

Componentes Angular con conexión a Firestone

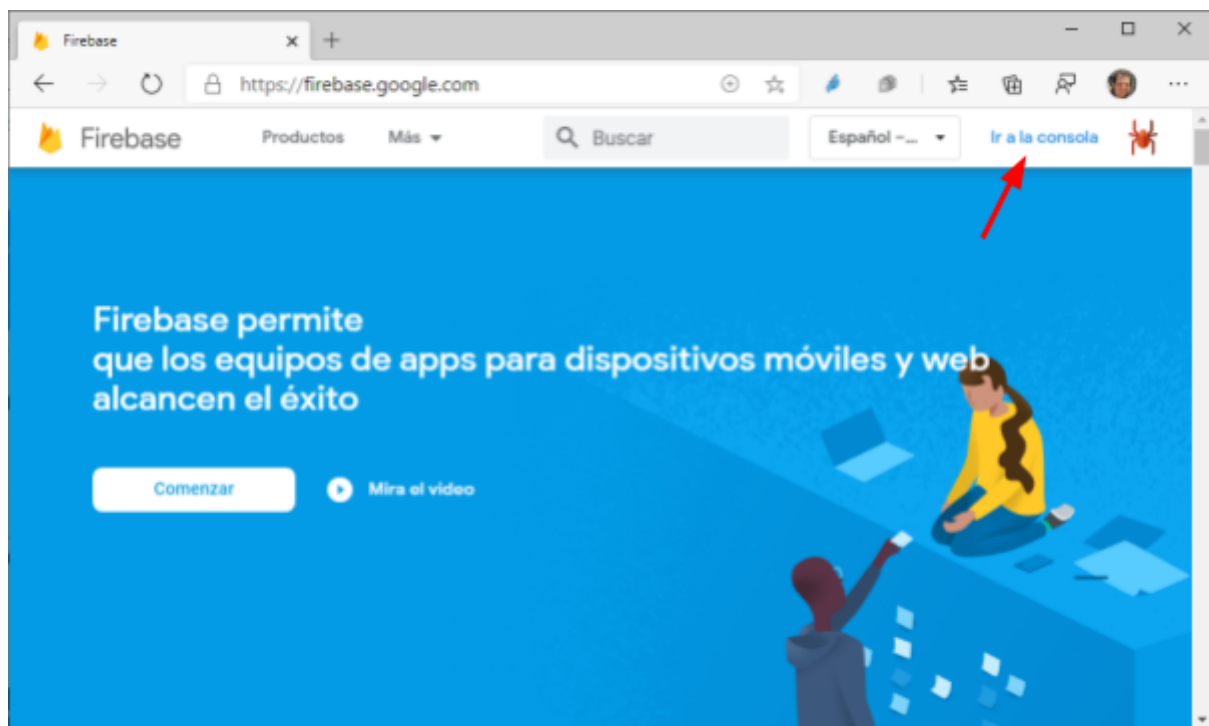
Base de Datos NoSQL de Firebase

Paso 1: Sincronizar Firebase con el proyecto de Angular

La conexión de confianza entre la cuenta de Firebase y el proyecto Angular se establece por la configuración del ambiente, es decir, el conjunto de claves públicas y privadas de servicios de internet.

Para este fin, debemos abrir la consola de Firebase en el proyecto de trabajamos.

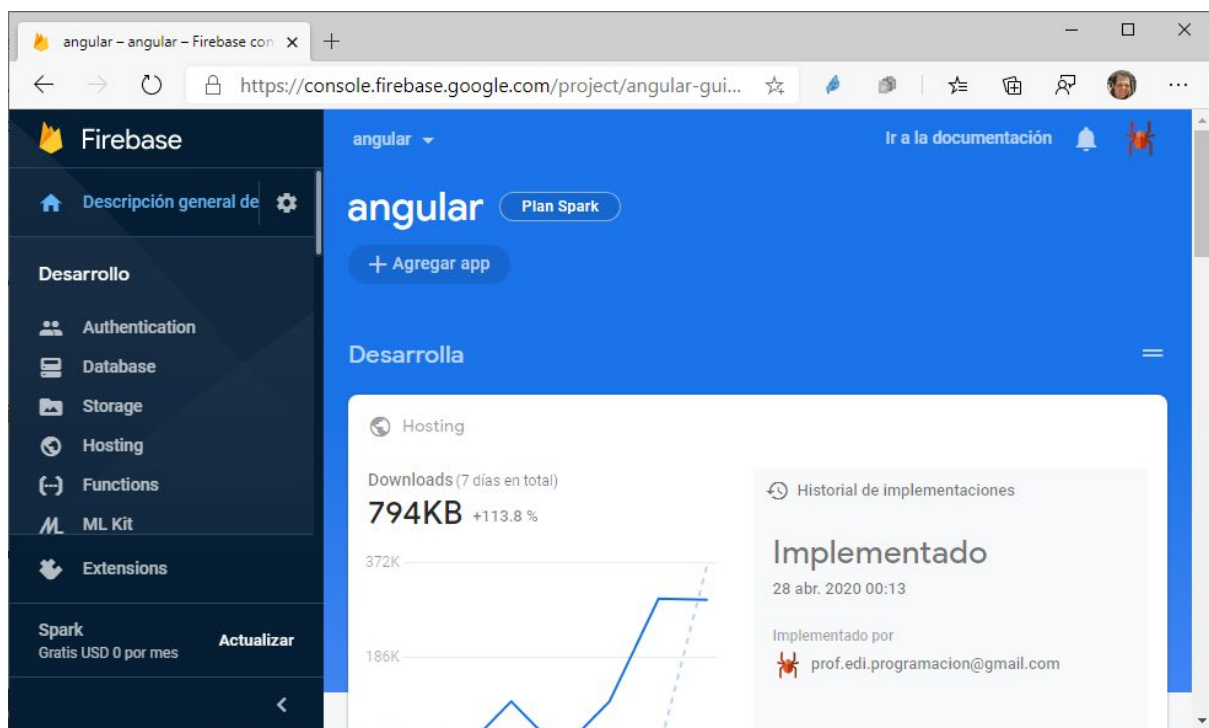
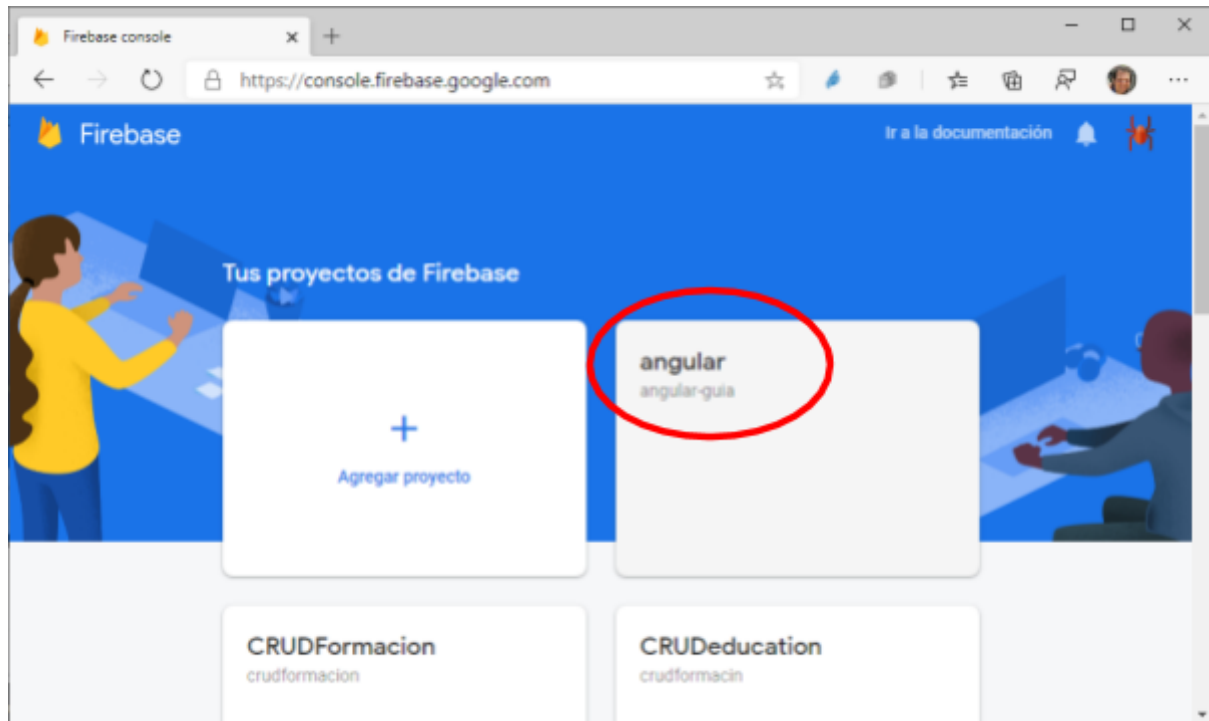
Recordar, la ruta es **firebase.google.com** y luego en la esquina superior derecha está el botón de “ir a consola”



En el caso de la guía, el proyecto de trabajo se llama angular

Angular y Firebase

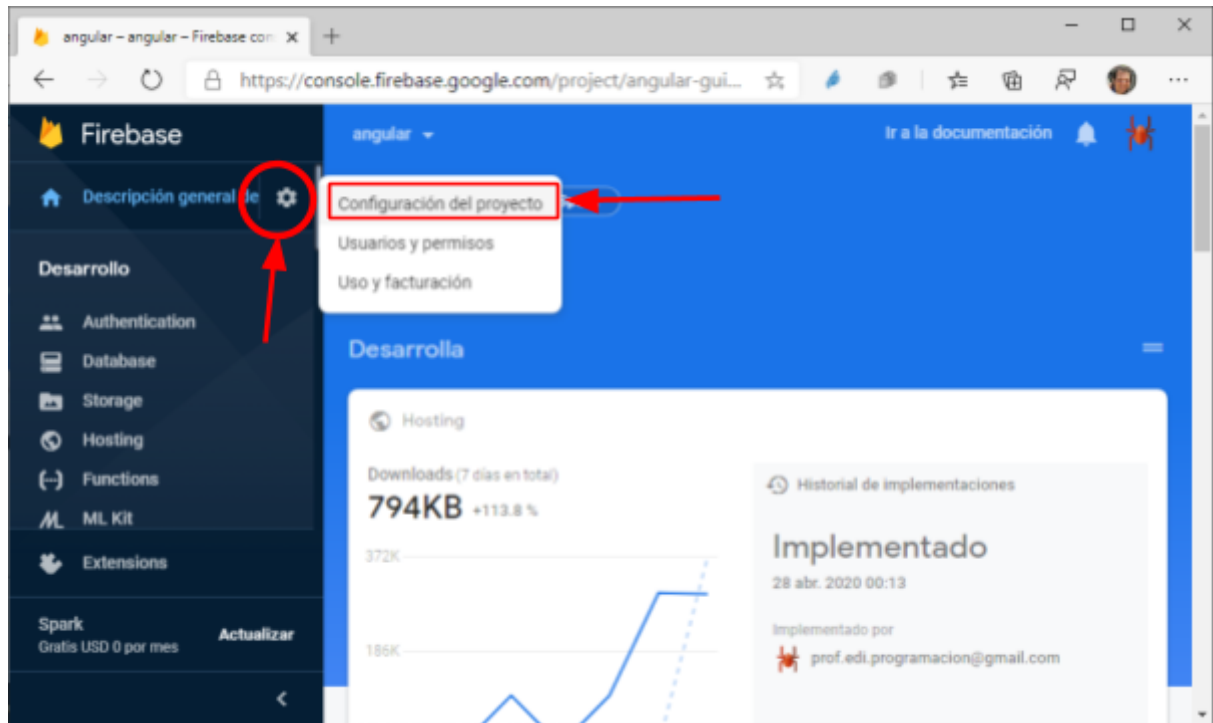
Proyecto Angular



Para obtener las clase del ambiente, debemos ir a la configuración del proyecto

Angular y Firebase

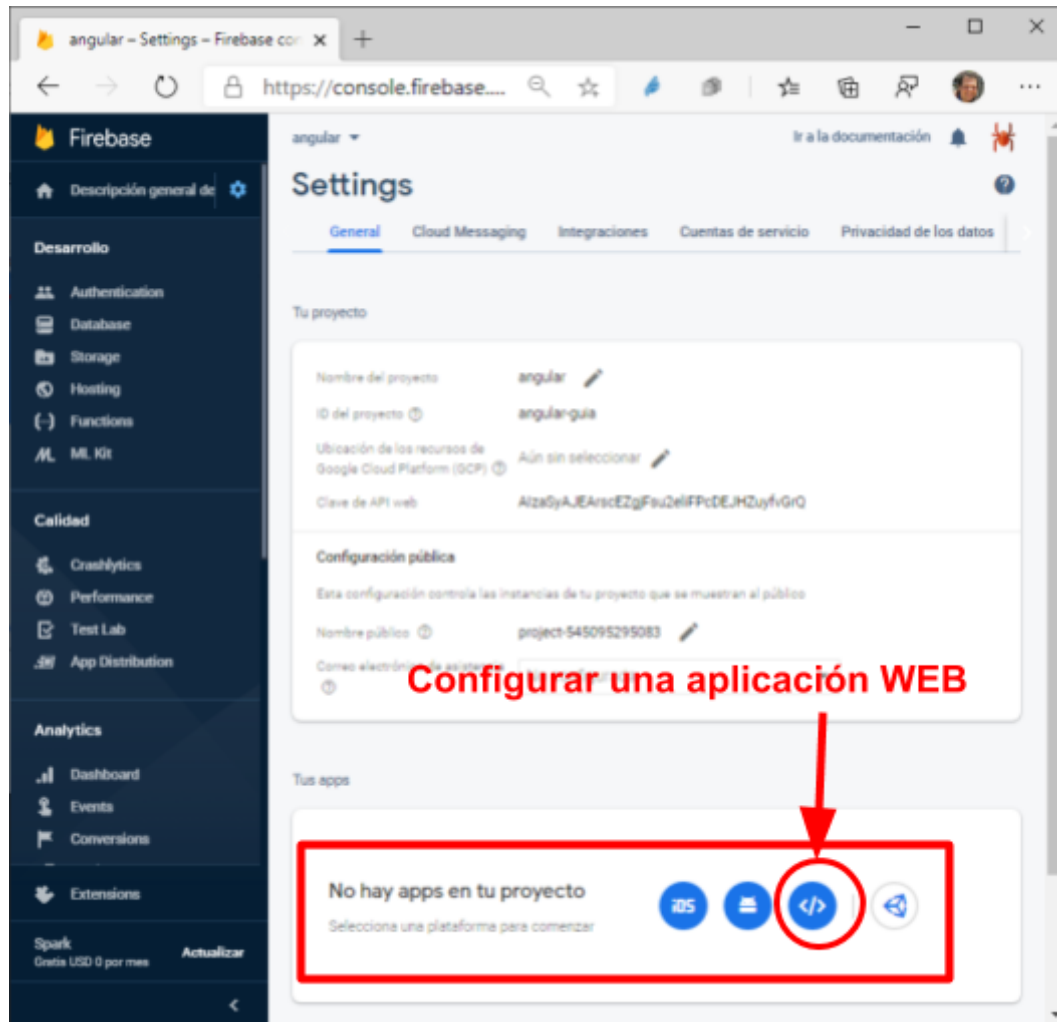
Proyecto Angular



Luego, debemos configurar una relación con una apps, y seleccionamos la opción para una aplicación WEB.

Angular y Firebase

Proyecto Angular



Debemos escribir un sobrenombre a la aplicación, este sobrenombre o identificador es de uso interno, los usuarios de la aplicación no lo visualizan.

Un proyecto en Firebase contiene los servicios de identificación, base de datos, hosting, etc. y se comparten con varias aplicaciones web, android, iOS, Unity, Python, Java, etc.

En esta guía usaremos una aplicación web.

Angular y Firebase

Proyecto Angular

angular - Settings - Firebase con x +

← → ↻ 🔒 https://console.firebase.... 🔍 ☆

Agregar una app web

1 2

Registrar app Agregar el SDK de Firebase

Sobrenombre de la app ⓘ

listaCompra ← Identificador interno de la aplicación

☐ Además, configura **Firebase Hosting** para esta app. [Más información](#) ⓘ

También puedes configurarlo más adelante. Puedes comenzar a usarlo sin costo en cualquier momento.

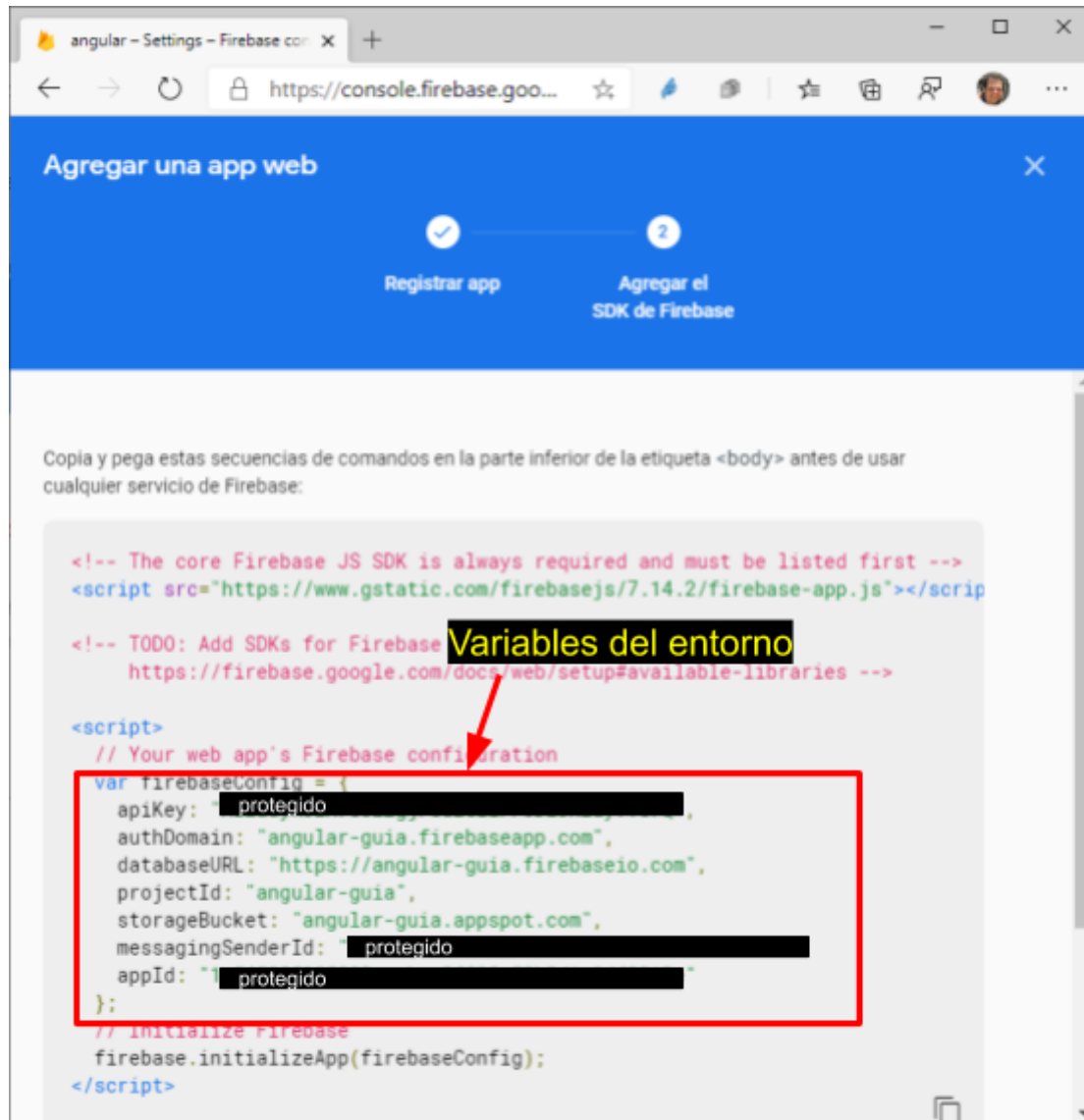
Registrar app

Registramos la aplicación, no es necesario adicionar hosting, porque ya lo hemos creado con el proceso de deploy de Angular.

Firebase nos muestra la configuración que necesitamos para enlazar el proyecto Angular con la aplicación en Firebase

Angular y Firebase

Proyecto Angular



Las variables del entorno se compone de:

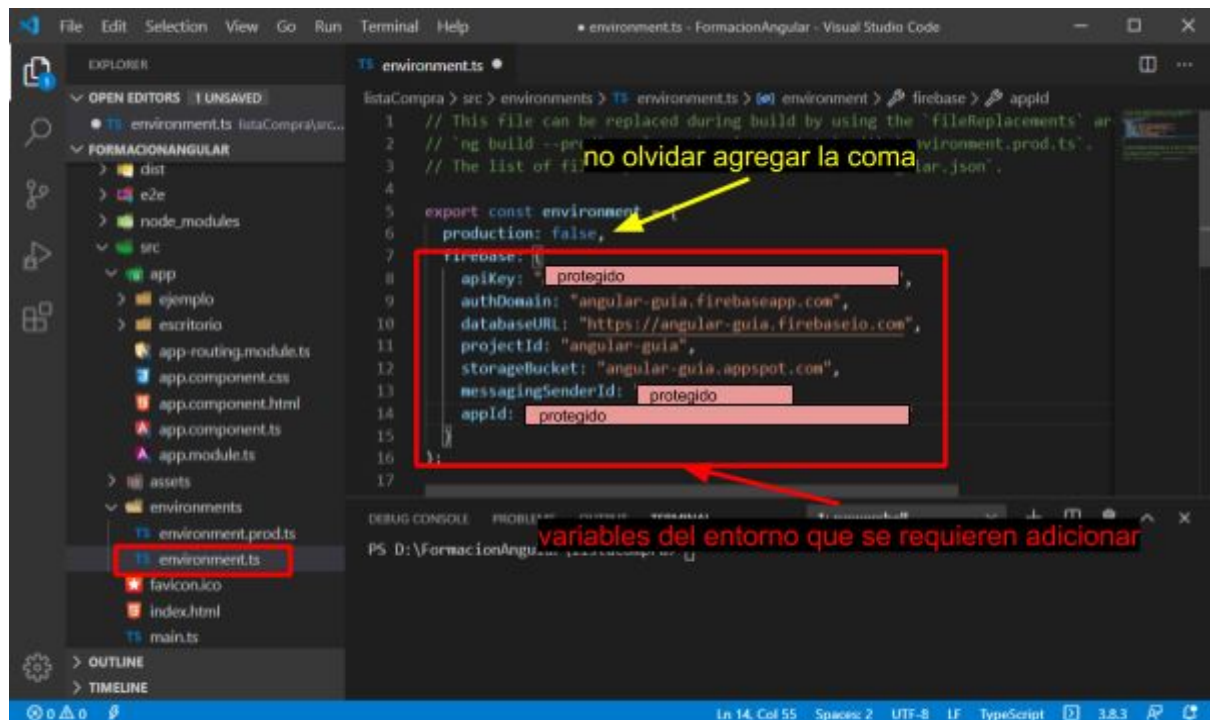
- apiKey
- authDomain
- databaseURL
- projectId
- storageBucket
- messagingSenderId
- appId

Esta configuración con las rutas y las claves propias del proyecto se deben copiar en el proyecto de Angular, en el archivo ...\\listaCompra\\src\\environments\\environment.ts

Se recomienda proteger las claves para hacer uso del proyecto, por esta razón, no se presentan y se sustituyen en esta guía por un cuadro negro.

Angular y Firebase

Proyecto Angular



...\listaCompra\src\environments\environment.ts

```
export const environment = {
  production: false,
  firebase: {
    apiKey: "Tu apiKey",
    authDomain: "Tu authDomain",
    databaseURL: "Tu databaseURL",
    projectId: "Tu projectId",
    storageBucket: "Tu storageBucket",
    messagingSenderId: "Tu messagingSenderId",
    appId: "Tu appId"
  }
};
```

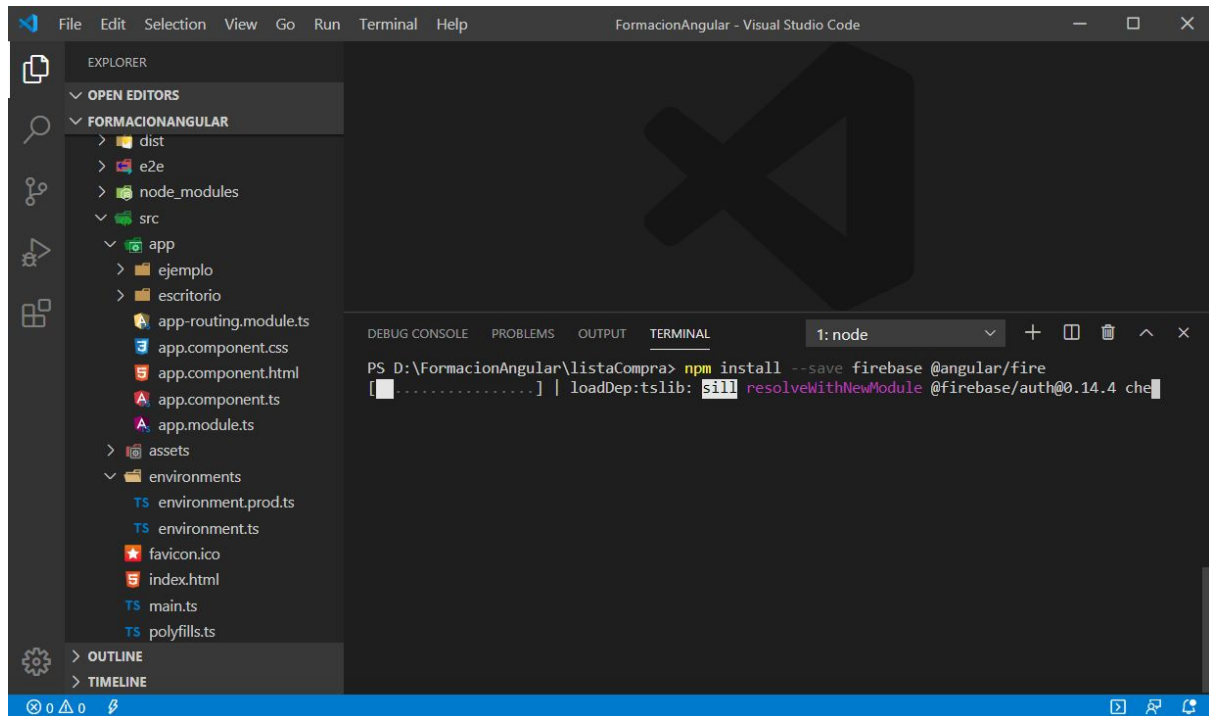
Ahora, debemos incorporar el módulo de Firebase al proyecto en Angular para tener las funciones que permiten hacer las diferentes actividades.

La instrucción para instalar el módulo firebase en un proyecto de Angular es

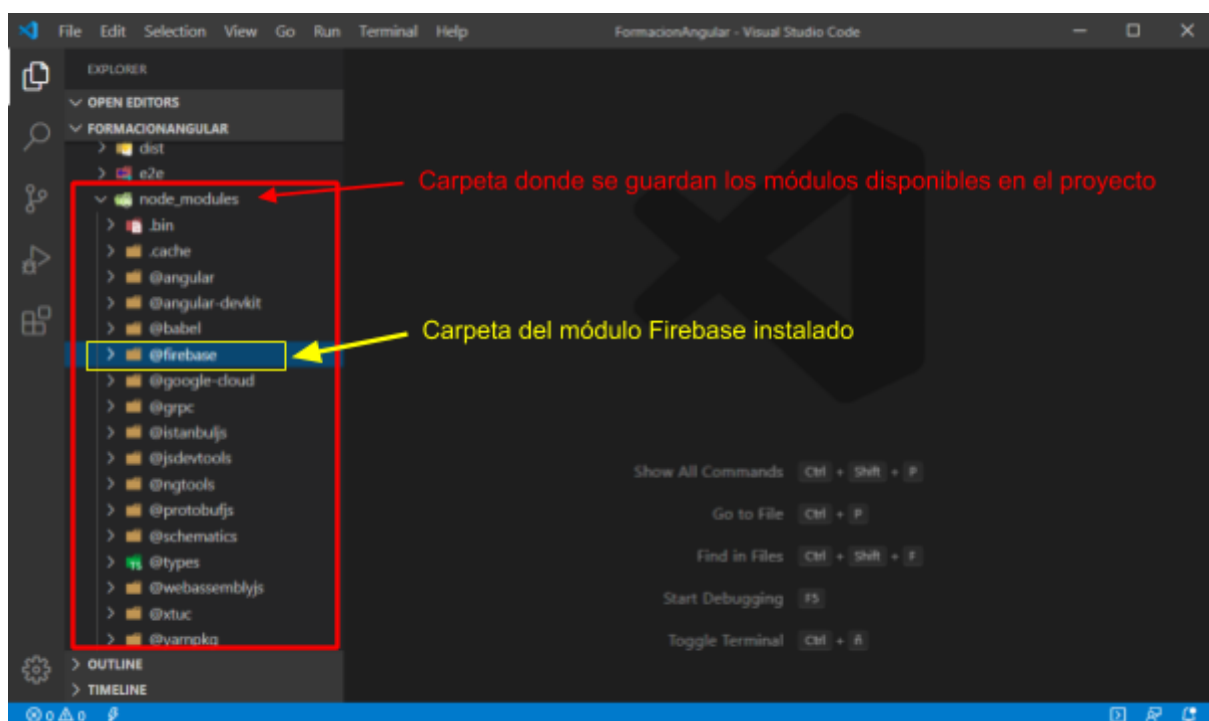
```
npm install --save firebase @angular/fire
```

Angular y Firebase

Proyecto Angular



Para verificar si quedó instalado en el proyecto, se busca las carpetas **@firebase** y **firebase** en la carpeta **node_modules**, si aparece, el módulo ha quedado bien instalado.



Para que el módulo sea visible desde en los componentes y servicios, se debe vincular al archivo `.../src/app/app.module.ts`

...\listaCompra\src\app\app.module.ts

```
import { BrowserModule } from '@angular/platform-browser';
import { NgModule } from '@angular/core';

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

//---[Firebase]-----
import { AngularFireModule } from '@angular/fire';
import { AngularFireDatabaseModule } from '@angular/fire/database';

import { environment } from '../environments/environment';

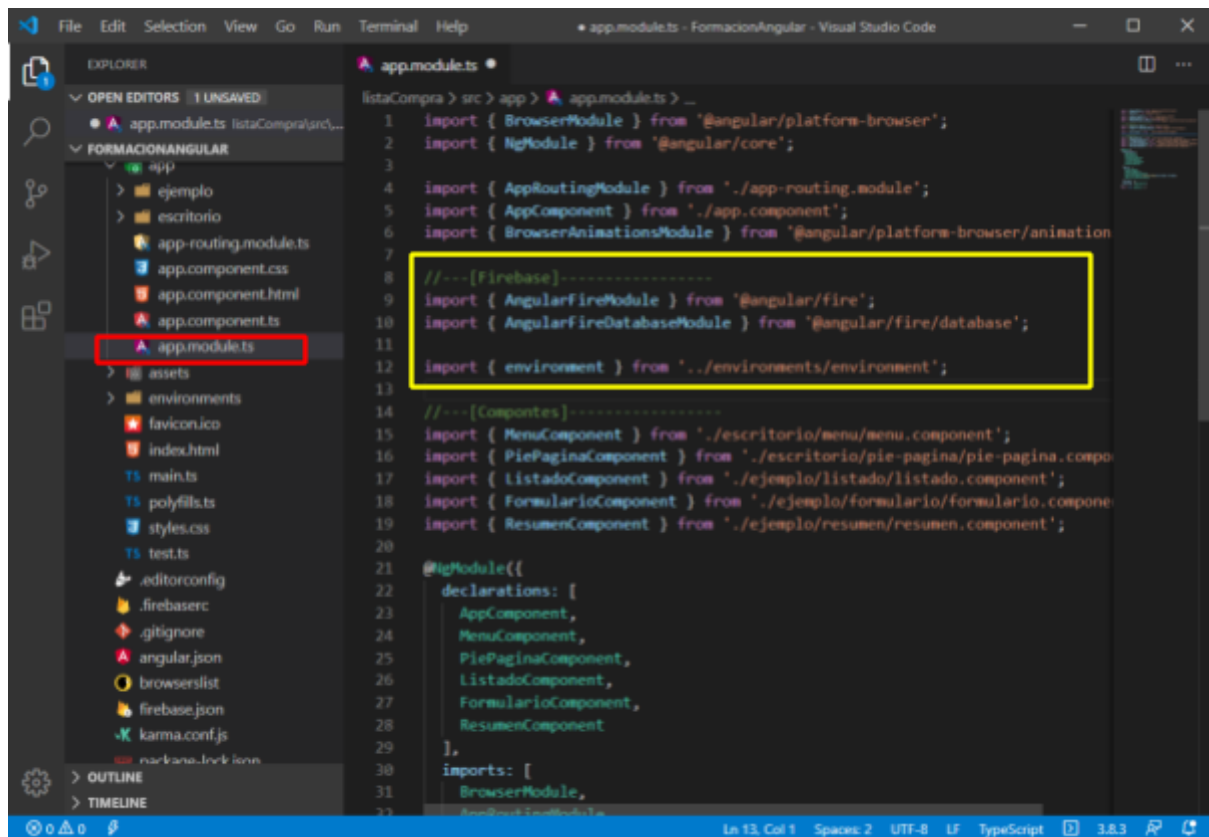
//---[Componetes]-----
import { MenuComponent } from './escritorio/menu/menu.component';
import { PiePaginaComponent } from
 './escritorio/pie-pagina/pie-pagina.component';
import { ListadoComponent } from
 './ejemplo/listado/listado.component';
import { FormularioComponent } from
 './ejemplo/formulario/formulario.component';
import { ResumenComponent } from
 './ejemplo/resumen/resumen.component';

@NgModule({
  declarations: [
    AppComponent,
    MenuComponent,
    PiePaginaComponent,
    ListadoComponent,
    FormularioComponent,
    ResumenComponent
  ],
  imports: [
    BrowserModule,
```

Angular y Firebase

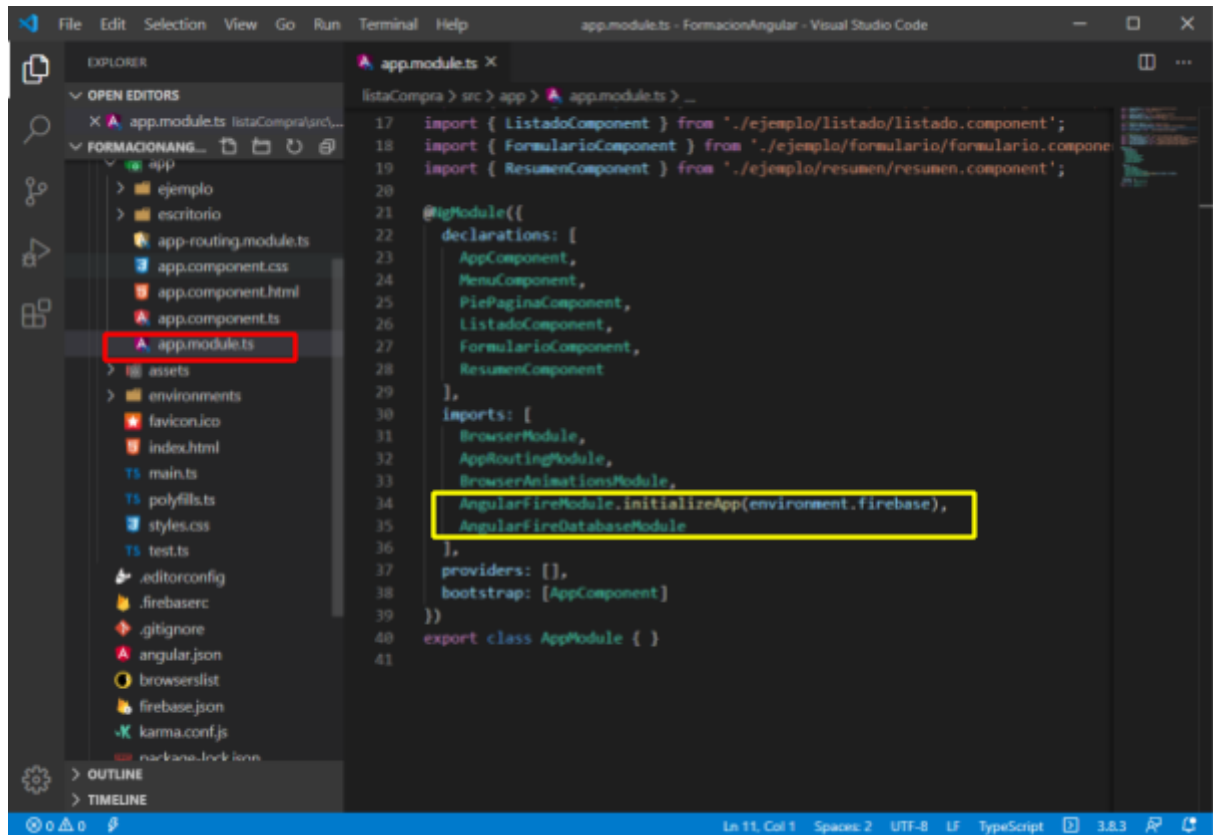
Proyecto Angular

```
    AppRoutingModule,  
    BrowserAnimationsModule,  
    AngularFireModule.initializeApp(environment.firebase),  
    AngularFireDatabaseModule  
  ],  
  providers: [],  
  bootstrap: [AppComponent]  
})  
  
export class AppModule { }
```



Angular y Firebase

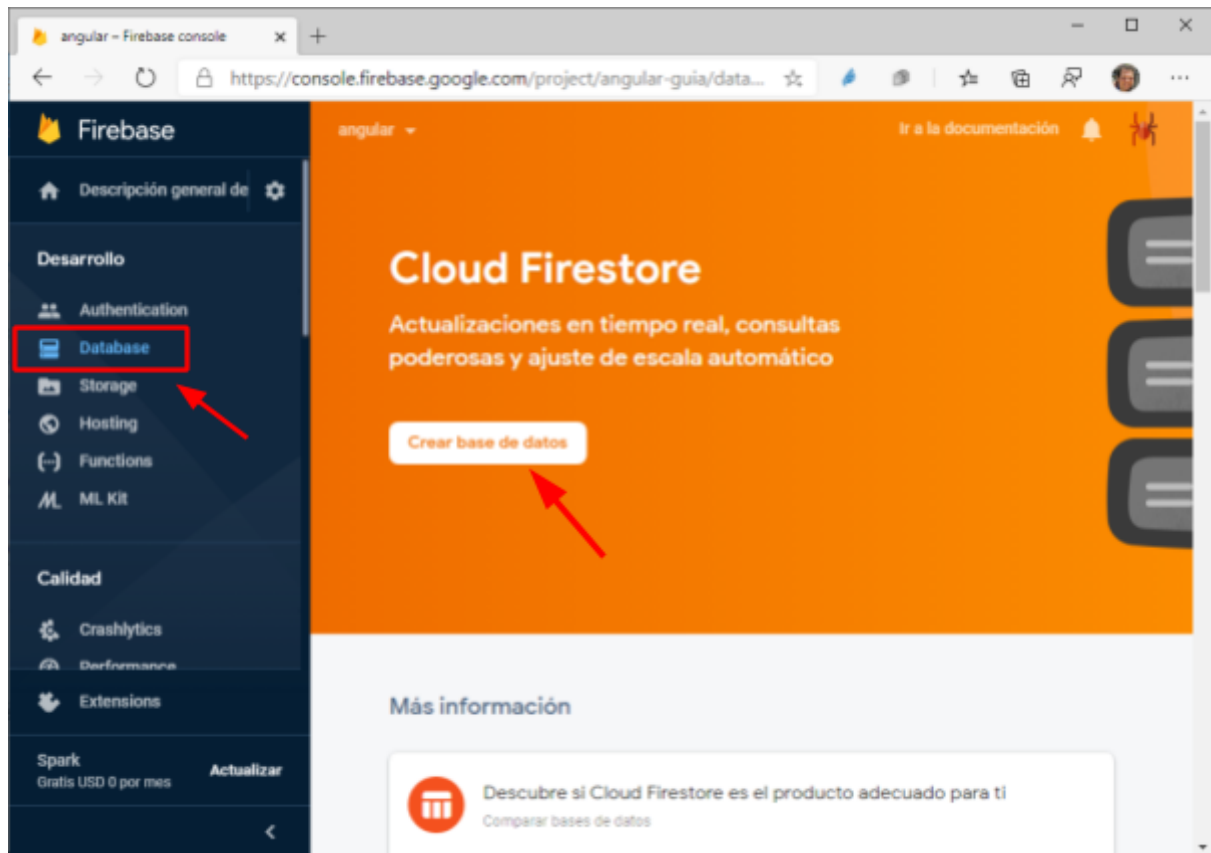
Proyecto Angular



Ya está el proyecto de Angular configurado y enlazado con la infraestructura Firebase como servicio en la nube.

Paso 2: Activar la base de datos en Firebase

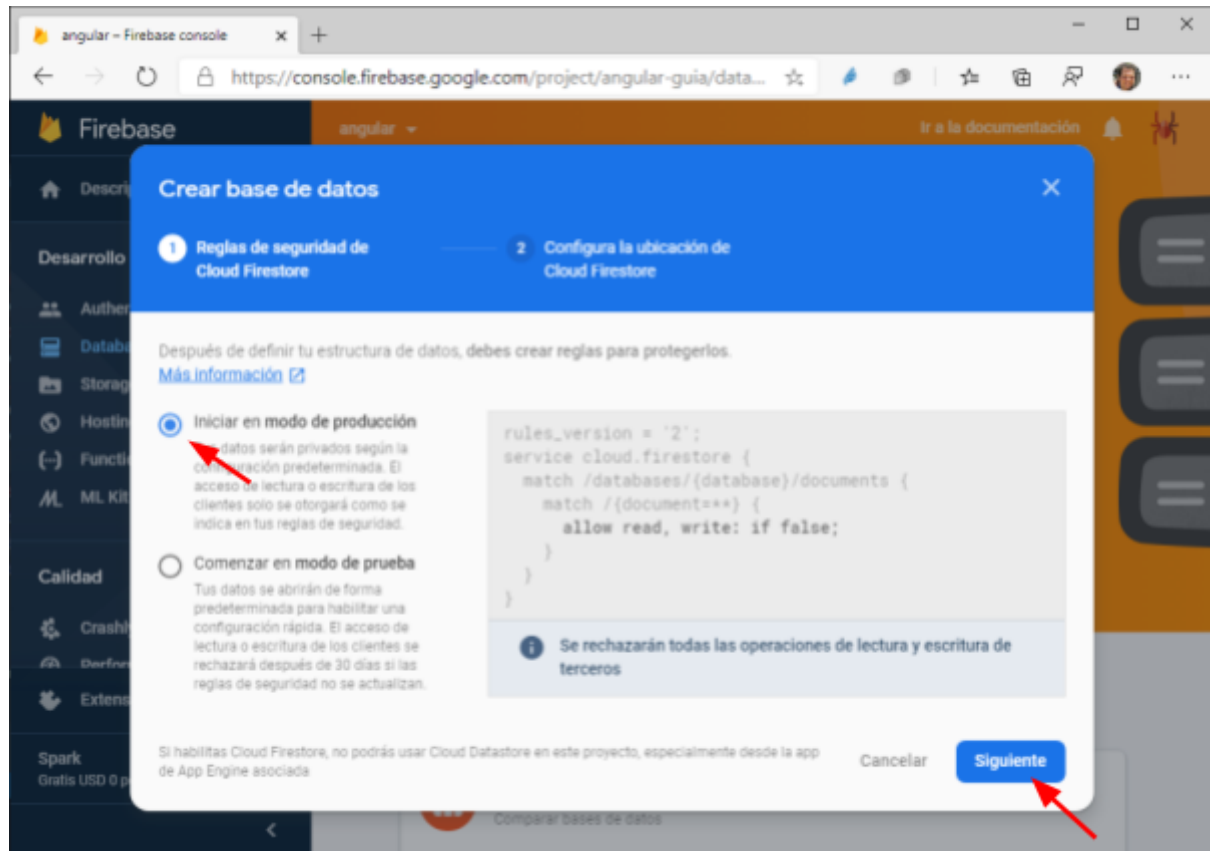
Para activar la base de datos debemos ingresar a la consola de Firebase con nuestra cuenta y dirigirnos a la opción Database y una vez aparece la página de Cloud Firestore o similar, nos dirigimos a crear base de datos y le damos click.



Para esta guía podemos activar la opción iniciar en modo producción, así estará activa la base datos sin vencimiento

Angular y Firebase

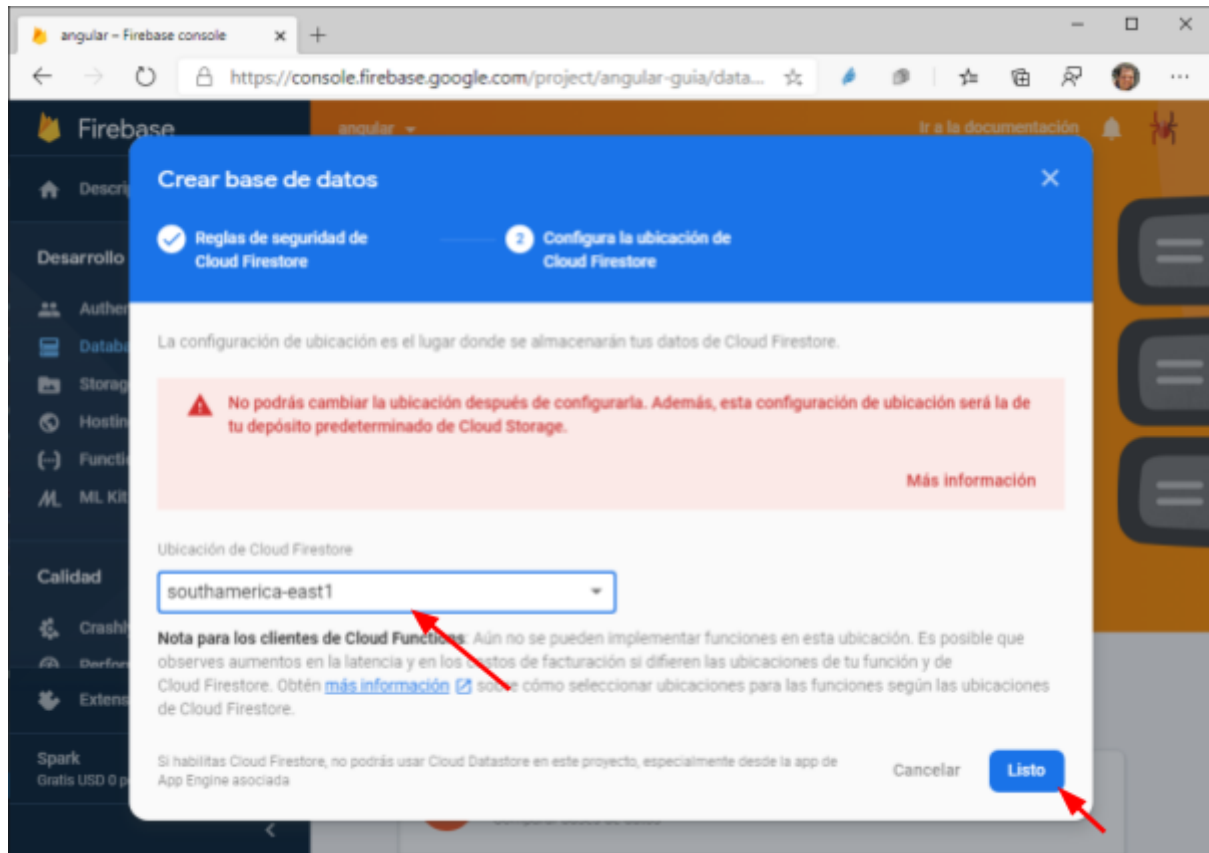
Proyecto Angular



Debemos escoger un servidor de google en alguna región del mundo, la opción es indiferente.

Angular y Firebase

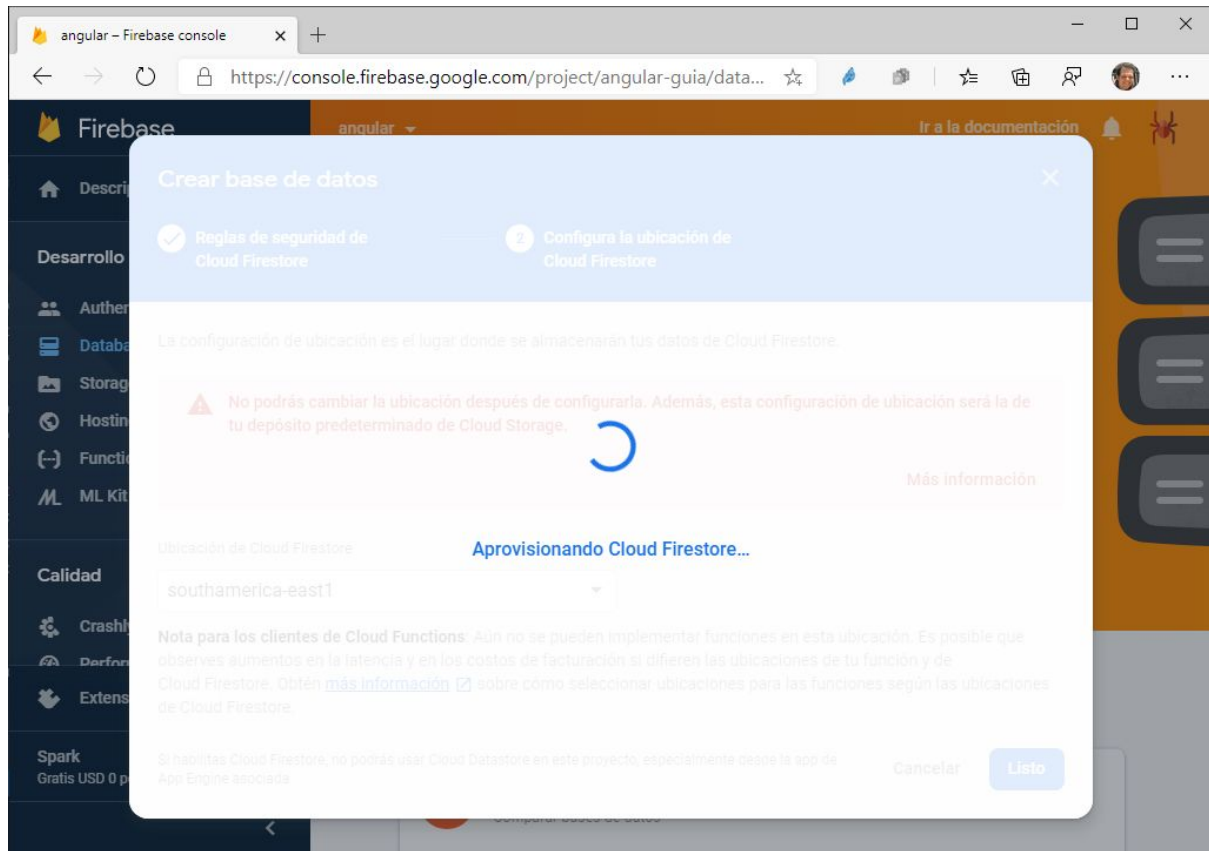
Proyecto Angular



Firebase se encargará de relacionar el servicio en nuestra cuenta de firebase

Angular y Firebase

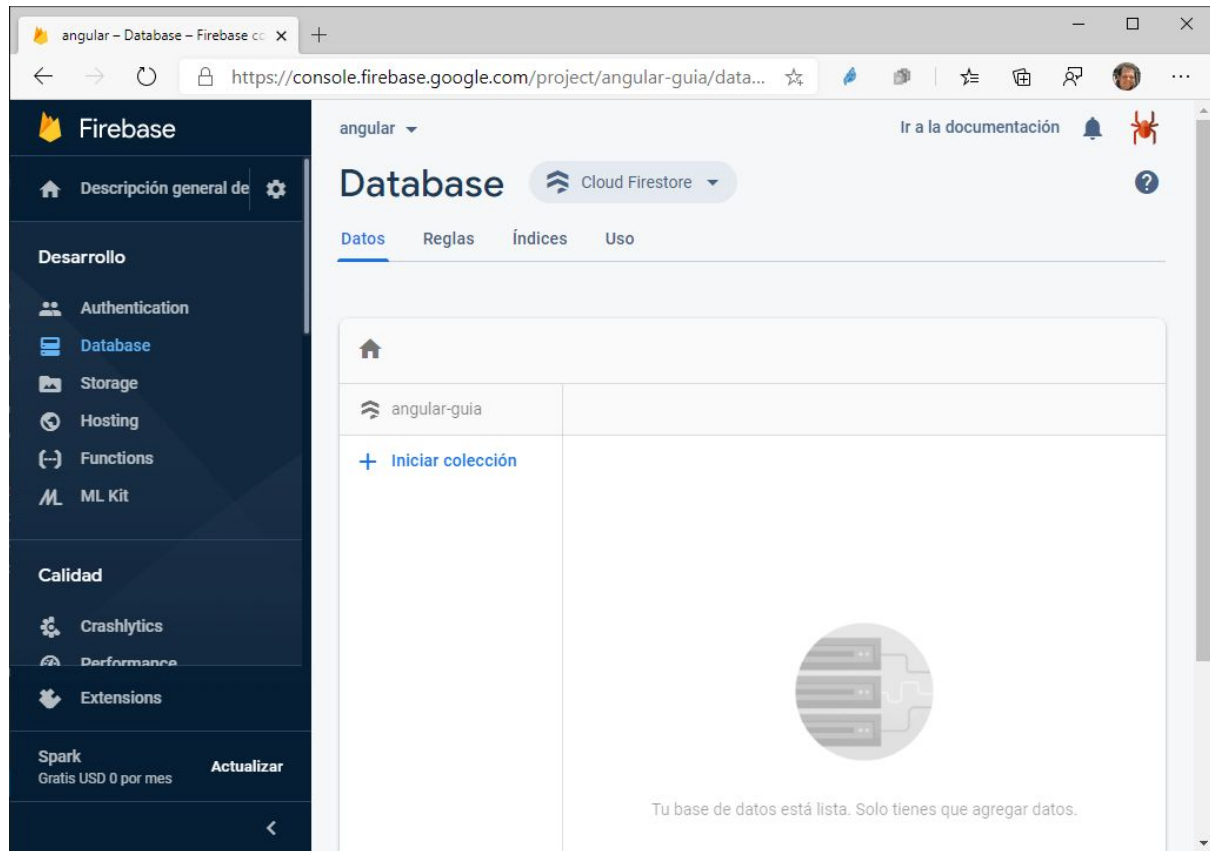
Proyecto Angular



Termina con el despliegue de una ventana que permite gestionar y monitorear los datos.

Angular y Firebase

Proyecto Angular



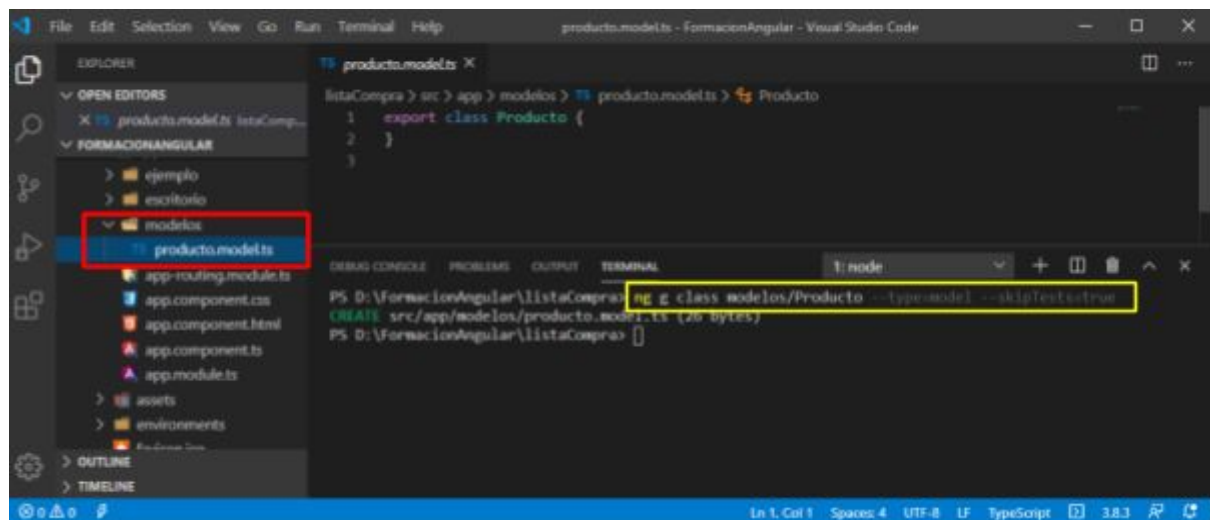
Paso 3: Modelo de datos y creación del servicio CRUD de datos

En las estructuras de datos NoSQL, la definición de los objetos o modelos de datos son los que definen la estructura de los registros, los cuales se almacenan en una colección.

En Angular para crear esta estructura de datos, requerimos tener un elemento tipo modelo.

La instrucción `ng g class modelos/Producto --type=model --skipTests=true`

Más información en: <https://angular.io/cli/generate#class>



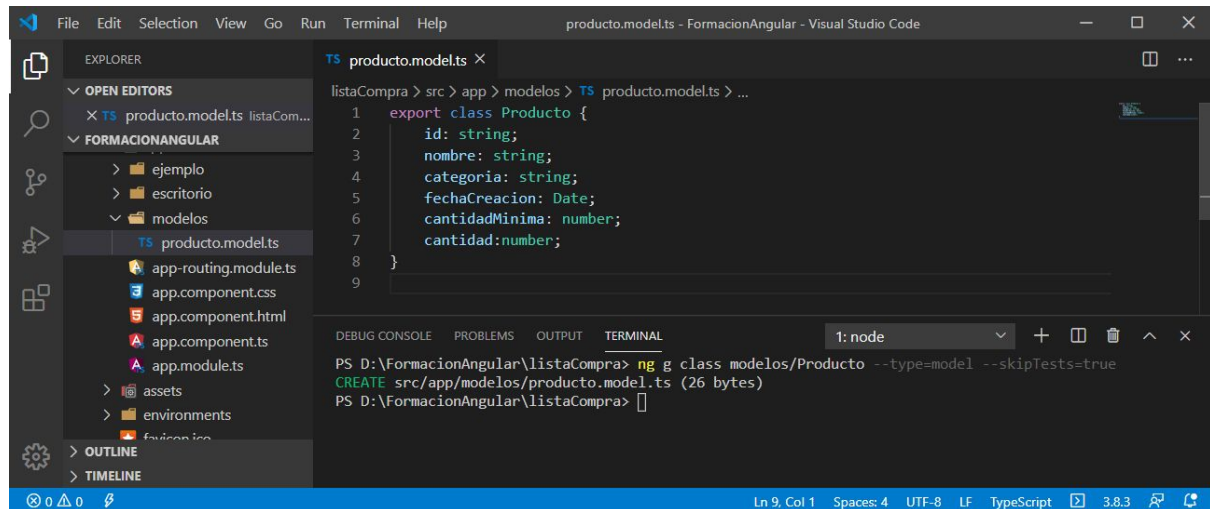
y en el archivo `..\src\app\modelos\producto.model.ts` escribir la estructura del registro, como ejemplo en esta guía, el registro de producto.

...\listaCompra\src\app\modelos\producto.model.ts

```
export class Producto {  
  id: string;  
  nombre: string;  
  categoria: string;  
  fechaCreacion: Date;  
  cantidadMinima: number;  
  cantidad: number;  
}
```

Angular y Firebase

Proyecto Angular



Creación del servicio CRUD

Un servicio CRUD es el conjunto de operaciones que genera acciones sobre la base de datos y la aplicación Angular.

Las funciones son:

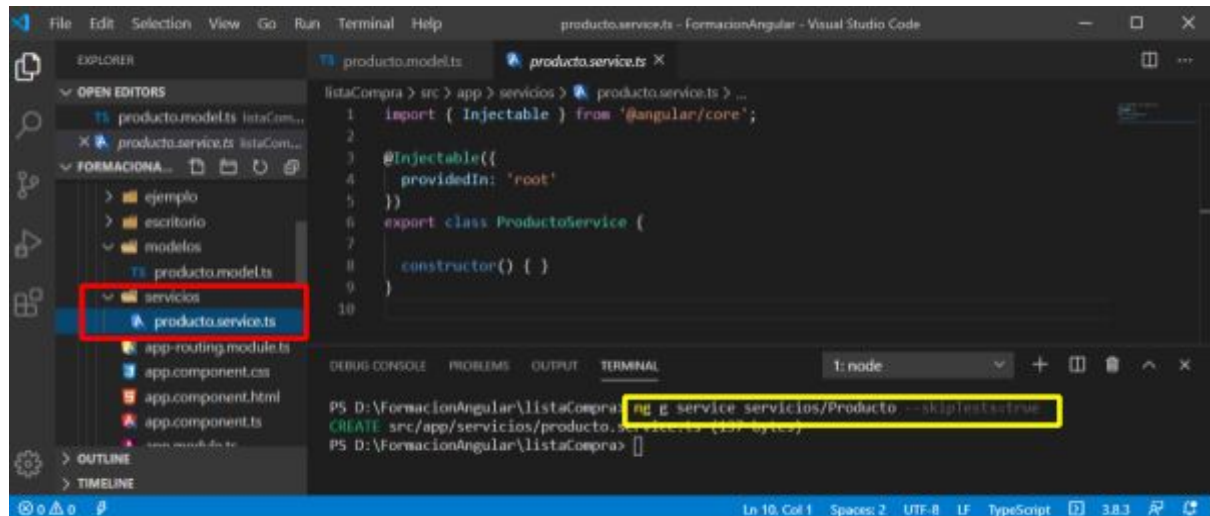
- **createProducto**
Crea un nuevo documentos (registro) en la base de datos
- **getProductos**
Consulta los documentos (registros) en la base de datos
- **updateProducto**
Actualiza los valores de los campos de un documento (registro) en la base de datos
- **deleteProducto**
Elimina un documento (registro) en la base de datos

La instrucción para crear un servicio es `ng g service servicios/Producto --skipTests=true`

Más información en <https://angular.io/cli/generate#service-command>

Angular y Firebase

Proyecto Angular



...\listaCompra\src\app\servicios\producto.service.ts

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

import { AngularFireStore } from '@angular/fire/firestore';
import { Producto } from 'src/app/modelos/producto.model';

@Injectable({
  providedIn: 'root'
})
export class ProductoService {

  constructor(private firestore: AngularFireStore) { }

  //---[create]-----
  createProducto(producto: Producto) {
    return this.firestore.collection('productos').add(producto);
  }

  //---[read]-----
  getProductos() {
    return this.firestore.collection('productos').snapshotChanges();
  }
}
```

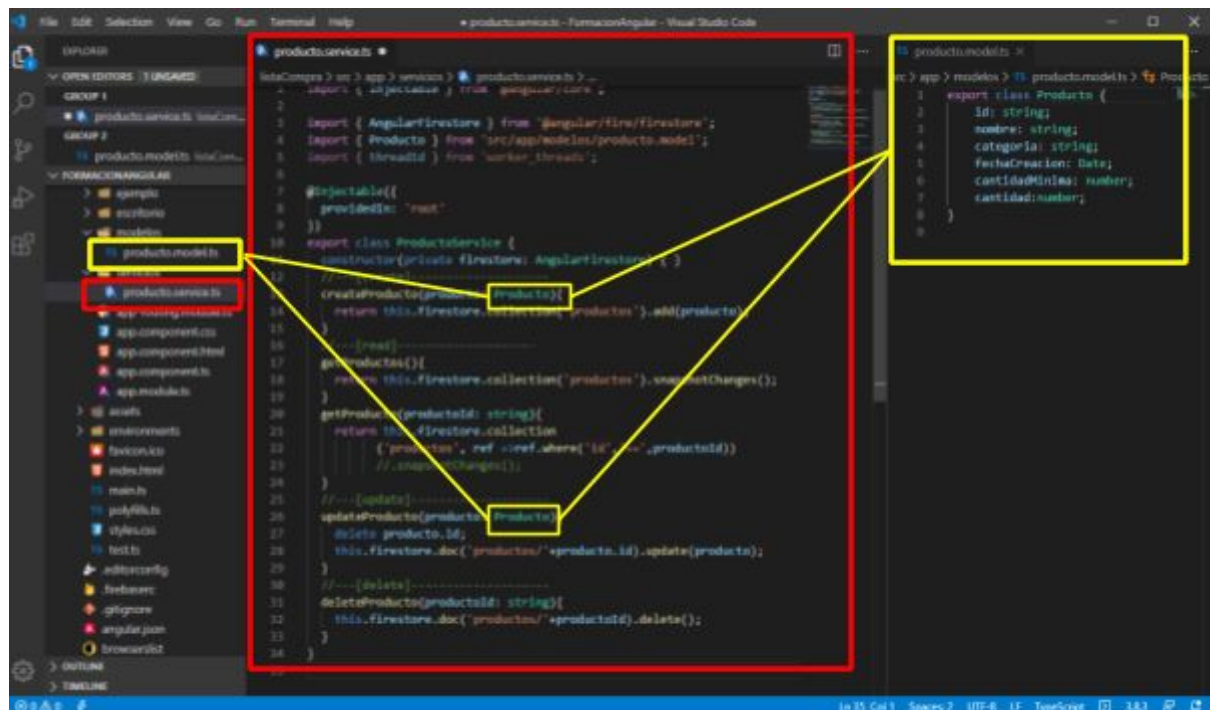
Angular y Firebase

Proyecto Angular

```
getProducto(productoId: string) {
    return this.firestore.collection
        ('productos', ref =>ref.where('id','==',productoId))
}

//---[update]-----
updateProducto(producto: Producto) {
    delete producto.id;
    this.firestore.doc('productos/'+producto.id).update(producto);
}

//---[delete]-----
deleteProducto(productoId: string) {
    this.firestore.doc('productos/'+productoId).delete();
}
}
```



Una vez se tienen los servicios, se crean los componentes para el formulario y la lista de registros.

Angular y Firebase

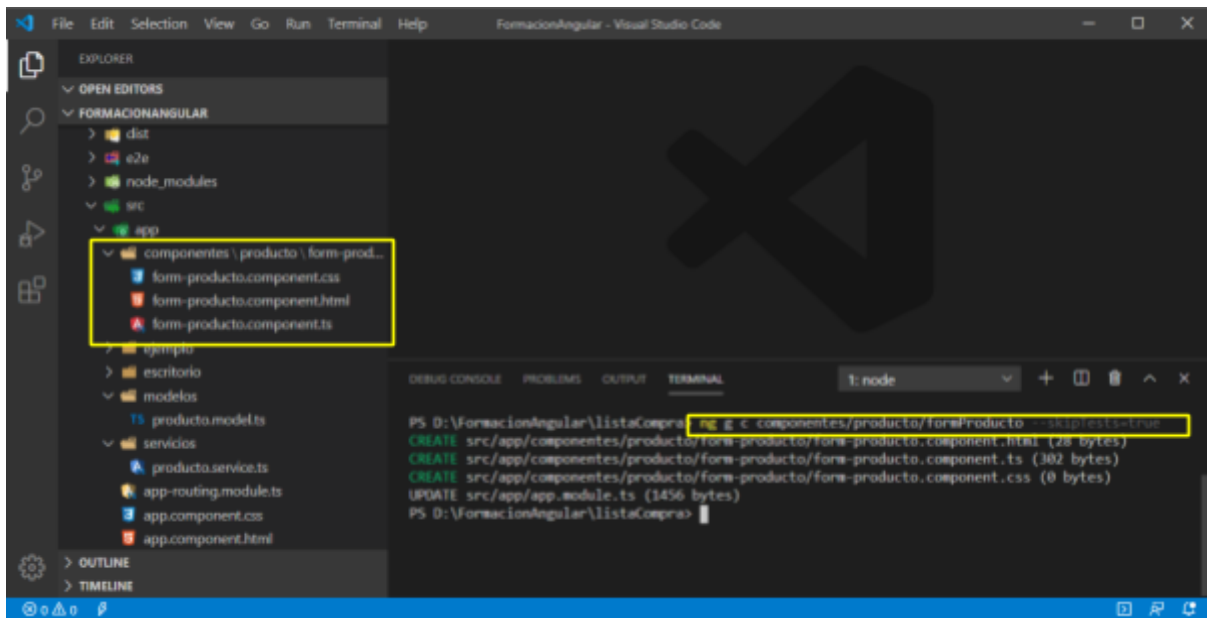
Proyecto Angular

Iniciemos con el formulario para ingresar datos, en esta guía crearemos un grupo llamado producto, y en este grupo crearemos los siguientes componentes:

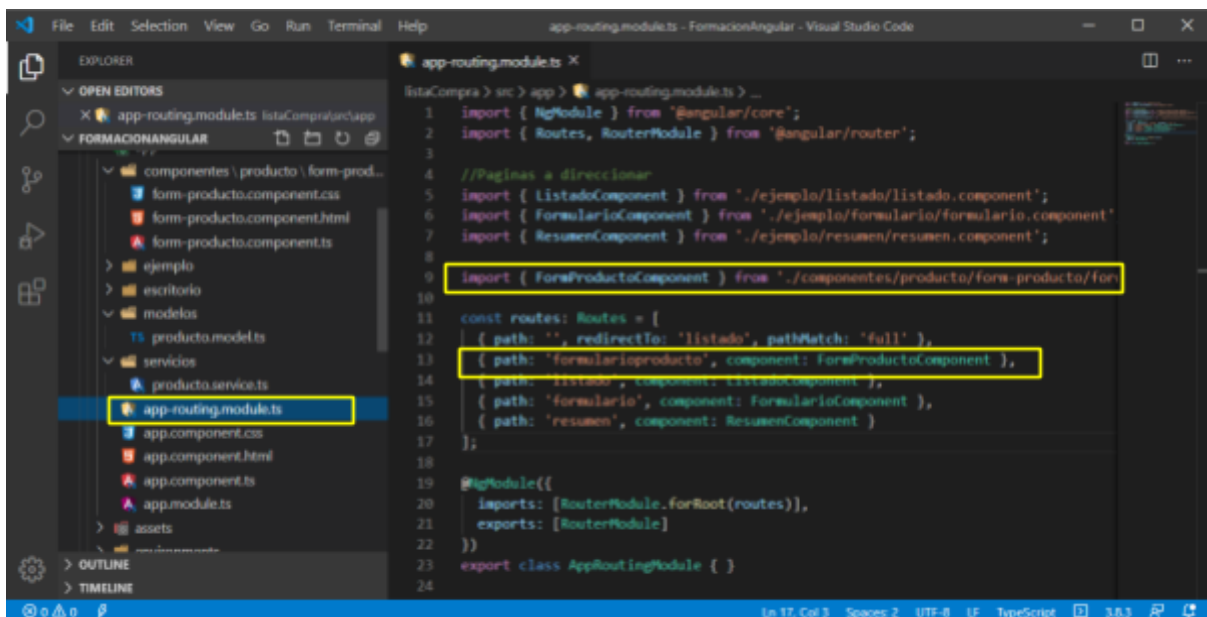
- formulario
nuevos productos o actualizar un producto
- listado
lista de los productos actuales en la colección

Para crear el componente formulario, la instrucción es:

```
ng g c componentes/producto/formProducto --skipTests=true
```



Para que el componente sea visible en la ruta URL lo adicionamos en el routing



...\listaCompra\src\app\app-routing.module.ts

```
import { NgModule } from '@angular/core';
import { Routes, RouterModule } from '@angular/router';

//Paginas a direccionar
import { ListadoComponent } from
'./ejemplo/listado/listado.component';
import { FormularioComponent } from
'./ejemplo/formulario/formulario.component';
import { ResumenComponent } from
'./ejemplo/resumen/resumen.component';

import { FormProductoComponent } from
'./componentes/producto/form-producto/form-producto.component';

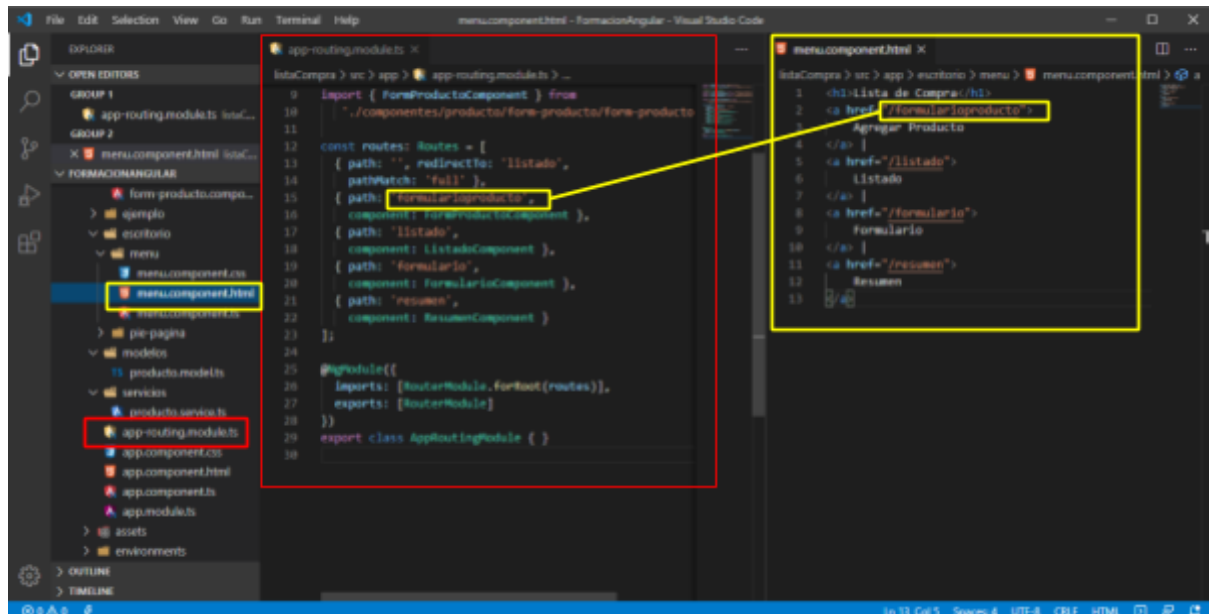
const routes: Routes = [
  { path: '', redirectTo: 'listado', pathMatch: 'full' },
  { path: 'formularioproducto', component: FormProductoComponent },
  { path: 'listado', component: ListadoComponent },
  { path: 'formulario', component: FormularioComponent },
  { path: 'resumen', component: ResumenComponent }
];

@NgModule({
  imports: [RouterModule.forRoot(routes)],
  exports: [RouterModule]
})
export class AppRoutingModule { }
```

y actualizamos el componente menú.

Angular y Firebase

Proyecto Angular



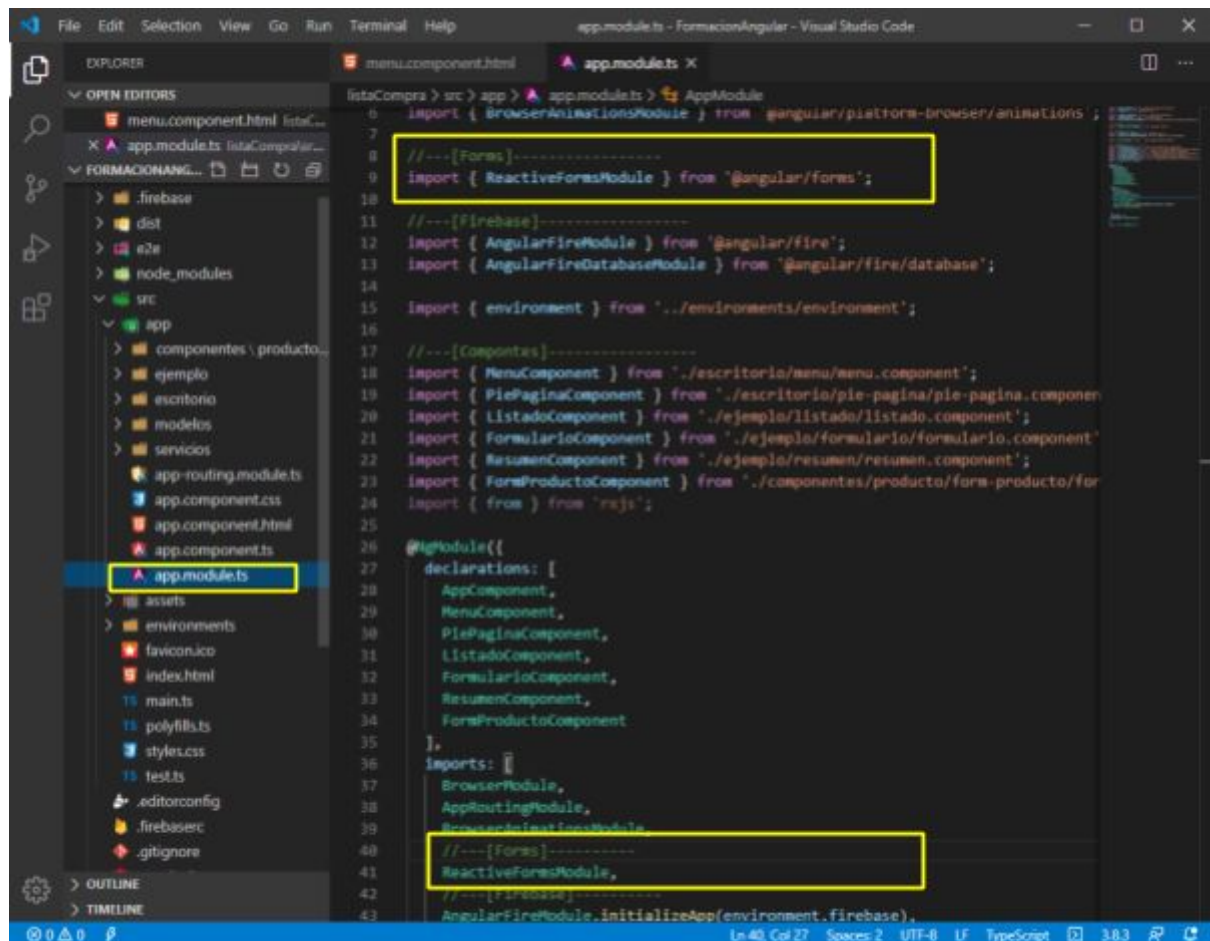
...\\listaCompra\\src\\app\\escritorio\\menu\\menu.component.html

```
<h1>Lista de Compra</h1>
<a href="/formularioproducto">Agregar Producto</a> |
<a href="/listado">Listado</a> |
<a href="/formulario">Formulario</a> |
<a href="/resumen">Resumen</a>
```

Para relacionar los módulos de formulario al proyecto Angular, se debe modificar el archivo module.ts. En esta guía, el archivo module.ts es ...\\listaCompra\\src\\app\\app.module.ts

Angular y Firebase

Proyecto Angular



...\listaCompra\src\app\app.module.ts

```
import { BrowserModule } from '@angular/platform-browser';
import { NgModule } from '@angular/core';

import { AppRoutingModule } from './app-routing.module';
import { AppComponent } from './app.component';
import { MenuComponent } from './componentes/menu/menu.component';
import { PiePaginaComponent } from './componentes/pie-pagina/pie-pagina.component';
import { ListadoComponent } from './componentes/listado/listado.component';
import { FormularioComponent } from './componentes/formulario/formulario.component';
import { ResumenComponent } from './componentes/resumen/resumen.component';
import { FormProductoComponent } from './componentes/producto/form-producto/form-producto.component';

import { AngularFireModule } from '@angular/fire';
import { AngularFireDatabaseModule } from '@angular/fire/database';
import { environment } from '../environments/environment';

@NgModule({
  declarations: [
    AppComponent,
    MenuComponent,
    PiePaginaComponent,
    ListadoComponent,
    FormularioComponent,
    ResumenComponent,
    FormProductoComponent
  ],
  imports: [
    BrowserModule,
    AppRoutingModule,
    AngularFireModule,
    AngularFireDatabaseModule
  ],
  providers: [],
  bootstrap: [AppComponent]
})
export class AppModule {}
```



```
import { environment } from '../environments/environment';

//---[Componetes]-----
import { MenuComponent } from './escritorio/menu/menu.component';
import { PiePaginaComponent } from
 './escritorio/pie-pagina/pie-pagina.component';
import { ListadoComponent } from
 './ejemplo/listado/listado.component';
import { FormularioComponent } from
 './ejemplo/formulario/formulario.component';
import { ResumenComponent } from
 './ejemplo/resumen/resumen.component';
import { FormProductoComponent } from
 './componentes/producto/form-producto/form-producto.component';
import { from } from 'rxjs';

@NgModule({
  declarations: [
    AppComponent,
    MenuComponent,
    PiePaginaComponent,
    ListadoComponent,
    FormularioComponent,
    ResumenComponent,
    FormProductoComponent
  ],
  imports: [
    BrowserModule,
    AppRoutingModule,
    BrowserModuleAnimationsModule,
    //---[Forms]-----
    ReactiveFormsModule,
    //---[Firebase]-----
    AngularFireModule.initializeApp(environment.firebase),
    AngularFireDatabaseModule
  ],
  providers: [],
  bootstrap: [AppComponent]
})
export class AppModule {}
```

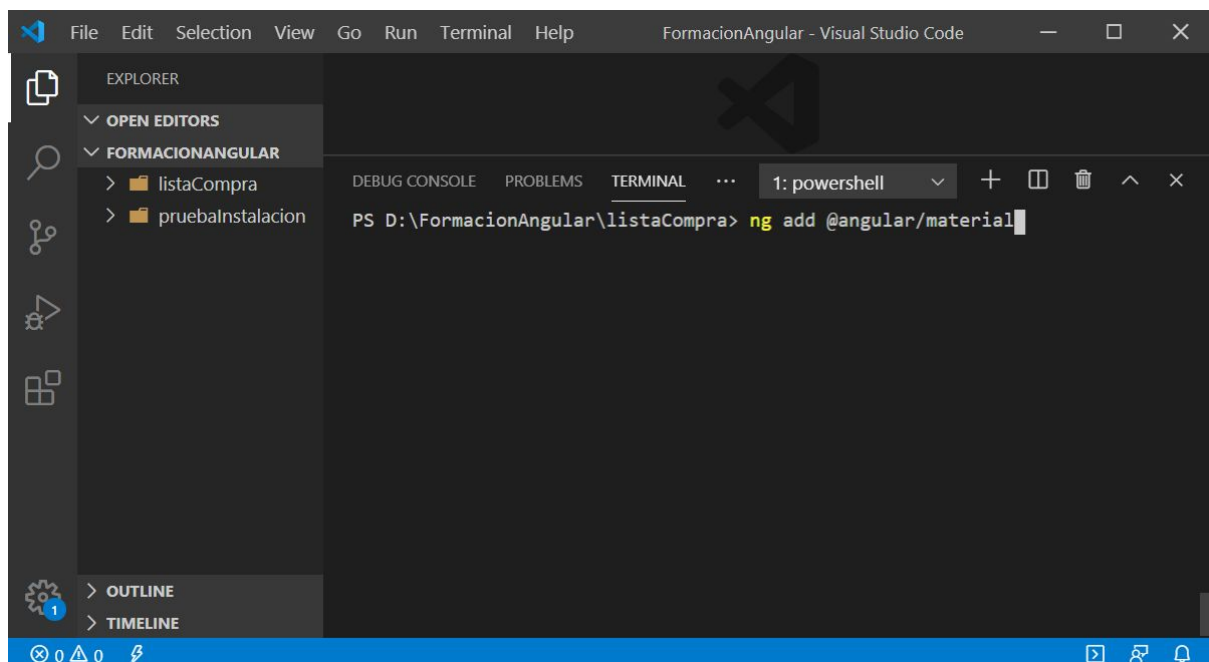
Angular y Firebase

Proyecto Angular

```
],  
  providers: [],  
  bootstrap: [AppComponent]  
})  
  
export class AppModule { }
```

Para la estética, se recomienda usar framework como el **Angular Material** { <https://material.angular.io/> }

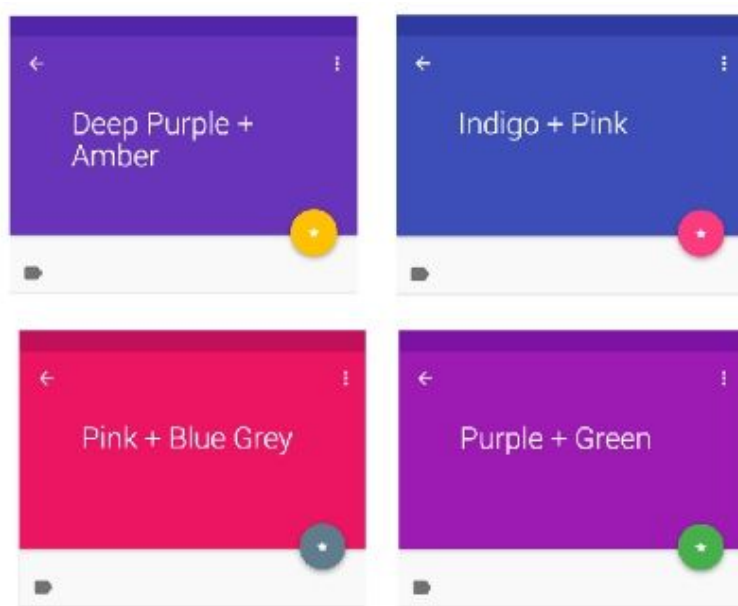
Para adicionar el Angular Material al proyecto de Angular, la instrucción es **ng add @angular/material**



Angular Material presenta varias configuraciones de temas de color, como:

Angular y Firebase

Proyecto Angular



Debes seleccionar uno o dejar la opción de custom para configurar el tema de colores desde el código.

A screenshot of the Visual Studio Code interface. The Explorer panel on the left shows a project named 'FormacionAngular' with two subfolders: 'listaCompra' and 'pruebaInstalacion'. The Terminal panel at the bottom shows the following command and output:

```
PS D:\FormacionAngular\listaCompra> ng add @angular/material
Installing packages for tooling via npm.
Installed packages for tooling via npm.
? Choose a prebuilt theme name, or "custom" for a custom theme: (Use arrow keys)
> Indigo/Pink [ Preview: https://material.angular.io/theme=indigo-pink ]
  Deep Purple/Amber [ Preview: https://material.angular.io/theme=deeppurple-amber ]
  Pink/Blue Grey [ Preview: https://material.angular.io/theme=pink-bluegrey ]
  Purple/Green [ Preview: https://material.angular.io/theme=purple-green ]
  custom [ Preview: https://material.angular.io/theme=custom ]
  ? (Move up and down to reveal more choices)
```

Seleccionar moviendo las flechas del teclado.

Luego pregunta:

Set up global Angular Material typography styles? (y/N)

¿Configurar estilos de tipografía de material angular global? (y/N)

La sugerencia es si (Yes)

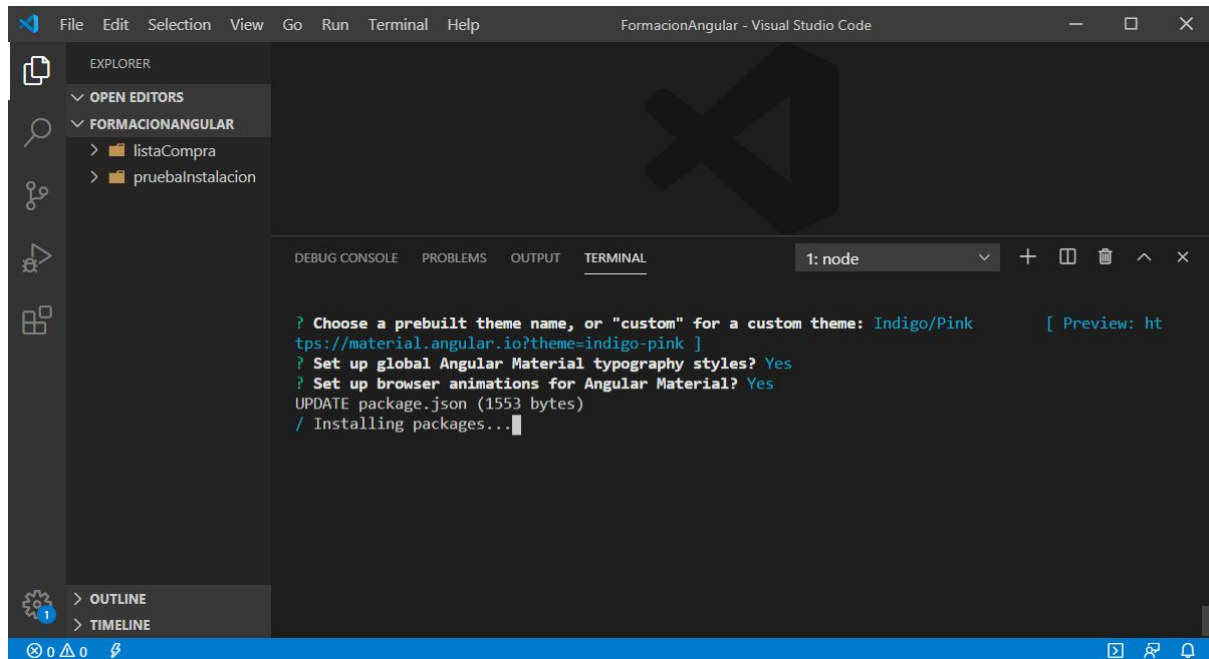
Set up browser animations for Angular Material? (Y/n)

Angular y Firebase

Proyecto Angular

¿Configurar animaciones de navegador para Material Angular? (Y/n)

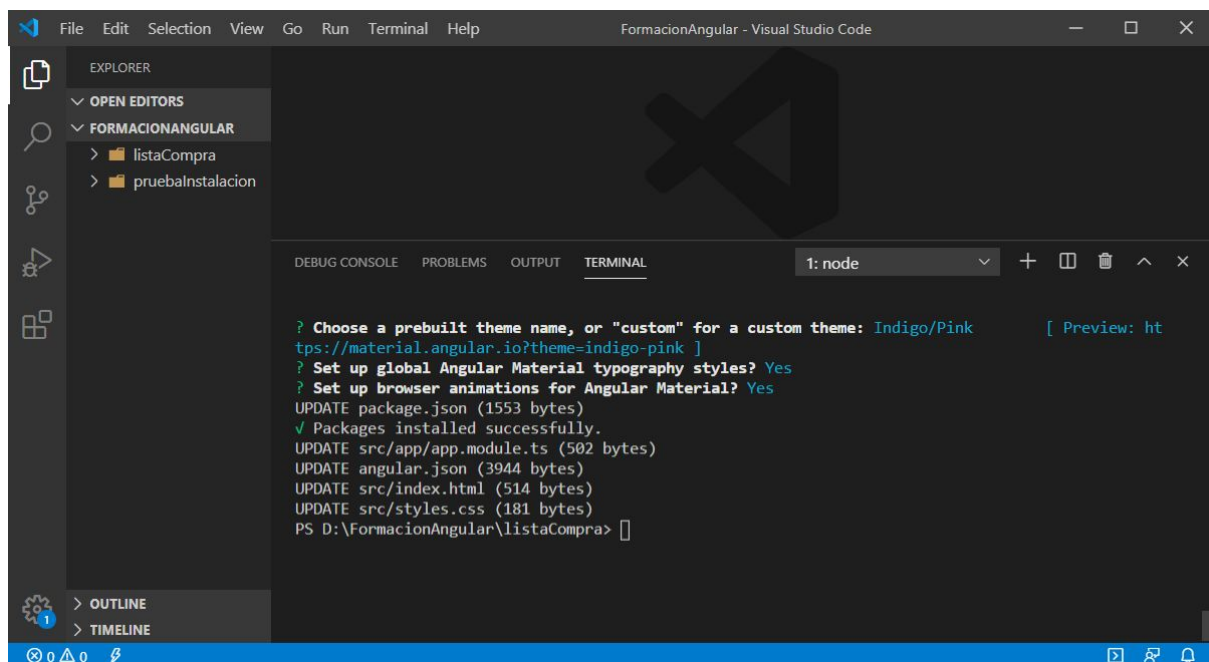
La sugerencia es si (Yes)



The screenshot shows the Visual Studio Code interface with the 'Terminal' panel active. The terminal is running a Node.js command to install Angular Material. The prompt asks for a theme name, and 'Indigo/Pink' is entered. It then asks for global typography styles and browser animations, both of which are confirmed with 'Yes'. The terminal shows the package.json being updated and packages being installed.

```
? Choose a prebuilt theme name, or "custom" for a custom theme: Indigo/Pink [ Preview: ht
tps://material.angular.io?theme=indigo-pink ]
? Set up global Angular Material typography styles? Yes
? Set up browser animations for Angular Material? Yes
UPDATE package.json (1553 bytes)
/ Installing packages...
```

Con estas respuestas, se ejecuta la instalación de los componentes gráficos de Angular Material en el proyecto



The screenshot shows the Visual Studio Code interface with the 'Terminal' panel active. The terminal shows the completion of the Angular Material installation. It confirms that the packages were installed successfully and lists the files that were updated: src/app/app.module.ts, angular.json, src/index.html, and src/styles.css. The prompt then shows the current directory as D:\FormacionAngular\listaCompra.

```
? Choose a prebuilt theme name, or "custom" for a custom theme: Indigo/Pink [ Preview: ht
tps://material.angular.io?theme=indigo-pink ]
? Set up global Angular Material typography styles? Yes
? Set up browser animations for Angular Material? Yes
UPDATE package.json (1553 bytes)
✓ Packages installed successfully.
UPDATE src/app/app.module.ts (502 bytes)
UPDATE angular.json (3944 bytes)
UPDATE src/index.html (514 bytes)
UPDATE src/styles.css (181 bytes)
PS D:\FormacionAngular\listaCompra>
```

Cuando la instalación finaliza, se han modificado 4 archivos del proyecto, relacionado con la estética y la configuración:

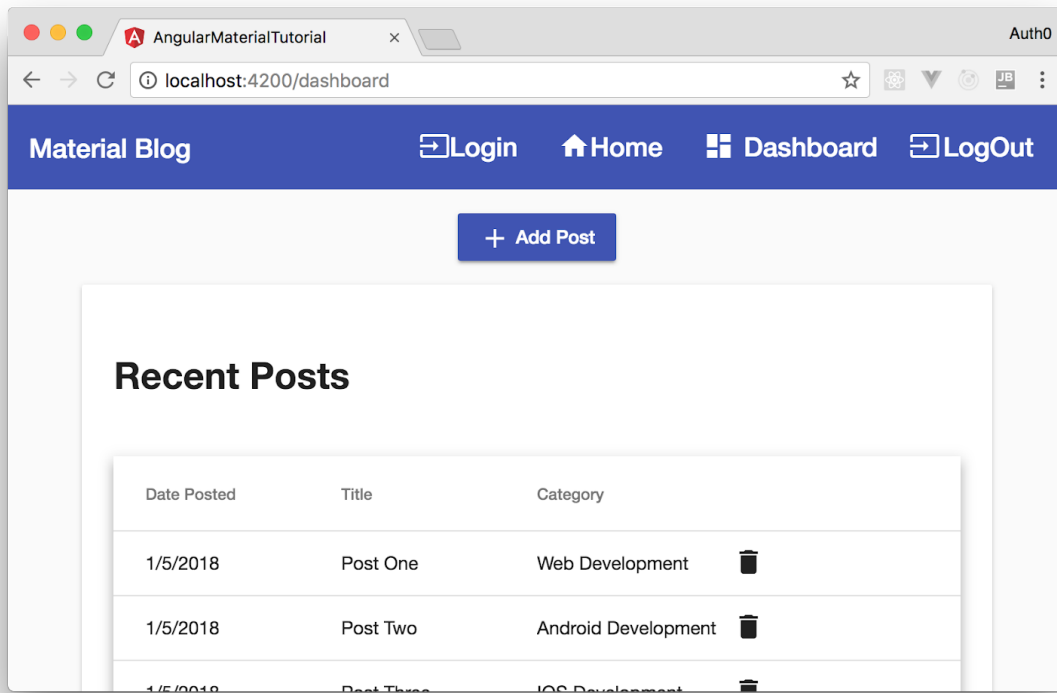
- src/app/app.module.ts
- angular.json

Angular y Firebase

Proyecto Angular

- src/index.html
- src/styles.css

A partir de este momento, de dispone de un conjunto de objetos gráficos que pueden ser usados para la estética del proyecto.



Para disponer de este módulo en el proyecto Angular, debe modificar el archivo **module.ts**. En esta guía, el archivo module.ts es ...\\listaCompra\\src\\app\\app.module.ts

...\\listaCompra\\src\\app\\app.module.ts

```
import { BrowserModule } from '@angular/platform-browser';
import { NgModule } from '@angular/core';

import { AppRoutingModule } from './app-routing.module';
import { AppComponent } from './app.component';
import { BrowserModuleAnimationsModule } from
'@angular/platform-browser/animations';

//---[Forms]-----
import { ReactiveFormsModule } from '@angular/forms';
import { from } from 'rxjs';
```

```
//---[Angular Material]-----
import {MatCheckboxModule} from '@angular/material/checkbox';
import {MatButtonModule} from '@angular/material/button';
import {MatInputModule} from '@angular/material/input';
import {MatAutocompleteModule} from '@angular/material/autocomplete';
import {MatDatepickerModule} from '@angular/material/datepicker';
import {MatFormFieldModule} from '@angular/material/form-field';
import {MatRadioModule} from '@angular/material/radio';
import {MatSelectModule} from '@angular/material/select';
import {MatSliderModule} from '@angular/material/slider';
import {MatSlideToggleModule} from '@angular/material/slide-toggle';
import {MatMenuModule} from '@angular/material/menu';
import {MatSidenavModule} from '@angular/material/sidenav';
import {MatToolbarModule} from '@angular/material/toolbar';
import {MatListModule} from '@angular/material/list';
import {MatGridListModule} from '@angular/material/grid-list';
import {MatCardModule} from '@angular/material/card';
import {MatStepperModule} from '@angular/material/stepper';
import {MatTabsModule} from '@angular/material/tabs';
import {MatExpansionModule} from '@angular/material/expansion';
import {MatButtonToggleModule} from
 '@angular/material/button-toggle';
import {MatChipsModule} from '@angular/material/chips';
import {MatIconModule} from '@angular/material/icon';
import {MatProgressSpinnerModule} from
 '@angular/material/progress-spinner';
import {MatProgressBarModule} from '@angular/material/progress-bar';
import {MatDialogModule} from '@angular/material/dialog';
import {MatTooltipModule} from '@angular/material/tooltip';
import {MatSnackBarModule} from '@angular/material/snack-bar';
import {MatTableModule} from '@angular/material/table';
import {MatSortModule} from '@angular/material/sort';
import {MatPaginatorModule} from '@angular/material/paginator';
import {MatNativeDateModule} from '@angular/material/core';

//---[Firebase]-----
```

```
import { AngularFireModule } from '@angular/fire'
import { AngularFirestoreModule } from '@angular/fire/firestore';

import { environment } from '../environments/environment';

//---[Componetes]-----
import { MenuComponent } from '../escritorio/menu/menu.component';
import { PiePaginaComponent } from
'../escritorio/pie-pagina/pie-pagina.component';
import { ListadoComponent } from
'../ejemplo/listado/listado.component';
import { FormularioComponent } from
'../ejemplo/formulario/formulario.component';
import { ResumenComponent } from
'../ejemplo/resumen/resumen.component';
import { FormProductoComponent } from
'../componentes/producto/form-producto/form-producto.component';

@NgModule({
  declarations: [
    AppComponent,
    MenuComponent,
    PiePaginaComponent,
    ListadoComponent,
    FormularioComponent,
    ResumenComponent,
    FormProductoComponent
  ],
  imports: [
    BrowserModule,
    AppRoutingModule,
    BrowserModuleAnimationsModule,
    //---[Forms]-----
    ReactiveFormsModule,
    //---[Angular Material]-----
    MatCheckboxModule,
    MatButtonModule,
    MatInputModule,
```

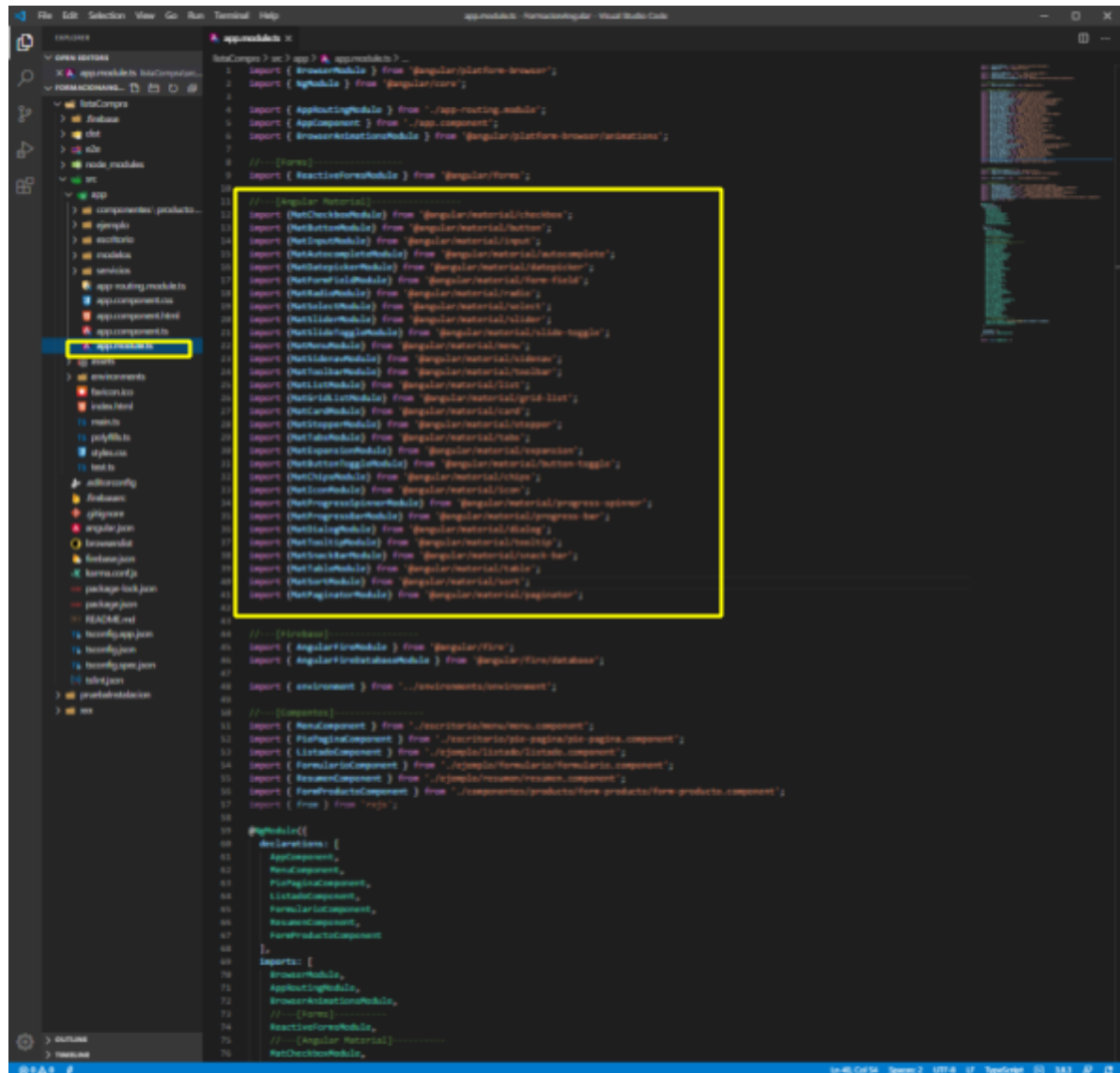
```
    MatAutocompleteModule,
    MatDatepickerModule,
    MatFormFieldModule,
    MatRadioModule,
    MatSelectModule,
    MatSliderModule,
    MatSlideToggleModule,
    MatMenuModule,
    MatSidenavModule,
    MatToolbarModule,
    MatListModule,
    MatGridListModule,
    MatCardModule,
    MatStepperModule,
    MatTabsModule,
    MatExpansionModule,
    MatButtonModuleToggleModule,
    MatChipsModule,
    MatIconModule,
    MatProgressSpinnerModule,
    MatProgressBarModule,
    MatDialogModule,
    MatTooltipModule,
    MatSnackBarModule,
    MatTableModule,
    MatSortModule,
    MatPaginatorModule,
    MatNativeDateModule,
    //---[Firebase]-----
    AngularFireModule.initializeApp(environment.firebaseConfig),
    AngularFirestoreModule
  ],
  providers: [
    MatDatepickerModule,
    MatNativeDateModule
  ],
  bootstrap: [AppComponent]
})
```


Angular y Firebase

Proyecto Angular

```
export class AppModule { }
```

Con la adición de las líneas que hacen relación a Angular Material, se dispone de las funciones de Material para el proyecto Angular.



Proyecto Angular

Paso 4: Creando componentes con los datos

En esta guía orientar la creación de este componente para ingresar datos a la base de datos en Angular.

The screenshot shows a web browser window with the title 'Lista de Compra'. The address bar displays 'localhost:4200/formularioproducto'. The page has a blue header with the text 'Lista de Compra' and a navigation bar with links: 'Agregar Producto', 'Listado', 'Formulario', and 'Resumen'. The main content area is a form titled 'Crear un producto nuevo'. The form contains the following fields and controls:

- Nombre del producto:** A text input field with a placeholder 'No más de 60 caracteres de largo.'
- Categoria:** A dropdown menu.
- Fecha de creación:** A date input field showing '5/6/2020' with a calendar icon and a placeholder 'Fecha DD/MM/AAAA de creación'.
- Cantidad mínima requerida:** A text input field with a placeholder 'Número acorde a las unidades.'
- Cantidad registrada requerida:** A text input field with a placeholder 'Número acorde a las unidades.'
- Buttons:** 'Adicionar registro' (grey) and 'Cancelar' (red).
- Footer:** 'Ejemplo de un formulario con Angular Material' and 'pie-pagina works!'

Un componente se compone de tres archivos:

- HTML (documentos con etiquetas, este documento lo ve el usuario)
- CSS (códigos de estilo)
- TS (códigos en el lenguaje TypeScript que realiza operaciones que el usuario no ve)

...\listaCompra\src\app\componentes\producto\form-producto\
form-producto.component.html

```
<section fxLayout="row wrap" fxLayoutAlign="center center">
  <mat-card fxFlex="500px" fxFlex.xs="100%">
    <mat-card-title>Crear un producto nuevo</mat-card-title>

    <form [formGroup]="formularioProducto"
      autocomplete="off"
      novalidate
      (ngSubmit)="accionEnviar(formularioProducto.value)"
```

```
        fxLayout="column wrap"
        fxLayoutAlign="center center"
        fxLayoutGap="10px">
    <mat-card-content>
        <!--[Campo texto con control de
longitud]----->
        <mat-form-field>
            <input matInput type="text"
                placeholder="Nombre del producto"
                formControlName="nombre" id="nombre">
            <mat-hint align="end">No más de 60 caracteres de
largo.</mat-hint>
            <mat-error *ngIf="hasError('nombre',
'required')">Nombre es requerido</mat-error>
            <mat-error *ngIf="hasError('nombre',
'maxlength')">Tienes más de 60 caracteres</mat-error>
        </mat-form-field>

        <!--[Lista de selección]----->
        <mat-form-field appearance="fill">
            <mat-label>Categoria</mat-label>
            <mat-select placeholder="Categoria del producto"
                formControlName="categoria">
                <mat-option
value="Proteina">Proteina</mat-option>
                <mat-option value="Harina">Harina</mat-option>
                <mat-option value="Granos">Granos</mat-option>
                <mat-option value="Frutas">Frutas</mat-option>
                <mat-option
value="Vegetales">Vegetales</mat-option>
                <mat-option
value="Lacteos">Lacteos</mat-option>
                <mat-option value="Licor">Licor</mat-option>
                <mat-option value="Dulces">Dulces</mat-option>
                <mat-option
value="Delicateses">Delicateses</mat-option>
                <mat-option value="Aseo">Aseo</mat-option>
            </mat-select>
```

```
</mat-form-field>

<!--[Campo de fecha]----->
<mat-form-field>
  <mat-label>Fecha de creación</mat-label>
  <input matInput [matDatepicker]="picker"
    placeholder="Fecha de creación"
    formControlName="fechaCreacion">
  <mat-hint align="end">Fecha
DD/MM/AAAA de creación</mat-hint>
  <mat-error
*ngIf="hasError('cantidadMinima', 'required')">Fecha de Creación es
requerida</mat-error>
    <mat-datepicker-toggle matSuffix
[for]="picker"></mat-datepicker-toggle>
    <mat-datepicker #picker></mat-datepicker>
</mat-form-field>

<!--[Campo numerico]----->
<mat-form-field>
  <input matInput type="number"
    placeholder="Cantidad mínima requerida"
    formControlName="cantidadMinima"
id="cantidadMinima">
  <mat-hint align="end">Número acorde a las
unidades.</mat-hint>
  <mat-error *ngIf="hasError('cantidadMinima',
'required')">Cantidad mínima es requerida</mat-error>
</mat-form-field>

<!--[Campo numerico]----->
<mat-form-field>
  <input matInput type="number"
    placeholder="Cantidad registrada
requerida"
    formControlName="cantidad" id="cantidad">
  <mat-hint align="end">Número acorde a las
unidades.</mat-hint>
```

```
        <mat-error *ngIf="hasError('cantidad',
'required')">Cantidad mínima es requerida</mat-error>
    </mat-form-field>

</mat-card-content>

<mat-card-actions align="end">
    <button mat-raised-button color="primary"
        [disabled]="!formularioProducto.valid">
        Adicionar registro
    </button>
    <button mat-raised-button color="warn"
        type="button"
        (click)="onCancel()">
        Cancelar
    </button>
</mat-card-actions>
</form>

</mat-card>
</section>

Ejemplo de un formulario con Angular Material
```

...\listaCompra\src\app\componentes\producto\form-producto\
form-producto.component.**CSS**

```
mat-form-field{
    width: 100%;
}
mat-card-title{
    text-align: center;
}
```

...\listaCompra\src\app\componentes\producto\form-producto\
form-producto.component.ts

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

//---[Formulario]-----
import { FormControl, FormGroup, Validators } from '@angular/forms';
import { Location } from '@angular/common';

//---[Servicio de datos]-----
import { ProductoService } from 'src/app/servicios/producto.service';
import { Producto } from 'src/app/modelos/producto.model';

@Component({
  selector: 'app-form-producto',
  templateUrl: './form-producto.component.html',
  styleUrls: ['./form-producto.component.css']
})

export class FormProductoComponent implements OnInit {
  public formularioProducto: FormGroup;
  public producto: Producto;

  constructor(private location: Location,
               private productoService: ProductoService) { }

  ngOnInit() {
    this.formularioProducto = new FormGroup({
      nombre: new FormControl('', [Validators.required,
Validators.maxLength(60)]),
      categoria: new FormControl(''),
      fechaCreacion: new FormControl(new Date()),
      cantidadMinima: new FormControl('', [Validators.required]),
      cantidad: new FormControl('', [Validators.required]),
    });
  }

  public hasError = (controlName: string, errorName: string) =>{
```

```
        return
    this.formularioProducto.controls[controlName].hasError(errorName);
    }

    public onCancel = () => {
        this.location.back();
    }

    public accionEnviar = (formularioProductoValue) => {
        if (this.formularioProducto.valid) {
            this.producto = new Producto();
            this.producto.id = "edi";
            this.producto.nombre = formularioProductoValue.nombre;
            this.producto.categoria = formularioProductoValue.categoria;
            this.producto.cantidadMinima =
formularioProductoValue.cantidadMinima;
            this.producto.cantidad = formularioProductoValue.cantidad;
            this.producto.fechaCreacion =
formularioProductoValue.fechaCreacion;

            this.create(this.producto);
        }
    }

    private create(producto: Producto) {
        this.productoService.createProducto(producto);
        this.location.back();
    }
}
```

El resultado es:

Angular y Firebase

Proyecto Angular

angular - Database - Firebase cc x ListaCompra x +

localhost:4200/formularioproducto

Lista de Compra

Agregar Producto | Listado | Formulario | Resumen

Crear un producto nuevo

Nombre del producto
Pollo

Categoría
Proteína

Fecha de creación
5/6/2020

Cantidad mínima requerida
400

Cantidad registrada requerida
100

Adicionar registro Cancelar

Ejemplo de un formulario con Angular Material

pie-pagina works!

angular - Database - Firebase cc x ListaCompra x +

https://console.firebase.google.com/project/angular-guia/dat...

Database

Cloud Firestore

Datos Reglas Índices Uso

angular-guia productos Lj4fxMaibhUTP080RUpc

+ Iniciar colección + Agregar documento + Iniciar colección

productos > Lj4fxMaibhUTP080R + Agregar campo

cantidad: 100
cantidadMinima: 400
categoria: "Proteína"
fechaCreacion: 28 de noviembre de 2019 a las 00:00:00 UTC-5
id: "edi"
nombre: "Pollo"

Angular y Firebase

Proyecto Angular

La librería de AngularFire está en modificación, lo que puede generar un error. Una de las solución es modificar la función **createProducto** el archivo de servicio (solo si les presenta error) de esta forma:

...\listaCompra\src\app

```
. . .
//---[create]-----
createProducto(producto: Producto) {

    /**
    this.firestore.collection('productos').add({
        id: producto.id,
        nombre: producto.nombre,
        categoria: producto.categoria,
        fechaCreacion: producto.fechaCreacion,
        cantidadMinima: producto.cantidadMinima,
        cantidad: producto.cantidad
    }).catch(function(error) {
        console.error('Error escribiendo el mensaje en la base de
datos', error);
    });
    return this.firestore.collection('productos').add(producto);
    /** */
}
. . .
```

Continuamos con la creación de los otros componentes que completan el CRUD