Tutorial Zend bagian 4

Wirawan Prasetyo
Bestfriends_wp@yahoo.com
http://www.wirawanprasetyo.web.id

4. Form Input dan list data dari database

4.1. Hasil Akhir

Jawaban yang akan kita dapatkan setelah mengerjakan tutorial pada bagian ke empat. Kita akan tahu bagaimana cara membuat aplikasi dengan menggunakan fasilitas autentifikasi database dan menggunakan Access Control List yang disediakan oleh Zend Framework

Disini kita membuat tampilan list user yg sudah terdaftar di database

4.2. Contekan

4.2.1. Pada halaman utama index.php kita akan masih menampilkan hello word yg sudah dibuat di sesi sebelumnya.

4.2.2. Pada header kita akan membuat beberapa link yaitu : list (untuk menampilkan list user) dan tombol login / logout dan tombol registrasi

4.2.3. Setelah login, user hanya bisa mengubah data pribadinya saja. Sedangkan admin bisa mengubah data semua user.

4.3. Kisi-Kisi

Klas-klas tambahan dari zend yang akan digunakan adalah :

- Zend_Db

Kita akan menggunakan database yg akan menyimpan data user

- Zend_Config

  Klass ini akan memanggil suatu file “namaKlass.ini” yang berisi data seperti username dan password ke db, nama table dan domain

4.4. Mebuat table untuk menampung data user

```
CREATE TABLE IF NOT EXISTS `user` (
  `id` int(11) NOT NULL auto_increment,
  `username` varchar(100) NOT NULL,
  `userpassword` varchar(100) NOT NULL,
  `email` varchar(100) default NULL,
  `level` varchar(30) NOT NULL default 'user',
  PRIMARY KEY  (`id`)
) ENGINE=MyISAM DEFAULT CHARSET=latin1
AUTO_INCREMENT=1 ;
```

Penjelasan 4.4 :

- Kita membuat table user dengan kolom ‘id’ sebagai unik
- Kita membuat kolom “level” dengan nilai default “user”. Ini yg menandaan hak akses seseorang.

Sebagai user pertama, kita membuat user “admin” dengan password “admin” dan level “admin”.

```
INSERT INTO `root`.`user` ( `id` , `username` , `userpassword` , `email` , `level` )
VALUES ( NULL , 'admin', MD5( 'admin' ) , 'admin@root.com', 'admin' ), ( NULL , '', '', NULL , 'user' );
```

Penjelasan :

- Password kita hash dengan MD5

4.5. Membuat file konfigurasi

Buatlah file “config.ini” kemudian simpan di “Root\application”. Dengan isi :

#\application\config.ini

```
[general]
```

```
db.adapter = PDO_MYSQL
db.config.host = localhost
db.config.username = gantidenganusernamekedatabase
db.config.password = gantidenganpasswordkedatabase
db.config.dbname = gantidengannamadatabase
```

4.6. Perubahan pada header

#header.phtml (Root\application\view\script\header.phtml)

```
.....
<body>
<div id="headerRoot">Header root<hr></div>
<div id="linkRoot">
<a href="<?php echo $this->baseUrl;?>">Index</a>
<a href="<?php echo $this->baseUrl;?>/index/list">List</a>
<a href="<?php echo $this-
>baseUrl;?>/index/register">Register</a>
</div>
<div id="isiRoot">
```

Penjelasan :

- Disini kita menambah tiga link, yaitu ke halaman index, halaman registrasi dan list

4.7. Membuat halaman list

Root\application\view\script\index\list.phtml

```
<?php echo $this->render('header.phtml'); ?>
<table id="tblList" border="1">
<tr><td>Nama</td><td>Email</td><td>Edit</td><td>Delete</
td></tr>
```

```
<?php foreach($this->listUser as $baris) : ?>

<tr>

<td><?php echo $this->escape($baris->username);?></td>

<td><?php echo $this->escape($baris->userpassword);?></td>


<td><a href="<?php echo $this->baseUrl; ?>/index/edit/id/<?php
echo $baris->id;?>">Edit</a></td>

<td><a href="<?php echo $this->baseUrl; ?
>/index/delete/id/<?php echo $baris->id;?>">Delete</a></td>


</tr>

<?php endforeach; ?>


</table>

<?php echo $this->render('footer.phtml'); ?>
```

Penjalasan :

- Pada controller kita mengirim variable array listUser. listUser kemudian ditampilkan
- Kemudian ditampilkan dengan *foreach*

4.8. Membuat halaman registrasi

Root\application\view\script\index\register.phtml

```
<?php echo $this->render('header.phtml'); ?>

        <?php echo $this->render('index/_form.phtml'); ?>

<?php echo $this->render('footer.phtml'); ?>
```

Penjelasan :

- Dari script terlihat kita mengambil halaman “_form.phtml”. Ini sama saja ketika kita memanggil

  header.phtml

- Kita melakukan ini untuk penghematan
- Ketika kita melakukan registrasi dan mengubah data kita, sebenarnya tampilan form bisa sama. Karena itu kita membuat sebuah halaman "_form" agar dapat digunakan selama berhubungan dengan data user.

4.9. Mebuat halaman "_form"

Root\application\view\script\index_form.phtml

```
<form action="<?php echo $this->baseUrl ?>/index/<?php echo
$this->action; ?>" method="post">

<table>

<tr><td>Masukkan username</td><td><input type="text"
name="txtUsername" value="<?php echo $this-
>escape(trim($this->user->username));?>"></td></tr>

<tr><td>Masukkan email</td><td><input type="text"
name="txtEmail" value="<?php echo $this->escape(trim($this-
>user->email));?>"></td></tr>

<tr><td>Masukkan pssword</td><td><input type="password"
name="txtPass" value="<?php echo $this->escape(trim($this-
>user->userpassword));?>"></td></tr>

<tr><td colspan="2">

                <input type="hidden" name="id" value="<?php echo
$this->user->id; ?>" />

                <input type="submit" name="add"    value="<?php
echo $this->escape($this->buttonText); ?>" />

</td></tr>

</table>

</form>
```

Penjelasan :

- Kita kita melakukan registrasi, maka tidak ada data yg ditampilkan
- Sedangkan ketika melakukan pengeditan, kita akan terlebih dahulu menampilkan data yang akan di edit. <?php echo $this->escape(trim($this->user->username));?>" dan ="<?php echo $this->escape(trim($this->user->userpassword));?>">.

4.10.Membuat halaman edit

Root\application\view\script\index\register.phtml

```
<?php echo $this->render('header.phtml'); ?>

        <?php echo $this->render('index/_form.phtml'); ?>

<?php echo $this->render('footer.phtml'); ?>
```

Penjelasan :

- Halaman ini sama persis dengan halaman registrasi. Namun disini, kita menampilkan data yg akan di edit.

4.11.Membuat halaman penghapusan

Root\application\view\script\index\delete.phtml

```
<?php echo $this->render('header.phtml'); ?>

<?php if ($this->user) :?>

<form action="<?php echo $this->baseUrl ?>/index/delete"
method="post">

<p>Apakah anda yakin ingin menghapus data

'<?php echo $this->escape($this->user->username); ?>' by

'<?php echo $this->escape($this->user->email); ?>'?

</p>

<div>

<input type="hidden" name="id" value="<?php echo $this->user-
>id; ?>" />

<input type="submit" name="del" value="Yes" />

<input type="submit" name="del" value="No" />

</div>

</form>

<?php else: ?>

<p>Cannot find album.</p>

<?php endif;?>
```

```
<?php echo $this->render('footer.phtml'); ?>
```

Penjelasan :

- Ketika user menekan link delete, maka user akan dibawa ke aksi “delete”.
- Kemudian aksi akan memvalidasi apakah $_REQUEST yang sudah dikirim.
- Jika belum ada, maka akan ditampilkan pesan untuk memvalidasi “Apakah anda yakin ingin menghapus data”
- Setelah memilih “yes / no “, maka user akan kembali di redirect ke aksi yang sama namun kali ini dengan mengirim $_REQUEST yang sudah diisi.
- Jika memilih “yes” dan “id” yg dikirim benar, maka user dengan “id” terkirim akan langsung dihapus
- User kemudian kembali di redirect ke halaman list

4.12.Menambahkan konfigurasi di bootstramp (Root\index.php)

```
include "Zend/Loader.php";
Zend_Loader::loadClass('Zend_Controller_Front');
// untuk mengambil nilai di config.inc
Zend_Loader::loadClass('Zend_Config_Ini');
Zend_Loader::loadClass('Zend_Registry');
Zend_Loader::loadClass('Zend_Db');
Zend_Loader::loadClass('Zend_Db_Table');

// load configuration
$config = new Zend_Config_Ini('./application/config.ini',
  'general');
$registry = Zend_Registry::getInstance();
// menampung nilai dari file config ke registry kita
$registry->set('config', $config);

// setup database
$db = Zend_Db::factory($config->db->adapter, $config->db-
  >config->toArray());
Zend_Db_Table::setDefaultAdapter($db);

// setup controller
$frontController = Zend_Controller_Front::getInstance();
.....
```

4.13.Membuat model User

Model ini menerangkan bahwa kita menggunakan table “user”

#Root\application\models\User.php

```
<?php
class User extends Zend_Db_Table
{
   protected $_name = 'user';
}
```

Pada script, menerangkan kita membuat klas User yg merupakan extend(turunan) dan mengambil data dari table user

4.14.Menambah di indexController.php

```
        function init()
        {          $this->initView();$this->view->baseUrl = $this->_request->getBaseUrl();
         Zend_Loader::loadClass('User');
        }
```

Pada 4.13. Kita telah membuat sebuah model klass baru bernama “User”. Disini kita memanggil (Load) klas model yang kita perlukan.

4.15.Mebuat aksi list di indexController.php

Aksi yg dibaca ketika kita menekan link “List”

```
function readAction()
{ ...
}
```

```
function listAction()
	{
		$this->view->title = "List User";
		$user = new User();
		$this->view->listUser = $user->fetchAll();
		$this->render();
	}
```

Ketika kita menekan link list, kita akan memanggil aksi list.

- $user = new User(); => Kita membuat objek baru dari model user.
- $user->fetchAll(); => melakukan Query ke model user kemudian di fetch
    - Karena kita membuat objek dari model user, fungsi "$user->fetchAll();" sama saja dengan query "select * from user". Mudah kan
- $this->view->listUser = $user->fetchAll(); => menampung hasil query ke variabel kita

4.16.Membuat aksi register di indexController.php

Aksi yg dibaca ketika kita menekan link "Register"

```
function listAction()
	{
	}
function registerAction()
	{
		$this->view->title = "Registrasi Member";

		if ($this->_request->isPost())
		{
```

```
        Zend_Loader::loadClass('Zend_Filter_StripTags');
                        $filter = new Zend_Filter_StripTags();


                        $txtUsername = $filter->filter($this-
>_request->getPost('txtUsername'));
                        $txtUsername = trim($txtUsername);
                        $txtEmail = trim($filter->filter($this-
>_request->getPost('txtEmail')));
                        $txtPass = trim($filter->filter($this-
>_request->getPost('txtPass')));



                        if ($txtUsername != '' && $txtEmail !=
'' && $txtPass!= '') {
                                $data = array(
                                'username' => $txtUsername,
                                'email' => $txtEmail,
                                'userpassword' => $txtPass,
                                );
                                $user = new User();
                                $user->insert($data);
                                $this->_redirect('/index/list');


                                return;
                        }
                }


                // set up an "empty"  user
                $this->view-> user = new stdClass();
                $this->view-> user ->id = null;
```

```
                $this->view-> user ->artist = '';
                $this->view-> user ->title = '';
              $this->view-> user ->email = '';

                $this->view->action = 'register';
                $this->view->buttonText = 'Daftar';
                $this->render();
        }
```

Penjelasan :

- if ($this->_request->isPost()) => Ketika seorang user menekan tombol register, sebenarnya dia mengkases kembali aksi yg sama (register). Tapi bedanya, kali ini web mengirim parameter post yang berarti sudah ada data yang akan dimasukkan.
- if ($txtUsername != '' && $txtEmail != '' && $txtPass!= '') =>Memastikan data sudah diisi semua
- $user->insert($data) => Data diinput ke dalam tabel user
- // set up an "empty" user => pada saat kita menampilkan form ke dalam web, form menampilkan nilai2. Jika kita tidak mendefinisikannya, maka akan muncul karakter aneh. Karena ikut kita isi dengan membuat form kosong.

4.17.aksi Edit

```
....
function registerAction()
        { .....
        }

function editAction()
```

```
        {
                $this->view->title = "Edit Member";

                $user = new User();

                if ($this->_request->isPost())
                {

        Zend_Loader::loadClass('Zend_Filter_StripTags');
                        $filter = new Zend_Filter_StripTags();

                        $id = (int)$this->_request->getPost('id');
                        $txtUsername = $filter->filter($this->_request->getPost('txtUsername'));
                        $txtUsername = trim($txtUsername);
                        $txtEmail = trim($filter->filter($this->_request->getPost('txtEmail')));
                        $txtPass = trim($filter->filter($this->_request->getPost('txtPass')));

                        if ($id !== false) {
                                if ($txtUsername != '' && $txtEmail != '' && $txtPass!= '')
                                {
                                $data = array(
                                'username' => $txtUsername,
                                'email' => $txtEmail,
                                'userpassword' => $txtPass,
                                );

                                        $where = 'id = ' . $id;
```

```
                                        $user->update($data,
$where);
                                        $this-
>_redirect('/index/list');
                                        return;
                                }
                                else
                                {
                                        $this->view->user =
$user->fetchRow('id='.$id);
                                }
                        }
                }
                else
                {
                        // album id should be $params['id']
                        $id = (int)$this->_request-
>getParam('id', 0);

                        if ($id > 0) {
                                $this->view->user = $user-
>fetchRow('id='.$id);
                        }
                }
                // additional view fields required by form
                $this->view->action = 'edit';
                $this->view->buttonText = 'Update';

                $this->render();
        }
```

Penjelasan :

- Aturan yg digunakan disini mirip dengan di aksi register. Yang membedakan : jika ada request (if ($this->_request->isPost())) dan validasi semua input berhasil, maka data akan di “update”. Sedangkan di register akan di “inser”
- Jika tidak ada , maka akan menamilkan data yg akan di edit ke dalam “_form”.

4.18.Membuat aksi delete

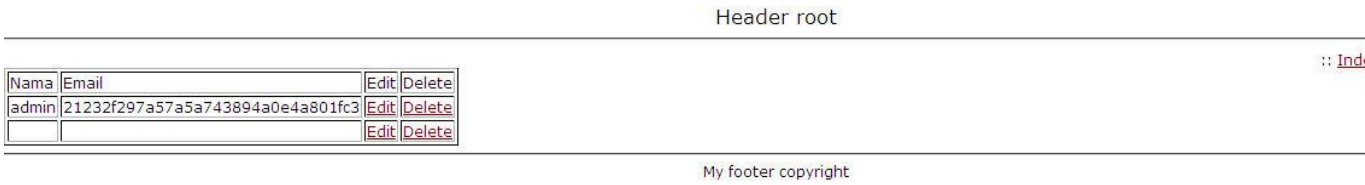
```
...
Function editAction()
{....
}

function deleteAction()
        {
                $this->view->title = "Delete User";
                $user = new User();

                if ($this->_request->isPost()) {

        Zend_Loader::loadClass('Zend_Filter_Alpha');
                        $filter = new Zend_Filter_Alpha();
                        $id = (int)$this->_request->getPost('id');
                        $del = $filter->filter($this->_request->getPost('del'));
                        if ($del == 'Yes' && $id > 0) {
                        $where = 'id = ' . $id;
                        $rows_affected = $user->delete($where);
```

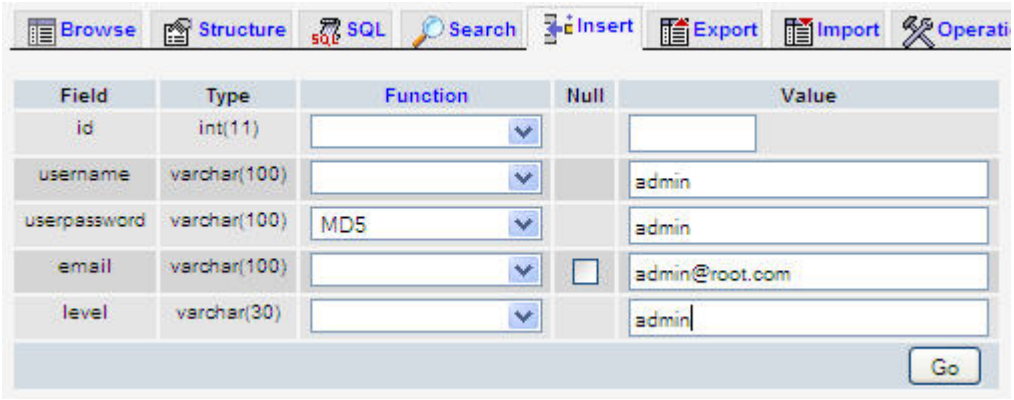
```
                        }
                } else
                {
                        $id = (int)$this->_request-
>getParam('id');
                        if ($id > 0) {
                        // only render if we have an id and can
find the album.
                        $this->view->user = $user-
>fetchRow('id='.$id);
                        if ($this->view->user->id > 0) {
                        $this->render();
                        return;
                        }
                        }
                }
```


```
                                                // redirect back
                                to the album list unless we
have rendered the view
                $this->_redirect('/index/list');
                $this->render();
        }
```

Penjelasan :

- Aksi disini mirip dengan aksi edit. Bedanya, jika di user ada reuest dan validasi input berhasil, maka data akan di updatet Sedangkan disini jika ada request dan validasi inpt berhasil, maka data akan dihapus dari database.
- Jika tidak ada request, maka akan ditampilkan form validasi apakah memang ingin menghapas data tersebut





Biografi Penulis

Wirawan Prasetyo (Wawan) lahir pada bulan Desember tahun 1984. Menyelesaikan S1 di Universitas Bina Nusantara, Jakarta jurusan Teknik Informatika pada tahun 2007. Saat ini berkerja di sebuah media hukum online (www.hukumonline.com) sebagai programmer.

http://www.wirawanprasetyo.web.id