

LAB 7 – Cookies e Sessões em PHP

1. Assuma-se neste lab que concluiu com sucesso o LAB6. Copie todos os ficheiros da pasta “LAB6” para a pasta “LAB7”.

```
a12345@daw2:~$ cd public_html
```

```
a12345@daw2:~/public_html$ cp -R LAB6 LAB7
```

```
a12345@daw2:~/public_html$ chmod g+w LAB7/templates_c
```

2. Construa o template **SMARTY** (formulario) `blog_template.tpl`

© 2016 Desenvolvimento de Aplicações Web Designed by [Aluno](#)

- {MENU_1} é um link (“home”) para “index.php”
- {MENU_2} é um self-link (“blog”)
- {BLOG} é um placeholder para texto
- {ACTION} é um placeholder para a acção (não está visível no template)

3. Construa o controlador PHP `blog.php` que:

- se a variável `micropost_id` recebida pelo método GET **não estiver definida**, o placeholder {BLOG} está vazio, e o placeholder {ACTION} contém `action="newblog_action.php"`
- se a variável `micropost_id` recebida pelo método GET **estiver definida**, coloca no placeholder {BLOG} o texto do blog que se vai actualizar, e o placeholder {ACTION} contém `action="updateblog_action.php"`¹
- em ambos os casos o método deverá ser `method="post"`.

¹ Em alternativa pode usar a estrutura `{if},{else}` no template Smarty

4. Construa o controlador `newblog_action.php` que

- tem acesso ao cookie da sessão
- com o id do utilizador registado na sessão insere na tabela “microposts” na base de dados o novo blog

```
INSERT INTO microposts (user_id, ...) VALUES(...)
```
- envia a mensagem “SUCCESS: New post submitted” e re-direcciona para o portal de entrada no site `index.php`. Utilize o controlador `message.php` para enviar a mensagem.

- se o cookie da sessão não estiver definido, deve ser enviada uma mensagem de erro “ERROR: Login first” e feito o redireccionamento para a página de rosto do site. Utilize o controlador `message.php` para enviar a mensagem.

5. Construa o controlador `updateblog_action.php` que

- tem acesso ao cookie da sessão
- com o id do utilizador registado e o id do blog² actualiza a tabela “microposts” na base de dados com o novo texto do blog

```
"UPDATE microposts SET content= ... WHERE id="
```
- envia a mensagem “SUCCESS: Post updated” e re-direcciona para o portal de entrada no site `index.php`

- se o cookie da sessão não estiver definido (ou o utilizador tentar fazer o update de um post que não lhe pertence...) deve ser enviada uma mensagem de erro “ERROR: Not allowed” e feito o redireccionamento para a página de rosto do site. Utilize o controlador `message.php` para enviar a mensagem.

² O id do blog pode ser passado de várias formas à escolha

1. `<input type="hidden" name="micropost_id" value="82" >`
2. `<form action="updateblog_action.php?micropost_id=82" method="post">`
3. `$_SESSION['micropost_id']=82;`

FUNCIONALIDADE “REMEMBER ME”

6. Altere o template `login_template.tpl`:

{MENU_1} {MENU_2} {MENU_3}

{MESSAGE}

Login

Email

Password

Go Remember Me [Forgot password?](#)

© 2016 Desenvolvimento de Aplicações Web Designed by Aluno

- Adicione uma nova tag “input” `<input type="checkbox" name="autologin" value="1">`
- Adicione um hyperlink `Forgot password?`

7. Altere o controlador `login_action.php` que vai consultar a base de dados para validar o login do utilizador.

Em caso de sucesso e a checkbox “Remember Me” foi activada:

- cria um “cookie” com o nome “rememberMe”
- coloca no valor do cookie um token “md5” do tempo actual `substr(md5(time()), 0, 32)`
- define a duração do cookie um mês: `time() + (3600 * 24 * 30)`
- guarda o valor do cookie na base de dados (tabela users, coluna remember_digest)

8. Altere o controlador `index.php` para, se o servidor web receber um cookie com o token guardado na base de dados, validar imediatamente o utilizador a quem pertence o token.

Pode encontrar a funcionalidade que se pretende em

http://all.deei.fct.ualg.pt/~a999990/smarty_exame2/

Considere o lab concluído quando tiver reproduzido a funcionalidade no seu portal no servidor "daw2" (<http://daw.deei.fct.ualg.pt>).

9. Faça o upload dos ficheiros

- `blog.php`
- `newblog_action.php`
- `updateblog_action.php`
- `login_action.php`
- `model.php` (opcional)

para a pasta "LAB7" ("lab7" NÃO, "Lab7" NÃO)

- `blog_template.tpl`
- `login_template.tpl`

para a pasta "LAB7/templates"

REFERÊNCIAS:

- <http://intranet.deei.fct.ualg.pt/DAW/cookies/files.html>
- <http://intranet.deei.fct.ualg.pt/DAW/auth-db-sessions/files.html>

ANEXO 1: Acesso à base de dados MySQL

- O acesso à base de dados MySQL na rede UALG pode ser feita em linha de comando

```
a12345@daw2:~$mysql -u a12345 -p -h 10.10.23.184 db_a12345
```

(substitua “12345” pelo seu número de aluno)

ou ainda utilizando o software **phpMyAdmin** disponível no URL

<http://daw.deei.fct.ualg.pt/phpMyAdmin>

ANEXO 2: Estrutura da base de dados

```
CREATE TABLE `users` (  
  `id` int(11) NOT NULL auto_increment,  
  `name` varchar(255) default NULL,  
  `email` varchar(255) default NULL,  
  `created_at` datetime NOT NULL,  
  `updated_at` datetime NOT NULL,  
  `password_digest` varchar(255) default NULL,  
  `remember_digest` varchar(255) default NULL,  
  `admin` tinyint(1) default NULL,  
  `activation_digest` varchar(255) default NULL,  
  `activated` tinyint(1) default NULL,  
  `activated_at` datetime default NULL,  
  `reset_digest` varchar(255) default NULL,  
  `reset_sent_at` datetime default NULL,  
  PRIMARY KEY (`id`),  
  UNIQUE KEY `index_users_on_email` (`email`)  
) ENGINE=InnoDB DEFAULT CHARSET=utf8;  
  
CREATE TABLE `microposts` (  
  `id` int(11) NOT NULL auto_increment,  
  `content` text,  
  `user_id` int(11) default NULL,  
  `created_at` datetime NOT NULL,  
  `updated_at` datetime NOT NULL,  
  `likes` int(11) NOT NULL DEFAULT '0',  
  PRIMARY KEY (`id`),  
  KEY `fk_user_id` (`user_id`),  
  CONSTRAINT `fk_user_id` FOREIGN KEY (`user_id`)  
REFERENCES `users` (`id`)  
) ENGINE=InnoDB DEFAULT CHARSET=utf8;
```

ANEXO 3: OPCIONAL

Este guião realiza inteiramente o lab no servidor de produção, mas se desejar pode realizar o lab no seu PC/portátil. As instruções para a instalação de um ambiente de desenvolvimento "XAMPP" no seu PC/portátil encontram-se em

<http://intranet.deei.fct.uaig.pt/IPM/XAMPP.pdf>

ANEXO 4: OPCIONAL

FUNCIONALIDADE “PRIVATE MESSAGE”

Pretende-se implementar a funcionalidade “private message” que permite a utilizadores registados no site trocarem mensagens privadas entre si

0. Crie a seguinte tabela na base de dados

```
CREATE TABLE `messages` (  
  `id` int(11) NOT NULL AUTO_INCREMENT,  
  `content` text,  
  `from_user_id` int(11) NOT NULL,  
  `to_user_id` int(11) NOT NULL,  
  `created_at` datetime NOT NULL,  
  `status` int(1) DEFAULT NULL,  
  PRIMARY KEY (`id`),  
  KEY (`from_user_id`),  
  KEY (`to_user_id`),  
  CONSTRAINT FOREIGN KEY (`from_user_id`) REFERENCES `users` (`id`),  
  CONSTRAINT FOREIGN KEY (`to_user_id`) REFERENCES `users` (`id`)  
) ENGINE=InnoDB DEFAULT CHARSET=utf8
```

1. Actualize o template `index_template.html`

- se o utilizador fez *log in*, para ter um “placeholder” adicional no menu para o *hyperlink*

```
<a href="showprivatemessages.php">Show private messages</a>
```

- se o utilizador é anónimo, o “placeholder” está vazio



home [Show private messages](#) [logout](#) [new post](#) Welcome Jose Bastos

Jose Bastos posted: [update post](#)

mais um teste
12345

updated: 2017-12-11 19:05:10 created: 2017-11-11 22:08:01 0

Tester posted: [send private message](#)

Novo post e continua

updated: 2017-11-09 17:40:24 created: 2017-11-09 16:50:55 1

2. Altere o ficheiro `index.php` para:

- A. actualizar o “placeholder” do ponto 1
- B. actualizar o “placeholder” `{ $update_pm }`
 - se o utilizador que fez login é o autor do post, com o hyperlink

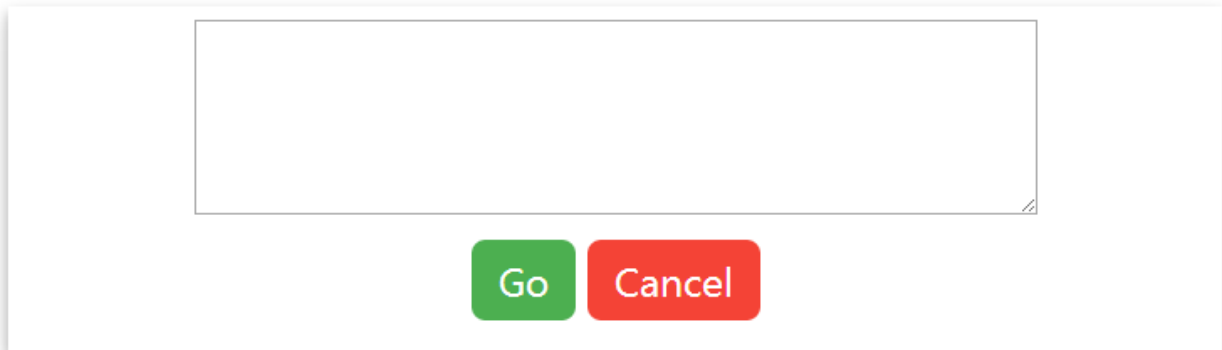
```
<a href="updateblog.php?micropost_id=' . $tuple['id'] . '">update  
post</a>
```

se o utilizador que fez login não é o autor do post, com o hyperlink

```
<a href="privatemessage.php?user_id=' . $tuple['user_id'] . '">send  
private message</a>
```

3. Construa o template “`privatemessage_template.html`”

SUGESTÃO: Copie e adapte o template “`blog_template.html`”



O formulário utiliza o método “post” e a acção é

```
method="post" action="privatemessage_action.php"
```

4. Construa o controlador do template “privatemessage.php”

SUGESTÃO: Copie e adapte o controlador “blog.php”

A passagem da variável “user_id” para o controlador da acção pode ser feito pelo método GET no URL da acção ou (recomendado) através de um input do tipo “hidden” no formulário

5. Construa o controlador `privatemessage_action.php`

que actualiza a tabela “messages” com a nova mensagem privada

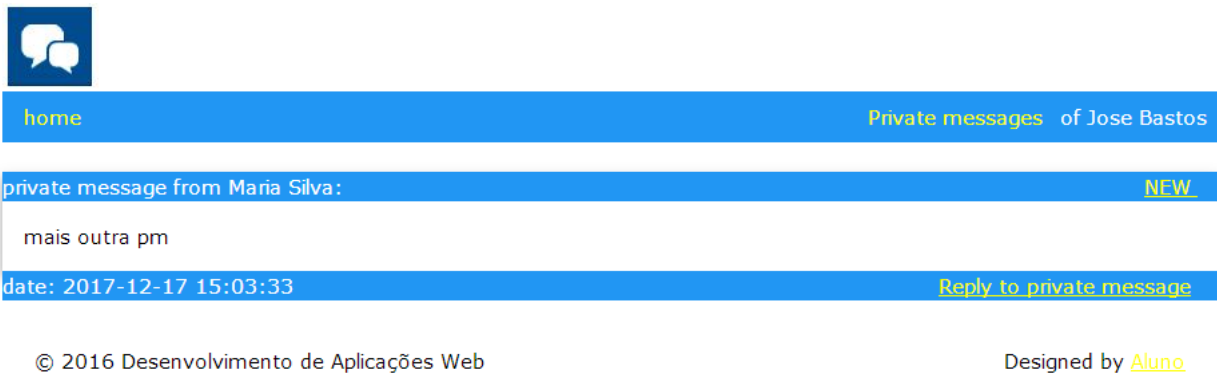
NOTA: o status da mensagem deve ser “0” (não lida)

```
$query = "INSERT INTO messages (from_user_id, to_user_id,
content,created_at, status) VALUES('" . $_SESSION['id'] . "','" .
$_POST['user_id'] . "','" . $_POST['pm'] . "','" . $present_date .
"', '0')"
```

SUGESTÃO: Copie e adapte o controlador “blog_action.php”

6. Construa o controlador `showprivatemessages.php`

Este controlador trabalha com o template `showprivatemessages_template.html` (o template é dado já completo)



que tem um hyperlink

```
<a href="privatemessage.php?user_id=' . $tuple['from_user_id'] .
'">Reply to private message</a>
```

que permite o utilizador responder às mensagens privadas

e outro hyperlink

```
<a href="markread_action.php?m_id=' . $tuple['m_id'] . '">NEW <span
class="w3-text">(<em>click to mark read</em></span></a>
```

que permite “marcar” uma mensagem como “lida”

7. Construa o controlador `markread_action.php` que actualiza o status da mensagem correspondente na tabela “messages” como “lida” (`status='1'`)