**NOTAS ANGULAR 18 SIN MODULO**

# Instalación

## Instalar Node.js

Descargar e instalar la última versión LTS de Node.js para Windows desde el sitio oficial: <https://nodejs.org/en/>

## Abrir PowerShell

Abrir ventana de comandos ***PowerShell*** de Windows con permisos de administrador.

## Habilitar la ejecución de scripts en Windows

Set-ExecutionPolicy -Scope CurrentUser -ExecutionPolicy RemoteSigned

## Instalar Angular CLI

npm install -g @angular/cli

# Comandos generales

## Versión de NodeJS

npm --version

## Versión de Angular

ng --version

## Ejecutar

ng serve

## Compilar para producción(Optimiza y minifica el código)

ng build –-configuration=production --aot

# Comandos comunes de un proyecto

## Crear nuevo proyecto independiente(Sin necesidad de módulos)

ng new nombre\_app

## Crear nuevo proyecto compartible con módulos

ng new nombre\_app –-standalone=false

## Recrear carpeta ***node\_modules*** o volver a instalar todas las dependencias del proyecto.

npm install

## Crear componente

Ng generate component nombre\_componente

Ó

Ng g c nombre\_componente

## Crear componente en línea(Sin archivo Style y sin archivo Template)

Ng g c nombre\_componente -s -t

## Instalar Bootstrap

npm install | i bootstrap

Agregar configuración de Bootstrap al archivo Angular.json

"styles": [ "src/styles.css", "node\_modules/bootstrap/dist/css/bootstrap.min.css" ],

"scripts": [ "node\_modules/bootstrap/dist/js/bootstrap.bundle.min.js" ]

## Deshabilitar verificación inicialización de propiedades

Agregar configuración para deshabilitar la verificación de inicialización de propiedades al archivo tsconfg.json

{

  "compileOnSave": false,

  "compilerOptions": {

    "strictPropertyInitialization": false,

    "outDir": "./dist/out-tsc",

    "strict": true,

# Interpolación | Texto dinámico en HTML

En el siguiente caso se declaran las propiedades y sus tipos, mas no siempre es necesario declarar u tipo.

## Propiedades

Component.ts:

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

@Component({

  selector: 'app-persona',

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

  styleUrl: './persona.component.css'

})

export class PersonaComponent {

  nombre: string = 'Juan';

  apellido: string = 'Pérez';

  edad: number = 28;

}

Component.html:

<p>

    Nombre: {{nombre}}

    Apellido: {{apellido}}

    edad: {{edad}}

</p>

## Métodos

Component.ts:

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

@Component({

  selector: 'app-persona',

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

  styleUrl: './persona.component.css',

})

export class PersonaComponent {

  private nombre: string = 'Juan';

  private apellido: string = 'Pérez';

  private edad: number = 28;

  getNombre(): string {

    return this.nombre;

  }

  getApellido(){

    return this.apellido;

  }

  getEdad(): number {

    return this.edad;

  }

}

Component.html:

<p>

    Nombre: {{getNombre()}}

    Apellido: {{getApellido()}}

    edad: {{getEdad()}}

</p>

## Template Reference Variable (Repetidor de escritura)

Component.html:

<p>

    Nombre: {{nombre}}

    Apellido: {{apellido}}

    edad: {{edad}}

</p>

<div (keyup)="0">

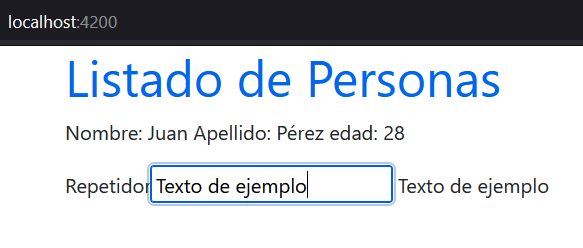
    <label>Repetidor</label>

    <input #entrada />

    {{ entrada.value}}

</div>

Resultado:



## Property Binding en Angular. Asignar valores dinámicos a propiedades HTML

Component.ts

import { Component } from "@angular/core";

@Component({

    selector: 'app-personas',

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

    styleUrl: './personas.component.css'

})

export class PersonasComponent{

    deshabilitar = false;

}

Component.html

<div class="container">

    <div class="row">

        <div class="col">

            <h1>Listado de Personas</h1>

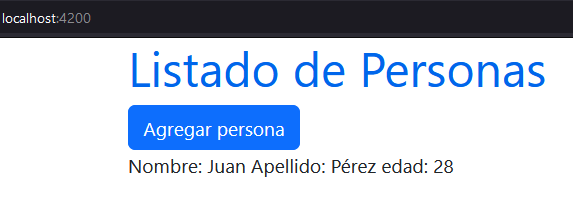
            <button class="btn btn-primary" [disabled]="deshabilitar" Agregar persona</button>

            <app-persona></app-persona>

        </div>

    </div>

</div>

Resultado:

## Event Binding. Escuchar eventos de componentes HTML

Component.ts

import { Component } from "@angular/core";

@Component({

    selector: 'app-personas',

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

    styleUrl: './personas.component.css'

})

export class PersonasComponent{

    deshabilitar = false;

    mensaje = 'No se ha agregado ninguna persona';

    agregarPersona(){

        this.mensaje= 'Persona agregada'

    }

}

Component.html

<div class="container">

    <div class="row">

        <div class="col">

            <h1>Listado de Personas</h1>

            <button class='btn btn-primary'

            (click)='agregarPersona()'>Agregar persona</button>

            <p>{{mensaje}}</p>

            <app-persona></app-persona>

        </div>

    </div>

</div>

## Event Binding. Eventos con parámetros

Component.ts

import { Component } from "@angular/core";

@Component({

    selector: 'app-personas',

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

    styleUrl: './personas.component.css'

})

export class PersonasComponent{

    deshabilitar = false;

    mensaje = 'No se ha agregado ninguna persona';

    titulo= '';

    agregarPersona(){

        this.mensaje= 'Persona agregada'

    }

    modificarTitulo(event: Event){

        this.titulo = (<HTMLInputElement>event.target).value;

    }

}

Component.html

<div class="container">

    <div class="row">

        <div class="col">

            <h1>Listado de Personas</h1>

            <button class='btn btn-primary'

            [disabled]='deshabilitar'

            (click)='agregarPersona()'>Agregar persona</button>

            <p>{{mensaje}}</p>

            <app-persona></app-persona>

            <label>Titulo persona</label>

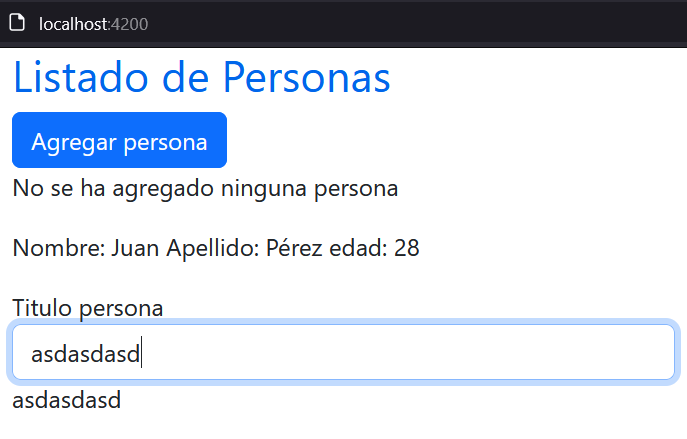
            <input type="text" class="form-control" (input)="modificarTitulo($event)">

            {{titulo}}

        </div>

    </div>

</div>

Resultado:

## Two Way Binding. Enlazar HTML 🡨🡪Componente.

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 { PersonasComponent } from './personas/personas.component';

import { PersonaComponent } from './persona/persona.component';

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

@NgModule({

  declarations: [

    AppComponent, PersonasComponent, PersonaComponent

  ],

  imports: [

    BrowserModule,

    AppRoutingModule,

    FormsModule

  ],

  providers: [],

  bootstrap: [AppComponent]

})

export class AppModule { }

Component.ts

import { Component } from "@angular/core";

@Component({

    selector: 'app-personas',

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

    styleUrl: './personas.component.css'

})

export class PersonasComponent{

    deshabilitar = false;

    mensaje = 'No se ha agregado ninguna persona';

    titulo= 'Ingeniero';

    agregarPersona(){

        this.mensaje= 'Persona agregada'

    }

}

Component.html

<div class="container">

    <div class="row">

        <div class="col">

            <h1>Listado de Personas</h1>

            <button class='btn btn-primary'

            [disabled]='deshabilitar'

            (click)='agregarPersona()'>Agregar persona</button>

            <p>{{mensaje}}</p>

            <app-persona></app-persona>

            <label>Titulo persona</label>

            <input type="text" class="form-control"

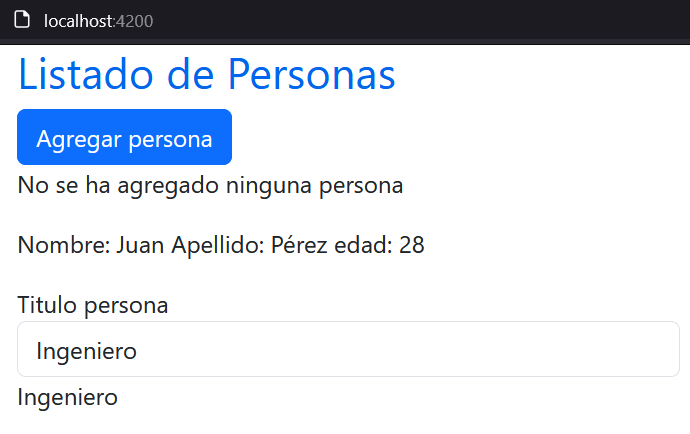
            [(ngModel)]="titulo">

            {{titulo}}

        </div>

    </div>

</div>

Resultado:

# Directivas de Angular

## Directiva \*ngIf

Componente.ts

import { Component } from "@angular/core";

@Component({

    selector: 'app-personas',

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

    styleUrl: './personas.component.css'

})

export class PersonasComponent{

    mostrar = false;

    deshabilitar = false;

    mensaje = 'No se ha agregado ninguna persona';

    titulo= 'Ingeniero';

    agregarPersona(){

        this.mostrar = true;

        this.mensaje= 'Persona agregada'

    }

}

Componente.html

<div class="container">

    <div class="row">

        <div class="col">

            <h1>Listado de Personas</h1>

            <button class='btn btn-primary'

            [disabled]='deshabilitar'

            (click)='agregarPersona()'>Agregar persona</button>

            <p \*ngIf="mostrar">{{mensaje}} con el titulo de {{titulo}}</p>

            <app-persona></app-persona>

            <label>Titulo persona</label>

            <input type="text" class="form-control"

            [(ngModel)]="titulo">

            {{titulo}}

        </div>

    </div>

</div>

## Directiva else con directiva \*ngif

Component.ts

import { Component } from "@angular/core";

@Component({

    selector: 'app-personas',

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

    styleUrl: './personas.component.css'

})

export class PersonasComponent{

    mostrar = false;

    deshabilitar = false;

    mensaje = '';

    titulo= 'Ingeniero';

    agregarPersona(){

        this.mostrar = true;

        this.mensaje= 'Persona agregada'

    }

}

Component.html

<div class="container">

    <div class="row">

        <div class="col">

            <h1>Listado de Personas</h1>

            <button class='btn btn-primary'

            [disabled]='deshabilitar'

            (click)='agregarPersona()'>Agregar persona</button>

            <p \*ngIf="mostrar; else sinPersona">{{mensaje}} con el título de {{titulo}}</p>

            <ng-template #sinPersona>

                <p>No se ha agregado ninguna persona</p>

            </ng-template>

            <app-persona></app-persona>

            <label>Titulo persona</label>

            <input type="text" class="form-control"

            [(ngModel)]="titulo">

            {{titulo}}

        </div>

    </div>

</div>

## Direciva \*ngFor

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 { FormsModule } from '@angular/forms';

@NgModule({

  declarations: [

    AppComponent

  ],

  imports: [

    BrowserModule,

    AppRoutingModule,

    FormsModule

  ],

  providers: [],

  bootstrap: [AppComponent]

})

export class AppModule { }

Persona.model.ts

export class Persona{

    nombre: string = '';

    apellido: string ='';

    constructor(nombre: string, apellido: string){

        this.nombre = nombre;

        this.apellido= apellido;

    }

}

Component.ts

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

import { Persona } from './persona.model';

@Component({

  selector: 'app-root',

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

  styleUrl: './app.component.css'

})

export class AppComponent {

  titulo = 'Listado de Personas';

  nombreInput: string ='';

  apellidoInput: string ='';

  personas: Persona[] = [

    new Persona("Juan","Pérez"),

    new Persona("Laura","Juarez"),

    new Persona("Carla", "Lara")

  ];

  agregarPersona(){

    this.personas.push(new Persona(this.nombreInput, this.apellidoInput));

  }

}

Component.html

<div class="container">

  <div class="col">

    <h3>{{titulo}}</h3>

  </div>

  <hr>

  <form class="row">

    <div class="col-2">

      <label for="nombre" class="sr-only">Nombre</label>

      <input type="text" name="nombre" id="nombre" placeholder="Nombre" class="form-control" [(ngModel)]="nombreInput">

    </div>

    <div class="col-2">

      <label for="apellido" class="sr-only">Apellido</label>

      <input type="text" name="apellido" id="apellido" placeholder="Apellido" class="form-control"

        [(ngModel)]="apellidoInput">

    </div>

    <br>

    <div class="col">

      <button type="submit" class="btn btn-primary" (click)="agregarPersona()">Agregar persona</button>

    </div>

  </form>

</div>

<br>

<div class="container">

  <div class="row">

    <div class="col">

      <div \*ngFor="let persona of personas; let i = index">

        {{i+1}} {{persona.nombre}} {{persona.apellido}}

      </div>

    </div>

  </div>

</div>

# Comunicación entre componentes

## Property Binding entre componetes( @Input ).

app.component.ts

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

import { Persona } from './persona.model';

@Component({

  selector: 'app-root',

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

  styleUrl: './app.component.css'

})

export class AppComponent {

  titulo = 'Listado de Personas';

  nombreInput: string ='';

  apellidoInput: string ='';

  personas: Persona[] = [

    new Persona("Juan","Pérez"),

    new Persona("Laura","Juarez"),

    new Persona("Carla", "Lara")

  ];

  agregarPersona(){

    this.personas.push(new Persona(this.nombreInput, this.apellidoInput));

  }

}

app.component.html

<div class="container">

  <div class="col">

    <h3>{{titulo}}</h3>

  </div>

  <hr>

  <form class="row">

    <div class="col-2">

      <label for="nombre" class="sr-only">Nombre</label>

      <input type="text" name="nombre" id="nombre" placeholder="Nombre" class="form-control" [(ngModel)]="nombreInput">

    </div>

    <div class="col-2">

      <label for="apellido" class="sr-only">Apellido</label>

      <input type="text" name="apellido" id="apellido" placeholder="Apellido" class="form-control"

        [(ngModel)]="apellidoInput">

    </div>

    <br>

    <div class="col">

      <button type="submit" class="btn btn-primary" (click)="agregarPersona()">Agregar persona</button>

    </div>

  </form>

  <br>

  <hr>

  <app-persona \*ngFor="let personaElemento of personas; let i = index" [persona]="personaElemento" [indice]="i">

  </app-persona>

</div>

persona.model.ts

export class Persona{

    nombre: string = '';

    apellido: string ='';

    constructor(nombre: string, apellido: string){

        this.nombre = nombre;

        this.apellido= apellido;

    }

}

persona.component.ts

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

import { Persona } from '../persona.model';

@Component({

  selector: 'app-persona',

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

  styleUrl: './persona.component.css'

})

export class PersonaComponent {

  @Input() persona: Persona;

  @Input() indice: number;

}

persona.component.html

<div class="container">

    <div class="row">

        <div class="col">

            <div>

                {{indice + 1}}: {{persona.nombre}} {{persona.apellido}}

            </div>

        </div>

    </div>

</div>

## Event Binding entre componetes( @Output ).

app.component.ts

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

import { Persona } from './persona.model';

@Component({

  selector: 'app-root',

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

  styleUrl: './app.component.css'

})

export class AppComponent {

  titulo = 'Listado de Personas';

  personas: Persona[] = [

    new Persona("Juan","Pérez"),

    new Persona("Laura","Juarez"),

    new Persona("Carla", "Lara")

  ];

  personaAgregada(persona: Persona){

    this.personas.push(persona);

  }

}

app.component.html

<div class="container">

  <div class="col">

    <h3>{{titulo}}</h3>

  </div> <hr>

  <app-formulario (personaCreada)="personaAgregada($event)">

  </app-formulario>

  <br> <hr>

  <app-persona \*ngFor="let personaElemento of personas; let i = index" [persona]="personaElemento" [indice]="i">

  </app-persona>

</div>

persona.module.ts

export class Persona{

    nombre: string = '';

    apellido: string ='';

    constructor(nombre: string, apellido: string){

        this.nombre = nombre;

        this.apellido= apellido;

    }

}

formulario.component.html

<form class="row">

    <div class="col-2">

        <label for="nombre" class="sr-only">Nombre</label>

        <input type="text" name="nombre" id="nombre" placeholder="Nombre" class="form-control"

            [(ngModel)]="nombreInput">

    </div>

    <div class="col-2">

        <label for="apellido" class="sr-only">Apellido</label>

        <input type="text" name="apellido" id="apellido" placeholder="Apellido" class="form-control"

            [(ngModel)]="apellidoInput">

    </div>

    <br>

    <div class="col">

        <button type="submit" class="btn btn-primary" (click)="agregarPersona()">Agregar persona</button>

    </div>

</form>

formulario.component.ts

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

import { Persona } from '../persona.model';

@Component({

  selector: 'app-formulario',

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

  styleUrl: './formulario.component.css',

})

export class FormularioComponent {

  @Output() personaCreada = new EventEmitter<Persona>();

  nombreInput: string = '';

  apellidoInput: string = '';

  agregarPersona() {

    this.personaCreada.emit(new Persona(this.nombreInput, this.apellidoInput));

  }

}

## Referencias locales

formulario.component.html

<form class="row">

    <div class="col-2">

        <label for="nombre" class="sr-only">Nombre</label>

        <input type="text" name="nombre" id="nombre"

        placeholder="Nombre" class="form-control"

        #nombreInput

        >

    </div>

    <div class="col-2">

        <label for="apellido" class="sr-only">Apellido</label>

        <input type="text" name="apellido" id="apellido"

        placeholder="Apellido" class="form-control"

        #apellidoInput

        >

    </div>

    <br>

    <div class="col">

        <button type="submit" class="btn btn-primary" (click)="agregarPersona(nombreInput, apellidoInput)">Agregar persona</button>

    </div>

</form>

formulario.component.ts

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

import { Persona } from '../persona.model';

@Component({

  selector: 'app-formulario',

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

  styleUrl: './formulario.component.css',

})

export class FormularioComponent {

  @Output() personaCreada = new EventEmitter<Persona>();

  agregarPersona(nombreInput:HTMLInputElement, apellidoInput:HTMLInputElement) {

    this.personaCreada.emit(new Persona(nombreInput.value, apellidoInput.value));

  }

}

## View Child

formulario.component.html

<form class="row">

    <div class="col-2">

        <label for="nombre" class="sr-only">Nombre</label>

        <input type="text" name="nombre" id="nombre"

        placeholder="Nombre" class="form-control"

        #nombreInput >

    </div>

    <div class="col-2">

        <label for="apellido" class="sr-only">Apellido</label>

        <input type="text" name="apellido" id="apellido"

        placeholder="Apellido" class="form-control"

        #apellidoInput >

    </div>

    <br>

    <div class="col">

        <button type="submit" class="btn btn-primary" (click)="agregarPersona()">Agregar persona</button>

    </div>

</form>

formulario.component.ts

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

import { Persona } from '../persona.model';

@Component({

  selector: 'app-formulario',

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

  styleUrl: './formulario.component.css',

})

export class FormularioComponent {

  @Output() personaCreada = new EventEmitter<Persona>();

  @ViewChild('nombreInput') nombreInput: ElementRef;

  @ViewChild('apellidoInput') apellidoInput: ElementRef;

  agregarPersona() {

    this.personaCreada.emit(

      new Persona(

        this.nombreInput.nativeElement.value,

        this.apellidoInput.nativeElement.value)

    );

  }

}

# Servicios

## Servicios en Angular

LoggingService.service.ts

export class LoggingService{

    enviaMensajeAConsola(mensaje: string){

        console.log(mensaje);

    }

}

formulario.component.ts

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

import { Persona } from '../persona.model';

import { LoggingService } from '../LoggingService.service';

@Component({

  selector: 'app-formulario',

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

  styleUrl: './formulario.component.css',

  providers: [LoggingService]

})

export class FormularioComponent {

  @Output() personaCreada = new EventEmitter<Persona>();

  @ViewChild('nombreInput') nombreInput: ElementRef;

  @ViewChild('apellidoInput') apellidoInput: ElementRef;

  constructor(private loggingService: LoggingService){}

  agregarPersona() {

    let persona: Persona;

    persona = new Persona(

      this.nombreInput.nativeElement.value,

      this.apellidoInput.nativeElement.value);

    this.loggingService.enviaMensajeAConsola(

      "Enviamos persona: " + persona.nombre + " " + persona.apellido

    );

    this.personaCreada.emit(persona);

  }

}

## Data service en Angular

personas.service.ts

import { Persona } from './persona.model';

export class PersonasService {

  personas: Persona[] = [new Persona('Juan', 'Pérez'), new Persona('Laura', 'Juarez'), new Persona('Carla', 'Lara'),];

  agregarPersona(persona: Persona) {

    this.personas.push(persona);

  }

}

formulario.component.html

<form class="row">

    <div class="col-2">

        <label for="nombre" class="sr-only">Nombre</label>

        <input type="text" name="nombre" id="nombre"

        placeholder="Nombre" class="form-control"

        #nombreInput >

    </div>

    <div class="col-2">

        <label for="apellido" class="sr-only">Apellido</label>

        <input type="text" name="apellido" id="apellido"

        placeholder="Apellido" class="form-control"

        #apellidoInput >

    </div>

    <br>

    <div class="col">

        <button type="submit" class="btn btn-primary" (click)="agregarPersona()">Agregar persona</button>

    </div>

</form>

formulario.component.ts

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

import { Persona } from '../persona.model';

import { LoggingService } from '../LoggingService.service';

import { PersonasService } from '../personas.service';

@Component({

  selector: 'app-formulario', templateUrl: './formulario.component.html',

  styleUrl: './formulario.component.css',

  providers: [LoggingService],

})

export class FormularioComponent {

  @ViewChild('nombreInput') nombreInput: ElementRef;

  @ViewChild('apellidoInput') apellidoInput: ElementRef;

  constructor(private loggingService: LoggingService, private personasService: PersonasService) {}

  agregarPersona() {

    let persona: Persona;

    persona = new Persona(this.nombreInput.nativeElement.value, …………this.apellidoInput.nativeElement.value);

    this.loggingService.enviaMensajeAConsola(

      'Enviamos persona: ' + persona.nombre + ' ' + persona.apellido

    );

    this.personasService.agregarPersona(persona);

  }

}

app.component.html

<div class="container">

  <div class="col">

    <h3>{{titulo}}</h3>

  </div>

  <hr>

  <app-formulario />

  <br>

  <hr>

  <app-persona \*ngFor="let personaElemento of personas; let i = index" [persona]="personaElemento" [indice]="i" />

</div>

app.component.ts

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

import { Persona } from './persona.model';

import { PersonasService } from './personas.service';

@Component({

  selector: 'app-root',

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

  styleUrl: './app.component.css',

})

export class AppComponent implements OnInit {

  titulo = 'Listado de Personas';

  personas: Persona[];

  constructor(private personasService: PersonasService) {}

  ngOnInit(): void {

    this.personas = this.personasService.personas;

  }

}

## Inyección de servicio dentro de otro servicio

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 { FormsModule } from '@angular/forms';

import { PersonaComponent } from './persona/persona.component';

import { FormularioComponent } from './formulario/formulario.component';

import { PersonasService } from './personas.service';

import { LoggingService } from './LoggingService.service';

@NgModule({

  declarations: [

    AppComponent,

    PersonaComponent,

    FormularioComponent

  ],

  imports: [

    BrowserModule,

    AppRoutingModule,

    FormsModule

  ],

  providers: [PersonasService, LoggingService],

  bootstrap: [AppComponent]

})

export class AppModule { }

LoggigService.service.ts

export class LoggingService{

    enviaMensajeAConsola(mensaje: string){

        console.log(mensaje);

    }

}

persona.service.ts

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

import { LoggingService } from './LoggingService.service';

import { Persona } from './persona.model';

@Injectable()

export class PersonasService {

  personas: Persona[] = [

new Persona('Juan', 'Pérez'),

new Persona('Laura', 'Juarez'),

new Persona('Carla', 'Lara'),];

  constructor(private loggingService: LoggingService){}

  agregarPersona(persona: Persona) {

    this.loggingService.enviaMensajeAConsola("Agregamos persona: " + persona.nombre);

    this.personas.push(persona);

  }

}

## Comunicación de componentes usando servicios

personas.service.ts

import { EventEmitter, Injectable } from '@angular/core';

import { LoggingService } from './LoggingService.service';

import { Persona } from './persona.model';

@Injectable()

export class PersonasService {

  personas: Persona[] = [

    new Persona('Juan', 'Pérez'),

    new Persona('Laura', 'Juarez'),

    new Persona('Carla', 'Lara'),

  ];

  saludar = new EventEmitter<number>();

  constructor(private loggingService: LoggingService) {}

  agregarPersona(persona: Persona) {

    this.loggingService.enviaMensajeAConsola(

      'Agregamos persona: ' + persona.nombre

    );

    this.personas.push(persona);

  }

}

persona.component.ts

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

import { Persona } from '../persona.model';

import { PersonasService } from '../personas.service';

@Component({

  selector: 'app-persona',

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

  styleUrl: './persona.component.css',

})

export class PersonaComponent {

  @Input() persona: Persona;

  @Input() indice: number;

  constructor(private personasService: PersonasService) {}

  emitirSaludo() {

    this.personasService.saludar.emit(this.indice);

  }

}

persona.component.html

<div class="container">

    <div class="row">

        <div class="col">

            <div>

                {{indice + 1}}: {{persona.nombre}} {{persona.apellido}}

                <button

                    class="btn btn-info"

                    (click)="emitirSaludo()"

                >Editar</button>

            </div>

        </div>

    </div>

</div>

formulario.component.ts

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

import { Persona } from '../persona.model';

import { PersonasService } from '../personas.service';

@Component({

  selector: 'app-formulario',

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

  styleUrl: './formulario.component.css',

})

export class FormularioComponent {

  @ViewChild('nombreInput') nombreInput: ElementRef;

  @ViewChild('apellidoInput') apellidoInput: ElementRef;

  constructor(private personasService: PersonasService,) {

    personasService.saludar.subscribe(

      (indice: number) => alert("El indice es: " + indice)

    );

  }

  agregarPersona() {

    let persona: Persona;

    persona = new Persona(

      this.nombreInput.nativeElement.value,

      this.apellidoInput.nativeElement.value

    );

    this.personasService.agregarPersona(persona);

  }

}

# Pipes

En Angular, los pipes son una característica que permite transformar los datos en las plantillas antes de mostrarlos en la vista. Son útiles para realizar operaciones comunes de formato o transformación de datos sin tener que hacerlo directamente en el código del componente.

**¿Qué hacen los pipes?**

Los pipes transforman los datos que se muestran en la vista. Algunos ejemplos comunes son:

* Formatear fechas.
* Convertir texto a mayúsculas/minúsculas.
* Formatear números o monedas.
* Manejar valores nulos o indefinidos.

**Sintaxis**

Se utilizan en las plantillas de Angular con el operador **|**. Aquí hay algunos ejemplos de cómo funcionan:

Ts

import {

  CurrencyPipe,

  DatePipe,

  DecimalPipe,

  JsonPipe,

  LowerCasePipe,

  PercentPipe,

  SlicePipe,

  TitleCasePipe,

  UpperCasePipe,

} from '@angular/common';

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

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

@Component({

  selector: 'app-root',

  standalone: true,

  imports: [

    RouterOutlet,

    DatePipe,

    CurrencyPipe,

    LowerCasePipe,

    UpperCasePipe,

    PercentPipe,

    DecimalPipe,

    TitleCasePipe,

    JsonPipe,

    SlicePipe

  ],

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

  styleUrl: './app.component.css',

})

export class AppComponent {

  title = 'movies-webapp';

}

**html**

<!-- Convertir una cadena a mayúsculas -->

<p>{{ 'hola mundo' | uppercase }}</p> <!-- Resultado: HOLA MUNDO -->

<!-- Formatear una fecha -->

<p>{{ fechaActual | date:'fullDate' }}</p> <!-- Resultado: "Monday, September 20, 2024" -->

<!-- Formatear un número como moneda -->

<p>{{ 1234.56 | currency:'MXN' }}</p> <!-- Resultado: $1,234.56 -->

## Pipes integrados en Angular

Angular ofrece varios pipes integrados, como:

### **DatePipe:** Formatea una fecha.

html

{{ today | date:'short' }} <!-- "9/20/24" -->

### **UpperCasePipe y LowerCasePipe:** Convierte cadenas a mayúsculas o minúsculas.

html

{{ 'angular' | uppercase }} <!-- "ANGULAR" -->

{{ 'ANGULAR' | lowercase }} <!-- "angular" -->

### **CurrencyPipe:** Formatea un número como moneda.

html

{{ 1234.56 | currency:'MXN' }} <!-- "$1,234.56" -->

### **DecimalPipe:** Formatea números decimales.

html

{{ 1234.5678 | number:'1.2-2' }} <!-- "1,234.57" -->

### **PercentPipe:** Formatea un número como porcentaje.

html

{{ 0.25 | percent }} <!-- "25%" -->

### **TitleCasePipe**

Este pipe convierte una cadena de texto en "Title Case", lo que significa que convierte la primera letra de cada palabra en mayúscula y las demás en minúscula.

**Ejemplo de uso:**

html

<p>{{ 'angular es genial' | titlecase }}</p>

**Resultado:** Angular Es Genial

Es útil para formatear títulos o textos que requieran tener cada palabra con la primera letra en mayúsculas.

### **JsonPipe**

Este pipe convierte un objeto o una estructura de datos en una representación **JSON** (JavaScript Object Notation) legible. Se usa comúnmente para depuración o visualización de datos en las plantillas de Angular.

**Ejemplo de uso:**

html

<p>{{ {nombre: 'Juan', edad: 30} | json }}</p>

**Resultado:**

json

{

  "nombre": "Juan",

  "edad": 30

}

Es útil cuando quieres ver rápidamente la estructura de un objeto o una lista en la interfaz de usuario sin necesidad de escribir código adicional.

### SlicePipe

Este pipe permite extraer una porción de una lista o cadena de texto. Funciona de manera similar al método slice() de JavaScript y toma dos argumentos: el índice de inicio y, opcionalmente, el índice de fin.

Ejemplo de uso con una cadena:

html

<p>{{ 'Angular' | slice:1:4 }}</p>

**Resultado:** ngu

Ejemplo de uso con una lista:

html

<p>{{ [10, 20, 30, 40, 50] | slice:1:3 }}</p>

**Resultado:** [20, 30]

Es útil cuando quieres mostrar solo una parte de una lista o cadena, por ejemplo, paginación de resultados o truncar texto.

## Pipes personalizados

Si necesitas una transformación que no esté disponible con los pipes integrados, puedes crear un pipe personalizado. Para hacerlo, necesitas crear una clase que implemente la interfaz **PipeTransform**.

Ejemplo de un pipe personalizado que convierte texto a título (cada palabra empieza con mayúscula):

### Generar el pipe:

bash

ng generate pipe titlecase

### Definir el pipe en el archivo generado:

typescript

import { Pipe, PipeTransform } from '@angular/core';

@Pipe({

  name: 'titlecase',

})

export class TitlecasePipe implements PipeTransform {

  transform(value: string): string {

    return value

      .split(' ')

      .map((word) => word.charAt(0).toUpperCase() + word.slice(1).toLowerCase())

      .join(' ');

  }

}

### Usar el pipe en la plantilla:

html

<p>{{ 'hola mundo' | titlecase }}</p> <!-- Resultado: "Hola Mundo" -->

# Apariencia

## ngClass (estilo condicional)

formulario.component.html

<input [ngClass]="{redfocus: tipo === 'egresoOperacion'}" type="text"

class="agregar\_descripcion"

    placeholder="Agregar Descripción" name="descripcionInput" [(ngModel)]="descripcionInput" required="true">

## Validación de formularios

formulario.component.html

<form (ngSubmit)="f.form.valid && agregarValor()" #f="ngForm">

    <div class="agregar">

        <div class="agregar\_contenedor">

            <select [ngClass]="{redfocus: tipo === 'egresoOperacion'}" (change)="tipoOperacion($event)"

                class="agregar\_tipo">

                <option value="ingresoOperacion" selected>+</option>

                <option value="egresoOperacion">-</option>

            </select>

            <input [ngClass]="{redfocus: tipo === 'egresoOperacion'}" type="text" class="agregar\_descripcion"

                placeholder="Agregar Descripción" name="descripcionInput" [(ngModel)]="descripcionInput"

                required="true">

            <input [ngClass]="{redfocus: tipo === 'egresoOperacion'}" type="number" class="agregar\_valor"

                placeholder="Valor" name="valorInput" [(ngModel)]="valorInput" required="true">

            <button type="submit" class="agregar\_btn">

                <i class="ion-ios-checkmark-outline" [ngClass]="{red: tipo === 'egresoOperacion'}"></i>

            </button>

        </div>

    </div>

</form>

# Routing

## RoutingModule

app-routing.module.ts

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

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

import { PersonasComponent } from './personas/personas.component';

import { FormularioComponent } from './personas/formulario/formulario.component';

const routes: Routes = [

  {path: '', component: PersonasComponent},

  {path: 'personas', component: PersonasComponent},

  {path: 'personas/agregar', component: FormularioComponent},

  {path: 'personas/:id', component: FormularioComponent}

];

@NgModule({

  imports: [RouterModule.forRoot(routes)],

  exports: [RouterModule]

})

export class AppRoutingModule { }

## Routing-outlet

La etiqueta ***<router-outlet>*** en el archivo HTML especifica a partir de donde se reemplazará el contenido a través de las rutas configuradas en el archivo app-routing.module.ts

app.component.html

<div class="box" id="heading">

  <h1>{{titulo}}</h1>

</div>

<router-outlet></router-outlet>

## RouterLink

persona.component.ts

<div class="item">

    <ul><li>

            <a [routerLink]="['/personas', indice]">

                {{persona.nombre}} {{persona.apellido}}

            </a>

        </li>

    </ul>

</div>

## Route params (parámetros)

Se define parámetro:

app-routing.module.ts

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

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

import { PersonasComponent } from './personas/personas.component';

import { FormularioComponent } from './personas/formulario/formulario.component';

const routes: Routes = [

  {path: '', component: PersonasComponent},

  {path: 'personas', component: PersonasComponent},

  {path: 'personas/agregar', component: FormularioComponent},

  {path: 'personas/:id', component: FormularioComponent}

];

@NgModule({

  imports: [RouterModule.forRoot(routes)],

  exports: [RouterModule]

})

export class AppRoutingModule { }

Envio de parámetro desde código:

persona.component.html

<div class="item">

    <ul><li>

            <a [routerLink]="['/personas', indice]">

                {{persona.nombre}} {{persona.apellido}}

            </a>

        </li>

    </ul>

</div>

Leer parámetro:

formulario.component.ts

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

import { Persona } from '../../persona.model';

import { PersonasService } from '../../personas.service';

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

@Component({

  selector: 'app-formulario',

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

  styleUrl: './formulario.component.css',

})

export class FormularioComponent implements OnInit {

  nombreInput: string;

  apellidoInput: string;

  index: number;

  modoEdicion: number;

  constructor(

    private personasService: PersonasService,

    private router: Router,

    private route: ActivatedRoute

  ) {

    personasService.saludar.subscribe((indice: number) =>

      alert('El indice es: ' + indice)

    );

  }

  ngOnInit() {

    this.index = this.route.snapshot.params['id'];

    if (this.index ¡= null) {

      let persona: Persona = this.personasService.encontrarPersona(this.index);

      this.nombreInput = persona.nombre;

      this.apellidoInput = persona.apellido;

    }

  }

}

## Route queyParams (parámetros)

Envio de queryParams:

persona.component.html

<div class="item">

    <ul>

        <li>

            <a [routerLink]="['/personas', indice]"

            [queryParams]="{modoEdicion:'1'}"

            >

                {{persona.nombre}} {{persona.apellido}}

            </a>

        </li>

    </ul>

</div>

Recibir / leer queryParams:

formulario.component.ts

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

import { Persona } from '../../persona.model';

import { PersonasService } from '../../personas.service';

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

@Component({

  selector: 'app-formulario',

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

  styleUrl: './formulario.component.css',

})

export class FormularioComponent implements OnInit {

  nombreInput: string;

  apellidoInput: string;

  index: number;

  modoEdicion: number;

  constructor(

    private personasService: PersonasService,

    private router: Router,

    private route: ActivatedRoute

  ) {

    personasService.saludar.subscribe((indice: number) =>

      alert('El indice es: ' + indice)

    );

  }

  ngOnInit() {

    this.modoEdicion = +this.route.snapshot.queryParams['modoEdicion'];

    if (this.modoEdicion != null && this.modoEdicion === 1) {

      let persona: Persona = this.personasService.encontrarPersona(this.index);

      this.nombreInput = persona.nombre;

      this.apellidoInput = persona.apellido;

    }

  }

}

## Navigate (redireccionar a ruta)

formulario.component.ts

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

import { Persona } from '../../persona.model';

import { PersonasService } from '../../personas.service';

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

@Component({

  selector: 'app-formulario',

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

  styleUrl: './formulario.component.css',

})

export class FormularioComponent implements OnInit {

  nombreInput: string;

  apellidoInput: string;

  index: number;

  modoEdicion: number;

  constructor(

    private personasService: PersonasService,

    private router: Router,

    private route: ActivatedRoute

  ) {

    personasService.saludar.subscribe((indice: number) =>

      alert('El indice es: ' + indice)

    );

  }

  onGuardarPersona() {

    let persona = new Persona(this.nombreInput, this.apellidoInput);

    if (this.modoEdicion != null && this.modoEdicion === 1) {

      this.personasService.modificarPersona(this.index, persona);

    } else {

      this.personasService.agregarPersona(persona);

    }

    this.router.navigate(['personas']);

  }

  eliminarPersona() {

    if (this.index != null) {

      this.personasService.eliminarPersona(this.index);

    }

    this.router.navigate(['personas']);

  }

}

## Child route (rutas hijas o secundarias)

app-routing.module.ts

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

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

import { PersonasComponent } from './personas/personas.component';

import { FormularioComponent } from './personas/formulario/formulario.component';

const routes: Routes = [

  {path: '', component: PersonasComponent},

  {path: 'personas', component: PersonasComponent, children:[

    {path: 'agregar', component: FormularioComponent},

    {path: ':id', component: FormularioComponent}

  ]}

];

@NgModule({

  imports: [RouterModule.forRoot(routes)],

  exports: [RouterModule]

})

export class AppRoutingModule { }

personas.component.html

<div style="text-align: center;">

  <button style="cursor: pointer;" (click)="agregar()">+</button>

</div>

<div class="box">

  <app-persona \*ngFor="let personaElemento of personas; let i = index" [persona]="personaElemento" [indice]="i" />

</div>

<router-outlet/>

## Componente de error (componente de error de ruta no encontrada)

app-routing.module.ts

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

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

import { PersonasComponent } from './personas/personas.component';

import { FormularioComponent } from './personas/formulario/formulario.component';

import { ErrorComponent } from './error/error.component';

const routes: Routes = [

  {path: '', component: PersonasComponent},

  {path: 'personas', component: PersonasComponent, children:[

    {path: 'agregar', component: FormularioComponent},

    {path: ':id', component: FormularioComponent}

  ]},

  {path: '\*\*', component: ErrorComponent}

];

@NgModule({

  imports: [RouterModule.forRoot(routes)],

  exports: [RouterModule]

})

export class AppRoutingModule { }

error.component.html

<div class="box">

    <p>

        ¡No se encuentra el recurso solicitado!

    </p>

</div>

# Peticiones HTTP (Cliente API REST)

## Configuración

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 { FormsModule } from '@angular/forms';

import { PersonaComponent } from './personas/persona/persona.component';

import { FormularioComponent } from './personas/formulario/formulario.component';

import { PersonasService } from './personas.service';

import { LoggingService } from './LoggingService.service';

import { PersonasComponent } from './personas/personas.component';

import { ErrorComponent } from './error/error.component';

import { DataService } from './data.service';

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

@NgModule({

  declarations: [

    AppComponent,

    PersonaComponent,

    FormularioComponent,

    PersonasComponent,

    ErrorComponent

  ],

  imports: [

    BrowserModule,

    AppRoutingModule,

    FormsModule,

    HttpClientModule

  ],

  providers: [PersonasService, LoggingService, DataService],

  bootstrap: [AppComponent]

})

export class AppModule { }

## Método POST

data.service.ts

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

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

import { Persona } from './persona.model';

import { Observable } from 'rxjs';

@Injectable()

export class DataService {

  private url: string =

    'https://listado-personas-cd6c1-default-rtdb.firebaseio.com/datos';

  constructor(private httpClient: HttpClient) {}

  guardarPersonas(personas: Persona[]): void {

    this.httpClient.post(this.url + '.json', personas).subscribe({

      next: (response) =>

        console.log('Resultado de guardar Personas: ', response),

      error: (error) => console.log('Error al guardar Personas: ', error),

      complete: () => console.log('Petición completada con éxito'),

    });

  }

}

## Método PUT

data.service.ts

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

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

import { Persona } from './persona.model';

import { Observable } from 'rxjs';

@Injectable()

export class DataService {

  private url: string =

    'https://listado-personas-cd6c1-default-rtdb.firebaseio.com/datos';

  constructor(private httpClient: HttpClient) {}

  modificarPersona(index: number, persona: Persona): void {

    this.httpClient.put(this.url + '/' + index + '.json', persona).subscribe({

      next: (response) =>

        console.log('Resultado modificar Persona: ', response),

      error: (error) => console.log('Error en modificar Persona: ', error),

      complete: () => console.log('Modificación completada'),

    });

  }

}

## Método GET

data.service.ts

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

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

import { Persona } from './persona.model';

import { Observable } from 'rxjs';

@Injectable()

export class DataService {

  private url: string =

    'https://listado-personas-cd6c1-default-rtdb.firebaseio.com/datos';

  constructor(private httpClient: HttpClient) {}

  cargarPersonas(): Observable<Persona[]> {

    return this.httpClient.get<Persona[]>(this.url + '.json');

  }

}

## Método DELETE

data.service.ts

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

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

import { Persona } from './persona.model';

import { Observable } from 'rxjs';

@Injectable()

export class DataService {

  private url: string =

    'https://listado-personas-cd6c1-default-rtdb.firebaseio.com/datos';

  constructor(private httpClient: HttpClient) {}

  eliminarPersona(index: number): void {

    this.httpClient.delete(this.url + '/' + index + '.json').subscribe({

      next: (response) =>

        console.log('Resultado de eliminar Persona: ', response),

      error: (error) => console.log('Error en eliminar Persona: ', error),

      complete: () => console.log('Eliminación completada'),

    });

  }

}

# Autenticación y autorización en Angular

## Login

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 { FormsModule } from '@angular/forms';

import { PersonaComponent } from './personas/persona/persona.component';

import { FormularioComponent } from './personas/formulario/formulario.component';

import { PersonasService } from './personas.service';

import { LoggingService } from './LoggingService.service';

import { PersonasComponent } from './personas/personas.component';

import { ErrorComponent } from './error/error.component';

import { DataService } from './data.service';

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

import { LoginComponent } from './login/login.component';

import { FirebaseService } from './firebase.service';

import { LoginService } from './login.service';

import { LoginGuardian } from './login-guardian.service';

@NgModule({

  declarations: [

    AppComponent,

    PersonaComponent,

    FormularioComponent,

    PersonasComponent,

    ErrorComponent,

    LoginComponent

  ],

  imports: [

    BrowserModule,

    AppRoutingModule,

    FormsModule,

    HttpClientModule

  ],

  providers: [PersonasService, LoginService, LoggingService, DataService,

    FirebaseService, LoginGuardian],

  bootstrap: [AppComponent]

})

export class AppModule { }

login.service.ts

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

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

import { signInWithEmailAndPassword } from 'firebase/auth';

import { FirebaseService } from './firebase.service';

@Injectable()

export class LoginService {

    token: string | null;

    constructor(private router: Router, private firebaseService: FirebaseService) {

        // Usa la instancia de auth desde FirebaseService

    }

    login(email: string, password: string) {

        const auth = this.firebaseService.auth;

        signInWithEmailAndPassword(auth, email, password)

        .then(() => {

            auth.currentUser?.getIdToken().then(token => {

                this.token = token;

                this.router.navigate(['/']);

                });

            })

            .catch(error => {

                console.error('Error al iniciar sesión:', error);

            }

        );

    }

    getIdToken() {

        return this.token;

    }

    isAutenticado(){

        return this.token != null;

    }

    logout(){

        const auth = this.firebaseService.auth;

        auth.signOut()

        .then(() => {

                this.token = null;

                this.router.navigate(['login']);

        }).catch(error => console.log("error logout: " + error));

    }

}

Login-guardian.service.ts

import { ActivatedRouteSnapshot, CanActivate, GuardResult, MaybeAsync, Router, RouterStateSnapshot } from "@angular/router";

import { LoginService } from "./login.service";

import { Injectable } from "@angular/core";

@Injectable()

export class LoginGuardian implements CanActivate{

    constructor(private loginService: LoginService,

        private router: Router

    ){}

    canActivate(route: ActivatedRouteSnapshot, state: RouterStateSnapshot): MaybeAsync<GuardResult> {

        if(this.loginService.isAutenticado()){

            return true;

        }else{

            this.router.navigate(['login']);

            return false;

        }

    }

}

data.service.ts

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

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

import { Persona } from './persona.model';

import { Observable } from 'rxjs';

import { LoginService } from './login.service';

@Injectable()

export class DataService {

  private url: string =

    'https://listado-personas-cd6c1-default-rtdb.firebaseio.com/datos';

  constructor(private httpClient: HttpClient,

    private loginService: LoginService) {}

  cargarPersonas(): Observable<Persona[]> {

    const token = this.loginService.getIdToken();

    return this.httpClient.get<Persona[]>(this.url + '.json?auth=' + token);

  }

  guardarPersonas(personas: Persona[]): void {

    const token = this.loginService.getIdToken();

    this.httpClient.put(this.url + '.json?auth=' + token, personas).subscribe({

      next: (response) => console.log('Resultado de guardar Personas: ', response),

      error: (error) => console.log('Error al guardar Personas: ', error),

      complete: () => console.log('Petición completada con éxito'),

    });}

  modificarPersona(index: number, persona: Persona): void {

    const token = this.loginService.getIdToken();

    this.httpClient.put(this.url + '/' + index + '.json?auth=' + token, persona).subscribe({

      next: (response) => console.log('Resultado modificar Persona: ', response),

      error: (error) => console.log('Error en modificar Persona: ', error),

      complete: () => console.log('Modificación completada'),

    });}

  eliminarPersona(index: number): void {

    const token = this.loginService.getIdToken();

    this.httpClient.delete(this.url + '/' + index + '.json?auth=' + token).subscribe({

      next: (response) => console.log('Resultado de eliminar Persona: ', response),

      error: (error) => console.log('Error en eliminar Persona: ', error),

      complete: () => console.log('Eliminación completada'),

    });}

}

app-routing.module.ts

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

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

import { PersonasComponent } from './personas/personas.component';

import { FormularioComponent } from './personas/formulario/formulario.component';

import { ErrorComponent } from './error/error.component';

import { LoginComponent } from './login/login.component';

import { LoginGuardian } from './login-guardian.service';

const routes: Routes = [

  {path: '', component: PersonasComponent, canActivate:[LoginGuardian]},

  {path: 'personas', component: PersonasComponent, canActivate:[LoginGuardian], children:[

    {path: 'agregar', component: FormularioComponent},

    {path: ':id', component: FormularioComponent}

  ]},

  {path: 'login', component: LoginComponent},

  {path: '\*\*', component: ErrorComponent}

];

@NgModule({

  imports: [RouterModule.forRoot(routes)],

  exports: [RouterModule]

})

export class AppRoutingModule { }

firebase.service.ts

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

import { initializeApp } from 'firebase/app';

import { getAuth, Auth } from 'firebase/auth';

import { getFirestore, Firestore } from 'firebase/firestore';

@Injectable()

export class FirebaseService {

  private firebaseConfig = {

    apiKey: 'AIzaSyC42sHkjIKcQ92fv804w-hGdOoeLQM49bI',

    authDomain: 'listado-personas-cd6c1.firebaseapp.com',

    databaseURL: 'https://listado-personas-cd6c1-default-rtdb.firebaseio.com',

    projectId: 'listado-personas-cd6c1',

    storageBucket: 'listado-personas-cd6c1.appspot.com',

    messagingSenderId: '136831904588',

    appId: '1:136831904588:web:c28408262cb6fef6c17de9',

  };

  public auth: Auth;

  public db: Firestore;

  constructor() {

    const app = initializeApp(this.firebaseConfig);

    this.auth = getAuth(app);

    this.db = getFirestore(app);

  }

}

login.component.ts

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

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

import { LoginService } from '../login.service';

@Component({

  selector: 'app-login',

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

  styleUrl: './login.component.css'

})

export class LoginComponent {

  constructor(private loginService: LoginService){}

    login(form: NgForm){

      const email = form.value.email;

      const password = form.value.password;

      this.loginService.login(email, password);

    }

}

login.component.html

<div class="box">

    <form class="item" #f="ngForm" (ngSubmit)="f.form.valid && login(f)">

        <div>

            <input type="email" name="email" placeholder="Email" ngModel required="true" size="20">

        </div>

        <div>

            <input type="password" name="password" placeholder="Password" ngModel required="true" size="15">

        </div>

        <button type="submit" style="cursor: pointer;">+</button>

    </form>

</div>