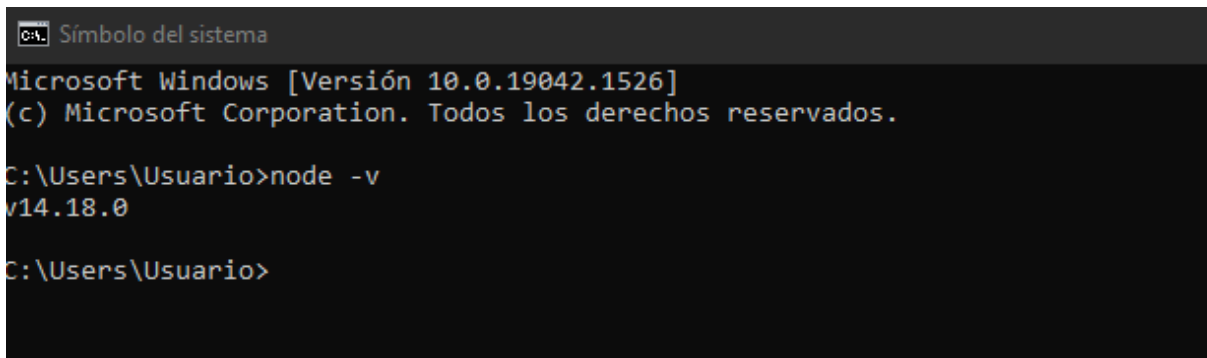


Aspectos generales de una app Angular

Primeros pasos

Tendremos que asegurar nos de tener instalado node.js en nuestro equipo para ello descargamos el instalador desde el siguiente enlace: <https://nodejs.org/es/download/>.

Podremos comprobar si se ha instalado correctamente desde la consola de windows ejecutando el comando `node -v`

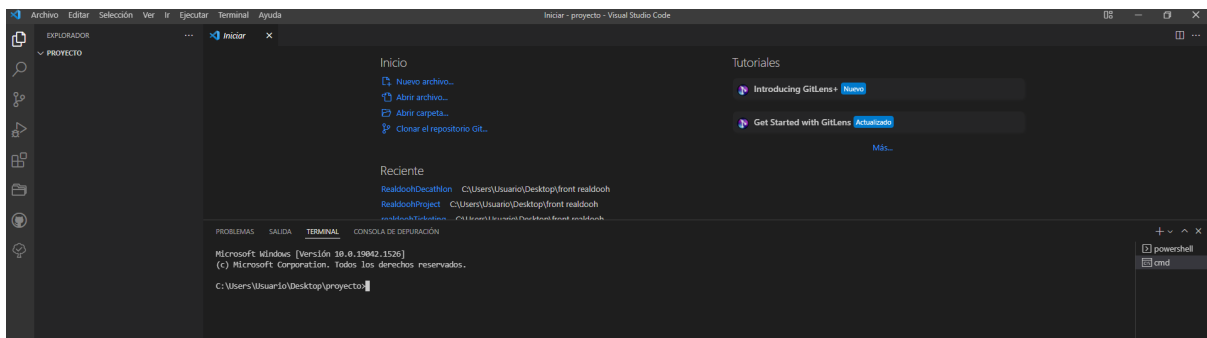


```
C:\> Símbolo del sistema
Microsoft Windows [Versión 10.0.19042.1526]
(c) Microsoft Corporation. Todos los derechos reservados.

C:\Users\Usuario>node -v
v14.18.0

C:\Users\Usuario>
```

Para crear nuestro proyecto angular nos situaremos en un directorio vacío donde no exista ningún otro proyecto anterior y ejecutaremos el comando `ng new proyecto`. Ya sea desde el cmd de windows o desde el terminal de VSCode.



Nos preguntará si necesitamos añadir el módulo routing a lo cual diremos que si .

```
Microsoft Windows [Versión 10.0.19042.1526]
(c) Microsoft Corporation. Todos los derechos reservados.

C:\Users\Usuario\Desktop\proyecto>ng new proyecto
? Would you like to add Angular routing? Yes
? Which stylesheet format would you like to use? (Use arrow keys)
> CSS
SCSS [ https://sass-lang.com/documentation/syntax#scss ]
Sass [ https://sass-lang.com/documentation/syntax#the-indented-syntax ]
Less [ http://lesscss.org ]
```

Nos preguntará por los estilos y seleccionaremos SCSS.

```
Microsoft Windows [Versión 10.0.19042.1526]
(c) Microsoft Corporation. Todos los derechos reservados.

C:\Users\Usuario\Desktop\proyecto>ng new proyecto
? Would you like to add Angular routing? Yes
? Which stylesheet format would you like to use?
> SCSS [ https://sass-lang.com/documentation/syntax#scss ]
Sass [ https://sass-lang.com/documentation/syntax#the-indented-syntax ]
Less [ http://lesscss.org ]
```

```
Microsoft Windows [Versión 10.0.19042.1526]
(c) Microsoft Corporation. Todos los derechos reservados.

C:\Users\Usuario\Desktop\proyecto>ng new proyecto
? Would you like to add Angular routing? Yes
? Which stylesheet format would you like to use? SCSS [ https://sass-lang.com/documentation/syntax#scss ]
CREATE proyecto/angular.json (3219 bytes)
CREATE proyecto/package.json (1070 bytes)
CREATE proyecto/README.md (1054 bytes)
CREATE proyecto/tsconfig.json (783 bytes)
CREATE proyecto/.editorconfig (274 bytes)
CREATE proyecto/.gitignore (604 bytes)
CREATE proyecto/.browserslistrc (703 bytes)
CREATE proyecto/karma.conf.js (1425 bytes)
CREATE proyecto/tsconfig.app.json (287 bytes)
CREATE proyecto/tsconfig.spec.json (333 bytes)
CREATE proyecto/src/favicon.ico (948 bytes)
CREATE proyecto/src/index.html (294 bytes)
CREATE proyecto/src/main.ts (372 bytes)
CREATE proyecto/src/polyfills.ts (2820 bytes)
CREATE proyecto/src/styles.scss (80 bytes)
CREATE proyecto/src/test.ts (788 bytes)
CREATE proyecto/src/assets/.gitkeep (0 bytes)
CREATE proyecto/src/environments/environment.prod.ts (51 bytes)
CREATE proyecto/src/environments/environment.ts (658 bytes)
CREATE proyecto/src/app/app-routing.module.ts (245 bytes)
CREATE proyecto/src/app/app.module.ts (393 bytes)
CREATE proyecto/src/app/app.component.html (24617 bytes)
CREATE proyecto/src/app/app.component.spec.ts (1079 bytes)
CREATE proyecto/src/app/app.component.ts (213 bytes)
CREATE proyecto/src/app/app.component.scss (0 bytes)
.: Installing packages (npm)...
```

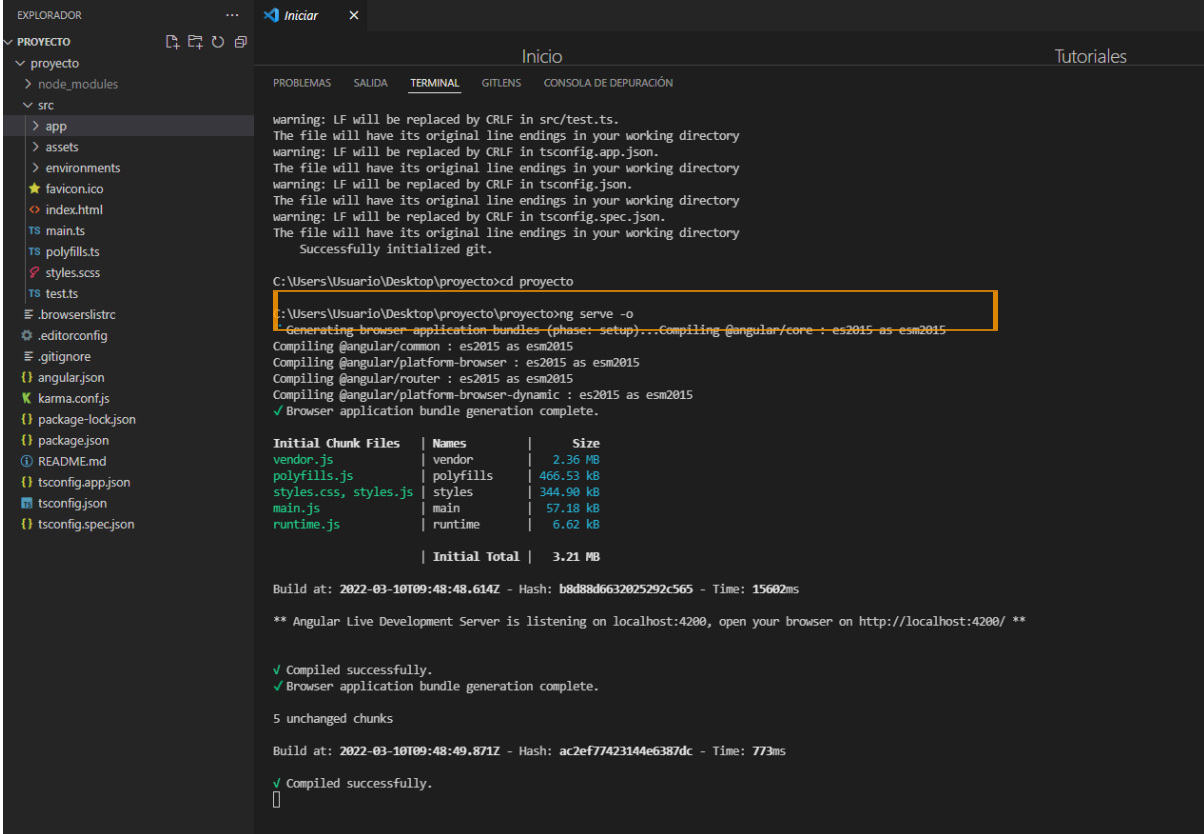
Y finalmente tendremos nuestro proyecto creado.

```
PROYECTO
└─ proyecto
  └─ node_modules
  └─ src
    ├── .browserslistrc
    ├── .editorconfig
    ├── .gitignore
    ├── angular.json
    ├── karma.conf.js
    ├── package-lock.json
    ├── package.json
    ├── README.md
    ├── tsconfig.app.json
    ├── tsconfig.json
    └── tsconfig.spec.json
```

Inicio

```
warning: LF will be replaced by CRLF in package-lock.json.
The file will have its original line endings in your working directory
warning: LF will be replaced by CRLF in package.json.
The file will have its original line endings in your working directory
warning: LF will be replaced by CRLF in src/app/app-routing.module.ts.
The file will have its original line endings in your working directory
warning: LF will be replaced by CRLF in src/app/app.component.html.
The file will have its original line endings in your working directory
warning: LF will be replaced by CRLF in src/app/app.component.spec.ts.
The file will have its original line endings in your working directory
warning: LF will be replaced by CRLF in src/app/app.component.ts.
The file will have its original line endings in your working directory
warning: LF will be replaced by CRLF in src/app/app.module.ts.
The file will have its original line endings in your working directory
warning: LF will be replaced by CRLF in src/environments/environment.prod.ts.
The file will have its original line endings in your working directory
warning: LF will be replaced by CRLF in src/environments/environment.ts.
The file will have its original line endings in your working directory
warning: LF will be replaced by CRLF in src/index.html
```

Podremos lanzar nuestro proyecto con el comando `ng serve -o`. Este comando nos abrirá una ventana de nuestro navegador con nuestro proyecto. Angular nos permite realizar cambios que se refrescan automáticamente siempre que la aplicación está lanzada.



```
EXPLORADOR
PROYECTO
  proyecto
    node_modules
    src
      app
      assets
      environments
      favicon.ico
      index.html
      main.ts
      polyfills.ts
      styles.scss
      test.ts
      .browserslistrc
      .editorconfig
      .gitignore
      angular.json
      karma.conf.js
      package-lock.json
      package.json
      README.md
      tsconfig.app.json
      tsconfig.json
      tsconfig.spec.json

Inicio
PROBLEMAS SALIDA TERMINAL GITLENS CONSOLA DE DEPURACIÓN

warning: LF will be replaced by CRLF in src/test.ts.
The file will have its original line endings in your working directory
warning: LF will be replaced by CRLF in tsconfig.app.json.
The file will have its original line endings in your working directory
warning: LF will be replaced by CRLF in tsconfig.json.
The file will have its original line endings in your working directory
warning: LF will be replaced by CRLF in tsconfig.spec.json.
The file will have its original line endings in your working directory
Successfully initialized git.

C:\Users\Usuario\Desktop\proyecto>cd proyecto
C:\Users\Usuario\Desktop\proyecto\proyecto>ng serve -o
Generating browser application bundles (phase: setup)...Compiling @angular/core : es2015 as esm2015
Compiling @angular/common : es2015 as esm2015
Compiling @angular/platform-browser : es2015 as esm2015
Compiling @angular/router : es2015 as esm2015
Compiling @angular/platform-browser-dynamic : es2015 as esm2015
✓ Browser application bundle generation complete.

Initial Chunk Files | Names | Size
vendor.js | vendor | 2.36 MB
polyfills.js | polyfills | 466.53 KB
styles.css, styles.js | styles | 344.98 KB
main.js | main | 57.18 KB
runtime.js | runtime | 6.62 KB
| Initial Total | 3.21 MB

Build at: 2022-03-10T09:48:48.614Z - Hash: b8d88d6632025292c565 - Time: 15602ms

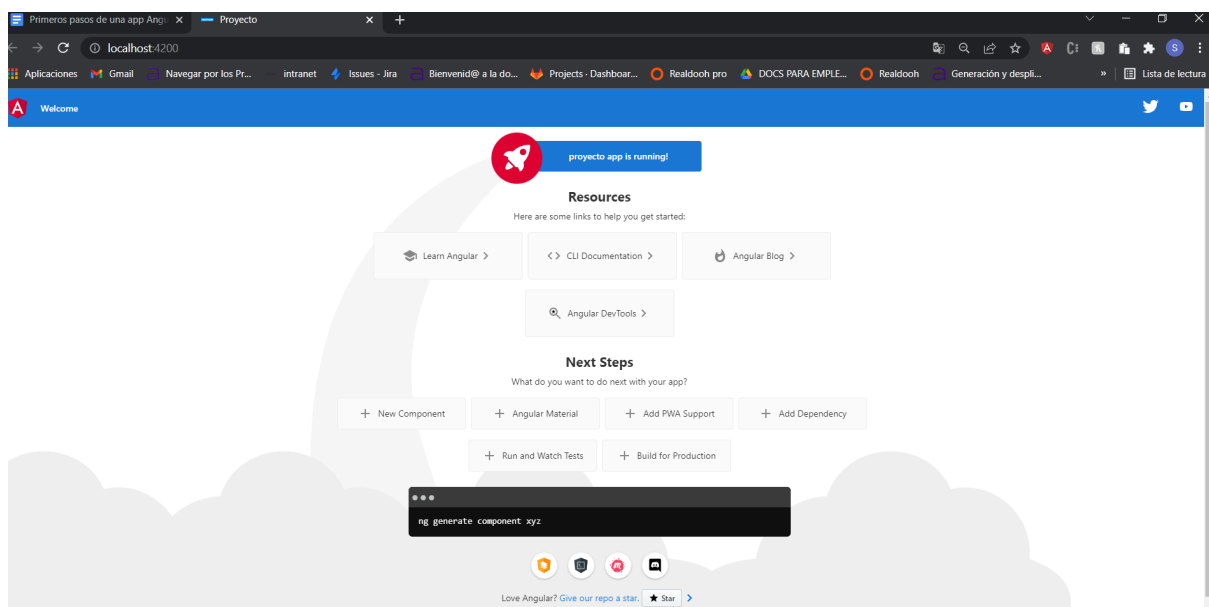
** Angular Live Development Server is listening on localhost:4200, open your browser on http://localhost:4200/ **

✓ Compiled successfully.
✓ Browser application bundle generation complete.

5 unchanged chunks

Build at: 2022-03-10T09:48:49.871Z - Hash: ac2ef77423144e6387dc - Time: 773ms

✓ Compiled successfully.
█
```



borraremos todo el código que hay en nuestro archivo `app.component.html` dejando solo la última etiqueta.

Antes:

```
474 Love Angular?&nbsp;
475 <a href="https://github.com/angular/angular" target="_blank" rel="noopener"> Give our repo a star.
476 <div class="github-star-badge">
477   <svg class="material-icons" xmlns="http://www.w3.org/2000/svg" width="24" height="24" viewBox="0 0 24 24"><path d="M0 0h24v24H0z"/></p
478   Star
479 </div>
480 </a>
481 <a href="https://github.com/angular/angular" target="_blank" rel="noopener">
482 <svg class="material-icons" xmlns="http://www.w3.org/2000/svg" width="24" height="24" viewBox="0 0 24 24"><path d="M10 6L8.59 7.41 13.17 12l-4.58 4.58
483 </a>
484 </footer>
485
486 <svg id="clouds" xmlns="http://www.w3.org/2000/svg" width="2611.084" height="485.677" viewBox="0 0 2611.084 485.677">
487 <title>Gray Clouds Background</title>
488 <path id="Path_39" data-name="Path 39" d="M2379.789,863.793c10-93-77-171-168-149-52-114-225-105-264,15-75,3-140,59-152,133-30,2.83-66.725,9.829-93.5,26.25
489 </svg>
490
491 </div>
492
493 <!-- ***** The content above ***** -->
494 <!-- ***** The content above ***** -->
495 <!-- ***** is only a placeholder ***** -->
496 <!-- ***** and can be replaced. ***** -->
497 <!-- ***** -->
498 <!-- ***** End of Placeholder ***** -->
499 <!-- ***** -->
500 </div>
501 <router-outlet></router-outlet>
502
```

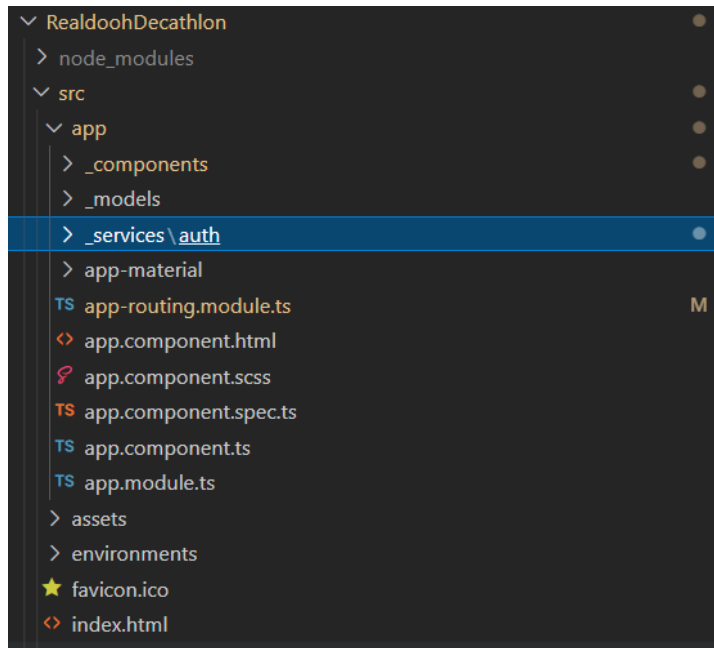
Después:

```
1 Go to component | You, hace 19 minutos | 1 author (You)
2 <router-outlet></router-outlet> You, hace 19 minutos • initial commit ...
```

Para parar nuestra aplicación solo tendremos que hacer Ctrl + c en el terminal de VSCode

Estructura de carpetas

Al iniciar un proyecto crearemos una estructura de carpetas de la siguiente manera:



dentro de app crearemos las carpetas **_componentes** donde irán los distintos componentes de la aplicación.

La carpeta **_models** donde irán las distintas clases que nos serán de utilidad para nuestra aplicación.

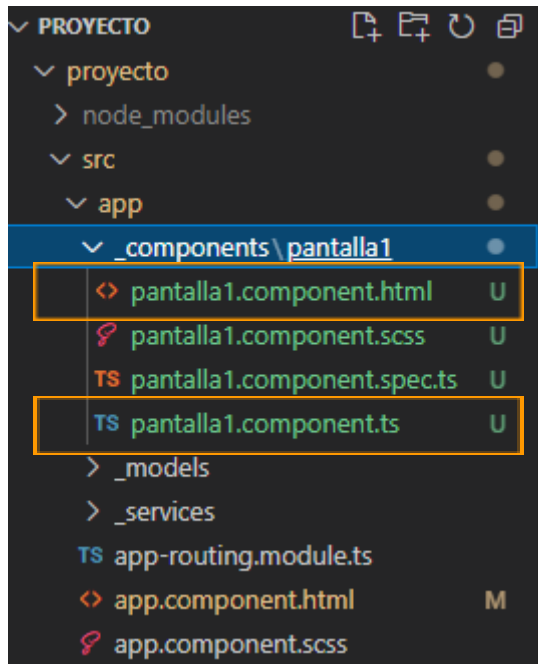
La carpeta **_services** dónde irán los distintos servicios para las peticiones http de nuestra aplicación.

Generación de elementos Angular

Para el desarrollo de nuestra aplicación haremos uso de distintos componentes ya sea para las pantallas, las clases o los servicios para las distintas peticiones http.

Por ejemplo si queremos crear una nueva pantalla o una nueva página nos moveremos a la carpeta **_components** y ejecutaremos el comando `ng generate component pantallal1`.

```
C:\Users\Usuario\Desktop\proyecto\proyecto\src\app\_components>ng generate component pantallal1
CREATE src/app/_components/pantallal1/pantallal1.component.html (24 bytes)
CREATE src/app/_components/pantallal1/pantallal1.component.spec.ts (647 bytes)
CREATE src/app/_components/pantallal1/pantallal1.component.ts (288 bytes)
CREATE src/app/_components/pantallal1/pantallal1.component.scss (0 bytes)
UPDATE src/app/app.module.ts (499 bytes)
C:\Users\Usuario\Desktop\proyecto\proyecto\src\app\_components>
```



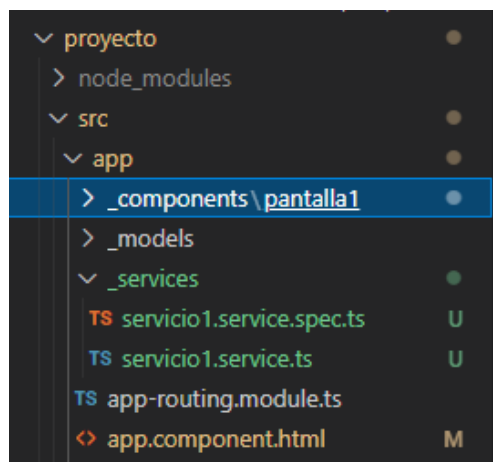
Esto nos creará una estructura de archivos en el cual tendremos el html y el archivo ts relacionado a este front.

Estos archivos son los que editaremos generalmente para codificar nuestra página.

Hay que tener en cuenta que cada componente se tiene que asignar en el archivo router que es el que nos permitirá navegar entre ellos (*El apartado del Router se explica más adelante*).

Para generar los servicios que serán los encargados de tener los métodos con las llamadas HTTP a nuestro back, nos moveremos a la carpeta **_services** y ejecutaremos el comando `ng generate service Servicio1`.

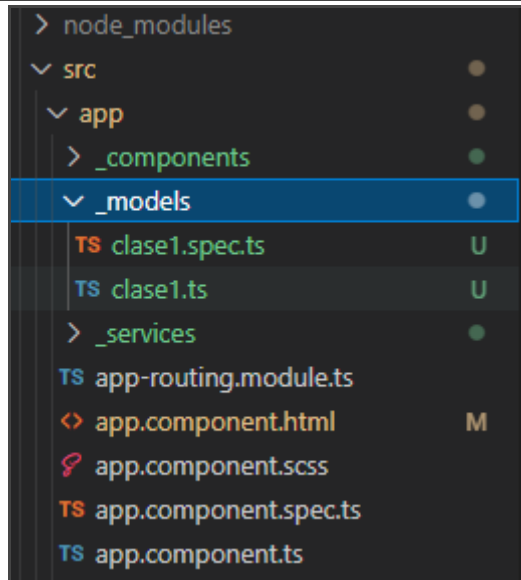
```
C:\Users\Usuario\Desktop\proyecto\proyecto\src\app\_services>ng generate service Servicio1
CREATE src/app/_services/servicio1.service.spec.ts (372 bytes)
CREATE src/app/_services/servicio1.service.ts (138 bytes)
C:\Users\Usuario\Desktop\proyecto\proyecto\src\app\_services>
```



También necesitaremos clases para manejar la información de manera eficiente así que para ello nos moveremos a nuestra carpeta **_models** y ejecutaremos el comando `ng generate class Clase1`

```
C:\Users\Usuario\Desktop\proyecto\proyecto\src\app\_models>ng generate class Clase1
CREATE src/app/_models/clase1.spec.ts (154 bytes)
CREATE src/app/_models/clase1.ts (24 bytes)

C:\Users\Usuario\Desktop\proyecto\proyecto\src\app\_models>
```



Módulo Material

Crearemos una carpeta **app-material** la cual nos servirá para tener un módulo con todas las clases Material de angular de esa manera podemos exportarnos el módulo en general, sin tener que hacer un importe de cada componente Material que necesitemos.

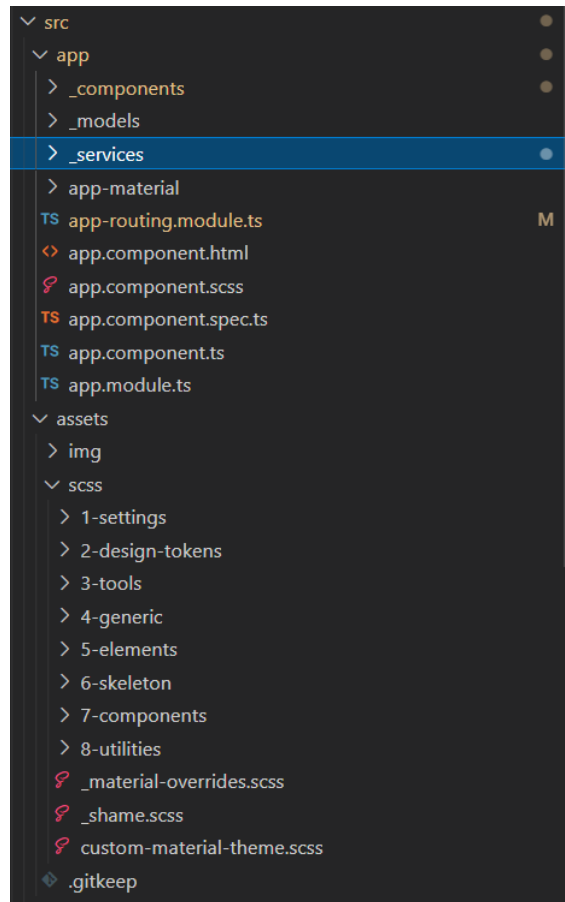
```
TS app.module.ts X
RealdoohDecathlon > src > app > TS app.module.ts > ...
You, 2 hours ago | 3 authors (Alba Fdz and others)
1 import { NgModule } from '@angular/core';
2 import { BrowserModule } from '@angular/platform-browser';
3
4 import { AppRoutingModule } from './app-routing.module';
5 import { AppComponent } from './app.component';
6 import { BrowserModuleAnimationsModule } from '@angular/platform-browser/animations';
7 import { MatMenuModule } from '@angular/material/menu';
8 import { HeaderComponent } from './_components/header/header.component';
9 import { MatIconModule } from '@angular/material/icon';
10 import { HomepageComponent } from './_components/homepage/homepage.component';
11 import { LoginComponent } from './_components/login/login.component';
12 import { MatFormFieldModule } from '@angular/material/form-field';
13 import { MatInputModule } from '@angular/material/input';
14 import { FormsModule, ReactiveFormsModule } from '@angular/forms';
15 import { MatButtonModule } from '@angular/material/button';
16 import { MatSelectModule } from '@angular/material/select';
17 import { MatDialogModule } from '@angular/material/dialog';
18 import { OpenIncidenceComponent } from './_components/incidence/open-incidence/open-incidence.component';
19 import { ImeiDialogComponent } from './_components/incidence/imei-dialog/imei-dialog.component';
20 import { MatRadioModule } from '@angular/material/radio';
21 import { TypeIncidenceComponent } from './_components/incidence/type-incidence/type-incidence.component';
22 import { ConfirmIncidenceComponent } from './_components/incidence/confirm-incidence/confirm-incidence.component';
23 import { NewDeviceComponent } from './_components/device/new-device/new-device.component';
24 import { ConfirmDeviceComponent } from './_components/device/confirm-device/confirm-device.component';
25 import { MatTableModule } from '@angular/material/table';
26 import { DeviceListingComponent } from './_components/device/device-listing/device-listing.component';
27 import { IncidenceListingComponent } from './_components/incidence/incidence-listing/incidence-listing.component';
28 import { MatAutocompleteModule } from '@angular/material/autocomplete';
29 import { AppMaterialModule } from './app-material/app-material.module';
30
31
```

```
TS app.module.ts X
src > app > TS app.module.ts > ...
Santiago, 3 months ago | 2 authors (Santiago and others)
1 import { NgModule } from '@angular/core';
2 import { BrowserModule } from '@angular/platform-browser';
3 import { HttpClientModule, HTTP_INTERCEPTORS } from '@angular/common/http';
4 import { FormsModule, ReactiveFormsModule } from '@angular/forms';
5 import { AppRoutingModule } from './app-routing.module';
6 // Components
7 import { AppComponent } from './app.component';
8 import { BrowserModuleAnimationsModule } from '@angular/platform-browser/animations';
9 import { platformBrowserDynamic } from '@angular/platform-browser-dynamic';
10 import { DataCollectComponent } from './_components/data-collect/data-collect.component';
11 import { AppMaterialModule } from './app-material/app-material.module'; Santiago, 4 months ago • Initial commit
12 import { DataMobileComponent } from './_components/data-mobile/data-mobile.component';
13 import { DataIncidenceComponent } from './_components/data-incidence/data-incidence.component';
14 import { DataCommunicationComponent } from './_components/data-communication/data-communication.component';
15 import { DialogContentComponent } from './_components/data-collect/dialog-content/dialog-content.component';
16 import { DialogCancelComponent } from './_components/data-mobile/dialog-cancel/dialog-cancel.component';
17 // Services
18 import { AuthInterceptor } from './_services/auth.interceptor';
19
20 cfernandez, 3 months ago | 2 authors (Santiago and others)
21 @NgModule({
22   declarations: [
23     AppComponent
24   ],
25   imports: [
26     BrowserModule,
27     AppRoutingModule,
28     AppMaterialModule
29   ],
30   providers: [
31     AuthInterceptor
32   ],
33   bootstrap: [AppComponent]
34 })
35 export class AppModule {}
36
```

Nótese la diferencia entre el app.module de un proyecto sin un modo general con todos los componentes de material y otro que sí tiene un módulo Material en general .

Custom-material-theme.scss

Todos los archivos css de la aplicación intentaremos dejarlos en la carpeta assets excepto el styles.css que es la hoja de estilos general de la aplicación.



Router

En el router crearemos una ruta definitiva de manera que cualquier ruta pueda apuntar a el componente principal de la aplicación en este caso el login.

```
const routes: Routes = [  
  { path: 'login', component: LoginComponent },  
  { path: 'home', component: HomepageComponent, canActivate: [AuthGuardGuard] },  
  { path: 'incidence', component: OpenIncidenceComponent, canActivate: [AuthGuardGuard] },  
  { path: 'incidence-type', component: TypeIncidenceComponent, canActivate: [AuthGuardGuard] },  
  { path: 'incidence-confirm', component: ConfirmIncidenceComponent, canActivate: [AuthGuardGuard] },  
  { path: 'new-device', component: NewDeviceComponent, canActivate: [AuthGuardGuard] },  
  { path: 'device-confirm', component: ConfirmDeviceComponent, canActivate: [AuthGuardGuard] },  
  { path: 'device-listing', component: DeviceListingComponent, canActivate: [AuthGuardGuard] },  
  { path: 'incidence-listing', component: IncidenceListingComponent, canActivate: [AuthGuardGuard] },  
  {  
    path: '',  
    pathMatch: 'full',  
    component: LoginComponent,  
    data: { title: 'login' },  
  },  
];
```

Como podemos ver arriba también podremos mapear las distintas rutas a nuestros distintos componentes de nuestra aplicación, de esa manera, si por ejemplo queremos pasar del el

componente login al componente home o cualquier otro componente. Apuntando al path asignado podremos navegar por ellos .

```
auth(){
  this.authService.auth(this.myForm.get('user')?.value, this.myForm.get('p
  this.authService.setToken(response.access_token);
  this.authService.login();
  this.router.navigate(['/home']);

}).catch((error: any) => {
  this.hasError = true;
  this.router.navigate(['/login']);
});
}
```

Por ejemplo, en este caso si la request es exitosa le decimos al router que nos lleve al path home relacionado con el componente HomeComponent, si no, nos vuelve al path login que es el path relacionado al LoginComponent.

Reload

También añadiremos las siguientes líneas de código que nos ayudarán a no perder datos cuando el usuario haga un refresh de la aplicación.

```
},
];

You, 3 hours ago | 2 authors (cfernandez and others)
@NgModule({
  imports: [RouterModule.forRoot(routes, { onSameUrlNavigation: 'ignore', useHash: true })],
  exports: [RouterModule]
})
export class AppRoutingModule { }
```

Favicon.ico

Procuraremos cambiar el favicon.ico de la aplicación ya que por defecto angular nos genera un favicon.ico con el logotipo de angular

