Header component:

```
<nav class="navbar navbar-dark bg-warning">
  <div class="container-fluid">
    <a class="navbar-brand text-bold"href="#">
      <img
        src="https://img.icons8.com/color/96/000000/angularjs.png"
        alt="logo"
        width="50"
        height=""
      />
      <span class="text-uppercase text-body">ngular Quiz</span>
    </a>
  </div>
</nav>
```

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

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

describe('HeaderComponent', () => {
  letcomponent:HeaderComponent;
  letfixture:ComponentFixture<HeaderComponent>;

  beforeEach(async () => {
    awaitTestBed.configureTestingModule({
      declarations: [ HeaderComponent ]
    })
    .compileComponents();

    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']
})
exportclassHeaderComponentimplementsOnInit {

  constructor() { }

  ngOnInit():void {
  }

}
```

Question Component:

```
<div class="container mt-5">
  <div class="card">
    <div class="d-flex justify-content-between p-3">
      <div class="image">
        <img
          src="https://img.icons8.com/color/96/000000/angularjs.png"
          alt="logo"
          width="90"
          height=""
        />
      </div>
      <div class="quiz-header">
        <h4 style="font-family: cursive">
          <span style="color: rgb(183, 7, 92)">Simplilearn</span> Online Quiz
        </h4>
        <span style="font-family: cursive"
          ><h4 class="nameColor">welcome {{ name }}</h4></span
        >
      </div>
    </div>
    <ng-container *ngIf="!isQuizCompleted">
      <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"
          >Question {{ currentQuestion + 1 }} of {{ questionList.length }}
        </span>
      </div>
      <div class="timer">
        <h5>{{ counter }} sec &#8986;</h5>
      </div>
    </div>
    <div class="progress mb-3">
      <div
        class="progress-bar progress-bar-striped bg-success"
        role="progressbar"
        [ngStyle]="{ width: progress + '%' }"
        aria-valuenow="25"
        aria-valuemin="0"
        aria-valuemax="100"
      ></div>
    </div>
    <div class="question">
      <div class="card">
        <h3>{{ questionList[currentQuestion]?.questionText }}</h3>
      </div>
    </div>
    <div class="options">
      <ol *ngFor="let option of questionList[currentQuestion]?.options">
        <li (click)="answer(currentQuestion + 1, option)">
          <divappChangeBg [isCorrect]="option.correct" class="card">
            {{ option.text }}
          </div>
        </li>
      </ol>
    </div>
    <div class="d-flex justify-content-between">
      <button
        [disabled]="currentQuestion === 0"
        class="btn"
        (click)="previousQuestion()"
      >
        <i
          class="fa-solid fa-chevron-left fa-3x fa text-primary"
          aria-hidden="true"
        ></i>
      </button>
```

```
        <button class="btn" (click)="resetQuiz()">
          <i
            class="fa-solid fa fa-refresh fa text-primary fa-3x"
            area-hidden="true"
          ></i>
        </button>
        <button
          [disabled]="currentQuestion === 8"
          class="btn"
          (click)="nextQuestion()"
        >
          <i
            class="fa-solid fa-chevron-right fa-3x fa text-primary"
            aria-hidden="true"
          ></i>
        </button>
      </div>
    </ng-container>

    <ng-container *ngIf="isQuizCompleted">
      <div class="row d-flex justify-content-between">
        <img
          style="width: 50%"
          class="img-fluid col-sm-12 auto"
          src="https://c.tenor.com/MmYFrzenjVgAAAAC/boy-good-job.gif"
          alt=""
        />
        <div class="result text-center col-md-6 col-sm-12">
          <h3>
            Congratulations!!!!!<br/>You have completed the quiz.....
            <br/>Below is your result:
          </h3>
          <p>Total question Attempted : {{ questionList.length }}</p>
          <p>Total Correct Answered :{{ correctAnswer }}</p>
          <p>Total Wrong Answered : {{ inCorrectAnswer }}</p>
          <p>Your Score : {{ points }} points</p>
        </div>
      </div>
    </ng-container>
  </div>
</div>
```

```
.card{
    max-width:800px;
    margin:0auto;
    padding:10px;
}
li{
    list-style-type:none;
    cursor:pointer;
    margin:10px0;

}
li.card:hover{
 border:1pxsolidrgb(99, 136, 51);
 background-color: lightcyan;
}
ol{
    padding:0;
}
.nameColor{
    padding:10px;
    color:rgb(65, 40, 3);
    background-color: rgb(118, 230, 118);
    border-radius:20px;
}
.nameColor:hover{
    color:black;
    background-color: bisque;

}
```

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

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

describe('QuestionComponent', () => {
  letcomponent:QuestionComponent;
  letfixture:ComponentFixture<QuestionComponent>;

  beforeEach(async () => {
    awaitTestBed.configureTestingModule({
      declarations: [ QuestionComponent ]
    })
```

```
    .compileComponents();

    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'],
})
exportclassQuestionComponentimplementsOnInit {
  publicname:String='';
  publicquestionList:any= [];
  publiccurrentQuestion:number=0;
  publicpoints:number=0;
  counter=60;
  correctAnswer:number=0;
  inCorrectAnswer:number=0;
  interval$:any;
  progress:string='0';
  isQuizCompleted:Boolean=false;

  constructor(privatequestionService: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();
        this.getProgressPercent();
      }, 1000);
      // this.points = this.points + 10;
    } else{
    setTimeout(() => {
        this.currentQuestion++;
        this.inCorrectAnswer++;
        this.resetCounter();
        this.getProgressPercent();
    }, 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$.unsubscribe();
    }, 6000000);
  }
  stopCounter() {
    this.interval$.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;
    this.progress='0';
  }
  getProgressPercent() {
    this.progress= (
      (this.currentQuestion/this.questionList.length) *
      100
    ).toString();
    returnthis.progress;
  }
}
```

Service Component:

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

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

describe('QuestionService', () => {
  letservice: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'
})
exportclassQuestionService {

  constructor(privatehttp:HttpClient) { }
  getQuestionJson(){
  returnthis.http.get<any>("assets/questions.json");
  }
}
```

Welcome Component:

```
<div class="container bg-light py-5">
  <h1 class="dislay-5 fw-bold">Welcome to Online Test Application</h1>
  <p class="col-md-8 fs-4">
    This quiz will contains total 9 questions.Each question holds 10 points
  </p>
  <h4>Rules:</h4>
  <ol>
    <li>Correct Question gives you 10 points.</li>
    <li>Incorrect question gives you -10 points.</li>
    <li>You will have 60 sec to answer each question.</li>
    <li>Refreshing the page will reset the Quit.</li>
    <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"/>
    </div>
    <button class="btnbtn-primary btn-lg hover"routerLink="/question"
(click)="startQuiz()">
      Start the Quiz
    </button>

  </ol>
</div>

.hover:hover{
background-color: blanchedalmond;
color: black;
}
```

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

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

describe('WelcomeComponent', () => {
  letcomponent:WelcomeComponent;
  letfixture:ComponentFixture<WelcomeComponent>;

  beforeEach(async () => {
    awaitTestBed.configureTestingModule({
      declarations: [ WelcomeComponent ]
    })
    .compileComponents();

    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'],
})
exportclassWelcomeComponentimplementsOnInit {
  @ViewChild('name') nameKey!:ElementRef;
  constructor() {}

  ngOnInit():void {

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

}
```

App (root) Componenet:

```
import{ NgModule } from'@angular/core';
import{ RouterModule, Routes } from'@angular/router';
import{ QuestionComponent } from'./question/question.component';
import{ WelcomeComponent } from'./welcome/welcome.component';

constroutes:Routes= [
  {path:'',redirectTo:'welcome',pathMatch:"full"},
  { path: "welcome", component: WelcomeComponent },
  {path:"question",component:QuestionComponent}
];

@NgModule({
  imports: [RouterModule.forRoot(routes)],
  exports: [RouterModule]
})
exportclassAppRoutingModule{ }
```

```
<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 () => {
    awaitTestBed.configureTestingModule({
      imports: [
        RouterTestingModule
      ],
      declarations: [
        AppComponent
      ],
    }).compileComponents();
  });

  it('should create the app', () => {
    constfixture=TestBed.createComponent(AppComponent);
    constapp=fixture.componentInstance;
    expect(app).toBeTruthy();
  });

  it(`should have as title 'OnlineTestApplication'`, () => {
    constfixture=TestBed.createComponent(AppComponent);
    constapp=fixture.componentInstance;
    expect(app.title).toEqual('OnlineTestApplication');
  });

  it('should render title', () => {
    constfixture=TestBed.createComponent(AppComponent);
    fixture.detectChanges();
    constcompiled=fixture.nativeElementasHTMLElement;
    expect(compiled.querySelector('.content
span')?.textContent).toContain('OnlineTestApplication app is running!');
  });
});

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

```
@Component({
  selector: 'app-root',
  templateUrl: './app.component.html',
  styleUrls: ['./app.component.scss']
})
exportclassAppComponent {
  title='OnlineTestApplication';
}
```

```
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{ 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]
})
exportclassAppModule{ }
```

Directive:

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

@Directive({
  selector: '[appChangeBg]',
})
exportclassChangeBgDirective {
  @Input() isCorrect:Boolean=false;
  constructor(privateel:ElementRef, privaterender: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');

    }
  }
}
```

Json:

```
{
  "questions": [
    {
      "questionText": "Which of the following does TypeScript use to specify
types?",
      "options": [
        {
          "text": ":",
          "correct": true
        },
        {
          "text": ";"
        },
        {
          "text": "!"
        },
        {
          "text": "&"
        }
      ],
      "explanation": "TS uses a colon (:) to separate the property name from the
property type"
    },

    {
      "questionText": "Which of the following is NOT a type used in TypeScript?",
      "options": [
        {
          "text": "number"
        },
        {
          "text": "string"
        },
        {
          "text": "boolean"
        },
        {
          "text": "enum",
          "correct": true
        }
      ],
```

```
      "explanation": "enum is not used as a type in TypeScript"
    },

    {
      "questionText": "How can we specify properties and methods for an object in
TypeScript?",
      "options": [
        {
          "text": "Use classes."
        },
        {
          "text": "Use interfaces.",
          "correct": true
        },
        {
          "text": "Use enums."
        },
        {
          "text": "Use async/await."
        }
      ],
      "explanation": "interfaces are typically used to list the properties and
methods for an object"
    },
    {
      "questionText": "How else can Array<number> be written in TypeScript?",
      "options": [
        {
          "text": "@number"
        },
        {
          "text": "number[]",
          "correct": true
        },
        {
          "text": "number!"
        },
        {
          "text": "number?"
        }
      ],
      "explanation": "number[] is another way of writing Array<number> in
TypeScript"
    },
    {
```

```
      "questionText": "In which of these does a class take parameters?",
      "options": [
        {
          "text": "constructor",
          "correct": true
        },
        {
          "text": "destructor"
        },
        {
          "text": "import"
        },
        {
          "text": "subscribe"
        }
      ],
      "explanation": "a constructor is used by a class to take in parameters"
    },
    {
      "questionText": "Which is NOT an access modifier?",
      "options": [
        {
          "text": "private"
        },
        {
          "text": "protected"
        },
        {
          "text": "public"
        },
        {
          "text": "async",
          "correct": true
        }
      ],
      "explanation": "async is not used as an access modifier type in TypeScript"
    },
    {
      "questionText": "Which keyword allows us to share information between files
in TypeScript?",
      "options": [
        {
          "text": "import"
        },
        {
```

```
          "text": "export",
          "correct": true
        },
        {
          "text": "async"
        },
        {
          "text": "constructor"
        }
      ],
      "explanation": "the export keyword allows for the information to be
transmitted between files"
    },
    {
      "questionText": "Which is an array method to generate a new array based on
a condition?",
      "options": [
        {
          "text": "filter",
          "correct": true
        },
        {
          "text": "map"
        },
        {
          "text": "async"
        },
        {
          "text": "enum"
        }
      ],
      "explanation": "filter is a method used to conditionally create a new
array"
    },
    {
      "questionText": "How is a property accessible within a class?",
      "options": [
        {
          "text": "Using this.propertyName",
          "correct": true
        },
        {
          "text": "Accessors"
        },
        {
```

```
          "text": "Destructuring"
        },
        {
          "text": "Arrow function"
        }
      ],
      "explanation": "this.propertyName is the way to access a specific property
within a class"
    }
  ]
}
```

Global html;

```
<!DOCTYPE html>
<html lang="en">
  <head>
    <meta charset="utf-8"/>
    <title>OnlineTestApplication</title>
    <basehref="/"/>
    <meta name="viewport" content="width=device-width, initial-scale=1"/>
    <linkrel="icon" type="image/x-icon"href="favicon.ico"/>
    <link

href="https://cdn.jsdelivr.net/npm/bootstrap@5.0.2/dist/css/bootstrap.min.css"
      rel="stylesheet"
      integrity="sha384-
EVSTQN3/azprG1Anm3QDgpJLIm9Nao0Yz1ztcQTwFspd3yD65VohhpuuCOmLASjC"
      crossorigin="anonymous"
    />
    <link
      rel="stylesheet"
      href="https://cdnjs.cloudflare.com/ajax/libs/font-awesome/4.7.0/css/font-
awesome.min.css"
    />
  </head>
  <body>
    <app-root></app-root>
    <script
```

```
src="https://cdn.jsdelivr.net/npm/bootstrap@5.0.2/dist/js/bootstrap.bundle.min.js
"
      integrity="sha384-
MrcW6ZMFYlzcLA8Nl+NtUVF0sA7MsXsP1UyJoMp4YLEuNSfAP+JcXn/tWtIaxVXM"
      crossorigin="anonymous"
    ></script>
  </body>
</html>
```

Name:Abhishek Zende