**Step1. register.component.ts**

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

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

import { FormBuilder, FormGroup, Validators, ReactiveFormsModule, AbstractControl } from '@angular/forms';

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

@Component({

selector: 'app-register',

standalone: true,

imports: [CommonModule, ReactiveFormsModule],

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

})

export class RegisterComponent {

registerForm: FormGroup;

submitted = false;

constructor(private fb: FormBuilder, private router: Router) {

this.registerForm = this.fb.group({

username: ['', [Validators.required, Validators.pattern(/^[A-Za-z]{3,}$/)]],

password: ['', [Validators.required, Validators.minLength(6)]],

confirmPassword: ['', Validators.required],

}, { validators: this.passwordsMatch });

}

// Getter for easy access to form controls

get f() {

return this.registerForm.controls;

}

passwordsMatch(group: AbstractControl): { [key: string]: boolean } | null {

const password = group.get('password')?.value;

const confirmPassword = group.get('confirmPassword')?.value;

return password === confirmPassword ? null : { mismatch: true };

}

canDeactivate(): boolean {

return this.submitted || confirm('Are you sure you want to discard your changes?');

}

save() {

if (this.registerForm.valid) {

this.submitted = true;

alert('Registered Successfully!');

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

} else {

this.registerForm.markAllAsTouched();

}

}

cancel() {

if (this.canDeactivate()) {

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

}

}

}

**Step 2. register.component.html**

<h2>Register</h2>

<form [formGroup]="registerForm" (ngSubmit)="save()">

<div>

<label>Username:</label>

<input type="text" formControlName="username" />

<div \*ngIf="f.username.touched && f.username.invalid">

<small \*ngIf="f.username.errors?.required">Username is required</small>

<small \*ngIf="f.username.errors?.pattern">Only letters, min 3 chars</small>

</div>

</div>

<div>

<label>Password:</label>

<input type="password" formControlName="password" />

<div \*ngIf="f.password.touched && f.password.invalid">

<small \*ngIf="f.password.errors?.required">Password is required</small>

<small \*ngIf="f.password.errors?.minlength">Min 6 characters</small>

</div>

</div>

<div>

<label>Confirm Password:</label>

<input type="password" formControlName="confirmPassword" />

<div \*ngIf="f.confirmPassword.touched && registerForm.errors?.mismatch">

<small>Passwords do not match</small>

</div>

</div>

<button type="submit">Save</button>

<button type="button" (click)="cancel()">Cancel</button>

</form>

Step3: ng generate guard guards/can-deactivate

Or

ng g g guards/can-deactivate

**step4. can-deactivate.guard.ts**

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

import { Observable } from 'rxjs';

export interface CanComponentDeactivate {

canDeactivate: () => Observable<boolean> | Promise<boolean> | boolean;

}

export const canDeactivateGuard: CanDeactivateFn<CanComponentDeactivate> = (

component

) => {

return component.canDeactivate ? component.canDeactivate() : true;

};

Step5  **app.routes.ts**

{

path: 'register',

component: RegisterComponent,

canDeactivate: [canDeactivateGuard]

},