



Trabalho Realizado Por:

Ana Fidalgo, 46704

Cristina Rocha, 46396

Diana Condeço, 46748

Jesebelle Rocha, 39571

Joana Cavaco, 40624

Índice

1. Introdução	3
1.2. Objectivo do documento.....	4
1.2. Visão geral do projecto	4
3. Implementação	7
3.1. Descrição das principais funcionalidades da aplicação Web	7
3.2. Mapa de páginas	7
3.2.1. Server Behaviours utilizados	7
3.2.2. Recordsets/Queries utilizados	10
3.2.3. Parâmetros passados entre Páginas	13
3.2.4. Outros Contéudos	18
4. Desenvolvimentos Futuros.....	23
5. Conclusões.....	24
6. Referências Bibliográficas	25

1. Introdução

A realização deste projecto consolida as matérias leccionadas nas disciplinas “Laboratório Multimédia 5”, “Implementação e Controlo de Projectos Multimédia” e “Gestão de Empresas” de forma a fortalecer os conhecimentos adquiridos em contexto de aulas numa avaliação conjunta.

No âmbito da disciplina de Laboratório Multimédia 5 pretende-se desenvolver as competências dos alunos, relativamente às componentes de base de dados relacionais e páginas dinâmicas para a Web, relacionando estas componentes e explorando as várias funcionalidades do software “*MySQL Workbench 5.2 CE*” e “*Adobe Dreamweaver CS3*”.

Neste contexto, a ideia que o grupo optou passa por criar uma plataforma para um centro de explicações - “*ExplainIt*”, com o intuito do esclarecimento de dúvidas entre explicadores e os vários usuários (explicandos).

Deste modo, pretende-se elaborar o processo construtivo da plataforma, partindo das ideias iniciais, passando pelas estratégias adoptadas e culminando no trabalho final.

O presente documento encontra-se organizado por capítulos, sendo estruturados da seguinte forma: introdução, em que se explanam os objectivos deste documento bem como a visão geral de todo o projecto. A base de dados relacional, mostra as entidades existentes e as relações entre as mesmas, as tabelas que foram criadas e os seus respectivos campos; a implementação da aplicação, com as funcionalidades fulcrais, como por exemplo *recordsets*, *queries*, entre outros; desenvolvimentos futuros com sugestões daquilo que poderá ser melhorado posteriormente; conclusões finais acerca do projecto e por fim as referências bibliográficas à qual o grupo se apoiou.

1.2. Objectivo do documento

Neste documento pretende-se contextualizar o conceito do projecto “*ExplainIt*” e o desenvolvimento do mesmo durante a sua execução. Outro dos objectivos deste documento prende-se com o registo do trabalho, nomeadamente a implementação do mesmo desenvolvido com os conhecimentos adquiridos durante o semestre.

Logo, o presente documento pretende clarificar em primeiro, em que consiste a ideia do projecto, seguida da planificação da base de dados desenvolvida e a implementação do projecto em *DreamWeaver*.

Ainda neste documento, pretende-se também registar as dificuldades sentidas, os problemas que ocorreram durante o desenvolvimento da plataforma e ainda se esses mesmos problemas foram resolvidos ou que técnicas de solução foram adoptadas.

1.2. Visão geral do projecto

O projecto que o grupo desenvolveu visa a criação de uma plataforma para um centro de explicações - “*ExplainIt*”, com o intuito do esclarecimento de dúvidas entre explicadores e os vários usuários (explicandos). Esta plataforma está dividida em várias comunidades que correspondem a disciplinas – Matemática, Ciências, Programação e Línguas – em que os usuários se podem inscrever. Estes podem ter acesso aos conteúdos disponíveis na aplicação (exercícios, matéria, e-books, manuais, etc.), colocar questões a um explicador e também trocar informações com outros usuários, atribuindo uma classificação aos explicadores. Consoante o número de comunidades/disciplinas em que o explicando está inscrito, será atribuído um valor mensal a pagar.

Nos dias de hoje torna-se importante acompanhar as novas tecnologias de modo a estarmos actualizados. Neste sentido, o objectivo da criação do “*ExplainIt*” passa por proporcionar uma maior interactividade, onde se possa partilhar e difundir tanto os conhecimentos como as dúvidas que possam surgir. Além disso, sendo o público-alvo muito abrangente, pretende-se que a plataforma vá ao encontro destes, colocando-os à vontade com as novas tecnologias.

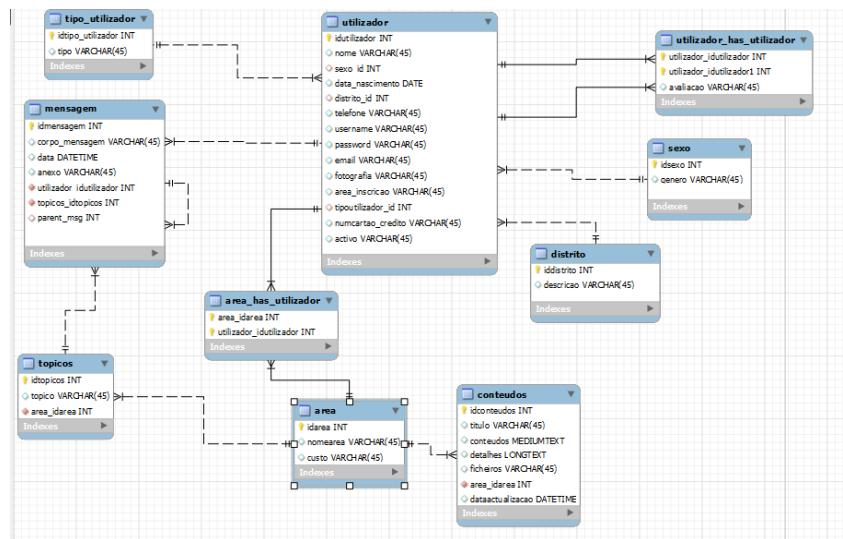
Este projecto destaca-se em relação à concorrência uma vez que nesta área existe uma escassez de plataformas deste tipo.

2. Base de Dados Desenvolvida

- Entidades e Relações

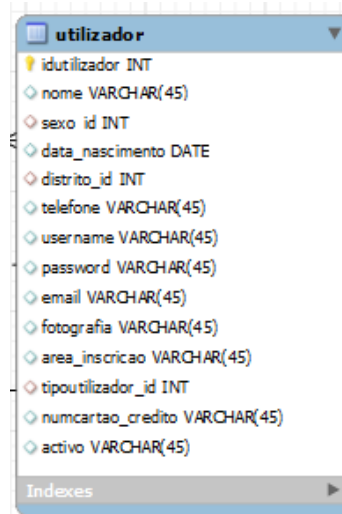
A base de dados desenvolvida no projecto “*ExplainIt*” é constituída pelas seguintes entidades: utilizador, tipo de utilizador, mensagem, tópicos, área, conteúdos, explicação, explicador e avaliação.

As relações existentes entre entidades podem ser visualizadas na imagem a seguir:

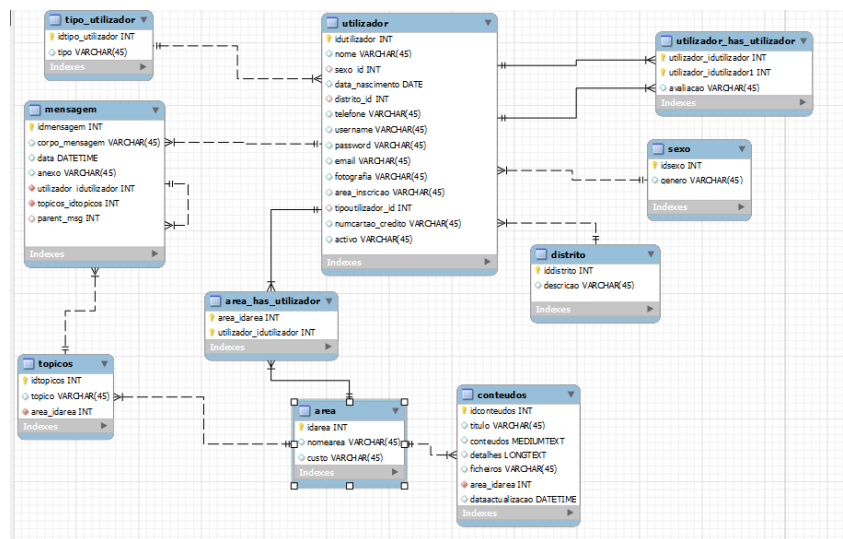


- Tabelas e Campos

Em cada coluna da tabela definiu-se o tipo de dados e a parametrização utilizada. Deste modo, obteve-se uma maior eficiência, optimização e desempenho da base de dados, das tabelas e dos dados aí armazenados.



- Diagrama ER (MySQL Workbench)



Tal como já foi mostrado anteriormente a imagem anterior é representativa da base de dados do ExplainIt.

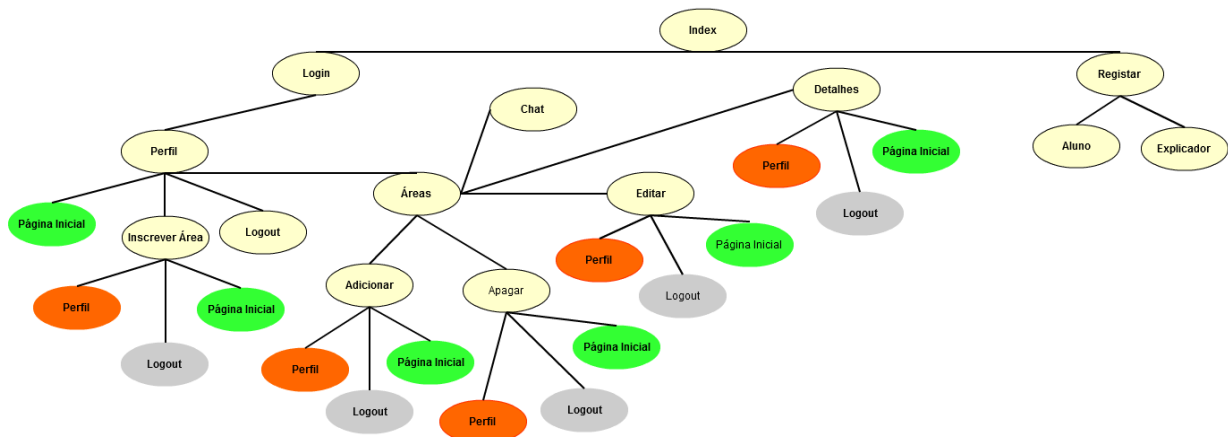
3. Implementação

3.1. Descrição das principais funcionalidades da aplicação Web

As principais funcionalidades que a aplicação web permite aceder são registo de utilizador, onde o explicador ou o aluno podem realizar o login e logout. Quando o utilizador realiza o login pode aceder à sua conta pessoal, às suas áreas, perfil, horário e acesso ao chat. Após a navegação o utilizador poderá terminar sessão de modo a garantir a confidencialidade e segurança dos seus dados. Além disso, existe a possibilidade de sugerir o *website* a amigos, de editar os dados pessoais, de carregar conteúdos, do campo de pesquisa, entre outros.

Os utilizadores têm diferentes acessos/participações. O administrador além de ter acesso a todas as áreas é quem controla toda a aplicação. Os utilizadores não registados apenas têm acesso à página inicial, enquanto os registados podem navegar em todas as áreas em que se inscrevem. No entanto estes não podem apagar publicações, algo apenas permitido ao administrador.

3.2. Mapa de páginas



3.2.1. Server Behaviours utilizados

- Página de Perfil

Na página de perfil foi necessário fazer uma restrição para passar os valores relativos ao utilizador que iniciou a sessão.

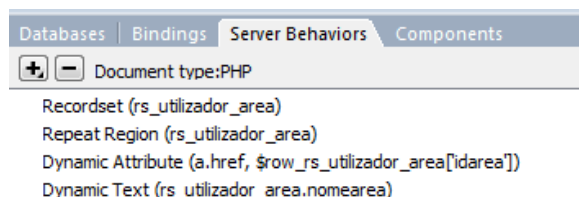
```
$colname_rs_dados = "-1";  
if (isset($_SESSION['MM_Username'])) {  
    $colname_rs_dados = $_SESSION['MM_Username'];  
}
```

De maneira a imprimir a informação na página, os campos do *behaviors* foram arrastados manualmente para as respectivas posições no ecrã.

```
"left"><h1>Bem-vindo <?php echo $row_rs_dados['username']; ?></h1></div>
"left"><h2>Dados Pessoais do <?php echo $row_rs_dados['tipo']; ?></h2>
:left">&Eacute; natural de <?php echo $row_rs_dados['descricao']; ?></p>


```

- Página de menu à direita

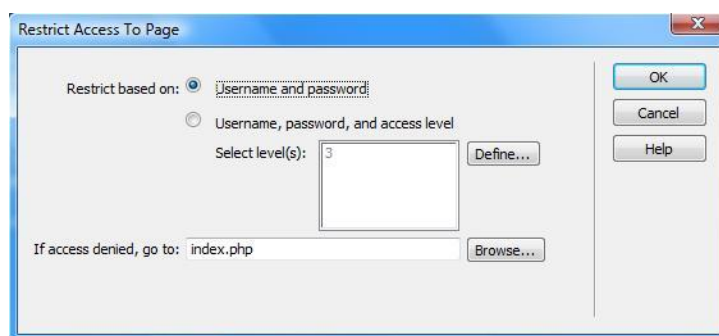


```
</tr>
<?php do { ?> <tr>

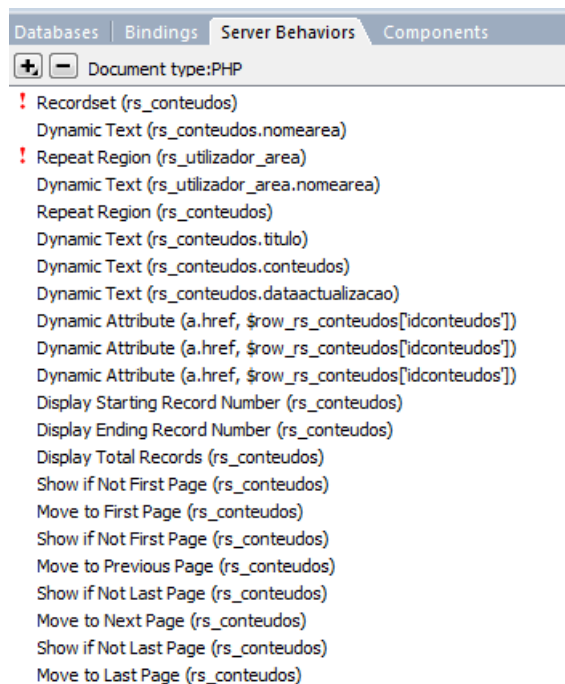
    <td class="menu2"><p align="center"><a href="areas.php?id_area=<?php echo $row_rs_utilizador_area['idarea']; ?>"><?php echo
$row_rs_utilizador_area['nomearea']; ?></a></p></td>
</tr> <?php } while ($row_rs_utilizador_area = mysql_fetch_assoc($rs_utilizador_area)); ?>
```

- Página das áreas

Na página de perfil do utilizador foi necessário fazer uma restrição para que se passassem apenas os dados registados na base de dados relativos ao utilizador que iniciou sessão.

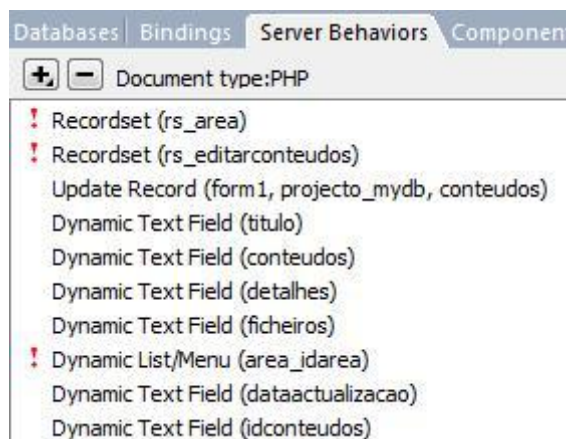


Os *server behaviors* criaram-se à medida que os campos do *recordset* foram arrastados para o ecrã de forma a passar a informação respectiva de cada *post* guardada na base de dados.



- Página de Editar Post

Foram automaticamente criados os *behaviors* da página de actualização dos conteúdos.



- Página de Adicionar um post

De modo a introduzir informação na base de dados recorreu-se à funcionalidade *Insert Record Form Wizard* onde foi configurado o formulário para introduzir conteúdos.

Record Insertion Form

Connection: projecto_mydb

Table: conteudos

After inserting, go to: Browse...

Form fields: + - ▲ ▼

Column	Label	Display As	Submit As
titulo	Titulo:	Text field	Text
conteudos	Conteudos:	Text area	Text
detalhes	Detalhes:	Text area	Text
ficheiros	Ficheiros:	Text field	Text
area_idarea	Area_idarea:	Menu	Numeric
dataactualiza	Dataactualizacão:	Text field	Date

Label: Detalhes:

Display as: Text area Submit as: Text

Default value:

OK Cancel Help

Databases Bindings Server Behaviors Components

Document type: PHP

- Recordset (rs_areas)
- Insert Record (form1, projecto_mydb, conteudos)
- ! Repeat Region (rs_utilizador_area)
- Dynamic Text (rs_utilizador_area.nomearea)
- Dynamic List/Menu (area_idarea)
- Dynamic Text Field (dataactualizacao)

3.2.2. Recordsets/Queries utilizados

- Pesquisa:

A query utilizada para a pesquisa é a seguinte:

```
mysql_select_db($database_projecto_mydb, $projecto_mydb);
$query_rs_conteudos = "SELECT * FROM conteudos WHERE conteudos LIKE '%" .
$_POST['input'] . "%' OR titulo LIKE '%" . $_POST['input'] . "%' OR detalhes
LIKE '%" . $_POST['input'] . "%' ORDER BY dataactualizacao DESC";
```

Permite seleccionar o título, os conteúdos e os detalhes dos posts da tabela “conteudos”. Através da variável `$_Post`, vai-se buscar o *input* do formulário da pesquisa para que se encontre a semelhança entre os dados inseridos e os conteúdos da tabela.

- Página de Adicionar uma Área (Query que dá informação do INNER JOIN e Insert Record)

```
mysql_select_db($database_projecto_mydb, $projecto_mydb);
query_rs_area = sprintf("SELECT bd_projecto.area.nomearea, bd_projecto.area_has_utilizador.* FROM bd_projecto.area_has_utilizador Inner Join
d_projecto.area On bd_projecto.area_has_utilizador.area_idarea = bd_projecto.area.idarea WHERE utilizador_idutilizador = %s",
etSQLValueString($colname_rs_area, "int"));
rs_area = mysql_query(query_rs_area, $projecto_mydb) or die(mysql_error());
row_rs_area = mysql_fetch_assoc($rs_area);
totalRows_rs_area = mysql_num_rows($rs_area);
```

- Página Registo

Para o preenchimento do formulário na página registo foi necessário criar três recordset para fazer a ligação às tabelas da base de dados.

Recordset

Name:

Connection: Define...

Table:

Columns: ☒ All ☐ Selected:

Filter: =

Sort:

OK
Cancel
Test
Advanced...
Help

Recordset

Name:

Connection: Define...

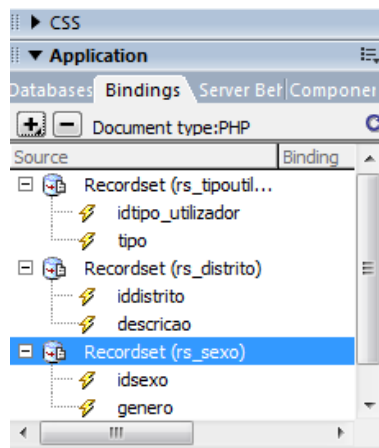
Table:

Columns: ☒ All ☐ Selected:

Filter: =

Sort:

OK
Cancel
Test
Advanced...
Help



3.2.3. Parâmetros passados entre Páginas

- Perfil (convidar amigo)

Para se convidar o amigo foi necessário criar um formulário onde é possível o utilizador fazer o convite através do nome e email do amigo. O processamento do formulário é realizado na mesma página através do parâmetro do **\$_Post** e do que foi estipulado na action.

Para que sejam reconhecidos os valores passados no **\$_Post** foi preciso criar variáveis com a indicação da passagem desses parâmetros através do método post. A mensagem que surge vai ser reconhecida através dessas variáveis.

```
236
237 <form action="<?php echo $_SERVER["PHP_SELF"]; ?>" method="post" name="formulario_amigo" id="amigo" onsubmit=
```

```
133
134 <?php
135 $nome= $_POST["nome"];
136 $email= $_POST["email"];
137 $convidar= $_POST["convidar"];
138 ?>
139
```

```
243
244 <?php
245 if (isset ($_POST["convidar"]) && $_POST["convidar"] != ""){
246
247 if (($nome != "") && ($email != "")){
248
249 $mensagem=" Acabaste de convidar o(a) <strong> " . $nome . " </strong>com o email <strong> " . $email . " </strong>para visitar o
250 Explaint.";
251 echo "<p>". $mensagem. "</p>" ;
252 }
253 };
254 ?>
```

- Página de Perfil (Query)

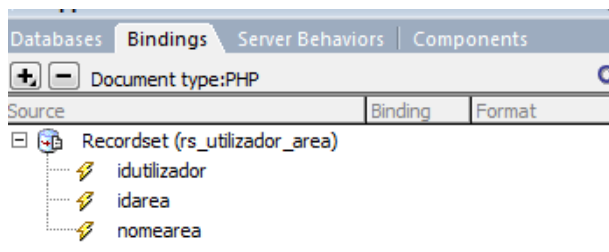
Para completar a informação relativa ao utilizador foi necessário recorrer a uma *query* do *FlySpeed* para passar não só os dados da tabela do utilizador, mas também os valores das tabelas **distrito** e **tipo de utilizadores**.

```
mysql_select_db($database_projecto_mydb, $projecto_mydb);
$query_rs_dados = sprintf("SELECT
    bd_projecto.utilizador.*,
    bd_projecto.distrito.descricao,
    bd_projecto.tipo_utilizador.tipo
FROM
    bd_projecto.utilizador INNER JOIN
    bd_projecto.distrito On bd_projecto.utilizador.distrito_id =
    bd_projecto.distrito.iddistrito INNER JOIN
    bd_projecto.tipo_utilizador On bd_projecto.utilizador.tipoutilizador_id =
    bd_projecto.tipo_utilizador.idtipo_utilizador WHERE email = %s", GetSQLValueString($colname_rs_dados, "text"));
```

Nesta mesma página há um *recordset* que é passado automaticamente devido à ligação ao **menu.php**.

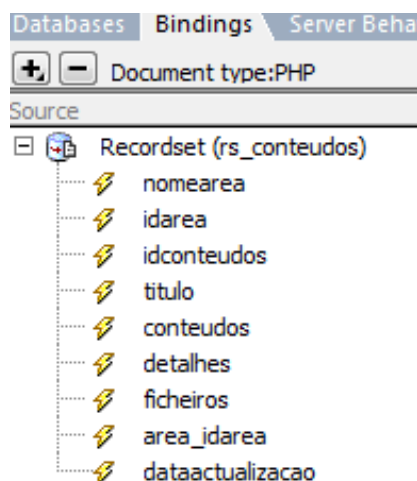
- Página de menu à direita

Criou-se um *recordset* com o objectivo de passar a informação relativa às áreas em que o utilizador entra na sessão que se inscreveu.



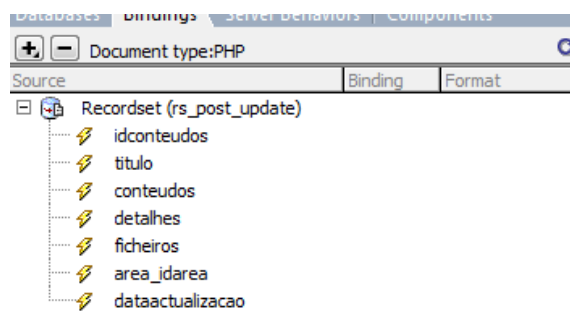
- Página das áreas

Pretendia-se que na página de cada área fosse implementado um mecanismo que mostrasse os *posts* publicados sobre informações importantes e acessíveis aos utilizadores registados nessa mesma área. Cada *post* apresenta apenas uma parte resumida da informação que poderá ser vista na sua totalidade através da ligação aos detalhes do *post*. De modo a receber os conteúdos provenientes da base de dados foi necessário criar um *recordset* para imprimir os campos de cada *post*. Foi ainda acrescentado um *repeat region* para que todas as publicações pudessem ser impressas em cada página.



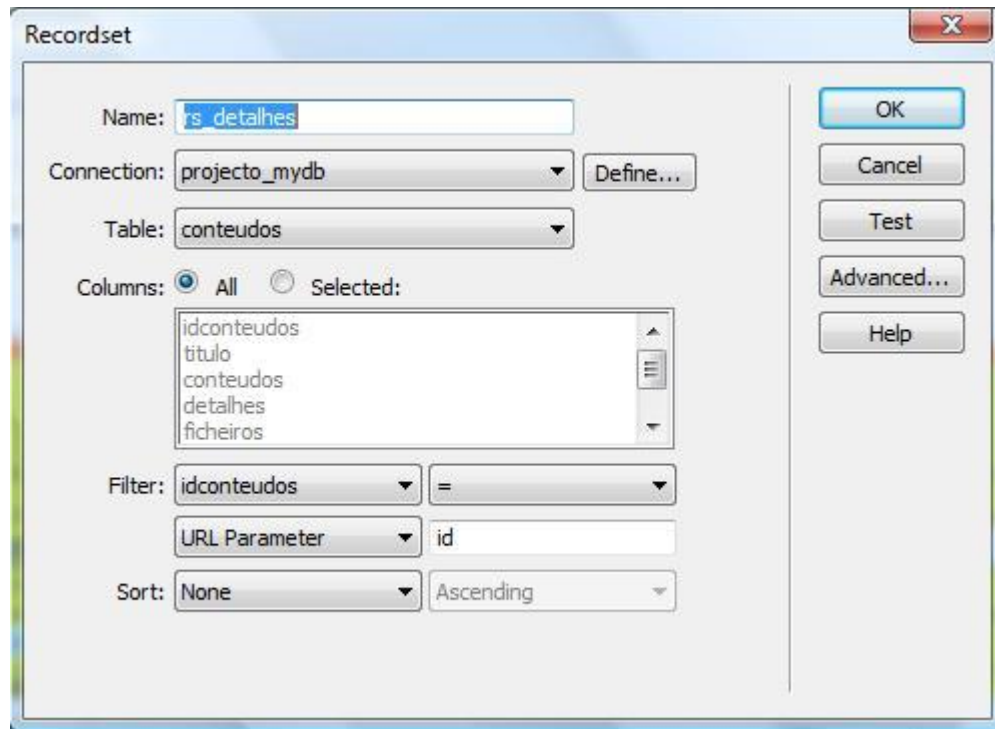
- Página de Editar um Post

Para editar o conteúdo dos *posts*, recorreu-se a um *Record update form wizard* e, tal como na página de detalhes, foi necessário criar um *recordset* com os dados do *post* a actualizar.



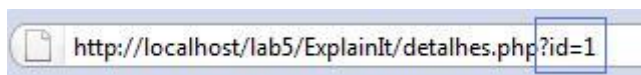
- Página das áreas e dos detalhes

Pretendia-se ter disponível uma página relativa à informação mais detalhada de cada publicação. Para tal, na página **detalhes.php** criou-se um *recordset* com todos os dados da tabela **conteúdos** tendo sido aplicado um filtro ao **idconteudos** indicando que o valor do parâmetro **id** passado por URL se refere à notícia pretendida.



Voltando à página da área, foi colocado um link em cada *post* que permite aceder à página **detalhes.php**, tendo sido dada uma instrução para passar o parâmetro **id** do post.

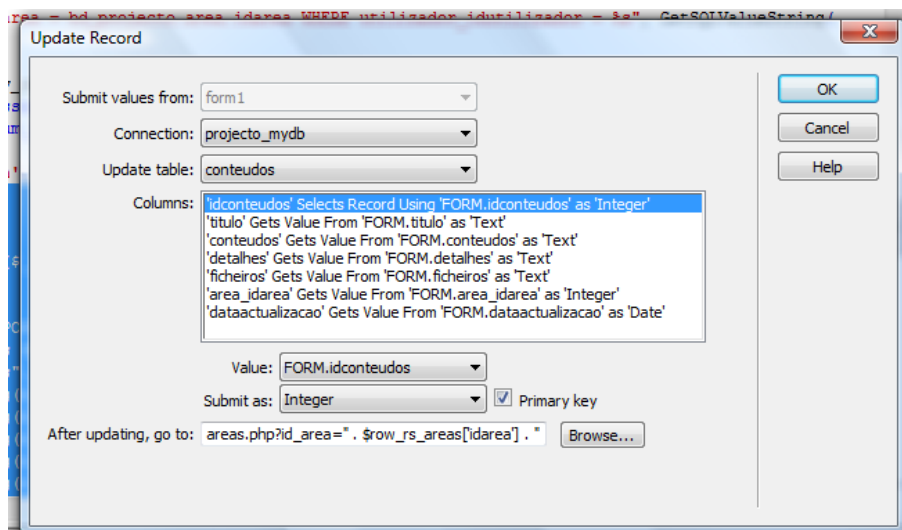
```
<a href="detalhes.php?id=<?php echo $row_rs_conteudos['idconteudos']; ?>">Ver Detalhes</a> &nbsp;
<a href="editar_post.php?id=<?php echo $row_rs_conteudos['idconteudos']; ?>">Editar Post</a>&nbsp;
<a href="apagar_post.php?id=<?php echo $row_rs_conteudos['idconteudos']; ?>">Apagar Post</a></div></th>
```



Passa o post com o id=1 da base de dados

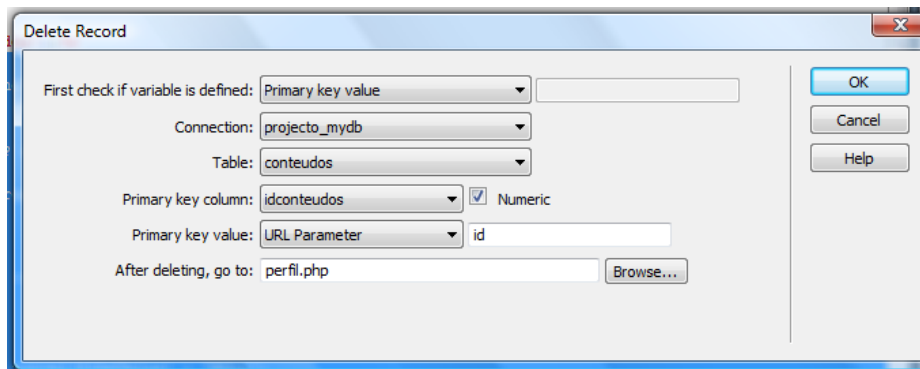
- Página de Editar post

Depois de criado o *recorset*, definiu-se o parâmetro **id** para passar apenas a informação do *post* pretendido.



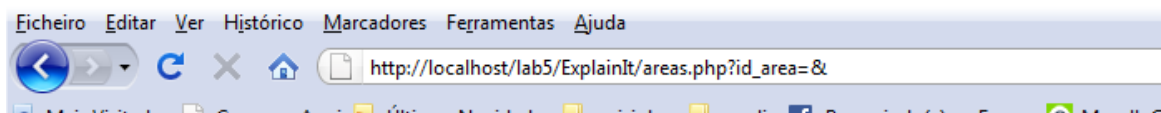
- Página de Apagar o post

Para apagar uma publicação recorreu-se à funcionalidade *Delete Record*, tendo-se filtrado o **idconteudos** e definido o valor a ser passado por parâmetro. É ainda importante referir que na ligação a esta página também foi indicado o valor **id** a ser passado no URL para que a conexão fosse executada com sucesso.

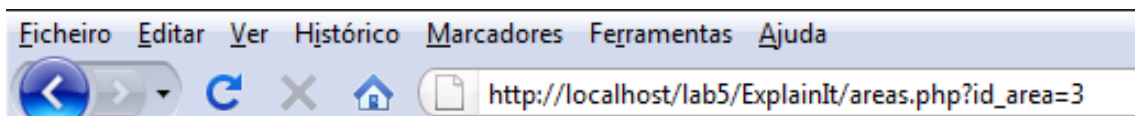


- Página Adicionar

Nesta página foram adoptados procedimentos semelhantes aos anteriores. No entanto surgiram problemas nomeadamente no momento de voltar à página da área onde se estava a adicionar a informação.



Quando se clicava para inserir um novo *post* verificava-se que este era inserido na área pretendida. O resultado que se pretendia era o que apresenta na imagem seguinte.



Depois de algumas tentativas para passar o parâmetro do **idarea** optou-se por, ao submeter, direccionar para a página de perfil.

- Página de Adicionar uma Área

Surgiram alguns os problemas que surgiram relativamente à passagem de parâmetros. Para inserir uma área nas preferências do utilizador começou-se por criar

um *recordset* na página **nova_area.php**, sendo este um INNER JOIN das tabelas **área** e **área_has_utilizador**. Nesse seguimento filtrou-se o campo **utilizador_idutilizador** passando a variável de sessão **MM_diUtilizador**. Depois foi feito um *Insert Record* para inserir dados na tabela **área_has_utilizador**.

3.2.4. Outros Contéúdos

- Data Actualização

Para a data de actualização gerou-se o código apresentado na imagem seguinte:

```
<?php $data= date("d-m-Y H:i:s"); echo $data;?>
```

- Perfil.php (convidar amigo)

Como já foi dito anteriormente, para convidar o amigo criou-se um formulário com campo nome e *email*. De modo a garantir que os utilizadores coloquem correctamente a informação dos respectivos campos definiu-se a validação do formulário. Para garantir que o convite foi enviado e que os campos contêm dados e que foi preenchido criou-se o *isset*.

```
236
237 <form action="<?php echo $_SERVER["PHP_SELF"]; ?>" method="post" name="formulario_amigo" id="amigo" onsubmit=
    "MM_validateForm('nome','R','email','R','RisEmail');return document.MM_returnValue" >
238 <p>Nome <input name="nome" type="text" id="nome" value=""/></p><br/>
239 <p>Email <input name="email" type="text" id="email" value=""/></p><br/>
240 <p><input name="convidar" type="submit" value="Convidar Amigo" /></p>
241 </form>
242
243 <?php
244 if (isset ($_POST["convidar"]) && $_POST["convidar"] != ""){
245
246     if (($nome != "") && ($email != "")){
247
248         $mensagem=" Acabaste de convidar o(a) <strong> " . $nome . " </strong>com o email <strong> " . $email . " </strong>para visitar o
        Explaint.";
249
250         echo "<p>". $mensagem. "</p>" ;
251     }
252 };
253 ?>
254
```

- Calendário

Os calendários dos explicadores foram construídos através de arrays. Para isso criou-se duas páginas, uma onde se inseriu os arrays e as suas posições e outra à qual se construiu uma tabela onde se desenvolveu dois ciclos for (para os dias e para as horas) e um foreach para percorrer todas as posições do array. Em cada iteração do ciclo será devolvido o valor armazenado numa dada posição do array. Para fazer a ligação entre as páginas fez-se um *require_once*.

```

23 <?php require_once("array_horario_linguas.php"); ?>
24 <div id="conteudos_hor" align="right" >
25 <?php
26 echo("<table class=\"horario\">");
27
28     echo("<tr>");
29     echo("<td class=\"amarelo\">Horas</td>");
30
31     for($hora=8; $hora<=23; $hora++){
32         $agora=$hora . "h00";
33         $proxima=$hora+1 . "h00";
34         echo("<td class=\"laranja\">" . $agora . "</td>");
35     }
36     echo("</tr>");
37
38     for($dias=0; $dias<=6; $dias++){
39
40         echo("<tr>");
41         echo("<td class=\"laranja\">".$array_horario[$dias][0]."</td>");
42
43         foreach($array_horario[$dias][1] as $professores){
44             echo("<td class=\"amarelo\">" . $professores . "</td>");
45         }
46
47         echo("</tr>");
48     }
49
50     echo("</table>");
51 }
52
53 </div>
54 ?>
55 </div>
56

```

Restrições

O administrador poderá aceder a todas as disciplinas, inserir conteúdos, ver os detalhes dos *posts*, fazer actualizações assim como apagar conteúdos impróprios. Os explicadores do ExplainIt poderão aceder ao chat, adicionar *posts* e editar conteúdos. Já os alunos, desde que inscritos nas áreas podem aceder aos posts, aos seus detalhes, ao chat assim como adicionar uma nova informação.

- Chat

O chat foi feito recorrendo a código retirado do site tutsplus. Começou-se por criar um chat básico recorrendo ao código disponibilizado no tutorial. Esta tarefa não foi tão linear como parece, uma vez que o código fornecido no tutorial continha muitos erros, e não funcionava convenientemente.

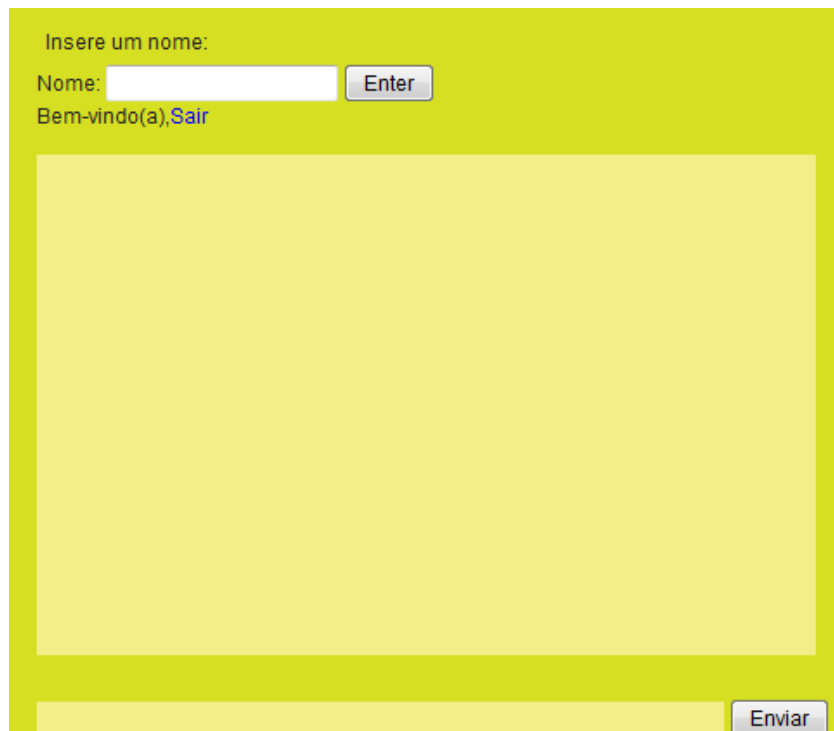
Please enter your name to continue:

Name:

Welcome, **user** [Exit Chat](#)

(12:32 PM) **Test**: asdf
User Test has left the chat session.
(12:47 PM) **user**: Hello
(12:47 PM) **user**: I'm a user.

Seguidamente estilizou-se para que se criasse coerência visual e retirou-se o formulário para inserção do nome de utilizador, substituindo esta funcionalidade pela variável de sessão: `$_SESSION['MM_Username']`. Desta forma, deixa de ser necessário que o utilizador insira um *username* sendo antes adoptado o e-mail utilizado anteriormente para o início de sessão.



Inicialmente, quando um utilizador saía do *chat*, toda a conversação feita anteriormente não era eliminada, fazendo com que ao regressar ao *chat*, essa conversação não fosse limpa. Para solucionar essa situação foi tido em conta o seguinte código: ***\$handle = fopen("log.html", "a");***. Após uma pesquisa, descobriu-se que se substituísse-mos o "a" por um "w", que faz com que o texto seja limpo após o regresso de um utilizador.

- Redimensionar Imagem

Através de uma livreria GD, descobriu-se o código para redimensionar a imagem. O código foi colocado na página "registo.php" na condição que faz o *upload* da imagem para a base de dados, por parecer aí o local mais adequado. No entanto o código acabou por não funcionar e, dada a escassez de tempo, acabou por se abandonar o código para se poder dar atenção a outros pormenores.

```

if((isset($_FILES['imagem']['name'])) and $_FILES['imagem']['name']!=""){
$uploadaddir = "images/";
$uploadfile = $uploadaddir . basename($_FILES['imagem']['name']);
if (move_uploaded_file($_FILES['imagem']['tmp_name'], $uploadfile)) {
    // Set the path to the image to resize
    $input_image = "images/" . basename($_FILES['imagem']['name']);
    // Get the size of the original image into an array
    $size = getimagesize( $input_image );
    // Set the new width of the image
    $thumb_width = "5";
    // Calculate the height of the new image to keep the aspect ratio
    $thumb_height = ( int )(( $thumb_width/$size[0] )*$size[1] );
    // Create a new true color image in the memory
    $thumbnail = ImageCreateTrueColor( $thumb_width, $thumb_height );
    // Create a new image from file
    $src_img = ImageCreateFromJPEG( $input_image );
    // Create the resized image
    ImageCopyResampled( $thumbnail, $src_img, 0, 0, 0, 0, $thumb_width, $thumb_height, $size[0], $size[1] );
    // Save the image as resized.jpg
    ImageJPEG( $thumbnail, basename($_FILES['imagem']['name']) );
    // Clear the memory of the temporary image
    ImageDestroy( $thumbnail );
}
}

```

4. Desenvolvimentos Futuros

Para desenvolvimentos futuros, o grupo esperaria melhorar a plataforma desenvolvida, nomeadamente no que respeita à definição da base de dados que desta feita gerou inúmeras controvérsias. Entre outras coisas, poderia ser implementado correctamente um sistema de classificação dos professores, o acesso aos horários de cada disciplina dinamicamente, bem como a possibilidade do utilizador escolher uma área de preferência e só mais tarde adicionar áreas às suas escolhas.

Relativamente a outras situações, futuramente poderia ser implementado um mecanismo de newsletter para que todos os utilizadores pudessem receber informações quinzenais (não justifica ser semanal, uma vez que neste tipo de plataforma não existe sempre informação a ser divulgada) como por exemplo, a entrada de novos formadores, novas áreas, notícias relativas ao *ExplainIt*, etc.

Outro recurso a implementar seria a encomenda de manuais escolares que seriam posteriormente enviados para a morada do utilizador. O *site* disponibilizaria uma “biblioteca de livros”, onde seria possível fazerem a escolha e efectuarem o respectivo pagamento.

Por último, a aplicação poderia ainda ter disponível uma área FAQ’S, para que os utilizadores pudessem tirar algumas dúvidas e ainda um mapa de *site*, para as pessoas que tem mais dificuldades na navegação do mesmo. Este último ponto acaba por ser uma mais-valia para este tipo de plataforma, uma vez que é um *site* frequentado por um público-alvo bastante abrangente.

5. Conclusões

Com a realização deste projecto o grupo teve a possibilidade de aplicar os conhecimentos adquiridos na unidade curricular de Laboratório Multimédia 5.

O seu desenvolvimento foi importante uma vez que permitiu ter uma noção da complexidade de uma plataforma deste tipo. Apesar de terem surgido algumas dificuldades, pode dizer-se que foi um projecto bastante motivador dado que foram acrescentadas novas aprendizagens para além dos conteúdos leccionados durante o semestre. No entanto, a equipa de trabalho tem consciência de que a plataforma poderia ser mais pormenorizada, porém a prioridade consistiu em criar uma aplicação Web 2.0, bem como o correcto funcionamento das páginas.

Após a execução do projecto “ExplainIt”, o grupo considera que há muitos erros que poderiam ser solucionados, sendo que a aplicação final não espelha na íntegra o que o grupo objectivou inicialmente. Ocorreram dificuldades nomeadamente na passagem de parâmetros entre páginas, na inscrição em áreas de preferência, na implementação do chat, entre outras situações. Importa realçar que à medida que o projecto foi avançando o grupo abandonou os passos lógicos que se esperavam de uma plataforma deste género, para ter implementados os conteúdos que foram leccionados. Apesar de todas as dificuldades, a equipa de trabalho realça que aprendeu imenso com o projecto, tendo mesmo implementado um chat, algo não aprendido durante as aulas.

Apesar de o grupo ter noção de que os prazos estabelecidos são para ser seguidos, a verdade é que o factor tempo foi algo que não permitiu ter uma aplicação a funcionar correctamente.

Para terminar, e apesar de todas as contrariedades, considera-se que se cumpriram os objectivos e requisitos pretendidos.

6. Referências Bibliográficas

- <http://www.escolavirtual.pt/> - Dezembro de 2010 e Janeiro de 2011
- <http://www.notaexito.com/> - Janeiro de 2011
- <http://www.w3schools.com/php/> - Janeiro de 2011
- <http://pt.fotolia.com/cat1/99001000> - Janeiro de 2011
- <http://asdesaber.pt/index.php> - Janeiro de 2011
- <http://www.rubblewebs.co.uk/imagemagick/GDexamples.php> - Janeiro 2011
- <http://net.tutsplus.com/tutorials/javascript-ajax/how-to-create-a-simple-web-based-chat-application/> - Janeiro 2011
- <http://php.net/manual/en/function.fopen.php> - Janeiro 2011