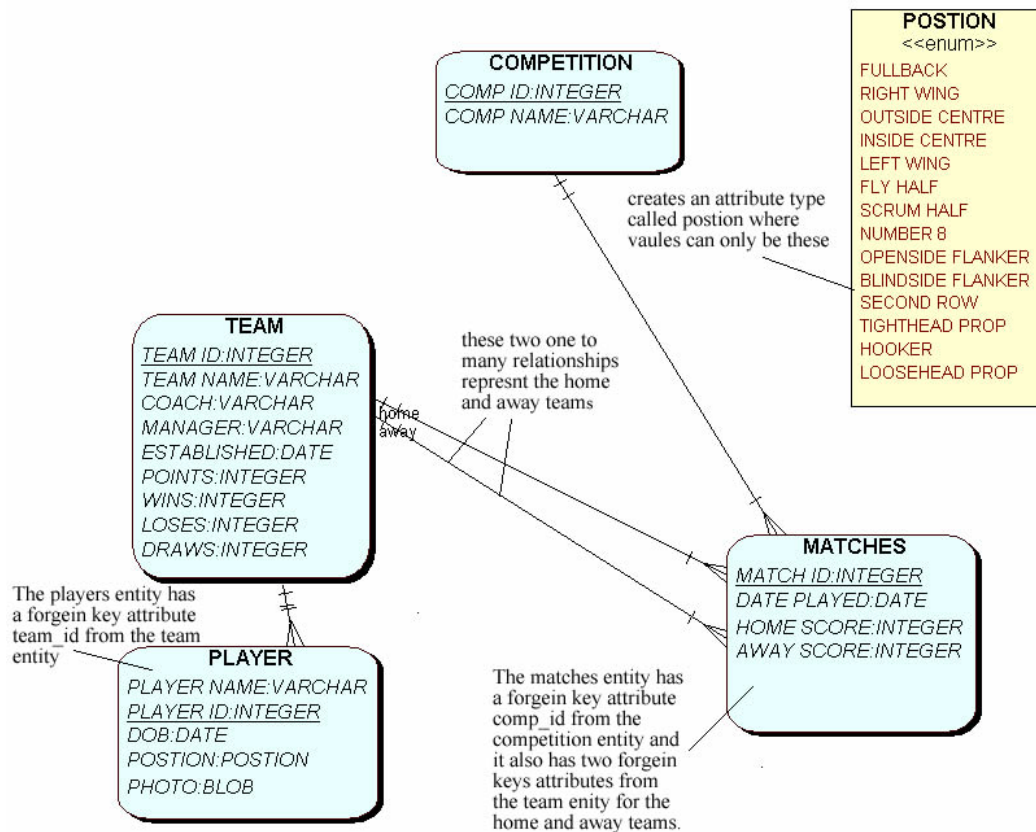


Development documentation

Database Design

I decided that I was going to develop an application to store all the details of the 6 nation's rugby competition. The first stage of design and development process was to design an entity relationship diagram to represent the relationships between all the entities and their attributes required for the database to store all the details of the competition. I designed the following entity relationship diagram in QSEE.

6 NATIONS RUGBY ENTITY RELATIONSHIP DIAGRAM



Once I had designed the entity relationship diagram to hold all the details for the competition I generated the SQL code to create the tables from QSEE.

I had to modify the code slightly to make the primary keys for each entity increment up by one for each entry that was put into the tables.

Here shows the code for the creation of the competition table with the “AUTO_INCREMENT” statement applied to the primary key.

```
-- Create a Database table to represent the "COMPETITION" entity.
CREATE TABLE COMPETITION(
    COMP_ID      INTEGER NOT NULL AUTO_INCREMENT,
    COMP_NAME    VARCHAR(20) NOT NULL,

    -- Specify the PRIMARY KEY constraint for table "COMPETITION".
    -- This indicates which attribute(s) uniquely identify each row of data.
    CONSTRAINT  pk_COMPETITION PRIMARY KEY (COMP_ID)
) TYPE=INNODB;
```

Once I had generated the SQL script I created the SQL database and filled it with some test data.

Front end application design

Now that I had created the database I set about making a simple html page which incorporated some PHP for the front end of the website. I used CSS to style the page and specify the colour of hyperlinks. I then created a HTML form for selecting teams and dynamically filled the form with the team options using a PHP function to query the SQL database and extract the data. The code for the form is below.

```
<form onSubmit="getTeam(); return false">
  <span><label for="TEAM">Select a Team</label>
    <select name="team" id="team"
      onchange="getTeam();"
      <option selected="yes">Select a Team</option>
  </select>
  <?php
    $link = db_connect();
    team_options($link);
    mysql_close($link);
  ?>
</span>
</form>
```

The form incorporates a request to the PHP function “team_options” and passes it the variable “\$link” which is created by calling the function “db_connect()” The code for the db_connect function is below.

```
function db_connect() {
// login to the remote database
// return a link to the database to use in subsequent queries
  $server = 'localhost';
  $user = 'root';
  $password = 'tony';
  $db = "rugby";
  $dblink = mysql_connect($server,$user,$password) ;
```

05973108

```
// select the database to use
mysql_select_db($db,$dblink);
return $dblink; }
```

The `db_connect()` function returns a variable with the value of a “`mysql_connect`” statement to access the database.

Once the “`team_option`” PHP function is called it queries the database to find all the teams by using the SQL command “`select * from TEAM`” it then returns the results from the query incorporated into a string using the “`print`” command so that they can be understood by the form as options. The code for the “`team_options` function is below.

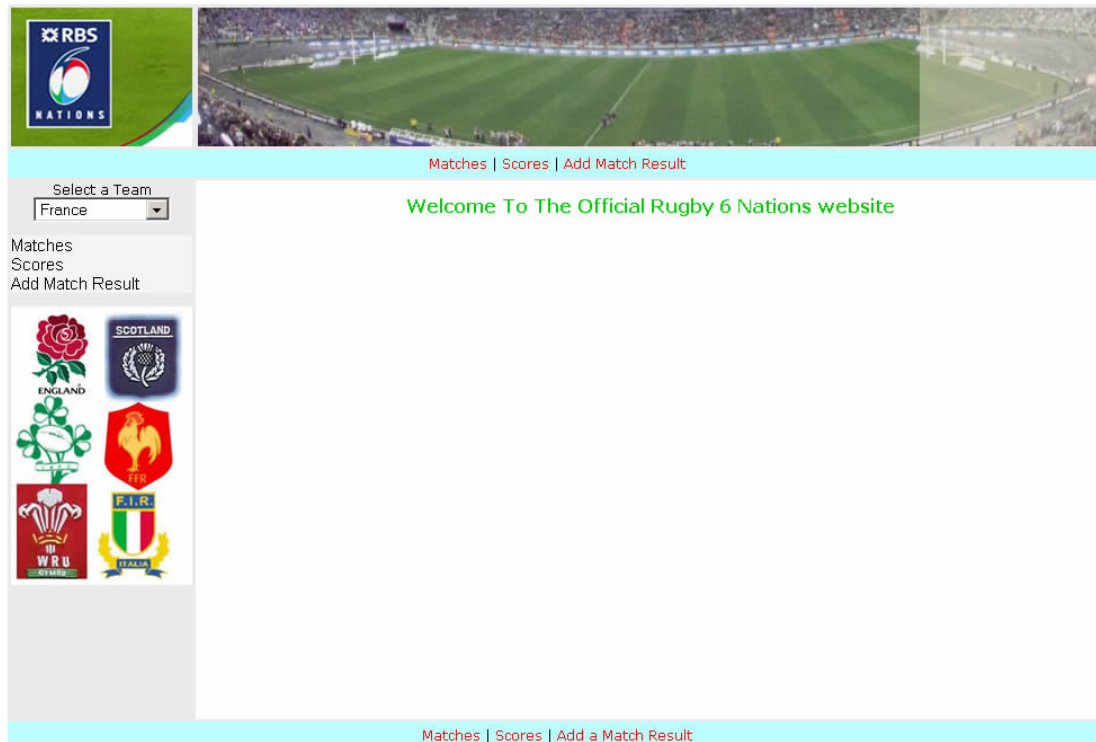
```
function team_options($dblink) {
    $query = "select * from TEAM;";
    $dbresult = mysql_query($query,$dblink);
    while ( $team= mysql_fetch_object($dbresult)) {
        print "<option value='$team->TEAM_NAME'>$team->TEAM_NAME</option>\n";
    }
}
```

I created an empty table in the main area of the home page where information is to be displayed and gave each element of the table `div ID`'s so they can be referenced by `AJAX` commands to insert another PHP page. The code for these tables is below

```
<table>
  <tr>
    <TD VALIGN="top">
      <div id="player"/>
    </td>
    <TD VALIGN="top">
      <div id="playerdetails"/>
    </td>
    <TD VALIGN="top">
      <div id="div3"/>
    </td>
  </tr>
</table>
```

These 3 elements of the table are where information is going to be displayed upon request from the user of the application.

The Index.php page so far looks like this:-



Ajax design

The form that I had created earlier calls an AJAX JavaScript function once submitted to display team details “<form onSubmit="getTeam(); return false">” It calls the function ‘getTeam’ from a JavaScript file called rugbyajax.js

Here is the code for this function:-

```
function getTeam() {
  if (!isWorking && http) {
    var team = document.getElementById("team").value;
    http.open("GET", "listteam.php?teamname=" + team);
    http.onreadystatechange = updateTeam;
    // this sets the call-back function to be invoked when a response from the HTTP request is
    returned
    isWorking = true;
    http.send(null);
  }
}
function updateTeam() {
  if (http.readyState == 4) {
    var div = document.getElementById('player');
    div.innerHTML = http.responseText;
    var playerdetails = document.getElementById('playerdetails');
    playerdetails.innerHTML= "<p/>";
    var div3 = document.getElementById('div3');
```

05973108

```
div3.innerHTML= "<p/>";  
isWorking = false;  
}  
}
```

The function then makes an ajax http.open request to get listteam.php and supplies it with the parameter team from the form “http.open("GET", "listteam.php?teamname=" + team);” It then calls the function “updateTeam()” to load the listteam.php page into the div element where the id=player in the main index.php page.

The result is a page that looks like this:-



The listteam.php page creates a table with all the details of that team, It does this by using the value it was passed and querying the database to get the correct result. It queries the SQL database using this statement:-

```
“select TEAM_NAME, COACH, MANAGER, ESTABLISHED, TEAM_ID from TEAM where  
TEAM_NAME='$teamname’”
```

It then fetches the results and puts them into a table to be displayed using this code:-

```
print "<tr><th>Team Name</th><td>$team->TEAM_NAME</td></tr>";  
print "<tr><th>Coach</th><td>$team->COACH</td></tr>";  
print "<tr><th>Manager</th><td>$team->MANAGER</td></tr>";  
print "<tr><th>Established</th><td>$team->ESTABLISHED</td></tr>";  
print "<tr><th>Players</th><td><a href='javascript:getPlayer($team->TEAM_ID)'>view  
Players</td></tr>";
```

```

print "<tr><th>Matches</th><td><a href='javascript:getMatches($team->TEAM_ID)'/>view
Matches</td></tr>";

}
print "</table>";
}

```

It also uses the values it fetches to create links to other AJAX functions in the rugbyajax.js script passing them the parameters of these values to display other PHP pages upon request from the user.

The rest of the site works in much the same way, calls are made to AJAX functions passing them a parameter then the AJAX function makes a request to get a PHP page passing it the parameter, the PHP page then queries the database using the parameter and fetches results from that query and formats them. The AJAX function then sends the response from its call to get the PHP page into a div element in the index.php page.

Adding a match result data validation

I created a form to add a match result and to also update the competition points table as shown in the picture below:-

The screenshot shows the RBS Nations website interface. At the top left is the RBS Nations logo. To its right is a wide image of a rugby stadium. Below the stadium image is a navigation bar with links: [Matches](#) | [Scores](#) | [Add Match Result](#). The main content area has a green header that says "Welcome To The Official Rugby 6 Nations website". On the left side, there is a sidebar with a "Select a Team" dropdown menu showing "Wales". Below this are links for "Matches", "Scores", and "Add Match Result". Further down is a grid of team logos for England, Scotland, Wales, and Ireland. The main form area contains:

- "Select Home Team" dropdown menu with "England" selected, followed by a "Home score" input field.
- "Select Away Team" dropdown menu with "England" selected, followed by an "Away score" input field.
- "Date played eg 'YYYY-MM-DD'" followed by three input fields for year, month, and day, and an "Add Match" button.

 At the bottom of the page is another navigation bar with links: [Matches](#) | [Scores](#) | [Add a Match Result](#).

To make sure that no incorrect data was inserted into the database for example where two teams had played each other, text was input into the score field or a date was invalid I took a number of measures to prevent these errors. Firstly I

used a java script function to only allow numbers to be input into the fields of the form. The function is shown below:-

```
<SCRIPT TYPE="text/javascript">
<!--
function numbersonly(myfield, e, dec)
{
var key;
var keychar;

if (window.event)
    key = window.event.keyCode;
else if (e)
    key = e.which;
else
    return true;
keychar = String.fromCharCode(key);

// control keys
if ((key==null) || (key==0) || (key==8) ||
    (key==9) || (key==13) || (key==27) )
    return true;

// numbers
else if (("0123456789").indexOf(keychar) > -1))
    return true;

// decimal point jump
else if (dec && (keychar == "."))
    {
    myfield.form.elements[dec].focus();
    return false;
    }
else
    return false;
}

//-->
</SCRIPT>
```

The ‘numbersonly’ function was called by the form by using the ‘onKeyPress’ command as shown here

```
<input name="dateplayed3" SIZE=2 MAXLENGTH=2
onKeyPress="return numbersonly(this, event)"/>
```

I also set the sizes of the form fields to be no bigger than required size of data input to prevent user error. Once the form was filled out and the data was posted to the PHP page for processing I put in a date validation check and a check to see if the two teams selected were the same. If either of these checks failed to meet the requirements of the function to add a new match result then the user would be presented with an error message explaining why the result could not be added.

This is the PHP function I used to check if the date entered into the form was valid:-

```
function is_valid_date($datestring) {
list($y, $m, $d) = explode('-', $datestring);
list($yy, $mm, $dd) = explode('-', date('Y-m-d',
strtotime($datestring)));
if (($y == $yy) and ($m == $mm) and ($d == $dd)) {
return true;
} else {
return false;
}
}
```

To check if both teams selected were the same I just used a simple if statement to compare them both

Adding a match result and updating points table

If the data posted to the process.php page from the form had passed the validation checks then they would be inserted into the database using the following SQL command:-

```
"INSERT INTO MATCHES
(
DATE_PLAYED,HOME_SCORE,AWAY_SCORE,FK1_COMP_ID,FK2_TEAM_ID,FK3_TEAM_ID)
VALUES ('$dateplayed','$hscore','$ascore',1,$steamresult->TEAM_ID,$hteamresult->TEAM_ID);"
```

As well as adding the match result to the matches table in the database the process.php script would also update the team table with the teams points they had gained from the match as well as there wins, loses and draws. To update the points they had gained from the match I queried the scores to see which team had won the match, if a team won they would get 3 points if they draw they would get 1 point and if they lost no points. I know this probably is not the correct official way the points are calculated for the 6 nations but this is just to demonstrate the processing. The code that does this is shown here:-

```
if ($hscore==$ascore){
$hpoinst = 1; $apoinst = 1; $hdraw = 1; $adraw = 1;}

if ($hscore>$ascore){
$hpoinst = 3; $apoinst = 0; $hwin = 1; $alose = 1;}

if ($ascore>$hscore){
$apoinst = 3; $hpoinst = 0; $awin = 1; $hlose = 1;}
```

05973108

I would then update the team's points by creating a query to get the teams current points and fetch this result, then add on the new points earned to that value and update the team points field with the new value. As shown here:-

```
//gets the home teams points and adds on the points gained from the new match
    $hteampoints = "select POINTS from team where TEAM_ID=$hteamresult->TEAM_ID;";
    $hpoint = mysql_query($hteampoints,$dblink);
    $hpointr = mysql_fetch_object($hpoint);
    $newhpoints = $hpointr->POINTS + $hpoints;

//updates the home teams points
    $hteampupdate = "update TEAM set POINTS = $newhpoints where TEAM_ID=$hteamresult-
>TEAM_ID;";
    mysql_query($hteampupdate,$dblink);
```

To update the wins, loses and draws for each team I used the same functions as above just replaced with the correct values.