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

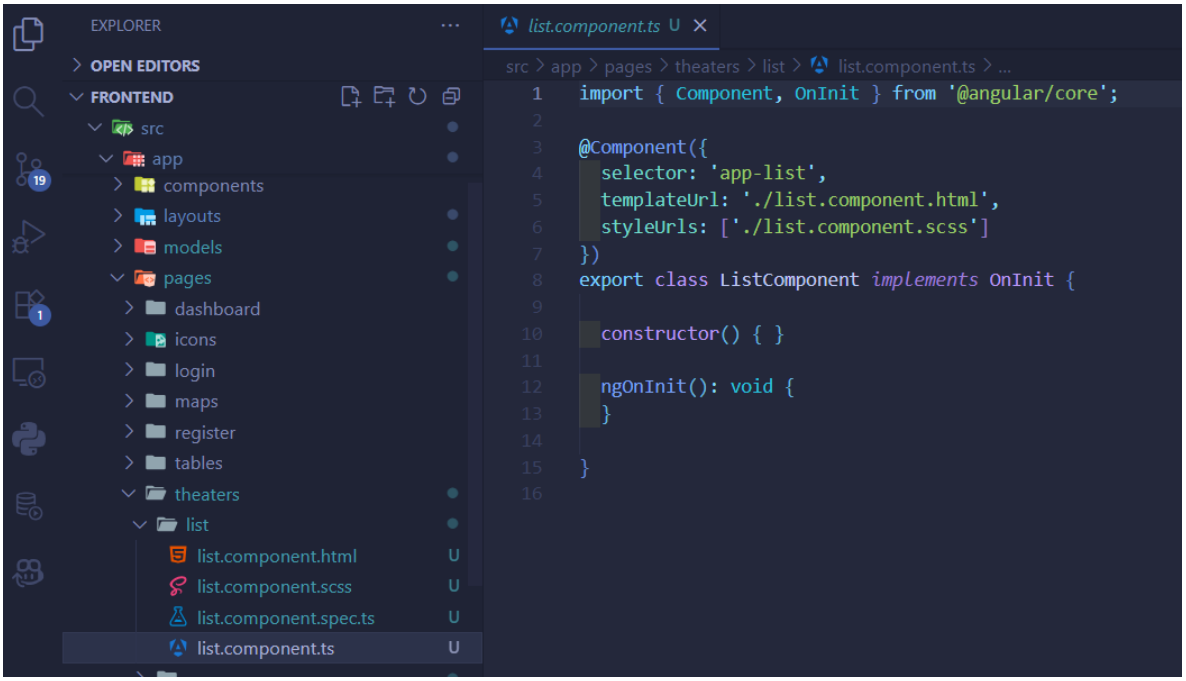
ng generate service ruta/theaters

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



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)

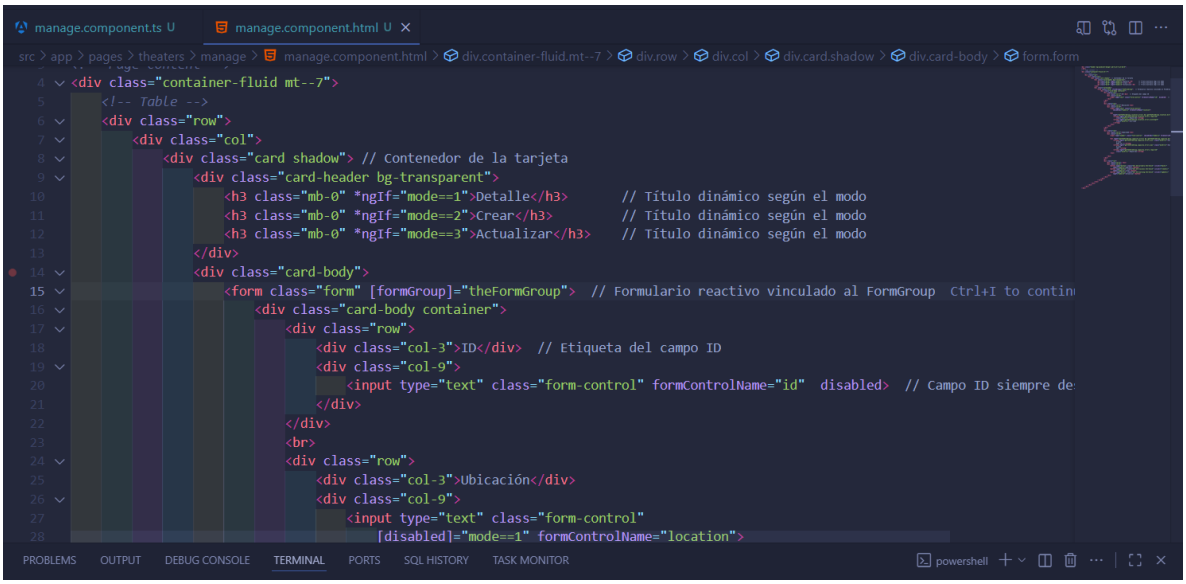
4. CRUD LISTAR-ELIMINAR
   4.1 Crear modelo (Manual o comando)
   4.2 Crear Servicio (ng g service services/nombre)
   4.3 Programar contador (TS)
   4.4 Programar HTML (Directiva Ng For)

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"



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

https://blog.angular-university.io/angular-custom-validators/

(Funciones de validaciones Angular, Documentación Oficial)

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

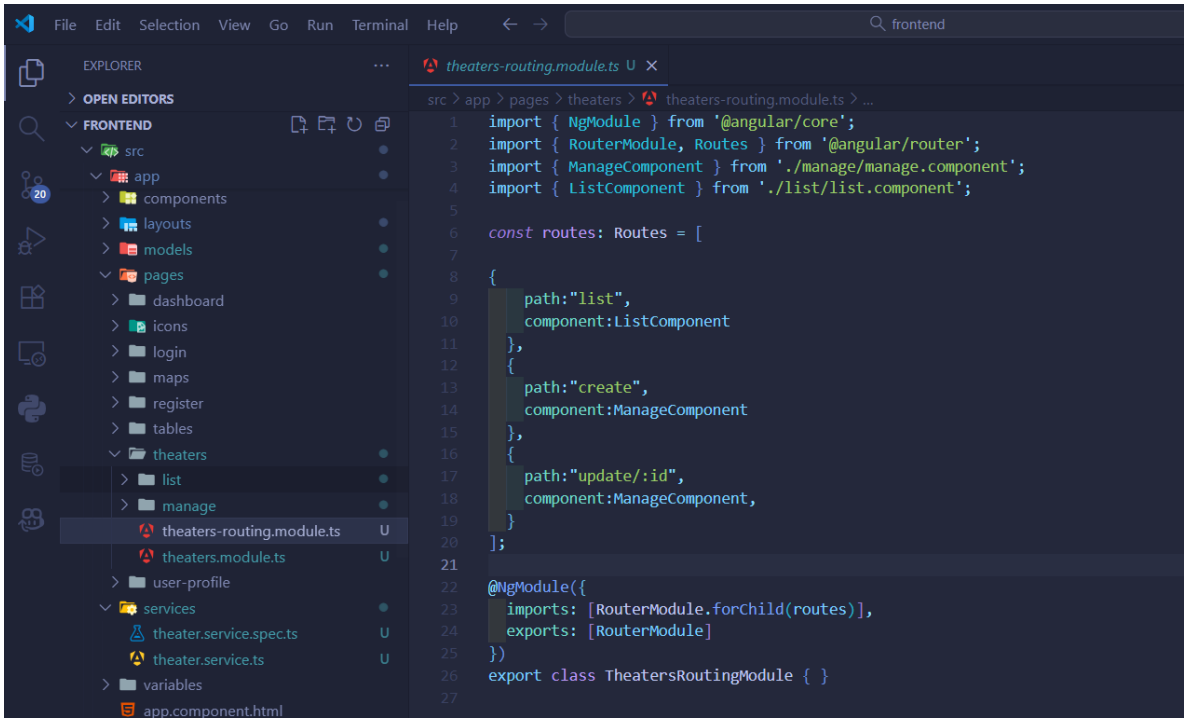
Sustentación (Coménteme la menor cantidad de líneas, para que deje de funcionar x función **manage.components.ts**)

Posible 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)

```
<div *ngIf="getTheFormGroup.capacity.errors && (getTheFormGroup.capacity.dirty || getTheFormGroup.capacity.touched || trySend)">
    <strong *ngIf="getTheFormGroup.capacity.errors.min" class="msnError">Valor
        minimo
        debe ser 1</strong>
    <strong *ngIf="getTheFormGroup.capacity.errors.max" class="msnError">Valor
        máximo
        100 </strong>
    <strong *ngIf="getTheFormGroup.capacity.errors.required"
        class="msnError">Requerido</strong>
</div>
```



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

Enlazado con User.ts → Lines 46 y línea 54

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

Login.component.ts → Linea 24-25-26 → Verificacion de Logeo del usuario

```
@Injectable({
  providedIn: "root",
})
export class SecurityService {  // Servicio para manejar la seguridad y autenticación de usuarios y sesiones y almacenamiento en local storage
  theUser = new BehaviorSubject<User>(new User());
  constructor(private http: HttpClient) {
    this.verifyActualSession();
  }

  /**
   * Realiza la petición al backend con el correo y la contraseña
   * para verificar si existe o no en la plataforma
   * @param infoUsuario JSON con la información de correo y contraseña
   * @returns Respuesta HTTP la cual indica si el usuario tiene permiso de acceso
   */
  login(user: User): Observable<any> { // Método para iniciar sesión del usuario y enviar sus credenciales al backend
    return this.http.post<any>(`${environment.url_security}/login`, user); // Realiza una petición POST al endpoint de login del backend con la info
  }
```

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

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

```
@Component({
  selector: "app-navbar",
  templateUrl: "./navbar.component.html",
  styleUrls: ["./navbar.component.scss"],
})
export class NavbarComponent implements OnInit {
  public focus;
  public listTitles: any[];
  public location: Location;
  user: User;  // Usuario logueado
  subscription: Subscription;
  constructor(
    location: Location,
    private element: ElementRef,
    private router: Router,
    private securityService: SecurityService
  ) {
    this.location = location;
    this.subscription = this.securityService.getUser().subscribe((data) => { //Estar pendiente de la variable global reactiva (Subscription --> simulacion de conectado a
      this.user = data; //Pendiente de que haya un cambio
    });
  }
```

```
<nav class="navbar navbar-top navbar-expand-md navbar-dark" id="navbar-main">
  <div class="container-fluid">
    <form class="navbar-search navbar-search-dark form-inline mr-3 d-none d-md-flex ml-lg-auto">
    </div>
  </form>
  <!-- User -->
  <ul class="navbar-nav align-items-center d-none d-md-flex">
    <li class="nav-item" ngbDropdown placement="bottom-right">
      <a class="nav-link pr-0" role="button" ngbDropdownToggle>
        <div class="media align-items-center">
          <span class="avatar avatar-sm rounded-circle">
            <img alt="Image placeholder" src="assets/img/theme/team-4-800x800.jpg">
          </span>
          <div class="media-body ml-2 d-none d-lg-block">
            <span class="mb-0 text-sm  font-weight-bold">{{user?user.name:"Invitado"}}</span>
          </div>
        </div>
      </a>
```

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

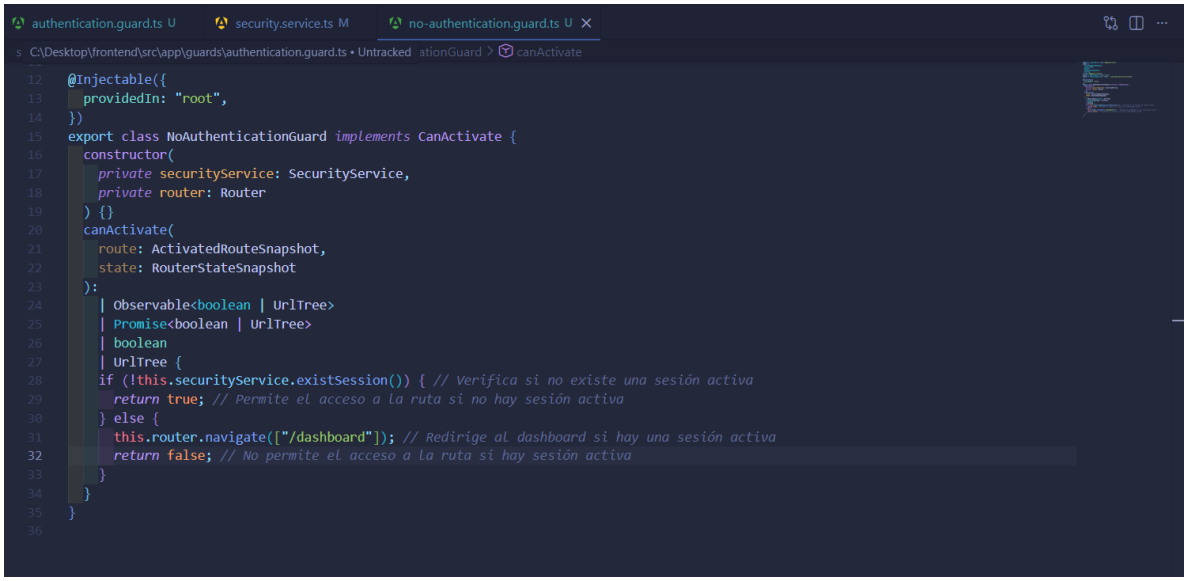
ng g guard guards/NoAuthentication

1. Crear el archive por medio del comando

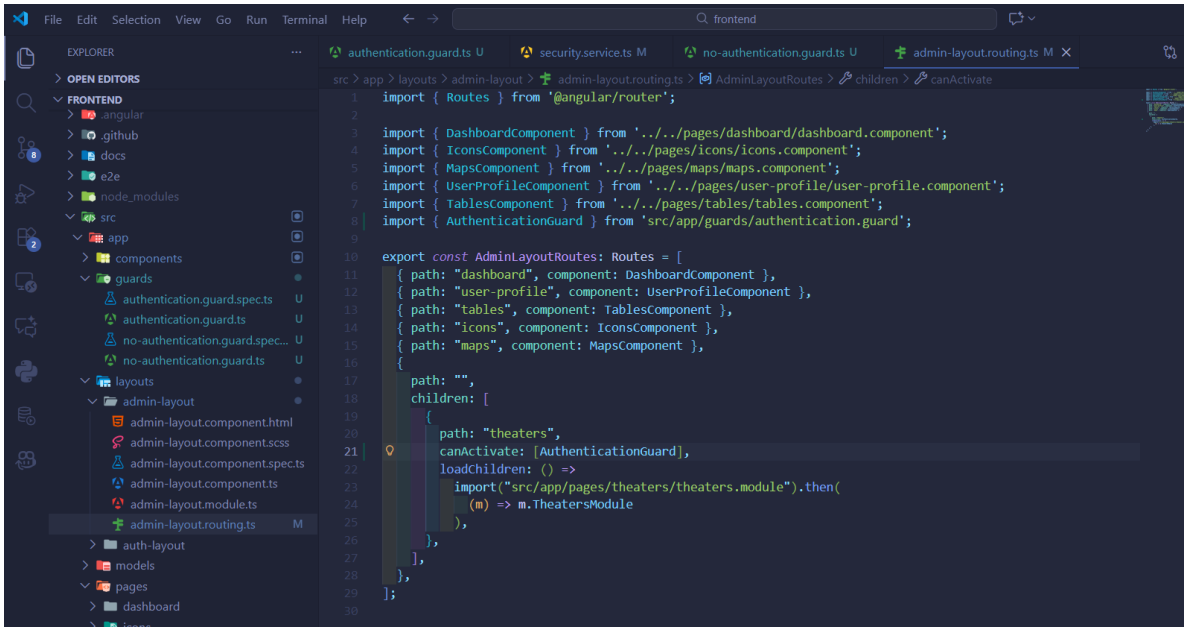
2. Programar el guardian

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)



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



Deja ver las paginas si esta autenticado

Crear interceptor de autentificación

ng g interceptor interceptors/Auth

Creamos nueva ventana en visual

Websockecketserver

Main.py →Codigo 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 codigo compartido