

Tecnológico de Estudios Superiores de Ecatepec

**División de Ingeniería en Sistemas
Computacionales**

Academia de Ingeniería Aplicada



**Asignatura: Especialidad de Base de Datos para
Dispositivos Móviles**

EQUIPO: TECHNOLOGY

INTEGRANTES:

CASTRO SANDOVAL DANIELA DONAJI

HERNÁNDEZ SAHAGÚN JORGE FABIAN

HERNÁNDEZ VERGARA DANIEL

MUÑOZ BÁEZ WENDY GUADALUPE

RAMOS HERNÁNDEZ ANTONIO

**MODULO ASIGNADO GENERAR GRÁFICAS DE BARRA DESDE UN
ARCHIVO JSON**

Grupo: 5851

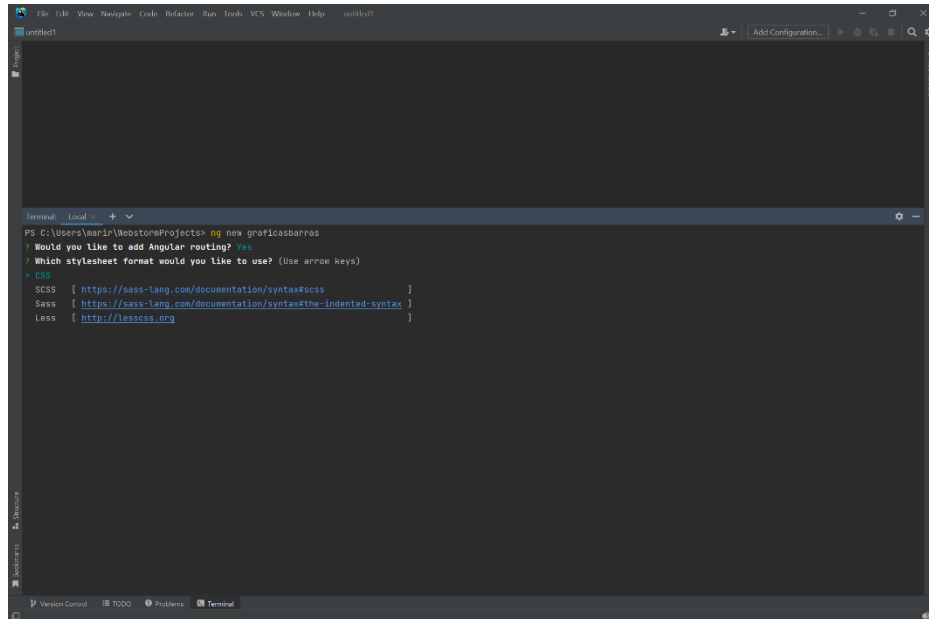
Fecha de entrega: 11 de mayo 2022

Profesora: Griselda Cortes Barrera

Generar Grafica de barras

1. Creación del proyecto:

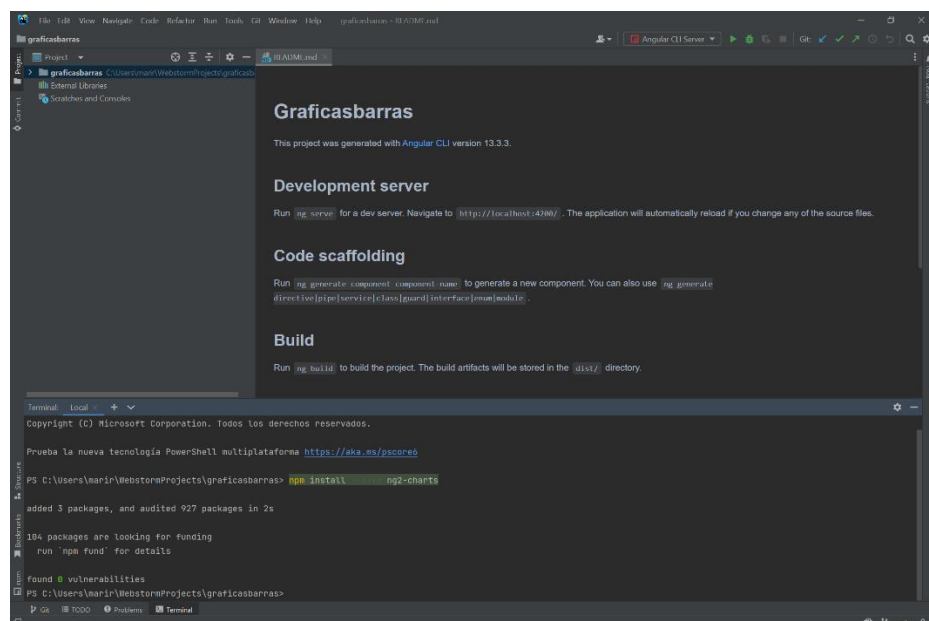
Primero, en la terminal, debemos de crear el proyecto con el comando ng new graficasbarras y dar en yes posteriormente escogemos css.



```
PS C:\Users\marin\Webstorm\Projects> ng new graficasbarras
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 ]
```

2. Instalar ng2-charts:

Para este paso se requiere de internet para instalar ng2-charts con el comando npm install --save ng2-charts



```
graficasbarras
This project was generated with Angular CLI version 13.3.3.

Development server
Run 'ng serve' for a dev server. Navigate to 'http://localhost:4200/'. The application will automatically reload if you change any of the source files.

Code scaffolding
Run 'ng generate component component-name' to generate a new component. You can also use 'ng generate directive|pipe|service|class|guard|interface|enum|module'.

Build
Run 'ng build' to build the project. The build artifacts will be stored in the 'dist/' directory.

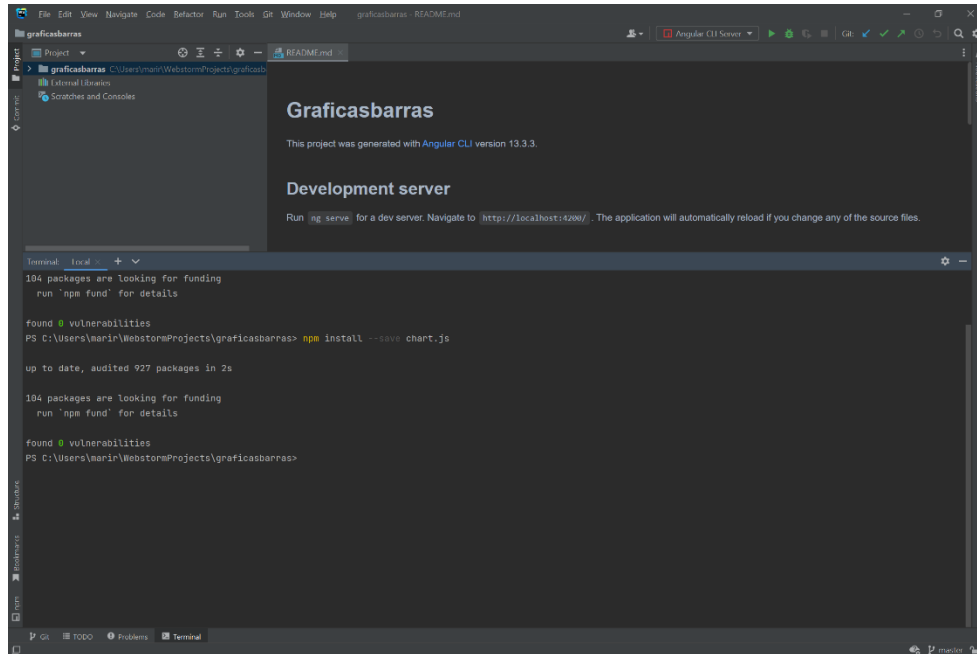
Terminal
Copyright (C) Microsoft Corporation. Todos los derechos reservados.

Prueba la nueva tecnología PowerShell multiplataforma https://aka.ms/powershell

PS C:\Users\marin\Webstorm\Projects\graficasbarras> npm install --save ng2-charts
added 3 packages, and audited 927 packages in 2s
104 packages are looking for funding
run 'npm fund' for details
found 0 vulnerabilities
PS C:\Users\marin\Webstorm\Projects\graficasbarras>
```

3. Instalar las librerías de ng2-charts:

Para este paso se requiere de internet para instalar las librerías de ng2-charts con el comando `npm install --save chart.js`



The screenshot shows the Visual Studio Code interface with the 'graficasbarras' project open. The terminal at the bottom displays the command `npm install --save chart.js` and its output, which includes a security audit of 927 packages. The editor shows the `README.md` file with instructions for running the development server.

```
graficasbarras - README.md
This project was generated with Angular CLI version 13.3.3.

Development server
Run ng serve for a dev server. Navigate to http://localhost:4200/. The application will automatically reload if you change any of the source files.

Terminal: Local
104 packages are looking for funding
run 'npm fund' for details

found 0 vulnerabilities
PS C:\Users\marin\WebstormProjects\graficasbarras> npm install --save chart.js

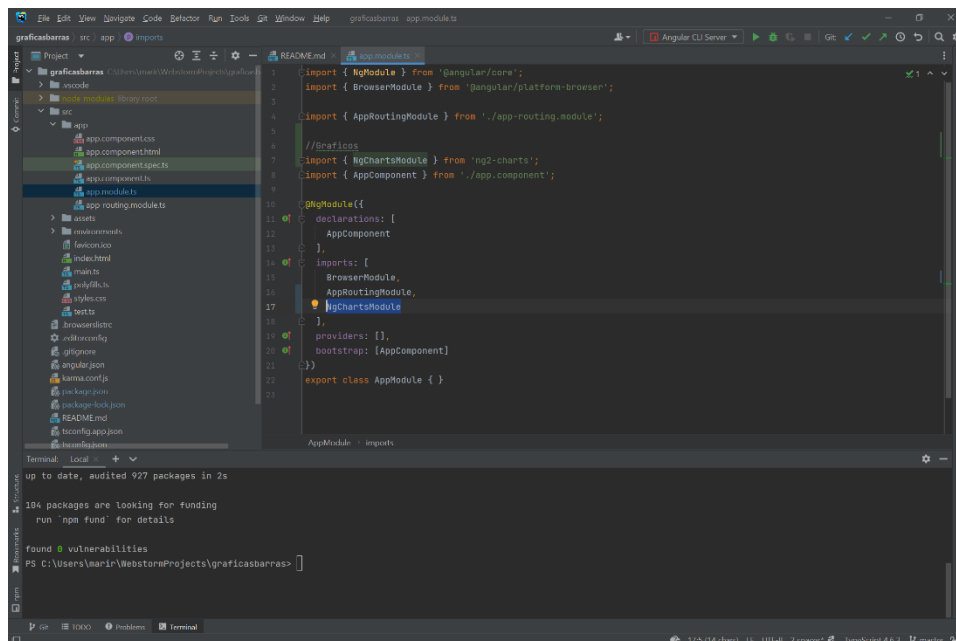
up to date, audited 927 packages in 2s

104 packages are looking for funding
run 'npm fund' for details

found 0 vulnerabilities
PS C:\Users\marin\WebstormProjects\graficasbarras>
```

4. Importación del módulo NgChartModule:

Nos dirigimos al `app.module` y en imports colocamos `NgChartModule` y se importara la ruta de `import { NgChartModule } from 'ng2-charts';`



The screenshot shows the Visual Studio Code interface with the 'graficasbarras' project open. The `app.module.ts` file is open in the editor, showing the import of `NgChartModule` from 'ng2-charts'. The terminal at the bottom displays the command `npm install --save chart.js` and its output, which includes a security audit of 927 packages.

```
graficasbarras - app.module.ts
import { NgModule } from '@angular/core';
import { BrowserModule } from '@angular/platform-browser';
import { AppRoutingModule } from './app-routing.module';
import { AppComponent } from './app.component';
import { NgChartModule } from 'ng2-charts';

@NgModule({
  declarations: [
    AppComponent
  ],
  imports: [
    BrowserModule,
    AppRoutingModule,
    NgChartModule
  ],
  providers: [],
  bootstrap: [AppComponent]
})
export class AppModule {}

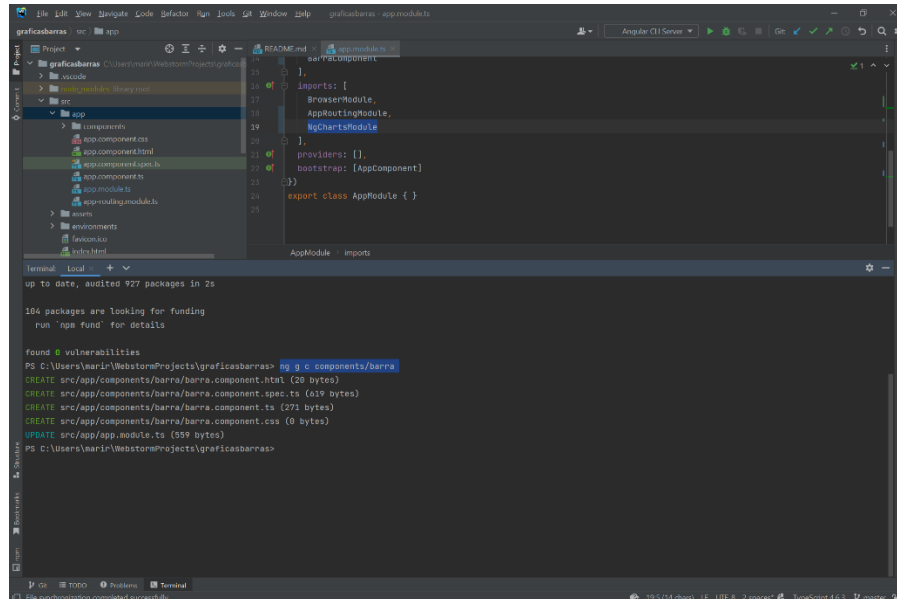
Terminal: Local
up to date, audited 927 packages in 2s

104 packages are looking for funding
run 'npm fund' for details

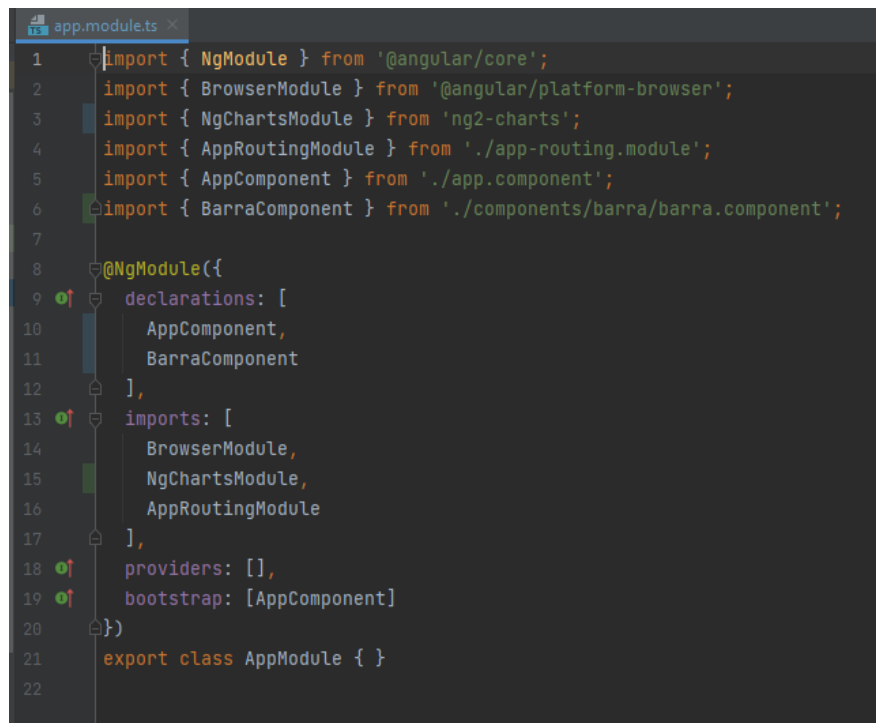
found 0 vulnerabilities
PS C:\Users\marin\WebstormProjects\graficasbarras>
```

5. Creación del componente barra:

En la terminal creamos el componente barra dentro de una carpeta nueva que se creará en automático con ng g c component/barra

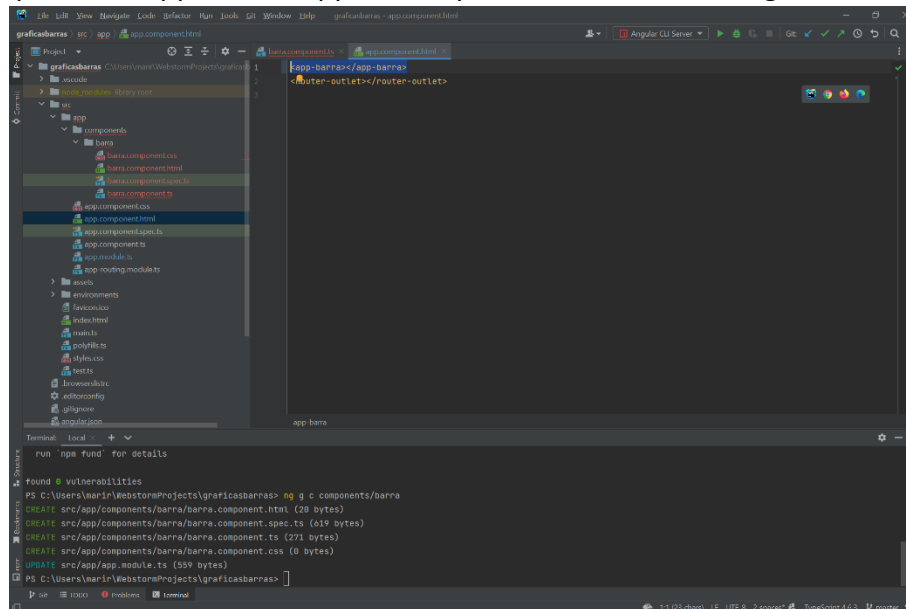


Nota: Procurar tener todos los componentes declarados en su modulo a trabajar e importar todos los módulos a ocupar.



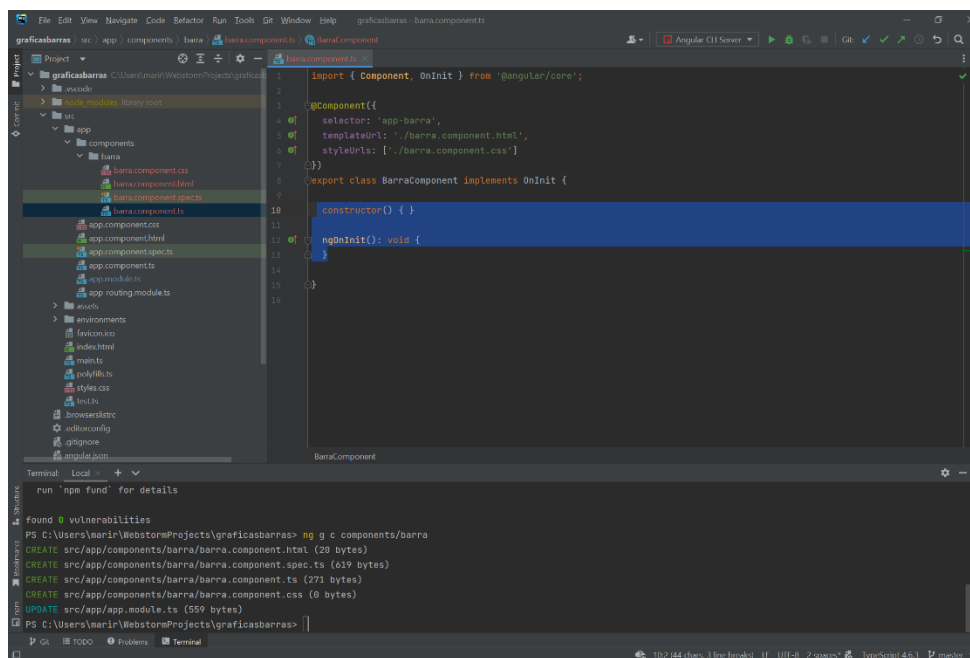
6. Llamar la etiqueta de referencia del componente barra:

Si instalamos todo correctamente en app.component.html introducimos la etiqueta de `<app-barr></app-barr>` y no debería marcar ningún error.



7. Introducción de código TypeScript para las gráficas de barras:

Eliminamos el código: `constructor(){} ngOnInit(): void {}`



En donde eliminamos el constructor e introducimos el siguiente código:

```

@ViewChild(BaseChartDirective) chart: BaseChartDirective | undefined;

public barChartOptions: ChartConfiguration['options'] = {
  responsive: true,
  // We use these empty structures as placeholders for dynamic theming.
  scales: {
    x: {},
    y: {
      min: 10
    }
  },
  plugins: {
    legend: {
      display: true,
    },
    datalabels: {
      anchor: 'end',
      align: 'end'
    }
  }
};

public barChartType: ChartType = 'bar';
public barChartPlugins = [
  DataLabelsPlugin
];

public barChartData: ChartData<'bar'> = {
  labels: [ '2006', '2007', '2008', '2009', '2010', '2011', '2012' ],
  datasets: [
    { data: [ 65, 59, 80, 81, 56, 55, 40 ], label: 'Series A' },
    { data: [ 28, 48, 40, 19, 86, 27, 90 ], label: 'Series B' }
  ]
};

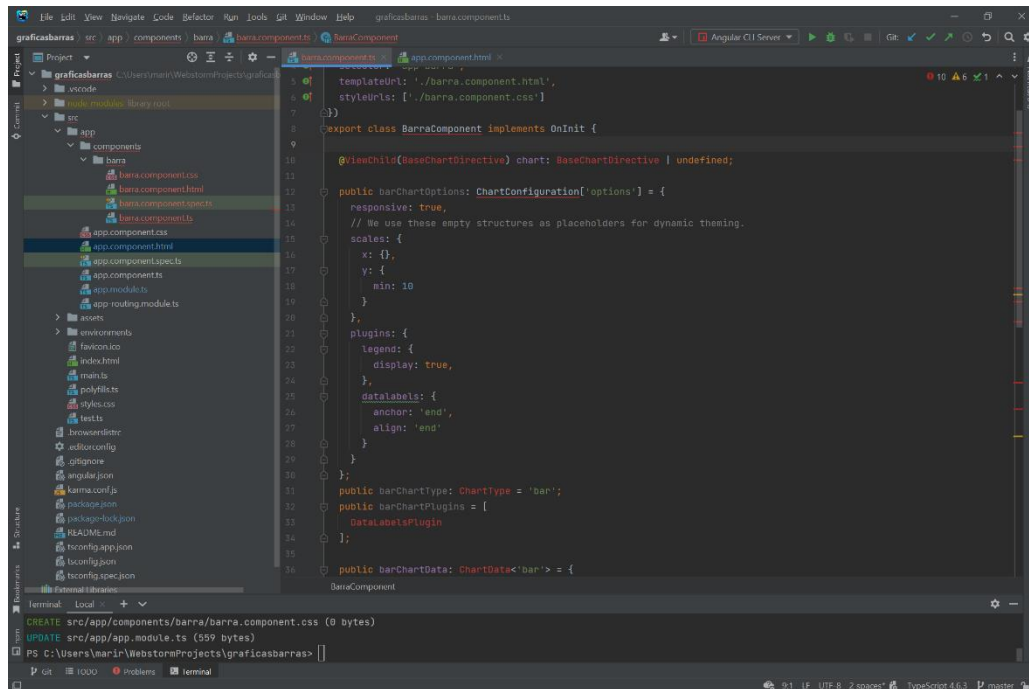
// events
public chartClicked({ event, active }: { event?: ChartEvent, active?: {}[] }): void {
  console.log(event, active);
}

public chartHovered({ event, active }: { event?: ChartEvent, active?: {}[] }): void {
  console.log(event, active);
}

public randomize(): void {
  // Only Change 3 values
  this.barChartData.datasets[0].data = [
    Math.round(Math.random() * 100),
    59,
    80,
    Math.round(Math.random() * 100),
    56,
    Math.round(Math.random() * 100),
    40 ];
}

```

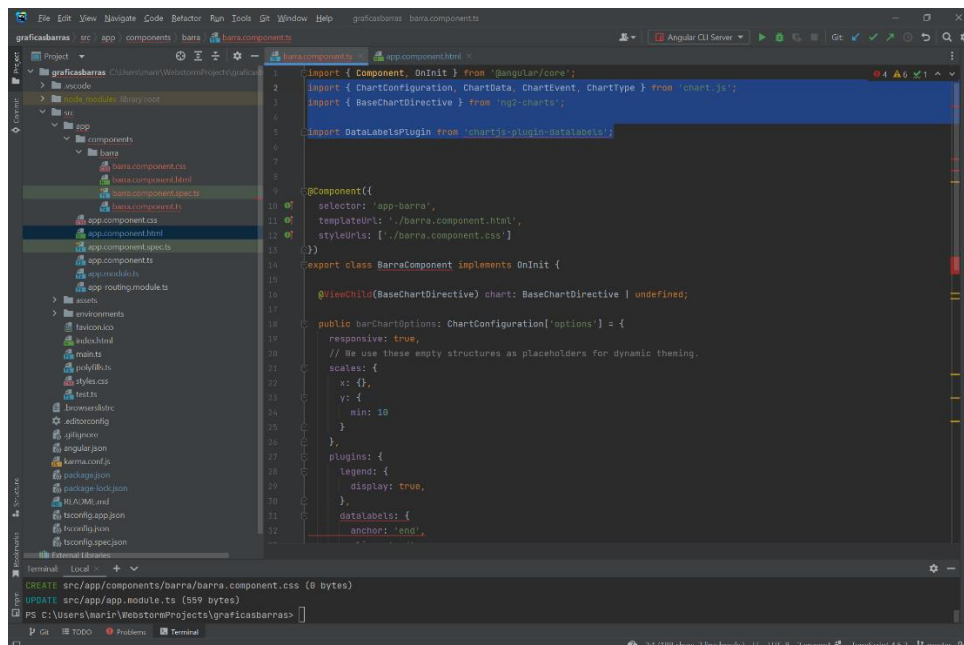
```
this.chart?.update();
}
```



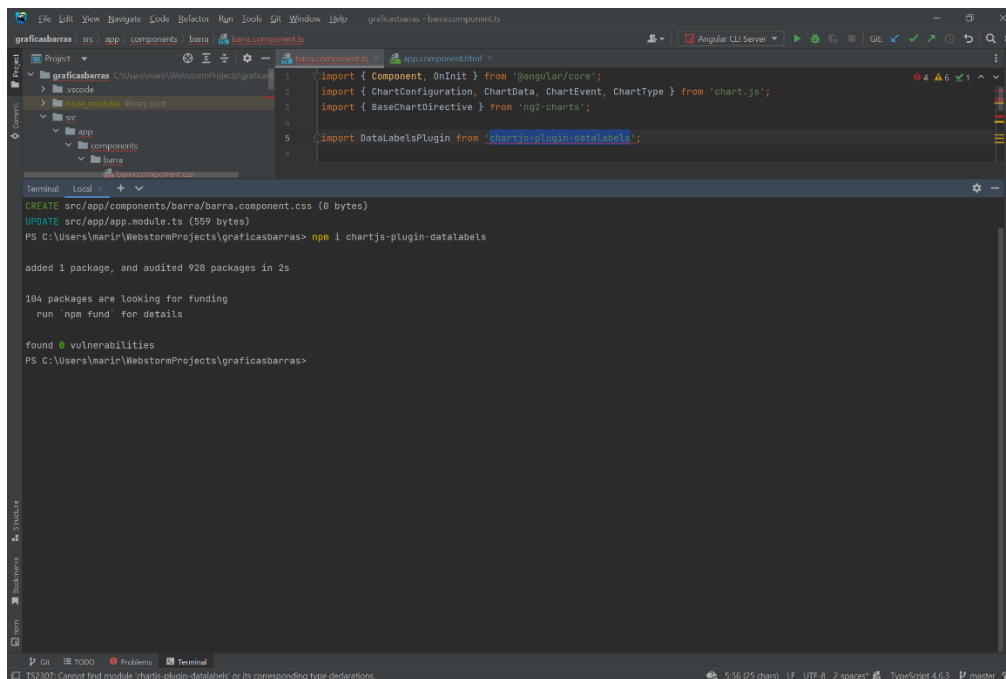
Colocamos el código de las importaciones:

```
import { ChartConfiguration, ChartData, ChartEvent, ChartType } from
'chart.js';
import { BaseChartDirective } from 'ng2-charts';

import DataLabelsPlugin from 'chartjs-plugin-datalabels';
```



Para eliminar el error que marca en 'chartjs-plugin-datalabels' es porque se debe de instalar el siguiente plugin en la terminal con el comando:
npm install chartjs-plugin-datalabels --save

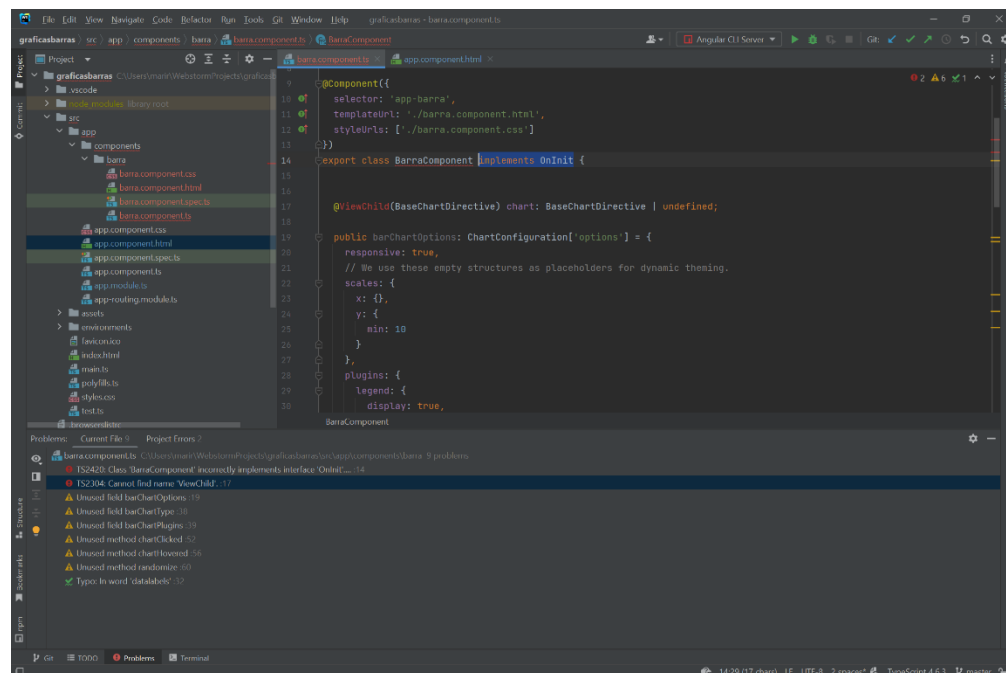


The screenshot shows a VS Code editor with a project named 'graficasbarras'. The file explorer on the left shows the project structure. The main editor displays the 'barra.component.ts' file with the following code:

```
1 import { Component, OnInit } from '@angular/core';
2 import { ChartConfiguration, ChartData, ChartEvent, ChartType } from 'chart.js';
3 import { BaseChartDirective } from 'ng2-charts';
4
5 import DataLabelsPlugin from 'chartjs-plugin-datalabels';
```

The terminal window at the bottom shows the command 'npm install chartjs-plugin-datalabels' being executed, with output indicating that 1 package was added and 104 packages are looking for funding.

Eliminamos el implements OnInit

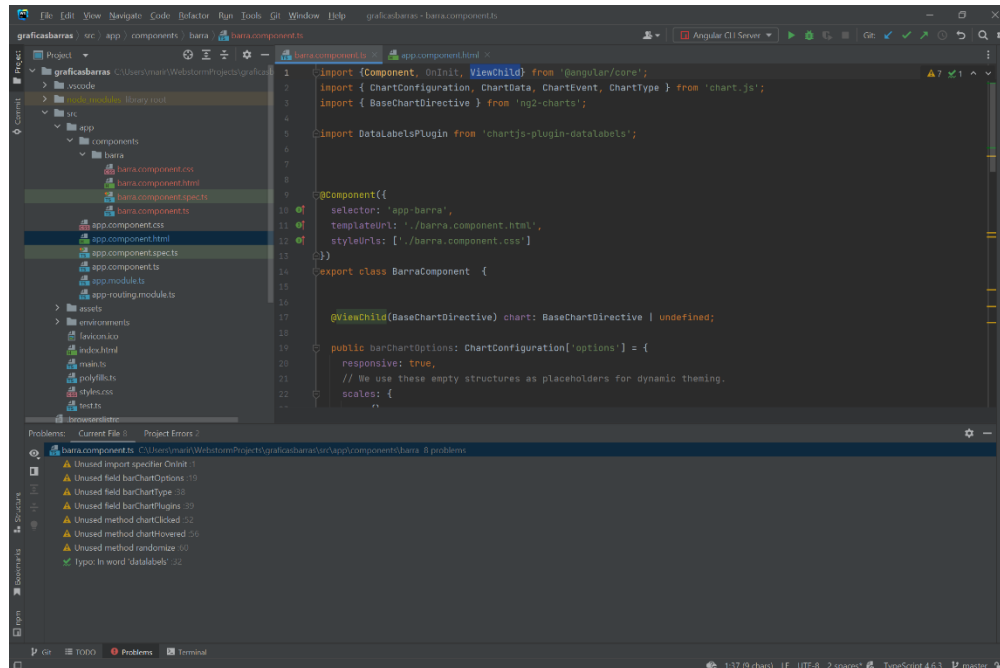


The screenshot shows the same VS Code editor with the 'barra.component.ts' file. The code is now:

```
9 @Component({
10   selector: 'app-barra',
11   templateUrl: './barra.component.html',
12   styleUrls: ['./barra.component.css']
13 })
14 export class BarraComponent {
15
16   @ViewChild(BaseChartDirective) chart: BaseChartDirective | undefined;
17
18   public barChartOptions: ChartConfiguration['options'] = {
19     responsive: true,
20     // We use these empty structures as placeholders for dynamic theming.
21     scales: {
22       x: {},
23       y: {
24         min: 10
25       }
26     },
27     plugins: {
28       legend: {
29         display: true,
30       }
31     }
32   };
33 }
```

The 'implements OnInit' has been removed from the class declaration. The Problems panel at the bottom shows several TypeScript errors, including 'TS2420: Class 'BarraComponent' incorrectly implements interface 'OnInit'' and 'TS2304: Cannot find name 'ViewChild''.

Colocamos el viewChild en import



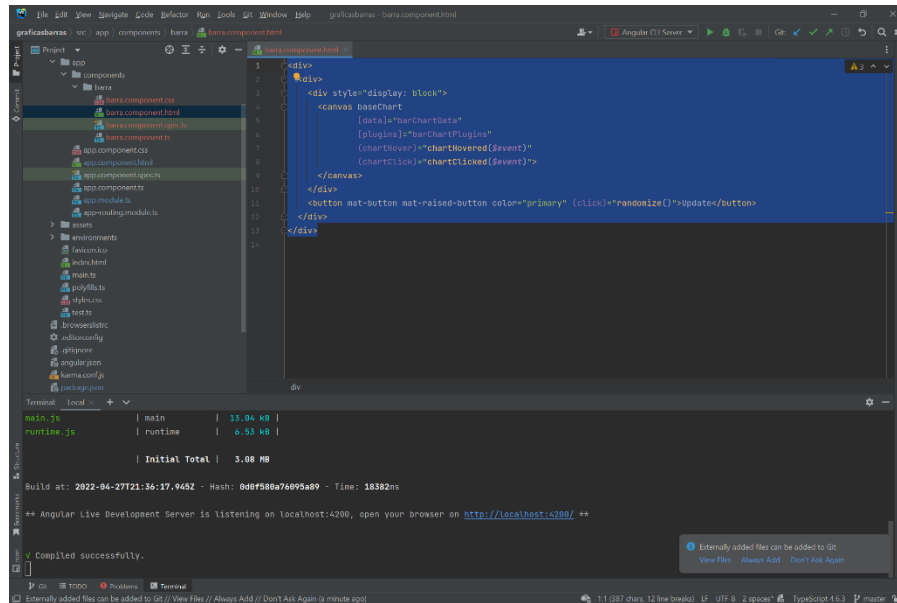
8. Código HTML para mostrar la grafica de barras:

En barra.component.html se coloca el siguiente código:

```

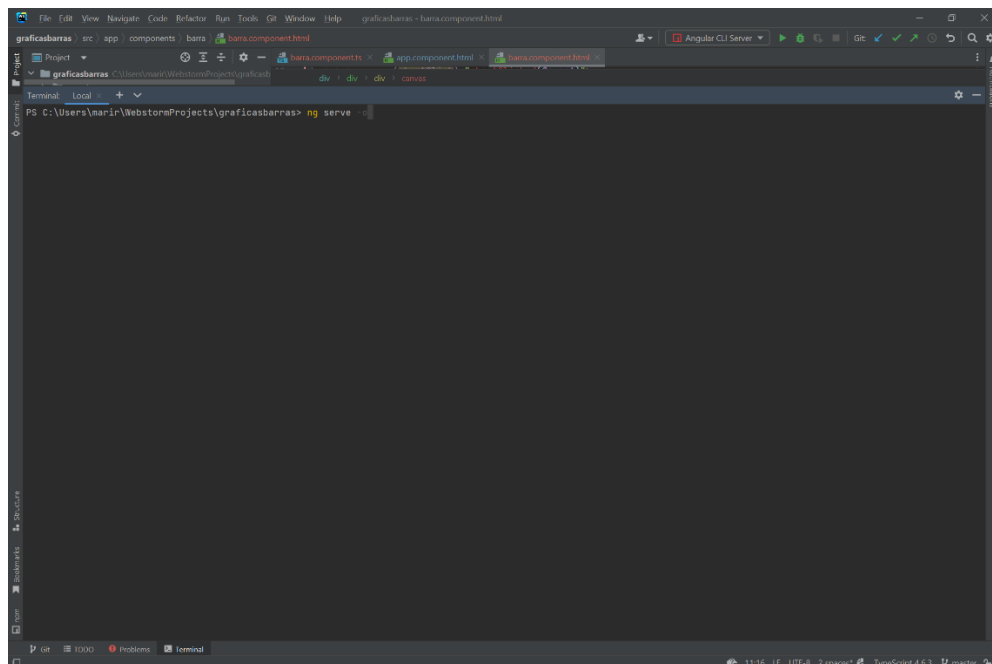
<div>
  <div>
    <div style="display: block">
      <canvas baseChart
        [data]="barChartData"
        [plugins]="barChartPlugins"
        (chartHover)="chartHovered($event)"
        (chartClick)="chartClicked($event)">
      </canvas>
    </div>
    <button mat-button mat-raised-button color="primary"
      (click)="randomize()">Update</button>
    </div>
  </div>

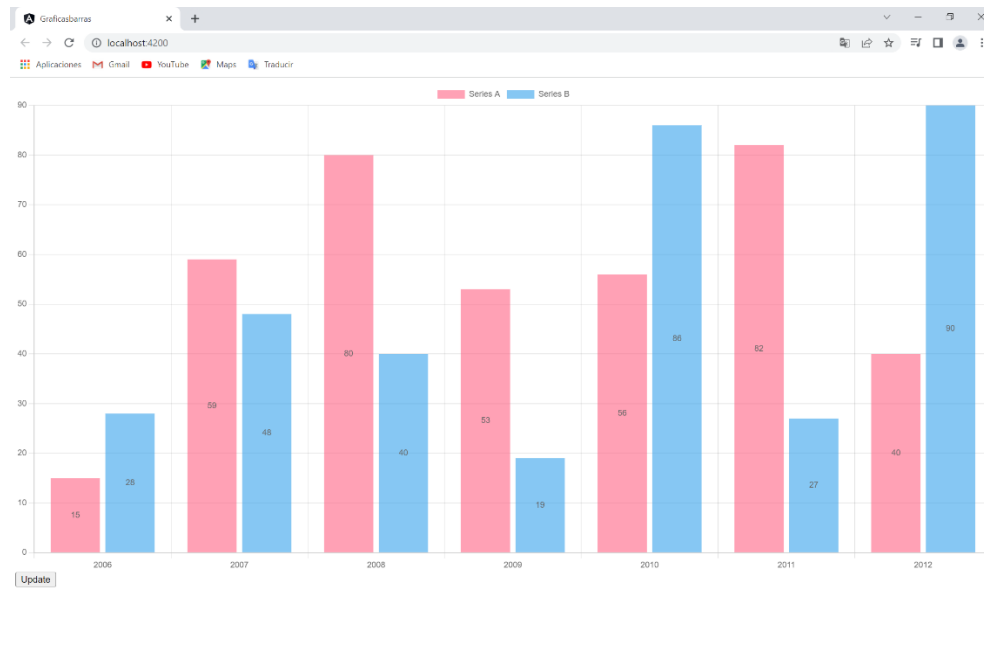
```



9. Ejecutar el servicio y visualizar la gráfica en el navegador:

Primero ejecutamos el comando de `ng serve -o` y posteriormente si se hizo todo correctamente se visualizará las graficas de barra en el navegador predeterminado.





Para consultar atributos y métodos de la librería se puede consultar la documentación en <https://www.chartjs.org/docs/latest/>

Generar Grafica de barras con datos de un archivo JSON

1. Archivo JSON:

Primero debemos de tener el archivo json de cual se va a recuperar los datos, y posteriormente meter ese archivo dentro de la carpeta assets. Aquí se usará como ejemplo un formato similar al que se usó en el código usado de la generación de graficas de barra sin json.

```
1  {
2    "anio": "2006",
3    "seriesA": 15,
4    "seriesB": 28
5  },
6  {
7    "anio": "2007",
8    "seriesA": 59,
9    "seriesB": 48
10 },
11 {
12   "anio": "2008",
13   "seriesA": 80,
14   "seriesB": 40
15 },
16 {
17   "anio": "2009",
18   "seriesA": 53,
19   "seriesB": 19
20 },
21 {
22   "anio": "2010",
23   "seriesA": 56,
24   "seriesB": 86
25 },
26 {
27   "anio": "2011",
28   "seriesA": 82,
29   "seriesB": 27
30 },
31 {
32   "anio": "2012",
33   "seriesA": 40,
34   "seriesB": 89
35 }
```

```
[
  {
    "anio":"2006",
    "seriesA":65,
    "seriesB": 28
  },
  {
    "anio":"2007",
    "seriesA":59,
    "seriesB":48
  },
  {"anio":"2008",
    "seriesA":80,
    "seriesB": 40
  }
,
  {"anio":"2009",
    "seriesA":81,
    "seriesB": 19
  }
,
  {"anio":"2010",
    "seriesA":56,
    "seriesB": 86
  }
,
  {"anio":"2011",
    "seriesA":55,
    "seriesB": 27
  }
,
  {"anio":"2012",
    "seriesA":40,
    "seriesB": 90
  },
  {"anio":"2013",
    "seriesA":44,
    "seriesB": 88
  },
  {"anio":"2014",
```

```

"seriesA":91,
"seriesB": 100
},
{"anio":"2015",
"seriesA":91,
"seriesB": 100
}
]

```

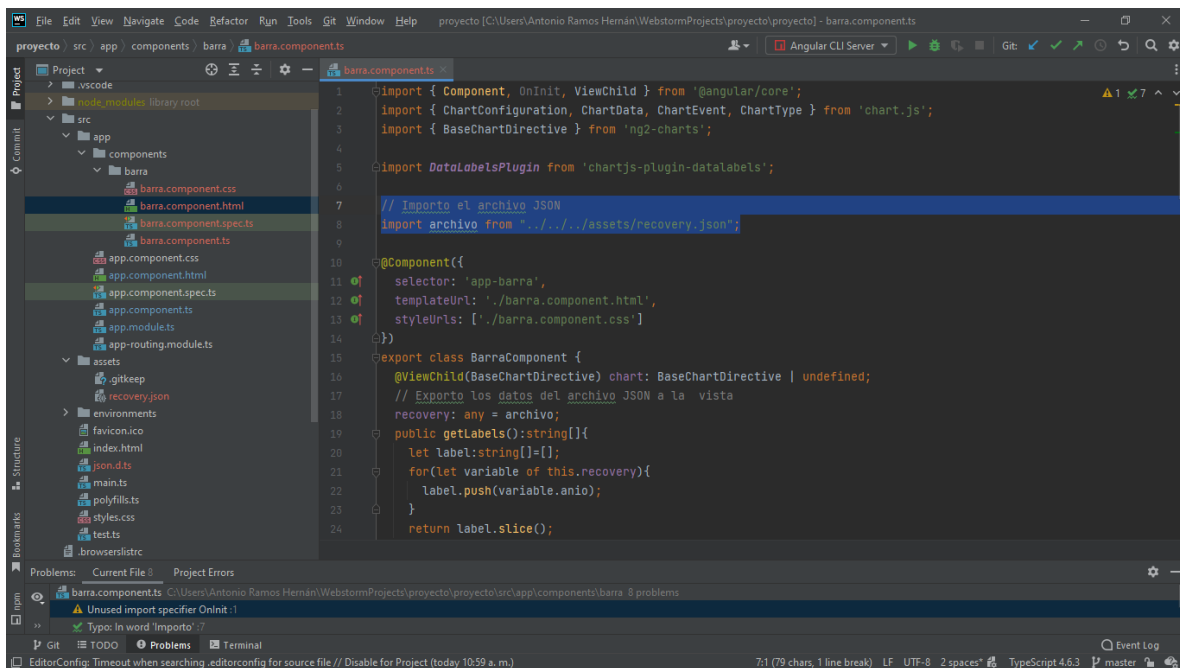
2. Importación del archivo a un componente JSON:

Después en el componente que hicimos de barras introducimos el código que va a importar los datos del archivo json.

```

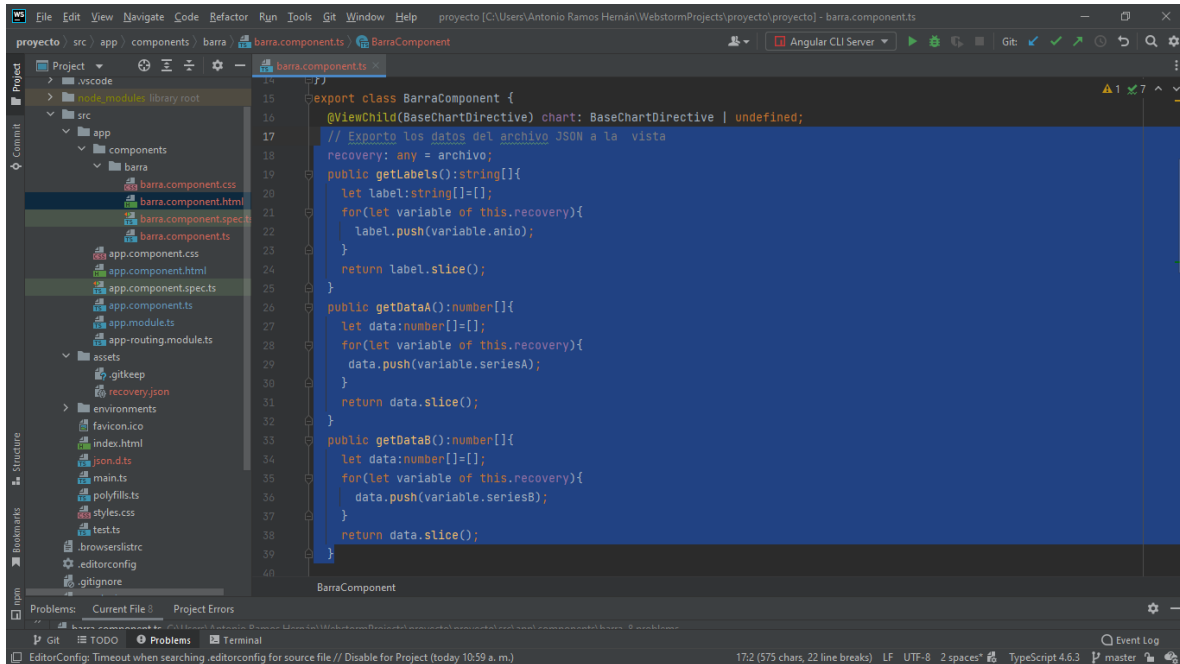
// Importo el archivo JSON
import archivo from "../../assets/recovery.json";

```



3. Importación del archivo a un componente JSON:

Posterior a ello, dentro de la clase BarraComponent y debajo de @ViewChild, debemos de colocar la variable recovery la cual va a obtener los datos del archivo json, posterior a ello se colocan 3 métodos la cual una va a recuperar los datos de los labels, otra del dataA y otro del dataB. De esta forma se puede recuperar todo tipo de información dependiendo del formato json que se haya hecho.



// Exporto los datos del archivo JSON a la vista

recovery: any = archivo;

public getLabels():string[]{

let label:string[]=[];

for(let variable of this.recovery){

label.push(variable.anio);

}

return label.slice();

}

public getDataA():number[]{

let data:number[]=[];

for(let variable of this.recovery){

data.push(variable.seriesA);

}

return data.slice();

}

public getDataB():number[]{

let data:number[]=[];

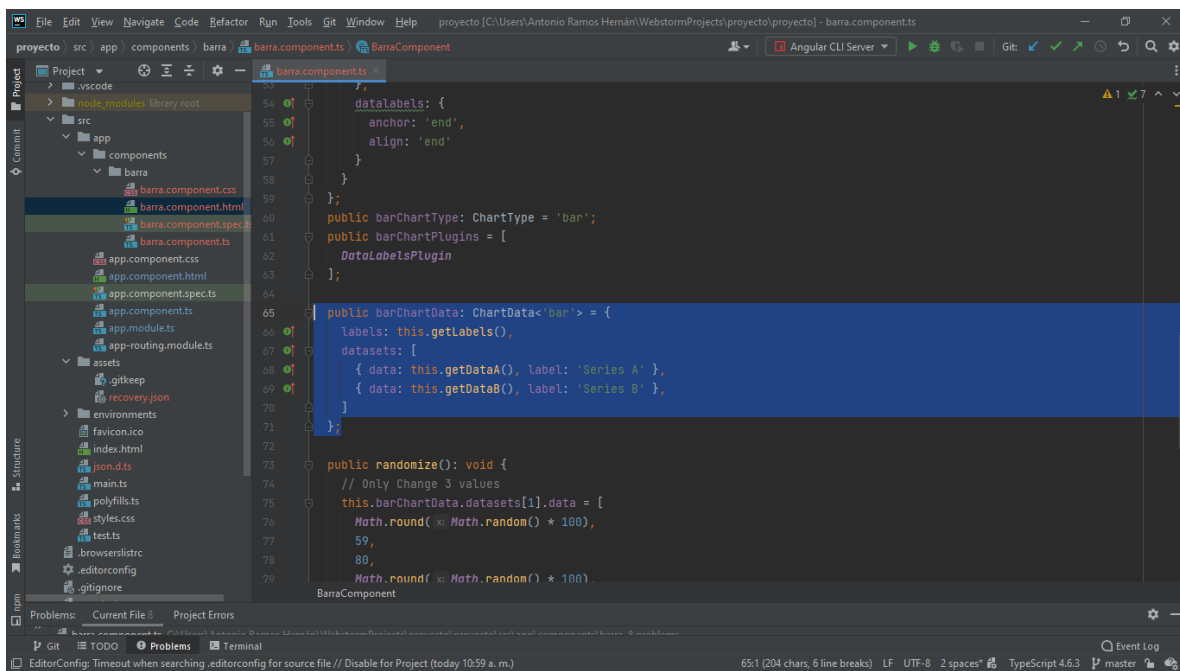
```

for(let variable of this.recovery){
  data.push(variable.seriesB);
}
return data.slice();
}

```

4. Pasar datos a la gráfica:

Para llamar los datos a la gráfica solo se debe de modificar la variable barChartData, por el siguiente código. El cambio que se hizo fue cambiar los arreglos que se habían introducido como argumento y colocar el método correspondiente que hicimos en el paso anterior, la cual cada método retornará un arreglo.



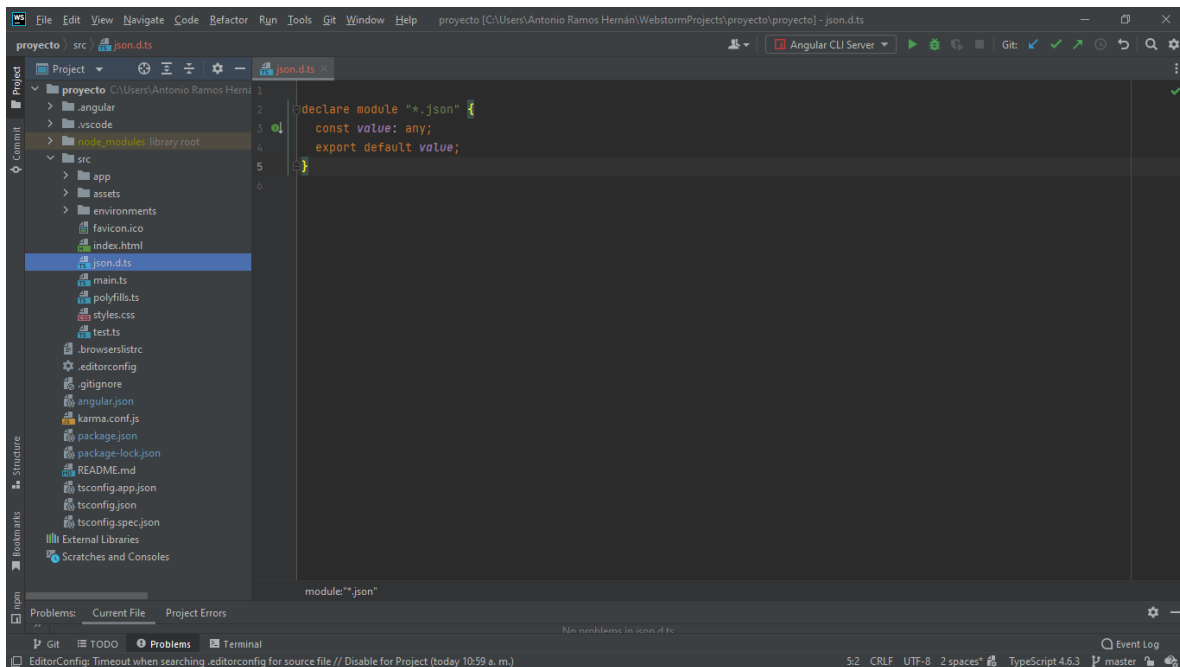
```

public barChartData: ChartData<'bar'> = {
  labels: this.getLabels(),
  datasets: [
    { data: this.getDataA(), label: 'Series A' },
    { data: this.getDataB(), label: 'Series B' },
  ]
};

```

5. Archivo de configuración json.d.ts:

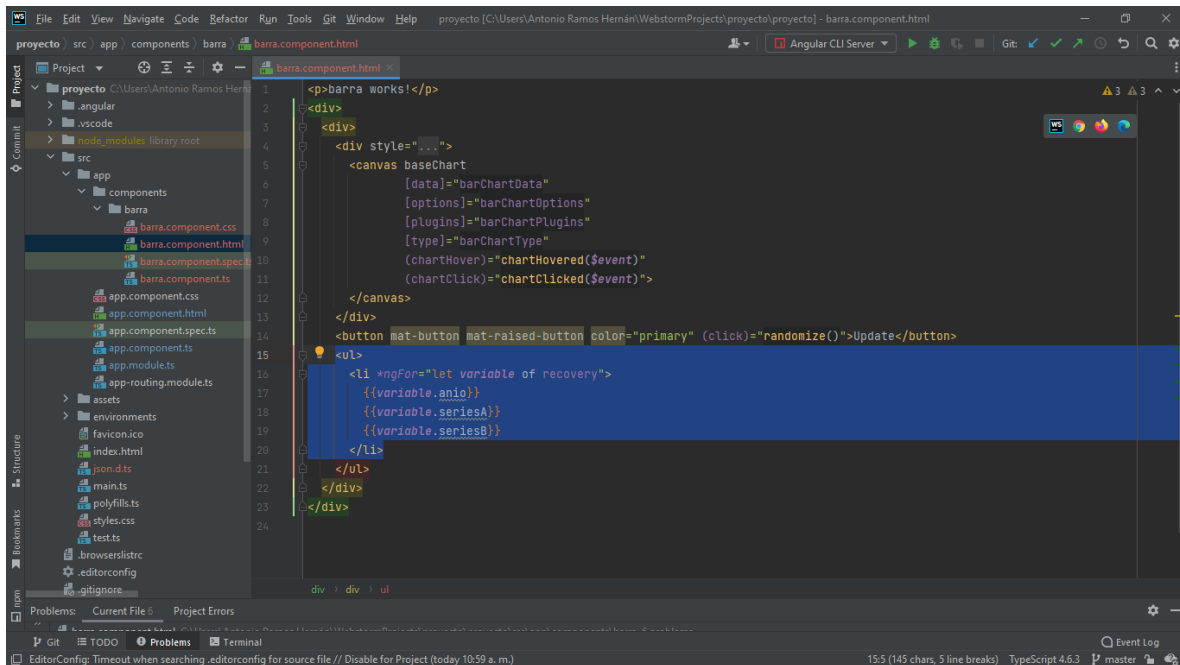
Creamos otro archivo llamado json.d.ts dentro de la carpeta src para configurar el modulo de json, el código solo consiste en 4 líneas de código como viene en la imagen.



```
declare module "*.json" {  
  const value: any;  
  export default value;  
}
```


6. Colocar información desde el html:

También podemos llamar la información en barra.component.html con directivas, en este ejemplo manda a llamar los datos del año, seriesA y seriesB de nuestro json. Para ello se debe de usar un *ngFor para crear un bucle la cual vaya leyendo cada información hasta llegar a su último índice. En este ejemplo se colocó el código de una etiqueta la cual enlista cada iterador.



```
<p>barra works!</p>
<div>
  <div style="...">
    <canvas baseChart
      [data]="barChartData"
      [options]="barChartOptions"
      [plugins]="barChartPlugins"
      [type]="barChartType"
      (chartHover)="chartHovered($event)"
      (chartClick)="chartClicked($event)">
    </canvas>
  </div>
  <button mat-button mat-raised-button color="primary" (click)="randomize()">Update</button>
  <ul>
    <li *ngFor="let variable of recovery">
      {{variable.anio}}
      {{variable.seriesA}}
      {{variable.seriesB}}
    </li>
  </ul>
</div>
</div>
```

```
<ul>
  <li *ngFor="let variable of recovery">
    {{variable.anio}}
    {{variable.seriesA}}
    {{variable.seriesB}}
  </li>
```

7. Mostrar graficas de barras con datos de un json:

Finalmente ejecutamos el código `ng serve -o` y visualizar la nueva grafica de barras creada a partir de nueva información obtenida con el archivo json. A diferencia del anterior que llegaba hasta la etiqueta 2012.



Link del repositorio:

<https://github.com/AntonioRamosH/Modulo-Graficas-de-Barras/tree/master>