

# *Apuntes Angular*

## *Comando Angular*

Primer Paso: Crear Modulo

```
ng generate module pages/theaters --routing
```

## ***CREAR COMPONENTES VERDES***

**ng serve → Correr el proyecto**

```
ng generate component [Ruta] pages/theaters/list
```

## **Clase 31/oct**

Principios SOLID (Responsabilidad Unica)

--Routing (Crea un archivo para hacer una redirección de los archivos)

Componente = La parte visual de la pagina

[Fundamento de POO] Casteo → Tipo de dato convertirlo a otro

## **Pasos CRUD**

### **1 Crear Modulo**

```
ng generate m ruta/theaters --routing
```

### **2 Crear Componente**

```
ng generate c ruta/theaters/list
```

### **3 Crear Modelo**

```
ng generate class ruta/models/theater --type=model
```

### **4 Crear Servicio**

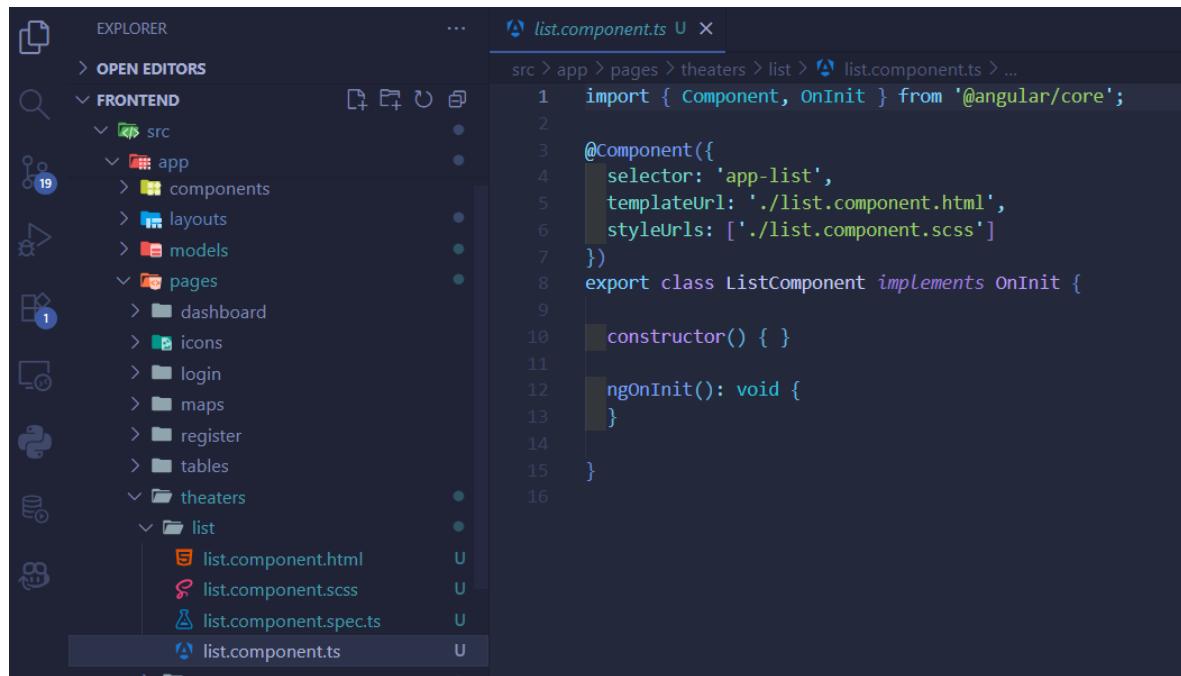
```
ng generate service ruta/theaters
```

## 5 Programar en el Contador

1. ng generate class models/Theater --type=model
2. ng generate service services/theater
3. ng generate class models/seat --type=model
4. Instalar Libreria → npm i sweetalert2 –force (**Librería de alertas**)

Framework → Patrón de diseño → Modelo / Vista controlador

Módulos – Carpetas con los componentes



```
src > app > pages > theaters > list > list.component.ts > ...
1 import { Component, OnInit } from '@angular/core';
2
3 @Component({
4   selector: 'app-list',
5   templateUrl: './list.component.html',
6   styleUrls: ['./list.component.scss']
7 })
8 export classListComponent implements OnInit {
9
10   constructor() { }
11
12   ngOnInit(): void {
13   }
14
15 }
16
```

Acá se aplica la lógica de la programación

Clase 05/11-/2025

1. Creacion modulo ng generate modulo /ruta/nombre --routing
2. Crear la componente: ng g c ruta/nombreModulo/nombreComponente
3. Configurar rutas (xxx.routing)

General /Layout admin)

Especifico (Componente)

#### 4. CRUD LISTAR-ELIMINAR

- 4.1 Crear modelo (Manual o comando)
- 4.2 Crear Servicio (ng g service services/nombre)
- 4.3 Programar contador (TS)
- 4.4 Programar HTML (Directiva Ng For)

#### 5. CRUD CREAR-ACTUALIZAR

5.1 Configurar librerías de formulas

Reactivos xxx.module.ts

**Importar:** FormsModule, (**Manipular datos desde vista a controlador**)

ReactiveFormsModule (**Tema de validaciones**)

(**Nos ayuda a poner a funcionar el controlador bien desde la vista**)

**CADA VEZ QUE SE TOQUE ESTE ARCHIVO, PARAR LA EJECUCION Y CORRER EL PROYECTO DE NUEVO.**

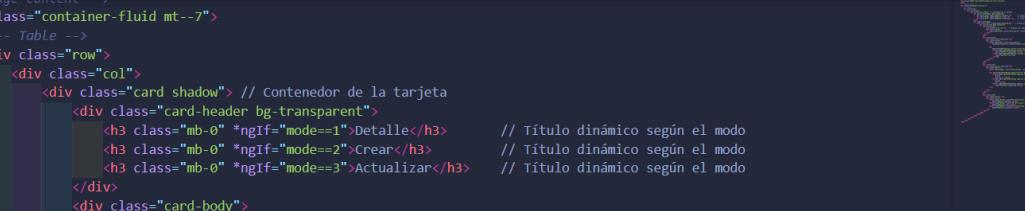
5.2 Programar “manage.component.ts”

- 5.2.1 A Observar Ruta (**ActivatedRoute**) Inyectar dependencia
- 5.2.2 B FormBuilder → Objeto que ayuda a definir reglas-validaciones en el constructor (**A-B Inyectar**)
- 5.2.3 C Crear variable de clase (**Form Group**) → Hace cumplir las reglas
- 5.2.4 D Crear las reglas “**ConfigFormGroup**”
- 5.2.5 E Validar comportamiento componente (Analizar la ruta: crear?, actualizar?) → Con **ActivatedRoute**
- 5.2.6 F Programar método de Crear/Actualizar (Pendiente)

5.3 Programar manage.component.html (VISTA)

Directivo NgIf → Condicionales dentro html (Vista/controlador) e inicializar el formGroup

- 5.3.1 A Crear una etiqueta de tipo “form” e inicializar el formGroup
- 5.3.2 B La caja de texto correspondiente asociar un formControlName
- 5.3.3 C Programar la visualización de alertas de error manage.component.html (Linea 31) Llamando al método “getFormGroup”



```
src > app > pages > theaters > manage > manage.component.html
  4 <div class="container-fluid mt--7">
  5   <!-- Table -->
  6   <div class="row">
  7     <div class="col">
  8       <div class="card shadow"> // Contenedor de la tarjeta
  9         <div class="card-header bg-transparent">
 10           <h3 class="mb-0" *ngIf="mode==1">Detalle</h3>      // Título dinámico según el modo
 11           <h3 class="mb-0" *ngIf="mode==2">Crear</h3>      // Título dinámico según el modo
 12           <h3 class="mb-0" *ngIf="mode==3">Actualizar</h3> // Título dinámico según el modo
 13         </div>
 14         <div class="card-body">
 15           <form class="form" [formGroup]="theFormGroup"> // Formulario reactivo vinculado al FormGroup Ctrl+I to continue
 16             <div class="card-body container">
 17               <div class="row">
 18                 <div class="col-3">ID</div> // Etiqueta del campo ID
 19                 <div class="col-9">
 20                   <input type="text" class="form-control" formControlName="id" disabled> // Campo ID siempre desabilitado
 21                 </div>
 22               </div>
 23               <br>
 24               <div class="row">
 25                 <div class="col-3">Ubicación</div>
 26                 <div class="col-9">
 27                   <input type="text" class="form-control"
 28                     [disabled]="" mode==1" formControlName="location">
```

Manage.component.html → Contiene zonas de operaciones, Cajas de texto donde se pone cuidado (**ETIQUETA FORM**) (**Possible sustentación**)

<https://blog.angular-university.io/angular-customValidators/>

(Funciones de validaciones Angular, Documentación Oficial)

Theaters-routing-module.ts → Se manejan las rutas y pueden cambiar variables

**Sustentación** (Coménteme la menor cantidad de líneas, para que deje de funcionar x función **manage.components.ts**)

### **Possible sustentación:**

```
configFormGroup() { // Definir las reglas del formulario
    this.theFormGroup = this.theFormBuilder.group([
        // primer elemento del vector, valor por defecto
        //Segundo elemento del vector, validaciones que se aplicarán a ese
        //Tercer elemento del vector, si hay más de una validación en la lista, serán las reglas
        id: [0, []],
        capacity: [0, [Validators.required, Validators.min(1), Validators.max(100)]], // Capacidad requerida entre 1 y 100
        location: ['', [Validators.required, Validators.minLength(2)]] // Ubicación requerida con longitud mínima de 2 caracteres
    ])
}
```

Que cumpla con alguna condición para una contraseña manage.component.ts (Linea 50)

```

45
46 <div *ngIf="getTheFormGroup.capacity.errors && (getTheFormGroup.capacity.dirty || getTheFormGroup.capacity.touched || trySend)">
47   <strong *ngIf="getTheFormGroup.capacity.errors.min" class="msnError">Valor
48     mínimo
49     debe ser 1</strong>
50   <strong *ngIf="getTheFormGroup.capacity.errors.max" class="msnError">Valor
51     máximo
52     100 </strong>
53   <strong *ngIf="getTheFormGroup.capacity.errors.required"
54     class="msnError">Requerido</strong>
55 </div>

```

```

theaters-routing.module.ts U X
src > app > pages > theaters > theaters-routing.module.ts > ...
1 import { NgModule } from '@angular/core';
2 import { RouterModule, Routes } from '@angular/router';
3 import { ManageComponent } from './manage/manage.component';
4 import { ListComponent } from './list/list.component';
5
6 const routes: Routes = [
7
8 {
9   path:"list",
10  component:ListComponent
11 },
12 {
13   path:"create",
14  component:ManageComponent
15 },
16 {
17   path:"update/:id",
18  component:ManageComponent,
19 }
20 ];
21
22 @NgModule({
23   imports: [RouterModule.forChild(routes)],
24   exports: [RouterModule]
25 })
26 export class TheatersRoutingModule { }

```

Theaters-Routing-modulo.ts (Rutas)

Theaters-module.ts (Rutas e importaciones de librerías)

## Clase 12/11/2025 miércoles

1. Usuario ingresa email y password (html)
2. Controlador que monitorea login.html
3. Llamado al servicio de seguridad (Llamar al backend0.)
4. Backend responde (Info usuario, **token**)
5. Almacenar token: **localStorage – Cookie** (**Hasta aquí proceso de autentificación**)
6. Utilizar framework de Angular → Configuración de la variable global reactiva (**Patrón diseño Observer**) (**Aquí pasamos información de un lado a otro**)
7. Consumo de la variable global

## Crear servicio

Ng generate service services/security

Pages → login

Services → security Service

Models → User.ts

Configuramos Mock-Server para crear una respuesta

Mockserver → + → créate → new collection

Post → login → responsive body (codigo)

Enviroments → url\_security: “Postman url”

Security service → Linea 26 → url\_security

Login.html → Linea 46 → input email

```
36 <div class="card-body px-lg-5 py-lg-5">
37   <form role="form">
38     <div class="form-group mb-3">
39       <div class="input-group input-group-alternative">
40         <div class="input-group-prepend">
41           <span class="input-group-text"><i class="ni ni-email-83"></i></span>
42         </div>
43         <input class="form-control" placeholder="Email" type="email" [(ngModel)]="user.email">
44       </div>
45     </div>
46     <div class="form-group">
47       <div class="input-group input-group-alternative">
48         <div class="input-group-prepend">
49           <span class="input-group-text"><i class="ni ni-lock-circle-open"></i></span>
50         </div>
51         <input class="form-control" placeholder="Password" type="password" [(ngModel)]="user.password">
52       </div>
53     </div>
54   </form>
```

Enlazado con User.ts → Lines 46 y línea 54

Login.component.ts → Línea 23 → Llama un Endpoint tipo POST

Login.component.ts → Línea 24-25-26 → Verificación de Logeo del usuario

```

src > app > services > security.service.ts > SecurityService > login
9  @Injectable({
10    providedIn: "root",
11  })
12  export class SecurityService { // Servicio para manejar la seguridad y autenticación de usuarios y sesiones y almacenamiento en local storage
13    theUser = new BehaviorSubject<User>(new User());
14    constructor(private http: HttpClient) {
15      this.verifyActualSession();
16    }
17
18    /**
19     * Realiza la petición al backend con el correo y la contraseña
20     * para verificar si existe o no en la plataforma
21     * @param infoUsuario JSON con la información de correo y contraseña
22     * @returns Respuesta HTTP la cual indica si el usuario tiene permiso de acceso
23     */
24    login(user: User): Observable<any> { // Método para iniciar sesión del usuario y enviar sus credenciales al backend
25      return this.http.post<any>(`${environment.url_security}/login`, user); // Realiza una petición POST al endpoint de login del backend con la info
26    }

```

Método que Llama por medio de un HTTP por medio del Post del MockServer

Refactoring.guru. (Pagina para aprender patrones de diseño Observer)

```

src > app > components > navbar > navbar.component.ts > NavbarComponent > constructor > subscribe() callback
12
13  @Component({
14    selector: "app-navbar",
15    templateUrl: "./navbar.component.html",
16    styleUrls: ["./navbar.component.scss"],
17  })
18  export class NavbarComponent implements OnInit {
19    public focus;
20    public listTitles: any[];
21    public location: Location;
22    user: User; // Usuario Logueado
23    subscription: Subscription;
24    constructor(
25      location: Location,
26      private element: ElementRef,
27      private router: Router,
28      private securityService: SecurityService
29    ) {
30      this.location = location;
31      this.subscription = this.securityService.getUser().subscribe((data) => [ //Estar pendiente de la variable global reactiva (subscription --> simulacion de conectado a
32        this.user = data; //Pendiente de que haya un cambio
33      ]);
34    }

```

```

src > app > components > navbar > navbar.component.html > li.nav-item > a.nav-link.pr-0
1
2  <nav class="navbar navbar-expand-md navbar-dark">
3    <div class="container-fluid">
4      <ul class="navbar-nav align-items-center d-none d-md-flex">
5        <li class="nav-item d-none d-md-flex">
6          <a href="#" class="nav-link pr-0">Inicio</a>
7        </li>
8        <li class="nav-item d-none d-md-flex">
9          <a href="#" class="nav-link pr-0">Nosotros</a>
10         </li>
11         <li class="nav-item d-none d-md-flex">
12           <a href="#" class="nav-link pr-0">Contacto</a>
13         </li>
14       </ul>
15       <form class="navbar-search navbar-search-dark form-inline mr-3 d-none d-md-flex ml-lg-auto">
16         <input type="text" class="form-control" placeholder="Search..."/>
17       </form>
18     </div>
19   </nav>
20
21   <!-- User -->
22   <ul class="nav navbar-nav align-items-center d-none d-md-flex">
23     <li class="nav-item ngbDropdown placement="bottom-right">
24       <a href="#" class="nav-link pr-0" role="button" ngbDropdownToggle>
25         
26       </a>
27       <div class="media align-items-center">
28         <span class="avatar avatar-sm rounded-circle">
29           
30         </span>
31         <div class="media-body ml-2 d-none d-lg-block">
32           <span class="mb-0 text-sm font-weight-bold">{{user?user.name:"Invitado"}}</span>
33         </div>
34       </div>
35     </li>
36   </ul>
37
38   <!-- Side Nav -->
39   <div class="side-nav">
40     <ul class="list-group list-group-flush">
41       <li class="list-group-item"><a href="#">Home</a></li>
42       <li class="list-group-item"><a href="#">Nosotros</a></li>
43       <li class="list-group-item"><a href="#">Contacto</a></li>
44     </ul>
45   </div>
46
47   <!-- Footer -->
48   <div class="footer">
49     <p>Footer Content</p>
50   </div>
51
52   <!-- Copyright -->
53   <div class="copyright">
54     <p>Copyright © 2020. All Rights Reserved</p>
55   </div>
56
57   <!-- Scripts -->
58   <script src="https://code.jquery.com/jquery-3.5.1.slim.min.js" integrity="sha384-  
...>

```

Cambio Usuario Jessica Jones

```

    /**
 * logout() [
 *   localStorage.removeItem("sesion"); // Elimina la información de la sesión del local storage
 *   this.setUser(new User()); //Actualiza por un nuevo usuario vacío
 * ]

```

Hacer Logout → Verificar si es new User o solo New en el archivo → security.service  
Linea 72

## Clase 14 Nov Viernes

### Guardianes e Interceptores

Crear → ng g guard guards/Authentication

ng g guard guards/NoAuthentication

1. Crear el archive por medio del comando
2. Programar el guardian
3. Activar el guardian (Que todo lo que vaya para el backend tiene que pasar por los interceptores)
  - Si el usuario ya tiene una sesión logeada, no puede abrir cosas de personas que aún no inician sesión (Por ejemplo el login)

```

authentication.guard.ts U security.service.ts M no-authentication.guard.ts U ...
s C:\Desktop\frontend\src\app\guards\authentication.guard.ts • Untracked  ationGuard > canActivate
12 @Injectable({
13   providedIn: "root",
14 })
15 export class NoAuthenticationGuard implements CanActivate {
16   constructor(
17     private securityService: SecurityService,
18     private router: Router
19   ) {}
20   canActivate(
21     route: ActivatedRouteSnapshot,
22     state: RouterStateSnapshot
23   ):
24     | Observable<boolean | UrlTree>
25     | Promise<boolean | UrlTree>
26     | boolean
27     | UrlTree {
28   if (!this.securityService.existSession()) { // Verifica si no existe una sesión activa
29     return true; // Permite el acceso a la ruta si no hay sesión activa
30   } else {
31     this.router.navigate(["/dashboard"]); // Redirige al dashboard si hay una sesión activa
32     return false; // No permite el acceso a la ruta si hay sesión activa
33   }
34 }
35 }
36

```

Se verifica si el usuario ya tiene una sesión activa, no debe dejar crear usuario o volver al login si ya tiene sesión activa.

```

File Edit Selection View Go Run Terminal Help < > Q: frontend
src > app > layouts > admin-layout > admin-layout.routing.ts > AdminLayoutRoutes > children > canActivate
1 import { Routes } from '@angular/router';
2
3 import { DashboardComponent } from '../../../../../pages/dashboard/dashboard.component';
4 import { IconsComponent } from '../../../../../pages/icons/icons.component';
5 import { MapsComponent } from '../../../../../pages/maps/maps.component';
6 import { UserProfileComponent } from '../../../../../pages/user-profile/user-profile.component';
7 import { TablesComponent } from '../../../../../pages/tables/tables.component';
8 import { AuthenticationGuard } from 'src/app/guards/authentication.guard';
9
10 export const AdminLayoutroutes: Routes = [
11   { path: "dashboard", component: DashboardComponent },
12   { path: "user-profile", component: UserProfileComponent },
13   { path: "tables", component: TablesComponent },
14   { path: "icons", component: IconsComponent },
15   { path: "maps", component: MapsComponent },
16   {
17     path: "",
18     children: [
19       {
20         path: "theaters",
21         canActivate: [AuthenticationGuard],
22         loadChildren: () =>
23           import("src/app/pages/theaters/theaters.module").then(
24             (m) => m.TheatersModule
25           ),
26       },
27     ],
28   },
29 ];
30

```

Deja ver las páginas si está autenticado

Crear interceptor de autenticación

ng g interceptor interceptors/Auth

Creamos nueva ventana en visual

Websocketserver

Main.py → Código compartido

Luego instalar:

python -m venv venv

venv\Scripts\activate

pip install --upgrade pip

pip install flask flask-socketio eventlet

python main.py

**En la carpeta del proyecto hacer lo siguiente:**

npm i ngx-socket-io@4.3.0 –force

ng g s services/WebSocketService → Pegar código compartido