**Angular-T08-HOL001\_TestCase [Reactive Form Testing]**

In this hands on we will understand how to perform unit testing of Angular applications using Jasmine and Karma on Reactive Form.

Tasks:

1. Create a Component called TraineeComponent. Add the below code in the component class:

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

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

@Component({

  selector: 'app-trainee',

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

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

})

export class TraineeComponent implements OnInit {

  title="Trainee Registration Form";

  constructor() { }

  studentForm;

  message:string;

  ngOnInit() {

    this.studentForm=new FormGroup(

      {

        studentName:new FormControl('',[

          Validators.required,

        ]),

        emailid:new FormControl('',[

          Validators.required,Validators.email

        ])

      });

  }

  public SubmitForm()

  {

    if(this.studentForm.valid)

    {

      this.message="Form Submitted Successfully";

    }

  }

}

1. On the trainee.component.html file create a form with student name and email id fields. Also both the input fields should contain proper validation information.
2. Create a submit button which should be disabled if the form is invalid. If the form is valid (i.e. the input fields are valid); a success message should be displayed on button click.

**Sample code of html file is given for your reference:**

<div class="container-fluid">

<h1>{{title}}</h1>

<br/>

    <div class="form-group">

        <span id="success-msg" \*ngIf="message!==''"  class="alert alert-success" >{{message}}</span>

    </div>

<br/>

<form [formGroup]="studentForm">

<div class="form-group">

    <label for="StudentName">

        Student Name

    </label>

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

    placeholder="Enter Name" class="form-control" formControlName="studentName">

    <div class="text-danger"

    \*ngIf="studentForm.controls['studentName'].touched &&

    studentForm.controls['studentName'].invalid">

    <div \*ngIf="studentForm.controls['studentName'].errors.required">

        \* Student Name Required

    </div>

    <div \*ngIf="studentForm.controls['studentName'].errors.minlength">

        \* Student Name should be 3 characters minimum long

    </div>

</div>

</div>

<br/>

<div class="form-group">

    <label for="emailid">

        Email Id:

    </label>

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

    placeholder="Enter Email Id" class="form-control" formControlName="emailid">

    <div class="text-danger"

    \*ngIf="studentForm.controls['emailid'].touched &&

    studentForm.controls['emailid'].invalid">

    <div \*ngIf="studentForm.controls['emailid'].errors.required">

        \* Email Id Required

    </div>

    <div \*ngIf="studentForm.controls['emailid'].errors.email">

        \* Invalid Email Id

    </div>

</div>

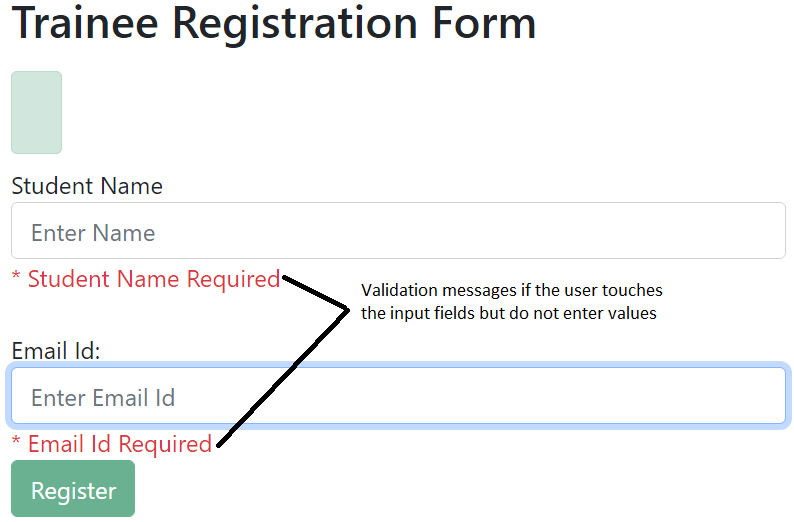
</div>

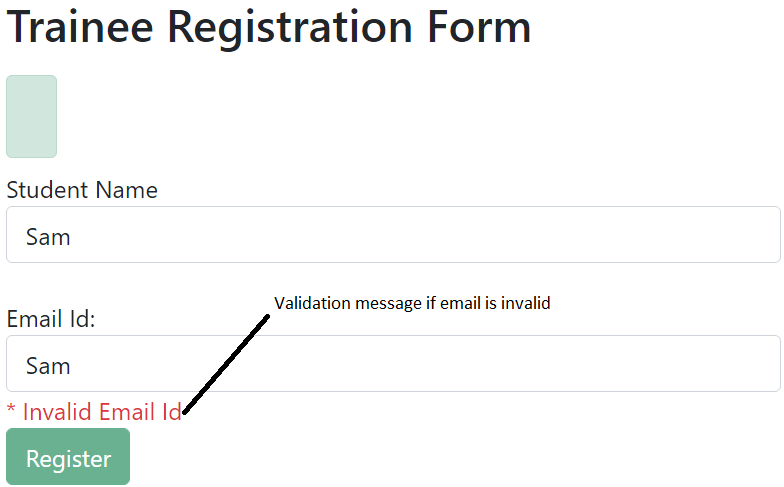
<input type="submit" id="submitBtn" [disabled]="studentForm.invalid" (click)="SubmitForm()" value="Register" class="btn btn-success">

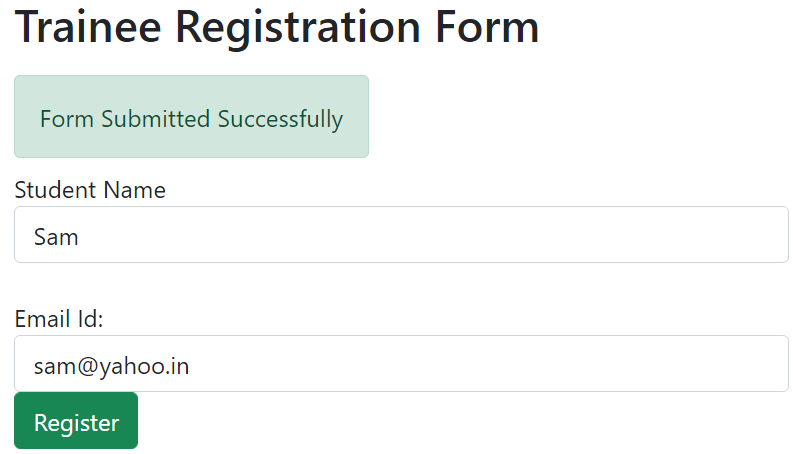
</form>

</div>

**Sample output is given below:**







1. Use xdescribe() for app.component.spec.ts
2. Check whether the file trainee.component.spec.ts is present in src/app/trainee.
3. Check if the describe () block for TraineeComponent is present.
4. Check whether a beforeEach() block that instantiates a new instance of TraineeComponent for each spec is present. Also under beforeEach() block add imports:[FormsModule,ReactiveFormsModule] as we are working with reactive form.
5. Check if there is a test case to check whether the TraineeComponent is created is present.
6. Create a test case to check if the title with value “Trainee Registration Form” is present.
7. Create a test case to check if the title is displayed in a h1 tag.
8. Create a test case to checkout if the studentname input field and studentForm validation fails if the input field is touched but no data is entered by user.
9. Create a test case to checkout if the emailid input field and studentForm validation fails if the input field is touched but no data is entered by user.
10. Create a test case to checkout if the emailid input field and studentForm validation fails if invalid email id is entered by user.
11. Create a test case to check if success message is displayed in the span with id=”success-msg” if the input fields are valid and studentForm is valid.

**Sample Test Case Code is given for your reference.**

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

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

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

import { TraineeComponent } from './trainee.component';

describe('TraineeComponent', () => {

  let component: TraineeComponent;

  let fixture: ComponentFixture<TraineeComponent>;

  beforeEach(async(() => {

    TestBed.configureTestingModule({

      declarations: [ TraineeComponent ],

      imports:[FormsModule,ReactiveFormsModule]

    })

    .compileComponents();

  }));

  beforeEach(() => {

    fixture = TestBed.createComponent(TraineeComponent);

    component = fixture.componentInstance;

    fixture.detectChanges();

  });

  it('should create Trainee Component', () => {

    expect(component).toBeTruthy();

  });

  it(`should have as title 'Trainee Registration Form'`, () => {

    const fixture = TestBed.createComponent(TraineeComponent);

    const app = fixture.debugElement.componentInstance;

    expect(app.title).toEqual('Trainee Registration Form');

  });

  it('should render title in a h1 tag', () => {

    const fixture = TestBed.createComponent(TraineeComponent);

    fixture.detectChanges();

    const compiled = fixture.debugElement.nativeElement;

    expect(compiled.querySelector('h1').textContent).toContain('Trainee Registration Form');

  });

  it('Trainee Name - Check | No Data Supplied But Input Field is Touched',()=>

  {

    let studentname=component.studentForm.controls['studentName'];

    studentname.setValue('');

    expect(studentname.touched).toBeFalsy();

    expect(studentname.valid).toBeFalsy();

    expect(studentname.errors['required']).toBeTruthy();

    expect(component.studentForm.valid).toBeFalsy();

  });

  it('Email Id - Check | No Data Supplied But Input Field is Touched',()=>

  {

    let emailid=component.studentForm.controls['emailid'];

    emailid.setValue('');

    expect(emailid.touched).toBeFalsy();

    expect(emailid.valid).toBeFalsy();

    expect(emailid.errors['required']).toBeTruthy();

    expect(component.studentForm.valid).toBeFalsy();

  });

  it('Email Id - Check | Invalid Email Id Supplied',()=>

  {

    let emailid=component.studentForm.controls['emailid'];

    emailid.setValue('abc');

    expect(emailid.errors['email']).toBeTruthy();

    expect(emailid.valid).toBeFalsy();

    expect(component.studentForm.valid).toBeFalsy();

  });

  it('When Both Trainee Name and Email Is Valid, Form Should be submitted Successfully',()=>{

      let studentname=component.studentForm.controls['studentName'];

      let emailid=component.studentForm.controls['emailid'];

      studentname.setValue('Sam');

      emailid.setValue('sam@gmail.com');

      component.SubmitForm();

      fixture.detectChanges();

      let successMsg=fixture.debugElement.query(By.css('#success-msg'));

      expect(successMsg.nativeElement.innerText).toBe(component.message);

    });

});

1. Run the command ng test in cli and click the debug button of Karma window in browser to see the jasmine test results.

**Sample output is given for your reference:**

