**Source Code for myPortfolio:**

**Index.html:**

<!doctype html>

<html lang="en">

<head>

  <meta charset="utf-8">

  <title>MyPortfolio</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@4.6.2/dist/css/bootstrap.min.css" integrity="sha384-xOolHFLEh07PJGoPkLv1IbcEPTNtaed2xpHsD9ESMhqIYd0nLMwNLD69Npy4HI+N" crossorigin="anonymous">

</head>

<body>

  <app-root></app-root>

  <script src="https://cdn.jsdelivr.net/npm/jquery@3.5.1/dist/jquery.slim.min.js" integrity="sha384-DfXdz2htPH0lsSSs5nCTpuj/zy4C+OGpamoFVy38MVBnE+IbbVYUew+OrCXaRkfj" crossorigin="anonymous"></script>

  <script src="https://cdn.jsdelivr.net/npm/bootstrap@4.6.2/dist/js/bootstrap.bundle.min.js" integrity="sha384-Fy6S3B9q64WdZWQUiU+q4/2Lc9npb8tCaSX9FK7E8HnRr0Jz8D6OP9dO5Vg3Q9ct" crossorigin="anonymous"></script>

</body>

</html>

**Dashboard.component.html:**

<div class="container-fluid">

    <h2>Dashboard</h2>

    <h3 style="color:rgb(232, 108, 66) ;">Welcome to MyPortfolio</h3>

    <img src="https://godesign.pk/wp-content/uploads/2020/11/portfolio.jpeg"

            width="100%" style="opacity:0.7 ;">

</div>

**Signin.component.html:**

<div class="container-fluid" style="padding-top: 40px;">

    <div class="container">

        <div class="row">

            <div class="col-md-6 offset-md-3 ">

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

                    <h2 style="text-align:center">Sign in</h2>

                    <!--email-->

                    <div class="form-group">

                        <label>Email</label>

                        <input type="email" formControlName="email" class="form-control" [ngClass]="{'is-invalid': submitted && f['email'].errors}"

                                placeholder="Email"/>

                        <!--Custom Validations-->

                        <div \*ngIf="submitted && f['email'].errors">

                            <div \*ngIf="submitted && f['email'].errors['required']">

                                Email  is required and Can't Be Empty

                            </div>

                        </div>

                    </div>

                    <!--Password-->

                    <div class="form-group">

                        <label>Password</label>

                        <input type="password" formControlName="password" class="form-control" [ngClass]="{'is-invalid': submitted && f['password'].errors}"

                                placeholder="Password"/>

                        <!--Custom Validations-->

                        <div \*ngIf="submitted && f['password'].errors">

                            <div \*ngIf="submitted && f['password'].errors['required']">

                                Password must be minimum 8 character long and Can't Be Empty

                            </div>

                        </div>

                    </div>

                    <div class="form-group text-center">

                        <button class="btn btn-primary" routerLink="/dashboard">Login</button>

                    </div>

                </form>

            </div>

        </div>

    </div>

</div>

**Signup.component.html:**

<div class="jumbotron">

    <div class="container">

        <div class="row">

            <div class="col-md-6 offset-md-3">

                <h2 style="text-align:center">Sign Up</h2>

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

                    <div class="form-group">

                        <label>FirstName</label>

                        <input type="text" formControlName="firstName" class="form-control" [ngClass]="{'is-invalid': submitted && f['firstName'].errors}"

                                placeholder="First Name"/>

                        <!--Custom Validations-->

                        <div \*ngIf="submitted && f['firstName'].errors">

                            <div \*ngIf="submitted && f['firstName'].errors['required']">

                                FirstName  is required and Can't Be Empty

                            </div>

                        </div>

                    </div>

                    <!--Last Name-->

                    <div class="form-group">

                        <label>LastName</label>

                        <input type="text" formControlName="lastName" class="form-control" [ngClass]="{'is-invalid': submitted && f['lastName'].errors}"

                        placeholder="Last Name"/>

                        <!--Custom Validations-->

                        <div \*ngIf="submitted && f['lastName'].errors">

                            <div \*ngIf="submitted && f['lastName'].errors['required']">

                                LastName  is required and Can't Be Empty

                            </div>

                        </div>

                    </div>

                    <!--email-->

                    <div class="form-group">

                        <label>Email</label>

                        <input type="email" formControlName="email" class="form-control" [ngClass]="{'is-invalid': submitted && f['email'].errors}"

                        placeholder="Email"/>

                        <!--Custom Validations-->

                        <div \*ngIf="submitted && f['email'].errors">

                            <div \*ngIf="submitted && f['email'].errors['required']">

                                Email  is required and Can't Be Empty

                            </div>

                        </div>

                    </div>

                    <!--Password-->

                    <div class="form-group">

                        <label>Password</label>

                        <input type="password" formControlName="password" class="form-control" [ngClass]="{'is-invalid': submitted && f['password'].errors}"

                        placeholder="Password"/>

                        <!--Custom Validations-->

                        <div \*ngIf="submitted && f['password'].errors">

                            <div \*ngIf="submitted && f['password'].errors['required']">

                                Password must be minimum 8 character long and Can't Be Empty

                            </div>

                        </div>

                    </div>

                    <div class="form-group text-center">

                        <button class="btn btn-primary" routerLink="/signin" >Register</button>

                    </div>

                </form>

            </div>

        </div>

    </div>

</div>

**App.component.html:**

<nav class="navbar bg-dark navbar-dark">

  <a routerLink=" " class="navbar-brand">MyPortfolio</a>

  <ul class="nav " >

    <li class="nav-item">

      <a routerLink="signin" class="nav-link">Signin</a>

    </li>

    <li class="nav-item">

      <a routerLink="signup" class="nav-link">Signup</a>

    </li>

  </ul>

</nav>

<router-outlet></router-outlet>

**Dashboard.component.spec.ts:**

import { ComponentFixture, TestBed } from '@angular/core/testing';

import { DashboardComponent } from './dashboard.component';

describe('DashboardComponent', () => {

  let component: DashboardComponent;

  let fixture: ComponentFixture<DashboardComponent>;

  beforeEach(async () => {

    await TestBed.configureTestingModule({

      declarations: [ DashboardComponent ]

    })

    .compileComponents();

    fixture = TestBed.createComponent(DashboardComponent);

    component = fixture.componentInstance;

    fixture.detectChanges();

  });

  it('should create', () => {

    expect(component).toBeTruthy();

  });

});

**Signin.component.spec.ts:**

import { ComponentFixture, TestBed } from '@angular/core/testing';

import { SigninComponent } from './signin.component';

describe('SigninComponent', () => {

  let component: SigninComponent;

  let fixture: ComponentFixture<SigninComponent>;

  beforeEach(async () => {

    await TestBed.configureTestingModule({

      declarations: [ SigninComponent ]

    })

    .compileComponents();

    fixture = TestBed.createComponent(SigninComponent);

    component = fixture.componentInstance;

    fixture.detectChanges();

  });

  it('should create', () => {

    expect(component).toBeTruthy();

  });

});

**Signup.component.spec.ts:**

import { ComponentFixture, TestBed } from '@angular/core/testing';

import { SignupComponent } from './signup.component';

describe('SignupComponent', () => {

  let component: SignupComponent;

  let fixture: ComponentFixture<SignupComponent>;

  beforeEach(async () => {

    await TestBed.configureTestingModule({

      declarations: [ SignupComponent ]

    })

    .compileComponents();

    fixture = TestBed.createComponent(SignupComponent);

    component = fixture.componentInstance;

    fixture.detectChanges();

  });

  it('should create', () => {

    expect(component).toBeTruthy();

  });

});

**App.component.spec.ts:**

import { TestBed } from '@angular/core/testing';

import { RouterTestingModule } from '@angular/router/testing';

import { AppComponent } from './app.component';

describe('AppComponent', () => {

  beforeEach(async () => {

    await TestBed.configureTestingModule({

      imports: [

        RouterTestingModule

      ],

      declarations: [

        AppComponent

      ],

    }).compileComponents();

  });

  it('should create the app', () => {

    const fixture = TestBed.createComponent(AppComponent);

    const app = fixture.componentInstance;

    expect(app).toBeTruthy();

  });

  it(`should have as title 'MyPortfolio'`, () => {

    const fixture = TestBed.createComponent(AppComponent);

    const app = fixture.componentInstance;

    expect(app.title).toEqual('MyPortfolio');

  });

  it('should render title', () => {

    const fixture = TestBed.createComponent(AppComponent);

    fixture.detectChanges();

    const compiled = fixture.nativeElement as HTMLElement;

    expect(compiled.querySelector('.content span')?.textContent).toContain('MyPortfolio app is running!');

  });

});

**App.module.ts:**

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

import { BrowserModule } from '@angular/platform-browser';

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

import { AppRoutingModule } from './app-routing.module';

import { AppComponent } from './app.component';

import { SigninComponent } from './signin/signin.component';

import { SignupComponent } from './signup/signup.component';

import { DashboardComponent } from './dashboard/dashboard.component';

@NgModule({

  declarations: [

    AppComponent,

    SigninComponent,

    SignupComponent,

    DashboardComponent

  ],

  imports: [

    BrowserModule,

    AppRoutingModule,

    ReactiveFormsModule,

    FormsModule

  ],

  providers: [],

  bootstrap: [AppComponent]

})

export class AppModule { }

**app.component.ts:**

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

@Component({

  selector: 'app-root',

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

  styleUrls: ['./app.component.css']

})

export class AppComponent {

  title = 'MyPortfolio';

}

**Signup.component.ts:**

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

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

@Component({

  selector: 'app-signup',

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

  styleUrls: ['./signup.component.css']

})

export class SignupComponent implements OnInit {

  registerForm: FormGroup;

  submitted:boolean=false;

  constructor(private builder:FormBuilder) { }

  ngOnInit(): void {

    this.registerForm= this.builder.group(

      {

        firstName:["",Validators.required],

        lastName:["",Validators.required],

        email:["",[Validators.required,Validators.email]],

        password:["",[Validators.required,Validators.minLength(8)]]

      }

    );

  }

   OnSubmit(){

    this.submitted=true;

    if(this.registerForm.invalid)

      return;

    else

      alert("User Registered");

  }

  get f(){

    return this.registerForm.controls;

  }

}

**Signin.component.ts:**

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

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

@Component({

  selector: 'app-signin',

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

  styleUrls: ['./signin.component.css']

})

export class SigninComponent implements OnInit {

   registerForm: FormGroup;

  submitted:boolean=false;

  constructor(private builder:FormBuilder) { }

  ngOnInit(): void {

    this.registerForm= this.builder.group(

      {

        email:["",[Validators.required,Validators.email]],

        password:["",[Validators.required,Validators.minLength(8)]]

      }

    );

  }

  OnSubmit(){

    this.submitted=true;

    if(this.registerForm.invalid)

      return;

    else

      console.log("Form Submitted Successfully");

  }

  get f(){

    return this.registerForm.controls;

  }

}

**Dashboard.component.ts:**

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

@Component({

  selector: 'app-dashboard',

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

  styleUrls: ['./dashboard.component.css']

})

export class DashboardComponent implements OnInit {

  constructor() { }

  ngOnInit(): void {

  }

}

**App-routing.module.ts:**

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

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

import { DashboardComponent } from './dashboard/dashboard.component';

import { SigninComponent } from './signin/signin.component';

import { SignupComponent } from './signup/signup.component';

const routes: Routes = [

  {"path":"signin",component:SigninComponent},

  {"path":"signup",component:SignupComponent},

  {"path":"dashboard",component:DashboardComponent},

];

@NgModule({

  imports: [RouterModule.forRoot(routes)],

  exports: [RouterModule]

})

export class AppRoutingModule { }