Phân trang trong CodeIgniter: Hướng dẫn Hoàn chỉnh

by [Sajal Soni](https://tutsplus.com/authors/sajal-soni)14 Aug 2017

Difficulty:BeginnerLength:LongLanguages:

[CodeIgniter](https://code.tutsplus.com/vi/categories/codeigniter)[CMS](https://code.tutsplus.com/vi/categories/cms)[PHP](https://code.tutsplus.com/vi/categories/php)[Web Development](https://code.tutsplus.com/vi/categories/web-development)

Vietnamese (Tiếng Việt) translation by [Dai Phong](https://tutsplus.com/authors/dai-phong) (you can also [view the original English article](https://code.tutsplus.com/tutorials/pagination-in-codeigniter-the-complete-guide--cms-29030?ec_unit=translation-info-language))

Lợi ích của việc sử dụng framework full stack cho ứng dụng web đó là bạn không phải lo lắng về các tác vụ thông thường như xử lý đầu vào, kiểm tra form và những thứ tương tự, vì framework đã cung cấp các wrapper cho các tính năng đó. Do đó, nó cho phép bạn tập trung vào logic của ứng dụng thay vì làm đi làm lại những công việc đó.

Hôm nay, chúng ta sẽ khám phá một thư viện quan trọng trong framework CodeIgniter - thư viện phân trang.

Để tôi khái quát các chủ đề mà chúng ta sẽ thảo luận trong bài viết này:

* Minh hoạ cơ bản về chức năng phân trang
* Khám phá các tùy chọn tùy biến
* Cấu hình phân trang

**Minh hoạ cơ bản về chức năng phân trang**

Trong phần này, chúng ta sẽ đi tìm hiểu một ví dụ minh hoạ việc sử dụng phân trang trong CodeIgniter. Đó là cách tốt nhất để hiểu mọi thứ hoạt động như thế nào.

Trong ví dụ của chúng ta, chúng ta sẽ xây dựng một danh sách người dùng khá đơn giản, trong đó chúng ta sẽ truy vấn các bản ghi từ bảng users trong CSDL MySQL. Để chạy thành công ví dụ này, hãy đảm bảo rằng bạn có các trường uid và uname trong bảng users của bạn.

Với thiết lập đó, chúng ta đã sẵn sàng để thực hiện.

Hãy tiếp tục và tạo một tập tin controller controllers/Paging.php với các nội dung sau.

|  |  |
| --- | --- |
| 001  002  003  004  005  006  007  008  009  010  011  012  013  014  015  016  017  018  019  020  021  022  023  024  025  026  027  028  029  030  031  032  033  034  035  036  037  038  039  040  041  042  043  044  045  046  047  048  049  050  051  052  053  054  055  056  057  058  059  060  061  062  063  064  065  066  067  068  069  070  071  072  073  074  075  076  077  078  079  080  081  082  083  084  085  086  087  088  089  090  091  092  093  094  095  096  097  098  099  100  101  102  103  104  105  106  107 | <?php  defined('BASEPATH') OR exit('No direct script access allowed');    class Paging extends CI\_Controller {      public function \_\_construct()    {          parent::\_\_construct();            // load Pagination library          $this->load->library('pagination');            // load URL helper          $this->load->helper('url');      }        public function index()      {          // load db and model          $this->load->database();          $this->load->model('Users');            // init params          $params = array();          $limit\_per\_page = 1;          $start\_index = ($this->uri->segment(3)) ? $this->uri->segment(3) : 0;          $total\_records = $this->Users->get\_total();            if ($total\_records > 0)          {              // get current page records              $params["results"] = $this->Users->get\_current\_page\_records($limit\_per\_page, $start\_index);                $config['base\_url'] = base\_url() . 'paging/index';              $config['total\_rows'] = $total\_records;              $config['per\_page'] = $limit\_per\_page;              $config["uri\_segment"] = 3;                $this->pagination->initialize($config);                // build paging links              $params["links"] = $this->pagination->create\_links();          }            $this->load->view('user\_listing', $params);      }        public function custom()      {          // load db and model          $this->load->database();          $this->load->model('Users');            // init params          $params = array();          $limit\_per\_page = 2;          $page = ($this->uri->segment(3)) ? ($this->uri->segment(3) - 1) : 0;          $total\_records = $this->Users->get\_total();            if ($total\_records > 0)          {              // get current page records              $params["results"] = $this->Users->get\_current\_page\_records($limit\_per\_page, $page\*$limit\_per\_page);                $config['base\_url'] = base\_url() . 'paging/custom';              $config['total\_rows'] = $total\_records;              $config['per\_page'] = $limit\_per\_page;              $config["uri\_segment"] = 3;                // custom paging configuration              $config['num\_links'] = 2;              $config['use\_page\_numbers'] = TRUE;              $config['reuse\_query\_string'] = TRUE;                $config['full\_tag\_open'] = '<div class="pagination">';              $config['full\_tag\_close'] = '</div>';                $config['first\_link'] = 'First Page';              $config['first\_tag\_open'] = '<span class="firstlink">';              $config['first\_tag\_close'] = '</span>';                $config['last\_link'] = 'Last Page';              $config['last\_tag\_open'] = '<span class="lastlink">';              $config['last\_tag\_close'] = '</span>';                $config['next\_link'] = 'Next Page';              $config['next\_tag\_open'] = '<span class="nextlink">';              $config['next\_tag\_close'] = '</span>';                $config['prev\_link'] = 'Prev Page';              $config['prev\_tag\_open'] = '<span class="prevlink">';              $config['prev\_tag\_close'] = '</span>';                $config['cur\_tag\_open'] = '<span class="curlink">';              $config['cur\_tag\_close'] = '</span>';                $config['num\_tag\_open'] = '<span class="numlink">';              $config['num\_tag\_close'] = '</span>';                $this->pagination->initialize($config);                // build paging links              $params["links"] = $this->pagination->create\_links();          }            $this->load->view('user\_listing', $params);      }  } |

Tiếp theo, chúng ta sẽ cần một tập tin model models/Users.php để lấy các bản ghi từ bảng users.

|  |  |
| --- | --- |
| 01  02  03  04  05  06  07  08  09  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28  29  30  31  32  33  34 | <?php  // models/Users.php  defined('BASEPATH') OR exit('No direct script access allowed');    class Users extends CI\_Model  {      function \_\_construct()      {        parent::\_\_construct();      }        public function get\_current\_page\_records($limit, $start)      {          $this->db->limit($limit, $start);          $query = $this->db->get("users");            if ($query->num\_rows() > 0)          {              foreach ($query->result() as $row)              {                  $data[] = $row;              }                return $data;          }            return false;      }        public function get\_total()      {          return $this->db->count\_all("users");      }  } |

Cuối cùng, hãy tạo một tập tin view views/user\_listing.php để hiển thị danh sách người dùng.

|  |  |
| --- | --- |
| 01  02  03  04  05  06  07  08  09  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28  29  30  31  32  33  34 | <!-- views/user\_listing.php -->  <html>      <head>          <title>Paging Example-User Listing</title>      </head>        <body>          <div class="container">              <h1 id='form\_head'>User Listing</h1>                <?php if (isset($results)) { ?>                  <table border="1" cellpadding="0" cellspacing="0">                      <tr>                          <th>ID</th>                          <th>NAME</th>                      </tr>                        <?php foreach ($results as $data) { ?>                          <tr>                              <td><?php echo $data->uid ?></td>                              <td><?php echo $data->uname ?></td>                          </tr>                      <?php } ?>                  </table>              <?php } else { ?>                  <div>No user(s) found.</div>              <?php } ?>                <?php if (isset($links)) { ?>                  <?php echo $links ?>              <?php } ?>          </div>      </body>  </html> |

Bây giờ, hãy truy cập vào trang tùy biến của chúng ta tại [http://your-code-igniter-site/paging/index](https://your-code-igniter-site/paging/index) và bạn sẽ thấy danh sách người dùng được phân trang! Như vậy là chúng ta thực hiện xong! Đừng lo lắng, tôi sẽ không bỏ mặc bạn sớm như vậy đâu, vì chúng ta sẽ bắt đầu phân tích từng phần của đoạn code ngay bây giờ.

Chúng ta sẽ bắt đầu với tập tin model models/Users.php vì đó là những thứ sẽ được gọi từ các phương thức controller của chúng ta. Có hai phương thức quan trọng, get\_current\_page\_records và get\_total, các cài đặt model đó của chúng ta nhằm xây dựng các liên kết phân trang.

Hãy tìm hiểu phương thức get\_total. Nó được sử dụng để đếm số bản ghi trong bảng users.

|  |  |
| --- | --- |
| 1  2  3  4 | public function get\_total()  {      return $this->db->count\_all("users");  } |

Tiếp theo, phương thức get\_current\_page\_records.

|  |  |
| --- | --- |
| 01  02  03  04  05  06  07  08  09  10  11  12  13  14  15  16  17 | public function get\_current\_page\_records($limit, $start)  {      $this->db->limit($limit, $start);      $query = $this->db->get("users");        if ($query->num\_rows() > 0)      {          foreach ($query->result() as $row)          {              $data[] = $row;          }            return $data;      }        return false;  } |

Có hai đối số quan trọng mà bạn nên lưu ý trong phương thức get\_current\_page\_records. Đối số đầu tiên, $limit, được sử dụng để chỉ định số bản ghi sẽ được trả về trong quá trình thực thi truy vấn. Và đối số thứ hai, $start, đóng vai trò như là chỉ số bắt đầu của bản ghi.

Như bạn thấy đấy, với các giá trị của $start và $limit chúng ta có thể lấy về các bản ghi theo trang. Đó là bản chất của phân trang, và như vậy chúng ta đã cài đặt phương thức quan trọng nhất của bài viết này!

Như vậy, đó là model của chúng ta - đơn giản!

Tiếp tục, chúng ta hãy tập trung vào tập tin controller. Hãy lấy code của phương thức constructor.

|  |  |
| --- | --- |
| 01  02  03  04  05  06  07  08  09  10 | public function \_\_construct()  {      parent::\_\_construct();        // load Pagination library      $this->load->library('pagination');        // load URL helper      $this->load->helper('url');  } |

Để sử dụng phân trang trong CodeIgniter, điều đầu tiên bạn cần làm là nạp thư viện phân trang (pagination). Và chúng ta có thể làm điều đó bằng cách sử dụng $this->load->library('pagination').

Đồng thời chúng ta nạp URL helper để chúng ta có thể sử dụng các hàm trợ giúp toàn cục được cung cấp bởi helper đó.

Bây giờ, chúng ta đã sẵn sàng để đi sâu vào trái tim của controller - phương thức index.

|  |  |
| --- | --- |
| 01  02  03  04  05  06  07  08  09  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28  29  30 | public function index()  {          // load db and model          $this->load->database();          $this->load->model('Users');            // init params          $params = array();          $limit\_per\_page = 1;          $start\_index = ($this->uri->segment(3)) ? $this->uri->segment(3) : 0;          $total\_records = $this->Users->get\_total();            if ($total\_records > 0)          {              // get current page records              $params["results"] = $this->Users->get\_current\_page\_records($limit\_per\_page, $start\_index);                $config['base\_url'] = base\_url() . 'paging/index';              $config['total\_rows'] = $total\_records;              $config['per\_page'] = $limit\_per\_page;              $config["uri\_segment"] = 3;                $this->pagination->initialize($config);                // build paging links              $params["links"] = $this->pagination->create\_links();          }            $this->load->view('user\_listing', $params);      } |

Để bắt đầu, chúng ta đảm bảo rằng cơ sở dữ liệu được nạp đúng cách. Sau đó, chúng ta nạp model Users để chúng ta có thể sử dụng các phương thức của model đó.

|  |  |
| --- | --- |
| 1  2 | $this->load->database();  $this->load->model('Users'); |

Tiếp theo, chúng ta khởi tạo một vài biến quan trọng.

|  |  |
| --- | --- |
| 1  2  3  4  5 | // init params  $params = array();  $limit\_per\_page = 1;  $start\_index = ($this->uri->segment(3)) ? $this->uri->segment(3) : 0;  $total\_records = $this->Users->get\_total(); |

Biến $limit\_per\_page xác định giới hạn trên mỗi trang. Tất nhiên, bạn có thể thiết lập nó tuỳ ý bạn; nó được thiết lập thành 1 tại thời điểm này nhằm mục đích minh hoạ.

Biến $start\_index giữ chỉ số bắt đầu của bản ghi trong MySQL. Khi CodeIgniter xây dựng các liên kết phân trang, mặc định nó sẽ nối thêm chỉ mục bắt đầu của trang như là phân đoạn thứ ba trong URL. Bạn có thể thay đổi hành vi mặc định này, nhưng đó là điều mà chúng ta sẽ để dành cho phần cuối cùng của bài viết này, nơi chúng ta sẽ thảo luận các tuỳ chọn tùy biến.

Cuối cùng, chúng ta gọi phương thức get\_total của model Users để lấy tổng số bản ghi của bảng users và nó được gán cho biến $total\_records.

Tiếp theo, chúng ta lấy các bản ghi của trang hiện tại bằng phương thức get\_current\_page\_records.

|  |  |
| --- | --- |
| 1  2 | // get current page records  $params["results"] = $this->Users->get\_current\_page\_records($limit\_per\_page, $start\_index); |

Trước khi chúng ta có thể thật sự xây dựng các liên kết phân trang, chúng ta cần phải khởi tạo cấu hình phân trang tối thiểu bằng phương thức initialize của thư viện phân trang.

|  |  |
| --- | --- |
| 1  2  3  4  5  6 | $config['base\_url'] = base\_url() . 'paging/index';  $config['total\_rows'] = $total\_records;  $config['per\_page'] = $limit\_per\_page;  $config["uri\_segment"] = 3;    $this->pagination->initialize($config); |

Và đó là bộ tham số tối thiểu để xây dựng các liên kết phân trang.

* **base\_url**: URL sẽ được sử dụng khi xây dựng các liên kết phân trang
* **total\_rows**: Tổng số bản ghi
* **per\_page**: Số bản ghi trên mỗi trang

Cuối cùng, chúng ta sử dụng phương thức create\_links để xây dựng các liên kết phân trang.

|  |  |
| --- | --- |
| 1  2 | // build paging links  $params["links"] = $this->pagination->create\_links(); |

Phần còn lại chỉ mang tính hình thức là gọi view user\_listing và kết xuất ra! Truy cập URL <http://your-code-igniter-site/paging/index> để xem danh sách người dùng cùng với các liên kết phân trang.

Như vậy, đó là một ví dụ phân trang vô cùng đơn giản nhưng hữu ích mà bạn có thể mở rộng để phù hợp với các yêu cầu của bạn.

Trong phần tiếp theo, chúng ta sẽ khám phá cách bạn có thể tùy biến phân trang mặc định liên quan đến giao diện và chức năng.

**Khám phá các Tùy chọn Tuỳ biến**

Trong phần này, chúng ta sẽ khám phá các tùy chọn có sẵn mà bạn có thể sử dụng nếu bạn muốn tùy biến các liên kết phân trang mặc định.

Advertisement

**Phân đoạn URL**

Mặc dù thư viện phân trang của CodeIgniter tự động nhận biết tham số liên quan đến phân trang từ URL, nhưng bạn có thể định nghĩa một giá trị tùy biến nếu bạn có mẫu URL khác.

|  |  |
| --- | --- |
| 1 | $config["uri\_segment"] = 4; |

**Số của Liên kết**

Tùy chọn num\_links cho phép bạn định nghĩa con số liên kết *chữ số* sẽ được hiển thị trước và sau số trang hiện tại trong liên kết phân trang.

|  |  |
| --- | --- |
| 1 | $config['num\_links'] = 2; |

**Số trang đóng vai trò là Phân đoạn URI**

Khi bạn truy cập vào phân đoạn URI phân trang, mặc định nó là một chỉ mục bắt đầu. Ví dụ, nếu bạn có mười bản ghi trên mỗi trang, thì phân đoạn URI phân trang là 20 cho trang thứ ba. Thay vào đó, nếu bạn muốn hiển thị số trang thực tế trong các liên kết phân trang, bạn có thể thiết lập use\_page\_numbers thành TRUE.

|  |  |
| --- | --- |
| 1 | $config['use\_page\_numbers'] = TRUE; |

Tất nhiên, bạn cần phải đảm bảo rằng bạn tính toán chỉ số bắt đầu sao cho thích hợp dựa trên số trang mà bạn lấy ra từ URL.

**Giữ nguyên Chuỗi Truy vấn**

Thường thì, bạn sẽ gặp tình huống nơi mà bạn muốn giữ nguyên các tham số của chuỗi truy vấn không liên quan đến phân trang. Bạn có thể sử dụng tùy chọn reuse\_query\_string để kích hoạt tính năng đó.

|  |  |
| --- | --- |
| 1 | $config['reuse\_query\_string'] = TRUE; |

Đây là một vài tuỳ chọn mà bạn có thể sử dụng để thay đổi chức năng phân trang mặc định. Tiếp theo, chúng ta sẽ xem xét một vài tùy chọn khác cho phép bạn thay đổi cái cách các liên kết phân trang được hiển thị.

**Thẻ Wrapper**

Nếu bạn muốn bọc code phân trang bằng bất kỳ thẻ HTML nào khác thì bạn có thể thực hiện bằng cách sử dụng các tùy chọn full\_tag\_open và full\_tag\_close.

|  |  |
| --- | --- |
| 1  2 | $config['full\_tag\_open'] = '<div class="pagination">';  $config['full\_tag\_close'] = '</div>'; |

Nó có thể thật sự hữu ích nếu bạn muốn áp dụng phong cách tùy biến cho các liên kết phân trang.

**Trang đầu, Trang cuối, Tiếp theo, và Trước đó**

Nếu bạn muốn thay đổi phần văn bản đại diện cho các liên kết trang đầu, trang cuối, tiếp theo và trước đó, thì bạn cũng có thể làm điều đó.

|  |  |
| --- | --- |
| 1  2  3  4 | $config['first\_link'] = 'First Page';  $config['last\_link'] = 'Last Page';  $config['next\_link'] = 'Next Page';  $config['prev\_link'] = 'Prev Page'; |

Ngoài ra, nếu bạn muốn bọc các liên kết riêng lẻ đó đó bằng bất kỳ thẻ HTML nào, bạn có thể thực hiện theo cách tương tự như chúng ta đã làm để bọc toàn bộ code phân trang.

|  |  |
| --- | --- |
| 01  02  03  04  05  06  07  08  09  10  11 | $config['first\_tag\_open'] = '<span class="firstlink">';  $config['first\_tag\_close'] = '</span>';    $config['last\_tag\_open'] = '<span class="lastlink">';  $config['last\_tag\_close'] = '</span>';    $config['next\_tag\_open'] = '<span class="nextlink">';  $config['next\_tag\_close'] = '</span>';    $config['prev\_tag\_open'] = '<span class="prevlink">';  $config['prev\_tag\_close'] = '</span>'; |

**Liên kết hiện tại và liên kết số**

Đôi khi, bạn muốn trang trí liên kết hiện tại khác những liên kết còn lại. Bạn có thể làm điều đó bằng cách áp dụng các thẻ wrapper như dưới đây.

|  |  |
| --- | --- |
| 1  2 | $config['cur\_tag\_open'] = '<span class="curlink">';  $config['cur\_tag\_close'] = '</span>'; |

Tương tự, nếu bạn muốn bọc các liên kết số bằng một cái gì đó:

|  |  |
| --- | --- |
| 1  2 | $config['num\_tag\_open'] = '<span class="numlink">';  $config['num\_tag\_close'] = '</span>'; |

Và điều đó đã kết thúc câu chuyện về tuỳ biến. Trên thực tế, bạn có thể xem ví dụ về tuỳ biến tại trang [http://your-code-igniter/site/custom](http://your-code-igniter-site/paging/custom) đã được bao gồm trong tập tin controller của chúng ta!

**Cấu hình Phân trang**

Bây giờ bạn đã tìm hiểu về cấu hình cần thiết để thiết lập một phân trang khá hoàn chỉnh với bất kỳ model nào. Và phần lớn, bạn muốn giữ cho nó trông giống nhau trên toàn bộ trang web. Bạn sẽ làm gì để đạt được điều đó? Nếu không biết, bạn có thể sao chép code cấu hình và dán nó vào mỗi hành động yêu cầu cấu hình phân trang.

Thực tế, có một cách tốt hơn để bạn có thể xử lý tình huống này. Bạn có thể tạo tập tin cấu hình phân trang tại application/config/pagination.php và sử dụng biến $config để định nghĩa các cài đặt của bạn.

|  |  |
| --- | --- |
| 01  02  03  04  05  06  07  08  09  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28 | <?php  $config['per\_page'] = 10;  $config["uri\_segment"] = 3;    $config['full\_tag\_open'] = '<div class="pagination">';  $config['full\_tag\_close'] = '</div>';    $config['first\_link'] = 'First Page';  $config['first\_tag\_open'] = '<span class="firstlink">';  $config['first\_tag\_close'] = '</span>';    $config['last\_link'] = 'Last Page';  $config['last\_tag\_open'] = '<span class="lastlink">';  $config['last\_tag\_close'] = '</span>';    $config['next\_link'] = 'Next Page';  $config['next\_tag\_open'] = '<span class="nextlink">';  $config['next\_tag\_close'] = '</span>';    $config['prev\_link'] = 'Prev Page';  $config['prev\_tag\_open'] = '<span class="prevlink">';  $config['prev\_tag\_close'] = '</span>';    $config['cur\_tag\_open'] = '<span class="curlink">';  $config['cur\_tag\_close'] = '</span>';    $config['num\_tag\_open'] = '<span class="numlink">';  $config['num\_tag\_close'] = '</span>'; |

Dựa vào đó, phương thức hành động index được sửa đổi sẽ như sau:

|  |  |
| --- | --- |
| 01  02  03  04  05  06  07  08  09  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28  29  30  31 | public function index()  {      // load db and model      $this->load->database();      $this->load->model('Users');        // init params      $params = array();      $start\_index = ($this->uri->segment(3)) ? $this->uri->segment(3) : 0;      $total\_records = $this->Users->get\_total();        // load config file      $this->config->load('pagination', TRUE);      $settings = $this->config->item('pagination');      $settings['total\_rows'] = $this->Users->get\_total();      $settings['base\_url'] = base\_url() . 'paging/config';        if ($total\_records > 0)      {          // get current page records          $params["results"] = $this->Users->get\_current\_page\_records($settings['per\_page'], $start\_index);            // use the settings to initialize the library          $this->pagination->initialize($settings);            // build paging links          $params["links"] = $this->pagination->create\_links();      }        $this->load->view('user\_listing', $params);  } |

Tất nhiên, các biến total\_rows và base\_url sẽ thay đổi ứng với các hành động khác nhau, vì vậy bạn cần thiết lập chúng một cách rõ ràng trong mỗi hành động.

Để thực hiện điều đó, bạn cần phải nạp cấu hình phân trang lúc bắt đầu.

|  |  |
| --- | --- |
| 1  2 | $this->config->load('pagination', TRUE);  $settings = $this->config->item('pagination'); |

Tiếp theo, bạn có thể thay thế các cài đặt cụ thể của hành động.

|  |  |
| --- | --- |
| 1  2 | $settings['total\_rows'] = $this->Users->get\_total();  $settings['base\_url'] = base\_url() . 'paging/config'; |

Và bạn hoàn thành nó!

Như vậy đó là câu chuyện về cấu hình phân trang, và nó cũng đã kết thúc bài viết này!

**Phần tóm tắt**

Hôm nay, chúng ta đã đi tìm hiểu về thư viện phân trang trong CodeIgniter.

Trong phần đầu của bài này, tôi đã minh hoạ cách bạn có thể sử dụng thư viện phân trang bằng cách cung cấp một ví dụ rất đơn giản nhưng hữu ích.

Sau đó, chúng ta đã thảo luận về các tùy chọn tùy biến sẵn có trong khi thiết lập phân trang.

Sau cùng, chúng ta đã thảo luận về việc cấu hình phân trang trong phần cuối.

CodeIgniter là một nền tảng PHP mạnh mẽ. Cho dù bạn chỉ mới bắt đầu làm quen hay bạn đang bắt đầu với phiên bản tiếp theo, thì cũng đừng quên tham khảo [những gì chúng tôi có thể cung cấp cho bạn](https://codecanyon.net/search?utf8=%E2%9C%93&term=codeigniter&referrer=homepage).

Tôi rất muốn biết phản hồi của bạn thông qua các yêu cầu và bình luận ở bên dưới đây!