Criando uma biblioteca para controle de acesso por usuário aplicada a classes e métodos com o CodeIgniter.

Nível: Intermediário

Autor: Ademir Cristiano Gabardo

As aplicações PHP estão se tornando a cada dia mais complexas, aplicações de uso comercial, acadêmico e das mais diversas finalidades fazem necessário um controle mais apurado de acesso.

No CodeIgniter obrigatóriamente estamos sempre acessando uma classe e um método de um controlador. Desta forma é possível mapear todas as possíveis ações e ou caminhos disponíveis em uma aplicação e adicionar o controle de acesso aos métodos que desejamos controlar.

Desenvolvi uma biblioteca (librarie) para controlar o acesso dos usuários, o funcionamento é o seguinte.

Em uma tabela no banco de dados (sys_metodos) ficam guardados os nomes de classe e métodos, mapeando todos os possíveis caminhos e em outra (sys_permissoes) ficam as ids dos usuários com os respectivos métodos a que eles tem acesso. Esta segunda tabela está relacionada a tabela de cadastro de usuários (tb_usuarios).

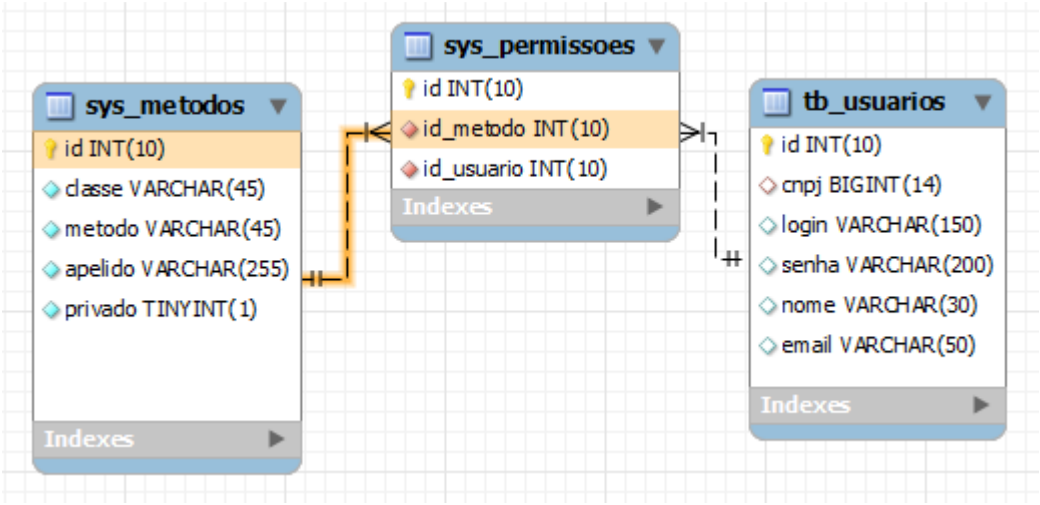

Figura 1 – Estrutura das tabelas necessárias ao sistema de autenticação

Seria bastante trabalhoso ter que popular a tabela de métodos manualmente, ou criar um Admin para ela, desta forma, foi incluída uma rotina na própria biblioteca para que quando um método invocado por ela não for encontrado na tabela sys_metodos ele seja automaticamente criado. Desta forma, na primeira vez que um método for acessado ninguém terá permissão de acesso, assim, será necessário parametrizar o sistema uma única vez no primeiro acesso. Depois poderá se construir uma área de administração para varrer a tabela de métodos e adicionar ou remover as permissões dos usuários.

O CodeIgniter provê uma maneira de checarmos “onde estamos”, ou seja, em que classe e em que método.

Para verificar o nome da classe utilize a seguinte linha de código.

```
$this->router->class
```

E para recuperar o nome do método utilize.

```
$this->router->method
```

Desta forma, podemos passar para a biblioteca o caminho completo de onde estamos da seguinte forma.

```
$this->auth->check_logged($this->router->class , $this->router->method);
```

Sendo auth o nome da biblioteca e check_logged o método que estamos acessando na biblioteca. Como esta biblioteca estará sendo acessada em muitas classes e métodos é preferível carregar no autoload para que ela esteja sempre disponível em qualquer parte da aplicação.

Para proteger um método, basta incluir a linha de código conforme mostrado na primeira linha do método da seguinte forma.

```
<?php
class AreaRestrita extends Controller {
		function __construct(){
			parent::Controller();
		}
		function index(){
			$this->auth->check_logged($this->router->class , $this->router->method);
		}
}
?>
```

Vejamos então como fica o código fonte da bilblioteca. Salve o código-fonte a seguir com o nome de arquivo auth.php na pasta system/application/libraries.

```
		<?php  if ( ! defined('BASEPATH')) exit('No direct script access allowed');
		class Auth
		{
			private $ci;
			public function __construct(){
			$this->ci = &get_instance();
		}

		function check_logged($classe,$metodo)
		{
			/*
			* Criando uma instância do CodeIgniter para poder acessar
			* banco de dados, sessionns, models, etc...
			*/
			$this->CI =& get_instance();

			/**
			* Buscando a classe e metodo da tabela sys_metodos
			*/
			$array = array('classe' => $classe, 'metodo' => $metodo);
			$this->CI->db->where($array);
			$query = $this->CI->db->get('sys_metodos');
			$result = $query->result();

			// Se este metodo ainda não existir na tabela sera cadastrado
			if(count($result)==0){
				$data = array(
					'classe' => $classe ,
					'metodo' => $metodo ,
					'apelido' => $classe .  '/' . $metodo,
					'privado' => 1
				);
				$this->CI->db->insert('sys_metodos', $data);
				redirect(base_url(). $classe . '/' . $metodo, 'refresh');
			}
			//Se ja existir tras as informacoes de publico ou privado
```

```
        else{
            if($result[0]->privado==0){
                // Escapa da validacao e mostra o metodo.
                return false;
            }
            else{
                // Se for privado, verifica o login
                $nome = $this->ci->session->userdata('nome');
                $logged_in = $this->ci->session->userdata('logged_in');
                $data = $this->ci->session->userdata('data');
                $email = $this->ci->session->userdata('email');
                $id_usuario =  $this->ci->session->userdata('id_usuario');

                $id_sys_metodos = $result[0]->id;

                // Se o usuario estiver logado vai verificar se tem permissao na tabela.
                if($nome && $logged_in && $id_usuario){

                    $array = array('id_metodo' => $id_sys_metodos, 'id_usuario' => $id_usuario);
                    $this->CI->db->where($array);
                    $query2 = $this->CI->db->get('sys_permissoes');
                    $result2 = $query2->result();

                    // Se não vier nenhum resultado da consulta, manda para página de
                    // usuario sem permissão.
                    if(count($result2)==0){
                        redirect(base_url().'home/sempermissao', 'refresh');
                    }
                    else{
                        return true;
                    }
                }
                // Se não estiver logado, sera redirecionado para o login.
                else{
                        redirect(base_url().'home/login', 'refresh');
                    }
                }
            }
        }

        /**
        * Método auxiliar para autenticar entradas em menu.
        * Não faz parte do plugin como um todo.
        */
        function check_menu($classe,$metodo){
            $this->CI =& get_instance();
            $sql = "SELECT SQL_CACHE
                count(sys_permissoes.id) as found
                FROM
                sys_permissoes
                INNER JOIN sys_metodos
                ON sys_metodos.id = sys_permissoes.id_metodo
                WHERE id_usuario = '" . $this->ci->session->userdata('id_usuario') . "'
                AND classe = '" . $classe . "'
                AND metodo = '" . $metodo . "'";
            $query = $this->CI->db->query($sql);
            $result = $query->result();
            return $result[0]->found;
        }
    }
```

Note que a biblioteca se encarrega de inserir os dados de métodos e classes que ainda não estejam cadastrados na tabela sys_metodos.

Quando um usuário tentar acessar uma área do website ou sistema que ele não tenha privilégios ele será redirecionado para uma página específica.

```
redirect(base_url().'home/sempermissao', 'refresh');
```

E que quando ele não estiver logado será redirecionado para a página de login.

redirect(base_url().'home/login', 'refresh');

Incluí também um método na biblioteca para checar se o usuário logado no sistema tem permissão para acessar um determinado método de uma classe chamado check_menu.

Para utilizar a autenticação no menu, basta checar se o usuário tem permissão para o caminho. Isto pode ser feito diretamente na view do menu.

```
<?php
    if($this->auth->check_menu('home','index')==1){
        echo "<li><a href='". base_url() ."home'>Home</a></li>";
    }
?>
```

Observe também que na tabela sys_metodos existe um campo chamado privado que tem por objetivo liberar o acesso de métodos que estejam protegidos, caso você construa um sistema de admin para gerenciar esta tabela, quando um método está com o campo privado definido como 1 ele está com seu acesso restrito somente a usuários com permissão de acesso e quando está ajustado como 0 está liberado como método público.

Outro detalhe desta tabela é um “apelido”, ele é cadastrado num primeiro momento como classe/metodo mas poderá ser alterado para qualquer nome amigável, mas compreensivel para os usuários. Como por exemplo.

Home/login poderia ser substituído por Acesso a tela de login.

Para autenticar um usuário no sistema, crie um formulário com os campos: usuario, cnpj e senha e aponte para uma classe de sua preferência (eu uso a classe de nome home) com os seguintes métodos.

```
<?php
class Home extends Controller {

    function __construct(){
        parent::Controller();
        $this->load->helper('logs');
        $this->load->helper('cookie');
    }

    function index(){
        redirect(base_url().'home/login', 'refresh');
    }

    function void(){
        $data['js_to_load'] = null;
        $this->load->view('libs/html-header',$data);
        $this->load->view('libs/menu');
        $this->load->view('libs/html-footer');
    }
    function sempermissao(){

        echo "<html>”;
        echo “<title>Acesso Negado</title>”;
        echo “<body bgcolor='#EEEEEE'>”;
        echo “<div style='padding:20px;background-color:#FFCC00;'>”;
        echo “<h2>Você não tem permissão para acessar esta funcionalidade.</h2>
        echo “</div>”;
```

```
        echo "</body>";
        echo "</html>";
        exit();
    }

    function login(){
        $this->load->view('login',$data);
    }

    function dologin(){
        $usuario = $this->input->post('usuario');
        $cnpj = $this->input->post('cnpj');
        $senha = md5($this->input->post('senha'));

        if($usuario=="" || $cnpj=="" || $this->input->post('senha')==""){
            redirect(base_url().'home/login', 'refresh');
            exit();
        }

        if(isset($_POST['lembrar'])){
            setcookie("usuario", $usuario);
            setcookie("cnpj", $cnpj);
            setcookie("lembrar", "checked");
        }

        $sql = "SELECT id,cnpj,login,nome,email
                FROM tb_usuarios
                WHERE login ='" . $usuario . "'
                AND cnpj ='" . $cnpj . "'
                AND senha ='" . $senha . "'";

        $query = $this->db->query($sql);
        $result = $query->result();
        if(count($result)<1){
            redirect(base_url().'home/login', 'refresh');
            exit();
        }
        else{
            $login = array(
                    'id_usuario'   =>    $result[0]->id,
                    'cnpj'   =>   $result[0]->cnpj,
                    'usuario'   =>   $result[0]->login,
                    'nome'      =>   $result[0]->nome,
                    'email'     =>    $result[0]->email,
                    'logged_in' => TRUE,
                    'data' => date("d/m/Y h:i:s")
                );

            $data['ip'] = getenv("REMOTE_ADDR");
            $data['usuario'] = $result[0]->id;
            $this->db->insert('tb_acessos',$data);

            $this->session->set_userdata($login);
            redirect(base_url().'home/void', 'refresh');
        }
    }

    function logout()
    {
        $this->session->sess_destroy();
        $this->login();
    }
}
```

Com isso concluímos nosso sistema de autenticação com controle de nível de acesso por classe e método. Não é exatamente a coisa mais simples do mundo, mas também não é a mais complicada. É importante resaltar que estou guardando as sessões do codeIgniter no banco de dados.

As demais funções de cadastro de usuário, e cadastro de usuários na tabela de permissão devem ser construídos separadamente.

Implementações futuras estão previstas e um inconveniente deste sistema é que quando você altera um nome de classe ou método será necessário ajustar a tabela sys_metodos.

Código fonte da biblioteca disponível aqui.

Espero que seja útil.