

Kurdistan regional government
Ministry of higher education And scientific research
University of koya faculty of engineering
Software engineering department
Third stage



Fc Official website

Web advance /php

Created By:

Barham Bapir

Supervised By:

DR.Safar

2023-2024

Table of Contents

Introduction	1
Sign in & sign up.....	2
Chapter 1.....	3
What the user can see & do	3
Home Page	4
About club.....	5
Squad	6
Table of matches.....	7
Contact us	9
Chapter 2.....	10
What the Admin can do and see	10
Admin Dashboard: Managing Match Information	11
Admin Dashboard: User Management and Feedback Display	12
Upmatches admin Dashboard	14
Players Admin dashboard.....	15
Table of matches : Admin dashboard	16
Reward Table	17
Conclusion	18

Introduction

In the ever-evolving landscape of web development, creating a website that not only captivates users but also empowers administrators with robust control features is paramount. This report documents the meticulous process of crafting a dynamic sports website, meticulously engineered to offer an immersive user experience while providing administrators with comprehensive management tools.

The cornerstone of our website lies in its multifaceted authentication system, seamlessly integrating login and signup functionalities. Users are greeted with a secure and intuitive interface, allowing for hassle-free account creation and access to personalized features. Concurrently, stringent security measures ensure the confidentiality and integrity of user data, fostering trust and reliability.

Central to our website's architecture is the incorporation of an administrative control panel, designed to empower administrators with unparalleled oversight and management capabilities. Through privileged access credentials, administrators gain entry to a dedicated dashboard, facilitating seamless content curation, user management, and database administration. A specialized login mechanism distinguishes admin access, fortifying system integrity and control.

At the forefront of user engagement is our website's dynamic content display. The homepage greets visitors with an interactive tableau, featuring real-time updates on ongoing matches, club information, and seasonal standings. Through intuitive design and dynamic data presentation, users are immersed in the latest sporting narratives, fostering a sense of community and excitement.

Integral to our website's functionality is a comprehensive database management system, meticulously cataloging player profiles, match statistics, and administrative records. Administrators wield unprecedented control over database operations, facilitating seamless content retrieval, manipulation, and update processes.

Additionally, our website features a dedicated section highlighting the match of the week and associated point system. Through an interactive interface, users stay informed about upcoming fixtures, recent results, and team standings. Real-time updates ensure users remain engaged and invested in the sporting narrative, fostering camaraderie and anticipation among fans.

In essence, the development of this website represents a convergence of innovation, functionality, and user-centric design. By seamlessly integrating login, signup, admin control, and dynamic content display features, we aim to deliver an unparalleled user experience while providing administrators with comprehensive management tools. This report chronicles our journey in building a website poised to redefine the standards of online sports platforms, catering to the diverse needs and preferences of our users.

Sign in & sign up

The provided code exemplifies the meticulous construction of a user authentication system within a sports website. At its core are two distinct yet interconnected sections: the signup and login pages.

The signup page (signup.php) greets potential users with a welcoming interface, inviting them to create an account. Through a carefully designed HTML form, users input essential details including their desired username, email address, and password. Client-side validation mechanisms ensure data integrity, enforcing email format and password complexity standards. Upon submission, PHP scripts take charge, seamlessly processing the user's data. Robust security measures, such as sanitization via `mysqli_escape_string`, safeguard against potential SQL injection vulnerabilities. The system rigorously checks if the provided email already exists in the database. If not, the user's credentials are securely stored, heralding their successful registration. Subsequently, a swift redirection to the login page (login.php) awaits, signaling readiness for the next step in their journey.

Transitioning to the login page, users are greeted with familiarity and assurance. The form entices with promises of swift access to their personalized space. Once again, HTML form elements stand at the forefront, collecting user credentials with precision. As the user submits their details, PHP scripts spring into action, orchestrating a series of checks and verifications. Notably, the system differentiates between regular users and administrative personnel. A predefined set of admin credentials acts as a gatekeeper, granting exclusive access to administrative privileges. Meanwhile, for regular users, meticulous database queries ascertain the accuracy of their provided information. Upon successful validation, users are ushered into the heart of the website, while any discrepancies prompt gentle reminders to try again.

In essence, this code embodies the harmonious convergence of frontend and backend technologies. Through intuitive interfaces and robust backend logic, users embark on a seamless journey of registration and authentication. As the foundation of user interaction, these pages epitomize the website's commitment to security, accessibility, and user-centric design, setting the stage for a compelling sports-centric experience.

Chapter 1

What the user can see & do

Home Page

The inclusion of 'config.php' at the beginning of the file ensures that essential configurations and database connections are set up before any database operations are performed. This file typically contains variables and functions necessary for establishing connections to the database and executing queries.

The dynamic image carousel implemented within the HTML body retrieves images from a database table named news. This carousel aims to display a rotating selection of images fetched from the database. This functionality is achieved through PHP's ability to execute SQL queries (*SELECT * FROM news*) and iterate over the result set using a while loop. Inside the loop, PHP generates HTML image tags () dynamically, embedding the URLs of the images retrieved from the database. By dynamically populating the carousel with images, the webpage ensures that users always see the latest content without requiring manual updates.

The JavaScript code further enhances user interaction by implementing a countdown timer for an upcoming match. This timer dynamically calculates and displays the remaining time until the scheduled match. The script utilizes JavaScript's setInterval function to update the countdown display at regular intervals, ensuring that users receive real-time updates on the match schedule. By calculating the time difference between the current date and the scheduled match date, the countdown timer provides users with an accurate and engaging way to anticipate upcoming events.

Additionally, PHP fetches data about upcoming matches from the matches table in the database. This data includes details such as team names, logos, timings, and locations. By executing an SQL query (*SELECT * FROM matches*) and iterating over the result set, PHP dynamically generates HTML content to display match information within the webpage. Each iteration of the loop constructs a structured HTML layout, ensuring that match details are presented in an organized and visually appealing manner. This dynamic integration of PHP and HTML allows the webpage to adapt to changes in match schedules and display up-to-date information to users.

Overall, these functionalities demonstrate the dynamic nature of the webpage, which leverages PHP and JavaScript to deliver interactive and engaging user experiences. By seamlessly integrating server-side and client-side technologies, the webpage provides users with up-to-date content, real-time updates, and interactive features, enhancing their overall browsing experience.

About club

The PHP code provided forms a detailed and well-structured webpage that represents the illustrious history and achievements of Real Madrid, leveraging HTML, CSS, PHP, and JavaScript to create an interactive and informative experience for its visitors.

Starting with the basic structure, the webpage is defined with standard HTML5 doctype, ensuring modern browser compatibility and performance. The head section of the document is enriched with meta tags that define the character set as UTF-8 for wide character compatibility and set the viewport to ensure responsive behavior on different devices. To enhance the visual appearance, external CSS files specific to different components of the webpage—such as the navigation bar, main club-related styling, and footer—are linked. This not only ensures that the styling is consistent and modular but also helps in maintaining and updating the styles without affecting the HTML structure.

The body of the document begins with the inclusion of a navigation bar from an external PHP file, which exemplifies the use of PHP include statements to reuse code efficiently across multiple pages. This is particularly useful for maintaining a consistent look and navigation experience across the website. The core content of the webpage is encapsulated within a div container that features a dynamic background image representing the club's fans, setting an engaging backdrop. Overlaying this image, a decorative logo and the title "About Club" are introduced, which are central to branding and thematically introducing the club.

In detailing the club's rich history, a dedicated section vividly outlines the significant milestones and figures associated with Real Madrid. From its foundation in 1902 through its rise under President Santiago Bernabéu to its recent triumphs in European championships, the narrative captures the essence of the club's journey. This historical recount not only serves to inform but also to instill pride and connect with the club's fans globally.

Further enriching the webpage is the "Club Presidents" section, which uses a slider to showcase various key figures who have led the club. Each president is featured on individual cards within the slider, complete with photographs and brief descriptions of their contributions and tenure. This section is made interactive through JavaScript, where buttons labeled 'prev' and 'next' allow users to navigate through the cards. This functionality is handled by JavaScript code that manipulates the display properties of the cards, ensuring that only one card is visible at a time, thereby facilitating a clean and user-friendly interface.

Another dynamic aspect of the site is the "REAL MADRID CHAMPIONS" section, which pulls data from a database to display the club's achievements. Using PHP to connect to the database and fetch records of trophies and awards, this section dynamically generates content based on the data retrieved. Each record is presented in a structured format, displaying images, titles, and years associated with each achievement. This not only keeps the webpage updated with the latest accolades but also highlights the club's ongoing success in an interactive format.

Lastly, the footer of the page is included through another PHP include statement, emphasizing reusability and ease of updates across the entire website. This approach of modular coding, combined with dynamic content generation and interactive elements, makes the webpage a robust, maintainable, and engaging digital representation of Real Madrid's legacy.

Squad

The PHP code you provided connects to a database and retrieves player data. It then loops through each player's information, creating HTML player cards dynamically based on their position. If a player is identified as a coach, goalkeeper, defender, midfielder, or attacker, the script generates a card displaying their image, name, and role. Each player card is structured using HTML elements, incorporating CSS classes defined in the accompanying CSS code for styling. Overall, this PHP script dynamically generates a webpage showcasing player information in an organized and visually appealing manner.

Let's delve into the PHP code and how it interacts with the CSS:

- `$query = mysqli_query($conn, "SELECT * FROM player JOIN squad ON (player.position = squad.squad_id)");`
This line executes a SQL query to fetch player data from a database. It retrieves all columns (*) from the player table, joining it with the squad table based on the condition that the position column in the player table matches the squad_id column in the squad table.
- `if($row['position'] == 1){`
Within the loop, it checks the value of the position column in each row. If the position is 1 (indicating a coach in this context), it proceeds to generate HTML code for a player card within the appropriate section. The number 1 on the database when we want to add to the goalkeeper we use 2 and for other positions 1-5
- `">`
`<div class="player-info">`
`<div class="player-name"><? = $row['name']; ?></div>`
`<div class="player-position"><? = $row['Role']; ?></div>`

This HTML markup represents a player card. It includes an image of the player (tag), player information such as name and role within a container (<div> tags with appropriate classes), and it uses PHP tags (<? = ?>) to dynamically insert values from the \$row array (e.g., player name and image URL).

The CSS classes (player-card, IMG-card, player-info, etc.) applied to these HTML elements determine their styling, as described in the CSS code you provided

the PHP code fetches player data from a database, iterates through each player, and generates HTML markup for player cards based on their position. The CSS styles defined in the provided CSS code then determine the visual presentation of these player cards on the webpage.

Table of matches

This HTML and PHP script collaboratively construct a dynamic table, meticulously capturing and presenting extensive insights into the performance metrics of diverse sports clubs within a league setting. The HTML framework, comprising class-defined elements, meticulously structures the visual layout, facilitating enhanced styling capabilities. Nested within the designated container, a comprehensively structured table is meticulously delineated, meticulously adorned with headers meticulously outlining vital metrics including club position, team identity, wins, draws, and losses. Simultaneously, the PHP component orchestrates a systematic process of data extraction from a dedicated 'club' database table, orchestrating a nuanced sorting mechanism based on descending points. Each iteration through the dataset precipitates the dynamic population of table rows, wherein club-specific data, encompassing nuanced attributes such as names, logos, win tallies, draw occurrences, loss counts, total match engagements, goals scored, and points accrued, is meticulously inscribed. Moreover, the implementation of sophisticated conditional PHP logic empowers the dynamic alteration of row appearances contingent upon club rankings, thereby endowing top-performing entities with distinctive visual highlights. This seamless amalgamation of HTML and PHP delineates an immersive and visually enriching tableau, seamlessly conveying intricate club standings and performance intricacies, thereby fostering heightened user engagement and informed decision-making processes within a singular, visually immersive interface.

W	D	L	matches	goalScored	points
---	---	---	---------	------------	--------

On this section the point is equal = 3win+ draw , we will do the calculate on the backend only show it here and also matches is win + draw + lose , and we will dedicate the the top by most point if point is equal we will sort them by most goalscored

And the position of them it's a counter top 5 is green and between is them is grey then the last 3 is red Its on the frontend

```
1 $query = mysqli_query($conn, 'SELECT * FROM `club` ORDER BY `point` DESC, `goalScored` DESC');
2 if (mysqli_num_rows($query) > 0) {
3     while ($row = mysqli_fetch_assoc($query)) {
4     }
5     <tr class="position">
6         <td>
7             <p class="positionPlace tdngo">
8                 <?php echo ++$position;
9                 if ($position >= 5) {
10                     echo '<script>';
11                     echo 'var positionPlaces = document.querySelectorAll(".positionPlace");';
12                     echo 'for (let i = 5; i < positionPlaces.length; i++) {';
13                     echo '    positionPlaces[i].style.backgroundColor = "grey";';
14                     echo '    if(i==positionPlaces.length-3){';
15                     positionPlaces[i].style.backgroundColor = "red";
16                     }';
17                     echo 'if(positionPlaces==18){';
18                     positionPlaces[i].style.backgroundColor = "green";
19                     }';
20
21                     echo '};';
22                     echo '</script>';
23                 }
24             </p>
25         </td>
```

Contact us

The code snippet provided combines PHP and HTML to create a functional contact form. The PHP section at the top interacts with form data submitted via the HTTP POST method.

When a user submits the form by clicking the "Submit" button, the PHP script checks if the 'Sumbit' (likely a typo, should be 'Submit') button was clicked using the `isset($_POST['Submit'])` condition. This is a common practice to ensure that the form data is being submitted through a specific action, in this case, clicking the submit button.

If the condition is met, indicating that the form was submitted, the script proceeds to extract data from the form fields using the `$_POST` superglobal array. It retrieves the values entered by the user for 'FirstName', 'Email', and 'feedback' fields, and sanitizes them using `mysqli_real_escape_string` to prevent SQL injection attacks.

After sanitizing the input, the script constructs an SQL query to insert the sanitized data into a database table named 'feedback'. If the query execution is successful (`$result` evaluates to true), a success message is echoed, and the user is redirected to the 'contactus.php' page using the `header()` function. Otherwise, if an error occurs during the query execution, an error message containing information about the error is echoed to assist in debugging.

The HTML section of the code defines the structure of the contact form and additional content such as location information and social media links. The form is set to submit data to the same page ('contactus.php') using the POST method, ensuring that the form data is sent securely without exposing it in the URL.

In summary, the code snippet demonstrates the use of the POST method to securely transmit form data to the server, where PHP scripts process and interact with the data to achieve desired functionality, such as inserting it into a database or performing other operations.

Chapter 2

What the Admin can do and see

Admin Dashboard: Managing Match Information

This admin dashboard provides a user-friendly interface for administrators to manage information about matches, including team names, match times, logos, and locations.

1. PHP Backend (Server-side):

The admin dashboard starts with including a common navigation bar (navbar.php) to maintain consistent navigation across the admin panel. It establishes a connection to the database by requiring the configuration file (config.php), which likely contains the database credentials and connection logic. The default timezone is set to UTC to ensure consistency in date and time handling across different locations. The backend PHP script handles form submissions. When the "Update" button is clicked, it captures the form data (\$_POST) including the club ID, team name, match time, logo path, and location. Using SQL, it constructs a query to update the matches table in the database with the new data based on the provided club ID. Upon successful execution of the query, the script redirects the user to the home page (../home.php) to reflect the changes.

2. HTML Frontend (Client-side):

The frontend of the admin dashboard is built using HTML to create a structured layout for managing match information. It consists of a form (<form>) with input fields for entering data about the match, such as the team name, logo path, match time, and location. Additionally, there's a dropdown menu (<select>) populated with options representing the matches available in the database. Only the first four matches are fetched and displayed as "game 1", "game 2", "game 3", and "game 4". The "Read" button functionality, as per your request, is omitted, leaving only the "Update" button for submitting form changes. Below the form, a table (<table>) is presented to visualize the existing match information. Data fetched from the database is displayed in rows and columns, showcasing details such as match ID, team names, match times, logos, and locations. In summary, this admin dashboard empowers administrators to efficiently manage match information through a well-organized interface, facilitating seamless updates and modifications to match details stored in the database.

Admin Dashboard: User Management and Feedback Display

This admin dashboard serves as a central hub for managing users and displaying feedback submissions. Let's dissect the code to understand its functionalities:

1. User Management Section:

Navbar Inclusion: The dashboard starts with including a common navigation bar (navbar.php) to ensure uniform navigation across the admin panel.

HTML Structure: The page is structured using HTML, with appropriate tags and elements for creating a tabular layout to display user information.

Styling: Basic CSS styles are applied to enhance the appearance of the tables, align content, and provide a visually pleasing user experience.

User Table: The table presents user information such as ID, name, email, and creation timestamp retrieved from the database. Each user entry also includes a "Delete" button to facilitate user deletion.

2. Feedback Display Section:

Feedback Table: The feedback table showcases feedback submissions received from users. Each row in the table represents a feedback entry, displaying the submitter's name, email, feedback content, and the timestamp when the feedback was submitted.

PHP Backend: Within the PHP section dedicated to handling feedback data, the code retrieves feedback entries from the feedback table in the database.

SQL Query: The script constructs an SQL query (`SELECT * FROM feedback`) to fetch all feedback entries stored in the database.

Result Processing: After executing the SQL query, the PHP script checks if there are any feedback entries available (`mysqli_num_rows($result) > 0`). If feedback entries exist, it iterates through each row of the result set using a while loop.

Data Presentation: For each feedback entry, the script retrieves relevant information such as the submitter's name, email, feedback content, and submission timestamp.

Timestamp Formatting: The submission timestamp is converted from its database representation to a human-readable format using PHP's `date()` function. This ensures that the timestamp is presented in a user-friendly format, including the day, date, year, and time of submission.

HTML Output: Inside the while loop, the script generates HTML table rows (`<tr>`) for each feedback entry, populating the cells (`<td>`) with the corresponding feedback details. The formatted timestamp is also displayed in the table row.

Conditional Handling: To ensure that only valid feedback entries are displayed, the script includes a conditional check to verify if the feedback content is not empty (if(\$row['Feedback']!=null)). This prevents displaying empty or null feedback entries in the table.

Error Handling: Additionally, error handling is implemented to gracefully handle cases where no feedback entries are found in the database. If no feedback entries exist, a message indicating "No users found" is displayed within a table row spanning all columns (<td colspan='4'>).

3. PHP Section for User Management:

User Deletion Handling: In the PHP section dedicated to user management, the code handles the deletion of user accounts based on admin action.

Form Submission Detection: The script checks if the "Delete" button associated with a user entry has been clicked (isset(\$_POST['deleteuser'])).

User ID Extraction: Upon detecting a form submission, the script retrieves the user ID (\$id) associated with the user entry scheduled for deletion from the form data (\$_POST['IDuser']).

SQL Deletion Query: Using the retrieved user ID, the script constructs an SQL query (DELETE FROM users WHERE id='\$id') to delete the corresponding user from the users table in the database.

Deletion Result Handling: After executing the deletion query, the script checks if the deletion operation was successful (\$result). If successful, a success message indicating "User deleted successfully" is echoed, and the page is refreshed to reflect the updated user list. Otherwise, an error message detailing the cause of the deletion failure is displayed.

UPMATCHES	USERS	PLAYERS	TABLE	REWARDS	LOGOUT
users					
ID	Name	Email	Created at	Action	
6	barham.200269674190	barhambaper6@gmail.com	Monday, May 13, 2024 at 11:52 PM	Delete	
7	mirawedali	barhampc6@gmail.com	Monday, May 13, 2024 at 11:54 PM	Delete	

feedbacks				
Name	Email	feedback	submitted at	
sath asare hwi	arg qry	hfhgjljfk:	Monday, May 13, 2024 at 11:56 PM	
sath asare hwi	arg qry	hfhgjljfk: nanbj	Monday, May 13, 2024 at 11:56 PM	
sath asare hwi	arg qry	hfhgjljfk: nanbj	Monday, May 13, 2024 at 11:56 PM	

Upmatches admin Dashboard

This PHP script comprises an admin dashboard interface for managing a database of matches. Here's a brief explanation of the code:

- Database Connection and Configuration:

It includes a configuration file (config.php) to establish a connection to the database using MySQLi extension.

- Update Functionality:

When the form with the name updateBtn is submitted (method="POST"), it updates the match details in the database based on the provided input. It retrieves data from the form fields (secondTeam, logoPath, timeClock, Location, and clubID) using \$_POST. The SQL query updates the matches table with the new values provided in the form fields, using UPDATE statement.

- HTML Form for Update:

Provides input fields (text, datetime-local, select) for updating match details. The select dropdown lists the available matches fetched from the database.

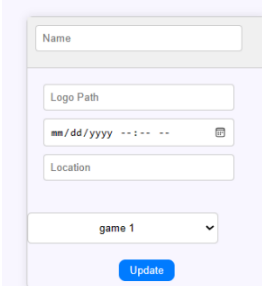
- Displaying Matches:

Fetches match data from the database and displays it in a table format. The matches table is populated with match details such as ID, team name, time, team logo, and location. Conditional logic is used to display the team logo if available; otherwise, it shows a default message. The PHP script for the admin dashboard manages a database of matches, but it's limited to updating existing data; it doesn't include functionality for adding new matches. Here's a brief summary with the mentioned constraint:

- Update Functionality:

Users can update match details using the provided form fields. The form submission triggers an SQL query to update existing match records in the database. No Addition Functionality: There's no provision for adding new matches. The interface solely focuses on modifying existing match data.

Matches				
ID	TeamName	Time	Team Second Logo	Location
18	paris	2024-06-01 17:57:00		Santiago
21	Arsenal FC	2024-06-04 15:37:30		santiago bernabeu
22	Inter Milan	2024-06-06 15:44:00		Stadio Giuseppe Meazza
24	chelsea	2024-06-22 15:41:08		stamford bridge



The form is titled 'Update' and contains the following fields: 'Name' (text input), 'Logo Path' (text input), a date and time picker set to 'mm/dd/yyyy --:-- --', 'Location' (text input), and a dropdown menu currently showing 'game 1'. An 'Update' button is at the bottom.

Players Admin dashboard

This PHP script presents an admin interface for managing player data in a football squad, featuring both creation and deletion functionalities. Here's a succinct summary:

Creating Players:

Users can input player details such as name, logo path, number, role, and position via a form. Upon submission, the entered data is inserted into the database table player using an SQL INSERT statement.

Deleting Players:

Each player entry in the displayed table includes a "Delete" button. Clicking this button triggers a form submission, which sends a request to delete the corresponding player record from the database using an SQL DELETE statement.

Displaying Player Information:

Player data is fetched from the database and presented in a table format. Each row includes player details like name, logo, number, role, and position. If no players are found in the database, a message indicating the absence of players is displayed.

The script implements a SQL join operation to retrieve player data along with their corresponding positions within the football squad. Specifically, it utilizes an SQL JOIN clause to combine data from the player table with data from the squad table based on a common field (position in player table and squad_id in squad table). Here's how the join operation is incorporated into the code:

```
1      $query = mysqli_query($conn, "SELECT * FROM player JOIN squad ON player.position = squad.squad_id");  
2
```

Within this SQL query:

*SELECT ** ensures all columns from both the player and squad tables are retrieved.

FROM player JOIN squad specifies data retrieval from both tables. *ON player.position = squad.squad_id* defines the condition for the join, linking the position column from the player table to the squad_id column from the squad table. This join operation enhances the displayed player information by incorporating their specific positions within the squad. Consequently, the admin dashboard provides a comprehensive overview of player details, including their assigned roles within the team structure. This integration of SQL joins enhances the efficiency and effectiveness of player management within the football squad administration interface.

Table of matches : Admin dashboard

The provided PHP script orchestrates a dynamic admin dashboard for managing football team data, allowing administrators to manipulate team records, including editing, deleting, and adding teams. The script begins with the inclusion of a navigation bar (navbar.php) for consistent site navigation across pages. It then proceeds to construct the HTML structure of the dashboard, which is predominantly comprised of a card-based layout to display team information.

Within the card-container division, the script initiates a database connection and retrieves team data from the club table. For each team retrieved, it dynamically generates a card element displaying essential team statistics, such as wins, losses, draws, goals scored, and points earned. Notably, the points are calculated based on the provided wins and draws.

```
$points = $row['win'] * 3 + $row['draw'];
```

Additionally, each card features an "Edit" button, which toggles the visibility of an edit form specific to the corresponding team. The edit form allows administrators to modify team statistics, including wins, losses, draws, and goals scored. Upon submission of the edit form, the script updates the team information in the database and reflects the changes dynamically without the need for a page refresh.

```
if (isset($_POST['update-btn'])) {  
    // Retrieval of form data and execution of SQL update query}
```

Furthermore, each card includes a "Delete" button, enabling administrators to remove teams from the database. Upon deletion, the script promptly updates the dashboard to reflect the changes, ensuring real-time data synchronization.

```
if (isset($_POST['delete-btn'])) {  
    // Execution of SQL delete query}
```

Moreover, the script facilitates the addition of new teams via a form located at the bottom of the dashboard. Administrators can input team details, including name, logo URL, wins, losses, draws, and goals scored. Upon submission of the form, the script inserts the new team information into the database, updating the dashboard dynamically to include the newly added team.

```
if (isset($_POST['add-team-btn'])) {  
    // Retrieval of form data and execution of SQL insert query}
```

Overall, this dynamic admin dashboard empowers administrators with comprehensive tools to manage football team data efficiently, fostering seamless interaction and manipulation of team records. Through the utilization of PHP, SQL, and JavaScript, the script delivers a responsive and intuitive user experience, facilitating streamlined team management operations.

Reward Table

The PHP script encapsulates a comprehensive administrative interface for managing rewards or titles within a web application. It seamlessly integrates with an underlying database through a configuration file (config.php), establishing a crucial connection that enables seamless data retrieval and manipulation. This robust script employs various functionalities, including dynamic content generation, form handling, input validation, and SQL database interaction, to empower administrators with efficient tools for reward management.

At its core, the script initiates by including the config.php file, which orchestrates the database connection. This modular approach enhances code maintainability and scalability by centralizing configuration settings. Once connected, the script proceeds to execute an SQL query targeting the rewards table to fetch all existing records. Leveraging the `mysqli_query()` function, the result set is retrieved and stored, laying the groundwork for dynamic content generation. Within a loop structure, each row of the result set is fetched as an associative array using `mysqli_fetch_assoc()`. This function enables seamless access to individual column values, facilitating the dynamic display of reward information within visually appealing card elements. The script leverages HTML and CSS to structure and style these cards, ensuring a user-friendly interface that aligns with modern design standards.

Key functionalities, such as updating and adding rewards, are seamlessly integrated into the script's logic. Conditional blocks, triggered by form submissions, govern the execution of these operations. For update operations, the script validates the presence of the update-btn form field before extracting and validating the updated reward information. Utilizing PHP's robust validation capabilities, the script ensures that all requisite fields are populated before proceeding with database interaction.

Upon successful validation, the script executes an SQL UPDATE query, seamlessly propagating modifications to the corresponding record in the database. This dynamic update mechanism enhances user experience by enabling administrators to effortlessly modify reward details without navigating away from the interface. Similarly, for adding new rewards, the script validates the submission of the add-team-btn form field before proceeding with data insertion.

The script's form validation logic serves as a critical safeguard against incomplete or erroneous data entry. By enforcing field completeness and integrity, the script maintains data consistency and accuracy within the database. Should any required fields remain empty, the script promptly notifies administrators, preventing submission until all necessary information is provided.

In conclusion, the PHP script exemplifies a sophisticated yet user-friendly administrative interface for managing rewards or titles within a web application. Through seamless integration of PHP, SQL, HTML, CSS, and form handling mechanisms, the script empowers administrators with efficient tools for reward management while prioritizing usability, accessibility, and code maintainability.

Conclusion

This document outlines the development of an official website for a football club, detailing the features and functionality available to both regular users and administrators. The website incorporates essential components such as user authentication, dynamic content display, and comprehensive database management.

For users, the website offers an immersive experience with interactive features like match schedules, team standings, and player profiles. The homepage dynamically showcases upcoming matches, while dedicated sections provide in-depth information about the club's history, achievements, and current squad.

Administrators are empowered with a robust control panel, enabling them to manage match information, user accounts, and feedback submissions efficiently. The admin dashboard facilitates updates to match details, player rosters, and the ability to moderate user data seamlessly.

The document meticulously describes the implementation of these features, leveraging a combination of server-side technologies like PHP and MySQL, along with client-side scripting using HTML, CSS, and JavaScript. This integration of technologies ensures a dynamic, responsive, and interactive experience for both users and administrators.

Overall, the document highlights the development of a comprehensive sports website that caters to the needs of football enthusiasts, providing them with a platform to stay updated on their favorite club while offering administrators the necessary tools to manage and curate content effectively.

