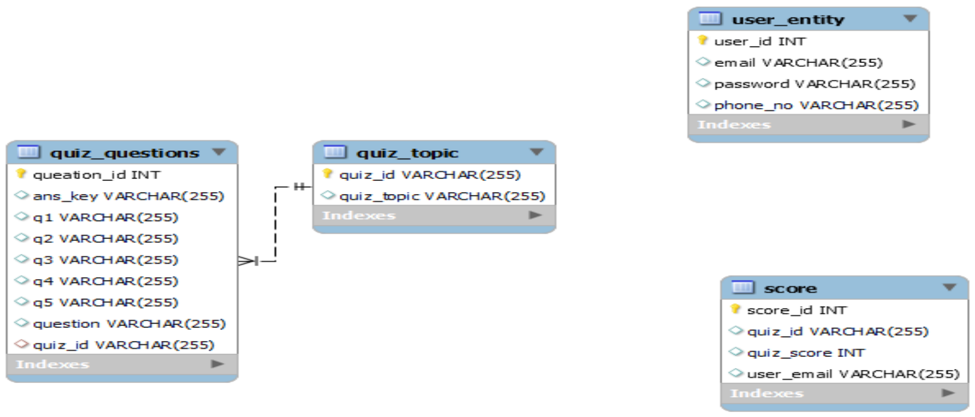
Quiz Application Documentation

Overview

This quiz application allows users to register, log in, select quiz topics, answer quiz questions, and view their scores. The application is built using Angular for the frontend and integrates with a backend service to manage quizzes and user data.

Architecture

ER-Diagram:-



Frontend

- **Angular**: A TypeScript-based framework used for building the user interface.
- **Components**: The application is composed of several Angular components, each responsible for different parts of the functionality.
  - **`LoginComponent`**: Handles user authentication.
  - **`RegistrationComponent`**: Manages user registration.
  - **`QuizComponent`**: Displays quiz questions, handles user answers, and manages the quiz timer.
  - **`Quiz-TopicsComponent`**: Lists available quiz topics for the user to choose from.
- **Services**: Angular services are used to handle business logic and communicate with the backend.
  - **`QuizService`**: Manages user authentication and registration. Fetches quiz data and submits user scores.

Backend

- **API**: The backend exposes RESTful endpoints to manage quizzes, user data, and authentication.
  - Endpoints:
    - **/ webquiz `/login`**: Authenticates users.

- **/ webquiz /`register`**: Registers new users.
- **/ webquiz /`addquiz`**: add new quiz.
- **/ webquiz / `allQuizs`**: Retrieves available quizzes.
- **/ webquiz / `getQuiz/{quizId}`**: Fetches specific quiz data.
- **/ webquiz / `quizscore`**: Submits user scores.

Technology Stack

- **Angular**: For building the frontend user interface.
- **TypeScript**: For adding type safety to the frontend code.
- **RxJS**: For handling asynchronous operations in Angular.
- **HTML/CSS**: For structuring and styling the frontend components.
- **Backend (assumed)**: The backend built using Spring Boot, Mysql Workbench.

Features

User Authentication

- **Login**: Users can log in with their email and password.
- **Registration**: New users can register by providing their details.

Quiz Functionality

- **Quiz Topics**: Users can select from a list of available quiz topics.
- **Quiz Questions**: Users answer questions within a set time limit.
- **Timer**: Each question has a timer, and if the time runs out, the question is skipped without penalizing the user.
- **Score Calculation**: Scores are calculated based on the number of correct answers.
- **Score Submission**: User scores are submitted to the backend and can be stored for future reference.

Additional Features

- **Responsive Design**: The application is designed to be responsive and work on various devices.
- **Error Handling**: Proper error messages are displayed for network issues and validation errors.
- **User Feedback**: Users receive instant feedback on their answers (correct/incorrect) during the quiz.
- **Time Limit per Question (30 seconds)** : Each question should have a time limit of 30 seconds. A countdown timer will be displayed alongside the question. If the user does not answer within 30 seconds and the next question will be displayed automatically.

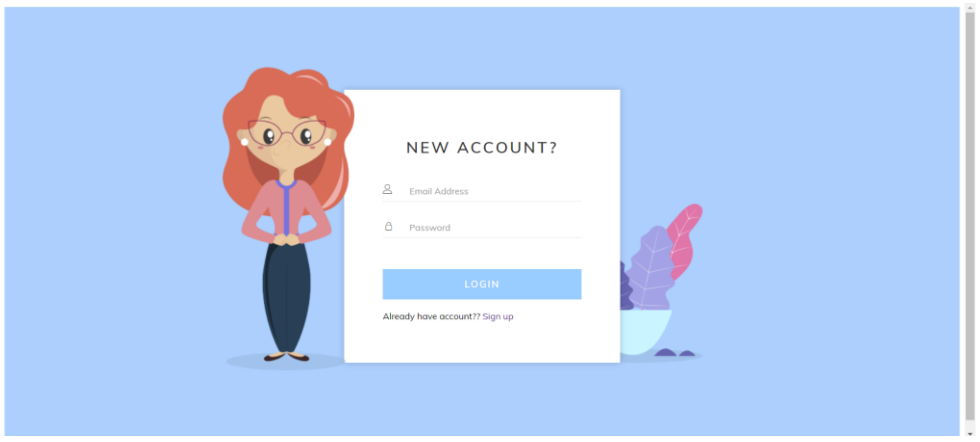
- **Current Progress Indicator:** Display the user's current progress during the quiz. Show the number of completed questions out of the total number of questions in the quiz (e.g., "Question 2 of 10").

Components Breakdown

LoginComponent

Handles user authentication.

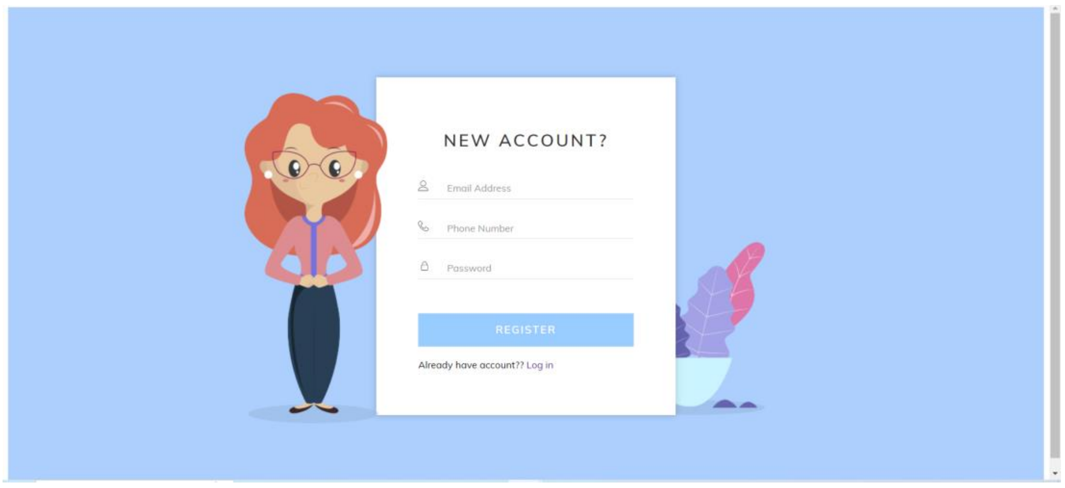
- **Template**: Form for user to enter email and password.
- **Logic**: Calls **`quizservice`** to authenticate the user.



RegistrationComponent

Manages user registration.

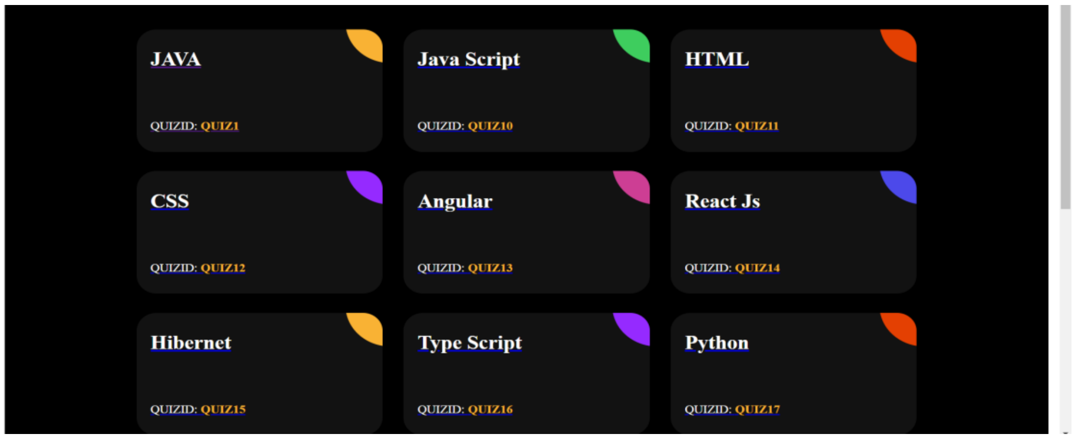
- **Template**: Form for user to enter registration details.
- **Logic**: Calls **`quizservice`** to register the user.



TopicsComponent

Lists available quiz topics.

- **Template**: Displays a list of quiz topics.
- **Logic**: Calls **`QuizService`** to fetch available quiz topics.



JAVA
QUIZID: QUIZ1

Java Script
QUIZID: QUIZ10

HTML
QUIZID: QUIZ11

CSS
QUIZID: QUIZ12

Angular
QUIZID: QUIZ13

React Js
QUIZID: QUIZ14

Hibernet
QUIZID: QUIZ15

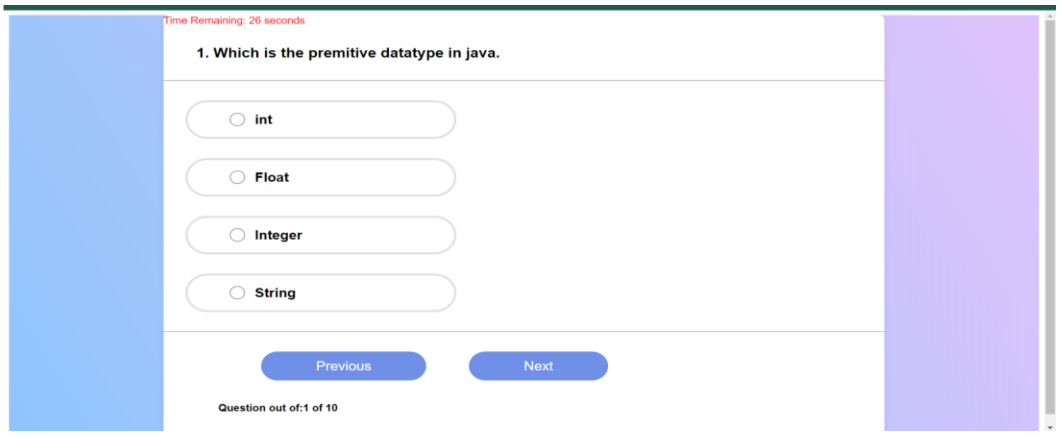
Type Script
QUIZID: QUIZ16

Python
QUIZID: QUIZ17

QuizComponent

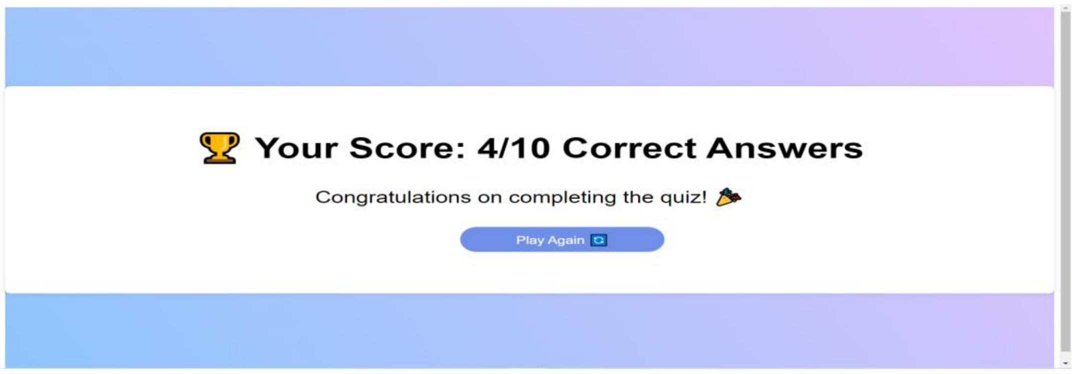
Displays quiz questions, handles user answers, and manages the timer.

- **Template**: Shows the current question, options, and timer.
- **Logic**:
  - Fetches quiz data from **`QuizService`**.
  - Handles user answers.
  - Manages the quiz timer.
  - Submits the quiz and calculates the score.


Time Remaining: 26 seconds

1. Which is the premitive datatype in java.

Question out of:1 of 10


🏆 Your Score: 4/10 Correct Answers

Congratulations on completing the quiz! 🎉

Services Breakdown

QuizService

Handles user authentication and registration.

- **Methods**:
  - `login(userCredentials)`: Authenticates the user.
  - `register(userDetails)`: Registers a new user.

Manages quiz data and user scores.

- **Methods**:
  - `getQuizzes()`: Fetches available quizzes.
  - `getQuizById(quizId)`: Fetches specific quiz data.
  - `scoreUser(scoreData)`: Submits user scores.

Usage

1. **User Registration**:
   - Navigate to the registration page.
   - Fill out the registration form and submit.
2. **User Login**:
   - Navigate to the login page.
   - Enter your email and password to log in.

3. **Select Quiz Topic**:
    - After logging in, select a quiz topic from the available list.
4. **Take the Quiz**:
    - Answer each question within the time limit.
    - Move to the next question or submit the quiz when completed.
5. **View Score**:
    - After submitting the quiz, view your score.
    - Optionally, retake the quiz or select a new quiz topic.