ساخت لینکدونی با CakePHP (بخش سوم)

پیش نیازها: ساخت لینکدونی با CakePHP (بخش اول) و (بخش دوم)

در مراحل قبل داده‌ها را از دیتابیس خواندیم و در قالب لینک نمایش دادیم. در این بخش قصد داریم امکان افزودن لینک جدید، ویرایش و حذف لینک را اضافه کنیم و مختصری هم با Routes در کیک پی‌اچ‌پی آشنا شویم.

ساخت فرم و افزودن لینک

طبق نمونه‌های گذشته به فایل links_controller.php کنش add را بصورت زیر اضافه می‌کنیم تا بتوانیم لینک‌های جدیدی در دیتابیس اضافه کنیم:

```
<?php
class LinksController extends AppController {
  var $name = 'Links';
  function index() {
    $this->set('links', $this->Link->find('all'));
  }
  function view($id = null) {
    $this->Link->id = $id;
    $this->set('link', $this->Link->read());
  }
  function add() {
    if (!empty($this->data)) {
     if ($this->Link->save($this->data)) {
     $this->flash('Your link has been saved.', '/links');
     }
    }
  }
}
?>
```

تابع add را به این صورت تعریف می‌کنیم که چنانچه فرم ارسال شده خالی نبود، با استفاده از مدل Link سعی شود داده ذخیره شود اما چنانچه بنا به دلایلی ذخیره نشد، نما نمایش داده شود. این خود فرصتی را در اختیارمان قرار می‌دهد تا خطاهای ناشی از محتوای فیلدهای ارسالی را نمایش دهیم.

هنگامی کاربر به روش POST داده‌ها را به برنامه ارسال می‌کند، این اطلاعات در $this->data وجود دارند با استفاده از تابع pr می‌توانید آن‌ها را چاپ کنید. تابع $this->flash() یک متد کنترلر است که برای چند ثانیه پیامی را به کاربر نشان می‌دهد. سپس کاربر را به صفحه‌ای که به عنوان پارامتر دوم دریافت می‌کند هدایت می‌کند. (در اینجا کاربر به صفحه‌ی links هدایت می‌شود)

در اینجا لازم است در مورد اشکال زدایی توکار فریم‌ورک (Debug) توضیح بدهم. کیک برای اشکال زدایی دو حالت تولید (Production) و توسعه (Development) دارد. حالت تولید مربوط به زمانی است که پروژه تان با کیک اتمام یافته و قصد دارید آن را ارائه کنید. حالت توسعه هم گویای زمان توسعه پروژه است. این حالت‌ها در فایل app\config\core.php با مقداردهی عددی از 0 تا 3 برای debug قابل تنظیم است.

مقدار **0** مربوط به حالت تولید می‌شود، در این حالت هیچ خطا و هشداری مشاهده نمی‌شود و تابع **Flash** که در بالا معرفی شد بصورت **redirect** عمل میکند (بطور خودکار کاربر به صفحه دیگر منتقل می‌شود) در حالتی که این مقدار بزرگتراز صفر تنظیم شود فریم‌ورک در حالت توسعه است و تابع **Flash** بطور خودکار عمل انتقال کاربر به صفحه جدید را انجام نمی‌دهد (بطور پیشفرض مقدار روی **2** تنظیم شده است)

متد **save** خطاها را بررسی خواهد کرد و چنانچه خطایی رخ دهد عمل ذخیره انجام نخواهد شد. در قسمت بعد توضیح خواهم داد که چگونه این خطاها را کنترل کنید.

معتبرسازی داده

هر برنامه‌نویس تحت وبی رویکردی برای بررسی صحت اطلاعات وارد شده در فیلدهای فرم‌ها دارد و تقریبا معتبرسازی اطلاعات وارد شده و هدایت کاربر به پرکردن صحیح فرم‌ها کار وقت گیری است. کیک در این مورد بی‌اندازه متنوع و انعطاف‌پذیر است.

برای بهره‌گیری از قابلیت‌های معتبرسازی فرم‌ها کافیست از **FormHelper** کیک استفاده کنید. **FormHelper** بطور پیشفرض در همه نماها که در آن عملگر $form بکار رود موجود است. نمای **add** را بصورت زیر ایجاد می‌کنیم:

```
/app/views/links/add.ctp
<h1>Add Link</h1>
< ?php
echo $form->create('Link');
echo $form->input('title');
echo $form->input('url');
echo $form->input('body', array('rows' => '3'));
echo $form->end('Save Link');
?>
```

در اینجا از ($form->create() بمنظور ایجاد تگ باز **form** استفاده کردیم این کد خروجی زیر را تولید می‌کند:

```
<form id="LinkAddForm" method="post" action="/cake/links/add">
```

اگر تابع **create** هیچ پارامتری نداشته باشد، بطور پیشفرض کیک **action** و **method** را مقداردهی می‌کند. از $form->input() هم برای ایجاد عناصر فرم با همان نامی که به عنوان پارامتر می‌گیرد استفاده می‌شود. کیک توسط اولین پارامتر تشخیص می‌دهد که فیلد از چه نوعی است و دومین پارامتر این امکان را فراهم می کند که آرایه وسیعی با اختیارات فراوان تعیین کنیم. در اینجا تعداد سطرهای **textarea** لحاظ شده است. input() عناصر فرمی متفاوتی را، مبنی بر مدل فیلد مشخص شده، ایجاد می‌کند و این یکی از شگردهای کیک است.

با فراخوانی $form->end() یک کلید ارسال ایجاد، و تگ **form** بسته می‌شود. رشته‌ای که به عنوان اولین پارامتر end() قرار می‌گیرد، برابر مقدار (**value**) کلید ارسال خواهد بود.

حال اجازه بدهید به صفحه اصلی لینکدونی برگردیم و در فایل **app/views/links/index.ctp** لینک صفحه جدیدی را که بمنظور افزودن داده به دیتابیس ساختیم، قبل از جدول اضافه کنیم:

```
<?php echo $html->link('Add Link','/links/add')?>
```

ممکنه تعجب کنید که چطور کیک صحت اعتبار این فیلدها را بررسی می‌کند. قاعده‌های معتبرسازی در مدل تعریف می‌شوند. بدین منظور مجدداً به مدل **Link** بر می‌گردیم و اصلاحاتی را به شکل زیر لحاظ می‌کنیم:

```
<?php
class Link extends AppModel {
   var $name = 'Link';
   var $validate = array(
     'title' => array(
     'rule' => array('minLength', 1)
    ),
     'url' => array(
     'rule' => array('minLength', 1)
    ),
     'body' => array(
     'rule' => array('minLength', 1)
    )
  );
}
?>
```

آرایه $validate به کیک می‌گوید که هنگامی که متد save() فراخوانی شد چگونه صحت اطلاعات بررسی شود. در اینجا سه فیلد عنوان، آدرس و توضیحات را چک می‌کنیم که نبایستی خالی باشد. یکبار دیگر تکرار می‌کنم موتور معتبرسازی کیک بسیار قوی است چرا که شمار زیادی از قواعد پیش‌ساخته شده (بررسی صحت آدرس ایمیل، شماره کارت اعتباری، تلفن و ...) در آن گنجانده شده است و سفارشی سازی آنها انعطاف پذیری فوق العاده‌ای دارد. برای اطلاعات بیشر می توانید به اینجا رجوع کنید.

هم‌اکنون قواعد معتبرسازی لحاظ شده‌اند. فرم افزودن لینک را با فیلدهای خالی تست کنید تا ببینید چگونه کار می‌کند. هنگامی که از input() برای ایجاد عناصر فرم استفاده می‌کنیم، خطاهای مربوط به عدم صحت داده بطور خودکار نمایش داده می‌شوند.

حذف کردن لینک

قصد داریم امکانی را ایجاد کنیم که کاربران بتوانند لینک‌ها را از دیتابیس پاک کنند. خوب! با کنش ()delete در **LinksControlle** به شکل زیر شروع می کنیم:

```
function delete($id) {
  $this->Link->del($id);
  $this->flash('The link with id: '.$id.' has been deleted.', '/links');
}
```

این تابع با گرفتن $id هر لینک، آن را از دیتابیس پاک می‌کند و با استفاده از ()flash پیام تاییدی را مبنی بر حذف لینک قبل از ری‌دایرکت (انتقال خودکار) شدن به صفحه‌ی **links** به کاربر نشان می‌دهد. با این وجود یکبار دیگر باید فایل **index.ctp** را بصورت زیر ویرایش کنیم و قابلیت حذف هر لینک را به آن بیفزاییم:

```
<h1>Links:</h1>
<p>< ?php echo $html->link('Add Link','/links/add')?></p>

<table>
<tr>
<th>Id</th>
<th>Title</th>
<th>Actions</th>
<th>Created</th>
</tr>
< ?php foreach ($links as $link): ?>
<tr>
<td>< ?php echo $link['Link']['id']; ?></td>
<td>< ?php echo $html->link($link['Link']['title'],
"/links/view/".$link['Link']['id']); ?></td>
<td>< ?php
echo $html->link('Delete', "/links/delete/{$link['Link']['id']}", null, 'Are you sure?' )
?></td>
<td>< ?php echo $link['Link']['created']; ?></td>
</tr>
< ?php endforeach; ?>
</table>
```

کد بالا از قابلیت توکار **HtmlHelper** بمنظور ایجاد یک دیالوگ جاوااسکریپتی تایید حذف لینک، قبل از اینکه کاربر لینک را حذف کند، استفاده می کند. می‌بینید این فریم‌ورک تا چه اندازه همه چیز را ساده کرده است!

ویرایش لینک

اگر مراحل قبل را فراگرفته باشید از الان شما یک حرفه‌ای کیک پی‌اچ‌پی هستید! پس می‌بایستی الگوی ادامه کار را حدس بزنید. یک کنش می‌سازیم و سپس نما را اضافه می‌کنیم. در زیر کنش edit() را که باید در **LinksControlle** لحاظ شود می‌بینید:

```
function edit($id = null) {
  $this->Link->id = $id;
  if (empty($this->data)) {
   $this->data = $this->Link->read();
  } else {
   if ($this->Link->save($this->data)) {
      $this->flash('Your link has been updated.','/links');
   }
  }
}
```

این تابع ابتدا داده ارسالی فرم را بررسی می‌کند ، اگر هیچ چیزی فرستاده نشده باشد لینک‌ها را به نما منتقل می‌کند. اگر داده‌ای در نتیجه تغییر، ارسال شده باشد سعی می‌کند از طریق مدل **Link** داده جدید را در دیتابیس ذخیره کند (یا خطاهای ناشی از عدم اعتبار داده را نشان خواهد داد) نمای **edit** بصورت زیر است:

```
/app/views/links/edit.ctp
<h1>Edit Link</h1>
< ?php
echo $form->create('Link', array('action' => 'edit'));
echo $form->input('title');
echo $form->input('url');
echo $form->input('body', array('rows' => '3'));
echo $form->input('id', array('type'=>'hidden'));
echo $form->end('Save Link');
```

خروجی کد بالا فرمی خواهد بود که درفیلدهای آن اطلاعات لینکی که درخواست ویرایش آن داده شده قرار دارد. باید در اینجا توجه کنید که **id** هر لینک بصورت یک عنصر مخفی (**hidden**) در فرم آمده است. در حقیقت هنگامی که **id** هر لینک با اطلاعات ارسال شده باشد کیک تشخیص می‌دهد که این لینک باید ویرایش شود در صورتی که اگر **id** با اطلاعات ارسالی نباشد هنگامی که تابع save() فراخوانی می‌شود یک لینک جدید اضافه خواهد شد.

مجدداً به نمای **index** برمی‌گردیم و قابلیت ویرایش هر لینک را به صورت زیر در کنار پیوند مربوط به حذف لینک اضافه می‌کنیم (تا در صفحه‌ی اصلی قابلیت ویرایش لینک را داشته باشیم):

```
<?php echo $html->link('Edit', '/links/edit/'.$link['Link']['id']);?>
```

تا اینجا آموختید که چگونه با **CakePHP** یک لینکدونی ساده بسازید. قابلیت‌هایی نظیر افزودن، ویرایش و حذف لینک تنها به این دلیل بود که با نحوه ایجاد کنش در کنترلر آشنا شوید و بتوانید نماهای مربوطه را ایجاد کنید. به ادمه بحث در مورد **Routes** (مسیرها) می‌پردازیم.

مسیرها (Routes) در CakePHP

برای بعضی‌ها تنظیمات پیشفرض کیک در مورد مسیرها کافی است. اما توسعه دهندگان نسبت به کاربرپسند کردن **URL**ها و سازگاری با موتورهای جستجو حساس هستند. از اینرو می‌توانند از تنظیمات **Routes** استفاده کنند. در این آموزش تغییرات ناچیزی را در این مورد لحاظ می‌کنیم برای اطلاعات بیشتر و تکنیک‌های مسیردهی پیشرفته، بخش **Routes Configuration** در مستندات کیک را ببینید.

تا اینجا که لینکدونی را ایجاد کردیم چنانچه کاربری به صفحه‌ی اصلی (شاخه دایرکتوری اصلی) به نشانی **example.com/cake** مراجعه کند بطور پیشفرض **PagesController** فراخوانی می‌شود و نمایی بنام **home** رندر می‌شود. قصد داریم مسیر را طوری تعیین کنیم که با درخواست شاخه اصلی، لیست لینک‌ها نمایش داده شود.

مسیرها در فایل **app/config/routes.php** تعیین می‌شوند. خط زیر مسیر پیشفرض شاخه اصلی را تعیین می‌کند. آن را توسط کامنت (//) غیرفعال کنید یا این خط را پاک کنید:

```
Router::connect('/', array('controller' => 'pages', 'action' => 'display', 'home'));
```

خط فوق آدرس '/' را با صفحه پیشفرض کیک مرتبط می‌کند. ما می‌خواهیم آن را با کنترلر خودمان ارتباط دهیم بنابراین خط زیر را به این فایل اضافه می‌کنیم:

```
Router::connect('/', array('controller' => 'links', 'action' => 'index'));
```

حال با رفتن به صفحه اصلی کیک به نشانی **example.com/cake** بجای صفحه پیشفرض لیست لینک‌ها را می‌بینید. می‌توانید نام شاخه اصلی را که در بخش اول آموزش **cake** گذاشتید، به **linkdump** تغییر دهید (تا هخوانی بیشتری با پروژه داشته باشد) بنابراین آدرس صفحه اصلی به **example.com/linkdump** تغییر خواهد کرد.

اگر نام شاخه را از **cake** به **linkdump** تغییر دادید فایل‌های تمپ را پاک کنید تا صفحه اصلی را ببینید. تنها جهت آشنایی شما با خطاهای پیکربندی اولیه کیک، نام دایرکتوری را ابتدا **cake** انتخاب کردم چون می‌توانست از همان ابتدا **linkdump** انتخاب شود.

سخن آخر

نوشتن این لینکدونی در اینجا به پایان رسید. ساده نبود؟! یادتان باشد این آموزش پایه بود CakePHP خیلی خیلی قابلیت‌های بیشتری از آنچه گفته شد دارد و بیش از آنچه تصور کنید انعطاف‌پذیر است. متاسفم از اینکه باید بگویم فرصت کافی ندارم تا آنچه از کیک می‌دانم بزبان ساده بیان کنم. از همین الان می‌توانید پروژه‌های آزمایشی را با کیک شروع کنید. بزرگترین راهنما یعنی Manual و API کیک در اختیار شماست. اگر مشکلی داشتید در حد دانشم درخدمتم. موفق باشید. (مرتضی الوانی آبانماه 1387)

دریافت فایل‌های ضمیمه

دایرکتوری app این پروژه بهمراه فایل sql ساخت جدول

آموزش کامل (بخش‌های اول و دوم و سوم) بصورت PDF