1. **Create the app (choose SCSS):**

Partial structure shown; create new components and service. You will still have the app component files (not shown).

angular-quiz-app/

│

├── src/

│ ├── app/

│ │ │

│ │ ├── services/

│ │ │ └── quiz.service.ts

│ │ │

│ │ ├── pages/

│ │ │ ├── home/

│ │ │ │ ├── home.component.ts

│ │ │ │ ├── home.component.html

│ │ │ │ └── home.component.scss

│ │ │ │

│ │ │ ├── quiz/

│ │ │ │ ├── quiz.component.ts

│ │ │ │ ├── quiz.component.html

│ │ │ │ └── quiz.component.scss

│ │ │ │

│ │ │ └── result/

│ │ │ ├── result.component.ts

│ │ │ ├── result.component.html

│ │ │ └── result.component.scss

│ │

│ ├── main.ts

**2. File: app/routes.ts**

* Define **navigation paths**:
  + / → Home page
  + /quiz → Quiz page
  + /result → Result page
* \*\* use a **wildcard route** that catches any undefined paths and redirects to home.

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

import { HomeComponent } from './pages/home/home.component';

import { QuizComponent } from './pages/quiz/quiz.component';

import { ResultComponent } from './pages/result/result.component';

export const routes: Routes = [

{ path: '', component: HomeComponent },

{ path: 'quiz', component: QuizComponent },

{ path: 'result', component: ResultComponent },

{ path: '\*\*', redirectTo: '' }

];

**3. File: app/app.component.ts**

* Logic for the **main layout**.
* RouterOutlet to put the current page’s content here.
* If you're on /quiz, Angular will render QuizComponent inside this.

@Component({

standalone: true,

selector: 'app-root',

imports: [RouterOutlet],

template: `<router-outlet></router-outlet>`

})

export class AppComponent {}

**4. File: services/quiz.service.ts**

Provide the **question data and score storage** — see questions.txt.

* Create a **service** — a class that shares data across components.
* questions contains your quiz.
* score tracks the user’s correct answers.
* Components like Quiz and Result use this service to get/set data.

@Injectable({ providedIn: 'root' })

export class QuizService {

private questions = [ ... ]; // Array of question objects

private score = 0;

getQuestions() { return this.questions; }

getScore() { return this.score; }

setScore(score: number) { this.score = score; }

}

**5. Home Page**

**home.component.ts**

@Component({

standalone: true,

...

})

export class HomeComponent {}

→ Declares the **HomeComponent**.

**home.component.html**

<h1>Welcome to the Angular Quiz</h1>

<button routerLink="/quiz">Start Quiz</button>

→ This is a simple home screen with a button that routes to the quiz.

**6. Quiz Page**

**quiz.component.ts**

This is where the **quiz logic** goes.

* Get the list of questions from the service.
* Track which question you’re on.
* Record the selected answer and running score.

export class QuizComponent implements OnInit {

questions: any[] = [];

currentQuestionIndex = 0;

selectedAnswer = '';

score = 0;

//Load questions when the component appears.

ngOnInit() {

this.questions = this.quizService.getQuestions();

}

//When the user clicks an answer, record it

selectAnswer(option: string) {

this.selectedAnswer = option;

}

//When you click “Next,” this checks your answer, updates the score, and moves to the next question — or ends the quiz

nextQuestion() {

if (this.selectedAnswer === this.questions[this.currentQuestionIndex].answer) {

this.score++;

}

this.selectedAnswer = '';

this.currentQuestionIndex++;

if (this.currentQuestionIndex >= this.questions.length) {

this.quizService.setScore(this.score);

this.router.navigate(['/result']);

}

}

}

**quiz.component.html**

<!—Display current question—>

<h2>Question {{ currentQuestionIndex + 1 }} of {{ questions.length }}</h2>

<p>{{ questions[currentQuestionIndex]?.question }}</p>

<!— Loop over answer options. Highlight the one you clicked. —>

<li \*ngFor="let option of questions[currentQuestionIndex]?.options"

[class.selected]="selectedAnswer === option"

(click)="selectAnswer(option)">

{{ option }}

</li>

<!— Only clickable after you select an answer —>

<button [disabled]="!selectedAnswer" (click)="nextQuestion()">Next</button>

**7. Result Page**

**result.component.ts**

// On load, gets the score and total questions from the service. If more questions are added, the result page updates automatically.

export class ResultComponent implements OnInit {

score = 0;

totalQuestions = 0;

ngOnInit() {

this.score = this.quizService.getScore();

this.totalQuestions = this.quizService.getQuestions().length;

}

}

**result.component.html**

//Show the result and allow restart of quiz

<h2>Quiz Complete!</h2>

<p>Your score: {{ score }} / {{ totalQuestions }}</p>

<button routerLink="/">Back to Home</button>

**EXTRA: add Bootstrap to app.**