Laboratorio – Operación CRUD en Angular

Se crea un proyecto denominado crud-angular

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

C:\tareatacna>ng new crus-angular
^C
C:\tareatacna>ng new crud-angular
? 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                                             ]
```

Una vez aquí, se comenzará agregando Bootstrap 5 a la página index.html.

```
<link rel="stylesheet" href="https://cdn.jsdelivr.net/npm/bootstrap@5.0.2/dist/css/bootstrap.min.css"
     rel="stylesheet" integrity="sha384-EVSTQN3/azprG1Anm3QDgpJLIm9Nao0Yz1ztcQTwFspd3yD65VohhpuuCOmLASjC"
     crossorigin="anonymous">
</head>
```

index.html

```
<!doctype html>
<html lang="en">
<head>
  <meta charset="utf-8">
  <title>CrudAngular</title>
  <base href="/">
  <meta name="viewport" content="width=device-width, initial-scale=1">
  <link rel="icon" type="image/x-icon" href="favicon.ico">
```

```
  <link                                          rel="stylesheet"
href="https://cdn.jsdelivr.net/npm/bootstrap@5.0.2/dist/css/boot
strap.min.css"
     rel="stylesheet"                          integrity="sha384-
EVSTQN3/azprG1Anm3QDgpJLIm9Nao0Yz1ztcQTwFspd3yD65VohhpuuCOmLASjC
"
     crossorigin="anonymous">
</head>
</head>
<body>
  <app-root></app-root>
</body>
</html>
```

Se define los estilos generales en styles.css

styles.css

```
body {
  padding-top: 3.5rem;
  font-family: "Roboto", sans-serif;
}
```

A continuación, se agregarán los archivos que se han usado en la carpeta assets de los laboratorios anteriores.

A continuación, en el archivo environment.ts debe agregarse los siguientes valores:

environment.ts

```
export const environment = {
  production: false,


  config: {
    /* SELECT ONE OF THOSE CONFIGURATIONS */


    /* LOCAL JSON (NO CRUD) */
    api: false,
```

```
    url: './assets/params/json/crud/',

    /* LOCAL REST API CRUD WITH POSTGRESQL */
    /* api: true,
    url: 'http://localhost:5004/', */
  },
};
```

a continuación, se definirán los elementos del component App

app.component.ts

sin cambios

app.component.html

```
<div class="container py-5">
  <div class="row justify-content-center">
    <div class="col-3 d-none d-lg-inline-block d-xl-inline-block">
      <div class="row">
        <div class="col mb-2">
          <span class="ml-4 text-primary"><strong>Features List</strong></span>
        </div>
      </div>
      <div class="row">
        <div class="col">
          <div class="accordion" id="accordionItems">
            <div class="nga-card-menu">
              <div class="card-header">
                <a data-bs-toggle="collapse" data-parent="#accordionItems" href="#collapseItems" aria-expanded="true">
```

```
                  <div class="card-item">
                    <i  class="fas  fa-film  me-2  "></i>Movies  /
Shows
                  </div>
                </a>
              </div>
              <div id="collapseItems" class="collapse show" aria-
labelledby="headingOne" data-parent="#accordionItems">
                <div class="card-body">
                  <ul class="list-group text-dark">
                    <li class="list-group-item">
                      <a routerLink="/crud/shows-images">
                        <i class="fas fa-desktop me-2"></i>
                        Shows Image
                      </a>
                    </li>
                    <li class="list-group-item">
                      <a routerLink="/crud/movies-images">
                        <i class="fas fa-film me-2"></i>
                        Movies Image
                      </a>
                    </li>
                    <li               class="list-group-item"><a
routerLink="/crud/shows">
                        <i class="fas fa-desktop me-2"></i>
                        Shows List
                      </a>
                    </li>
                    <li               class="list-group-item"><a
routerLink="/crud/movies">
                        <i class="fas fa-film me-2"></i>
                        Movies List
```

```
                    </a>
                  </li>
                  <li                class="list-group-item"><a
routerLink="/crud/shows/0">
                      <i class="fas fa-desktop me-2"></i>
                      New Show
                    </a>
                  </li>
                  <li                class="list-group-item"><a
routerLink="/crud/movies/0">
                      <i class="fas fa-film me-2"></i>
                      New Movie
                    </a>
                  </li>
                </ul>
              </div>
            </div>
          </div>
          <div class="nga-card-menu">
            <div class="card-header">
              <a        data-bs-toggle="collapse"        data-
parent="#accordionMovies"      href="#collapseMovies"      aria-
expanded="true">
                <div class="card-item">
                  <i class="fas fa-tools me-2 "></i>Generics
                </div>
              </a>
            </div>
            <div   id="collapseMovies"   class="collapse   show"
aria-labelledby="headingOne" data-parent="#accordionMovies">
              <div class="card-body">
                <ul class="list-group text-dark">
```

```
<li class="list-group-item">
  <a routerLink="/crud/continents">
    <i class="fas fa-globe me-2"></i>
    Continents List
  </a>
</li>
<li class="list-group-item">
  <a routerLink="/crud/countries">
    <i class="far fa-flag me-2"></i>
    Countries List
  </a>
</li>
<li class="list-group-item">
  <a routerLink="/crud/cities">
    <i class="fas fa-city me-2"></i>
    Cities List
  </a>
</li>
<li class="list-group-item">
  <a routerLink="/crud/continents/0">
    <i class="fas fa-globe me-2"></i>
    New Continent
  </a>
</li>
<li class="list-group-item">
  <a routerLink="/crud/countries/0">
    <i class="far fa-flag me-2"></i>
    New Country
  </a>
</li>
```

```
                    <li class="list-group-item">
                      <a routerLink="/crud/cities/0">
                        <i class="fas fa-city me-2"></i>
                        New City
                      </a>
                    </li>
                  </ul>
                </div>
              </div>
            </div>
          </div>
        </div>
      </div>
    </div>
    <div class="col-12 col-sm-12 col-md-12 col-lg-9 col-xl-9">
      <div class="row">
        <router-outlet></router-outlet>
      </div>
    </div>
  </div>
</div>
```

app.component.css

```
.nga-btn-shows {
  -webkit-box-shadow: 0 2px 5px 0 rgba(0, 0, 0, 0.16), 0 2px 10px 0 rgba(0, 0, 0, 0.12);
  box-shadow: 0 2px 5px 0 rgba(0, 0, 0, 0.16), 0 2px 10px 0 rgba(0, 0, 0, 0.12);
  padding: 0.5rem 0.5rem 0.5rem 0.5rem;
  font-size: 0.81rem;
```

```
  border: 0;
  -webkit-border-radius: 0.125rem;
  border-radius: 0.125rem;
}

.nga-btn-shows:hover,      .nga-btn-shows:active,      .nga-btn-shows:focus {
  -webkit-box-shadow: 0 5px 11px 0 rgba(0, 0, 0, 0.18), 0 4px 15px 0 rgba(0, 0, 0, 0.15);
  box-shadow: 0 5px 11px 0 rgba(0, 0, 0, 0.18), 0 4px 15px 0 rgba(0, 0, 0, 0.15);
  outline: 0;
}

.nga-btn-shows-outline-primary {
  border: 2px solid #4285f4 !important;
  color: #4285f4 !important;
  background-color: transparent !important;
}

.nga-card-movie-date a {
  color: gray;
  text-decoration: none;
  background-color: transparent;
  -webkit-text-decoration-skip: objects;
}

.nga-card-movie-date {
  padding: 0.05rem;
  position: relative;
  font-size: 9px;
```

```
  border: 0px solid #ddd;
  border-radius: 0.25rem;
  -webkit-transition: all 0.2s ease-in-out;
  -o-transition: all 0.2s ease-in-out;
  transition: all 0.2s ease-in-out;
  color: gray;
  text-align: right;
  overflow: hidden;
  text-overflow: ellipsis;
  white-space: nowrap;
  background-color: transparent;
}

.nga-card-movie-date a:hover {
  color: black;
  text-decoration: underline;
}

.nga-card-movie-text a {
  color: #3f729b;
  text-decoration: none;
  background-color: transparent;
  -webkit-text-decoration-skip: objects;
}

.nga-card-movie-text a:hover {
  color: #3f729b;
  text-decoration: underline;
}
```

```
.nga-card-movie-text {
  padding: 0.05rem;
  position: relative;
  background-color: transparent;
  border: 0px solid #ddd;
  border-radius: 0.25rem;
  -webkit-transition: all 0.2s ease-in-out;
  -o-transition: all 0.2s ease-in-out;
  transition: all 0.2s ease-in-out;
  color: #3f729b;
  text-align: center;
  overflow: hidden;
  text-overflow: ellipsis;
  white-space: nowrap;
}

.nga-card-movie-img {
  opacity: 1;
}

.nga-card-movie-img:hover {
  opacity: 0.9;
}

.nga-card-menu {
  position: relative;
  display: flex;
  flex-direction: column;
  min-width: 0;
  word-wrap: break-word;
```

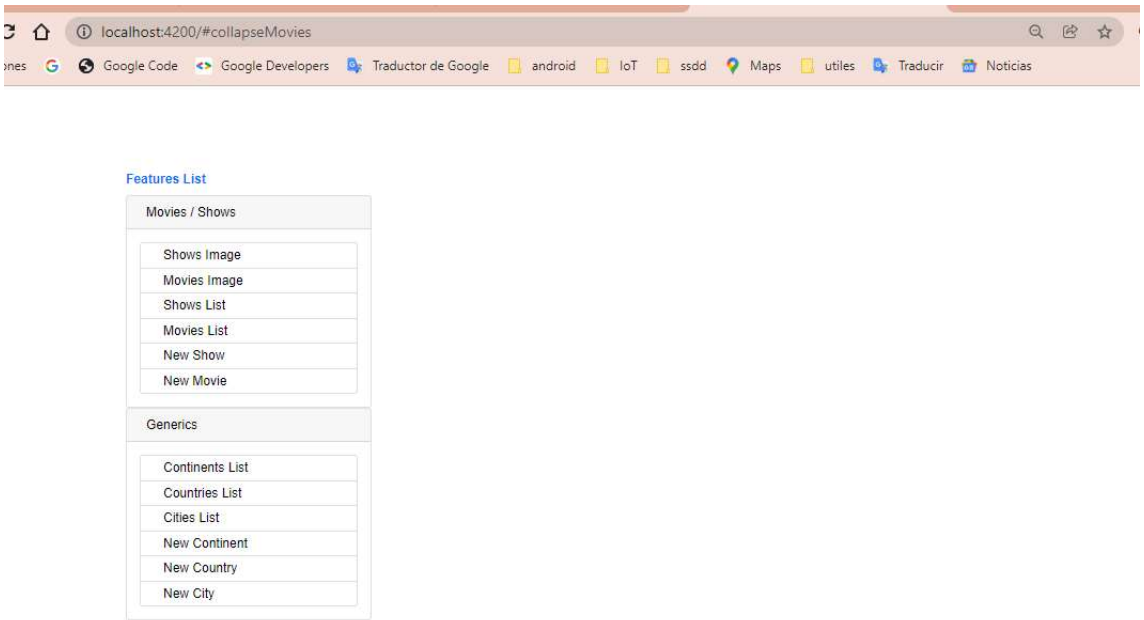
```
  background-color: #fff;
  background-clip: border-box;
  border: 1px solid rgba(0, 0, 0, 0.125);
  border-radius: 0.25rem;
  font-weight: 500;
}

.list-group-item {
  padding: 3px 10px
}

.list-group-item a {
  padding: 3px 10px;
  color: black;
  text-decoration: none;
}

.list-group-item a i {
  color: #0d6efd;
}

.list-group-item a:hover {
  text-decoration: none;
  color: #0d6efd;
}

.nga-card-menu .card-header a {
  text-decoration: none;
}
```

```
.nga-card-menu .card-header a .card-item {
  color: black;
}

.nga-card-menu .card-header a .card-item i {
  color: #0d6efd;
}

.nga-card-menu .card-header a:hover .card-item {
  text-decoration: none;
  color: #0d6efd;
}
```

una vez cargado este código inicial, se ejecuta el programa usando:

```
ng serve -o
```

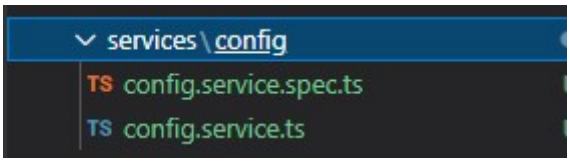
obteniéndose esta vista inicial:



Luego, se declararán los servicios que serán usados en este proyecto, para ello se creara la carpeta **services**:

1. Servicio **config**:

Primero, se creará la carpeta config, que alojara al servicio a crear



dentro de esta carpeta, se creará la clase config.ts

config.ts

```
export class Config {

  public api: boolean;
  public url: string;

  constructor() {
    this.api = false;
    this.url = '';
  }
}
```

y a continuación, el servicio **Config**:

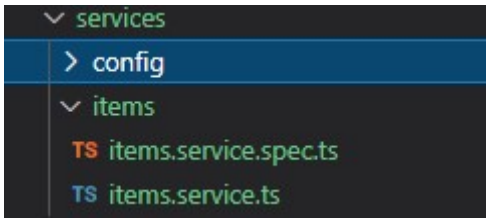
config.service.ts

```
import { Injectable } from '@angular/core';
import { Config } from './config';
import { environment } from 'src/environments/environment';

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

  public config: Config = new Config();
```

```
  constructor() {
    this.config.api = environment.config.api;
    this.config.url = environment.config.url;
  }
}
```

2. Servicio Items

se crea la carpeta ítems y se agrega el servicio indicado:



items.service.ts

```
import { Injectable } from '@angular/core';
import { HttpClient, HttpHeaders } from '@angular/common/http';
import { Observable, of } from 'rxjs';
import { catchError, map, tap } from 'rxjs/operators';
const httpOptions = {
  headers: new HttpHeaders({
    'Content-Type': 'application/json',
  })
};

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

  constructor(private http: HttpClient) { }
```

```
filterJsonItem(value: any, id: any): any {
  let dataTmp = null;
  value.map((row: any, index: any, data: any) => {
    const idTmp = parseInt(id, 10);
    if (data[index].id === idTmp) {
      dataTmp = data[index];
    }
  });
  return dataTmp;
}

filterJsonItemsCount(value: any, text: any): any {
  let resultCount = 0;
  if ((text !== undefined) && (text != null)) {
    const lcText = text.toString().toLowerCase();
    const result = value.filter(
      (e: { name: string; }) => (
        (e.name.toLowerCase().indexOf(lcText) === 0)
      )
    );
    resultCount = result.length;
  } else {
    resultCount = value.length;
  }
  return { count: resultCount };
}

filterJsonItems(value: any, text: any, itemsPerPage: number, page: number): any {
  let result: any;
```

```
  if ((text !== undefined) && (text != null)) {
    const lcText = text.toString().toLowerCase();
    result = value.filter(
      (e: { name: string; }) => (
        (e.name.toLowerCase().indexOf(lcText) === 0)
      )
    );
  } else {
    result = value;
  }
  const start = itemsPerPage * (page - 1);
  const end = itemsPerPage * (page - 1) + itemsPerPage - 1;
  const data: any[] = [];
  result.map((row: any, index: any) => {
    if ((index >= start) && (index <= end)) {
      data.push(result[index]);
    }
  });
  return data;
}

getItemsCount(api: boolean, url: any, query: any): Observable<any> {
  if (api) { url = url + '/count'; } else { url = url + '.json'; }
  let filter = '';
  if (query !== undefined) {
    if ((query !== '') && (query !== null)) { filter = '?q=' + query; }
  }
  const urlParameter = url + filter;
  let result: Observable<any>;
```

```
  if (api) {
    result = this.http.get<any>(urlParameter)
      .pipe(
        tap(heroes => this.log(`fetched items`)),
        catchError(this.handleError('getItems', []))
      );
  } else {
    result = this.http.get<any>(urlParameter)
      .pipe(
        map((value: string) => this.filterJsonItemsCount(value, query)),
        catchError(this.handleError('getItems', []))
      );
  }
  return result;
}


getItems(api: boolean, url: string, itemsPerPage: number, page: number, query: any):
Observable<any> {
  if (!api) { url = url + '.json'; }
  let filter = '';
  if ((itemsPerPage !== undefined) || (page !== undefined) || (query !== undefined)) {
    let limit: number;
    let offset: number;
    limit = itemsPerPage;
    offset = 0;
    if (page === 0) {
      page = 1;
    }
    if (page !== undefined) {
      offset = (page - 1) * itemsPerPage;
```

```
    }
    if (query !== undefined) {
      if ((query !== '') && (query !== null)) {
        filter = '?q=' + query;
      }
    }
    if (filter !== '') {
      filter = filter + '&limit=' + limit + '&offset=' + offset;
    } else {
      filter = '?limit=' + limit + '&offset=' + offset;
    }
  }
  const urlParameter = url + filter;
  let result: Observable<any>;
  if (api) {
    result = this.http.get<any[]>(urlParameter)
      .pipe(
        catchError(this.handleError('getItems', []))
      );
  } else {
    result = this.http.get<any>(urlParameter)
      .pipe(
        map((value: string) => this.filterJsonItems(value, query, itemsPerPage, page)),
        catchError(this.handleError('getItems', []))
      );
  }
  return result;
}
```

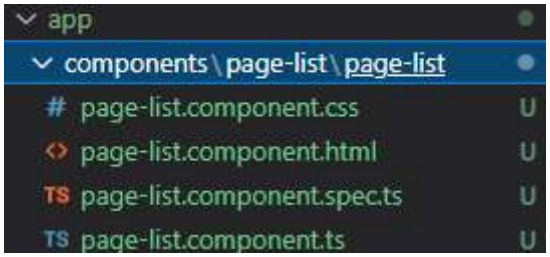
```
getItem(api: boolean, url: any, id: number): Observable<any> {
  if (!api) { url = url + '.json'; }
  let result: any = {};
  if (id !== undefined) {
    if (api) {
      const urlParameter = url + '/' + id;
      result = this.http.get<any>(urlParameter).pipe(
        tap(_ => this.log(`fetched item id=${id}`)),
        catchError(this.handleError<any>(`getItem id=${id}`))
      );
    } else {
      const urlParameter = url;
      result = this.http.get<any>(urlParameter).pipe(
        map((value: string) => this.filterJsonItem(value, id)),
        catchError(this.handleError('getItems', []))
      );
    }
  }
  return result;
}

addItem(url: any, item: any): Observable<any> {
  const body = JSON.stringify(item);
  return this.http.post<any>(url, body, httpOptions).pipe(
    tap((itemData: any) => this.log(`added item w/ id=${item.id}`)),
    catchError(this.handleError<any>('addItem'))
  );
}
```

```
  updateItem(body: object, id: number, link: any): Observable<any> {
    const url = link + '/' + id;
    return this.http.put(url, body, httpOptions).pipe(
      tap(_ => this.log(`updated item id=id`)),
      catchError(this.handleError<any>('updateItem'))
    );
  }

  deleteItem(link: any, id: number): Observable<any> {
    const url = link + '/' + id;
    return this.http.delete<any>(url, httpOptions).pipe(
      tap(_ => this.log(`deleted item id=${id}`)),
      catchError(this.handleError<any>('deleteItem'))
    );
  }

  private handleError<T>(operation = 'operation', result?: T): any {
    return (error: any): Observable<T> => {
      console.error(error);
      this.log(`${operation} failed: ${error.message}`);
      return of(result as T);
    };
  }

  private log(message: string): void {
  }
}
```

A continuación, se creara la carpeta **components**, en la que se colocaran Componentes usados en este aplicativo.

a. Componente page-list

Este componente será utilizado en el modulo inicial de la aplicación (App), crea la carpeta page-list para su creación.



inicialmente, se crea la clase Params.ts, donde se colocara el siguiente código:

```
export class Params {
    public q: string;
   public page: string;

   constructor() {
    this.q = '';
    this.page = '';
   }

 }
```

page-list.component.ts

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

import { Router, ActivatedRoute } from '@angular/router';

import { ItemsService } from './../../../services/items/items.service';
import { ConfigService } from './../../../services/config/config.service';
import { Params } from './Params';

@Component({
```

```
  selector: 'app-page-list',
  templateUrl: './page-list.component.html',
  styleUrls: ['./page-list.component.css']
})
export class PageListComponent implements OnInit {

  api: any;
  url: any;
  endpoint: any;
  items: any;
  icon: any;
  columns: any;
  link: any;
  filter = '';
  itemsPerPageDefault = 5;

  placeholder: any;
  results: any;
  creation: any;
  found: any;
  linkRoute: any;
  searchField = '';

  loaded: any;
  query: string;
  params = new Params();

  itemsCount = 0;
  itemsPage = 1;
```

```
itemsPerPage = 4;

public route: ActivatedRoute;
public router: Router;
public configService: ConfigService;
public itemsService: ItemsService;

constructor(injector: Injector) {
  this.query = '';
  this.route = injector.get(ActivatedRoute);
  this.router = injector.get(Router);
  this.configService = injector.get(ConfigService);
  this.itemsService = injector.get(ItemsService);

  this.initialize();

}

ngOnInit() {
}

initialize(): void {
  this.api = this.configService.config.api;
  this.url = this.configService.config.url + this.endpoint;
  this.readQueryParams();
}

readQueryParams(): void {
  this.route.queryParams
```

```
    .subscribe(params => {
      this.params.q = params['q'];
      if (params['page'] !== undefined) {
        this.params.page = params['page'];
        this.itemsPage = parseInt(this.params.page, 10);
      }
      this.searchField = this.params.q;
      this.getItems();
    });
}



getItems(): void {
  this.loaded = false;
  this.query = this.searchField;
  if (this.endpoint !== undefined) {
    this.itemsService.getItemsCount(this.api, this.url, this.query)
      .subscribe(item => {
        this.itemsCount = item.count;
        if (this.itemsPerPage < 1) {
          this.itemsPerPage = this.itemsPerPageDefault;
        }
        const page = this.itemsPage;
        const totalPages = Math.ceil(this.itemsCount / this.itemsPerPage);
        if (page >= totalPages) {
          this.itemsPage = totalPages;
        }
        this.itemsService.getItems(
          this.api, this.url, this.itemsPerPage, this.itemsPage, this.query)
```

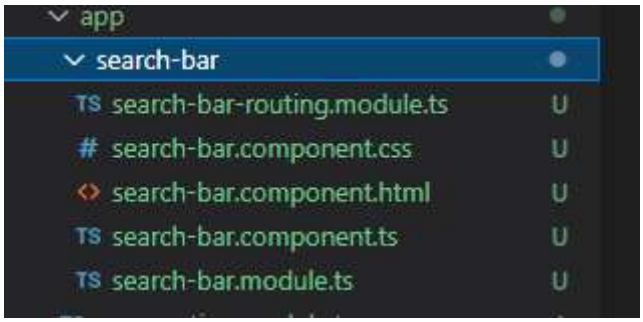
```
        .subscribe(items => {
          this.items = items;
          this.loaded = true;
        });
      });
  }
}

writeQueryParams(search?: boolean): void {
  let query = this.searchField;
  if ((query === '') || (query === undefined)) {
    query = '';
  }
  const url = '/' + this.linkRoute;
  let page = '';
  if (this.itemsPage > 1) {
    page = this.itemsPage.toString();
  }
  this.params.q = query;
  this.params.page = page;
  this.router.navigate(['crud/' + url], { queryParams: this.params });
}

search(): void {
  this.query = this.searchField;
  this.writeQueryParams();
  this.getItems();
}
```

```
  changePage(page: number): void {
    this.itemsPage = page;
    this.writeQueryParams();
    this.getItems();
  }

  selectItem(id: any): void {
    this.router.navigate(['/crud/' + this.link, id]);
  }

  onChangePage(page: any): void {
    this.changePage(page);
  }

  onSearch(query: any): void {
    this.searchField = query;
    this.search();
  }
}
```

A continuación, implementarán los módulos que serán usados en este ejercicio:

a. Modulo search-bar

```
ng generate module search-bar-routing --module search-bar –
-flat
```



search-bar.component.ts

```
import { Component, Input, OnInit } from '@angular/core';
import { Output, EventEmitter } from '@angular/core';
import { ElementRef, ViewChild } from '@angular/core';

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

  @ViewChild('edit', { static: false })
  edit!: ElementRef;

  @Input() searchField: any;
  @Input() placeholder: any;
  @Input() results: any;
  @Input() itemsCount: any;
  @Input() icon: any;

  @Output() search = new EventEmitter<string>();
```

```
  constructor() { }

  ngOnInit() {
  }

  searching(): void {
    this.search.emit(this.edit.nativeElement.value);
  }
}
```

search-bar.component.html

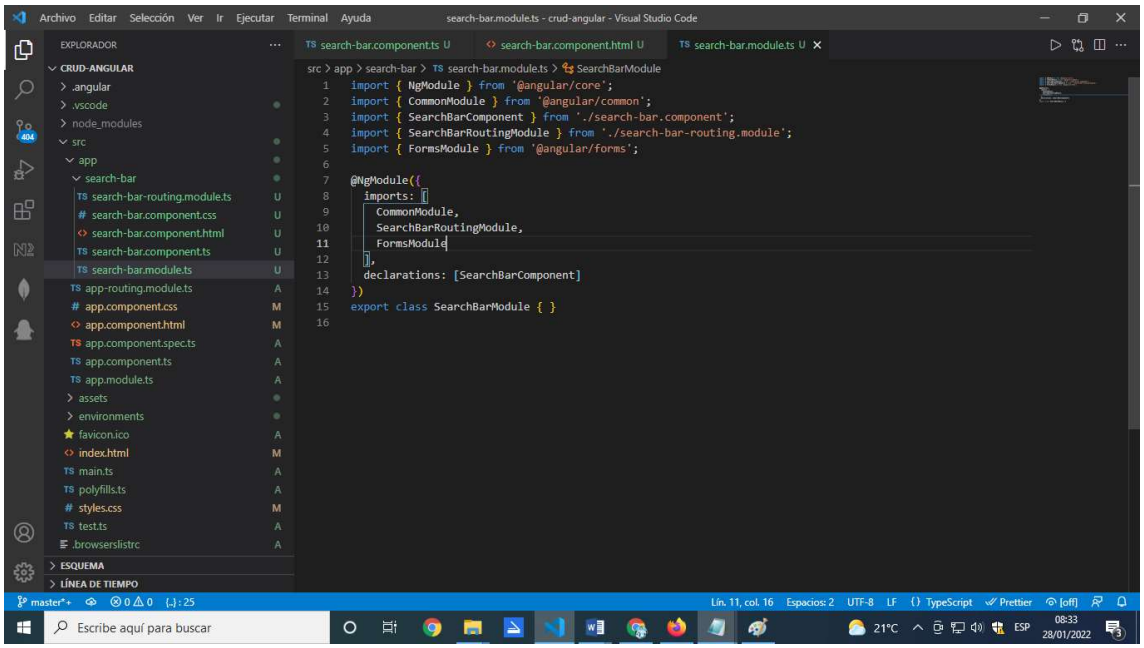
```
<div class="row">
  <div class="col">
    <span class="ml-4"><strong>Feature</strong></span>
    <div class="d-flex pt-2">
      <div class="d-flex me-auto">
        <input #edit type="text" class="form-control me-2" name="searchField" [ngModel]="searchField" placeholder="{{placeholder}}">
        <button type="button" class="btn btn-nga-primary btn-sm btn-nga" (click)="searching()"><i
            class="fa fa-search"></i></button>
      </div>
      <div class="d-flex me-auto d-none d-md-block">
        <div class=" pt-2 ml-4">
          <b>
            <span style="color:gray;">{{ itemsCount }} {{ results }}</span>
          </b>
          <span style="color: #e0e0e0;">( 0.432 ms )</span>
        </div>
```

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

search-bar.component.css

```
.btn-nga {
  -webkit-box-shadow: 0 2px 5px 0 rgba(0, 0, 0, 0.16), 0 2px 10px 0 rgba(0, 0, 0, 0.12);
  box-shadow: 0 2px 5px 0 rgba(0, 0, 0, 0.16), 0 2px 10px 0 rgba(0, 0, 0, 0.12);
  padding: 0.5rem 0.5rem 0.5rem 0.5rem;
  font-size: 0.81rem;
  border: 0;
  -webkit-border-radius: 0.125rem;
  border-radius: 0.125rem;
}

.btn-nga:hover, .btn-nga:active, .btn-nga:focus {
  -webkit-box-shadow: 0 5px 11px 0 rgba(0, 0, 0, 0.18), 0 4px 15px 0 rgba(0, 0, 0, 0.15);
  box-shadow: 0 5px 11px 0 rgba(0, 0, 0, 0.18), 0 4px 15px 0 rgba(0, 0, 0, 0.15);
  outline: 0;
}

.btn-nga-primary {
  border: 2px solid #4285f4 !important;
  color: #4285f4 !important;
  background-color: transparent !important;
}
```

En **search-bar.module.ts** se agrega *FormsModule* de la siguiente manera:

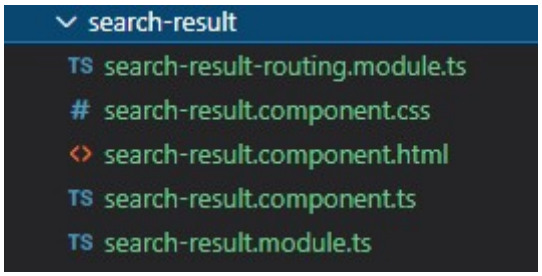


y en **search-bar.routing.module.ts** se agrega lo siguiente:

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

const routes: Routes = [];

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

b. Modulo search-result

```
ng generate module search-result-routing --module search-result –-flat
```



search-result.component.ts

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



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

  @Input() searchField: any;
  @Input() creation: any;
  @Input() found: any;
  @Input() link: any;

  constructor(public router: Router) { }

  createItem(): void {
    this.router.navigate(['/crud/' + this.link, 0]);
  }
```

```
  ngOnInit() {
  }

}
```

search-result.component.html

```
<div class="card mb-4 text-center">
  <div class="card-body">
    <img src="./assets/params/images/logo/dharma.png" height="70" width="70">
    <p class="card-text mt-3">No {{ found }} matches the specified search terms</p>
    <h6 class="card-title font-weight-bold text-primary">
      {{ searchField }}
    </h6>
    <blockquote class="blockquote">
      <p class="text-left text-primary">Suggestions</p>
      <h6 class="text-left">Try other keywords</h6>
      <h6 class="text-left">Delete search filters</h6>
      <hr>
      <h6 class="text-left text-info">Create new Item</h6>
      <button type="button" class="font-weight-bold btn btn-outline-info btn-sm mt-4" (click)="createItem()">
        <i class="fas fa-plus fa-lg mr-2"></i>Create {{ creation }}
      </button>
    </blockquote>
  </div>
</div>
```

search-results.component.css

```
.card {
  display: block;
  background-color: rgba(255, 255, 255, .8);
```

```
  box-shadow: 0 1px 3px rgba(0, 0, 0, .12), 0 1px 2px rgba(0, 0, 0, .24);
  border-radius: 2px;
  transition: all .2s ease-in-out;
  cursor: pointer;
}

.card:hover {
  box-shadow: 0 10px 20px rgba(0, 0, 0, .19), 0 6px 6px rgba(0, 0, 0, .23);
}

blockquote {
  border-left: 3px solid #4285F4;
  margin: 10px;
  padding: 10px 20px;
}
```

search-result.module.ts

```
import { NgModule } from '@angular/core';
import { CommonModule } from '@angular/common';
import { SearchResultComponent } from './search-result.component';
import { SearchResultRoutingModule } from './search-result-routing.module';
import { FormsModule } from '@angular/forms';

@NgModule({
  imports: [
    CommonModule,
    SearchResultRoutingModule,
    FormsModule
  ],
  declarations: [SearchResultComponent],
```

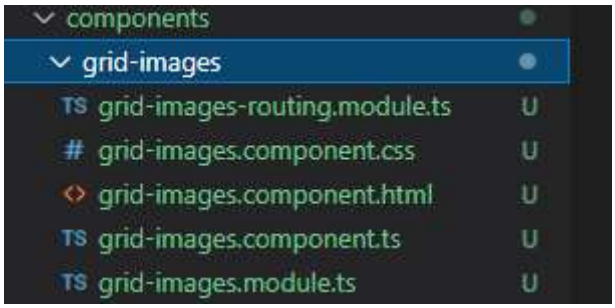
```
  exports: [
    SearchResultComponent
  ],
})
export class SearchResultModule { }
```

search-result.routing.module.ts

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

const routes: Routes = [];



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

c. Modulo grid-images

```
ng generate module grid-images-routing --module grid-images
–-flat
```



grid-images.components.ts

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



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

  @Input() items: any;
  @Input() columns: any;
  @Input() link: any;
  @Input() filter: any;
  @Input() itemsCount: any;
  @Input() pagination: any;

  constructor(public router: Router) { }

  selectItem(id: any): void {
    this.router.navigate(['/crud/' + this.link, id]);
  }
```

```
  ngOnInit() {
  }

}
```

grid-images.components.html

```
<div class="row">
  <div *ngFor="let record of items; let i=index" class="col-6 col-sm-6 col-md-4 col-lg-3 col-xl-2 px-2 mb-1">
    <div class="row">
      <div class="col-12">
        <div class="card-movie-date">
          <a class="waves-effect" style="cursor: pointer" (click)="selectItem(record['id'])">
            <b>08/12/1932</b>
          </a>
        </div>
      </div>
    </div>
    <div class="row">
      <div class="col">
        <div class="card-movie-img">
          <a (click)="selectItem(record['id'])">
            <img src="{{ record.image }}"
              style="cursor: pointer" class="img-fluid shadow-lg rounded" alt="recordname">
          </a>
        </div>
      </div>
    </div>
    <div class="row">
      <div class="col">
```

```
      <div class="card-movie-text">
        <a class="waves-effect" style="cursor: pointer" (click)="selectItem(record['id'])">
          <b style="font-size: 13px;">A Farewell to Arms</b>
        </a>
      </div>
    </div>
  </div>
</div>
</div>
```

grid-images.components.css

```
.card-movie-date a {
  color: gray;
  text-decoration: none;
  background-color: transparent;
  -webkit-text-decoration-skip: objects;
}

.card-movie-date {
  padding: 0.05rem;
  position: relative;
  font-size: 9px;
  border: 0px solid #ddd;
  border-radius: 0.25rem;
  -webkit-transition: all 0.2s ease-in-out;
  -o-transition: all 0.2s ease-in-out;
  transition: all 0.2s ease-in-out;
  color: gray;
```

```
  text-align: right;
  overflow: hidden;
  text-overflow: ellipsis;
  white-space: nowrap;
  background-color: transparent;
}

.card-movie-date a:hover {
  color: black;
  text-decoration: underline;
}

.card-movie-text a {
  color: #3f729b;
  text-decoration: none;
  background-color: transparent;
  -webkit-text-decoration-skip: objects;
}

.card-movie-text a:hover {
  color: #3f729b;
  text-decoration: underline;
}

.card-movie-text {
  padding: 0.05rem;
  position: relative;
  background-color: transparent;
  border: 0px solid #ddd;
```

```
  border-radius: 0.25rem;
  -webkit-transition: all 0.2s ease-in-out;
  -o-transition: all 0.2s ease-in-out;
  transition: all 0.2s ease-in-out;
  color: #3f729b;
  text-align: center;
  overflow: hidden;
  text-overflow: ellipsis;
  white-space: nowrap;
}

.card-movie-img {
  opacity: 1;
}

.card-movie-img:hover {
  opacity: 0.9;
}
```

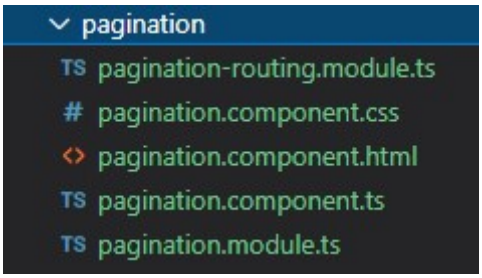
grid-images.module.ts

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

const routes: Routes = [];

@NgModule({
  declarations: [],
  imports: [
    CommonModule,
```

```
  RouterModule.forChild(routes)
 ],
 exports: [RouterModule]
})
export class GridImagesRoutingModule { }
```

d. Modulo Pagination

```
ng generate module pagination-routing --module pagination –-flat
```



Inicialmente, se creara la clase Pagination, que contendrá el siguiente código:

Pagination.ts

```
export class Pagination {
 public pages: any;
 public browser: any;
 public current: any;
 public total: any;
 public count: any;
 public page: any;
 public perPage: any;

 constructor(
 ) { }
}
```

a continuación, se creará el servicio pagination que tendrá los métodos usados por el componente

pagination.service.ts

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

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

constructor() { }

range(start: any, stop: any, step: any): any {
  if (stop == null) {
    stop = start || 0;
    start = 0;
  }
  if (!step) {
    step = stop < start ? -1 : 1;
  }
  const length = Math.max(Math.ceil((stop - start) / step), 0);
  const range = Array(length);
  for (let idx = 0; idx < length; idx++, start += step) {
    range[idx] = start;
  }
  return range;
}

getPagination(totalItems: number, currentPage: number = 1,
perPage: number): any {
```

```
const totalPages = Math.ceil(totalItems / perPage);
let startPage: number;
let endPage: number;
const visiblePages = 7;

if (currentPage < 1) {
  currentPage = 1;
}
if (currentPage > totalPages) {
  currentPage = 1;
}
if (totalPages <= visiblePages) {
  startPage = 1;
  endPage = totalPages;
} else {
  if (currentPage <= visiblePages) {
    startPage = 1;
    endPage = visiblePages;
  } else {
    if (currentPage + 1 >= totalPages) {
      startPage = totalPages - visiblePages + 1;
      endPage = totalPages;
      if (currentPage < startPage) {
        startPage = currentPage + 1;
        endPage = currentPage + 1;
      }
    } else {
      endPage = currentPage;
      startPage = endPage - visiblePages + 1;
    }
```

```
    }
  }

  const pages = this.range(startPage, endPage + 1, 1);
  let browser = false;
  if (totalPages > visiblePages) {
    browser = true;
  }

  const pagination = new Pagination();
  pagination.pages = pages;
  pagination.browser = browser;
  pagination.total = totalPages;
  pagination.current = currentPage;
  pagination.perPage = perPage;

  return pagination;

}

}
```

a continuación, se construirá el componente **pagination.**

pagination.component.ts

```
import { Component, OnChanges, OnInit, SimpleChanges } from '@angular/core';
import { Input, Output, EventEmitter } from '@angular/core';
import { Pagination } from './pagination';
import { PaginationService } from './pagination.service';
```

```
@Component({
  selector: 'app-pagination',
  templateUrl: './pagination.component.html',
  styleUrls: ['./pagination.component.css'],
})
export class PaginationComponent implements OnInit, OnChanges {
  @Input() count: any;
  @Input() page: any;
  @Input() perPage: any;

  @Output() changePage = new EventEmitter<number>();

  pagination = new Pagination();

  constructor(private paginationService: PaginationService) {}

  ngOnChanges(changes: SimpleChanges): void {
    this.pagination.count = this.count;
    this.pagination.perPage = this.perPage;
    this.pagination.page = this.page;
    this.pagination = this.paginationService.getPagination(
      this.pagination.count,
      this.pagination.page,
      this.pagination.perPage
    );
  }

  ngOnInit() {}

  selectPage(page: number): void {
```

```
    this.changePage.emit(page);
  }
}
```

pagination.component.html

```
<nav aria-label="pagination" *ngIf="pagination.pages && pagination.pages.length">
  <ul class="pagination d-flex justify-content-center pagination-sm">
    <li *ngIf="pagination.browser" class="page-item" [ngClass]="{disabled:pagination.current == 1}">
      <a class="page-link waves-effect waves-effect" (click)="selectPage(1)">&laquo;</a>
    </li>
    <li *ngIf="pagination.browser" class="page-item" [ngClass]="{disabled:pagination.current == 1}">
      <a class="page-link" (click)="selectPage(pagination.current - 1)">&lsaquo;</a>
    </li>
    <li class="page-item" *ngFor="let page of pagination.pages" [ngClass]="{active:pagination.current == page}">
      <a class="page-link" (click)="selectPage(page)">{{page}}</a>
    </li>
    <li *ngIf="pagination.browser" class="page-item" [ngClass]="{disabled:pagination.current == pagination.total}">
      <a class="page-link" (click)="selectPage(pagination.current + 1)">&rsaquo;</a>
    </li>
    <li *ngIf="pagination.browser" class="page-item" [ngClass]="{disabled:pagination.current == pagination.total}">
      <a class="page-link" (click)="selectPage(pagination.total)">&raquo;</a>
    </li>
  </ul>
```

```
</nav>
```

pagination.module.ts

```
import { NgModule } from '@angular/core';
import { CommonModule } from '@angular/common';
import { PaginationComponent } from './pagination.component';
import { PaginationRoutingModule } from './pagination-routing.module';

@NgModule({
  imports: [
    CommonModule,
    PaginationRoutingModule
  ],
  declarations: [PaginationComponent],
  exports: [
    PaginationComponent,
  ],
})
export class PaginationModule { }
```

y finalmente el archivo routing:

pagination.routing.module.ts

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

const routes: Routes = [];

@NgModule({
  declarations: [],
```

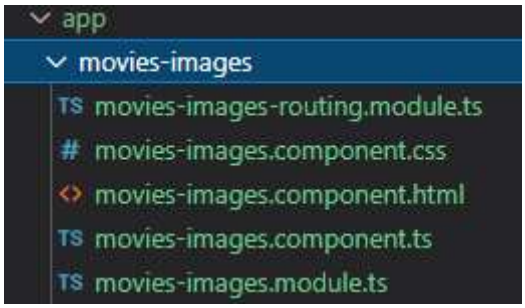
```
  imports: [
    CommonModule,
    RouterModule.forChild(routes)
  ],
  exports: [RouterModule]
})
export class PaginationRoutingModule { }
```

el modulo Pagination, finalmente quedara de esta manera:

Una vez que todos estos modulos han sido creados, toca el turno del modulo ***movie-images***, quien reunirá a todos los modulos anteriores.

e. Modulo movies-images

```
ng generate module movies-images-routing --module movies-images –-flat
```



movies-images.component.ts

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


import { Meta, Title } from '@angular/platform-browser';
```

```
import { PageListComponent } from '../components/page-list/page-list/page-list.component';


@Component({
  selector: 'app-movies-images',
  templateUrl: './movies-images.component.html',
  styleUrls: ['./movies-images.component.css']
})
export class MoviesImagesComponent extends PageListComponent implements OnInit {


  constructor(
    private meta: Meta,
    private titleService: Title,
    injector: Injector) {


    super(injector);


  }


  override ngOnInit() {
    this.titleService.setTitle('New Movies: Instructor TIC');
    this.meta.addTag({
      name: 'angular.tic',
      content: 'jorge guerra'
    });
    this.meta.updateTag(
      {
        name: 'description',
        content: 'Todas las nuevas peliculas'
      });
```

```
  }

  override selectItem(id: any): void {
    this.router.navigate(['/crud/' + this.link, id]);
  }

  override initialize(): void {

    this.endpoint = 'movies';
    this.link = 'movies';
    this.placeholder = 'movies...';
    this.results = 'Movies';
    this.found = 'movies';
    this.creation = 'Movie';
    this.loaded = false;
    this.icon = 'fas fa-film';
    this.itemsCount = 0;
    this.itemsPerPage = 24;
    this.linkRoute = 'movies-images';

    this.columns = [
      { name: 'Id', field: 'id', align: 'left', color: 'black',
font: '' },
      { name: 'Name', field: 'name', align: 'left', color: 'text-
primary', font: 'bold' },
      { name: 'Date', field: 'releaseDate', align: 'center',
color: 'text-primary', font: '' },
    ];

    super.initialize();
  }
```

```
}
```

movies-image.component.html

```
<app-search-bar [searchField]="searchField" [itemsCount]="itemsCount" [icon]="icon" [results]="results"
  [placeholder]="placeholder" (search)="onSearch($event)"></app-search-bar>


<div class="row" *ngIf="loaded && itemsCount==0">
  <div class="col">
    <app-search-result [searchField]="searchField" [found]="found" [creation]="creation" [link]="link">
    </app-search-result>
  </div>
</div>
<div class="row" *ngIf="loaded">
  <div class="col mt-4">


    <app-pagination [count]="itemsCount" [page]="itemsPage" [perPage]="itemsPerPage"
      (changePage)="onChangePage($event)">
    </app-pagination>


    <app-grid-images [items]="items" [columns]="columns" [link]="link"></app-grid-images>


  </div>
</div>
```

se observa que <roter-outlet> esta con error, lo que se arreglara mas adelante.

movies-image.module.ts

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

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

import { MoviesImagesComponent } from ./movies-images.component';

import { MoviesImagesRoutingModule } from ./movies-images-routing.module';

import { SearchBarModule } from ../search-bar/search-bar.module';

import { SearchResultModule } from ../search-result/search-result.module';

import { PaginationModule } from ../components/pagination/pagination.module';

import { GridImagesModule } from ../components/grid-images/grid-images.module';

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


@NgModule({
  imports: [
    CommonModule,
    MoviesImagesRoutingModule,
    SearchBarModule,
    SearchResultModule,
    PaginationModule,
    GridImagesModule,
     RouterModule
  ],
  declarations: [MoviesImagesComponent],
  exports: [
    MoviesImagesComponent
  ],
})
export class MoviesImagesModule { }
```

Finalmente, se cargan los datos en routing:

movies-images.routing.module.ts

```
import { NgModule } from '@angular/core';
import { CommonModule } from '@angular/common';
import { Routes, RouterModule } from '@angular/router';
import { MoviesImagesComponent } from ./movies-images.component';
const routes: Routes = [
  { path: '', component: MoviesImagesComponent },
];

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

De esta forma, el modulo MoviesImagesModule ya esta completo. Solo falta ruteralo con App, y establecer las rutas de los otros modulos que serán invocados.

Luego, en app.module.ts se registra este modulo:

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

```
import { ConfigService } from './services/config/config.service';

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

y luego se implementa el routing de los modulos

app.routing.module.ts