

# LAB 10 – Programação com o framework Symfony [parte 3]

---

O objectivo deste laboratório é implementar a funcionalidade “Recover Password” com o framework **Symfony** 4.4 e a *template engine* **Twig**.

Assume-se neste lab que concluiu com sucesso o LAB9

## 1 .. PRELIMINARES

**Altere o ficheiro “.env” para utilizar o servidor de email do DEEI**

```
a12345@daw2:~/public_html/LAB8_10$ nano .env
```

retire o “#” no principio da linha e substitua “localhost” por “10.10.23.49”

```
###> symfony/mailer ###  
MAILER_DSN=smtp://10.10.23.49  
###< symfony/mailer ###
```

## FUNCIONALIDADE “RECOVER PASSWORD”<sup>1</sup>

### 2. Desenhe o template “password\_reset\_template.html.twig”

{MENU\_1} {MENU\_2} {MENU\_3}

{MESSAGE}

Password reset

Email

Go

© 2016 Desenvolvimento de Aplicações Web Designed by Aluno

Este template utiliza o método “POST” para enviar o campo de Email ao servidor:

```
<form method="post" action="password_reset_action">
```

### 3. Construa as funções “password\_reset()” e “password\_reset\_action()” associadas a este template<sup>2</sup>

```
/**
 * @Route("/blog/password_reset", name="password_reset")
 */
public function password_reset()
{
}

/**
 * @Route("/blog/password_reset_action",
name="password_reset_action")
 */
public function password_reset_action()
{
}
```

<sup>1</sup> Se desejar pode usar o MakerBundle (php bin/console make:reset-password) para realizar toda a funcionalidade!

<sup>2</sup> Se desejar pode utilizar um só controlador password\_reset() para o formulário e para a acção

No caso do email enviado não existir na base de dados, o “placeholder” {MESSAGE} deve mostrar a mensagem “Error: email does not exist”

(Nota: a passagem de informação entre `password_reset_action()` e `password_reset()` é feita através de uma variável da sessão)

**4.** Construa a função `password_reset_action()` (e as funções que realizam as queries correspondente na base de dados) que:

- verifica se o email introduzido consta da base de dados
- em caso de insucesso faz redirect para `password_reset()`
- em caso de sucesso
  - Cria um token “md5” do tempo actual  
`$reset_digest = substr(md5(time()), 0, 32);`
  - guarda este valor na base de dados (tabela `users`, coluna `reset_digest`), bem como tempo actual (tabela `users`, coluna `reset_sent_at`)
  - envia ao utilizador registado um email personalizado com o assunto “Password reset” e com o texto

```
Olá Sr.(a) {{name}}  
Para obter uma nova password clique no link  
http://daw.deei.fct.ualg.pt/~a12345/LAB8_10/new_password/{{reset_digest}}  
Este link tem a validade de uma hora.  
Se NÃO pediu uma nova password IGNORE este email.  
  
Cumprimentos,  
webmaster!  
Página web:      http://all.deei.fct.ualg.pt/~a12345/LAB8/  
E-mail:          a12345@deei.fct.ualg.pt  
  
NOTA: Não responda a este email, não vai obter resposta!
```

Note que o texto do email contem um link para a página “new\_password” (substituir 12345 pelo seu numero de aluno) **com o valor do token embutido**

- faz redirect para o URL `message`

**5.** Adapte o template `message_template.tpl` e construa o template `message_template.html.twig`



{MESSAGE}

© 2016 Desenvolvimento de Aplicações Web

Designed by Aluno

## 6. Construa<sup>3</sup> a função `message()` associada a este template

Sugere-se o seguinte conteúdo para o placeholder {MESSAGE}

"Password reset activated! <br> Email sent to you :-)"

## 7. Desenhe o template "new\_password\_template.html.twig"

que utiliza o método "POST" para enviar a nova password ao servidor, bem como o token recebido (como um input do tipo "hidden")

```
<form method="post" action="new_password__action">
<input type="hidden" name="token" value="{{token}}">
```

## 8. Construa o código PHP `new_password()` associado a este template que actualiza o placeholder `{{token}}` (com o valor da variável "token" recebida embutida no link)

```
/**
 * @Route("/new_password/{token}", name="new_password")
 */
public function new_password(token)
{
}
```

<sup>3</sup> Os passos 5 e 6 apenas são necessários se não criou `message()` e `message_template.html.twig` no lab anterior

(Opcional: pode adicionar o placeholder {MESSAGE} para alertar se a confirmação da password é diferente da password introduzida)

**9.** Crie a função `new_password_action()` que vai actualizar a base de dados com a nova password do utilizador.

```
/**
 * @Route("/new_password_action name="new_password_action")
 */
public function new_password_action()
{
    }
}
```

- verifica se o token recebido existe na base de dados
- em caso de sucesso e se não passou mais de uma hora entre a hora actual e a hora de envio do email
  - encripta e actualiza o hash da password na base de dados
  - faz redirect para o URL message

(sugere-se que o placeholder {MESSAGE} tenha o seguinte texto a amarelo: Password reset successfully! )

- em caso de insucesso faz redirect para o URL message

(sugere-se que o placeholder {MESSAGE} tenha o seguinte texto a vermelho:

ERROR: WRONG TOKEN OR TOKEN EXPIRED, PASSWORD RESET FAILED!

Considere o lab concluído quando tiver reproduzido a funcionalidade do site exemplo

[http://daw.deei.fct.ualg.pt/~a9999993/LV\\_exam2/blog](http://daw.deei.fct.ualg.pt/~a9999993/LV_exam2/blog)

no seu portal.

**NOTA: Por razões de segurança preferencialmente envie emails para o dominio ualg.pt. Faça testes com um utilizador registado no portal com o seu email válido: a12345@ualg.pt.**

Caso tenha trabalhado no seu portátil/PC, faça o upload dos ficheiros para o servidor

para a pasta "src/Controller":

- BlogController.php
- Blog\_modelController.php

para a pasta "templates/blog":

- password\_reset\_template.html.twig
- new\_password\_template.html.twig
- message\_template.html.twig

## REFERÊNCIAS:

- [http://daw.deei.fct.ualg.pt/~a999993/LV\\_exame2/blog](http://daw.deei.fct.ualg.pt/~a999993/LV_exame2/blog)
- <https://symfony.com/doc/4.4/index.html>
- <https://symfonycasts.com/screencast/symfony4>
- [http://intranet.deei.fct.ualg.pt/DAW/slides/SF\\_overview.pdf](http://intranet.deei.fct.ualg.pt/DAW/slides/SF_overview.pdf)
- [http://intranet.deei.fct.ualg.pt/DAW/slides/SF\\_doctrine.pdf](http://intranet.deei.fct.ualg.pt/DAW/slides/SF_doctrine.pdf)
- <http://all.deei.fct.ualg.pt/symfony/>

## **ANEXO 1** Acesso à base de dados MySQL

- O acesso à base de dados MySQL pode ser feita utilizando um cliente gráfico à sua escolha (por exemplo <http://www.heidisql.com/>),

ou em linha de comando (apenas funciona na rede UALG)

```
a12345@daw2:~$mysql -u a12345 -p -h 10.10.23.183 db_a12345
```

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

<http://daw.deei.fct.ualg.pt/phpMyAdmin> (funciona na Internet e na rede UALG)

## 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) NOT 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 (`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,  
  PRIMARY KEY (`id`),  
  KEY (`user_id`),  
  CONSTRAINT FOREIGN KEY (`user_id`) REFERENCES `users`  
  (`id`)  
) ENGINE=InnoDB DEFAULT CHARSET=utf8;
```