DUA CHALLENGE APP

Author: Muhamed Zahiri
Platform: iOS
Framework: UIKit
Language: Swift
External Packages: None

SHORT SUMMARY

The app is a soccer league management tool that generates 20 unique soccer teams and simulates fixtures and results. It allows users to view the teams ordered by their points in a table view. The app focuses on the logic and functionality rather than complex UI design elements.

Each team has properties such as name, points, and additional attributes. The fixtures are generated to ensure that each team plays every other team twice, both home and away. Random goal scores determine the winners or draws, and the teams' points are updated accordingly. The TeamStandingsController displays the teams in descending order based on their points, showing team names, points, and additional attributes. Users can tap on a team cell to view more details about the team. The app's code is organized following Swift best practices, with appropriate classes and methods. Code comments are included to explain key sections, and this brief documentation provides an overview of the challenge and specific instructions.

The app's code is hosted on GitHub, which you can find it in this link: https://github.com/EDI1IDE/Dua-Challenge-iOS.git

P.s. Below you will find the task as it was originally given:

CHALLENGE TASK

Here are the requirements for our code challenge that involves generating soccer teams, fixtures, results, and displaying them in a table view based on points collected. The challenge will be focused on logic rather than UI design and will use UIKit for the user interface.

1. Team Class

- Create a Team class with properties such as name, points, and any additional creative attributes you'd like to include, such as team logo, coach name, or city.
- Implement a method to calculate the goal difference for a team.

2. Team Generation

- Generate 20 unique soccer teams with random names and assign initial points to each team (e.g., start with 0 points for all teams).
- Assign additional creative attributes to each team object.

3. Fixture Generation

- Implement logic to ensure that each team plays every other team twice, both home and away.
- Randomly generate fixture pairs and store them in an appropriate data structure.

4. Result Generation

- For each fixture, generate a random number between 0 and 5 to represent the number of goals scored by the home and away teams.
- Determine the winner (or draw) based on the goal scores.
- Update the points for each team accordingly: 3 points for a win, 1 point for a draw, and 0 points for a loss.

5. Order by Points View Controller

- Create a new view controller that displays the teams in a table view.
- Sort the teams based on their points in descending order.
- Display the team name, points, and any additional creative attributes in the table view cells.

6. Navigation and User Interaction

- Implement navigation between the initial view controller (team generation) and the order by points view controller.
- Allow users to tap on a team cell to view additional details about the team, such as its creative attributes.

7. UI Design

- Create a simple and clean user interface using UIKit.
- Focus on functionality rather than complex UI design elements.
- Use a table view to display the teams and their information in the order by points view controller.

8. Code Organization and Documentation

- Organize your code into appropriate classes and methods, following Swift best practices.
- Include code comments to explain the purpose and functionality of key sections.
- Provide a brief documentation outlining the purpose of the challenge and any specific instructions or considerations.

Remember to focus on the logic and functionality of the challenge while implementing the required features.

Note: The project should be uploaded in Github and then share the link with us.