

Lab 07. Bài thực hành Kiểm tra sự hợp lệ dữ liệu và phân trang

Bài 01. Hãy viết các đoạn mã kiểm tra hợp lệ dữ liệu cho các thành phần trên form sau đây:

The image shows a file structure in VS Code with a folder named 'BTVALIDATION' containing files like '.vscode', 'fields.php', 'index.php', and 'validate.php'. Below this is a screenshot of a browser window titled 'Validation Data' showing a registration form. The form has four fields: 'Fullname', 'Date of birthday', 'Telephone', and 'Email', each with a red validation message 'Data entry required!' displayed next to it. There are also 'Register' and 'Reset' buttons at the bottom of the form.

+ Nếu đăng ký thông tin thành công sẽ hiển thị thông báo:

The image shows a browser window with a success message dialog box. The message says 'localhost:8080 says Registration information is successful!' and has an 'OK' button.

- **B1.** Định nghĩa lớp **fields.php** chứa các phương thức lấy tên và lấy thông điệp lỗi ... các thành phần (fields) trên form.

```
<?php
class Field {
    private $name;
    private $message = "";
    private $hasError = false;
    public function __construct($name, $message = "") {
        $this->name = $name;
```

```
$this->message = $message;  
}  
  
public function getName() { return $this->name; }  
  
  
public function getMessage() { return $this->message; }  
public function hasError() { return $this->hasError; }  
public function setErrorMessage($message) {  
  
    $this->message = $message;  
    $this->hasError = true;  
}  
  
public function clearErrorMessage() {  
  
    $this->message = " ";  
    $this->hasError = false;  
}  
  
public function getHTML() {  
  
    $message = htmlspecialchars($this->message);  
    if ($this->hasError()) {  
        return '<span style="color:red">' . $message . '</span>';  
    } else {  
        return '<span>' . $message . '</span>';  
    }  
}  
}  
  
class Fields {  
  
    private $fields = array();  
  
    public function addField($name, $message = "") {  
  
        $field = new Field($name, $message);  
        $this->fields[$field->getName()] = $field;  
    }  
  
    public function getField($name) {  
  
        return $this->fields[$name];  
    }  
  
    public function hasErrors() {  
  
        foreach ($this->fields as $field) {  
  
            if ($field->hasError()) { return true; }  
        }  
    }  
}
```

```

    }
    return false;
}
}
?>
```

- **B2.** Định nghĩa lớp **validate.php** chứa các phương thức xử lý kiểm tra hợp lệ dữ liệu như text(), phone(), email() ...

```

<?php

class Validate {
    private $fields;
    public function __construct() {
        $this->fields = new Fields();
    }
    public function getFields() {
        return $this->fields;
    }
    // Validate a generic text field
    public function text($name, $value, $required = true, $min = 1, $max = 255) {
        // Get Field object
        $field = $this->fields->getField($name);
        // If field is not required and empty, remove error and exit
        if (!$required && empty($value)) {
            $field->clearErrorMessage();
            return;
        }
        // Check field and set or clear error message
        if ($required && empty($value)) {
            $field->setErrorMessage('Data entry required !');
        } else if (strlen($value) < $min) {
            $field->setErrorMessage('Data entered is too short !');
        } else if (strlen($value) > $max) {
            $field->setErrorMessage('Data entered is too long !');
        } else {
            $field->clearErrorMessage();
        }
    }
}
```

```
        }

    }

// Validate a generic number field

public function number($name, $value, $required = true) {

    // Get Field object

    $field = $this->fields->getField($name);

    // Call the text method and exit if it yields an error

    $this->text($name, $value, $required);

    if ($field->hasError()) { return; }

    // Check field and set or clear error message

    if (!is_numeric($value)) {

        $field->setErrorMessage('Must be a valid number.);

    } else {

        $field->clearErrorMessage();

    }

}

// Validate a field with a generic pattern

public function pattern($name, $value, $pattern, $message, $required = true) {

    // Get Field object

    $field = $this->fields->getField($name);

    // If field is not required and empty, remove errors and exit

    if (!$required && empty($value)) {

        $field->clearErrorMessage();

        return;

    }

    // Check field and set or clear error message

    $match = preg_match($pattern, $value);

    if ($match === false) {

        $field->setErrorMessage('Error testing field.);

    } else if ( $match != 1 ) {

        $field->setErrorMessage($message);

    } else {

        $field->clearErrorMessage();

    }

}
```

```
}

public function phone($name, $value, $required = false) {
    $field = $this->fields->getField($name);
    // Call the text method and exit if it yields an error
    $this->text($name, $value, $required);
    if ($field->hasError()) { return; }
    // Call the pattern method to validate a phone number
    $pattern = '/^[:digit:]]{10}$/';
    $message = 'Invalid phone number.';
    $this->pattern($name, $value, $pattern, $message, $required);
}

public function email($name, $value, $required = true) {
    $field = $this->fields->getField($name);
    // Call the text method and exit if it yields an error
    $this->text($name, $value, $required);
    if ($field->hasError()) { return; }
    // Use filter_var method to validate the email address
    $email = filter_var($value, FILTER_VALIDATE_EMAIL);
    // If email address is not valid, set error message and exit
    if ($email === false) {
        $field->setErrorMessage("Invalid email address");
    } else {
        $field->clearErrorMessage();
    }
}
?>
```

- B3. Thiết kế form như giao diện và viết code xử lý

The screenshot shows a browser window with the title 'Validation Data'. The address bar displays 'localhost:8080/BtValidation/'. The main content is a form titled 'Registration information' with the following fields:

- Fullname:
- Date of birthday: mm/dd/yyyy
- Telephone:
- Email:
-

```
<?php

include_once('fields.php');

include_once('validate.php');

$action = filter_input(INPUT_POST,'action');

$fullname = filter_input(INPUT_POST,'fullname');

$birthday = filter_input(INPUT_POST,'birthday');

$telephone = filter_input(INPUT_POST,'telephone');

$email = filter_input(INPUT_POST,'email');

//Sets up fields

$validate = new Validate();

$fields = $validate->getFields();

$fields->addField('fullname');

$fields->addField('birthday');

$fields->addField('telephone');

$fields->addField('email');

if(!empty($action) && $action == 'register')

{

    //Validate form data

    $validate->text('fullname',$fullname);

    $validate->text('birthday',$birthday);

    $validate->phone('telephone',$telephone,true);

    $validate->email('email',$email);
```

```
if(!$fields->hasErrors())  
{  
    echo "<script>alert('Registration information is successful !');</script>";  
}  
}  
?  
<!doctype html>  
<html>  
<head>  
<meta charset="utf-8">  
<title>Validation Data</title>  
</head>  
<body>  
<div style="width:500px; margin:auto;">  
    <form action="" method="post">  
        <fieldset>  
            <legend>Registration information</legend>  
            <table>  
                <tr>  
                    <td>Fullname:</td>  
                    <td>  
                        <input type="text" name="fullname"  
                               value="<?php if(isset($fullname)) echo $fullname;?>">  
                        <!-- Dispaly message error -->  
                        <?php echo $fields->getField('fullname')->getHTML(); ?>  
                    </td>  
                </tr>  
                <tr>  
                    <td>Date of birthday:</td>  
                    <td>  
                        <input type="date" name="birthday"  
                               value="<?php if(isset($birthday)) echo $birthday;?>">  
                        <?php echo $fields->getField('birthday')->getHTML(); ?>  
                    </td>  
                </tr>
```

```
<tr>
    <td>Telephone:</td>
    <td>
        <input type="text" name="telephone"
            value="<?php if(isset($telephone)) echo $telephone;?>">
        <?php echo $fields->getField('telephone')->getHTML(); ?>
    </td>
</tr>
<tr>
    <td>Email:</td>
    <td>
        <input type="email" name="email"
            value="<?php if(isset($email)) echo $email;?>">
        <?php echo $fields->getField('email')->getHTML(); ?>
    </td>
</tr>
<tr>
    <td colspan="2">
        <input type="hidden" name="action" value="register">
        <input type="submit" value="Register">&ampnbsp
        <input type="reset" value="Reset">
    </td>
</tr>
</table>
</fieldset>
</form>
</div>
</body>
</html>
```

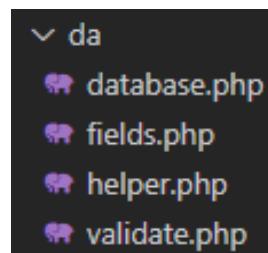
Bài 02. Hãy viết các đoạn mã kiểm tra hợp lệ dữ liệu trên form **Add Product** (tương tự các em thực hành kiểm tra hợp lệ dữ liệu trên form **Add Category**).

+)**Cách 1:** sử dụng hai file: **fields.php** và **validate.php** định nghĩa các phương thức xử lý nghiệp vụ kiểm tra hợp lệ dữ liệu trên form. (sử dụng hai file này ở ví dụ bài tập 01)

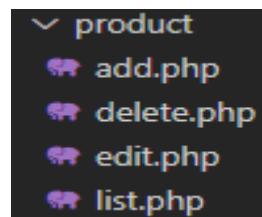
- B1. Giao diện form **Add Product**

The screenshot shows a web page titled "My Laptop Shop" with a "Product List" header. On the left, there's a sidebar with links for "Categories" (Dell, Asus, Hp) and "Add Product". The main area contains a form for adding a product. It has fields for "Category" (a dropdown menu showing "Dell"), "Code" (an input field), "Name" (an input field), and "List Price" (an input field with a validation message: "Must be a valid number"). Below the form is a button labeled "Add Product". At the bottom, there's a link "View Product List" and a copyright notice "© 2020 My Laptop Shop".

- B2. Thêm hai file **fields.php** và **validate.php** vào thư mục **lab07\model\da**



- B3. Bổ sung thêm đoạn mã xử lý kiểm tra hợp lệ dữ liệu trên form **Add Product** (cụ thể file **product\add.php**)



```
<?php
$categories = CategoryDB::getCategories();
//Sets up fields
$validate = new Validate();
$fields = $validate->getFields();
```

```
$fields->addField('code');

$fields->addField('name');

$fields->addField('price','Must be a valid number.');

if(Helper::is_submit('add_product'))

{

$product = new Product();

$product->setCategoryId(Helper::input_value('category_id'));

$product->setCode(Helper::input_value('code'));

$product->setName(Helper::input_value('name'));

$product->setPrice(Helper::input_value('price'));

//Validate form data

$validate->text('code',$product->getCode(),true,1,10);

$validate->text('name',$product->getName());

$validate->number('price',$product->getPrice());

if(!$fields->hasErrors() && ProductDB::addProduct($product))

{

    Helper::redirect('?c=listpro');

}

}

?>

<h1>Add Product</h1>

<form action="" method="post" id="action_form">

<input type="hidden" name="action" value="add_product">

<label>Category:</label>

<select name="category_id">

<?php

    if(!empty($categories))

        foreach ($categories as $category) : ?>

            <option value="<?php echo $category->getId(); ?>">

                <?php echo $category->getName(); ?>

            </option>

        <?php endforeach; ?>

    </select>

    <br>

    <label>Code:</label>
```

```

<input type="input" name="code" value=<?php echo Helper::input_value('code');?>>
<!-- Dispaly message error -->
<?php echo $fields->getField('code')->getHTML(); ?>
<br>
<label>Name:</label>
<input type="input" name="name" value=<?php echo Helper::input_value('name'); ?>>
<?php echo $fields->getField('name')->getHTML(); ?>
<br>
<label>List Price:</label>
<input type="input" name="price" value=<?php echo Helper::input_value('price'); ?>>
<?php echo $fields->getField('price')->getHTML(); ?>
<br>
<label>&nbsp;</label>
<input type="submit" value="Add Product">
<br>
</form>
<p><a href=<?php echo Helper::get_url('admin/?c=listpro');?>>View Product List</a></p>

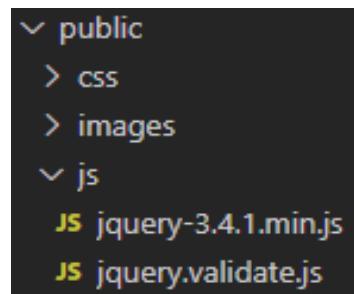
```

- B4. Giao diện form Add Product khi người dùng không nhập dữ liệu.

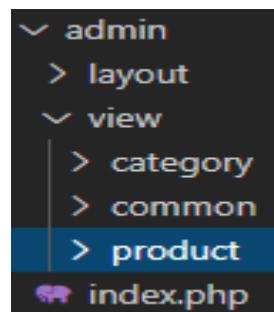
The screenshot shows a web browser window titled "My Laptop Shop". The URL in the address bar is "localhost:8080/Lab07/admin/?c=addpro". The page content includes a sidebar with categories: Dell, Asus, Hp. The main area has a title "Product List" and a sub-section "Add Product". It features a dropdown menu for "Category" set to "Dell", and three input fields for "Code", "Name", and "List Price", all of which are empty. Below these fields is a button labeled "Add Product". To the right of each empty input field is a red error message: "Data entry required!". At the bottom of the form is a link "View Product List". The footer of the page says "© 2020 My Laptop Shop."

+) **Cách 2:** sử dụng file thư viện **jquery.validate.js** kiểm tra hợp lệ dữ liệu trên form.

- **B1.** Thêm hai file **jquery-3.4.1.min.js** và **jquery.validate.js** vào project



- **B2.** Mở file **admin\index.php** (controller), khai báo đường dẫn đến hai file **jquery-3.4.1.min.js** và **jquery.validate.js**



```
<script src="../public/js/jquery-3.4.1.min.js"></script>
<script src="../public/js/jquery.validate.js"></script>
```

- **B3.** Thêm đoạn mã **jquery** sau vào file file **product\add.php**, kiểm tra sự hợp lệ dữ liệu form **Add Product**.

```
<?php
    $categories = CategoryDB::getCategories();
    if(Helper::is_submit('add_product'))
    {
        $product = new Product();
        $product->setCategoryId(Helper::input_value('category_id'));
        $product->setCode(Helper::input_value('code'));
        $product->setName(Helper::input_value('name'));
        $product->setPrice(Helper::input_value('price'));
        if(ProductDB::addProduct($product))
        {
            Helper::redirect('?c=listpro');
        }
    }
?>
```

```

<h1>Add Product</h1>

<form action="" method="post" id="action_form">
    <input type="hidden" name="action" value="add_product">
    <label>Category:</label>
    <select name="category_id">
        <?php
            if(!empty($categories))
                foreach ($categories as $category) : ?>
                    <option value="<?php echo $category->getId(); ?>">
                        <?php echo $category->getName(); ?>
                    </option>
                <?php endforeach; ?>
        </select>
        <br>
        <label>Code:</label>
        <input type="input" style="color:black;" name="code" value="<?php echo Helper::input_value('code');?>">
        <br>
        <label>Name:</label>
        <input type="input" style="color:black;" name="name" value="<?php echo Helper::input_value('name');?>">
        <br>
        <label>List Price:</label>
        <input type="input" style="color:black;" name="price" value="<?php echo Helper::input_value('price');?>">
        <br>
        <label>&nbsp;</label>
        <input type="submit" value="Add Product">
        <br>
    </form>
    <p><a href="<?php echo Helper::get_url('admin/?c=listpro');?>">View Product List</a></p>
    <script>
        $(document).ready(function () {
            //validation
            $("#action_form").validate({
                rules: {
                    code: {
                        required: true
                }
            })
        })
    </script>

```

```

    },
    name: {
        required: true
    },
    price: {
        required: true,
        number:true
    }
},
messages: {
    code: {
        required: "<span style='color:red;width:200px'>Data entry required !</span>"
    },
    name: {
        required: "<span style='color:red;width:200px'>Data entry required !</span>"
    },
    price: {
        required: "<span style='color:red;width:200px'>Data entry required !</span>",
        number: "<span style='color:red;width:200px'>Must be a valid number !</span>"
    }
}
});
});
});

</script>

```

- B4. Giao diện kiểm tra hợp lệ dữ liệu trên form

The screenshot shows a web application interface for managing a laptop shop. On the left, a sidebar lists product categories: Dell, Asus, and Hp. The main content area is titled "My Laptop Shop" and displays a "Product List". Below this, there is an "Add Product" form. The "Category" field is currently set to "Dell". The "Code" field is empty, and a red validation message "Data entry required !" is displayed next to it. The "Name" and "List Price" fields are also empty, with their own red validation messages. At the bottom of the form is a "Add Product" button and a "View Product List" link.

Bài 03. Hãy viết các đoạn mã phân trang cho trang **list Product** (tương tự các em thực hành phân trang **list Category**).

Giao diện trang **product\list.php** sau khi phân trang.

- **B1.** Định nghĩa phương thức **paging()** cho lớp **Helper** để xử lý phân trang.

```
public static function paging($link, $total_records, $current_page, $limit)
{
    $total_page = ceil($total_records / $limit);
    // Limit current_page in 1 to total_page
    if ($current_page > $total_page) {
        $current_page = $total_page;
    } else if ($current_page < 1) {
        $current_page = 1;
    }
    $start = ($current_page - 1) * $limit;
    $html = '<ul class="pagination">';
    if ($current_page > 1 && $total_page > 1) {
        $html .= '<li class="page-item"><a class="page-link" href="" . str_replace('{page}', $current_page - 1, $link) . ">Prev</a></li>';
    }
    for ($i = 1; $i <= $total_page; $i++) {
        if ($i == $current_page) {
            $html .= '<li class="page-link bg-warning">' . $i . '</li>';
        }
    }
}
```

```

} else {
    $html .= '<li class="page-item"><a class="page-link" href="" . str_replace('{page}', $i, $link) . '">' . $i . '</a></li>';
}
}

if ($current_page < $total_page && $total_page > 1) {
    $html .= '<li class="page-item"></li><a class="page-
link" href="" . str_replace('{page}', $current_page + 1, $link) . '">Next</a></li></ul>';
}

return array(
    'start' => $start,
    'limit' => $limit,
    'html' => $html
);
}
}

```

- **B2.** Thêm phương thức **db_num_rows()** cho file **database.php** để lấy số bản ghi từ câu truy vấn.

```

public static function db_num_rows($sql = "")

{
    $count = 0;
    if(!is_null(self::$con))
    {
        $result = self::$con->prepare($sql);
        $result->execute();
        $count = $result->rowCount();
        $result->closeCursor();
        return $count;
    }
    return false;
}

```

- **B3.** Thêm phương thức **getProductsPagingByCategoryId()** cho file **product_db.php** để xử lý nghiệp vụ phân trang cho bảng Products.

```

public static function getProductsPagingByCategoryId($categoryid,&$paging_html)
{
    $link = Helper::get_url("admin/?c=listpro&catid=$categoryid&page={page}");
}

```

```

$sql = "select * from products where categoryID=$categoryid";
$total_records = self::db_num_rows($sql);
$current_page = Helper::input_value('page');
$limit = 5;
$paging = Helper::paging($link,$total_records,$current_page,$limit);
$paging_html = $paging['html'];
$sql = "select * from products where categoryID=:categoryID limit {$paging['start']},{$paging['limit']}";
$params = ['categoryID' => $categoryid];
if(!empty(self::db_get_list_condition($sql,$params)))
{
    foreach(self::db_get_list_condition($sql,$params) as $row)
    {
        $product = new Product();
        $product->setId($row['productID']);
        $product->setCategoryId($row['categoryID']);
        $product->setCode($row['productCode']);
        $product->setName($row['productName']);
        $product->setPrice($row['listPrice']);
        $products[] = $product;
    }
}
return $products;
}
}

```

- B4. Bổ sung đoạn mã sau cho file **product\list.php** (view) để hiển thị phân trang.

```

<style>
ul.pagination {margin: 0px;padding:0px;}
ul.pagination li{float:left;padding-right: 5px;}
</style>
<?php
$catid = Helper::input_value('catid');
if(empty($catid))
{
    $catid = ProductDB::getMinCatId('CategoryId','Categories');
}

```

```
$category = CategoryDB::getCategoryById($catid);
/*Paging*/
$paging_html = "";
$products = ProductDB::getProductsPagingByCategoryId($catid,$paging_html);
?>
/* .....Đây là đoạn mã hiển thị nội dung bảng Products.....*/
<div>
<?php
echo $paging_html;
?>
</div>
<p><a href="<?php echo Helper::get_url('admin/?c=addpro');?>">Add Product</a></p>
```