Punjab University College of Information Technology

BS (SE) Fall 2022 Morning

Web Engineering Quiz 01

Time Duration: 40 mints, Total Marks: 40

Case Study: Online Quiz Leaderboard Management System

Problem Statement

In a competitive online quiz application, players' performance is tracked and displayed on a leaderboard based on their scores. Each player's quiz results should be saved, and the leaderboard should dynamically update to reflect the latest rankings. The system should support functions for fetching and displaying player data, updating rankings, displaying top players, and saving new quiz results with updated player statistics.

Requirements

The **LeaderboardManager** class will serve as the primary class to manage leaderboard functions. It must:

1. **Fetch Player Data:** Pull player records from the database and store them in memory.
2. **Calculate and Update Rankings:** Adjust rankings based on each player's updated scores.
3. **Display Top Players:** Show the top 10 players on the leaderboard, ranked by their scores.
4. **Save New Quiz Results:** Store quiz results in the database and update relevant player statistics.

Task

1. Fetch Player Data

- ***Goal:*** *Retrieve and store player data in memory for quick access and processing.*
- ***Method:*** *Initialize an in-memory* `DataSet` *from the Players table, which will contain fields such as PlayerID, Username, Ranking, TotalQuizzesTaken, TotalCorrectAnswers, TotalIncorrectAnswers, and Score.*

2. Calculate and Update Rankings

- ***Goal:*** *Use each player's total scores to determine their rankings.*
- ***Method:*** *Develop a ranking calculation method to update the rankings whenever a new quiz result is added. Higher scores should correlate with a higher rank.*

3. Display Top Players

- ***Goal:*** *Show a list of the top 10 players based on their rankings.*
- ***Method:*** *Sort players by ranking and retrieve the top 10 to display on the leaderboard.*

4. Save Quiz Results and Update Statistics

- **Goal:** Record new quiz results and update player statistics such as the total number ofquizzes taken, total correct and incorrect answers, and the player's cumulative score.
- **Method:** Save each quiz result to the *Quizzes* table, then update the *Players* table to reflectthe new statistics.

Entities Description

Player

- **PlayerID** (int): Unique identifier for each player.
- **Username** (string): Name used by the player to log in and be displayed on the leaderboard.
- **Ranking** (int): Current rank of the player based on their score.
- **TotalQuizzesTaken** (int): Total number of quizzes attempted by the player.
- **TotalCorrectAnswers** (int): Number of correct answers provided by the player in quizzes.
- **TotalIncorrectAnswers** (int): Number of incorrect answers provided by the player in quizzes.
- **Score** (int): Total score accumulated by the player across all quizzes.

Quiz

- **QuizID** (int): Unique identifier for each quiz.
- **QuizDate** (DateTime): Date and time when the quiz was taken.
- **PlayerID** (int): Identifier for the player who took the quiz.
- **TotalQuestions** (int): Total number of questions in the quiz.
- **CorrectAnswers** (int): Number of questions answered correctly by the player.
- **ScoreEarned** (int): Score obtained by the player for the quiz.

Code Skeleton:

```
using System.Data;

using Microsoft.Data.SqlClient;


namespace QuizLeaderboard

{

  public class Player

  {


  }

  public class Quiz

  {
}

public class LeaderboardManager
{



  public LeaderboardManager()
  {

  }

    private void FetchPlayerData()
  {
    // Implementation
  }

    public void DisplayLeaderboard()
  {
    // Implementation
  }

  public void SaveQuizResult(int playerId, int totalQuestions, int correctAnswers)
  {
          int incorrectAnswers =
          int scoreEarned ______________   // Assuming 5 points per correct answer
    // Implementation
  }

  private void UpdatePlayerStatistics(int playerId, int correctAnswers, int
incorrectAnswers, int scoreEarned)
```

```
    {
        // Implementation
    }

    private int CalculateRanking(int score)
    {
        // Implementation removed
    }

    private void SavePlayerData()
    {
        // Implementation
    }
}

internal class Program
{
    static void Main(string[] args)
    {
        LeaderboardManager leaderboardManager = new LeaderboardManager();
        ShowMenu(leaderboardManager);
    }

    public static void ShowMenu(LeaderboardManager leaderboardManager)
    {
                //implementation of your own
    }
}
```