HƯỚNG DẪN CODEIGNITER

Giới thiệu

Tài liệu này mình tham khảo trên mạng và chép lại. Đây là tài liệu hướng dẫn sử dụng cho CodeIgniter các phiên bản 3, các phiên bản sau này (từ bản 4 trở đi) có thể sẽ khác (về thư mục và cách viết code). Nên tài liệu này chỉ sử dụng tốt cho các phiên bản 3.

Nội dung tài liệu gồm các phần sau:

1. TỔNG QUAN

Các thư mục trong ci:

- **application**: chứa ứng dụng, nơi ta viết các file code cho website.
- **system**: chứa thư viện của ci, không nên làm gì ở đây.
- **user_guide**: thư mục hướng dẫn, có thể xóa thư mục này.

Các thư mục con trong application:

- ***config***: chứa cấu hình website, database, đường dẫn, ngôn ngữ...
- ***core***: viết core của hệ thống, có thể viết các main controller, model...
- ***controller***: chứa các file xử lý dữ liệu
- ***model***: các file làm việc với csdl
- ***views***: chứa các file hiển thị web
- ***helpers***: chứa các hàm tự xây dựng
- ***libraries***: chứa các thư viện tự phát triển
- ***language***: chứa các file ngôn ngữ.

2. CONTROLLER VÀ URLS

URLs

- Mặc định các truy vấn đều gọi qua trang index.php.
- Các phân đoạn trên một url có dạng:

```
example.com/class/function/id
```

Trong đó:

- Phân đoạn đầu đại diện cho class (hay controller) được gọi lên.
- Phân đoạn 2 đại diện cho phương thức trong class.
- Phân đoạn 3 trở đi đại diện cho các tham số của phương thức hoặc các biến gửi lên url.

Để lấy giá trị của phân đoạn trên url sử dụng cú pháp:

```
$this->uri->segment('n');
```

Trong đó n chính là vị trí các phân đoạn trên url.

Trường hợp url đã được rewrite thì sử dụng cú pháp sau để có thể lấy chính xác giá trị phân đoạn:

```
$this->uri->rsegment('n');
```

Ẩn tên index.php trên url:

Tạo file .htaccess ngang hàng với file index.php với nội dung sau:

```
RewriteEngine on
```

```
RewriteCond $1 !^(index\.php|resources|robots\.txt)
RewriteCond %{REQUEST_FILENAME} !-f
RewriteCond %{REQUEST_FILENAME} !-d
RewriteRule ^(.*)$ index.php?$1 [L,QSA]
```

Controller

Quy tắc đặt tên controller và file controller:

- Tên controller phải viết hoa ký tự đầu tiên.

- Tên file controller giống tên class và viết thường.

Ví dụ:

```
class Test extends CI_Controller {
        //function...
}
```

Nếu trong controller có hàm khởi tạo thì trước hết cần kế thừa hàm khởi tạo của ci:

```
class Test extends CI_Controller {
        public function __construct()
        {
                parent::__construct();
                        //hàm khởi tạo của mình viết ở đây
        }
}
```

Ban đầu mặc định controller sẽ chạy khi truy cập trang web là controller Welcome, ta có thể thay đổi bằng cách vào file routes.php trong thư mục application/config/:

Tìm đến dòng:

```
$route['default_controller'] = 'ten_controller';
```

Trong đó ten_controller là tên controller mà bạn muốn chạy mặc định khi truy cập web.

Dòng code **if (! defined('BASEPATH')) exit('No direct script access allowed');** là dòng Security bảo vệ file của các bạn, nó không cho truy cập thẳng vào file mà phải thông qua file index.php ở mức ngoài cùng.

3. VIEW

Để sử dụng được view thì trong controller dùng cú pháp:

```
$this->load->view("folder/view_name");
```

Trong đó folder là thư mục con trong thư mục application/views/

Nếu file view_name nằm trong thư mục application/views thì sẽ là:

```
$this->load->view("view_name");
```

Trong các file view cũng có thể sử dụng cú pháp trên để load các file view khác (giống như include file), điều này thường được sử dụng để làm master layout, ví dụ load các view header, footer...

```
$this->load->view("layout/header");
```

Gửi tham số sang view

Cú pháp:

```
$this->load->view("url_view", $data);
```

Trong đó $data là dữ liệu gửi sang view, $data phải là một mảng dữ liệu, khi sang view thì khóa key của mảng $data chính là biến ở view.

Ví dụ, trong một controller:

```
public function index()
{
        $data = array();
        $data['title'] = 'hello';
        $this->load->view("home", $data);
}
```

Trong file home ở view có thể viết:

```
<body>
        <?php echo $title ?>
</body>
```

4. MODEL

– Model là tầng xử lý những tác vụ liên quan đến tương tác cơ sở dữ liệu từ những yêu cầu của controller.

– Muốn kết nối với CSDL. Chúng ta chỉ việc gọi phương thức:

```
$this->load->database();
```

Ví dụ trong model:

```
class User_model extends CI_Model {
```

```
    public function __construct()
    {
      parent::__construct();
      $this->load->database();
    }
   public function get_list()
 {
        return array();
  }
}
```

- Giống với controller, khi tạo lớp User_model thì chứ đầu tiên chúng ta phải viết hoa.
- Mọi model phải được kế thừa từ CI_Model
- Tên Model không được giống với tên controller, và tên class và tên file model phải giống nhau, tên file viết thường.
- Nếu trong model sử dụng hàm khởi tạo __construct() thì bạn cần thêm parent::__construct(); để kế thừa từ CI_Model (tránh bị ghi đè).

- Để gọi 1 file model nào đó ta sử dụng cú pháp:

```
$this->load->model('tên_model');
```

- Để gọi 1 phương thức trong model ta sử dụng:

```
$this->tên_model->tên_phương_thức();
```

Ví dụ trong controller:

```
class User extends CI_Controller {
    public function index()
    {
      //load model
      $this->load->model("user_model");
      //su dung phuong thuc get_list trong user_model
      $list = $this->user_model->get_list();
    }
}
```

Các thao tác trong model:

- Lựa chọn tên cột cần lấy:

```
$this->db->select('tên cột 1, tên cột 2');
```

- Nếu lấy hết các cột:

```
$this->db->select('*');
```

- Truy vấn với điều kiện:

```
$this->db->where('tên cột', 'giá trị');
```

- Hoặc có thể sử dụng mảng các điều kiện:

```
$this->db->where(array('tên cột'=>'giá trị'));
```

- Sắp xếp kết quả trả về theo một trường nào đó:

```
$this->db->order_by('tên cột asc hoặc desc');
```

- Hoặc `$this->db->order_by('tên cột', 'asc hoặc desc');`
- Giới hạn kết quả trả về:

```
$this->db->limit('số dòng dữ liệu muốn lấy', 'vị trí');
```

- Lấy tất cả record từ câu truy vấn:

```
$query->result_array();
```

- Dữ liệu trả về dạng đối tượng:

```
$query->result();
```

- Lấy 1 record trong dữ liệu:

```
$query->row_array();
```

- Dữ liệu trả về dạng đối tượng:

```
$query->row();
```

- Đếm số dòng dữ liệu:

```
$query->num_rows();
```

- Thêm dữ liệu:

```
$this->db->insert('tên bảng', 'mảng dữ liệu');
```

- Cập nhật dữ liệu (cho bản ghi có id = $id):

```
$this->db->where('id', $id);
$this->db->update('tên bảng', mảng dữ liệu');
```

- Xóa dữ liệu:

```
$this->db->where('id', $id);
$this->db->delete('tên bảng');
```

- Câu truy vấn bình thường:

```
$this->db->query('câu query');
```

5. HELPERS VÀ LIBRARIES

* Helper:

- Helper là các hàm được xây dựng để sử dụng khi lập trình.

- 1 số helper trong ci như: url, language, date...

- Có thể tự tạo helper.

Khi dùng có thể load tự động nếu dùng nhiều, trong file application/config/autoload.php:

```
$autoload['helper'] = array('url', 'date', 'language');
```

- Để tạo 1 helper vào thư mục application/helpers tạo file:

ten_helper.php

+ Khi tạo 1 file helper thì tên file bắt buộc phải có _helper.

+ Nội dung trong helper là các hàm được xây dựng.

- Để load helper sử dụng lệnh:

```
$this->load->helper('ten_helper');
```
hoặc
```
$this->load->helper('ten');
```

Load mảng helper:
```
$this->load->helper(array('ten1', 'ten2'));
```

* Library:

- Thư viện là nơi chứa các class được xây dựng sẵn.

- Có thể tự tạo thư viện.

- Để load tự động thư viện, mở file application/config/autoload.php tìm tới dòng

```
$autoload['libraries']
```

để khai báo các thư viện load tự động.

Ví dụ:

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

- Để tạo 1 library vào application/libraries tạo 1 file.

+ Tên file giống tên class.

+ Tên class viết hoa chữ cái đầu tiên.

- Load library:

```
$this->load->library('ten_library');
```

- Load nhiều library:

```
$this->load->library(array('ten1','ten2'));
```

- Sử dụng phương thức trong thư viện:

```
$this->ten_library->function();
```

Note: Khi sử dụng thư viện hay helper ở view, model, controller thì có thể dùng lệnh $this->load->.. nhưng khi sử dụng ngay trong helper hoặc library thì không dùng $this được, mà phải khởi tạo siêu đối tượng bằng lệnh & get_instance();

6. FORM VALIDATION

- Tải thư viện:

```
$this->load->library('form_validation');
```

- Tải helper đi kèm:

```
$this->load->helper('form');
```

- Sử dụng các phương thức:

```
$this->form_validation->phuong_thuc();
```

* Các phương thức:

- Phương thức ***set_rules***: dùng tạo ra các tập luật, có 3 tham số:

  + vị trí 1: tên của thẻ trong form muốn kiểm tra.

  + vị trí 2: tên khi xuất thông báo lỗi.

  + vị trí 3: tập luật đưa ra, cách nhau dấu "|".

Ví dụ:

```
$this->form_validation->set_rules('name', 'Họ tên', 'required|min_length[8]');
```

- Các tập luật hay dùng:

  + required: yêu cầu phải nhập không được để trống.

  + matches: sử dụng khi 2 ô nhập liệu giống nhau (nhập lại pass).

  + min_length: số ký tự tối thiểu phải nhập.

  + max_length: số ký tự tối đa phải nhập.

  + numeric: yêu cầu trong textbox phải nhập là con số.

  + valid_email: email nhập liệu bắt buộc đúng định dạng.

  + xss_clean: xóa xss của input, bảo mật.

  + exact_length: kiểm tra điều kiện bằng độ dài được gắn giá trị (exact_length[8])

  + greater_than: kiểm tra điều kiện phải lớn hơn độ dài được gắn giá trị (greater_than[8])

  + less_than: kiểm tra điều kiện phải nhỏ hơn độ dài được gắn giá trị (less_than[8])

  + callback_: gọi lại 1 hàm xử lý khác.

- Phương thức ***run***(): có nhiệm vụ kiểm tra các tập luật trên có hợp lệ hay không, nếu không hợp lệ nó sẽ trả về kết quả là false.

- Phương thức ***validation_errors()***: trong trường hợp phương thức run() trả về false nghĩa là 1 trong các tập luật không hợp lệ, lúc này có thể hiển thị thông báo lỗi bằng phương thức validation_errors().

- Phương thức ***form_error()***: chỉ hiển thị lỗi của 1 thẻ input.

```
form_error(ten_input);
```

- Phương thức ***set_value()***: cho phép lưu lại nội dung của thẻ input nhập vào trước đó (khi sai phải nhập lại).

```
set_value(ten_input);
```

Ví dụ:

```
<input type="text"  name="email" value="<?php echo set_value('email')?>"
/>
```

Ví dụ form đăng ký ở view:

```
<form name='register' action='' method='POST'>
        Tên đăng nhập: <input type="text" name="username"  value="<?php
echo set_value('username')?>">
        <div class="error" id="username_error"><?php echo
form_error('username')?></div>
        Mật khẩu: <input type="password" name="password">
       <div class="error" id="password_error"><?php echo
form_error('password')?></div>
        Nhập lại mật khẩu: <input type="password" name="re_password">
        <div class="error" id="re_password_error"><?php echo
form_error('re_password')?></div>
        Họ tên: <input type="text" name="name"  value="<?php echo
set_value('name')?>">
        <div class="error" id="name_error"><?php echo
form_error('name')?></div>
         Số điện thoại: <input type="text" name="phone"  value="<?php echo
set_value('phone')?>">
        <div class="error" id="phone_error"><?php echo
form_error('phone')?></div>
        Email: <input type="text" name="email"  value="<?php echo
set_value('email')?>">
        <div class="error" id="email_error"><?php echo
form_error('email')?></div>
       <input type="submit" value="Đăng ký">
```

```
</form>
```

Ở controller:

```
<?php if (!defined('BASEPATH')) exit('No direct script access allowed');
class User extends CI_Controller
{
   public function __construct()
   {
     parent::__construct();
   }
      /*
    * Phuong thuc dang ky thanh vien
   */
   public function register()
   {
     //load thu vien validation
     $this->load->library('form_validation');
     $this->load->helper('form');
     //tao cac tap luat
        //email:  bắt buộc - đúng định dạng email
     $this->form_validation->set_rules('email', 'Email',
'required|valid_email|xss_clean');
        //name:  bắt buộc - tối thiểu 8 ký tự
     $this->form_validation->set_rules('name', 'Họ và tên',
'required|min_length[8]|xss_clean');
        //phone:  bắt buộc - tối thiểu 8 ký tự - phai la số
     $this->form_validation->set_rules('phone', 'Số điện thoại',
'required|min_length[8]|numeric|xss_clean');
        //password:  bắt buộc - tối thiểu 6 ký tự - phai la số
     $this->form_validation->set_rules('password', 'Mật khẩu',
'required|min_length[6]|numeric|xss_clean');
        //re_password:  bắt buộc - phải giống với password
     $this->form_validation->set_rules('re_password', 'Nhập lại mật khẩu',
'required|matches[password]|xss_clean');
    //chạy và kiểm tra các tập luật
     if($this->form_validation->run())
```

```
        {
            //các dữ liệu nhập hợp lệ
            //đăng ký thành viên thành công
        }
        //hiển thị view
        $this->load->view('user/register');
    }
```

7. SESSION

Session trong CI là 1 mảng chứa các thông tin như:

+ ID là duy nhất của người sử dụng, được mã hóa bằng md5, được tự động sinh ra khi tạo session.

+ Địa chỉ IP của người dùng.

+ User Agent (trình duyệt) của người dùng.

+ timestamp (thời gian truy cập gần nhất).

- Lấy các tham số:

```
$this->session->userdata('tham_so');
```

Ví dụ:

```
$id = $this->session->userdata('session_id');
$ip_address = $this->session->userdata('ip_address');
$user_agent = $this->session->userdata('user_agent');
$last_activity = $this->session->userdata('last_activity');
```

- Tải thư viện:

```
$this->load->library('session');
```

(Cũng có thể load tự động trong file application/config/autoload.php:

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

)

- Khi sử dụng thư viện này phải cấu hình key trong file application/config/config.php:

```
$config['encryption_key'] = 'abcxyz123...';
```

key là chuỗi bất kỳ.

- Lưu thông tin vào session:

```
$this->session->set_userdata('tên', 'giá trị');
```

Giá trị có thể là 1 giá trị hoặc mảng các giá trị.

Ví dụ:

```
$userdata = array(
        'name' => 'linh',
        'email' => 'linhbk72@gmail.com',
        'logged_in' => true
        );
$this->session->set_userdata('user', $userdata);
```

- Lấy thông tin từ session:

```
$this->session->userdata('tên');
```

- Lấy toàn bộ session:

```
$this->session->all_userdata();
```

Ví dụ:

```
$userdata = $this->session->userdata('user');
print_r($userdata);
```

- Hủy session:

```
$this->session->unset_userdata('tên');
```

Nếu hủy tất cả các session hiện tại:

```
$this->session->sess_destroy();
```

* Sử dụng flashdata:

- Trong CI hỗ trợ "flashdata", rất hữu ích trong việc tạo ra các câu thông báo, hoặc thông tin trạng thái:

- Tạo flashdata:

```
$this->session->set_flashdata('name', 'value');
```

- Đọc flashdata:

```
$this->session->flashdata('name');
```

8. PHÂN TRANG (PAGINATION)

```
$this->load->library('pagination');
$config = array (
  'base_url' => base_url('product/'),
  'total_rows' => 20,
  'per_page' => 3,
  'next_link' => 'Next',
```

```
  'prev_link' => 'Prev',
);
$this->pagination->create_links();
```

9. GIỎ HÀNG (SHOPPING CART)

Tải thư viện:

```
$this->load->library('cart');
```

Thông tin lưu vào giỏ hàng:

```
$cart = array(
        'id'    => 8,
        'qty'   => 1,
        'price' => 50000,
        'name'  => 'san pham 01'
    );
```

- Thêm vào giỏ hàng:

```
public function add(){
        $cart = array(
                'id' => 8,
                'qty' => 1,
                'price' => 12000,
                'name' => 'san pham 01'
        )
        $this->cart->insert($cart);
}
```

- Lấy dữ liệu từ giỏ hàng:

```
$cart = $this->cart->contents();
```

Ngoài các thông tin chúng ta lưu vào giỏ hàng, nó còn tự động sinh ra 2 biến, rowid và subtotal. Biến rowid là một mã ngẫu nhiên, biến subtotal được tự động tính bằng qty*price.

- Cập nhật giỏ hàng:

```
public function update(){
  $carts = $this->cart->contents();
  $id = 8;
  foreach($carts as $key => $value){
```

```
            if($value['id'] == $id){
                $data = array(
                    'rowid' => $key,
                    'qty' => 3
                );
                $this->cart->update($data);
                break;
            }
        }
    }
```

- Xóa cả giỏ hàng:

```
$this->cart->destroy();
```

- Nếu muốn xóa từng sản phẩm trong giỏ hàng thì chỉ cần cập nhật lại số lượng sản phẩm có trong giỏ hàng = 0 (qty = 0)

Ví dụ xóa sản phẩm có id = 8:

```
public function del(){
    $id = 8;
    $carts = $this->cart->contents();
    foreach($carts as $key => $value){
        if ($value['id'] == $id){
            $data = array(
                'rowid' => $key,
                'qty' => 0,
            );
            $this->cart->update($data);
            break;
        }
    }
}
```

10. UPLOAD FILE

```
$config['upload_path'] = './uploads/';
$config['allowed_types'] = 'gif|jpg|png';
$config['max_size'] = '100';
$config['max_width']  = '1024';
$config['max_height']  = '768';
$this->load->library('upload', $config); //tải thư viện
$this->upload->do_upload('filename'); //thực hiện upload
```

(filename chính là tên thẻ input)

```
$this->upload->display_errors(); //hiển thị lỗi trong quá trình upload
$this->upload->data(); //lưu 1 mảng dữ liệu chứa thông tin của file upload
```

thành công

(lưu ý nhớ thêm thuộc tính enctype="multipart/form-data" vào thẻ form trong view)

Ví dụ trong controller tạo 1 file upload như sau:

```
<?php
class Upload extends CI_Controller
{
  function index()
  {
     if($this->input->post('submit'))
     {
        $config = array();
        $config['upload_path']   = './public/upload/user';
        $config['allowed_types'] = 'jpg|png|gif';
        $config['max_size']      = '500';
        $config['max_width']     = '1028';
        $config['max_height']    = '1028';
        $this->load->library('upload', $config);

        if($this->upload->do_upload('image'))
        {
           $data = $this->upload->data();
        }
```

```
            else
            {
                echo $this->upload->display_errors();
             }

        }
        $this->load->view('upload/index', $this->data);
    }
}
```

Các phần tử của data: file_name, file_type, file_path, raw_name, orig_name, file_ext, file_size, is_image, image_width, image_height, image_type, image_size_str

11. UPLOAD NHIỀU FILE

Trong thẻ input file, thuộc tính name thêm ký hiệu [] và thêm thuộc tính multiple.

Ví dụ ở view:

```
<form method="post" action="" enctype="multipart/form-data">
            <label>Ảnh minh họa:</label>
            <input type="file"  id="image" name="image">
            <br />
           <label>Ảnh kèm theo:</label><input type="file"  id="image_list"
          name="image_list[]" multiple>
            <br />
            <input type="submit" class="button" value="Upload"
name='submit' />
 </form>
```

Ở controller:

```
<?php
class Upload extends MY_Controller
{
  function index()
  {
     if($this->input->post('submit'))
     {
```

```
            //Khai bao bien cau hinh upload
            $config = array();
            //thuc mục chứa file
            $config['upload_path']   = './public/upload/user/';
            //Định dạng file được phép tải
            $config['allowed_types'] = 'jpg|png|gif';
            //Dung lượng tối đa
            $config['max_size']      = '500';
            //Chiều rộng tối đa
            $config['max_width']     = '1028';
            //Chiều cao tối đa
            $config['max_height']    = '1028';
            //load thư viện upload
            //bien chua cac ten file upload
           $name_array = array();
           //lưu biến môi trường khi thực hiện upload
           $file  = $_FILES['image_list'];
           $count = count($file['name']);//lấy tổng số file được upload
           for($i=0; $i<=$count-1; $i++) {

                $_FILES['userfile']['name']     = $file['name'][$i];  //khai báo tên của
file thứ i
                $_FILES['userfile']['type']     = $file['type'][$i]; //khai báo kiểu của
file thứ i
                $_FILES['userfile']['tmp_name'] = $file['tmp_name'][$i]; //khai báo
đường dẫn tạm của file thứ i
                $_FILES['userfile']['error']    = $file['error'][$i]; //khai báo lỗi của file
thứ i
                $_FILES['userfile']['size']     = $file['size'][$i]; //khai báo kích cỡ của
file thứ i
                //load thư viện upload và cấu hình
                $this->load->library('upload', $config);
                //thực hiện upload từng file
                if($this->upload->do_upload())
                {
```

```
                //nếu upload thành công thì lưu toàn bộ dữ liệu
                $data = $this->upload->data();
                //in cấu trúc dữ liệu của các file
                echo "<pre>";
                print_r($data);
                echo "</pre>";
            }
        }

    }
    // Hien thi view
    $this->load->view('site/upload/index', $this->data);
  }
}
```

+ Biến $_FILES['image_list'] chính là biến môi trường chứa thông tin của file upload lên server.

+ Hàm count là hàm lấy tổng số phần từ trong mảng,và nó có chức năng lấy tổng số file tải lên,lúc này chúng ta sẽ lặp toàn bộ các file tải lên thông qua vòng lặp for.

+ Biến userfile mặc định trong CodeIgniter phải khai báo lên các bạn giữ nguyên tên biến này nhé.

– Như vậy qua mỗi vòng lặp chúng ta đã thực hiện upload 1 file lên.Mỗi lần upload thành công các bạn có thể lấy được tên file ảnh tải lên và lưu vào trong CSDL.

12. GỬI EMAIL

- Để sử dụng đc thư viện email trc tiên chúng ta cần load thư viện đó ra:

```
$this->load->library('email');
```

- Cấu hình gửi mail trong codeigniter bằng mảng các giá trị tùy chọn thông qua phương thức initialize:

```
$config['protocol'] = 'mail';
$config['charset'] = 'utf-8';
$config['mailtype'] = 'html';
$config['wordwrap'] = TRUE;
$this->email->initialize($config);
```

- Nếu sử dụng phương thức smtp thì cấu hình thêm:

```
$config['protocol'] =  'smtp';
$config['smtp_host'] = 'ssl://smtp.googlemail.com';
$config['smtp_user'] = 'email@gmail.com';
$config['smtp_pass'] = '******';
$config['smtp_port'] = '465';
```

- Các phương thức chính:

```
+ $this->email->from('linhbk72@gmail.com', 'ngoc linh');
+ $this->email->to('nglinh223@gmail.com, website200k@gmail.com');
+ $this->email->subject('tieu de');
+ $this->email->message('noi dung');
+ $this->email->attach('/path/file.zip');
+ $this->email->send(); //lệnh gửi
+ $this->email->clear(); //khởi tạo lại cấu hình, sử dụng trong vòng lặp gửi nhiều mail.
```

- Muốn in lỗi dùng:

```
+ $this->email->print_debugger();
```

13. ĐĂNG KÝ THÀNH VIÊN

** Tạo view register:*

```
<form method="post" action="">
    <label for="param_email" class="form-label">Email:<span class="req">*</span></label>
    <input type="text" class="input" id="email" name="email" value="<?php echo set_value('email')?>">
    <div class="error" id="email_error"><?php echo form_error('email')?></div>

    <label for="param_password" class="form-label">Mật khẩu:<span class="req">*</span></label>
    <input type="password" class="input" id="password" name="password">
    <div class="error" id="password_error"><?php echo form_error('password')?></div>
     <label for="param_re_password" class="form-label">Gõ lại mật khẩu:<span class="req">*</span></label>
```

```
            <input type="password" class="input" id="re_password"
name="re_password">
            <div class="error" id="re_password_error"><?php echo
form_error('re_password')?></div>
          <label for="param_name" class="form-label">Họ và tên:<span
class="req">*</span></label>
            <input type="text" class="input" id="name" name="name" value="<?php
echo set_value('name')?>">
            <div class="error" id="name_error"><?php echo
form_error('name')?></div>
          <label for="param_phone" class="form-label">Số điện thoại:<span
class="req">*</span></label>
            <input type="text" class="input" id="phone" name="phone" value="<?php
echo set_value('phone')?>">
            <div class="error" id="phone_error"><?php echo
form_error('phone')?></div>
          <label for="param_address" class="form-label">Địa chỉ:<span
class="req">*</span></label>
            <textarea class="input" id="address" name="address"><?php echo
set_value('address')?></textarea>
            <div class="error" id="address_error"><?php echo
form_error('address')?></div>
          <input type="submit" class="button" value="Đăng ký" name='submit'>
      </form>
```

- set_value(''): lưu lại giá trị cũ khi nhập lại.
- form_error(''): hiển thị lỗi.

** Tạo model:*

```
<?php
class User_model extends MY_Model
{
   //ten bang du lieu
   public $table = 'user';
}
```

(đọc bài main model để biết MY_Model - mục số 17).

** Tạo controller:*

```
<?php if (!defined('BASEPATH')) exit('No direct script access allowed');
```

```
class User extends MY_Controller
{
    public function __construct()
    {
      parent::__construct();
      //load model user
      $this->load->model('user_model');
    }
    /*
    * Phuong thuc dang ky thanh vien
    */
    public function register()
    {
      //load thu vien validation
      $this->load->library('form_validation');
      $this->load->helper('form');

      //tao cac tap luat
      $this->form_validation->set_rules('email', 'Email',
'required|valid_email|xss_clean');
      $this->form_validation->set_rules('name', 'Họ và tên',
'required|min_length[8]|xss_clean');
      $this->form_validation->set_rules('phone', 'Số điện thoại',
'required|min_length[8]|numeric|xss_clean');
      $this->form_validation->set_rules('password', 'Mật khẩu',
'required|min_length[6]|numeric|xss_clean');
      $this->form_validation->set_rules('re_password', 'Nhập lại mật khẩu',
'required|matches[password]|xss_clean');
      $this->form_validation->set_rules('address', 'Địa chỉ',
'required|xss_clean');
      //chạy phương thức kiểm tra dữ liệu
      if($this->form_validation->run())
      {
          //code here
      }
```

```
            //thực hiện load view
            $this->load->view('site/user/register', $this->data);
        }
    }
```

(đọc bài main controller để biết MY_Controller - mục 16).

- Ở đây sử dụng email để đăng ký, nên 1 email chỉ được đăng ký 1 tài khoản. Như vậy cần kiểm tra xem email đã có trong csdl chưa. Chúng ta sử dụng hàm callback để kiểm tra.

Đoạn code kiểm tra email sửa lại như sau:

```
$this->form_validation->set_rules('email', 'Email', 'required|valid_email|xss_clean|callback_check_email');
```

- Hàm check email:

```
function check_email()
{
    $email = $this->input->post('email');
    $where = array();
    $where['email'] = $email;
    //kiểm tra điều kiện email có tồn tại trong csdl hay không
    if($this->user_model->check_exists($where))
    {
        //trả về thông báo lỗi nếu đã tồn tại email này
     $this->form_validation->set_message(__FUNCTION__, 'Email đã sử dụng');
        return FALSE;
    }
    return TRUE;
}
```

- Khi đã nhập đầy đủ thông tin và hợp lệ, lúc này sẽ thêm thành viên vào csdl:

```
if($this->form_validation->run())
        {
            //du lieu them vao bang thanh vien
            $data = array(
                'name'     => $this->input->post('name'),
                'email'    => $this->input->post('email'),
```

```
                'password' => md5($this->input->post('password')),
                'phone'    => $this->input->post('phone'),
                'address'  => $this->input->post('address')
            );
            //them thanh vien vao trong csdl
            if($this->user_model->create($data))
            {
                $this->session->set_flashdata('flash_message', 'Dang ky thanh vien thanh cong');
                redirect();//chuyen toi trang chu
            }
        }
```

14. ĐĂNG NHẬP – ĐĂNG XUẤT

** Tạo view login:*

```
<form class="t-form form_action" method="post" action="" name="login" >
<div class="form-row">
<label for="param_email" class="form-label">Email: <span class="req"> * </span> </label>
<div class="form-item">
<input type="text" class="input" id="login_email" name="email" value="<?php echo set_value('email')?>">
<div class="clear"></div>
<div class="error" id="login_email_error"><?php echo form_error('email')?></div>
</div>
<div class="clear"></div>
</div>
<div class="form-row">
<label for="param_password" class="form-label">Mật khẩu:<span class="req"> * </span></label>
<div class="form-item">
<input type="password" class="input" id="login_password" name="password">
<div class="clear"></div>
```

```
<div class="error" id="login_password_error"><?php echo
form_error('password')?></div>
</div>
<div class="clear"></div>
</div>
<div class="form-row">
<label class="form-label"> </label>
<div class="form-item">
<div class="error" id="login_error"><?php echo form_error('login')?></div>
<input type="submit" class="button"  name='login' value="Đăng nhập" >
</div>
</div>
</form>
```

** Tạo model (user_model):*

```
<?php
class User_model extends MY_Model
{
   //ten bang du lieu
   public $table = 'user';
    //ten bang du lieu
   public function check_login($email, $password)
   {
      //them dieu kien kiem tra email va password
      $where = array('email' => $email, 'password' => $password);
      $this->db->where($where);
      //thuc hien cau truy van
      $query = $this->db->get($this->table);
      if($query->num_rows() > 0)
      {
         return true;
      }
      return false;
   }
```

```
    /*
     * lay thong tin thanh vien
     */
    public function get_user_info($where = array())
    {
        //tao dieu kien cho cau truy van
        $this->db->where($where);
        $result = $this->db->get('user');
        return $result->row();
    }
}
```

**** Tạo controller (User):***

```
<?php if (!defined('BASEPATH')) exit('No direct script access allowed');
class User extends MY_Controller
{
    public function __construct()
    {
      parent::__construct();
      //load model user
      $this->load->model('user_model');
    }
 /*
    * Kiem tra dang nhap
    */
    public function check_login()
    {
        //lay du lieu tu form
        $email    = $this->input->post('email');
        $password = $this->input->post('password');
        $password = md5($password);
        $where = array('email' => $email, 'password' => $password);
        if(!$this->user_model->check_exists($where))
        {
```

```
                //trả về thông báo lỗi nếu đã tồn tại email này
                $this->form_validation->set_message(__FUNCTION__, 'Đăng nhập không thành công');
                return FALSE;
            }
            return true;
        }
    /*
        * Phuong thuc dang nhap
        */
        public function login()
        {
          //load thu vien validation
          $this->load->library('form_validation');
          $this->load->helper('form');

           //tao cac tap luat
          $this->form_validation->set_rules('email', 'Email', 'required|valid_email|xss_clean');
          $this->form_validation->set_rules('password', 'Mật khẩu', 'required|xss_clean');
          $this->form_validation->set_rules('login', 'Đăng nhập', 'callback_check_login');

          if($this->form_validation->run())
          {
              //lay du lieu tu form post sang
              $email    = $this->input->post('email');
              $password = $this->input->post('password');
              $password = md5($password);
              $where = array('email' => $email, 'password' => $password);
              //lay thong tin thanh vien
              $user = $this->user_model->get_info_rule($where);
```

```
                //luu thong tin thanh vien vao session
                $this->session->set_userdata('login', $user);
                //tạo thông báo
                $this->session->set_flashdata('flash_message', 'Đăng nhập thành
công');
                redirect();//chuyen toi trang chu
            }

            //gui du lieu sang view
            $this->data['temp'] = 'site/user/login';
            $this->load->view('site/layout', $this->data);
        }
    }
```

** Chức năng đăng xuất:*

- Trong controller xây dựng hàm logout:

```
public function logout()
    {
        if($this->_user_is_login())
        {
            //neu thanh vien da dang nhap thi xoa session login
            $this->session->unset_userdata('login');
        }
        $this->session->set_flashdata('flash_message', 'Đăng xuất thành công');
        redirect();
    }
```

15. HELPER LANGUAGE

- Để tạo 1 ngôn ngữ trong CI cần truy cập vào thư mục application/language, tạo thư mục ngôn ngữ.

Ví dụ vietnamese và english. Trong mỗi thư mục tạo 1 file user_lang.php.

+ File language/english/user_lang.php với nội dung:

```
<?php
$lang['full_name']    = 'Full name';
$lang['email']        = 'Email';
```

```
$lang['phone']        = 'Phone';
$lang['password']     = 'Password';
$lang['address']      = 'address';
?>
```

+ File language/vietnamese/user_lang.php với nội dung:

```
<?php
$lang['full_name']  = 'Họ tên';
$lang['email']      = 'Địa chỉ email';
$lang['phone']      = 'Số điện thoại';
$lang['password']   = 'Mật khẩu';
$lang['address']    = 'Địa chỉ';
?>
```

- Sử dụng language helper:

```
$this->load->helper('language');
```

- Tải file ngôn ngữ cần dùng:

```
$this->lang->load('ten_file' , 'ten_thu_muc');
```

Nếu bỏ trống tên thư mục thì lúc đó nó sẽ load mặc định thư mục trong file cấu hình, được cài đặt trong file config.php:

```
$config['language'] = 'english';
```

Ví dụ:

```
$this->lang->load('user', 'vietnamese');
```

- Gọi tới tham số trong file ngôn ngữ:

```
lang($tham_so);
```

hoặc `$this->lang->line($tham_so);`

Ví dụ: (controller home)

```
<?php if (!defined('BASEPATH')) exit('No direct script access allowed');
class Home extends CI_Controller
{
  public function index()
  {
    //load helper language
    $this->load->helper('language');
    //load file ngon ngu
```

```
        $this->lang->load('user');
        echo lang('full_name').'<br />';
        echo lang('email').'<br />';
        echo lang('phone').'<br />';
        echo lang('password').'<br />';
        echo lang('address').'<br />';
    }
}
```

- Khi muốn thay đổi file ngôn ngữ, cách 1 là thay đổi thư mục trong config.

Cách thứ 2 hay dùng là:

```
$this->config->set_item('language', 'vietnamese');
$this->lang->load('user');
```

Cách thứ 3 là sử dụng tham số thứ 2 khi load file tham số:

```
$this->lang->load('user', 'vietnamese');
```

16. XÂY DỰNG MAIN CONTROLLER

Trong những yêu cầu thực tế, có thể cần xây dựng 1 controller nhằm xử lý các dữ liệu chung cho các controller khác. Ví dụ truy cập các trang admin, cần phải kiểm tra admin đã đăng nhập hay chưa (ở tất cả các controller). Nếu có 1 main controller thì chỉ cần kiểm tra ở main controller.

Trong CI hỗ trợ chúng ta kế thừa hoặc ghi đè trong thư mục application/core. Đây là nơi có thể viết overwrite toàn bộ core codeigniter.

Tất cả các file sẽ có tiền tố là "MY_". Mặc định trong ci các file viết trong core có tiền tố là "MY_". Có thể sửa trong file config:

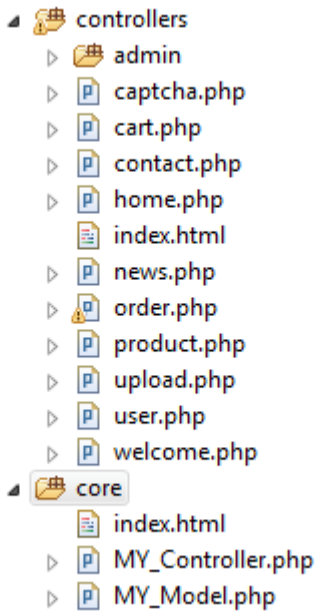
```
$config['subclass_prefix'] = 'MY_';
```

Ban đầu tạo file main controller có tên là MY_Controller.php với nội dung sau:

```
<?php if ( ! defined('BASEPATH')) exit('No direct script access allowed');
class MY_Controller extends CI_Controller {
    // Bien luu thong tin gui den view
    var $data = array();
}
```

Với đoạn code trên, ta đang tạo 1 class kế thừa từ CI_Controller, và biến $data để chứa các tham số gửi sang view. Như vậy tất cả các controller khác có thể kế thừa từ MY_Controller này rồi.

Giả sử có cấu trúc thư mục như sau:



Trong MY_Controller xây dựng hàm khởi tạo như sau:

```
/**
 * Ham khoi dong
 */
function __construct()
{
    //kế thừa từ CI_Controller
    parent::__construct();

    // Xu ly controller
    $controller = $this->uri->segment(1);
    $controller = strtolower($controller);
    //kiểm tra xem trang hiện tại đang truy cập
    switch ($controller)
    {
        //Nếu đang truy cập vào trang Admin
        case 'admin':
        {
            //load các file sử dụng nhiều cho trang admin
            //kiểm tra admin đăng nhập hay chưa
            //kiểm tra quyền của admin
```

```
            //....
            break;
        }

        //Trang chủ
        default:
        {
            //load các file sử dụng nhiều cho trang chủ
            //Xử lý ngôn ngữ
            //Xử lý tiền tệ
            //....
            break;
        }
    }
}
```

Cấu trúc switch case kiểm tra xem trang đang truy cập có phải thư mục admin hay không. Nếu đang là trang admin sẽ viết code xử lý như kiểm tra đăng nhập, kiểm tra phân quyền... và tải các model, helper, library hay sử dụng trong admin. Nếu trang truy cập là trang chủ, có thể viết code xử lý chung cho các trang người dùng như ngôn ngữ, tiền tệ... và tải các model, helper, library hay sử dụng.

17. XÂY DỰNG MAIN MODEL

Các hành động thêm, sửa , xóa, lấy dữ liệu... từ các bảng là giống nhau, chỉ khác nhau ở tên bảng và các cột. Vậy có thể xây dựng main model để thực hiện các hành động trên.

Trong thư mục application/core, tạo 1 file có tên là MY_Model, với nội dung như sau:

```
<?php if ( ! defined('BASEPATH')) exit('No direct script access allowed');
class MY_Model extends CI_Model {
    // Ten table
    var $table = '';
}
```

Xây dựng phương thức thêm mới dữ liệu:

```
/**
* Them row moi
```

```
*/
function create($data)
{
   if($this->db->insert($this->table, $data))//thêm dữ liệu
   {
     return TRUE;
   }else{
     return FALSE;
   }
}
```

Xây dựng phương thức cập nhật dữ liệu:

```
/**
* Cap nhat row tu id
*/
function update($id, $data)
{
   if (!$id)
   {
      return FALSE;
   }
   $where = array();
   $where['id'] = $id;//điều kiện khóa chính bằng $id truyền vào
      return $this->update_rule($where, $data);
}

/**
* Cap nhat row tu dieu kien
* $where: điều kiện
*/
function update_rule($where, $data)
{
   if (!$where)
   {
```

```
        return FALSE;
    }
    $this->db->where($where);//thêm điều kiện
    if($this->db->update($this->table, $data))//cập nhật dữ liệu
    {
        return TRUE;
    }
    return FALSE;
}
```

Xây dựng phương thức xóa dữ liệu:

```
/**
*Xoa row tu id
*/
function delete($id)
{
    if (!$id)
            {
        return FALSE;
    }
    if(is_numeric($id))//nếu $id là số
    {
        $where = array('id' => $id);
    }else
    {
                //id nằm trong chuoi các id truyền vào
        $where =  "id IN (".$id.") ";
    }
    return $this->del_rule($where);
}

/**
* Xoa row tu dieu kien
*/
```

```
function del_rule($where)
{
    if (!$where)
    {
        return FALSE;
    }
    $this->db->where($where);//thêm điều kiện
    if($this->db->delete($this->table))//thực hiện xóa
    {
        return TRUE;
    }
    return FALSE;
}
```

Xây dựng phương thức lấy thông tin 1 dữ liệu:

```
/**
* Lay thong tin cua row tu id
* $id: Khóa chính muốn lấy thông tin
*/
function get_info($id)
{
    if (!$id)
    {
        return FALSE;
    }
    $where = array();
    $where['id'] = $id;
    return $this->get_info_rule($where);
}

/**
 * Lay thong tin cua row tu dieu kien
 * $where: Mảng điều kiện
 */
```

```
function get_info_rule($where = array())
{
   $this->db->where($where);
   $query = $this->db->get($this->table);
   if ($query->num_rows())
   {
      return $query->row();
   }
   return FALSE;
}
```

Xây dựng phương thức lấy danh sách dữ liệu:

```
/**
* Lay danh sach
*/
function get_list($input = array())
{
        //gắn các tùy chọn nếu có
   $this->get_list_set_input($input);
   //thuc hien truy van du lieu
   $query = $this->db->get($this->table);
        //tra ve du lieu
   return $query->result();
}

/**
* Gan cac thuoc tinh trong input khi lay danh sach
*/
protected function get_list_set_input($input)
{
    // Select
 if (isset($input['select']))
 {
     $this->db->select($input['select']);
```

```
        }
            // Thêm điều kiện cho câu truy vấn truyền qua biến $input['where']

            if ((isset($input['where'])) && $input['where'])
            {
                $this->db->where($input['where']);
            }
                    // Thêm sắp xếp dữ liệu thông qua biến $input['order'] (ví dụ
$input['order'] = array('id','DESC'))
            if (isset($input['order'][0]) && isset($input['order'][1]))
            {
                $this->db->order_by($input['order'][0], $input['order'][1]);
            }
            else
            {
                //mặc định sẽ sắp xếp theo id giảm dần
                $this->db->order_by('id', 'desc');
            }

            // Thêm điều kiện limit cho câu truy vấn thông qua biến $input['limit'] (ví
dụ $input['limit'] = array('10' ,'0'))
            if (isset($input['limit'][0]) && isset($input['limit'][1]))
            {
                $this->db->limit($input['limit'][0], $input['limit'][1]);
            }
        }
```

Xây dựng phương thức lấy tổng số dữ liệu:

```
        /**
        * Lay tong so
        */
        function get_total($input = array())
        {
                //gắn các tùy chọn nếu có
            $this->get_list_set_input($input);
```

```
    //thuc hien truy van du lieu
    $query = $this->db->get($this->table);
    //tra ve du lieu
    return $query->num_rows();
}
```

Ví dụ:

Model product_model.php:

```
<?php
class Product_model extends MY_Model
{
    //ten bang du lieu
    public $table = 'product';
}
```

Controller product.php:

```
<?php if (!defined('BASEPATH')) exit('No direct script access allowed');

class Product extends MY_Controller
{
  /*
  * Ham khi khoi tao
  */
  public function __construct()
  {
     parent::__construct();
     $this->load->model('product_model');//tải file model
  }
   /*
   * Trang dang danh sách sản phẩm
   */
  public function index()
  {
        //gan diều kiện cho câu truy vấn
    $input = array();
```

```
        $input['limit'] = array('5', 0);
            $products = $this->product_model->get_list($input);
            print_r($products);
        }
        /**
         * Them moi
        */
        function add()
        {
            //luu toan bo thong tin san pham vao csdl
            $data = array();
            $data['name']   = 'Nokia 520';
            $data['price']  = '3000000';
            if($this->product_model->create($data))
            {
                //gui thong bao thanh cong
            }
        }
    }
```

18. UPLOAD FILE VỚI THƯ VIỆN FTP

Thư viện FTP trong ci cho phép upload, chỉnh sửa tên, di chuyển hoặc xóa file trên server 1 cách dễ dàng (trên server khác với server chứa website).

Trước tiên load thư viện:

```
$this->load->library('ftp');
```

1. Phương thức upload:

```
$this->load->library('ftp');
$config['hostname'] = 'ftp.example.com';
$config['username'] = 'your-username';
$config['password'] = 'your-password';
$config['debug']    = TRUE;//TRUE cho phép hiển thị lỗi,FALSE không cho phép
$this->ftp->connect($config);
$this->ftp->upload('path_serverA', 'path_serverB');
```

```
$this->ftp->close();
```

2. Phương thức dowload:

```
$this->ftp->download('/public_html/photo.jpg', '/local/path/to/photo.jpg');
```

3. Phương thức rename:

```
$this->ftp->rename('/public_html/images/photo.jpg', '/public_html/images/photo1.jpg');
```

4. Phương thức move:

```
$this->ftp->move('/public_html/images/photo.jpg', '/public_html/img/photo.jpg');
```

5. Phương thức delete file:

```
$this->ftp->delete_file('/public_html/images/photo.jpg');
```

6. Phương thức delete dir:

```
$this->ftp->delete_dir('/public_html/images/');
```

7. Phương thức list_files:

```
$list = $this->ftp->list_files('/public_html/images/');
print_r($list);
```

19. THƯ VIỆN ZIP FILE

- Để sử dụng thư viện cần load nó ra:

```
$this->load->library('zip');
```

- Thêm dữ liệu vào file zip:

```
$name = 'file.txt';
$data = 'dữ liệu file zip';
$this->zip->add_data($name, $data);
```

- Có thể sử dụng mảng:

```
$data = array(
        'data1.txt' => 'dữ liệu trong file zip1';
        'data2.txt' => 'dữ liệu trong file zip2';
);
$this->zip->add_data($data);
```

- Nén và tải file đã tồn tại:

```
$path = 'file.txt';
$this->zip->read_file($path);
$this->zip->download('my_file.zip');
```

20. THƯ VIỆN USER AGENT

User Agent giúp xác định thông tin trình duyệt mà người dùng đang sử dụng. Khi muốn chuyển giao diện theo thiết bị, có thể sử dụng thư viện này để kiểm tra xem người dùng đang sử dụng thiết bị gì (mobile hay desktop).

- Trước tiên load thư viện:

```
$this->load->library('user_agent');
```

- Các cấu hình được định sẵn trong file application/user_agent.php, có thể thêm các cấu hình khác vào.

- Kiểm tra trình duyệt:

```
$this->agent->is_browser('Chrome');
```

- Kiểm tra mobile:

```
$this->agent->is_mobile('iphone');
```

Ví dụ sử dụng:

```
if ($this->agent->is_mobile('iphone')){
        $this->load->view('iphone/home');
} elseif ($this->agent->is_mobile()){
        $this->load->view('mobile/home');
} else {
        $this->load->view('web/home');
}
```

- Kiểm tra robot:

```
$this->agent->is_robot('string');
```

21. CACHING TRONG CODEIGNITER

- Caching website là việc lưu trữ bản sao của những dữ liệu web sao cho gần với người dùng, cả về mặt chức năng trong web client hoặc những web caching servers riêng biệt.

- Ưu điểm của web caching:

+ Giảm tải băng thông.

+ Hạn chế việc truy vấn trực tiếp vào database.

+ Load nhanh hơn.

<u>* Caching file:</u>

- Hệ thống sẽ sinh ra file lưu trữ mã html của web với lần truy cập đầu tiên và những lần sau hệ thống sẽ load trực tiếp file này.

File này được lưu trữ trong thư mục application/cache.

- Sử dụng hàm cache nội dung:

```
$this->output->cache($n)
```

Trong đó $n là thời gian cache tính bằng phút (cứ n phút thì file cache sẽ được làm mới).

Ví dụ:

```
function index()
   {
        $this->load->view('index');
        $this->output->cache(20) ;
   }
```

* Caching driver:

- Lấy dữ liệu bằng phương thức get():

```
$product_data = $this->cache->get('product');
```

- Lưu dữ liệu bằng phương thức save():

```
$this->cache->save('product', $product_data, 600);
```

//600 là thời gian lưu cache đơn vị tính là giây.

- Xóa dữ liệu bằng phương thức delete():

```
$this->cache->delete('product');
```

Ví dụ:

```
function get_list_product()
  {
    if ( ! $products_data = $this->cache->get('product'))
    {
       $this->load->model('product_model');
       $products_data = $this->product_model->get_list();
       $this->cache->save('product', $products_data, 600);
    }
    return $products_data;
  }
```

==========//==========