

## Ficha 9 – Revisões de desenvolvimento de aplicações *web* em PHP

### Objetivos

O objetivo desta ficha é aplicar os conhecimentos de HTML e PHP, adquiridos em DEAPC, no desenvolvimento de uma aplicação baseada na *web*. Para tal, será desenvolvido um *website* que, através de *scripts* PHP, permite manipular os dados armazenados no servidor.

O *website* deverá oferecer as seguintes funcionalidades:

- Possibilidade de inserção, numa base de dados, de registos constituídos por 3 campos relativos à caracterização de viaturas: matrícula (6 carateres), proprietário (máximo de 80 carateres) e valor comercial (valor numérico com um máximo de duas casas decimais).
- Listagem de todos os registos armazenados na base de dados.

Do lado do cliente, pretende-se que existam 4 vistas possíveis:

- Página inicial
- Página de listagem
- Página de inserção
- Página de confirmação de inserção

Adicionalmente, pretende-se que todas as páginas apresentem um menu com as seguintes opções de navegação: Página inicial, Listar registos, Inserir registo.

Do lado do servidor, os dados serão manipulados através de um conjunto de funções PHP definidas especificamente para esta aplicação e descritas no Apêndice I. Estas funções são fornecidas no ficheiro `db_functions.php`.

Uma análise do código do ficheiro irá revelar que a base de dados é implementada recorrendo a um simples ficheiro de texto. Isto é feito por motivos de simplicidade, nomeadamente com a finalidade de evitar a necessidade de instalação de um sistema gestor de base de dados ou bibliotecas adicionais. Adicionalmente, este ficheiro de texto é criado num diretório temporário especial que é eliminado cada vez que o sistema é desligado. Por este motivo, cada vez que reiniciar o sistema, a base de dados estará vazia.

Leia atentamente e siga os passos descritos no guião. O único exercício de programação que deverá resolver encontra-se no ponto 7 (além de uma pequena edição de código no ponto 4). Todos os ficheiros apresentados nas figuras são fornecidos no ficheiro zip disponível no Moodle.

### Estrutura do *website*

Esta secção destina-se a descrever a estrutura geral do *website* e algum do código PHP utilizado no mesmo.

De forma a uniformizar o aspeto da aplicação, cada página terá o mesmo cabeçalho e rodapé. O código HTML correspondente é fornecido nos ficheiros `header.html` (Figura 1) e `footer.html` (Figura 2). Estes ficheiros são usados para gerar cada uma das páginas, através de PHP. O processo é ilustrado para a página inicial, `index.php`, na Figura 3, sendo o resultado final apresentado na Figura 4.

```

<!doctype html>
<html>
<head>
<title>SISTC 2016/17 - Lab3</title>
<meta charset="UTF-8">
</head>
<body>

<a href='.'>Home</a>
<a href='vehicles.php?o=html_table'>List all</a>
<a href='insert_form.php'>Insert new record</a>
<br>
<br>

```

Figura 1 – Ficheiro header.html

```

</body>
</html>

```

Figura 2 – Ficheiro footer.html

```

<?php include("header.html");?>

<h1> Homepage </h1>
<!-- Editar a gosto. -->

<?php include("footer.html");?>

```

Figura 3 – Ficheiro index.php



Figura 4 – Aspeto da página inicial

O código PHP é adicionado entre as anotações “<?php” e “?>”. Este código é sempre interpretado do lado do servidor. Um ficheiro PHP pode ter várias secções de PHP intercaladas com HTML estático. No caso da página index.php, o PHP é usado apenas para incluir o código HTML dos ficheiros header.html e footer.html, sendo o resultado enviado para o cliente (o *web browser*, neste caso).

A análise do ficheiro header.html revela ligações para dois recursos adicionais: vehicles.php<sup>1</sup> e insert\_form.php. O *uniform resource locator* (URL) “vehicles.php” irá corresponder a um ficheiro com esse nome que deverá implementar

<sup>1</sup> O significado e finalidade do texto “?o=html\_table” são explicados mais abaixo.

o serviço de listagem. O URL “insert\_form.php” irá corresponder a um ficheiro com esse nome que faz parte da componente de apresentação, sendo responsável pela apresentação do formulário de introdução de novos registos (Figura 5).

```
<?php include("header.html");?>

<form action='vehicle.php' method=POST>
Plate: <input name='plate' pattern=".{6,6}" size="6" required> <br>
Owner: <input name='owner' pattern=".{3,80}" size="80" required> <br>
Value: <input name='value' type="number" step="0.01" value="0" required><br>
<br>
<input type=submit value='Insert'>
</form>
<?php include("footer.html");?>
```

Figura 5 – Ficheiro insert\_form.php

O ficheiro insert\_form.php usa o método HTTP POST para fazer o pedido de inserção de um novo veículo. O URL utilizado é “vehicle.php”, que neste caso irá corresponder a um ficheiro PHP com esse nome.

### Configuração do servidor *web*

No caso de usar o seu computador pessoal, certifique-se que tem o servidor *web* instalado e funcional. O servidor e o sistema PHP podem ser instalados e iniciados através da seguinte sequência de comandos:

```
su
dnf install httpd php
systemctl enable httpd.service
systemctl start httpd.service
```

O comando `systemctl enable` configura o serviço para ser iniciado automaticamente em cada arranque do sistema, evitando a necessidade do comando `systemctl start` cada vez que o sistema é reiniciado.

Verifique se o servidor *web* está ativo acedendo ao endereço <http://localhost> num *web browser*. Deverá ser-lhe apresentada uma página *web*. Em caso de erro, verifique se os comandos anteriores foram executados sem a apresentação de mensagens de erro.

De forma a ser possível alojar as páginas *web* na própria área do utilizador, é necessário editar o ficheiro `/etc/httpd/conf.d/userdir.conf` de forma a que este fique com um aspeto semelhante ao apresentado abaixo (o símbolo `#` é usado para comentar linhas):

```
<IfModule mod_userdir.c>

#COMENTAR
#UserDir disabled

#DESCOMENTAR
UserDir public_html
</IfModule>
```

Caso altere o ficheiro, deverá reiniciar o serviço:

```
systemctl restart httpd.service
```

Crie um diretório com o nome `public_html` no diretório *default* da sua conta de utilizador normal (`sistc`, nos computadores do laboratório) e conceda-lhe permissões de leitura e execução para todos os utilizadores do sistema, tal como se explica de seguida. Na linha de comando, pode mudar para o diretório *default* de um utilizador `user` usando o comando `cd ~user`. O diretório `public_html` é criado com o seguinte comando :

```
mkdir public_html
```

As permissões são alteradas com o comando `chmod`:

```
chmod a+rx public_html
```

Altere também as permissões do diretório base da sua conta:

```
chmod a+rx ~
```

No caso dos computadores do laboratório, uma vez que a mesma conta poderá ser usada por vários alunos, deverá ainda criar um subdiretório com nome à sua escolha (e.g., o seu número de aluno), onde irá alojar os ficheiros do *website*.

Verifique se consegue aceder ao endereço <http://localhost/~sistc> (no caso de um computador pessoal, substitua `sistc` pelo nome do utilizador que usa para fazer *login* no sistema).

Após a configuração do servidor web, deverá terminar a sessão do utilizador `root` (comando `exit`).

## Publicação das páginas e teste do servidor

Copie os ficheiros `index.php`, `header.php`, `footer.php` e `insert_form.php` para o diretório acabado de criar. De seguida, num *web browser*, aceda ao seguinte endereço:

[http://localhost/~sistc/subdiretorio\\_do\\_aluno](http://localhost/~sistc/subdiretorio_do_aluno)

O resultado deverá ser a apresentação de uma página semelhante à da Figura 4.

Adicionalmente, aceda ao seguinte endereço:

[http://localhost/~sistc/subdiretorio\\_do\\_aluno/populate.php](http://localhost/~sistc/subdiretorio_do_aluno/populate.php)

O *script* `populate.php` irá preencher a base de dados com alguns registos, de forma a facilitar o teste do *website*.

## Criação dos serviços web

1. Com um editor de texto (e.g., `geany`), crie um ficheiro chamado `vehicles.php` no diretório do *website*. Tal como descrito nos pontos seguintes, o código deste ficheiro será responsável por:
  - I. Aceder à base de dados e obter o vetor com todos os registos. Para tal deverá fazer o *include* do ficheiro `db_functions.php` e usar as funções `db_connect` e `db_read`, apresentadas no início do guião.
  - II. Gerar a lista de registos, em formato HTML e amigável para leitura.Eventuais situações de erro deverão ser reportadas ao utilizador.
2. Comece por introduzir o código da Figura 6 (disponibilizado no ficheiro `vehicles_v1.php`) no ficheiro `vehicles.php`. Este código faz a leitura dos dados a partir da BD, mas não os apresenta em formato amigável. Esta primeira

iteração é apresentada desta forma apenas para ilustrar o uso da função de depuração (*debug*) `var_dump`. Através desta função, será possível comprovar o tipo de dados retornado pela função `db_read`.

3. Aceda à opção de Listagem do *website* e observe os resultados. Note que o formato HTML não traduz o `'\n'` (carácter *newline*) para um fim de linha. Para se obter um fim de linha deverão ser usadas as anotações `<br>` (quebra de linha) ou `<p></p>` (parágrafo). Por esse motivo, o *output* da função `var_dump` poderá ser mais facilmente analisando recorrendo à opção “view page source” do *browser*.

Adicionalmente, é implementada a notificação de erros através da função `report_error`. Observe que esta função usa a instrução `echo` para fazer o envio do código HTML para o cliente. Dentro das secções PHP, a resposta ao cliente deverá ser gerada com recurso às instruções `echo` ou `print`.

```
<?php
include "db_functions.php";

if(($db = db_connect()) == FALSE) {
    db_report_error("DB access");
    exit(1);
}

if( ($res = db_read($db)) === NULL) {
    db_report_error("DB access");
    exit(2);
}

db_close($db);

// Output section
include("header.html");

var_dump($res);

include("footer.html");

?>
```

Figura 6 – Primeira iteração para o ficheiro `vehicles.php` (fornecido em `vehicles_v1.php`)

4. De forma a organizar a impressão dos dados, é conveniente aceder a cada um dos registos individualmente. A instrução `foreach` permite aceder sucessivamente a cada um dos elementos de um vetor<sup>2</sup>. Neste caso, uma possível abordagem seria:

```
foreach($res as $line)
    echo "$line[0], $line[1], $line[2] <br>\n";
```

Na instrução acima, cada elemento é sucessivamente acedido através da variável `$line`. Note que, tal como definido anteriormente, cada elemento do vetor `$res` é, por sua vez, também um vetor.

<sup>2</sup> Pode encontrar em <http://php.net/manual/en/language.control-structures.php> uma lista de todas as instruções de controlo de PHP juntamente com a respetiva sua descrição.

Ao contrário da linguagem C, o PHP permite construir *strings* a partir de diferentes variáveis sem ter que recorrer a funções do tipo `fprintf` ou `sprintf`. Para se poder utilizar as variáveis na construção da *string*, esta deve ser definida usando as aspas como delimitadores.

5. Substitua a chamada à instrução `var_dump` pelo código apresentado no ponto anterior e teste a página (opção de Listagem) novamente no *browser*.
6. O protocolo HTTP permite passar parâmetros através do URL. Essa parte do URL é denominada *query string* e é indicada através do carácter `'?'`. O tamanho máximo da *query string* é limitado pelos servidores *web*, pelo que este mecanismo não deve ser utilizado para enviar grandes quantidades de dados. Além disso, esses dados podem ser capturados através de analisadores de tráfego, mesmo quando é usado o protocolo HTTPS. No ficheiro `header.html`, pode ser encontrado o URL relativo `"vehicles.php?o=html_table"`. Neste caso, a *query string* contém apenas um parâmetro, `"o"`, cujo valor é `"html_table"`.

Os parâmetros da *query string* podem ser consultados no código PHP através da variável "superglobal" `$_GET`. Esta variável é um vetor associativo, em que cada elemento é o valor de um dos parâmetros da *query string* e a sua chave é o nome do respetivo parâmetro. Neste caso, o valor do parâmetro `"o"` é acedido através de `$_GET['o']`.

A Figura 7 apresenta a versão final do ficheiro `vehicles.php`. Observe que, através da utilização da variável `$_GET` e da instrução `switch`, o *script* oferece duas alternativas de apresentação dos dados.

7. Atualize o código do ficheiro `vehicles.php` de acordo com a listagem da Figura 7 (código disponibilizado no ficheiro `vehicles_v2.php`) e teste novamente o *site*. Noutro separador do *browser*, acesse diretamente ao URL

[http://localhost/~sistc/subdiretorio\\_do\\_aluno/vehicles.php](http://localhost/~sistc/subdiretorio_do_aluno/vehicles.php)

e comprove que ambas as possibilidades de apresentação estão disponíveis.

8. Complete o *website* através da implementação do ficheiro `vehicle.php` (o ficheiro `vehicles.php` deverá permanecer inalterado). Este ficheiro será chamado em resposta ao pedido POST da página `insert_form.php` (ver Figura 5). Os parâmetros passados através de pedidos POST podem ser consultados no código PHP através da variável "superglobal" `$_POST`. A utilização desta variável é análoga a utilização da variável `$_GET`.

No caso de sucesso, deverá ser apresentada uma página com o texto `"Record inserted"`. Os casos de erro devem ser tratados de forma análoga ao que foi feito para o *script* `vehicles.php`.

9. Teste a versão final do *website*, verificando se os registos introduzidos são apresentados na página de listagem.
10. Guarde uma cópia de segurança do seu trabalho, pois estes ficheiros serão necessários

na próxima ficha.

```
<?php
include("db_functions.php");

if(($db = db_connect()) == FALSE) {
    db_report_error("DB access");
    exit(1);
}

if( ($res = db_read($db)) === NULL) {
    db_report_error("DB access");
    exit(2);
}

db_close($db);

// Output section
switch($_GET['o']) {

//HTML format, table
case 'html_table':
    include("header.html");

    //CSS
    echo "<style type=\"text/css\">\n
table, th, td { border: 1px solid black;}\n
th, td { padding: 15px;}\n
</style>\n";

    echo "<table>\n";
    //table headers
    echo "<tr><th>Registration plate</th>
        <th>Owner</th><th>Comercial value</th></tr>\n";
    //table rows
    foreach($res as $line) {
        echo "<tr>";
        foreach($line as $col)
            echo "<td>".$col."</td>";
        echo "</tr>\n";
    }
    echo "</table>\n";
    echo "<p>".count($res)." records.</p>\n";

    include("footer.html");
    break;

//HTML format, multiple CSV lines
default:
    include("header.html");

    foreach($res as $line)
        echo "$line[0], $line[1], $line[2] <br>\n";

    include("footer.html");
}

?>
```

Figura 7 – Versão final do ficheiro vehicles.php (fornecido em vehicles\_v2.php)

## Apêndice I

O ficheiro `db_functions.php` contém a definição das seguintes funções

- `db_connect()` - cria uma ligação à base de dados e retorna identificador da ligação.
- `db_insert($db, $rec)` - insere o registo `$rec` na base de dados. O registo `$rec` deverá ser um vetor associativo contendo as seguintes chaves: `'plate'` (para o número de matrícula), `'owner'` (nome do proprietário) e `'value'` (valor comercial). Retorna `FALSE` em caso de erro.
- `db_read($db)` - retorna um vetor multidimensional com todos os registos armazenados na base de dados. Cada elemento deste vetor é um vetor cujos elementos são os valores dos três campos do registo respetivo.
- `db_close($db)` - fecha a ligação à base de dados.