Bài 17: Chức năng thêm – xóa – sửa trong CodeIgniter

Để có thể hiểu rõ cách thức thêm – xóa – sửa dữ liệu trong CodeIgniter, chúng ta sẽ thực hiện chức năng quản lý thành viên.

1. Xây dựng database

Chúng ta xây dựng bảng User có cấu trúc như sau.

```
CREATE TABLE `user` (
  `id` int(11) unsigned NOT NULL AUTO_INCREMENT,
  `username` varchar(100) COLLATE utf8_unicode_ci DEFAULT NULL,
  `password` char(32) COLLATE utf8_unicode_ci DEFAULT NULL,
  `email` varchar(100) COLLATE utf8_unicode_ci DEFAULT NULL,
  `level` int(1) DEFAULT '1',
  PRIMARY KEY (`id`)
) ENGINE=InnoDB AUTO_INCREMENT=23 DEFAULT CHARSET=utf8
COLLATE=utf8_unicode_ci;
```

Sau đó thêm vài dòng dữ liệu mẫu như sau:

```
insert  into `user`(`id`,`username`,`password`,`email`,`level`) values
(1,'admin','123','admin@tkpro.edu.vn,2),
(2,'htkhai','123456','lienhe@tkpro.edu.vn,1);
```

Sau khi hoàn tất việc tạo dữ liệu, thì chúng ta tiến hành khai báo cấu hình database trong CI như sau (trong file *application/config/database.php*).

```
$db['default']['hostname'] = 'localhost';
$db['default']['username'] = 'root';
$db['default']['password'] = '';
$db['default']['database'] = 'codeigdb';
$db['default']['dbdriver'] = 'mysql';
$db['default']['dbprefix'] = '';
$db['default']['pconnect'] = TRUE;
$db['default']['db_debug'] = TRUE;
$db['default']['cache_on'] = FALSE;
$db['default']['cachedir'] = '';
$db['default']['char_set'] = 'utf8';
$db['default']['dbcollat'] = 'utf8_general_ci';
$db['default']['swap_pre'] = '';
$db['default']['autoinit'] = TRUE;
$db['default']['stricton'] = FALSE;
```

Các bạn sửa lại thông số cho đúng với máy tính của mình.

Hoàn thành công việc cấu hình database xong, tiếp theo chúng ta cho nó autoload, đỡ phải mất công khai báo trong file model. Với chức năng chúng ta đang xây dựng, chúng ta sẽ dùng tới session, form validation, database, nên chúng ta autolod các thư viện này. Để autoload thư viện, trong folder config, mở file autoload.php lên và tìm đến dòng 55 thêm vào như sau:

```
$autoload['libraries'] = array('database', 'session', 'form_validation');
```

Như vậy là xong bước 1, tiếp theo chúng ta sẽ cấu hình **master layout** để tiện việc thao tác với giao diện hơn.

2. Cấu hình master layout

Vào folder application/views, tạo folder user và thêm 4 file vào folder này. File main.php là file chứa giao diện cố định. Tiếp theo chúng ta tạo controller user và action index, tiến hành khai báo thông tin cấu hình master layout như sau.

```
class User extends CI_Controller {

    protected $_data;

    public function __construct() {
        parent::__construct();
        $this->load->helper('url');
    }

    public function index() {
        $this->_data['subview'] = 'user/index_view';
        $this->_data['titlePage'] = 'List All User';
        $this->load->view('user/main.php', $this->_data);
    }
}
```

Chúng ta khởi tạo biến $_data để có thể gọi trong mọi action. Tiếp theo chúng ta vào file main.php xử lý html và load **$subview** vào div content. Chúng ta sẽ phải dùng tới hàm **base_url()** nên các bạn gọi helper url ngay trong constructor để có thể tái sử dụng lại trong mọi action.

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
   <head>
      <title><?php echo $titlePage; ?></title>
      <meta http-equiv="content-type" content="text/html;charset=utf-8" />
      <meta name="author" content="Hasegawa Kaito" />
      <link rel="stylesheet" type="text/css" href="<?php echo base_url();
?>asset/admin/css/style.css" charset="utf-8" />
      <script language="javascript">
         function xacnhan() {
            if (!window.confirm('You want delete user ?')) {
               return false;
             }
          }
        </script>
    </head>
    <body id="manage">
        <div id="header">
```

```
            <ul id="menu">
                <li><a href="<?php echo base_url();
?>index.php/user">User</a></li>
                <li><a href="#">Chuyên mục</a></li>
                <li><a href="#">Bài viết</a></li>
                <li><a href="#">Comment</a></li>
            </ul>
            <p>
                Xin chào <span>Admin</span>
                <a href="#">[ Thoát ]</a>
            </p>
        </div><!-- End header -->
        <div id="content"><!-- ========== Begin #content ============ -->
            <?php echo $this->load->view($subview); ?>
        </div><!-- End #content -->
        <div id="footer">
            &copy; 2018, TKPro.edu.vn
        </div><!-- End #footer -->
    </body>
</html>
```

Chúng ta tạo thêm folder asset/admin/css dùng để chứa file css của layout.

```
*{margin: 0px; padding: 0px; font: normal 12px/120% tahoma; color: #333;}
ul{list-style: none;}
a{text-decoration: none;}
body{ width: 960px; margin: 0px auto;}
a:hover, a:active, a:visited{color: #268;}
h1,h2,h3,h4,h5,h6{font: 12px/140% tahoma;}
html{background: #f0f0f0;}

/*==== login ====*/
#header p{text-align: center; margin: 10px; color: #f1f1f1;}
#header p a{color: #f1f1f1; font-weight: 900;}
#header p span{color: #ffffcf; font-weight: 900;}
#header p a:hover{text-decoration: underline;}

.show{padding: 20px 10px; margin-bottom: 10px; background: #dfdfdf;
    border-left: 1px solid #bfbfbf; border-top: 1px solid #bfbfbf;
    border-right: 1px solid #fff; border-bottom: 1px solid #fff;
}
.show legend{color: #268; font: 900 14px arial; padding: 0px 10px;}
.show label{display: block; float: left; width: 110px; margin-top: 2px;}

.input,select{margin-bottom: 5px;height:20px;padding-left:3px;}
.input:focus{background: #ffffcf; border: 1px solid #cfcfcf;}

.button{display: block; margin: auto;}
.btn{background: green; color: white; border:none; padding:5px;
    font-weight: bold;}
.btn:hover{ background: orange; cursor: pointer;}
```

```
/*==== manage ====*/
#header{background: #999; border-bottom: 1px solid #fff;
    padding-bottom: 5px; margin: 10px 0px 10px 0px;}

#menu{overflow: hidden; margin: 0px auto; width: 400px;}
#menu li{float: left;}
#menu li a{font: 900 12px/30px arial; color: #f5f5f5; width: 98px;
    display: block; text-align: center;
    border-top: 1px solid #afafaf; border-bottom: 1px solid #afafaf;
    border-left: 1px solid #afafaf; border-right: 1px solid #777;}
#menu li a:hover{color: #fff; background: #888;}

#footer{background: #999; height: 30px; line-height: 30px;
    text-align: center; color: #f5f5f5;}

#user,#categories,#articles,#comment{margin: 20px auto; width: 400px;}

#articles{width: 960px;}
#comment{width: 500px;}

#user fieldset .input{width: 220px;}
#articles fieldset textarea,#comment fieldset textarea{margin-bottom: 5px;}
#articles fieldset textarea:focus,#comment fieldset
textarea:focus{background: #ffffcf;border: 1px solid #cfcfcf;}

ul.err{padding: 10px;}
ul.err li{color: #f00;}

.message{color: green; text-align: center; line-height: 30px;
    font-size: 14px;}

.list{width: 960px; margin: 0px auto 20px; border: 1px solid #cfcfcf;}
.list tr:hover{background: #ffffcf;}
.list th{font: 900 12px/30px tahoma; background: #cfcfcf;
    border-left: 1px solid #efefef; border-right: 1px solid #afafaf;}
.list td{text-align: center; line-height: 26px;
    border-top: 1px solid #cfcfcf;}
.list_page{height: 26px;}
.list_page span{color: #333; padding: 2px 5px; margin: 0px 5px;
    border: 1px solid #cfcfcf; font-weight: 900;}
.list_page a{margin: 0px 4px;}

#list_user,#list_articles,#list_comment{width: 960px;}
#list_cate{width: 600px;}

.admin{ color:red; font-weight: bold;}

.mess_succ{background: #99CC00;color: #FFFFFF;text-align: center;
font-weight: bold;margin:15px;}
```

```
.mess_succ ul{list-style-type:none;padding:0px;margin:0px;}
.mess_succ ul li{padding:10px 10px 10px 15px;}
.mess_error{background: #FF6600; color: #FFFFF; margin:15px;}
.mess_error ul{list-style-type:none; padding:0px; margin:0px;}
.mess_error ul li{padding:10px 10px 10px 15px;}
```

Tại file index_view.php, chúng ta cho nội dung đơn giản để test xem cấu hình master layout thành công chưa, chạy link localhost/citest/index.php/user.

Sau khi cấu hình thành công master layout, việc tiếp theo là chúng ta sẽ liệt kê danh sách các thành viên ra layout.

3. Viết chức năng liệt kê thành viên

Nhắc tới liệt kê danh sách thành viên thì chúng ta sẽ nghĩ ngay tới việc phải thao tác với model & active record, vậy trong folder models, chúng ta tạo file muser.php.

```
<?php
class Muser extends CI_Model{
    protected $_table = 'user';
    public function __construct() {
        parent::__construct();
    }

    public function getList(){
        $this->db->select('id, username, email, level');
        return $this->db->get($this->_table)->result_array();
    }

   public function countAll(){
        return $this->db->count_all($this->_table);
    }
}
```

Chúng ta không muốn liệt kê toàn bộ cột trong bảng user mà chỉ hiển thị các cột: id, email, username, level. Do chúng ta đang viết hàm vì thế cần phải trả kết quả về với cú pháp như sau: return $this->db->get($this->_table)->result_array(). Với phương thức result_array(), chúng ta sẽ liệt kê toàn bộ record có trong table. Chúng ta cũng có thể đếm tổng số record trong table với phương thức count_all().

Quay lại controller user, tiến hành load model ra, gán biến $this->_data['info'] trỏ tới phương thức **getList** trong model, sau đó sang index_view in dữ liệu ra

```
echo "<pre>";
print_r($info);
echo "</pre>";
```

```
Array
(
    [0] => Array
        (
```

```
            [id] => 1
            [username] => admin
            [email] => admin@tkpro.edu.vn
            [level] => 2
        )

    [1] => Array
        (
            [id] => 2
            [username] => htkhai
            [email] => lienhe@tkpro.edu.vn
            [level] => 1
        )
)
```

Chạy link *localhost/citest/index.php/user*, nếu trình duyệt trả về kết quả như hình trên thì chúng ta đã lấy dữ liệu thành, tiếp theo chúng ta sẽ dùng vòng lặp foreach đổ dữ liệu vào table để hiển thị đẹp mắt hơn nhé.

```
<table cellpadding="0" cellspacing="0" border="0" width="100%"
id="list_cate" class="list">
    <tr>
         <td colspan="6" align="center"><a href="<?php echo base_url();
?>index.php/user/add">Add User</a><br /></td>
     </tr>
    <tr>
        <th width="20">STT</th>
        <th width="80">Name</th>
        <th width="80">Email</th>
        <th width="30">Level</th>
        <th width="10">Edit</th>
        <th width="10">Delete</th>
    </tr>
    <tr>
        <?php
            $stt=0;
            foreach($info as $item){
                $stt++;
                echo "<tr>";
                    echo "<td>$stt</td>";
                    echo "<td>$item[username]</td>";
                    echo "<td>$item[email]</td>";
                    if($item['level'] == 2){
                        echo "<td class='admin'>Administrator</td>";
                    }else{
                        echo "<td>Member</td>";
                    }
                    echo "<td><a
href=".base_url()."index.php/user/edit/$item[id]>Edit</a></td>";
                    echo "<td><a
href=".base_url()."index.php/user/del/$item[id]  onclick='return
xacnhan();'>Delete</a></td>";
```

```
                echo "</tr>";
            }
        ?>
    </tr>
     <tr>
         <td colspan="6" align="center">Có tổng cộng <?php echo
$total_user; ?> thành viên.<br /></td>
     </tr>
</table>
```

Chúng ta có 6 cột tất cả, khởi tạo $stt = 0 ở phía ngoài vòng lặp sau đó cho nó tăng dần $stt++, check ngay chỗ hiển thị level, nếu $info['level'] = 2 thì là administrator, ngược lại bằng 1 thì là member, $total_user là biến truyền từ controller sang thông qua hàm countAll trong model.

```
<table cellpadding="0" cellspacing="0" border="0" width="100%"
id="list_cate" class="list">
    <tr><th width="20">STT</th>
        <th width="80">Name</th>
        <th width="80">Email</th>
        <th width="30">Level</th>
        <th width="10">Edit</th>
        <th width="10">Delete</th>
    </tr>
    <tr>
        <?php
            $stt=0;
            foreach($info as $item){
                $stt++;
                echo "<tr>";
                    echo "<td>$stt</td>";
                    echo "<td>$item[username]</td>";
                    echo "<td>$item[email]</td>";
                    if($item['level'] == 2){
                        echo "<td class='admin'>Administrator</td>";
                    }else{
                        echo "<td>Member</td>";
                    }
                    echo "<td><a
href=".base_url()."index.php/user/edit/$item[id]> Edit</a></td>";
                    echo "<td><a
href=".base_url()."index.php/user/del/$item[id]  onclick='return
xacnhan();'>Delete</a></td>";
                echo "</tr>";
            }
        ?>
    </tr>
     <tr><td colspan="6" align="center">Có tổng cộng <?php echo
$total_user; ?> thành viên.<br /></td>
     </tr>
</table>
```

Vậy chúng ta có action index như sau.

```
public function index() {
        $this->_data['subview'] = 'user/index_view';
        $this->_data['titlePage'] = 'List All User';
        $this->load->model('Muser');

        $this->_data['info'] = $this->Muser->getList();
        $this->_data['total_user'] = $this->Muser->countAll();
        $this->load->view('user/main.php', $this->_data);
}
```

Tiếp theo chúng ta sẽ thao tác với một chức năng khá đơn giản với bất kỳ ứng dụng nào, đó là chức năng Delete User.

4. Viết chức năng delete user

Chúng ta tạo thêm action del, đây là action thao tác đến việc xóa user.

```
public function del($id) {
        $this->load->model('Muser');
        $this->Muser->deleteUser($id);
        $this->session->set_flashdata("flash_mess", "Delete Success");
        redirect(base_url() . "index.php/user");
 }
```

Giải thích cho việc truyền $id vào hàm del: thay vì chúng ta dùng uri->segment để xác định vị trí của id, vì thật chất nó là phương thức **GET** trong php thuần nên nó không thật sự an toàn, vì thế chúng ta có thể đếm vị trí id bằng cách sau: Ví dụ link của chúng ta có cấu trúc là *localhost/citest/index.php/user/del/id...* thì user là 1, del là 2, id là 3. Vậy ngay tại action del ta truyền trực tiếp $id vào nó vẫn hiểu id đang ở vị trí thứ 3.

Với việc delete một user, chúng ta cũng cần phải xuất ra thông báo là delete thành công hay gì đó, và chúng ta lựa chọn giải pháp dùng **flashdata** trong **session** xuất ra thông báo, vì flashdata chỉ xuất hiện 1 lần, sau khi F5 thì nó sẽ biến mất.

Vào file model muser, viết phương thức **deleteUser()**, muốn xóa một id nào đó chúng ta phải so sánh id bằng với $id mà chúng ta đang muốn truyền vào. Sau đó mình sẽ sử dụng $this->db->delete($this->_table), ngay tại controller chúng ta có cú pháp như sau: $this->Muser->deleteUser($id). Sau khi xóa xong chúng ta phải xuất ra thông báo, vậy ngay tại action del, chúng ta khởi tạo flashdata bằng cú pháp $this->session->set_flashdata("flash_mess", "Delete Success"), flash_mess là khóa chúng ta đặt tên, còn tham số thứ 2 là câu thông báo lỗi muốn xuất ra.

Khởi tạo xong, tại action index, chúng ta gọi nó ra bằng cú pháp $this->_data['mess'] = $this->session->flashdata('flash_mess'). Tại index_view, chúng ta thêm đoạn code sau để có thể xuất ra thông báo lỗi, xóa xong tự về trang list user với hàm **redirect**.

```
<?php
    if(isset($mess) && $mess != ''){
```

```
        echo "<div class='mess_succ'>";
        echo "<ul>";
            echo "<li>$mess</li>";
        echo "</ul>";
        echo "</div>";
    }
?>
```

Nếu tồn tại $mess và $mess khác rỗng thì xuất ra thông báo là Delete Success. Bay giờ chúng ta refresh trình duyệt và click xóa 1 user nào đó và xem kết quả.

Đã xong chức năng delete user, tiếp theo chúng ta sẽ viết tiếp chức năng add user.

5. Viết chức năng add user

Vào controller user tạo thêm action add, với các thông số như sau.

```
public function add() {
       $this->_data['titlePage'] = 'Add A User';
       $this->_data['subview'] = 'user/add_view';
       $this->load->view('user/main.php', $this->_data);
   }
```

Vào trong file add_view.php thêm vào html như sau.

```
<form action="<?php echo base_url(); ?>index.php/user/add" method="post"
id="categories">
        <?php
        echo "<div class='mess_error'>";
        echo "<ul>";
            if(validation_errors() != ''){
                echo "<li>".validation_errors()."</li>";
            }
        echo "</ul>";
        echo "</div>";
        ?>
    <fieldset class="show">
        <legend align="center">Username Infomations</legend>
        <label>Username:</label><input type="text" name="username"
size="28" class="input"/>
        <label>Email:</label><input type="text" name="email" size="28"
class="input"/>
        <label>Password:</label><input type="password" name="password"
size="28" class="input"/>
        <label>Re-Pass:</label><input type="password" name="password2"
size="28" class="input"/><br />
        <label>Level:</label><select name="level">
            <option value="1" selected>Member</option>
            <option value="2" >Administrator</option>
        </select></br />
        <label> </label><input type="submit" name="ok" value="Add
User" class="btn" />
    </fieldset>
</form>
```

Hàm validation_errors() dùng để xuất ra thông báo lỗi kiểm soát dữ liệu nhập vào. Chúng ta chạy link *localhost/citest/index.php/user/add* và xem kết quả.

Sau khi tạo form thành công, việc tiếp theo là chúng ta phải xử lý kiểm tra xem người dùng có nhấn nút submit hay chưa, nếu chưa thì báo lỗi, sử dụng **form validation** tạo ra các luật theo ý của các bạn.

Ở đây mình không giải thích lại toàn bộ các luật trên, ví chúng ta đã tìm hiểu chúng trong các bài trước rồi nhé, mình chỉ nói về hai luật mới, đó là matches & callback. Luật matches kiểm tra xem password có trùng với repass hay không, giá trị truyền vào là tên form của re-pass. Luật callback sẽ kiểm tra sự trùng lặp dữ liệu trong database.

Sau khi khai báo các luật validation, chúng ta cần kiểm tra xem các **set_rules** ở trên có hợp lệ hay không bằng phương thức $this->form_validation->run(), nếu nó bằng TRUE thì chúng ta tiến hành insert dữ liệu vào database. Nhưng trước khi insert, chúng ta phải kiểm tra xem user đó đã có trong csdl hay chưa, để làm được thao tác đó chúng ta sẽ có phương thức **check_user** ngay tại controller như sau.

```
public function check_user($user) {
        $this->load->model('Muser');
        $id=$this->uri->segment(3);
        if ($this->Muser->checkUsername($user) == FALSE) {
            $this->form_validation->set_message("check_user", "Your
username has been registed, please try again!");
            return FALSE;
        } else {
            return TRUE;
        }
    }
```

Tương tự như username, chúng ta cũng cần check luôn email xem có trùng lặp hay không, 2 phương thức khá giống nhau:

```
public function check_email($email) {
        $this->load->model('Muser');
        $id=$this->uri->segment(3);
        if ($this->Muser->checkUsername($email) == FALSE) {
            $this->form_validation->set_message("check_email", "Your email
has been registed, please try again!");
            return FALSE;
        } else {
            return TRUE;
        }
    }
```

Như vậy chúng ta phải có một phương thức trong model để thực thi check_user & check_email, vào file muser tạo ra 2 phương thức, checkUsename & checkEmail, với các thông số sau.

```
public function checkUsername($user){
        $this->db->where('username',$user);
```

```
        $query=$this->db->get($this->_table);
        if($query->num_rows() > 0){
            return FALSE;
        }else{
            return TRUE;
        }
    }

    public function checkEmail($email){
        $this->db->where('email',$email);
        $query=$this->db->get($this->_table);
        if($query->num_rows() > 0){
            return FALSE;
        }else{
            return TRUE;
        }
    }
```

Nếu nó lớn hơn 0 tức là nó tồn tại rồi thì chúng ta chỉ return FALSE, nếu nó tồn tại thì xuất ra thông báo lỗi.

Sau khi hoàn tất việc kiểm tra trùng lặp dữ liệu, bước tiếp theo chúng ta tiến hành insert. Quay lại file muser viết phương thức insert như sau.

```
public function insertUser($data_insert){
        $this->db->insert($this->_table,$data_insert);
    }
```

Tạo ra một cái array, chứa các thông tin mà chúng ta cần thêm vào CSDL, chúng ta dùng phương thức $this->input->post() để thêm dữ liệu. Như vậy, insert xong sẽ chuyển về trang index, đồng thời xuất ra thông báo Added.

```
public function add() {
       $this->_data['titlePage'] = 'Add A User';
       $this->_data['subview'] = 'user/add_view';

       $this->form_validation->set_rules("username", "Username",
"required|xss_clean|trim|min_length[4]|callback_check_user");
       $this->form_validation->set_rules("password", "Password",
"required|matches[password2]|trim|xss_clean");
       $this->form_validation->set_rules("email", "Email",
"required|trim|xss_clean|valid_email|callback_check_email");

       if ($this->form_validation->run() == TRUE) {
           $this->load->model("Muser");
           $data_insert = array(
               "username" => $this->input->post("username"),
               "password" => $this->input->post("password"),
               "email"     => $this->input->post("email"),
               "level"     => $this->input->post("level"),
           );
           $this->Muser->insertUser($data_insert);
```

```
            $this->session->set_flashdata("flash_mess", "Added");
            redirect(base_url() . "index.php/user");
        }
        $this->load->view('user/main.php', $this->_data);
    }
```

Như vậy là chúng ta đã hoàn tất việc thêm một thành viên vào CSDL, tiếp theo chúng ta sẽ viết chức năng edit user.

6. Viết chức năng edit user

Để edit được, các bạn phải nhấn vào link edit và khi đó thông qua một id mà chúng ta muốn sửa thể hiện trên đường dẫn, chúng ta load lên thông tin của user cần sửa. Sau đó, chúng ta sẽ cập nhập dữ liệu từ trong form ngược trở lại với CSDL, và dĩ nhiên phải trải qua thao tác validation trước khi tiến hành update vào CSDL. Đó là mục đích mà chúng ta phải cần làm ở trong phần edit này.

Phần này khá giống với các thao tác mà chúng ta đã làm ở phần add user, cho nên phần này mình không giải thích nhiều, chỉ tập trung vào sự khác biệt giữa add & edit.

Chúng ta tạo action edit và truyền $id vào, để có thể edit bất kì record nào thì chúng ta phải lấy ra được cái id của record đó. Chúng ta vào file muser viết phương thức **getUserById** như sau (row_array là hàm hỗ trợ chúng ta lấy ra một record):

```
public function getUserById($id){
        $this->db->where("id", $id);
        return $this->db->get($this->_table)->row_array();
    }
```

Sau khi có phương thức trên, chúng ta sẽ tiến hành lấy dữ liệu ra form ra bằng cú pháp $this->_data['info'] = $this->Muser->getUserById($id). Sau đó, chúng ta sang edit_view copy phần form bên add_view vào đổi lại dường dẫn trong form như sau:

```
<form action="<?php echo base_url(); ?>index.php/user/edit/<?php echo
$info['id']; ?>" method="post" id="categories">
        <?php
        echo "<div class='mess_error'>";
        echo "<ul>";
            if(validation_errors() != ''){
                echo "<li>".validation_errors()."</li>";
            }
        echo "</ul>";
        echo "</div>";
        ?>
    <fieldset class="show">
        <legend align="center">Edit Username: <?php echo $info['username'];
?></legend>

        <label>Username:</label><input type="text" name="username"
value="<?php echo $info['username']; ?>" size="28" class="input"/>
```

```
        <label>Email:</label><input type="text" name="email" size="28"
value="<?php echo $info['email']; ?>" class="input"/>
        <label>Password:</label><input type="password" name="password"
size="28" class="input"/>
        <label>Re-Pass:</label><input type="password" name="password2"
size="28" class="input"/><br />

        <label>Level:</label><select name="level">
            <option value='1' <?php if ($info['level'] == 1) echo '
selected ';  ?> >Member</option>
            <option value='2'  <?php if ($info['level'] == 2) echo '
selected ';  ?> >Administrator</option>
        </select></br />
        <label> </label><input type="submit" name="ok" value="Edit
User" class="btn" />
    </fieldset>
</form>
```

Tiếp theo, chúng ta quay lại file muser và viết phương thức **updateUser** như sau:

```
public function edit($id) {
        $this->load->library("form_validation");
        $this->load->model('Muser');
        $this->_data['titlePage'] = "Edit A User";
        $this->_data['subview'] = "user/edit_view";

        $this->_data['info'] = $this->Muser->getUserById($id);
        $this->form_validation->set_rules("username", "Username",
"required|xss_clean|trim|min_length[4]|callback_check_user");
        $this->form_validation->set_rules("password", "Password",
"matches[password2]|trim|xss_clean");
        $this->form_validation->set_rules("email", "Email",
"required|trim|xss_clean|valid_email|callback_check_email");
        if ($this->form_validation->run() == TRUE) {
            $data_update = array(
                "username" => $this->input->post("username"),
                "email" => $this->input->post("email"),
                "level" => $this->input->post("level"),
            );
            if ($this->input->post("password")) {
                $data_update['password'] = $this->input->post("password");
            }
            $this->Muser->updateUser($data_update, $id);
            $this->session->set_flashdata("flash_mess", "Update Success");
            redirect(base_url() . "index.php/user");
        }
        $this->load->view('user/main.php', $this->_data);
    }
```

Khá giống với action add, chỉ khác ở vài chỗ: ví dụ như các bạn chỉ muốn change email, username không muốn change password thì check như sau: nếu tác động vào textbox password thì mới update, còn không vẫn giữ nguyên password cũ.

```
if ($this->input->post("password")) {
   $data_update['password'] = $this->input->post("password");
}
```

Vấn đề chúng ta mắc phải ở đây là, chúng ta đang dùng lại 2 phương thức check_user & check_email, và sự khác biệt đối với edit thì chúng ta phải kiểm tra thêm cái id nữa. Tuy nhiên chúng ta hoàn toàn có thể sử dụng một lúc cả 2 phương thức cho add & edit bằng cách: quay trở lại file muser ngay tại vị trí của 2 phương thức, chúng ta thêm vào $id = "", đối với add thì không có id và với edit thì có id. Nếu $id != "" tức là nó có giá trị thì chúng ta sẽ có $this->db->where("id !=", $id), tương tự ở email cũng vậy.

```
public function checkUsername($user, $id=""){
        if($id != ""){
            $this->db->where("id !=", $id);
        }
        $this->db->where('username',$user);
        $query=$this->db->get($this->_table);
        if($query->num_rows() > 0){
            return FALSE;
        }else{
            return TRUE;
        }
    }

    public function checkEmail($email,$id=""){
        if($id != ""){
            $this->db->where("id !=", $id);
        }
        $this->db->where('email',$email);
        $query=$this->db->get($this->_table);
        if($query->num_rows() > 0){
            return FALSE;
        }else{
            return TRUE;
        }
    }
```

Như vậy, trong controller user, tại vị trí check_user chúng ta chỉnh sửa lại như sau:

```
public function check_user($user, $id) {
        $this->load->model('Muser');
        $id=$this->uri->segment(3);
        if ($this->Muser->checkUsername($user, $id) == FALSE) {
            $this->form_validation->set_message("check_user", "Your
username has been registed, please try again!");
            return FALSE;
        } else {
            return TRUE;
        }
    }

    public function check_email($email,$id) {
```

```
        $this->load->model('Muser');
        $id=$this->uri->segment(3);
        if ($this->Muser->checkEmail($email, $id) == FALSE) {
            $this->form_validation->set_message("check_email", "Your email
has been registed, please try again!");
            return FALSE;
        } else {
            return TRUE;
        }
    }
```

Toàn bộ action edit:

```
public function edit($id) {
        $this->load->model('Muser');
        $this->_data['titlePage'] = "Edit A User";
        $this->_data['subview'] = "user/edit_view";

        $this->_data['info'] = $this->Muser->getUserById($id);
        $this->form_validation->set_rules("username", "Username",
"required|xss_clean|trim|min_length[4]|callback_check_user");
        $this->form_validation->set_rules("password", "Password",
"matches[password2]|trim|xss_clean");
        $this->form_validation->set_rules("email", "Email",
"required|trim|xss_clean|valid_email|callback_check_email");
        if ($this->form_validation->run() == TRUE) {
            $data_update = array(
                "username" => $this->input->post("username"),
                "email" => $this->input->post("email"),
                "level" => $this->input->post("level"),
            );
            if ($this->input->post("password")) {
                $data_update['password'] = $this->input->post("password");
            }
            $this->Muser->updateUser($data_update, $id);
            $this->session->set_flashdata("flash_mess", "Update Success");
            redirect(base_url() . "index.php/user");
        }
        $this->load->view('user/main.php', $this->_data);
    }
}
```