Laboratorio – Uso de Formularios en Angular

Los formularios reactivos son uno de los dos enfoques para construir formularios que ofrece Angular. Ambos enfoques procesan y gestionan los datos del formulario de forma diferente y, por lo tanto, ambos tienen sus propias ventajas. Para soluciones muy simples, que no requieren reutilización, los formularios basados en plantillas son una excelente opción. Esto es particularmente cierto para los equipos que no tienen mucha experiencia en formularios o reactivos. En la mayoría de los otros casos de uso, los beneficios de usar formularios reactivos realmente comienzan a brillar; especialmente cuando su aplicación escala.

@angular/forms

Implementa un conjunto de directivas y proveedores para comunicarse con los elementos nativos del DOM cuando se construyen formularios para capturar la entrada del usuario.

Utiliza esta API para registrar directivas, construir modelos de formularios y datos, y proporcionar validación a tus formularios. Los validadores pueden ser síncronos o asíncronos dependiendo de tu caso de uso. También puedes ampliar la funcionalidad incorporada proporcionada por los formularios en Angular utilizando las interfaces y los tokens para crear validadores y elementos de entrada personalizados.

Los formularios de Angular le permiten:

- Capturar el valor actual y el estado de validación de un formulario.
- Seguir y escuchar los cambios en el modelo de datos del formulario.
- Validar la corrección de la entrada del usuario.
- Crear validadores y elementos de entrada personalizados.

Se puede construir formularios de dos maneras:

- Los formularios reactivos utilizan instancias existentes de un FormControl o FormGroup para construir un modelo de formulario. Este modelo de formulario se sincroniza con los elementos de entrada del formulario a través de directivas para rastrear y comunicar los cambios al modelo de formulario. Los cambios en el valor y el estado de los controles se proporcionan como observables.
- Los formularios basados en plantillas dependen de directivas como NgModel y NgModelGroup que crean el modelo de formulario por ti, por lo que cualquier cambio en el formulario se comunica a través de la plantilla

FormsModule

Exporta los proveedores y las directivas necesarias para los formularios basados en plantillas, haciéndolos disponibles para ser importados por NgModules que importen este módulo.

ReactiveFormsModule

Exporta la infraestructura y las directivas necesarias para los formularios reactivos, haciéndolos disponibles para la importación por parte de los NgModules que importan este módulo.

Los formularios reactivos nos proporcionan una manera de crear formularios inmutables y un enfoque orientado a objetos para crear formularios. De esta manera podemos declarar y manejar fácilmente todas las validaciones y eventos del formulario en un solo lugar y reutilizar el código.

Antes de empezar a construir nuestro propio formulario, es un buen momento para explicar los tres tipos principales de componentes de formulario en Angular forms. Es una buena idea familiarizarse con estos tres, ya que los utilizarás con bastante frecuencia.

FormControls: Son los controles individuales de un formulario que mantienen el valor y la validez.

FormGroup: Es una colección de controles que llevan la cuenta del valor y la validez.

FormArrays: Son un array o lista de controles que mantienen un seguimiento del valor y la validez.

Implementación

Tomando como base el laboratorio anterior, se va a construir un modulo que mostrara diferentes tipos de formularios que pueden desarrollarse en Angular. Este ejercicio también es interesante debido a que se trata de un modulo que llama a otros módulos dentro de su estructura, es decir submodulos.

Se creara el modulo forms dentro de app, usando GENERATE MODULE.
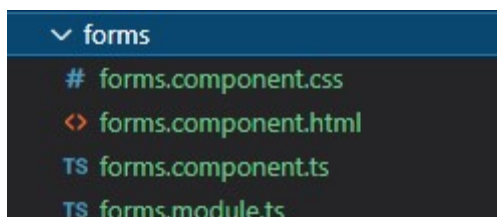
Como en anteriores ocasiones, no se ha creado automáticamente el archivo *routing.module.ts*, por lo que se debe agregar, mediante la línea de comandos el siguiente código:

```
ng generate module forms-routing --module forms --flat
```

```
C:\Windows\system32\cmd.exe                                              —    □    ×

Microsoft Windows [Versión 10.0.19043.1466]
(c) Microsoft Corporation. Todos los derechos reservados.

C:\tareatacna\routing-prueb-modulo\src\app\forms>ng generate module forms-routing --module forms --flat
CREATE src/app/forms/forms-routing.module.ts (198 bytes)
UPDATE src/app/forms/forms.module.ts (340 bytes)

C:\tareatacna\routing-prueb-modulo\src\app\forms>
```

Luego, se ingresara el código de los elementos iniciales de este módulo.

forms.component.ts

sin cambios

forms.component.html

```
<div class="container bd-layout">

  <div class="bd-sidebar">

    <ul class="list-unstyled p-2">

      <li class="mb-1">

        <button class="btn btn-toggle align-items-center rounded collapsed" data-bs-toggle="collapse"

          data-bs-target="#reactive-collapse"                 aria-expanded="true" aria-current="true">

          Reactive

        </button>

        <div class="collapse show" id="reactive-collapse">

        <ul class="btn-toggle-nav list-unstyled fw-normal pb-1 small">

          <li><a   routerLink="/forms/prototype"   class="link-dark rounded">Prototype</a></li>

          <li><a routerLink="/forms/form-control" class="link-dark rounded">form-control</a></li>

          <li><a         routerLink="/forms/form-control-class" class="link-dark rounded">form-control-class</a></li>

          <li><a   routerLink="/forms/form-group"   class="link-dark rounded">form-group</a></li>
```

```html
            <li><a routerLink="/forms/form-builder" class="link-
dark rounded">form-builder</a></li>
            <li><a       routerLink="/forms/form-builder-nested"
class="link-dark rounded">form-builder-nested</a></li>
            <li><a  routerLink="/forms/form-array"  class="link-
dark rounded">form-array</a></li>
            <li><a  routerLink="/forms/form-multi"  class="link-
dark rounded">form-multi</a></li>
        </ul>
      </div>
    </li>
    <li class="mb-1">
      <button class="btn btn-toggle align-items-center rounded
collapsed" data-bs-toggle="collapse"
        data-bs-target="#template-collapse"              aria-
expanded="false">
        Basado en plantillas
      </button>
      <div class="collapse" id="template-collapse">
        <ul class="btn-toggle-nav list-unstyled fw-normal pb-1
small">
          <li><a  routerLink="/forms/single"  class="link-dark
rounded">single</a></li>
          <li><a   routerLink="/forms/multi"   class="link-dark
rounded">multi</a></li>
          <li><a  routerLink="/forms/init-class"  class="link-
dark rounded">init-class</a></li>
        </ul>
      </div>
    </li>
  </ul>
</div>


<div class="row">
```

```
    <div class="col p-4">

      <router-outlet></router-outlet>

    </div>

  </div>


</div>
```

forms.component.css

```css
.btn-toggle {

  display: inline-flex;

  align-items: center;

  padding: .25rem .5rem;

  font-weight: 600;

  color: rgba(0, 0, 0, .65);

  background-color: transparent;

  border: 0;

}


.btn-toggle:hover, .btn-toggle:focus {

  color: rgba(0, 0, 0, .85);

  background-color: #cfe2ff;

}


.btn-toggle::before {

  width: 1.25em;

  line-height: 0;

  content: url("data:image/svg+xml,%3csvg
xmlns='http://www.w3.org/2000/svg' width='16' height='16'
viewBox='0 0 16 16'%3e%3cpath fill='none'
stroke='rgba%280,0,0,.5%29' stroke-linecap='round' stroke-
linejoin='round' stroke-width='2' d='M5 14l6-6-6-
6'/%3e%3c/svg%3e");

  transition: transform .35s ease;
```

```css
  transform-origin: .5em 50%;
}


.btn-toggle[aria-expanded="true"] {
  color: #0d6efd;
}


.btn-toggle[aria-expanded="true"]::before {
  transform: rotate(90deg);
}


.btn-toggle-nav a {
  display: inline-flex;
  padding: .1875rem .5rem;
  margin-top: .125rem;
  margin-left: 1.25rem;
  text-decoration: none;
}


.btn-toggle-nav a:hover, .btn-toggle-nav a:focus {
  color: #0d6efd;
  background-color: #cfe2ff;
  font-weight: bold;
}


@media (min-width: 768px) {
  .bd-layout {
    display: grid;
    gap: 1.5rem;
    grid-template-areas: "sidebar main";
```

```
      grid-template-columns: 1fr 3fr

    }

}


@media (min-width: 992px) {

  .bd-layout {

    grid-template-columns: 1fr 5fr

  }

}


.bd-sidebar {

  overflow: auto;

  font-weight: 600

}


@media (min-width: 768px) {

  .bd-sidebar {

    position: -webkit-sticky;

    position: sticky;

    top: 5rem;

    display: block !important;

    height: calc(100vh - 7rem);

    padding-left: .25rem;

    margin-left: -.25rem;

    overflow-y: auto

  }

}
```

el siguiente paso es el registro en module.ts

forms.module.ts

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

```typescript
import { CommonModule } from '@angular/common';

import { FormsComponent } from './forms.component';

import { FormsRoutingModule } from './forms-routing.module';

import { HttpClientModule } from '@angular/common/http';

import { RouterModule } from '@angular/router';


@NgModule({
  imports: [

    CommonModule,

    FormsRoutingModule,

    HttpClientModule,

    RouterModule

  ],
    exports: [

      FormsComponent,

  ],
  declarations: [FormsComponent]
})
export class FormsModule { }
```

como se observa, se ha agregado al modulo **HttpClientModule** a las declaraciones, asi como **RouterModule**, indispensable para usar la etiqueta <router-outlet></router-outlet>.

Luego se establece el registro de routing.module.ts

forms.routing.module.ts

```typescript
import { NgModule } from '@angular/core';

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

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

import { FormsComponent } from './forms.component';


const routes: Routes = [

  { path: '', component: FormsComponent, children: [
```
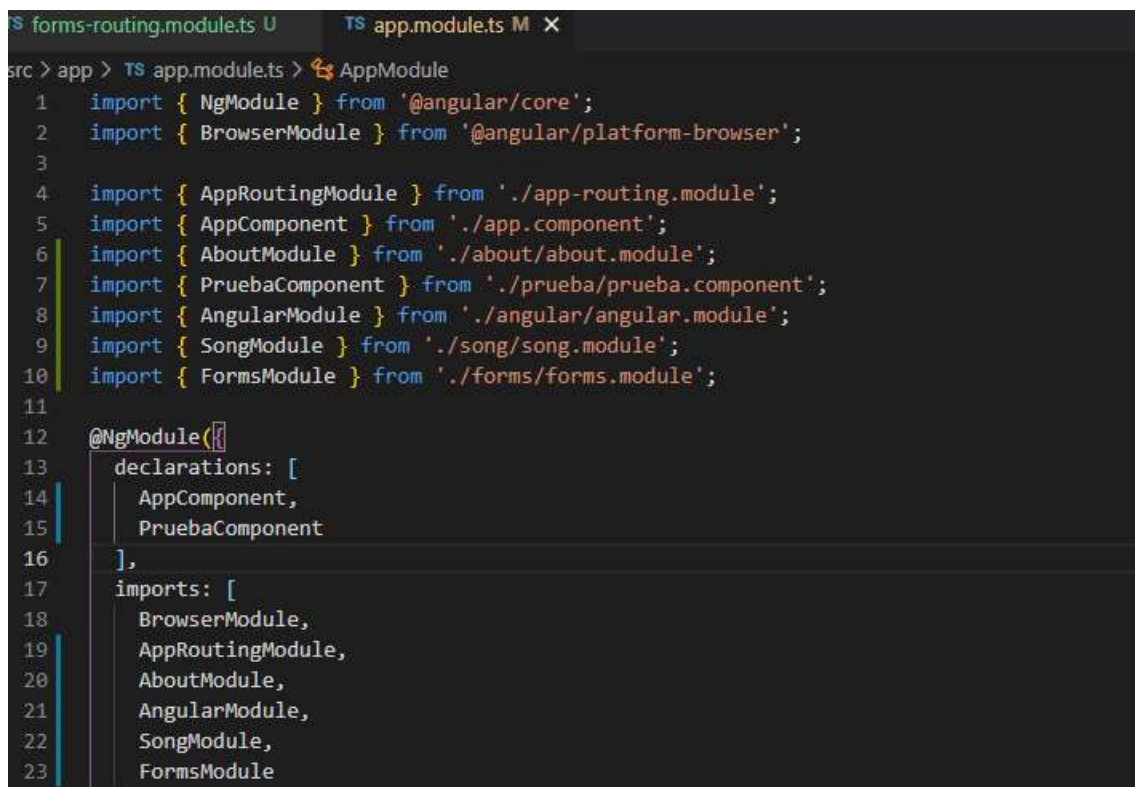
```
        { path: 'forms', component: FormsComponent }
    ]}
]


@NgModule({
    declarations: [],
    imports: [CommonModule, RouterModule.forChild(routes)],
    exports: [RouterModule]
})
export class FormsRoutingModule { }
```

A continuación, se agregara el registro en app.module.ts:

luego, se crea el route para este módulo en routing.module.ts inicial

```
import { PruebaComponent } from './prueba/prueba.component';

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

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

import { AboutModule } from './about/about.module';
```

```typescript
import { AngularModule } from './angular/angular.module';

import { SongModule } from './song/song.module';

import { FormsModule } from './forms/forms.module';


const routes: Routes = [
  { path: 'about', loadChildren: () => import(`./about/about.module`).then(m => m.AboutModule) },
  { path: 'angular', loadChildren: () => import(`./angular/angular.module`).then(m => m.AngularModule) },
  { path: 'song', loadChildren: () => import(`./song/song.module`).then(m => m.SongModule) },
  { path: 'forms', loadChildren: () => import('./forms/forms.module').then(m => m.FormsModule) },
  { path: '', component: PruebaComponent}
];


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

Ahora, el siguiente paso es crear submodulos del módulo **forms**

a. Modulo single

este submodulo será parte de la pantalla inicial de Forms.

Como siempre debe crear el archivo routing de este módulo.

```
ng generate module single-routing --module single --flat
```

se ingresarán los datos de este submodulo:

archivo **ítems.ts**

```
export const ITEMS: any[] =

  [

    { id: 1, name: 'prototype', link: 'prototype', icon: 'far fa-
address-card' },

    { id: 2, name: 'form-control', link: 'form-control', icon:
'fas fa-user' },

    { id: 3, name: 'form-control-class', link: 'form-control-
class', icon: 'fas fa-user-friends' },

    { id: 4, name: 'form-group', link: 'form-group', icon: 'fas
fa-house-user' },

    { id: 5, name: 'form-builder', link: 'form-builder', icon:
'fab fa-app-store' },

    { id: 6, name: 'form-builder-nested', link: 'form-builder-
nested', icon: 'fab fa-artstation' },

    { id: 7, name: 'form-array', link: 'form-array', icon: 'fab
fa-asymmetrik' },

    { id: 8, name: 'form-multi', link: 'form-multi', icon: 'fas
fa-atom' },

    { id: 9, name: 'single', link: 'single2', icon: 'fab fa-
centos' },

    { id: 10, name: 'multi', link: 'multi', icon: 'fas fa-chart-
line' },

    { id: 11, name: 'init-class', link: 'init-class', icon: 'fas
fa-cogs' },


  ];
```

single.component.ts

```
<div class="row mb-1">

 <div class="col-md-12 text-center">
```

```html
    <h2 class="h4">Forms Features  <i class="fab fa-bootstrap"></i></h2>
  </div>
</div>


<div class="row pt-2">
  <div *ngFor="let item of items" class="col-12 col-sm-12 col-md-6 col-lg-4 col-xl-3 mb-2">
    <div class="nga-card-rotate">
      <a style="cursor:pointer" routerLink="/forms/{{ item.link }}">
        <div class="card">
          <div class="nga-card-header">
            <div class="row mt-2">
              <div class="col-9 col-xl-9">
                <h5 class="card-title text-primary">{{ item.name }}</h5>
              </div>
              <div class="col-3 col-xl-3">
                <i class="{{ item.icon }} fa-lg text-dark"></i>
              </div>
            </div>
          </div>
        </div>
      </a>
    </div>
  </div>
</div>
```

single.component.css

```css
.card {
 -webkit-transition: -webkit-transform 1s;
 -moz-transition: -moz-transform 1s;
 -o-transition: -o-transform 1s;
```

```css
  transition: transform 1s;

  -webkit-transform-style: preserve-3d;

  -moz-transform-style: preserve-3d;

  -o-transform-style: preserve-3d;

  transform-style: preserve-3d;

  position: relative;

}


.nga-card-header {

  padding: 0.5rem 1rem;

  margin-bottom: 0;

  background-color: white;

}


.nga-card-rotate {

  -webkit-perspective: 800px;

  -moz-perspective: 800px;

  -o-perspective: 800px;

  perspective: 800px;

  margin-bottom: 30px;

}


.nga-card-rotate a {

  text-decoration: none;

}


.nga-card-rotate:not(.manual-flip):hover .card,

.nga-card-rotate.hover.manual-flip .card {

  -webkit-transform: rotateY(180deg);
```

```
 -moz-transform: rotateY(180deg);

 -o-transform: rotateY(180deg);

 transform: rotateY(180deg);

}
```

se muestra el código de module.ts

single.module.ts

Luego se completa el código con el routing.module.ts

single.routing.module.ts

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

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

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

import { SingleComponent } from './single.component';

const routes: Routes = [

  { path: '', component: SingleComponent, children: [] }

];


@NgModule({

  declarations: [],

  imports: [
```
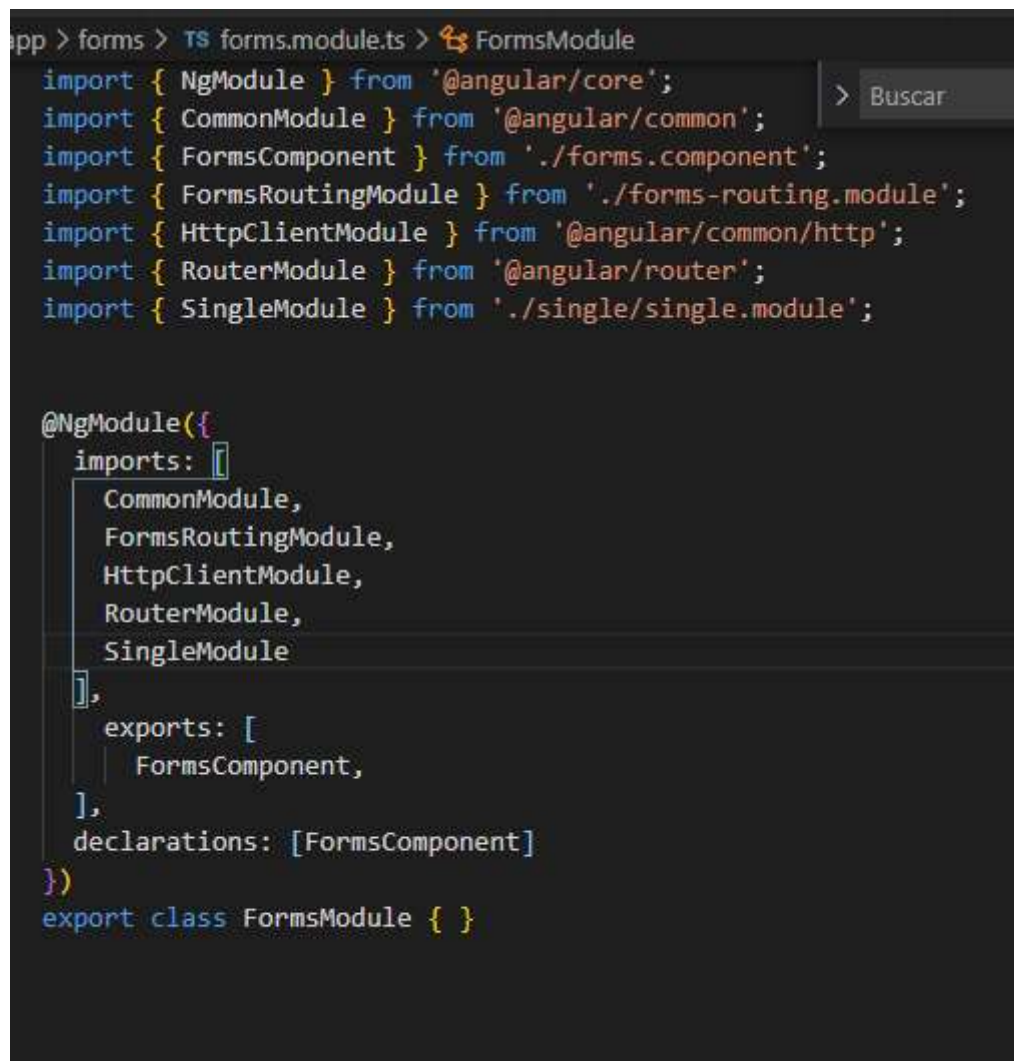
```
    CommonModule,

    RouterModule.forChild(routes)

  ],

  exports: [RouterModule]

})

export class SingleRoutingModule { }
```

Luego, definiremos el módulo en *forms.module.ts*

```
app > forms > TS forms.module.ts > 🏷 FormsModule
  import { NgModule } from '@angular/core';              > Buscar
  import { CommonModule } from '@angular/common';
  import { FormsComponent } from './forms.component';
  import { FormsRoutingModule } from './forms-routing.module';
  import { HttpClientModule } from '@angular/common/http';
  import { RouterModule } from '@angular/router';
  import { SingleModule } from './single/single.module';


  @NgModule({
    imports: [
      CommonModule,
      FormsRoutingModule,
      HttpClientModule,
      RouterModule,
      SingleModule
    ],
    exports: [
      FormsComponent,
    ],
    declarations: [FormsComponent]
  })
  export class FormsModule { }
```

y finalmente, se modifica **forms.routing.module.ts**

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

```
import { CommonModule } from '@angular/common';

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

import { FormsComponent } from './forms.component';

import { SingleModule } from './single/single.module';


const routes: Routes = [
  { path: '', component: FormsComponent, children: [
    { path: '',
      loadChildren: () => import('./single/single.module')
        .then(mod => mod.SingleModule)},
    ]}
]



@NgModule({
  declarations: [],
  imports: [CommonModule, RouterModule.forChild(routes)],
  exports: [RouterModule]
})
export class FormsRoutingModule { }
```
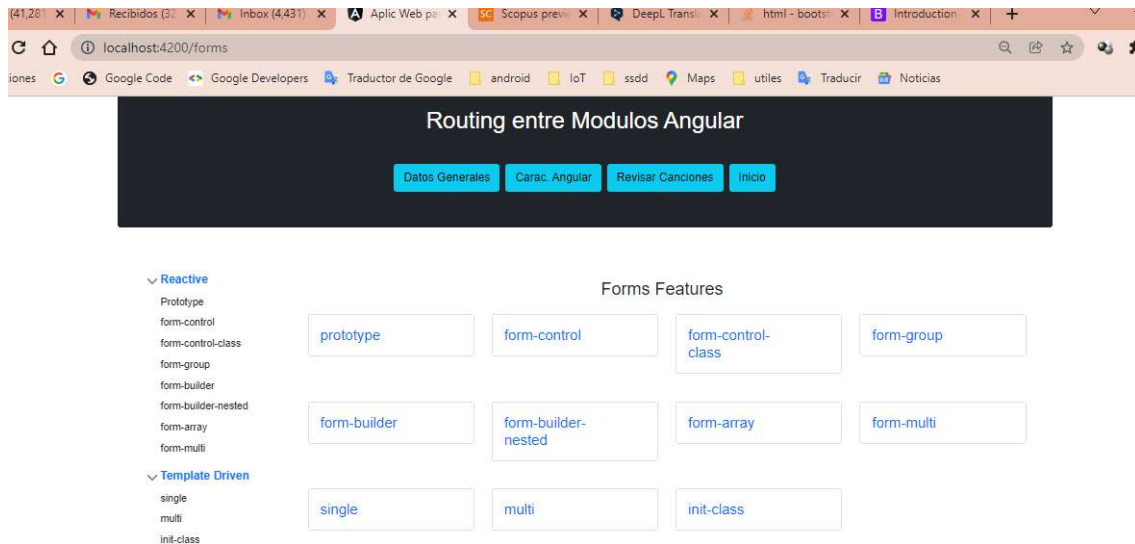
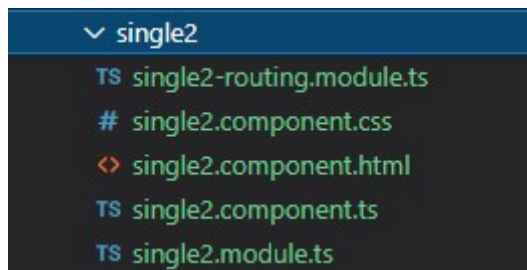una vez hecho esto, se obtiene la pantalla básica de Forms

Una vez que la vista básica se ha construido, se implementaran los submodulos que serán llamados a travez de esta vista.

b. Modulo single2

ng generate module single2-routing --module single2 –-flat

single2component.ts

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


@Component({

  selector: 'app-single2',

  templateUrl: './single2.component.html',

  styleUrls: ['./single2.component.css']

})

export class Single2Component implements OnInit {
```

```
  firstName = 'Paul';

  lastName = 'Atreides';


  constructor() { }


  ngOnInit() {

  }


}
```

single2component.html

```
<div class="card">
  <div class="card-body">
    <h5 class="card-title">Template Driven Form Single Test</h5>
    <hr>
    <div class="row">
      <div class="col-6">
        <form>
          <div class="mb-3">
            <label    for="firstName"    class="form-label">First
Name</label>
            <input        type="text"        class="form-control"
id="firstName" name="firstName" [(ngModel)]="firstName"
              aria-describedby="firstNameHelp">
          </div>
          <div class="mb-3">
            <label    for="lastName"    class="form-label">Last
Name</label>
            <input type="text" class="form-control" id="lastName"
name="lastName" [(ngModel)]="lastName"
```

```
                    aria-describedby="lastNameHelp">

            </div>

            <button        type="submit"        class="btn        btn-
primary">Submit</button>

        </form>

      </div>

      <div class="col-6">

        Input Result

        <div class="alert alert-primary" role="alert">

          firstName : {{ firstName }}

        </div>

        <div class="alert alert-primary" role="alert">

          lastName : {{ lastName }}

        </div>

      </div>

    </div>

  </div>

</div>
```

single2.module.ts

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

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

import { Single2Component } from './single2.component';

import { Single2RoutingModule } from './single2-routing.module';

import { FormsModule } from '@angular/forms';


@NgModule({

  imports: [

    CommonModule,

    Single2RoutingModule,

    FormsModule
```

```
  ],
  declarations: [Single2Component]
})
export class Single2Module { }
```
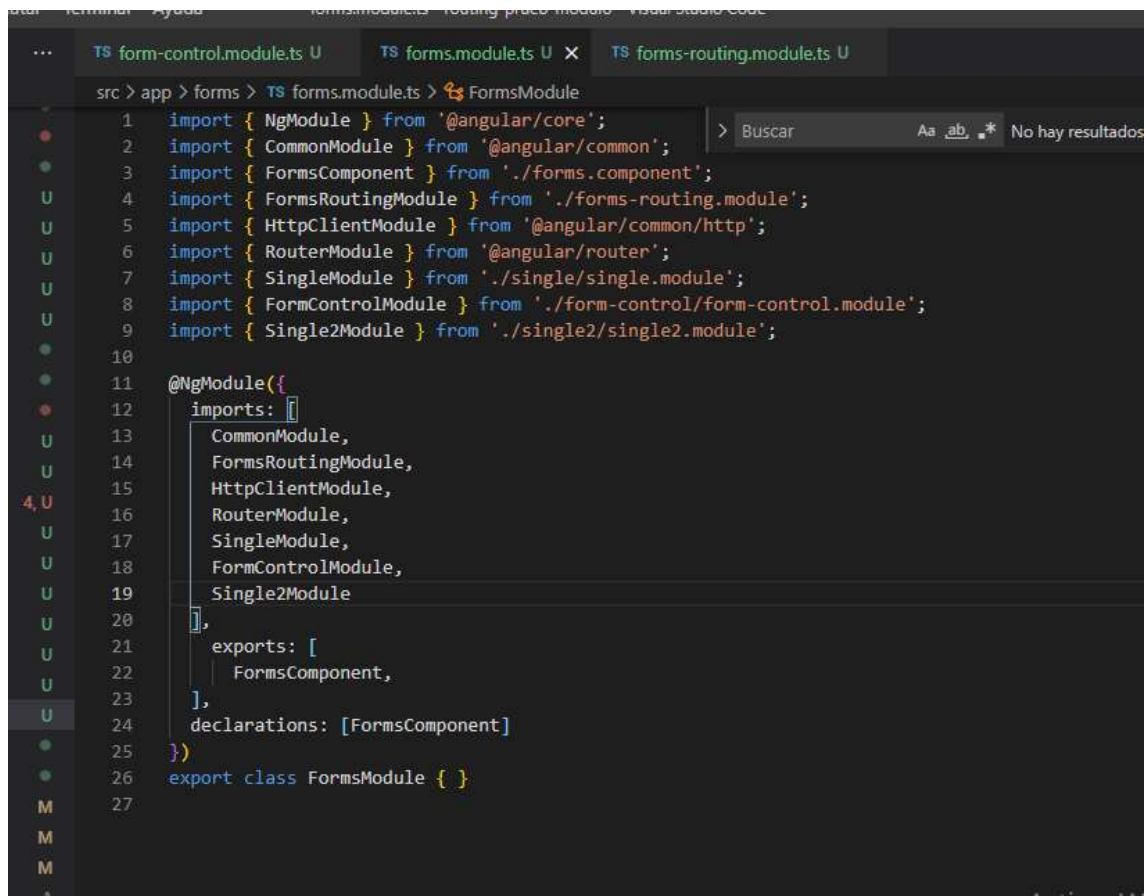
single2.routing.module.ts

```typescript
import { NgModule } from '@angular/core';

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

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

import { Single2Component } from './single2.component';


const routes: Routes = [
  {
    path: '',
    component: Single2Component,
  },
];


@NgModule({
  declarations: [],
  imports: [
    CommonModule,
    RouterModule.forChild(routes)],
  exports: [RouterModule]
})
export class Single2RoutingModule { }
```

se modifica **forms.module.ts**

```
TS form-control.module.ts U      TS forms.module.ts U ✕      TS forms-routing.module.ts U

src > app > forms > TS forms.module.ts > ⚛ FormsModule
 1   import { NgModule } from '@angular/core';
 2   import { CommonModule } from '@angular/common';
 3   import { FormsComponent } from './forms.component';
 4   import { FormsRoutingModule } from './forms-routing.module';
 5   import { HttpClientModule } from '@angular/common/http';
 6   import { RouterModule } from '@angular/router';
 7   import { SingleModule } from './single/single.module';
 8   import { FormControlModule } from './form-control/form-control.module';
 9   import { Single2Module } from './single2/single2.module';
10
11   @NgModule({
12     imports: [
13       CommonModule,
14       FormsRoutingModule,
15       HttpClientModule,
16       RouterModule,
17       SingleModule,
18       FormControlModule,
19       Single2Module
20     ],
21       exports: [
22         FormsComponent,
23     ],
24     declarations: [FormsComponent]
25   })
26   export class FormsModule { }
27
```

finalmente, se modifica **forms.routing.module.ts**

const routes: Routes = [

 { path: '', component: FormsComponent, children: [

  {

    path: '',

    loadChildren: () => import('./single/single.module')

     .then(mod => mod.SingleModule)},

  {

    path: 'form-control',

    loadChildren: () => import('./form-control/form-control.module')

     .then(mod => mod.FormControlModule)

    },

    {

     path: 'single2',

     loadChildren: () => import('./single2/single2.module')
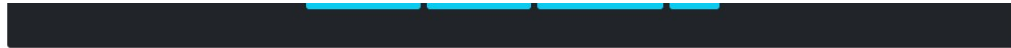
```
        .then(mod => mod.Single2Module)

    },

  ]}

]
```


c. Modulo FormControl

ng generate module form-control-routing --module form-control —-flat
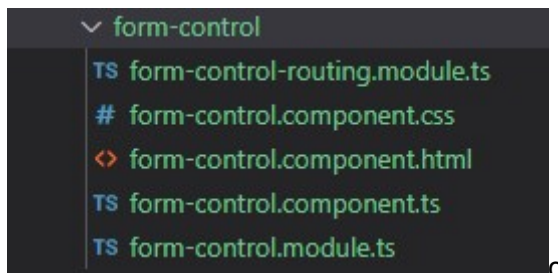
form-control.component.ts

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

import { FormControl } from '@angular/forms';

@Component({

 selector: 'app-form-control',

 templateUrl: './form-control.component.html',

```typescript
  styleUrls: ['./form-control.component.css']
})
export class FormControlComponent implements OnInit {

  name = new FormControl('');
  releaseDate = new FormControl('');
  franchise = new FormControl('');
  budget = new FormControl('');
  worldwide = new FormControl('');
  summary = new FormControl('');
  constructor() { }

  ngOnInit() {
    this.updateControls();
  }

  updateControls(): void {
    this.name.setValue('Avengers: Endgame');
    this.releaseDate.setValue('26/04/2019');
    this.franchise.setValue(true);
    this.budget.setValue('356000000');
    this.worldwide.setValue('2797800564');
    this.summary.setValue('Tras los devastadores acontecimientos de Vengadores: Infinity War (2018), ' +
      'el universo esta en ruinas.');
  }

  resetControls(): void {
    this.name.setValue(null);
    this.releaseDate.setValue(null);
```

```
      this.franchise.setValue(null);

      this.budget.setValue(null);

      this.worldwide.setValue(null);

      this.summary.setValue(null);

  }


}
```

form-control.component.html

```html
<div class="card">
  <div class="card-body">
    <div class="row">
      <div class="col-12 col-sm-12 col-md-7 col-lg-7 col-xl-7">
        <h5 class="card-title text-center text-info">FormControl</h5>
        <form class="row g-3">
          <div class="col-12 col-sm-12 col-md-6 col-lg-6 col-xl-6">
            <label for="name" class="form-label">Name</label>
            <input type="text" class="form-control" id="name" [formControl]="name">
          </div>
          <div class="col-12 col-sm-12 col-md-6 col-lg-6 col-xl-6">
            <label for="releaseDate" class="form-label">Release Date</label>
            <input type="text" class="form-control" id="releaseDate" [formControl]="releaseDate">
          </div>
          <div class="col-12 col-sm-12 col-md-6 col-lg-6 col-xl-6">
            <label for="budget" class="form-label">Budget</label>
            <input type="text" class="form-control" id="budget" [formControl]="budget">
          </div>
          <div class="col-12 col-sm-12 col-md-6 col-lg-6 col-xl-6">
            <label for="worldwide" class="form-label">Worldwide</label>
```

```html
        <input        type="text"        class="form-control"        id="worldwide"
[formControl]="worldwide">

    </div>

    <div class="col-12 col-sm-12 col-md-12 col-lg-12 col-xl-12">

    <label for="summary" class="form-label">Summary</label>

    <textarea    class="form-control"    id="summary"    rows="3"    id="summary"
[formControl]="summary"></textarea>

    </div>

    <div class="col-12 col-sm-12 col-md-4 col-lg-4 col-xl-4">

    <div class="form-check">

      <input      class="form-check-input"      type="checkbox"      id="franchise"
[formControl]="franchise">

      <label class="form-check-label" for="franchise">

       Franchise

      </label>

     </div>

    </div>

    <div class="col-12 col-sm-12 col-md-4 col-lg-4 col-xl-4">

     <button  type="submit"  (click)="resetControls()"  class="btn  btn-primary  btn-
sm">Reset Controls</button>

    </div>

    <div class="col-12 col-sm-12 col-md-4 col-lg-4 col-xl-4">

     <button  type="submit"  (click)="updateControls()"  class="btn  btn-primary  btn-
sm">Update Controls</button>

    </div>

   </form>

  </div>

  <div class="col-12 col-sm-12 col-md-5 col-lg-5 col-xl-5">

   <h5 class="card-title text-center text-info">FormControl Result</h5>

   <strong>Name</strong> : {{ name.value }}<br>

   <strong>Release Date</strong> : {{ releaseDate.value }}<br>
```

```html
        <strong>Budget</strong> : {{ budget.value }}<br>

        <strong>Worldwide</strong> : {{ worldwide.value }}<br>

        <strong>Summary</strong> : {{ summary.value }}<br>

        <strong>Franchise</strong> : {{ franchise.value }}<br>

      </div>

    </div>

  </div>

</div>
```

form-control.module.ts

```typescript
import { NgModule } from '@angular/core';

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

import { FormControlComponent } from './form-control.component';

import { FormControlRoutingModule } from './form-control-routing.module';

import { FormsModule, ReactiveFormsModule } from '@angular/forms';


@NgModule({
  imports: [
    CommonModule,

    FormControlRoutingModule,

    FormsModule,

    ReactiveFormsModule
  ],
  declarations: [FormControlComponent],
  exports: [
    FormControlComponent
  ],
})
export class FormControlModule { }
```
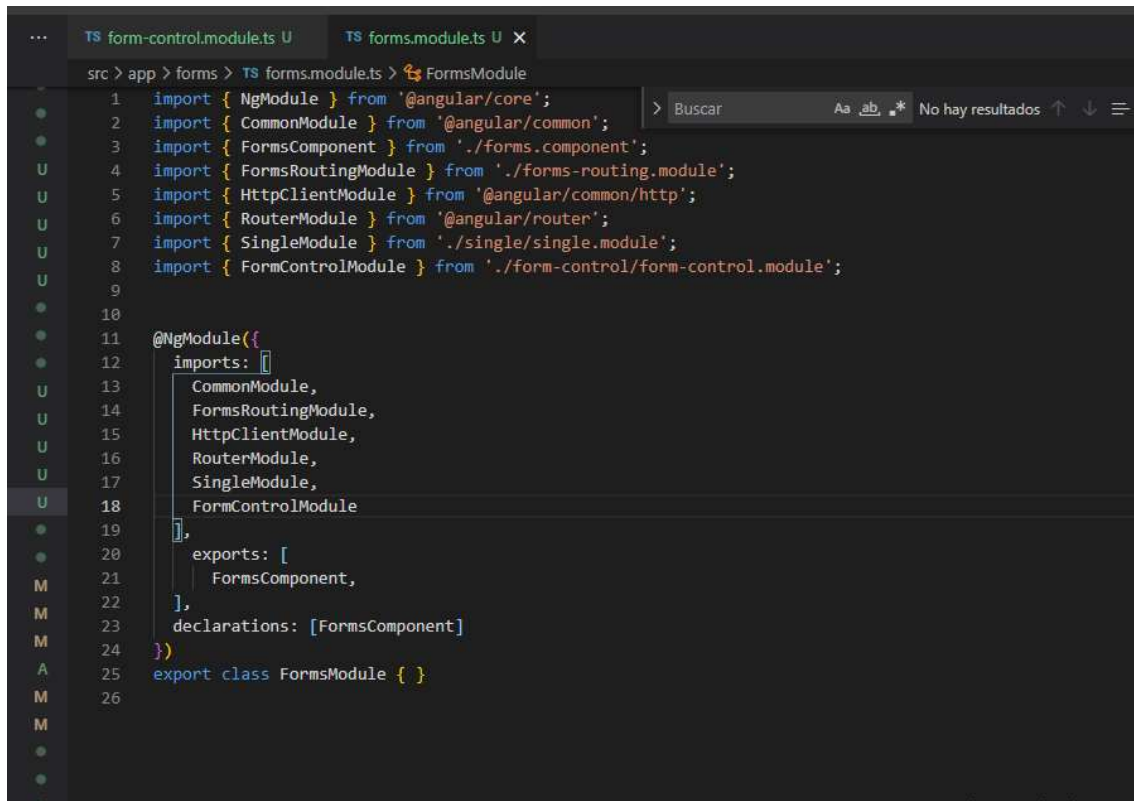
form-control.routing.module.ts

```typescript
import { NgModule } from '@angular/core';

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

import { FormControlComponent } from './form-control.component';

import { FormControlRoutingModule } from './form-control-routing.module';

import { FormsModule, ReactiveFormsModule } from '@angular/forms';

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


const routes: Routes = [
  { path: '', component: FormControlComponent, children: [] }
];


@NgModule({
  imports: [
    CommonModule,
    FormControlRoutingModule,
    FormsModule,
    ReactiveFormsModule,
    RouterModule.forChild(routes)
  ],
  declarations: [FormControlComponent],
  exports: [
    FormControlComponent, RouterModule
  ],
})
export class FormControlModule { }
```

modificando **forms.module.ts**

```
···        TS form-control.module.ts U        TS forms.module.ts U  ×

src > app > forms > TS forms.module.ts > ⚡ FormsModule
     1    import { NgModule } from '@angular/core';          > Buscar              Aa ab .*  No hay resultados  ↑  ↓  ☰
     2    import { CommonModule } from '@angular/common';
     3    import { FormsComponent } from './forms.component';
     4    import { FormsRoutingModule } from './forms-routing.module';
     5    import { HttpClientModule } from '@angular/common/http';
     6    import { RouterModule } from '@angular/router';
     7    import { SingleModule } from './single/single.module';
     8    import { FormControlModule } from './form-control/form-control.module';
     9
    10
    11    @NgModule({
    12      imports: [
    13        CommonModule,
    14        FormsRoutingModule,
    15        HttpClientModule,
    16        RouterModule,
    17        SingleModule,
    18        FormControlModule
    19      ],
    20        exports: [
    21          FormsComponent,
    22      ],
    23      declarations: [FormsComponent]
    24    })
    25    export class FormsModule { }
    26
```

y modificando **forms.routing.module.ts**

const routes: Routes = [

  { path: '', component: FormsComponent, children: [

   {

     path: '',

     loadChildren: () => import('./single/single.module')

      .then(mod => mod.SingleModule)},

   {

     path: 'form-control',

     loadChildren: () => import('./form-control/form-control.module')

       .then(mod => mod.FormControlModule)

     },

   ]}

]

d. Modulo Prototype

ng generate module prototype-routing --module prototype –-flat

prototype.component.ts

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


@Component({
  selector: 'app-prototype',
  templateUrl: './prototype.component.html',
  styleUrls: ['./prototype.component.css']
})
export class PrototypeComponent implements OnInit {
```

```
  constructor() { }


  ngOnInit() {

  }


}
```

prototype.component.html

```html
<div class="card">
  <div class="card-body">
    <form class="row g-3">
      <div class="col-md-6">
        <label for="inputEmail4" class="form-label">Email</label>
        <input type="email" class="form-control" id="inputEmail4">
      </div>
      <div class="col-md-6">
        <label for="inputPassword4" class="form-label">Password</label>
        <input type="password" class="form-control" id="inputPassword4">
      </div>
      <div class="col-12">
        <label for="inputAddress" class="form-label">Address</label>
        <input type="text" class="form-control" id="inputAddress" placeholder="1234 Main St">
      </div>
      <div class="col-12">
        <label for="inputAddress2" class="form-label">Address 2</label>
```

```html
      <input type="text" class="form-control"
id="inputAddress2" placeholder="Apartment, studio, or floor">
    </div>
    <div class="col-md-6">
      <label for="inputCity" class="form-label">City</label>
      <input type="text" class="form-control" id="inputCity">
    </div>
    <div class="col-md-4">
      <label for="inputState" class="form-label">State</label>
      <select id="inputState" class="form-select">
        <option selected>Choose...</option>
        <option>...</option>
      </select>
    </div>
    <div class="col-md-2">
      <label for="inputZip" class="form-label">Zip</label>
      <input type="text" class="form-control" id="inputZip">
    </div>
    <div class="col-12">
      <div class="form-check">
        <input    class="form-check-input"    type="checkbox"
id="gridCheck">
        <label class="form-check-label" for="gridCheck">
          Check me out
        </label>
      </div>
    </div>
    <div class="col-12">
      <button   type="submit"   class="btn   btn-primary">Sign
in</button>
      <button   type="submit"   class="btn   btn-primary">Sign
in</button>
```

```
            <button   type="submit"   class="btn   btn-primary">Sign
in</button>
            <button   type="submit"   class="btn   btn-primary">Sign
in</button>
            <button   type="submit"   class="btn   btn-primary">Sign
in</button>
        </div>
      </form>
    </div>
</div>


<div class="card">
  <div class="card-body">
    <div class="row">
      <div class="col-12 col-sm-12 col-md-7 col-lg-7 col-xl-7">
        <h5       class="card-title      text-center      text-
info">FormControl</h5>
        <form class="row g-3">
          <div class="col-12 col-sm-12 col-md-6 col-lg-6 col-xl-
6">
            <label for="name" class="form-label">Name</label>
            <input  type="text"  class="form-control"  id="name"
value="Avengers: Endgame">
          </div>
          <div class="col-12 col-sm-12 col-md-6 col-lg-6 col-xl-
6">
            <label  for="releaseDate"  class="form-label">Release
Date</label>
            <input      type="text"      class="form-control"
id="releaseDate" value="April 24, 2019">
          </div>
          <div class="col-12 col-sm-12 col-md-6 col-lg-6 col-xl-
6">
```

```html
            <label            for="budget"            class="form-
label">Budget</label>
            <input type="text" class="form-control" id="budget"
value="$356,000,000">
          </div>
          <div class="col-12 col-sm-12 col-md-6 col-lg-6 col-xl-
6">
            <label            for="worldwide"            class="form-
label">Worldwide</label>
            <input           type="text"           class="form-control"
id="worldwide" value="$2,797,800,564">
          </div>
          <div class="col-12 col-sm-12 col-md-6 col-lg-6 col-xl-
6">
            <label            for="domestic"            class="form-
label">Domestic</label>
            <input type="text" class="form-control" id="domestic"
value="$858,373,000">
          </div>
          <div class="col-12 col-sm-12 col-md-6 col-lg-6 col-xl-
6">
            <label         for="international"         class="form-
label">International</label>
            <input           type="text"           class="form-control"
id="international" value="$1,939,427,564">
          </div>
          <div class="col-12 col-sm-12 col-md-12 col-lg-12 col-
xl-12">
            <label            for="summary"            class="form-
label">Summary</label>
            <textarea class="form-control" id="summary" rows="3"
          id="summary">After  the  devastating  events  of
Avengers:   Infinity   War  (2018),   the  universe  is  in
ruins.</textarea>
          </div>
```

```html
            <div class="col-12 col-sm-12 col-md-4 col-lg-4 col-xl-
4">

                <div class="form-check">

                    <input class="form-check-input" type="checkbox"
id="franchise">

                    <label class="form-check-label" for="franchise">

                      Franchise

                    </label>

                </div>

            </div>

            <div class="col-12 col-sm-12 col-md-4 col-lg-4 col-xl-
4">

                <button type="submit" class="btn btn-primary btn-
sm">Create</button>

            </div>

            <div class="col-12 col-sm-12 col-md-4 col-lg-4 col-xl-
4">

                <button type="submit" class="btn btn-primary btn-
sm">Save</button>

            </div>

            <div class="col-12 col-sm-12 col-md-4 col-lg-4 col-xl-
4">

                <button type="submit" class="btn btn-primary btn-
sm">Copy</button>

            </div>

            <div class="col-12 col-sm-12 col-md-4 col-lg-4 col-xl-
4">

                <button type="submit" class="btn btn-primary btn-
sm">Delete</button>

            </div>

        </form>

      </div>

    </div>

  </div>
```
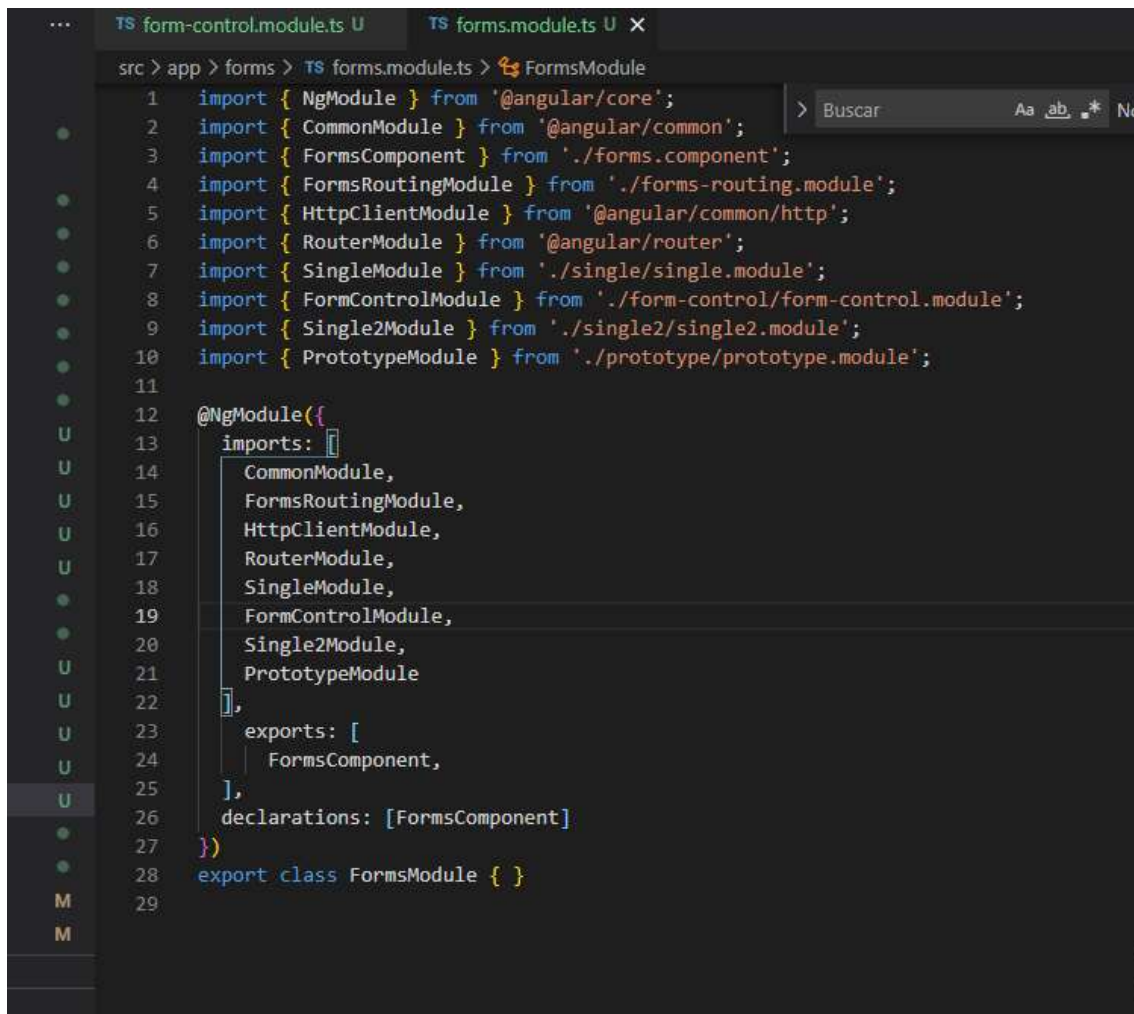
```
</div>
```

prototype.module.ts

```typescript
import { NgModule } from '@angular/core';

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

import { PrototypeComponent } from './prototype.component';

import { PrototypeRoutingModule } from './prototype-
routing.module';

import { FormsModule, ReactiveFormsModule } from
'@angular/forms';



@NgModule({
  imports: [

    CommonModule,

    PrototypeRoutingModule,

    FormsModule,

    ReactiveFormsModule

  ],

  declarations: [PrototypeComponent],

  exports: [

    PrototypeComponent

  ],

})
export class PrototypeModule { }
```

prototype.routing.module.ts

```typescript
import { NgModule } from '@angular/core';

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

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

import { PrototypeComponent } from './prototype.component';
```

```
const routes: Routes = [
  { path: '', component: PrototypeComponent, children: [] }
];


@NgModule({
  declarations: [],
  imports: [
    CommonModule,
    RouterModule.forChild(routes)
  ],
  exports: [RouterModule]
})
export class PrototypeRoutingModule { }
```

A continuación, el registro del modulo en **forms.module.ts**

```ts
TS form-control.module.ts U        TS forms.module.ts U  ×

src > app > forms > TS forms.module.ts > ⟨⟩ FormsModule
  1    import { NgModule } from '@angular/core';
  2    import { CommonModule } from '@angular/common';
  3    import { FormsComponent } from './forms.component';
  4    import { FormsRoutingModule } from './forms-routing.module';
  5    import { HttpClientModule } from '@angular/common/http';
  6    import { RouterModule } from '@angular/router';
  7    import { SingleModule } from './single/single.module';
  8    import { FormControlModule } from './form-control/form-control.module';
  9    import { Single2Module } from './single2/single2.module';
 10    import { PrototypeModule } from './prototype/prototype.module';
 11
 12    @NgModule({
 13      imports: [
 14        CommonModule,
 15        FormsRoutingModule,
 16        HttpClientModule,
 17        RouterModule,
 18        SingleModule,
 19        FormControlModule,
 20        Single2Module,
 21        PrototypeModule
 22      ],
 23        exports: [
 24          FormsComponent,
 25      ],
 26      declarations: [FormsComponent]
 27    })
 28    export class FormsModule { }
 29
```

y finalmente definiendo la ruta en **forms.routing.module.ts**

```ts
const routes: Routes = [
  { path: '', component: FormsComponent, children: [
    {
      path: '',
      loadChildren: () => import('./single/single.module')
        .then(mod => mod.SingleModule)},
    {
      path: 'form-control',
      loadChildren:    ()    =>    import('./form-control/form-
control.module')
          .then(mod => mod.FormControlModule)
    },
    {
```

```
      path: 'single2',

      loadChildren: () => import('./single2/single2.module')

        .then(mod => mod.Single2Module)

    },

    {

      path: 'prototype',

      loadChildren:                    ()                    =>
import('./prototype/prototype.module')

        .then(mod => mod.PrototypeModule)

    },

  ]}

]
```