201148056

DYNAMIC WEB PROGRAMMING

COMM2735

Report

1712 WORDS

Project brief

My website can be accessed from: http://me17jf.leedsnewmedia.net/COMM2735/worldcupsquads/index.php. It was created to allow fans to easily see what players were selected to represent their country at the 2018 World Cup in Russia. Through implementing a comments section, fans can offer their opinions in what they thought of managers selections and discuss the competition.

Project report

The user first visits the homepage, where they can quickly view what players have been selected for the different teams at the World Cup. This is important as "people tend to spend very little time reading most Web pages" (Krug, 2005). This is achieved through the use of a drop-down list of all the teams at the tournament. The user selects which team they want to view and click submit, which acts as a search function, and then displays the list of the players that have been selected for that squad. A reCAPTHA validation feature has been included to prevents robots from destroying and bringing down the website. This is an important security mechanism as reCAPTHA will be able to distinguish that the user is human and allow them to use the website.

Users then have the option from this page to register or login. This is where key validation and verification techniques have been implemented. With the register form, one being 'username is too long.' This is needed to limit data input, as too much data could slow down the performance of the site. As well as 'password must be 6-25 characters long' to ensure a secure enough password has been submitted so it harder for hackers to figure out. Usernames and passwords are key security features of the site. 'Username already exists' is important validation as your "username is your identification" (Forbes, 2017). Meaning the identity of an account can be identified which means actions can be pursued for a specific user, while also giving a user and identity in the comments section.

'Both password fields must match' is used to ensure the user is aware of the specific password being entered, as an error can easily be made if only entered once. "Please fill in all fields" is used to alert the user that important information needed is missing. If the user successfully passes the checks, the website will prompt the user with the message "Registration successful!". Now they have an account with the website, they can log in. Further verification is used here to ensure that only eligible users that have signed up can gain access to the site. Which is achieved by retrieving the data from the database that was entered by users and verifying their account details.

Through completion, they will gain access to the main part of the website where they can view and add comments based on the squad selections. They will acquire their own unique username and can have discussions with other fans in this area of the site. Once a comment has been published by a user, it will display in the comments section. Which will include their username, the timestamp and give them the option to delete their comment if they want.

From here they will also have the option to change their password, delete their account and log out. Users may want to change their password to something more memorable or if want to make it more secure if they are worried about security issues. This feature should therefore give users confidence in using the site. Similarly, for deleting their account, this would wipe any information connected to a user off the system, or if they just have no more interest in using the site. A logout page is also important, this prevents people from going onto your account and posting harmful comments that

you did not intend to send. Without logging out, it could also be easier for hackers to identify personal information. I would also recommend not to allow Google so save credentials to prevent this.

Technical report

Drop down

This code selects the ID and team columns from the table 'teams' and orders them by team name. The drop-down menu is then created through "str_options". This is outputted through an echo statement in the html section further down.

```php
<?php
require_once "db_connect.php";
require_once "functionsnew.php";
include_once("header.php");

if (!$db_server){
die("Unable to connect to MySQL: " .
mysqli_connect_error());
$db_status = "not connected";
}else{
// If connected, get outlet names from DB and
write out DropDownMenu
mysqli_select_db($db_server, $db_database);
$query = " SELECT ID, team FROM teams ORDER BY
team";
$result = mysqli_query($db_server, $query);
if (!$result) die("Query failed: " .
mysqli_error($db_server));
while($row = mysqli_fetch_array($result)){
$str_options .= "<option value='" . $row['ID']
. "'>";
$str_options .= $row['team'];
$str_options .= "</option>";
}
```

```php
if (isset($_POST['team'])) {
  $team = clean_string($db_server,
  $_POST['team']);
  // create the SQL query
  $query = "SELECT teamID, position, name
  FROM worldcupsquads
  WHERE teamID=$team";
  // query the database
  mysqli_select_db($db_server,
  $db_database);
  $result = mysqli_query($db_server,
  $query);
  if (!$result) die("Database access
  failed: " . mysqli_error($db_server));
  // if there are any rows, print out the
  contents
  while ($row =
  mysqli_fetch_array($result)) {
     $output .= '<p>' . $row['name'] . ' '
     . $row['position'] . "</p>";
  }

  mysqli_free_result($result);
} else {
  $output = 'No team was requested';
}
```

This code uses the SELECT statement to get the player names and positions from the database, to then automatically insert them onto the website when a specific team has been chosen.

Each team has an ID, this is linked to the 'worldcupsquads' table where each player has a teamID associated with them. Therefore, the query will only display the players that have the same teamID because this is a specific team at the World Cup.

reCAPTHA

These lines of code ensure the reCAPTHA process works correctly.

```
if (!$db_server){
  die("Unable to connect to MySQL: " .
  mysqli_error($db_server));
  $db_status = "not connected";
}else{
  $captcha=$_POST['g-recaptcha-response'];
  $url =
  'https://www.google.com/recaptcha/api/siteve
  rify';
  $secretkey =
  "6Le4CAETAAAAAGQftFiDise1KTxFd6qTsowFR-TL";
  //secret key
  $response = file_get_contents($url."?
  secret=".$secretkey."&response=".$_POST['g-
  recaptcha-
  response']."&remoteip=".$_SERVER['REMOTE_ADD
  R']);
  $data = json_decode($response);
  $output = "";
```

Register

This code checks whether the username inputted already exists, if it does it will display an error message. ‘else’ statements are used for this and the ‘echo’ function is how the message is outputted to the user. If not and everything passes the query, the users details of username and password are ‘inserted’ into the ‘users’ table on the database. Meaning it can refer back to this data when users try and log in to verify account details.

gister.php

```
51  mysqli_select_db($db_server, $db_database);
52  // check whether username exists
53  $query="SELECT username FROM users WHERE
    username='$username'";
54  $result=mysqli_query($db_server, $query);
55  if ($row = mysqli_fetch_array($result)){
56  $message = "Username already exists. Please
    try again.";
57  }else{
58  // Process further here
59    $hash = password_hash($password,
      PASSWORD_DEFAULT);
60  $query = "INSERT INTO users (username,
    password) VALUES
61  ('$username', '$hash')";
62  mysqli_query($db_server, $query) or
63  die("Insert failed. ".
    mysqli_error($db_server));
64  $message = "<strong>Registration successful!
    </strong>";
```

login.php

```php
<?php
require_once("functions.php");
$username = trim($_POST['username']);
$password = trim($_POST['password']);
if ($username&&$password){
session_start();
require_once("db_connect.php");
mysqli_select_db($db_server, $db_database) or
die("Couldn't find db");
$username = clean_string($db_server,
$username);
$password = clean_string($db_server,
$password);
$query = "SELECT * FROM users WHERE
username='$username'";
$result = mysqli_query($db_server, $query);
if($row = mysqli_fetch_array($result)){
 $db_password = $row['password'];
 $db_id = $row['ID'];
 if (password_verify($password, $db_password))
 {
$_SESSION['username']=$username;
$_SESSION['userID']=$db_id;
$_SESSION['logged']="logged";
header('Location: home.php');
 }else{
$message = "<p>Incorrect password!</p>";
 }
}else{
 $message = "<p>That user does not exist!</p>"
```

Line 23. Column 41 — 62 Lines

Log in

This code uses the data inserted into the users table from registration. Using the $username variable to check if this is in the users table, and then the $password variable to verify this matches a user's ID. If successful, the session 'logged' will be activated and grant them access to the website.

Comments

This code allows comments to be outputted successfully. The query selects all from the comments table and then selects the date, ID, userID, the actual comment, username from the comments table and joins users associated with the comment through userID. The 'JOIN' function enables data from two tables to be combined. It will then fetch the array of data and display user's username and date along with their comment. This is possible through SQL, which enables the database tables to be joined, and links the users with the comments.

Further down the code checks whether the comment userID matches the session ID of the user. If matched, the user will be able to delete their comment from the website.

```php

mysqli_free_result($result);
// Print out existing comment
$query = "SELECT * FROM comments";
$query = "SELECT comments.commDate,
comments.ID, comments.userID,
comments.comment, users.username FROM comments
JOIN users ON
comments.userID = users.ID";
$result = mysqli_query($db_server, $query);
if (!$result) die("Database access failed:
".mysqli_error($db_server));
while($row = mysqli_fetch_array($result)){
$comments .= "<p><em>" . $row['username'] . ":
(" . $row['commDate'] .
")</em><br /> ";
//CHECK THAT THE COMMENT USERID MATCHES
SESSION USER ID
if ($row['userID'] == $_SESSION['userID']){
$comments .=" <a href='delete_post.php?
pID=".$row['ID']."'>Delete</a>";
}
$comments .= "<br />" . $row['comment'] . "
</p><hr />";
}


```

Change password

This works by when the change button is pressed, it checks whether the new password matches the repeat password. Then goes through the validation procedures of a password and if passed it will uses the session variable of the user to match the same username stored in the users table. Then the UPDATE query takes the old password variable and replaces it with the new password and ensures it is the right user through an ID.

changepassword.php

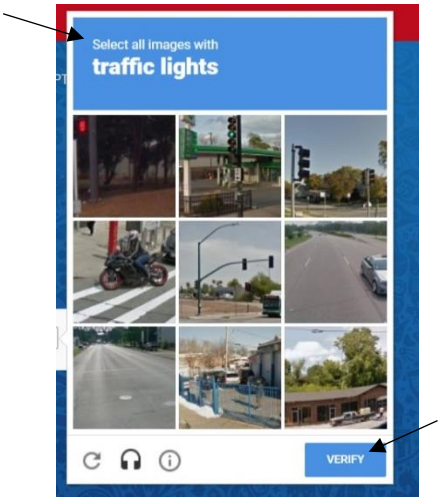
```
 9 ▼ if ($submit=='Change'){
10 ▼ if ($oldpassword&&$newpassword&&$repeatpassword){
11 ▼ if ($newpassword==$repeatpassword){
12   // Process username details here
13 ▼ if (strlen($newpassword)>25||strlen($newpassword)<6) {
14   $message = " <h3> Password must be 6-25 characters long</h3>";
15 ▼ }else{
16   // Process details here
17   require_once("db_connect.php"); //include file to do db connect
18 ▼ if($db_server){
19   //clean the input now that we have a db connection
20   $oldpassword = clean_string($db_server, $oldpassword);
21   $newpassword = clean_string($db_server, $newpassword);
22   $repeatpassword = clean_string($db_server, $repeatpassword);
23   mysqli_select_db($db_server, $db_database);
24   // check whether username exists
25   $query = "SELECT * FROM users WHERE username='" .
     $_SESSION['username'] . "'";
26   $result = mysqli_query($db_server, $query);
27 ▼ if($row = mysqli_fetch_array($result)){
28   $db_password = $row['password'];
29   $db_id = $row['ID'];
30 ▼ if (password_verify($oldpassword, $db_password)) {
31
32
33   // Encrypt password
34   $hash = password_hash($newpassword, PASSWORD_DEFAULT);
35   $query = "UPDATE users SET password = '$hash' WHERE ID = $db_id";
36   mysqli_query($db_server, $query) or
37   die("Update failed. ". mysqli_error($db_server));
38   $message = " <h3> <strong>Password change successful!</strong>
     </h3>";
39   }
```

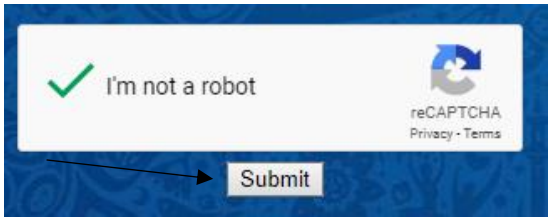
User guide

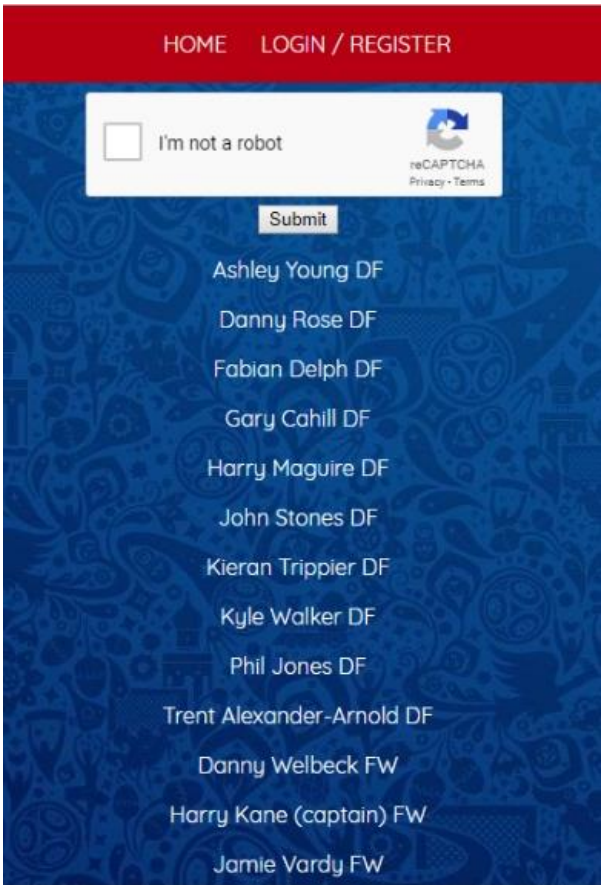
Homepage

1. From here, they will be on the homepage of the website. To use the form, they must click the drop-down menu. This will bring down a list of team names. They can select a team they want to view, click the square box where it says, “I’m not a robot”, wait for it to tick green, and then press submit. If it doesn’t tick green, the user will have to follow the instructions on screen to verify they are human, once completed the box will tick green. The user will then be able to see all the players that have been selected for that team at the World Cup.


I'm not a robot
reCAPTCHA
Privacy - Terms
Submit

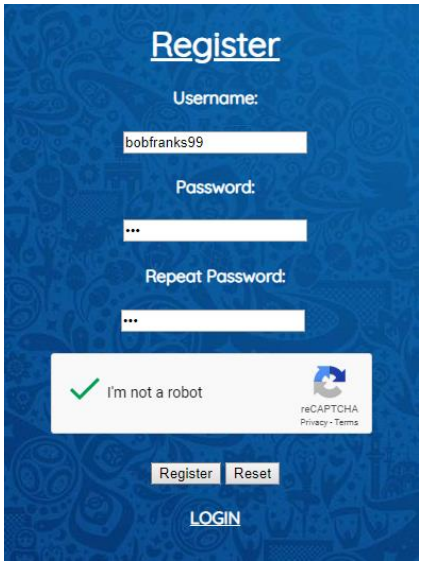

Select all images with
traffic lights
VERIFY

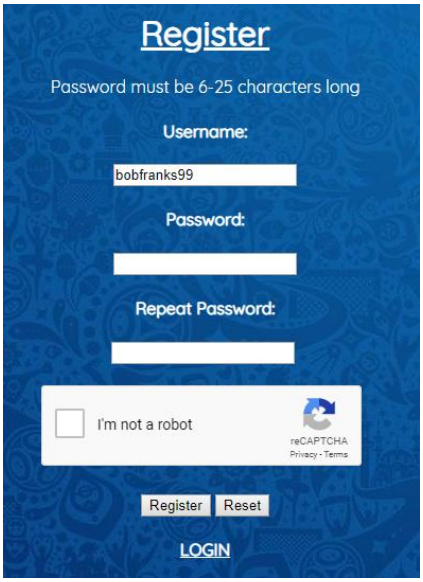

I'm not a robot
reCAPTCHA
Privacy - Terms
Submit

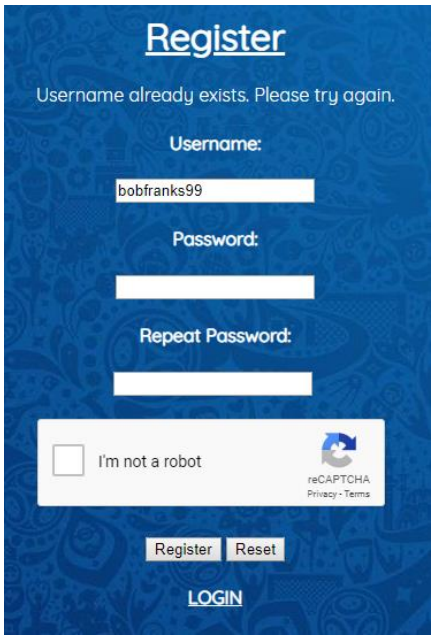

HOME
LOGIN / REGISTER
I'm not a robot
reCAPTCHA
Privacy - Terms
Submit
Ashley Young DF
Danny Rose DF
Fabian Delph DF
Gary Cahill DF
Harry Maguire DF
John Stones DF
Kieran Trippier DF
Kyle Walker DF
Phil Jones DF
Trent Alexander-Arnold DF
Danny Welbeck FW
Harry Kane (captain) FW
Jamie Vardy FW

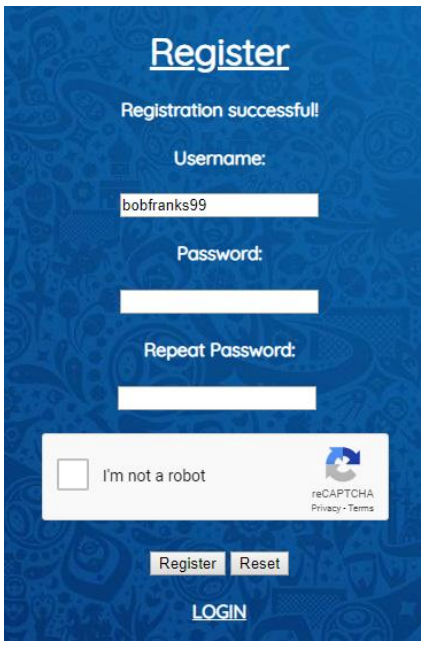
Register

2. If users want to view or add comments based on this, they must go to the register by clicking the navigation bar or the link at the bottom of the page. Users must give themselves a username and a password to gain access. As shown below, users must enter a unique username that no other user has, a password between 6-25 characters and that these must match. Once verification has been passed, the user will hold an account with the website and proceed to log in.



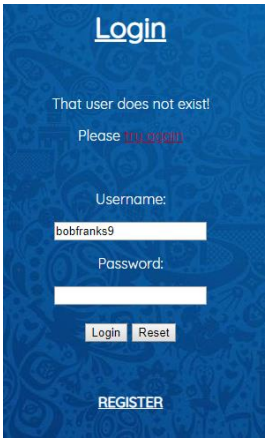


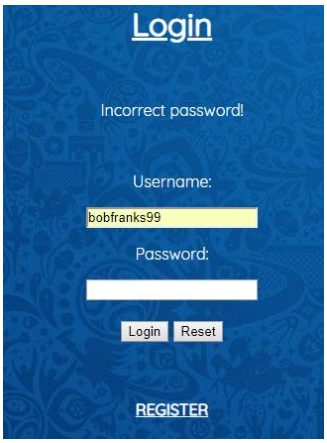


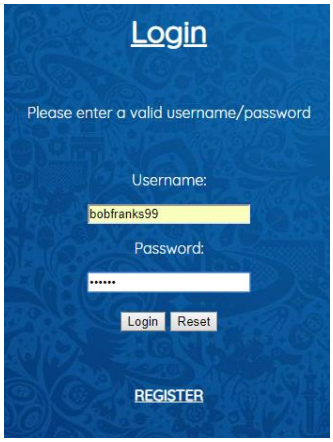


Log in

3. The user proceeds to enter their details. As shown they must exactly match to what the user registered with otherwise they won't be able to access the full website. Once entered correctly, they will gain access to the comments page.

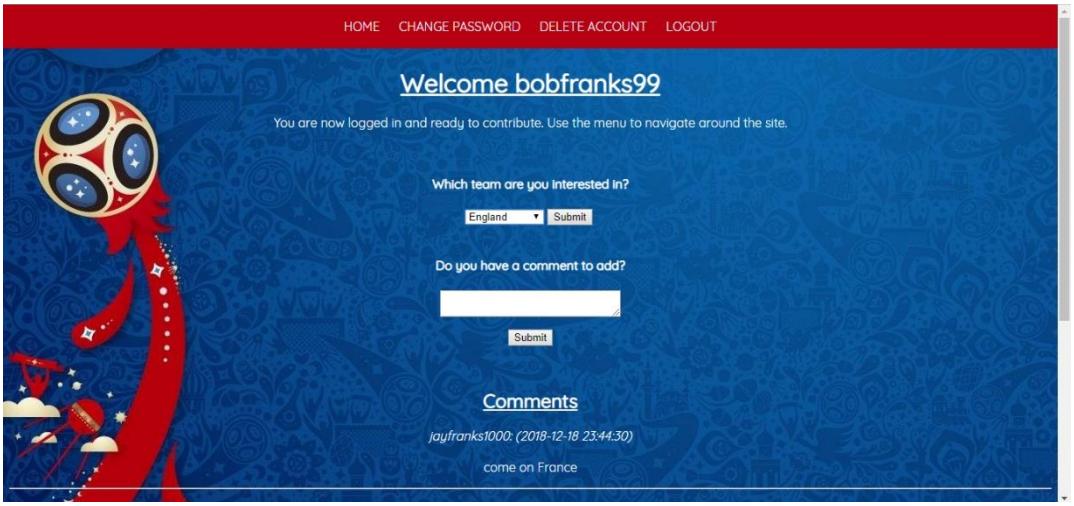






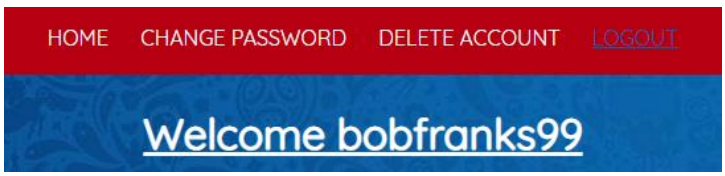
Comments page

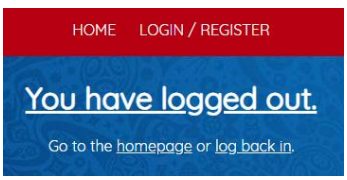
4. From here, users can use the form again to refresh themselves with the players that have been selected for the squads. They can then scroll down and type a comment in the white box and press submit. The comment will then appear below, and users can easily delete their comment by clicking the delete button and then it will no longer appear in the comments section. This should be obvious because "when affordances are taken advantage of, the user knows what to do just by looking" (Norman,1998). The red underlined text clearly identifies it as a function that is usable to the user.



LOGOUT

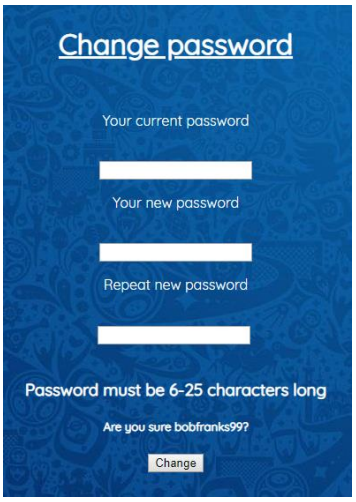
5. Once finished using the site, users can easily logout by clicking the logout button on the navigation bar. This will transfer them to a page confirming they have logged out and allow to go to the homepage or log back in.

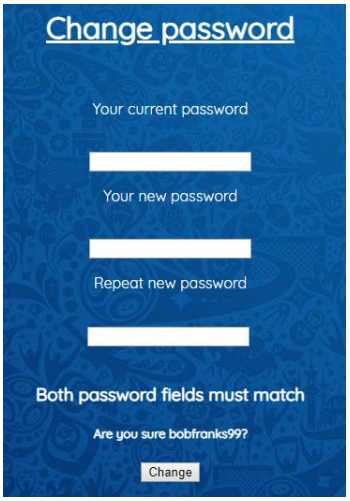


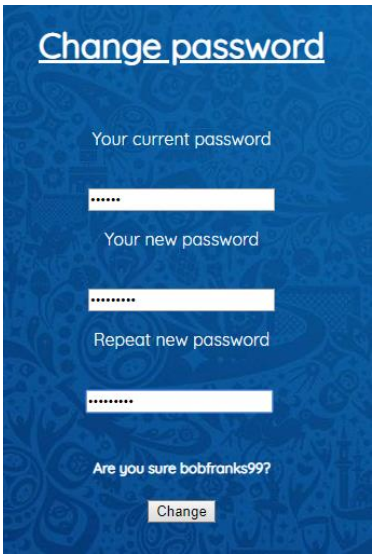


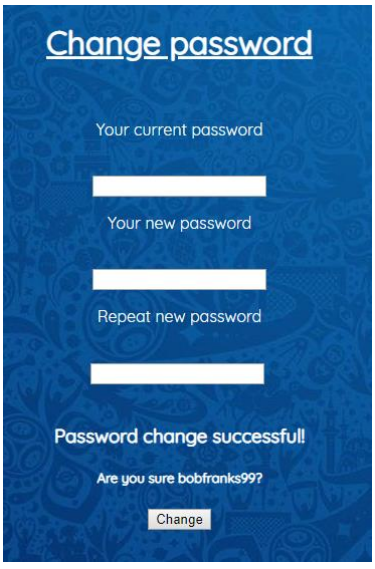
Change password

6. For users to change their password, they can use the “change password” link on the navigation bar. Same verification techniques are applied to ensure a valid password is inputted. When successfully completed it will alert the user “password change successful!”, and then can log in with their new password.





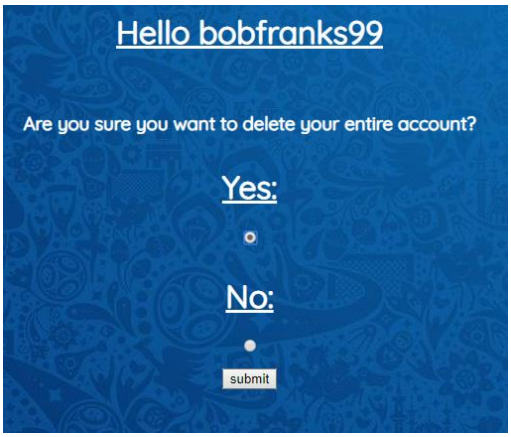


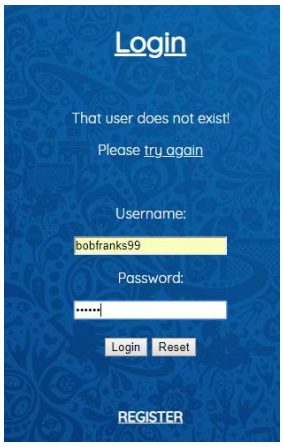


Delete account

7. To delete your account, simply click the delete account link on the navigation bar, and click yes, and press submit. All account details will then be wiped off the system, as shown below.

.



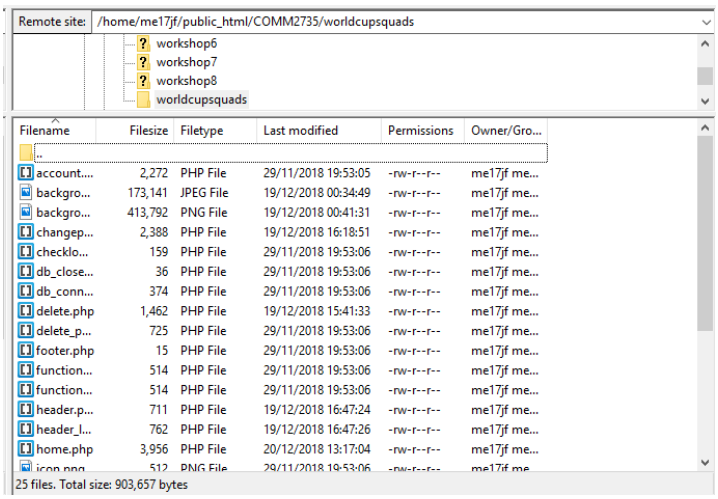


Administration

8. To enable someone to take over the website, they would need to access cPanel. This is so they can access the database and edit the data through a back-end function. This would be important in case the squad list changed due to a player's injury. Meaning that a player's data can be removed and replaced with another player. Moreover, they will have the ability to manually delete comments/remove user accounts.

Amending the website site

This is achieved by using a file transfer protocol such as FileZilla, to update the website. They would connect to the remote site me17jf@leedsnewmedia.net and update the files by uploading to them to the 'worldcupsquads' folder as shown.



REFERENCES

Forbes. 2017. *Why Do We Use Usernames And Passwords?* [Online]. [Accessed 18th December 2018]. Available from: https://www.forbes.com/sites/forbestechcouncil/2017/10/31/why-do-we-use-usernames-and-passwords/#541fd4a01fda

Krug, Steve. 2005. *Don't Make Me Think, Revisited: A Common-Sense Approach to Web Usability*. Amazon (3rd ed.). New Riders.

Norman, Donald A. 1998. *The Invisible Computer: Why Good Products Can Fail, the Personal Computer Is So Complex and Information Appliances Are the Solution*. MIT Press.