

Online Test Application

This document contains the following

- Project and developer details
- Sprint planning and tasks completion
- Core concepts used in the project
- Links to the GitHub repository
- General info
- Using the application
- Technologies Used
- Demonstration of product capabilities, appearance, and user interactions
- Unique Selling Points of the application
- Conclusion

- **Project and Developer details:**

Project objective:

The Online Test Application system creates an application that enables users to provide online tests, review them, and display the results.

System details :

This system contains three main modules: Quiz, Review, and Result. The quiz section of the online test application accepts the questions in JSON format. The JSON file can be easily shared from the server in the pre-defined format. The application renders the test at the client-side. The "Review and display result" section allows users to declare the results immediately. You can simply call another JSON with the answers in it and evaluate and display the results immediately.

Developer Details:

Mahendra Kumar Singh

Mahendrakumarsingh9893@gmail.com

General Info

Online Quiz is Online Test Application platform which enables users to test their knowledge by giving quizzes.

This project was generated with Angular CLI version 13.3.6.

Using the application

Use npm install to install all the dependencies used in the Project.

- Run the Application in Visual Studio Code.
- Home page will be displayed. Home page contains list of quizzes that user can choose from.
- Based on User's Choice, Quiz is rendered in the Client Side by fetching data from Quiz Server or JSON file with Questions and Answers.
- User can select option for a question, Users also have the option to navigate through the questions using Navigation Button (First , Last , Next and Prev).
- Users can directly go to First and Last Questions by clicking on Navigation Buttons (First and Last).
- Users have the option to review their Quiz by clicking on Review button. Here user can see details of answered and not answered questions and navigate directly to the question which they want to Answer.
- After completing the quiz, User can submit the quiz where the Quiz Score is displayed on the Results page. As well as user can see their responses are right or wrong.
- There is timer set for quiz, If user exceeds quiz time then the quiz is automatically submitted and results are displayed to User.

Technologies Used

HTML5

CSS3

Bootstrap5

Angular

JSON

TypeScript

VS Code(Editor)

Development server

Run ng serve for a dev server. Navigate to <http://localhost:4200/>. The application will automatically reload if you change any of the source files.

Code scaffolding

Run ng generate component component-name to generate a new component. You can also use ng generate directive|pipe|service|class|guard|interface|enum|module.

Build

Run ng build to build the project. The build artifacts will be stored in the dist/ directory.

Running unit tests

Run `ng test` to execute the unit tests via Karma.

Running end-to-end tests

Run `ng e2e` to execute the end-to-end tests via a platform of your choice. To use this command, you need to first add a package that implements end-to-end testing capabilities.

Further help

To get more help on the Angular CLI use `ng help` or go check out the Angular CLI Overview and Command Reference page.

Links to the GitHub repository:

<https://github.com/Mahendra1272/OnlineTestApplication.git>

Pushing the code to GitHub Repository:

- Open Git Bash and navigate to the folder where you have created your files.
cd OnlineTestApplication \ OnlineTestApplication
- Initialize the repository using the below command **git init**
- Add all the files to your git repository using the below command **git add .**
- Commit the changes using the below command **git commit -m "Initial commit"**
- Add the URL for the remote repository where your local repository will be pushed **git remote add origin**
<https://github.com/Mahendra1272/OnlineTestApplication.git>
- Push the files to the folder you initially created using below command **git push -u origin master**

Sprint Plan:

Sprint 1:

- 1) - Plan flow of “Online Test Application”
- 2) - install and configure angular and node
- 3) - Create App
- 4) - Add design elements using CSS and HTML and bootstrap

Sprint 2:

- 1) - Create JASON data for the quiz
- 2) - create quiz component and integrate with JASON and HTML CSS and Bootstrap
- 3) - Test code, functionality, and design to ensure the app is ready to use
- 4) - Write final specification doc

System Details

This system contains three main modules: Quiz, Review, and Result. The quiz section of the online test application accepts the questions in JSON format. The JSON file can be easily shared from the server in the pre-defined format. The application renders the test at the client-side.

The “Review and display result” section allows users to declare the results immediately. You can simply call another JSON with the answers in it and evaluate and display the results immediately.

Source Code

Header

Header.component.html

```
<nav class=" navbar-dark mb-3" style="background-color: #99f7bd;">
  <div class="container-fluid">
    <a class="navbar-brand text-bolt" href="#">
      
      <span class="text-uppercase fw-bold" style="color: #200F8C; font-size:
18px;"> QUIZ</span>
    </a>
  </div>
</nav>
```

Header.component.spec.html

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

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

describe('HeaderComponent', () => {
  let component: HeaderComponent;
  let fixture: ComponentFixture<HeaderComponent>;

  beforeEach(() => {
    TestBed.configureTestingModule({
      declarations: [HeaderComponent]
    });
    fixture = TestBed.createComponent(HeaderComponent);
    component = fixture.componentInstance;
    fixture.detectChanges();
  });

  it('should create', () => {
    expect(component).toBeTruthy();
  });
});
```

Header.component.ts

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

@Component({
  selector: 'app-header',
  templateUrl: './header.component.html',
  styleUrls: ['./header.component.css']
})
export class HeaderComponent {

}
```

Question

Question.component.css

```
.card {
  max-width: 800px;
  margin: 0 auto;
  padding: 10px;
  color: #00ED5B;
  font-weight: 500;
  border: 2px solid #200F8C;
}
li {
  list-style-type: none;
  cursor: pointer;
  margin: 10px 0;
}

li .card:hover{
  background-color: #200F8C;
  border: 2px solid;
}
ol {
  padding: 0;
}
```

Question.component.html

```

<div class="container">
  <div class="card">
    <div class="d-flex justify-content-between p-3">
      <div class="imag">
        
      </div>
      <div class="quiz-header">
        <h4 style="font-family: cursive; color: #200F8C; font-weight:
700;"> SQL Quiz</h4>
        <span style="font-style: italic; color: #00ED5B; font-weight:
500;">Welcome {{name}}</span>
      </div>
    </div>
    <ng-container *ngIf="!isQuizCompleted">
      <div class="d-flex justify-content-around py-2">
        <div class="score">
          <h5 style="color: hwb(143 0% 7%);">{{points}} points</h5>
        </div>

        <div class="question-remain">
          <span style="font-style: italic; color: #200F8C; font-weight:
500;">Question {{currentQuestion+1}} of {{questionList.length}}</span>
        </div>

        <div class="timer" style="color: #00ED5B; font-weight: 500;">
          {{counter}} sec □
        </div>
      </div>
      <div class="progress mb-3"> <div class="progress-bar progress-bar-striped
bg-success" [ngStyle]="{'width':progress+'%'}">
    </div>

    </div>

    <div class="question">
      <div class="card" style="background-color: #99f7bd;">
        <h3 style="color:
#200F8C;">{{questionList[currentQuestion]?.questionText}}</h3>
      </div>
    </div>

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

```

```

        <li (click)="answer(currentQuestion+1,option)">
            <div appChangeBg [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 fa-chevron-left fa-2x" aria-
hidden="true" style="color: #200F8C;"></i></button>
    <button class="btn"(click)="resetQuiz()"><i class="fa fa-refresh fa-2x"
aria-hidden="true" style="color: #200F8C;"></i></button>
    <button [disabled]="currentQuestion===questionList.length-1" class="btn"
(click)="nextQuestion()"><i class="fa fa-chevron-right fa-2x" aria-hidden="true"
style="color: #200F8C;"></i></button>
</div>
</ng-container>

<ng-container *ngIf="isQuizCompleted && points>=50">
    <div class="row d-flex justify-content-between py-3">
        

        <div class="result text-center col-md-6 col-sm-12">
            <h3>Congratulations!! <br> You have complited the Quiz. <br>
Below is your result: </h3>
            <p>Total Questions 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>

<ng-container *ngIf="isQuizCompleted && points<=40">
    <div class="row d-flex justify-content-between py-3">
        

```



```

        <div class="result text-center col-md-6 col-sm-12">
            <h3>Sorry! <br> You have completed the Quiz. <br> Below is
your result: </h3>
            <p>Total Questions 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>

```

Question.component.spec.ts

```

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

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

describe('QuestionComponent', () => {
    let component: QuestionComponent;
    let fixture: ComponentFixture<QuestionComponent>;

    beforeEach(() => {
        TestBed.configureTestingModule({
            declarations: [QuestionComponent]
        });
        fixture = TestBed.createComponent(QuestionComponent);
        component = fixture.componentInstance;
        fixture.detectChanges();
    });

    it('should create', () => {
        expect(component).toBeTruthy();
    });
});

```

Question.component.ts

```

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

```

```

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

@Component({
  selector: 'app-question',
  templateUrl: './question.component.html',
  styleUrls: ['./question.component.css']
})

export class QuestionComponent implements OnInit {
  public name:any;
  public questionList:any=[];
  public currentQuestion:number=0;
  public points:number=0;
  counter=60;
  correctAnswer:number=0;
  incorrectAnswer:number=0;
  interval$:any;
  progress:string="0";
  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();
      this.getProgressPerchant();
    },1000)

  }
  else
  {
    setTimeout(()=>{
      this.currentQuestion++;
      this.incorrectAnswer++;
      this.resetCounter();
      this.getProgressPerchant();
      this.points-=10;
    },1000)

  }
}

```

```

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();
    },60000);
}

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";
}

getProgressPerchant()
{
    this.progress
    =((this.currentQuestion/this.questionList.length)*100).toString();
    return this.progress
}

}

```

Services

Question.services.spec.ts

```
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();
  });
});
```

Question.services.ts

```
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");
  }
}
```

Welcome

Welcome.component.css

```
li {
  color: #00ED5B;
}
input {
  border: 1px solid #200F8C;
}
```

Welcome.component.services.ts

Welcome.component.html

```
<div class="container p-5">
  <h1 class="display-8 fw-bold" style="color: #200F8C;">Welcome to Quiz
App</h1>
  <p class="col-md-8 fs-4" style="color: #00ED5B;">This quiz contains total
  {{questionList.length}} questions. Each Question holds 10 Points</p>
  <h4 style="color: #200F8C;">Rules:</h4>
  <ol>
    <li>Correct Question gives you 10 points</li>
    <li>Incorrect question gives to -10 points</li>
    <li>You will have 60 sec to answer each question</li>
    <li>Refereshing the page will reset the Quiz</li>
    <h1 style="font-family:cursive;text-align:center; color: #200F8C;">All
the best!!</h1>
    <div class="name col-md-4 my-3">
      <label for="" style="color: #00ED5B;">Enter Name</label>
      <input #name type="text" class="form-control" placeholder="Enter
Name">
    </div>
    <button class="btn btn-lg" routerLink="/question" (click)="startQuiz()"
style="background-color: #200F8C; color: #00ED5B; font-weight: 600;">Start The
Quiz</button>

  </ol>

</div>
```

Welcome.component.spec.ts

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

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

describe('WelcomeComponent', () => {
  let component: WelcomeComponent;
  let fixture: ComponentFixture<WelcomeComponent>;

  beforeEach(() => {
    TestBed.configureTestingModule({
      declarations: [WelcomeComponent]
    });
    fixture = TestBed.createComponent(WelcomeComponent);
    component = fixture.componentInstance;
    fixture.detectChanges();
  });

  it('should create', () => {
    expect(component).toBeTruthy();
  });
});
```

Welcome.component.ts

```
import { Component, OnInit, ViewChild, ElementRef } from '@angular/core';
import { QuestionService } from '../service/question.service';

@Component({
  selector: 'app-welcome',
  templateUrl: './welcome.component.html',
  styleUrls: ['./welcome.component.css']
})
export class WelcomeComponent implements OnInit{
  public questionList:any=[];
```

```

@ViewChild('name') namekey!:ElementRef;
constructor(private questionService:QuestionService)
{

}

ngOnInit(): void
{
    this.getAllQuestions();

}
startQuiz()
{
    localStorage.setItem("name",this.namekey.nativeElement.value)
}
getAllQuestions()
{
    this.questionService.getQuestionJson().subscribe(res=>{this.questionList=res.
questions;})
}
}

```

App-routing.moduls.app

```

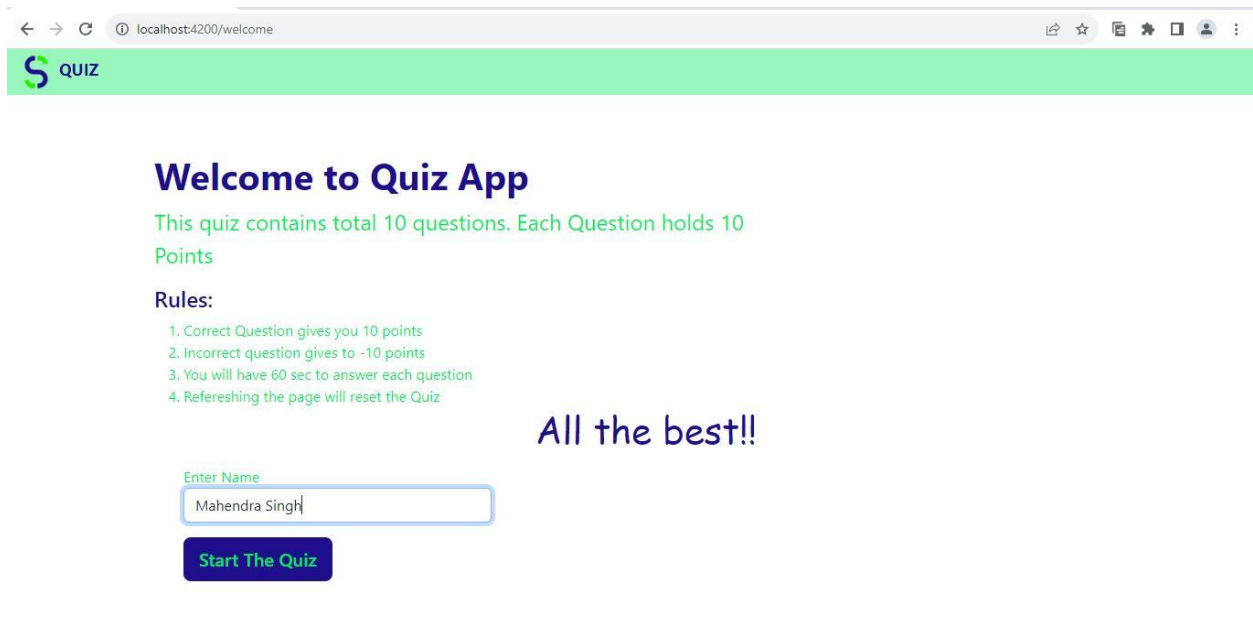
import { NgModule } from '@angular/core';
import { RouterModule, Routes } from '@angular/router';
import { WelcomeComponent } from './welcome/welcome.component';
import { QuestionComponent } from './question/question.component';

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 AppRoutingModuleModule { }

```


ScreenShot



The screenshot shows a web browser window with the address bar displaying 'localhost:4200/welcome'. The browser's toolbar includes back, forward, and refresh buttons, as well as icons for bookmarks, extensions, and a user profile. Below the address bar is a green header bar with a logo consisting of a green 'S' and the word 'QUIZ' in white. The main content area has a white background. It features a large heading 'Welcome to Quiz App' in dark blue, followed by a green text block stating 'This quiz contains total 10 questions. Each Question holds 10 Points'. Below this is a section titled 'Rules:' with a list of four rules in green text. To the right of the rules, the text 'All the best!!' is displayed in dark blue. On the left side, there is a text input field with a green placeholder 'Enter Name' and the text 'Mahendra Singh' entered. Below the input field is a blue button with the text 'Start The Quiz' in white. A thin black horizontal line is positioned at the bottom of the page.

localhost:4200/welcome

S QUIZ

Welcome to Quiz App

This quiz contains total 10 questions. Each Question holds 10 Points

Rules:

1. Correct Question gives you 10 points
2. Incorrect question gives to -10 points
3. You will have 60 sec to answer each question
4. Refereshing the page will reset the Quiz

All the best!!

Enter Name

Mahendra Singh

Start The Quiz



SQL Quiz

Welcome Mahendra Singh

0 points

Question 1 of 10

51 sec ⌚

What does SQL stand for?

Structured Query Language

Structured Query List

Sample Query Language

None of these





SQL Quiz

Welcome Mahendra Singh

-10 points

Question 3 of 10

39 sec ⌚

Which SQL keyword is used to retrieve only unique values from a database table?

UNIQUE

DISTINCT

SELECT DISTINCT

UNIQUE VALUES





SQL Quiz

Welcome Mahendra Singh

-20 points

Question 5 of 10

57 sec ⌚

Which SQL clause is used to filter the results of a query based on a specified condition?

WHERE

FROM

FILTER

HAVING





SQL Quiz

Welcome Mahendra Singh

-30 points

Question 7 of 10

56 sec

Which SQL statement is used to remove a table from a database?

DELETE TABLE

DROP

REMOVE TABLE

ERASE





SQL Quiz

Welcome Mahendra Singh

-40 points

Question 9 of 10

55 sec ⌚

Which SQL function is used to calculate the average value of a numeric column?

AVG

MEAN

AVERAGE

TOTAL





SQL Quiz

Welcome Mahendra Singh



Sorry!
You have completed the Quiz.

Below is your result:

Total Questions Attempted :10

Total Correct Answered : 0

Total Wrong Answered : 10

Your Score : -50 Points