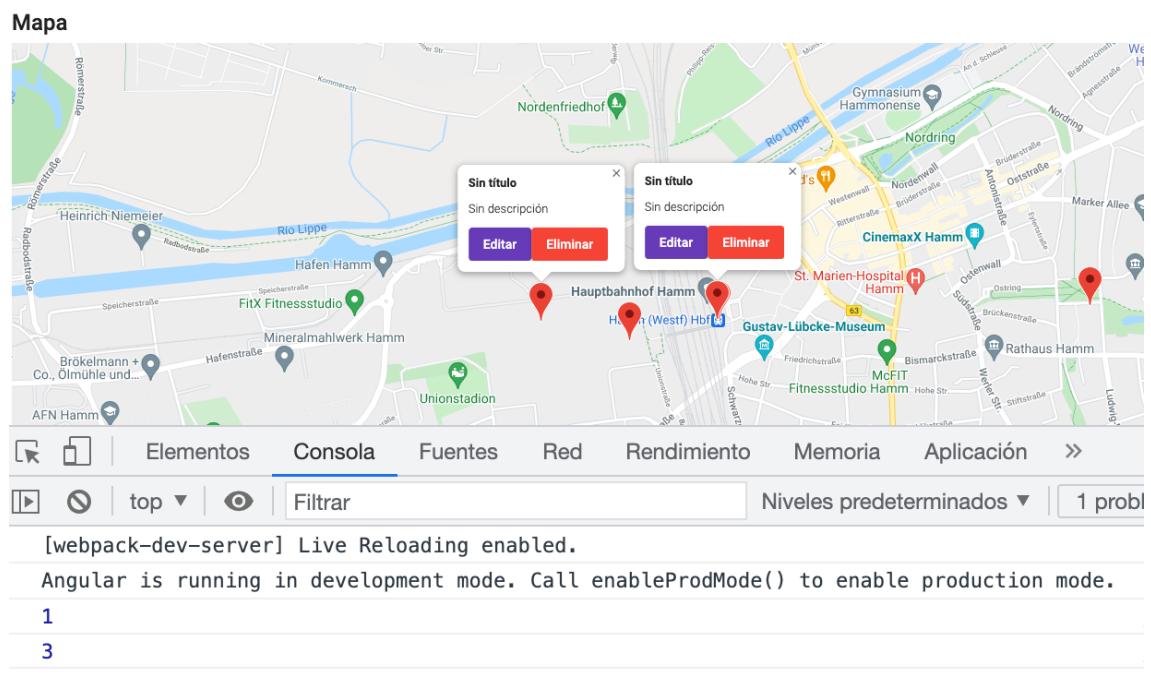
```
*ngFor = "let marcador of marcadores; let i= index"
```

De esta forma i va a ser el número que ocupa cada uno de los elementos dentro del array.

Implementemos esta función en mapa.component.ts:

```
44     | borrarMarcador(i:any){
45     |     console.log(i);
46 }
```

Probamos. Pulsamos en los marcadores y le damos al botón eliminar:



Ahora, sabiendo cuál es:

```
44  |  borrarMarcador(i:number){  
45  |  |  this.marcadores.splice(i,1);  
46  |  }  
|
```

Esto me permite borrarlo del mapa pero no del localStorage, por lo que si recargo la página sigue en el mismo sitio. Para ello simplemente debemos guardar marcadores en el localStorage:

```
44  |  borrarMarcador(i:number){  
45  |  |  this.marcadores.splice(i,1);  
46  |  |  this.guardarStorage();  
47  |  }  
|
```

## 10. Mensajes snackBar:

En la página de material.angular.io:

**Opening a snackbar**

A snackbar can contain either a string message or a given component.

```
// Simple message.
let snackBarRef = snackBar.open('Message archived');

// Simple message with an action.
let snackBarRef = snackBar.open('Message archived', 'Undo');

// Load the given component into the snackbar.
let snackBarRef = snackbar.openFromComponent(MessageArchivedComponent);
```

In either case, a `MatSnackBarRef` is returned. This can be used to dismiss the snackbar or to receive notification of when the snackbar is dismissed. For simple messages with an action, the `MatSnackBarRef` exposes an observable for when the action is triggered. If you want to close a custom snackbar that was opened via `openFromComponent`, from within the component itself, you can inject the `MatSnackBarRef`.

```
snackBarRef.afterDismissed().subscribe(() => {
  console.log('The snackbar was dismissed');
});

snackBarRef.onAction().subscribe(() => {
  console.log('The action was triggered');
});
```

Así la función a llamar sería la segunda:

```
// Simple message with an action.
let snackBarRef = snackBar.open('Message archived', 'Undo');
```

Copiamos esta línea y la ponemos en la función agregarMarcador()

```
38  guardarStorage(){
39
40    localStorage.setItem('marcadores', JSON.stringify(this.marcadores));
41    snackBar.open('Marcador agregado', 'Cerrar');
42  }
43
44  borrarMarcador(i:number){
45    this.marcadores.splice(i,1);
46    this.guardarStorage();
47    snackBar.open('Marcador eliminado', 'Cerrar');
48  }
```

Para quietar los errores debemos importar la librería en mapa.component.ts.  
Examinamos los ejemplos:

```

import {Component} from '@angular/core';
import {MatSnackBar} from '@angular/material/snack-bar';

/**
 * @title Snack-bar with a custom component
 */
@Component({
  selector: 'snack-bar-component-example',
  templateUrl: 'snack-bar-component-example.html',
  styleUrls: ['snack-bar-component-example.css'],
})
export class SnackBarComponentExample {
  durationInSeconds = 5;

  constructor(private _snackBar: MatSnackBar) {}

  openSnackBar() {
    this._snackBar.openFromComponent(PizzaPartyComponent, {
      duration: this.durationInSeconds * 1000,
    });
  }
}

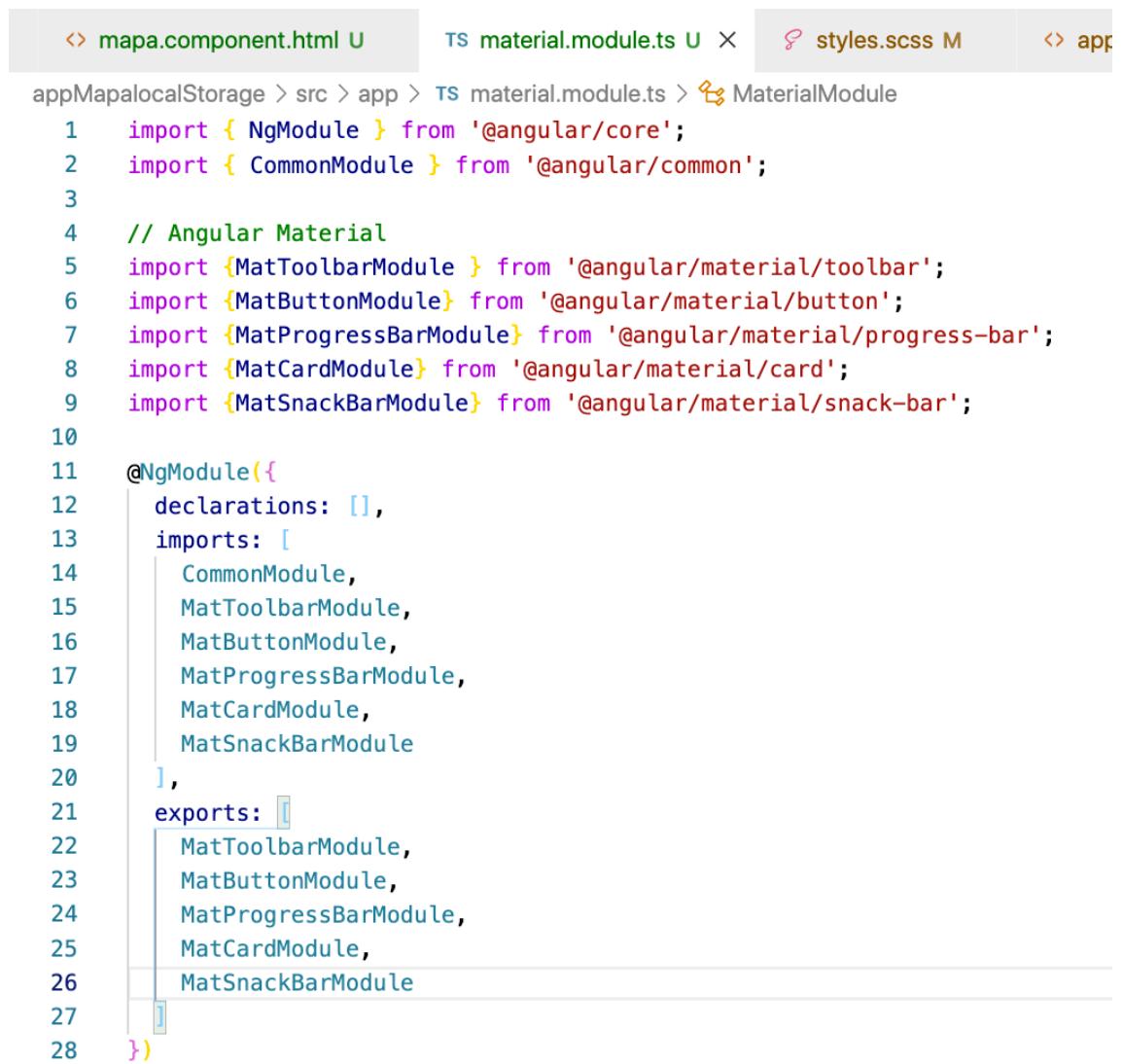
```

Ahora en mapa.component.ts:

```

27  constructor(private snackBar: MatSnackBar) {
28
29    if(localStorage.getItem("marcadores")){
30      this.marcadores = JSON.parse(localStorage.getItem("marcadores")!);
31    }
32  }
33
34  guardarStorage(){
35
36    localStorage.setItem('marcadores', JSON.stringify(this.marcadores));
37    this.snackBar.open('Marcador agregado', 'Cerrar');
38  }
39
40  borrarMarcador(i:number){
41    this.marcadores.splice(i,1);
42    this.guardarStorage();
43    this.snackBar.open('Marcador eliminado', 'Cerrar');
44  }
--
```

Pero debo importar el módulo snackBar en material.module.ts:

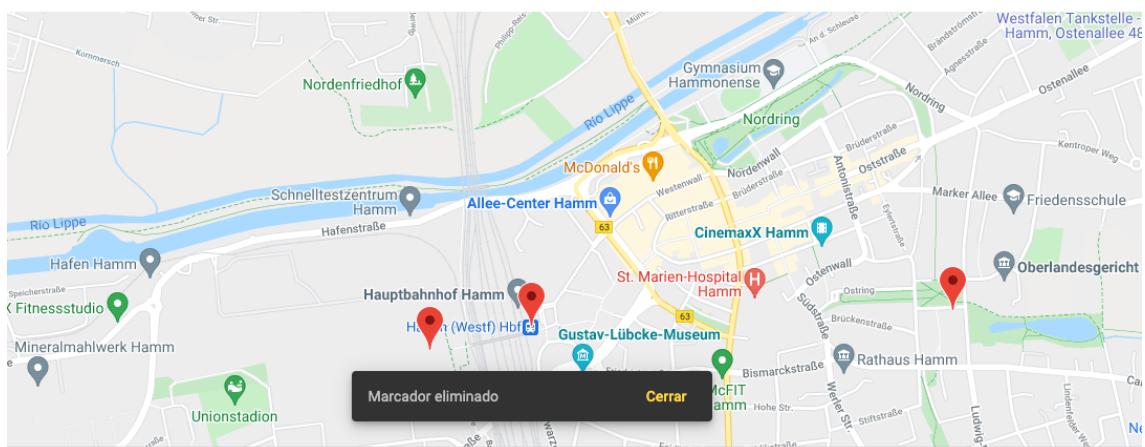


```

<> mapa.component.html U      TS material.module.ts U X   ⚡ styles.scss M   <> app
appMapalocalStorage > src > app > TS material.module.ts > 🛡 MaterialModule
1  import { NgModule } from '@angular/core';
2  import { CommonModule } from '@angular/common';
3
4  // Angular Material
5  import {MatToolbarModule} from '@angular/material/toolbar';
6  import {MatButtonModule} from '@angular/material/button';
7  import {MatProgressBarModule} from '@angular/material/progress-bar';
8  import {MatCardModule} from '@angular/material/card';
9  import {MatSnackBarModule} from '@angular/material/snack-bar';
10
11 @NgModule({
12   declarations: [],
13   imports: [
14     CommonModule,
15     MatToolbarModule,
16     MatButtonModule,
17     MatProgressBarModule,
18     MatCardModule,
19     MatSnackBarModule
20   ],
21   exports: [
22     MatToolbarModule,
23     MatButtonModule,
24     MatProgressBarModule,
25     MatCardModule,
26     MatSnackBarModule
27 ]
28 })

```

Probamos:



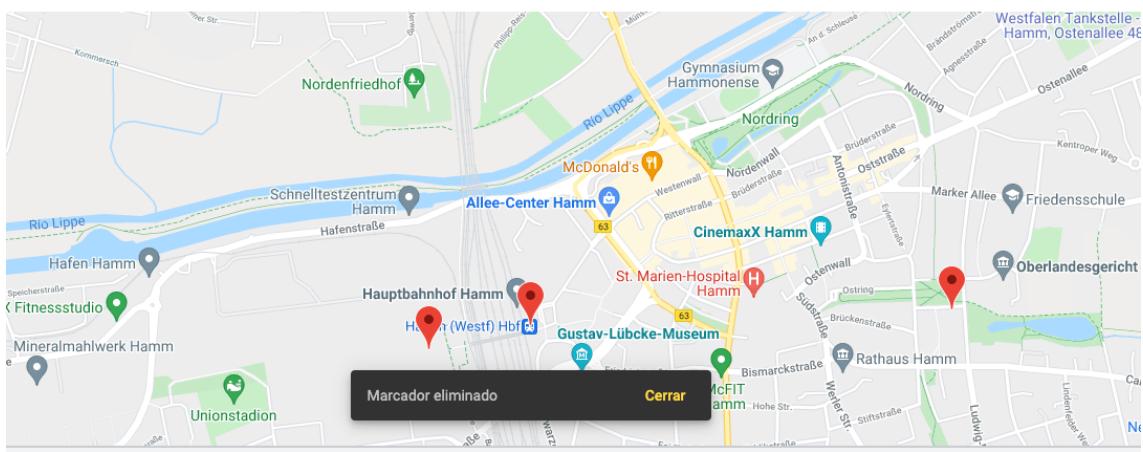
Tal y como está configurado, el snackBar sólo se cerrará si le damos al botón cerrar. Le ponemos un temporizador:

```

34  guardarStorage(){
35
36      localStorage.setItem('marcadores', JSON.stringify(this.marcadores));
37      this.snackBar.open('Marcador agregado', 'Cerrar', {duration: 3000 });
38  }
39
40  borrarMarcador(i:number){
41      this.marcadores.splice(i,1);
42      this.guardarStorage();
43      this.snackBar.open('Marcador eliminado', 'Cerrar', {duration: 3000 });
44  }

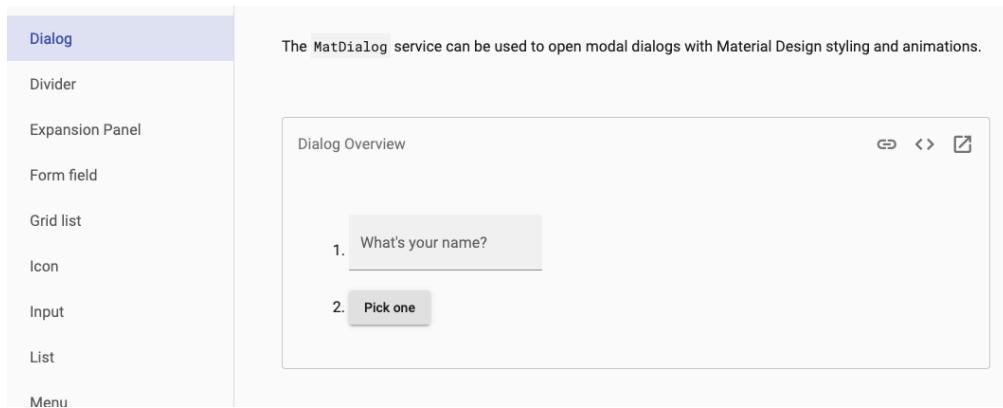
```

Probamos y comprobamos como después de tres segundo desaparece el mensaje.



## 11. Dialog. Mostrar un modal para la edición del marcador.

En material.angular.io:



En la parte del API, me dice qué debo importar:

Pues como siempre, en material.module.ts:

```

10   import {MatDialogModule} from '@angular/material/dialog';
11
12 @NgModule({
13   declarations: [],
14   imports: [
15     CommonModule,
16     MatToolbarModule,
17     MatButtonModule,
18     MatProgressBarModule,
19     MatCardModule,
20     MatSnackBarModule,
21     MatDialogModule
22   ],
23   exports: [
24     MatToolbarModule,
25     MatButtonModule,
26     MatProgressBarModule,
27     MatCardModule,
28     MatSnackBarModule,
29     MatDialogModule
30   ]

```

En los ejemplos:

```
Dialog with header, scrollable content and actions
HTML      TS      dialog-content-example-dialog.html
<button mat-button (click)="openDialog()">Open dialog</button>
Open dialog
```

Para editar el marcador necesitará una pantalla o formulario, que será un nuevo componente que crearemos:

```
jsersan@iMac-de-Jose appMapalocalStorage % ng g c components/mapa/mapa-editar --flat --skip-tests --skip-import --module=app.module
CREATE src/app/components/mapa/mapa-editar.component.scss (0 bytes)
CREATE src/app/components/mapa/mapa-editar.component.html (26 bytes)
CREATE src/app/components/mapa/mapa-editar.component.ts (295 bytes)
jsersan@iMac-de-Jose appMapalocalStorage %
```

Actualizamos app.module.ts:

```
11 import { MapaEditarComponent } from './components/mapa/mapa-editar.component';
12
13
14 @NgModule({
15   declarations: [
16     AppComponent,
17     MapaComponent,
18     MapaEditarComponent
19   ],

```

Según la documentación también hay que importar MatDialog, MatDialogRef, MAT\_DIALOG\_DATA en mapa.component.ts que es la página que va a llamar el modal.

```
1 import { Component, OnInit } from '@angular/core';
2 import { Marcador } from '../../classes/marcador.class';
3 import { MatSnackBar } from '@angular/material/snack-bar';
4 import { MatDialog, MatDialogRef } from '@angular/material/dialog';
```

Inyectamos en el constructor siguiendo la referencia oficial:

```
constructor(public dialog: MatDialog) {}

openDialog(): void {
  const dialogRef = this.dialog.open(DialogOverviewExampleDialog, {
    width: '250px',
    data: {name: this.name, animal: this.animal},
  });

  dialogRef.afterClosed().subscribe(result => {
    console.log('The dialog was closed');
    this.animal = result;
  });
}
```

Antes debemos crear la función editarMarcador(). En mapa.component.html agregamos el evento en el botón editar:

```
<div>
  <button (click)="editarMarcador(marcador)"
    | mat-raised-button color="primary">Editar</button>
  <button (click)="borrarMarcador(i)"
    | mat-raised-button color="warn">Eliminar</button>
</div>
```

En mapa.component.ts creamos la función editarMarcador y copiamos el código de la documentación oficial:

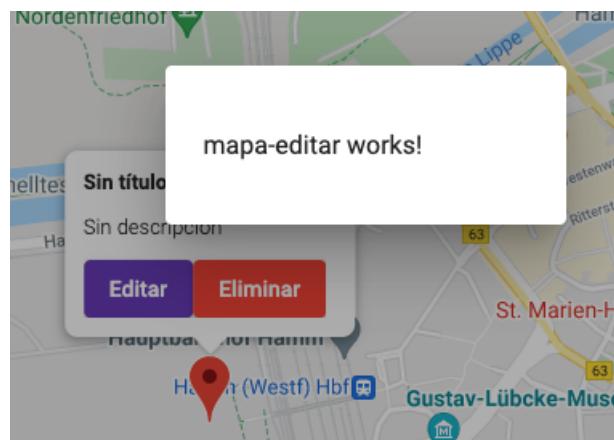
```
48  editarMarcador(marcador: Marcador){
49    const dialogRef = this.dialog.open(DialogOverviewExampleDialog, {
50      width: '250px',
51      data: {name: this.name, animal: this.animal},
52    );
53 }
```

Debemos realizar unos cambios para eliminar los errores:

- DialogOverviewExampleDialog: MapaEditarComponent
- Data: titulo, desc

```
49  editarMarcador(marcador: Marcador){
50    const dialogRef = this.dialog.open(MapaEditarComponent, [
51      width: '250px',
52      data: {titulo: marcador.titulo, desc: marcador.desc},
53    );
54 }
```

Si lo probamos, vemos que funciona. Nos abre un modal, ahora falta editarlo.



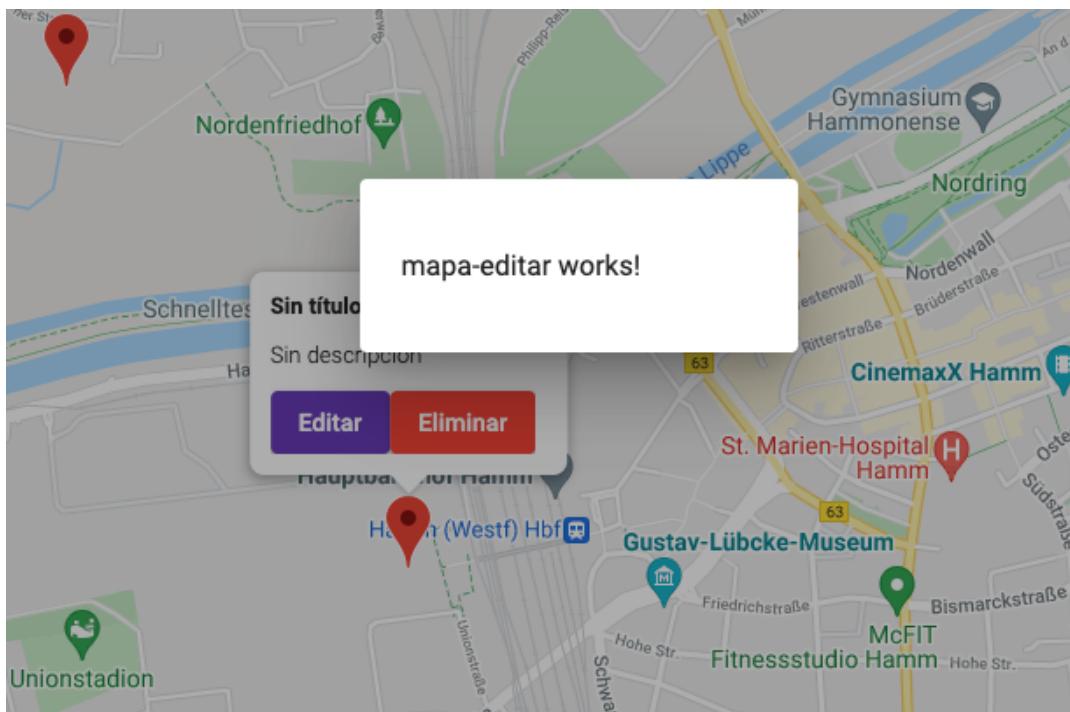
Ahora definimos en ap.module.ts una entryComponents que defina lo que va a ser el formulario de mapaEditarComponent:

```

14  @NgModule({
15    entryComponents: [
16      MapaEditarComponent
17    ],
18    declarations: [
19      AppComponent,
20      MapaComponent,
21      MapaEditarComponent
22    ],
23  })

```

Pruebo de nuevo:



Como vemos, sigue funcionando. Debemos recibir los parámetros en el modal. En la documentación oficial nos dice que debemos inyectar MAT\_DIALOG\_DATA

```

export class DialogOverviewExampleDialog {
  constructor(
    public dialogRef: MatDialogRef<DialogOverviewExampleDialog>,
    @Inject(MAT_DIALOG_DATA) public data: DialogData,
  ) {}

  onNoClick(): void {
    this.dialogRef.close();
  }
}

```

Así en mapa-editar.component.ts:



```

module.ts M X   TS mapa-editar.component.ts U X   <> mapa.component.html U   TS material.module.ts
mapalocalStorage > src > app > components > mapa > TS mapa-editar.component.ts > MapaEditarComponent
import { Component, OnInit } from '@angular/core';
import { MatDialog, MatDialogRef, MAT_DIALOG_DATA} from '@angular/material/dialog';

```

Ahora para utilizarlo debemos inyectarlo en el constructor:

```

export class DialogOverviewExampleDialog {
  constructor(
    public dialogRef: MatDialogRef<DialogOverviewExampleDialog>,
    @Inject(MAT_DIALOG_DATA) public data: DialogData,
  ) {}

  onNoClick(): void {
    this.dialogRef.close();
  }
}

```

Lo importante es que inyectemos el MAT\_DIALOG\_DATA en mapa-editar.component.ts.

```

11  constructor(
12    public dialogRef: MatDialogRef<MapaEditarComponent>,
13    @Inject(MAT_DIALOG_DATA) public data: any,
14  ) {}

```

@Inject debemos importarlo desde @angular/core. El componente mapa-editar.component.ts queda, mostrando por consola la data que se recibe:

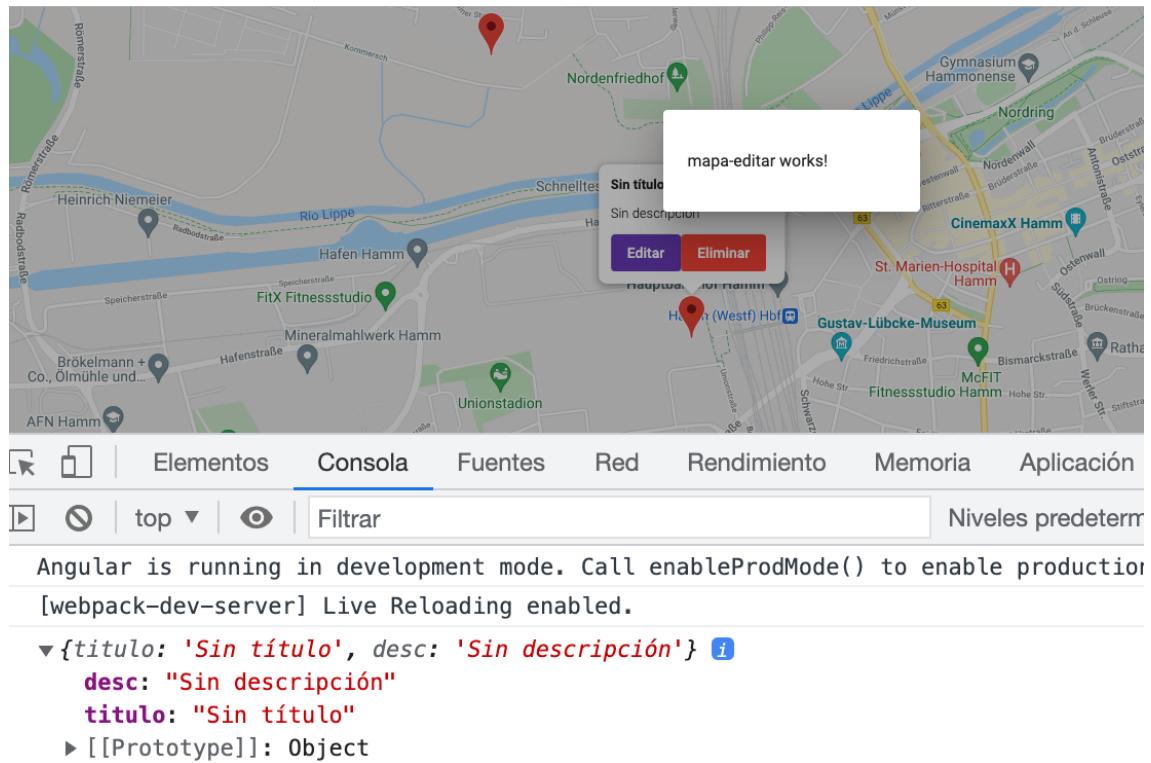


```

TS mapa-editar.component.ts U X   TS app.module.ts M   <> mapa.component.html U
appMapalocalStorage > src > app > components > mapa > TS mapa-editar.component.ts > MapaEditarComponent
1  import { Component, Inject } from '@angular/core';
2  import {MatDialogRef, MAT_DIALOG_DATA} from '@angular/material/dialog';
3
4  @Component({
5    selector: 'app-mapa-editar',
6    templateUrl: './mapa-editar.component.html',
7    styleUrls: ['./mapa-editar.component.scss']
8  })
9  export class MapaEditarComponent{
10
11    constructor(
12      public dialogRef: MatDialogRef<MapaEditarComponent>,
13      @Inject(MAT_DIALOG_DATA) public data: any){
14        console.log(data);
15      }
16
17  }

```

Probamos:



Faltaría que con los datos recibidos, formar un formulario y con el botón Guardar (Editar) actualizar los cambios.

## 12. Pantalla para editar el marcador:

Modificamos el HTML de mapa-editar.component.html para generar el formulario.

En la documentación oficial en Components > Input:

The screenshot shows the 'OVERVIEW' tab selected in the Angular Material Input component documentation. It displays two examples of the matInput directive. The first example shows a single-line input field with the placeholder 'Favorite food' and the value 'Sushi'. The second example shows a multi-line textarea with the placeholder 'Leave a comment'.

En API nos permite ver qué tenemos que importar:

The screenshot shows the 'API' tab selected in the Angular Material Input component documentation. It displays the API reference for the Angular Material input component. The code snippet shown is: `import {MatInputModule} from '@angular/material/input';`

Debemos importarla como todas las anteriores en material.module.ts. Seguimos con el ejemplo de la documentación oficial:

The screenshot shows the 'HTML' tab selected in the Angular Material Input component documentation. It displays the HTML code for the form. The code includes two mat-form-field components: one for 'Favorite food' with a placeholder 'Ex. Pizza' and value 'Sushi', and another for 'Leave a comment' with a placeholder 'Ex. It makes me feel...'. Both fields have the appearance set to 'fill'.

```

<form class="example-form">
  <mat-form-field class="example-full-width" appearance="fill">
    <mat-label>Favorite food</mat-label>
    <input matInput placeholder="Ex. Pizza" value="Sushi">
  </mat-form-field>

  <mat-form-field class="example-full-width" appearance="fill">
    <mat-label>Leave a comment</mat-label>
    <textarea matInput placeholder="Ex. It makes me feel..."></textarea>
  </mat-form-field>
</form>

```

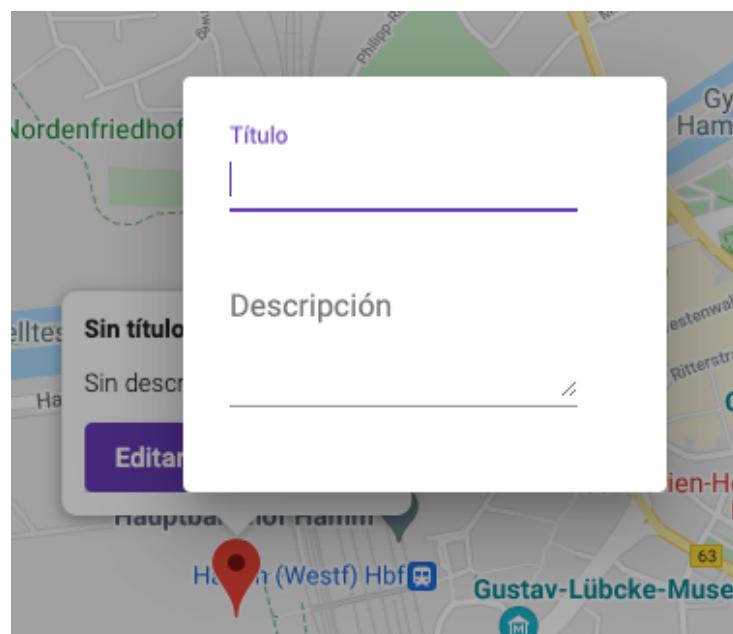
Este código lo ponemos en mapa-editar.html:

```
④ mapa-editar.component.html U ④ mapa.component.html U TS material.moc ④
appMapalocalStorage > src > app > components > mapa > ④ mapa-editar.component.html > ...
1  <form class="example-form">
2    <mat-form-field class="example-full-width" appearance="fill">
3      <mat-label>Favorite food</mat-label>
4      <input matInput placeholder="Ex. Pizza" value="Sushi">
5    </mat-form-field>
6
7    <mat-form-field class="example-full-width" appearance="fill">
8      <mat-label>Leave a comment</mat-label>
9      <textarea matInput placeholder="Ex. It makes me feel..."></textarea>
10   </mat-form-field>
11 </form>
```

Hacemos un par de cambios:

```
④ mapa-editar.component.html U ④ mapa.component.html U TS material.module.ts U
> app > components > mapa > ④ mapa-editar.component.html > form.example-form > mat-
1  <form class="example-form">
2    <mat-form-field>
3      <input matInput placeholder="Título">
4    </mat-form-field>
5
6    <mat-form-field>
7      <textarea rows="3" matInput placeholder="Descripción"></textarea>
8    </mat-form-field>
9  </form>
```

Queda:



Copiamos los botones de mapa.component.html:

```
<div>
  <button (click)="editarMarcador(marcador)"
    mat-raised-button color="primary">Editar</button>
  <button (click)="borrarMarcador(i)"
    mat-raised-button color="warn">Eliminar</button>
</div>
```

Hacemos unos cambios:

```
10   <button (click)="guardarCambios()"
11     mat-raised-button color="primary">Editar</button>
12   <button (click)=""
13     mat-raised-button color="warn">Cancelar</button>
14 </form>
```

Implementamos la función guardarCambios() en mapa-editar.component.ts:

```
17   guardarCambios(){
18     console.log("Datos guardados!!")
19 }
```

Hasta que le asociemos un evento o función al botón cancelar, no se quitará este error:

```
✖ ▶ [webpack-dev-server] ERROR
src/app/components/mapa/mapa-editar.component.html:12:22 - error NG
12   <button (click)="" ~
src/app/components/mapa/mapa-editar.component.ts:6:16
  6  templateUrl: './mapa-editar.component.html',
~~~~~
Error occurs in the template of component MapaEditarComponent.
```

Miramos en la documentación cuál es el código para cerrar el dialog:

```
onNoClick(): void {
  this.dialogRef.close();
}
```

Así en mapa-editar.component.ts:

```
17   guardarCambios(){
18     console.log("Datos guardados!!")
19 }
20
21   onNoClick(): void {
22     this.dialogRef.close();
23 }
```

Esta función onNoClick() la voy a llamar desde el botón cancelar en mapa.component.html:

```

10   <button (click)="guardarCambios()"
11     | mat-raised-button color="primary">Editar</button>
12   <button (click)="onNoClick()"
13     | mat-raised-button color="warn">Cancelar</button>
14 </form>

```

Como el botón está dentro de un formulario cuando pulsamos dentro de un pin>editar y luego cancelamos hace un refresh no deseado que ralentiza la ejecución. Para evitarlo hacemos que el botón sea de tipo button:

```

10   <button (click)="guardarCambios()"
11     | mat-raised-button color="primary">Editar</button>
12   <button (click)="onNoClick()" type="button"
13     | mat-raised-button color="warn">Cancelar</button>
14 </form>

```

Para cargar el formulario con la data utilizamos el componente de angular ReactiveFormsModuleModule. Lo importamos en app.module.ts:

```

24 imports: [
25   BrowserModule,
26   BrowserAnimationsModule,
27   MaterialModule,
28   ReactiveFormsModule,
29   AgmCoreModule.forRoot({
30     apiKey: 'AIzaSyDY3YQ6aNb7YXoz13RIxq5fVyzdbz
31 })

```

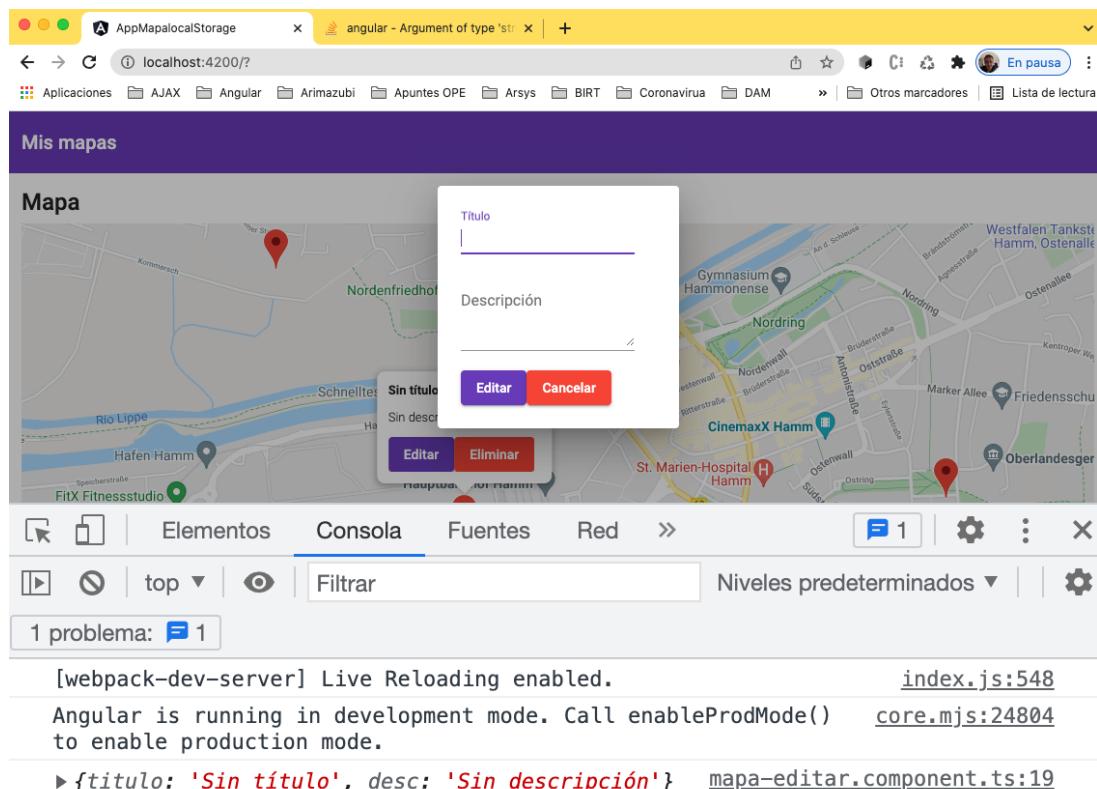
Ahora en mapa-editar.component.ts importamos el FormBuilder y generar un formulario con la data de forma fácil de esta manera:

```

4 import { FormBuilder, FormGroup } from '@angular/forms';
5
6 @Component({
7   selector: 'app-mapa-editar',
8   templateUrl: './mapa-editar.component.html',
9   styleUrls: ['./mapa-editar.component.scss']
10})
11 export class MapaEditarComponent{
12
13   forma: FormGroup;
14
15   constructor(
16     public fb: FormBuilder,
17     public dialogRef: MatDialogRef<MapaEditarComponent>,
18     @Inject(MAT_DIALOG_DATA) public data: any){
19       console.log(data);
20       this.forma = fb.group({
21         'titulo': data.titulo,
22         'desc': data.desc
23       });
24     }

```

En funcionamiento:



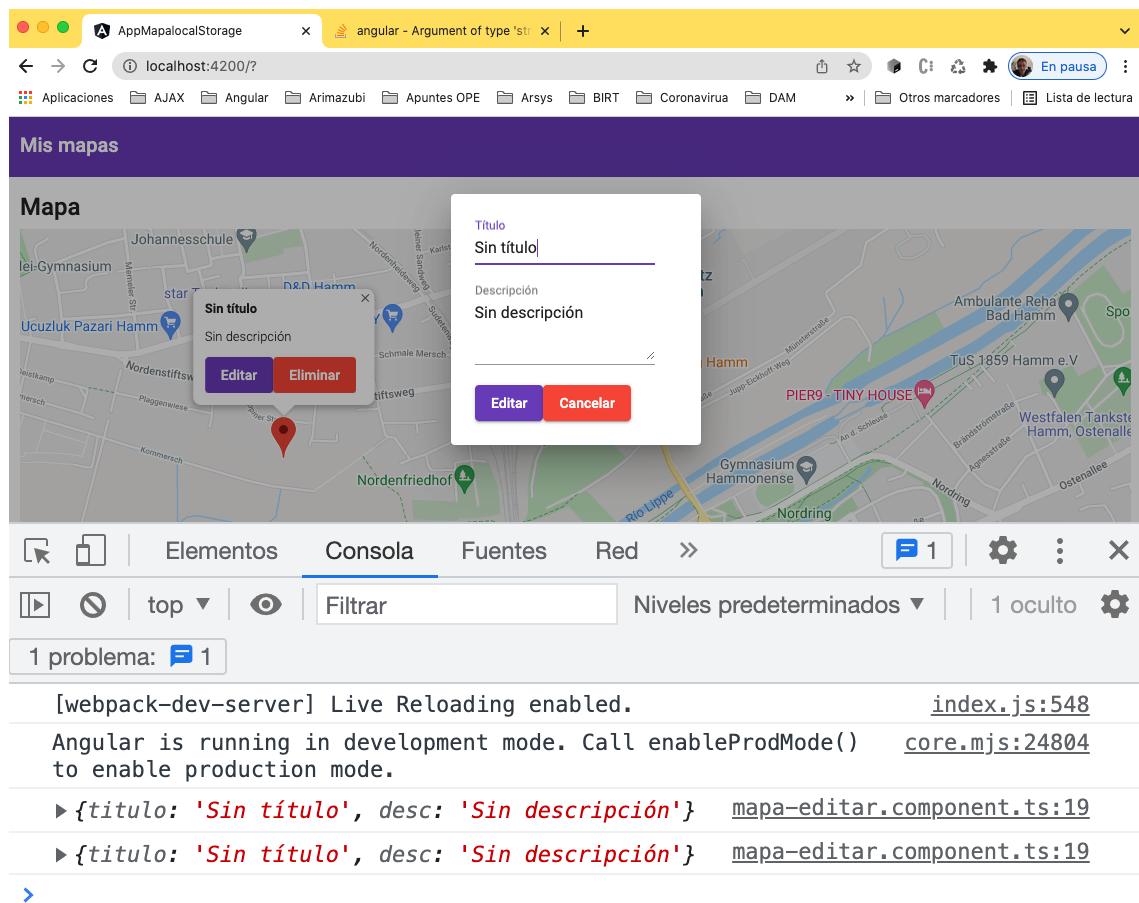
En mapa-editar.component.html:

```

<mapa-editar.component.html U X < mapo-editar.component.scss U < mapa.cor
storage > src > app > components > mapa > mapa-editar.component.html > form > mat
1   <form [formGroup]="forma">
2     <mat-form-field>
3       <input matInput placeholder="Título" formControlName="titulo">
4     </mat-form-field>
5
6     <mat-form-field>
7       <textarea rows="3" matInput placeholder="Descripción"
8           formControlName="desc"></textarea>
9     </mat-form-field>
10
11    <button (click)="guardarCambios()">
12      mat-raised-button color="primary">Editar</button>
13    <button (click)="onNoClick()" type="button">
14      mat-raised-button color="warn">Cancelar</button>
15  </form>
  
```

El formulario tiene de valor formGroup = forma y luego cada elemento tiene un formControlName asociado distinto.

Si pulsamos el botón Editar de un pin:



Recuerden que esto viene de aquí de mapa-editar.component.ts:

```

15  constructor(
16    public fb: FormBuilder,
17    public dialogRef: MatDialogRef<MapaEditarComponent>,
18    @Inject(MAT_DIALOG_DATA) public data: any){
19      console.log(data);
20      this.forma = fb.group({
21        'titulo': data.titulo,
22        'desc': data.desc
23      });
24

```

### 13. Actualizar el marcador con la información del dialog:

Recuerden que si pulsamos el botón Editar:

```

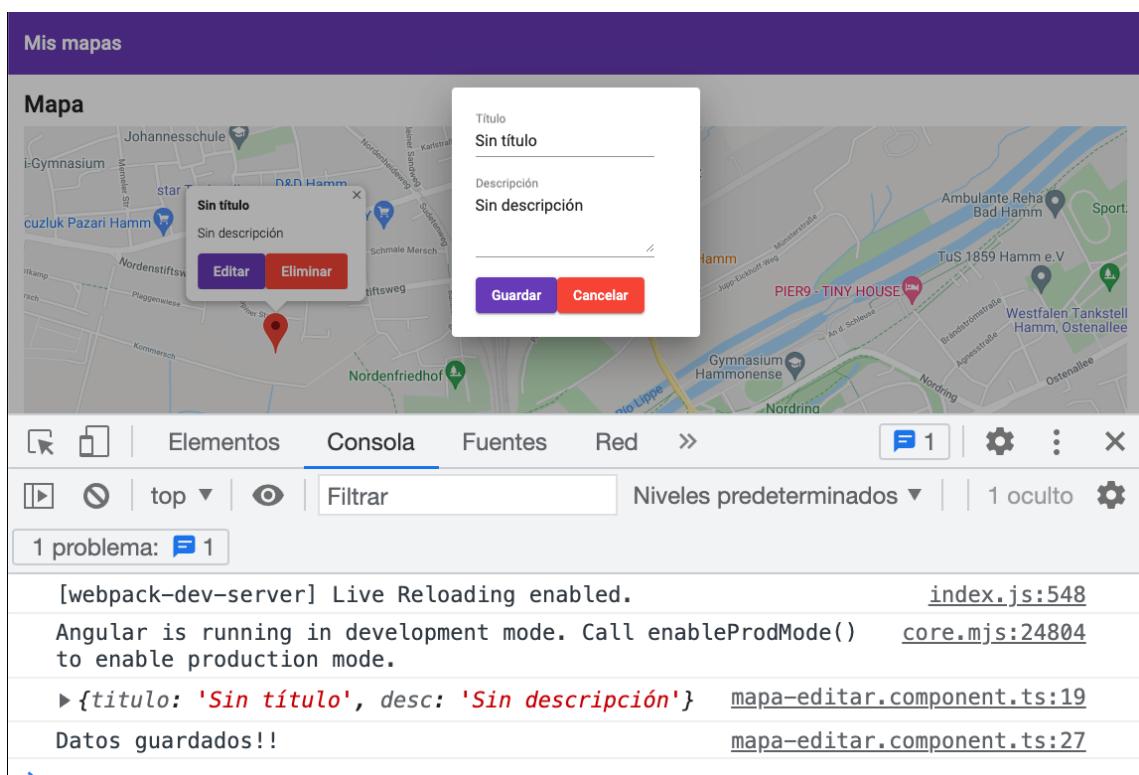
11   <button (click)="guardarCambios()" mat-raised-button color="primary">Guardar</button>
12   <button (click)="onNoClick()" type="button" mat-raised-button color="warn">Cancelar</button>
13
14
15 </form>
--
```

La función guardarCambios:

```

26   guardarCambios(){
27     console.log("Datos guardados ! !")
28   }
29
30   onNoClick(): void {
31     this.dialogRef.close();
32 }
```

En funcionamiento:

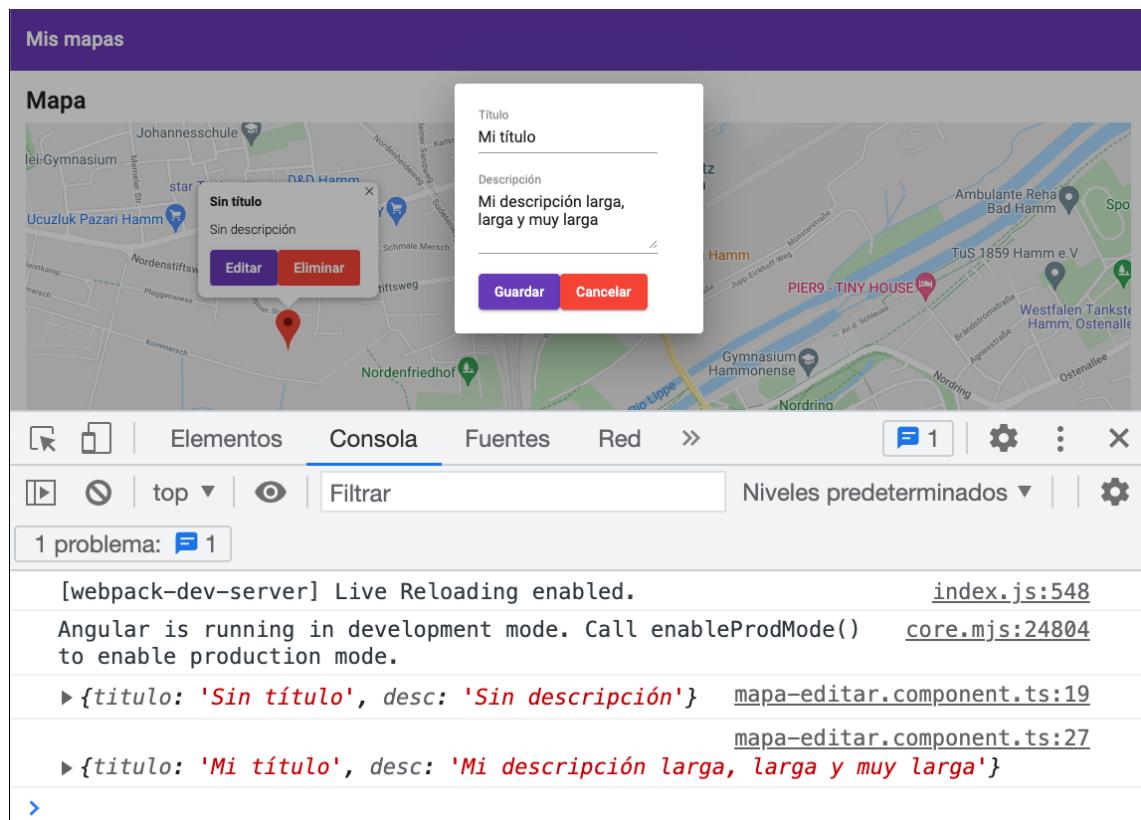


Debemos modificar la función guardarCambios()

Para obtener la información del formulario basta con:

```
26  guardarCambios(){}
27    console.log(this.forma.value)
28 }
```

En marcha:

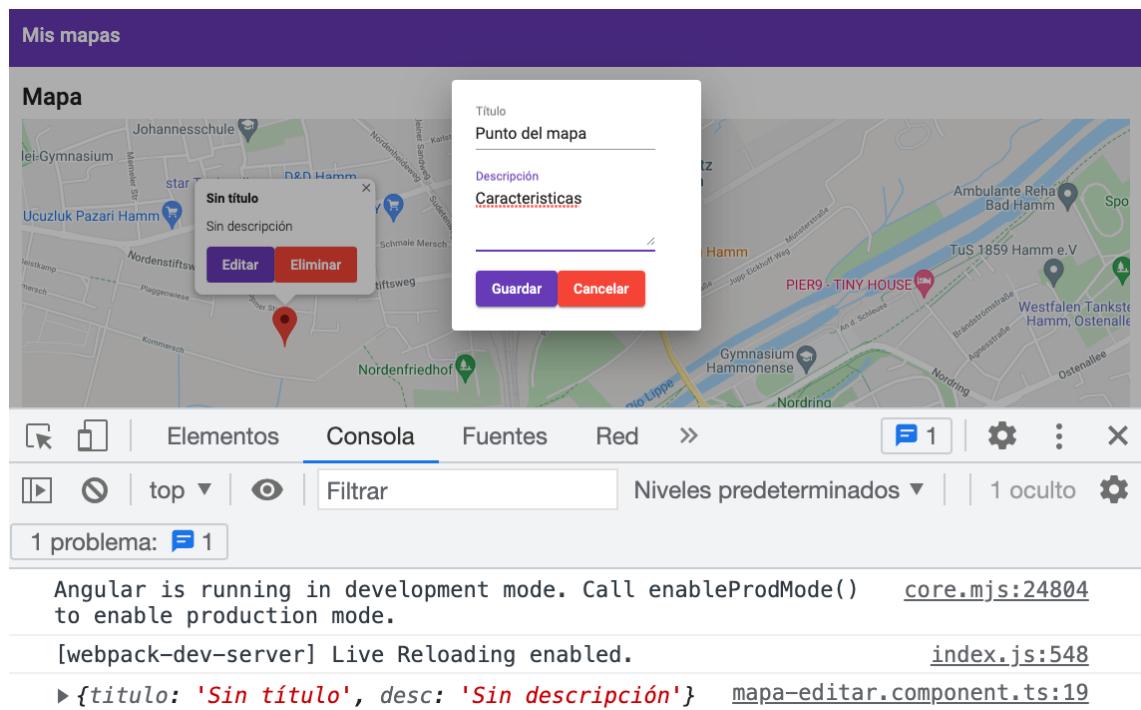


La idea es que cuando le demos al botón guardar, actualice el localStorage y cierre el modal:

```
26  guardarCambios(){
27    this.dialogRef.close(this.forma.value);
28 }
```

Esta parte debemos hacerla en la función editarMarcador(). Lo que hacemos es modificar el objeto marcador miembro del array marcadores por los valores devueltos por el Dialog y luego llamar a la función guardarStorage (ver página siguiente).

Probamos:



Cuando cerremos los dialog debemos guardar en el localStorage lo que ha devuelto la promesa afterClosed de dialogRef:

```

49  editarMarcador(marcador: Marcador){
50    const dialogRef = this.dialog.open(MapaEditarComponent, {
51      width: '250px',
52      data: {titulo: marcador.titulo, desc: marcador.desc},
53    );
54
55    dialogRef.afterClosed().subscribe(result => {
56      console.log('Hemos cerrado el dialogRef');
57
58      if ( !result ) {
59        return;
60      }
61
62      marcador.titulo = result.titulo;
63      marcador.desc = result.desc;
64
65      this.guardarStorage();
66    );
67
68

```

El resultado es:



Pulsamos en Editar:

Mapa

Título  
Mi restaurante favorito

Descripción  
Pescado Fresco a buen precio

Guardar Cancelar

Elementos Consola Fuentes Red

Filtrar

Niveles predeterminados

1 problema: 1

▶ {titulo: 'Sin título', desc: 'Sin descripción'} mapa-editar.component.ts:19

Le damos a Guardar:

Mapa

Elementos Consola Fuentes Red

Filtrar

Niveles predeterminados

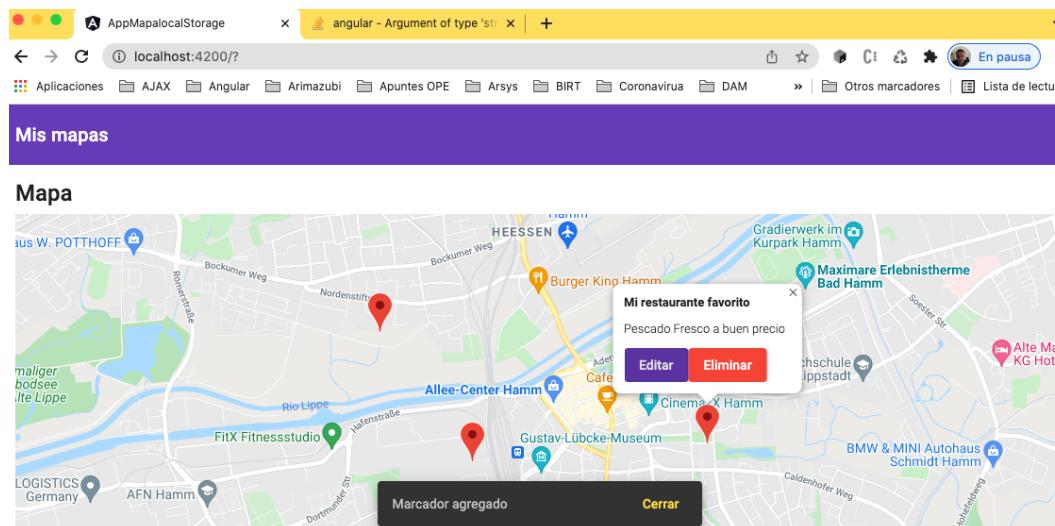
1 problema: 1

▶ {titulo: 'Sin título', desc: 'Sin descripción'} mapa-editar.component.ts:19

Hemos cerrado el dialogRef

mapa.component.ts:56

Ha salido un mensaje durante tres segundos de marcador guardado.



Habrá que corregirlo por lo de Datos Guardados en mapa.component.ts.

```

37  guardarStorage(){
38
39      localStorage.setItem('marcadores', JSON.stringify(this.marcadores));
40      this.snackBar.open('Datos guardados', 'Cerrar', {duration: 3000 });
41

```

Resultado:

