Online Test Application

Source Code

Programming code

<nav class="navbar navbar-light bg-primary">

    <div class="container-fluid">

      <a class="navbar-brand text-bold" href="#">

        <img src="https://encrypted-tbn0.gstatic.com/images?q=tbn:ANd9GcQZFg7WEQMvbdHg-4g6xXVUQaNrt628ac0PeQ&usqp=CAU" alt="logo" width="90" height="70">

      </a>

    </div>

  </nav>

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

import { HeaderComponent } from './header.component';

describe('HeaderComponent', () => {

  let component: HeaderComponent;

  let fixture: ComponentFixture<HeaderComponent>;

  beforeEach(async () => {

    await TestBed.configureTestingModule({

      declarations: [ HeaderComponent ]

    })

    .compileComponents();

  });

  beforeEach(() => {

    fixture = TestBed.createComponent(HeaderComponent);

    component = fixture.componentInstance;

    fixture.detectChanges();

  });

  it('should create', () => {

    expect(component).toBeTruthy();

  });

});

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

@Component({

  selector: 'app-header',

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

  styleUrls: ['./header.component.scss']

})

export class HeaderComponent implements OnInit {

  constructor() { }

  ngOnInit(): void {

  }

}

<div class="container mt-5">

<div class="card">

    <div class="d-flex justify-content-between p-3">

        <div class="image">

            <img src="https://encrypted-tbn0.gstatic.com/images?q=tbn:ANd9GcQZFg7WEQMvbdHg-4g6xXVUQaNrt628ac0PeQ&usqp=CAU" alt="logo" width="90" height="70">

        </div>

        <div class="quiz-header">

           <h4 style="font-family: cursive;">Angular & TypeScript Quiz!</h4>

           <span style="font-style: italic;">Welcome {{name}}</span>

        </div>

    </div>

<ng-container \*ngIf="!isQuizCompleted">

    <div>

        <div class="d-flex justify-content-around py-3">

            <div class="score">

             <h5>{{points}} points</h5>

            </div>

            <div class="question-remain">

              <span style="font-style: italic;"><h5>questions {{currentQuestion+1}} of {{questionList.length}}</h5></span>

            </div>

            <div class="timer">

              <h5> {{counter}} sec </h5>

            </div>

        </div>

    </div>

    <div class="question">

    <div class="card">

        <h3>{{questionList[currentQuestion]?.questionText}}</h3>

    </div>

    </div>

    <div class="option">

    <ol \*ngFor="let option of questionList[currentQuestion]?.options">

        <li (click)="answer(currentQuestion+1,option)">

            <div appChangeBg [isCorrect]="option.correct" class="card">

              {{option.text}}

            </div>

    </ol>

    </div>

    <div class="d-flex justify-content-between">

        <button [disabled]="currentQuestion===0" class="btn"(click)="previousQuestion()"><i class="fa text-primary fa-chevron-left fa-3x" aria-hidden="true"></i></button>

        <button class="btn" (click)="resetQuiz()"><i class="fa fa-refresh text-primary fa-3x" aria-hidden="true"></i></button>

        <button class="btn" (click)="nextQuestion()"><i class="fa text-primary fa-chevron-right fa-3x" aria-hidden="true"></i></button>

    </div>

</ng-container>

<ng-container \*ngIf="isQuizCompleted">

    <div class="row d flex justify-content-between">

        <img src="https://t3.ftcdn.net/jpg/02/80/01/64/360\_F\_280016442\_I5DcWCRT7JTr5Ut86a9VvqNoOfDt854G.jpg" alt="">

        <div class="result text-center col-md-6 col-sm-12">

            <h3>Congratulation!! <br> you have completed the quiz. <br> Below is the Result:</h3>

            <p>Toltal Question Attement :{{questionList.length}}</p>

            <p>Toltal Correct Answered :{{correctAnswer}}</p>

            <p>Toltal Wrong Answer :{{inCorrectAnswer}}</p>

            <p>your Score : {{points}} points</p>

        </div>

    </div>

</ng-container>

</div>

</div>

.card

{

    max-width: 800px;

    margin: 0 auto;

    padding: 10px;

}

li{

    list-style-type: none;

    cursor: pointer;

    margin: 10px 0;

}

li  .card:hover{

    border: 1px solid blue;

}

ol{

    padding: 0;

}

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

import { QuestionComponent } from './question.component';

describe('QuestionComponent', () => {

  let component: QuestionComponent;

  let fixture: ComponentFixture<QuestionComponent>;

  beforeEach(async () => {

    await TestBed.configureTestingModule({

      declarations: [ QuestionComponent ]

    })

    .compileComponents();

  });

  beforeEach(() => {

    fixture = TestBed.createComponent(QuestionComponent);

    component = fixture.componentInstance;

    fixture.detectChanges();

  });

  it('should create', () => {

    expect(component).toBeTruthy();

  });

});

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

import { interval } from 'rxjs';

import { QuestionService } from '../service/question.service';

@Component({

  selector: 'app-question',

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

  styleUrls: ['./question.component.scss']

})

export class QuestionComponent implements OnInit {

  public name : string="";

  public questionList : any=[];

  public currentQuestion:number=0;

  public points:number=0;

  counter=60;

  correctAnswer:number=0;

  inCorrectAnswer:number=0;

  interval$:any;

  isQuizCompleted : boolean=false;

  constructor(private questionService : QuestionService) { }

  ngOnInit(): void {

    this.name=localStorage.getItem("name")!;

    this.getAllQuestions();

    this.startCounter();

  }

 getAllQuestions(){

this.questionService.getQuestionJson()

.subscribe(res=>{

  this.questionList=res.questions;

})

 }

 nextQuestion(){

 this.currentQuestion++;

 }

 previousQuestion(){

this.currentQuestion--;

 }

 answer(currentQno:number,option:any){

   if (currentQno===this.questionList.length) {

     this.isQuizCompleted=true;

     this.stopCounter();

   }

if(option.correct){

  this.points+=10;

  this.correctAnswer++;

  setTimeout(()=> {

  this.currentQuestion++;

  this.resetCounter();

 },1000);

  this.currentQuestion++;

  this.resetCounter();

}else{

  setTimeout(()=>{

    this.currentQuestion++;

 this.inCorrectAnswer++;

 this.resetCounter();

  },1000);

 this.points-=10;

}

}

startCounter(){

 this.interval$=interval(1000)

 .subscribe(val=>{

   this.counter--;

   if (this.counter==0) {

     this.currentQuestion++;

     this.counter=60;

     this.points-=10;

   }

  });

  setTimeout(()=>{

  this.interval$.unsubscribe();

  },600000);

}

stopCounter(){

this.interval$.unsubscribe();

this.counter=0;

}

resetCounter(){

this.stopCounter();

this.counter=60;

this.startCounter();

}

resetQuiz(){

  this.resetCounter();

  this.getAllQuestions();

  this.points=0;

  this.counter=60;

  this.currentQuestion=0;

}

}

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

import { QuestionService } from './question.service';

describe('QuestionService', () => {

  let service: QuestionService;

  beforeEach(() => {

    TestBed.configureTestingModule({});

    service = TestBed.inject(QuestionService);

  });

  it('should be created', () => {

    expect(service).toBeTruthy();

  });

});

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

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

@Injectable({

  providedIn: 'root'

})

export class QuestionService {

  constructor(private http: HttpClient) { }

  getQuestionJson(){

    return this.http.get<any>("assets/questions.json");

  }

}

<div class="container bg-light py-5">

<h1 class="display-5 fw-bold">Welcome to  Quiz!!!</h1>

<p class="col-md-8 fs-4">

    This Quiz will contain total 10 questions.

    Each question will contain 10 ponints.

</p>

<h4>Rules</h4>

<ol>

    <li>Correct Question gives vou 10 points.</li>

    <li>InCorrect Question gives vou -10 points.</li>

    <li>You will have 60 sec to answer each question</li>

    <li>Refershing the page will reset the Quiz</li>

</ol>

<h1 style="font-family: cursive; text-align: center;">All the best!!!</h1>

<div class="name col-md-4 my-3">

    <label for="">Enter your name :</label>

    <input #name type="text" class="form-control" placeholder="FULL NAME">

</div>

<button class="btn btn-primary btn-lg" routerLink="/question" (click)="startQuiz()">Start quiz</button>

</div>

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

import { WelcomeComponent } from './welcome.component';

describe('WelcomeComponent', () => {

  let component: WelcomeComponent;

  let fixture: ComponentFixture<WelcomeComponent>;

  beforeEach(async () => {

    await TestBed.configureTestingModule({

      declarations: [ WelcomeComponent ]

    })

    .compileComponents();

  });

  beforeEach(() => {

    fixture = TestBed.createComponent(WelcomeComponent);

    component = fixture.componentInstance;

    fixture.detectChanges();

  });

  it('should create', () => {

    expect(component).toBeTruthy();

  });

});

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

@Component({

  selector: 'app-welcome',

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

  styleUrls: ['./welcome.component.scss']

})

export class WelcomeComponent implements OnInit {

  @ViewChild('name') nameKey! : ElementRef;

  constructor() { }

  ngOnInit(): void {

  }

startQuiz(){

  localStorage.setItem("name",this.nameKey.nativeElement.value);

}

}

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

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

import { QuestionComponent } from './question/question.component';

import { WelcomeComponent } from './welcome/welcome.component';

const routes: Routes = [

  {path:'',redirectTo:'welcome',pathMatch:"full"},

  {path:"welcome",component:WelcomeComponent},

  {path:"question",component:QuestionComponent}

];

@NgModule({

  imports: [RouterModule.forRoot(routes)],

  exports: [RouterModule]

})

export class AppRoutingModule { }

<app-header></app-header>

<router-outlet></router-outlet>

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 'quizapp\_angular'`, () => {

    const fixture = TestBed.createComponent(AppComponent);

    const app = fixture.componentInstance;

    expect(app.title).toEqual('quizapp\_angular');

  });

  it('should render title', () => {

    const fixture = TestBed.createComponent(AppComponent);

    fixture.detectChanges();

    const compiled = fixture.nativeElement as HTMLElement;

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

  });

});

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

@Component({

  selector: 'app-root',

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

  styleUrls: ['./app.component.scss']

})

export class AppComponent {

  title = 'quizapp\_angular';

}

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

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

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

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

import { WelcomeComponent } from './welcome/welcome.component';

import { QuestionComponent } from './question/question.component';

import { HeaderComponent } from './header/header.component';

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

import { ChangeBgDirective } from './change-bg.directive';

@NgModule({

  declarations: [

    AppComponent,

    WelcomeComponent,

    QuestionComponent,

    HeaderComponent,

    ChangeBgDirective

  ],

  imports: [

    BrowserModule,

    AppRoutingModule,

    HttpClientModule

  ],

  providers: [],

  bootstrap: [AppComponent]

})

export class AppModule { }

import { Directive, ElementRef, HostListener, Input, Renderer2 } from '@angular/core';

@Directive({

  selector: '[appChangeBg]'

})

export class ChangeBgDirective {

  @Input() isCorrect : Boolean=false;

  constructor(private el: ElementRef,private render: Renderer2) { }

 @HostListener('click') answer(){

   if (this.isCorrect){

     this.render.setStyle(this.el.nativeElement,'background','green');

     this.render.setStyle(this.el.nativeElement,'color','#fff');

     this.render.setStyle(this.el.nativeElement,'border','2px solid grey');

   }else{

    this.render.setStyle(this.el.nativeElement,'background','red');

    this.render.setStyle(this.el.nativeElement,'color','#fff');

     this.render.setStyle(this.el.nativeElement,'border','2px solid grey');

   }

 }

}

<!doctype html>

<html lang="en">

<head>

  <meta charset="utf-8">

  <title>QuizappAngular</title>

  <base href="/">

  <meta name="viewport" content="width=device-width, initial-scale=1">

  <link rel="icon" type="image/x-icon" href="favicon.ico">

  <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.1.3/dist/css/bootstrap.min.css" rel="stylesheet" integrity="sha384-1BmE4kWBq78iYhFldvKuhfTAU6auU8tT94WrHftjDbrCEXSU1oBoqyl2QvZ6jIW3" crossorigin="anonymous">

  <link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/font-awesome/4.7.0/css/font-awesome.min.css">

</head>

<body>

  <script src="https://cdn.jsdelivr.net/npm/bootstrap@5.1.3/dist/js/bootstrap.bundle.min.js" integrity="sha384-ka7Sk0Gln4gmtz2MlQnikT1wXgYsOg+OMhuP+IlRH9sENBO0LRn5q+8nbTov4+1p" crossorigin="anonymous"></script>

  <app-root></app-root>

</body>

</html>

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

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

import { AppModule } from './app/app.module';

import { environment } from './environments/environment';

if (environment.production) {

  enableProdMode();

}

platformBrowserDynamic().bootstrapModule(AppModule)

  .catch(err => console.error(err));

/\*\*

 \* This file includes polyfills needed by Angular and is loaded before the app.

 \* You can add your own extra polyfills to this file.

 \*

 \* This file is divided into 2 sections:

 \*   1. Browser polyfills. These are applied before loading ZoneJS and are sorted by browsers.

 \*   2. Application imports. Files imported after ZoneJS that should be loaded before your main

 \*      file.

 \*

 \* The current setup is for so-called "evergreen" browsers; the last versions of browsers that

 \* automatically update themselves. This includes recent versions of Safari, Chrome (including

 \* Opera), Edge on the desktop, and iOS and Chrome on mobile.

 \*

 \* Learn more in https://angular.io/guide/browser-support

 \*/

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

 \* BROWSER POLYFILLS

 \*/

/\*\*

 \* By default, zone.js will patch all possible macroTask and DomEvents

 \* user can disable parts of macroTask/DomEvents patch by setting following flags

 \* because those flags need to be set before `zone.js` being loaded, and webpack

 \* will put import in the top of bundle, so user need to create a separate file

 \* in this directory (for example: zone-flags.ts), and put the following flags

 \* into that file, and then add the following code before importing zone.js.

 \* import './zone-flags';

 \*

 \* The flags allowed in zone-flags.ts are listed here.

 \*

 \* The following flags will work for all browsers.

 \*

 \* (window as any).\_\_Zone\_disable\_requestAnimationFrame = true; // disable patch requestAnimationFrame

 \* (window as any).\_\_Zone\_disable\_on\_property = true; // disable patch onProperty such as onclick

 \* (window as any).\_\_zone\_symbol\_\_UNPATCHED\_EVENTS = ['scroll', 'mousemove']; // disable patch specified eventNames

 \*

 \*  in IE/Edge developer tools, the addEventListener will also be wrapped by zone.js

 \*  with the following flag, it will bypass `zone.js` patch for IE/Edge

 \*

 \*  (window as any).\_\_Zone\_enable\_cross\_context\_check = true;

 \*

 \*/

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

 \* Zone JS is required by default for Angular itself.

 \*/

import 'zone.js';  // Included with Angular CLI.

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

 \* APPLICATION IMPORTS

 \*/

// This file is required by karma.conf.js and loads recursively all the .spec and framework files

import 'zone.js/testing';

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

import {

  BrowserDynamicTestingModule,

  platformBrowserDynamicTesting

} from '@angular/platform-browser-dynamic/testing';

declare const require: {

  context(path: string, deep?: boolean, filter?: RegExp): {

    <T>(id: string): T;

    keys(): string[];

  };

};

// First, initialize the Angular testing environment.

getTestBed().initTestEnvironment(

  BrowserDynamicTestingModule,

  platformBrowserDynamicTesting(),

);

// Then we find all the tests.

const context = require.context('./', true, /\.spec\.ts$/);

// And load the modules.

context.keys().map(context);