PERTEMUAN KE – 4

Validasi Masukkan dengan Kelas CValidator

A. TUJUAN

1. Memahami penggunaan kelas Validasi Input, mengunakan kelas CValidator,
2. Vaidasi menggunakan ajax.

B. TEORI SINGKAT

Kelas CValidator mempunyai beberapa tipe validator, misalnya: required, numerical. tipe validator ini merupakan alias dari kelas-kelas validator di Yii, misalnya required alias dari kelas CRequiredValidator, numerical alias dari kelas CNumberValidator. Nama model Atribut yang akan di validasi Opsional, tambahan option dari validator misalnya untuk numerical terdapat opsi min dan max, yang mengatur panjang minimal dan maksimal dari data.

Untuk daftar tipe validator yang lengkap bisa di lihat di bawah ini:

- required: CRequiredValidator
- email: CEmailValidator
- url: CUrlValidator
- unique: CUniqueValidator
- compare: CCompareValidator
- length: CStringValidator
- numerical: CNumberValidator
- captcha: CCaptchaValidator
- type: CTypeValidator
- file: CFileValidator
- default: CDefaultValueValidator
- exist: CExistValidator
- date: CDateValidator
- safe: CSafeValidator
- unsafe: CUnsafeValid, dst.

contoh :

```
array('password', 'length', 'min'=>7, 'max'=>64),
array('password','pattern'=>'/^[A-Za-z0-9_!@#$%^&*()+=?.,]+$/u',
 'message'=>'Spaces or given characters are not allowed'),

array('email', 'email','message'=>"The email isn't correct"),
array('email', 'unique','message'=>'Email already exists!'),
array('myDate', 'type', 'type' => 'date',
      'message' => '{attribute}: is not a date!',
      'dateFormat' => 'yyyy-MM-dd'),
```

C. PRAKTIK

1. Gunakan tabel dan dengan model Barang pada kasus sebelumnya.
   Buka model `Barang.php`, perhatikan di bagian fungsi rules() seperti pada kode program berikut :

file Barang.php

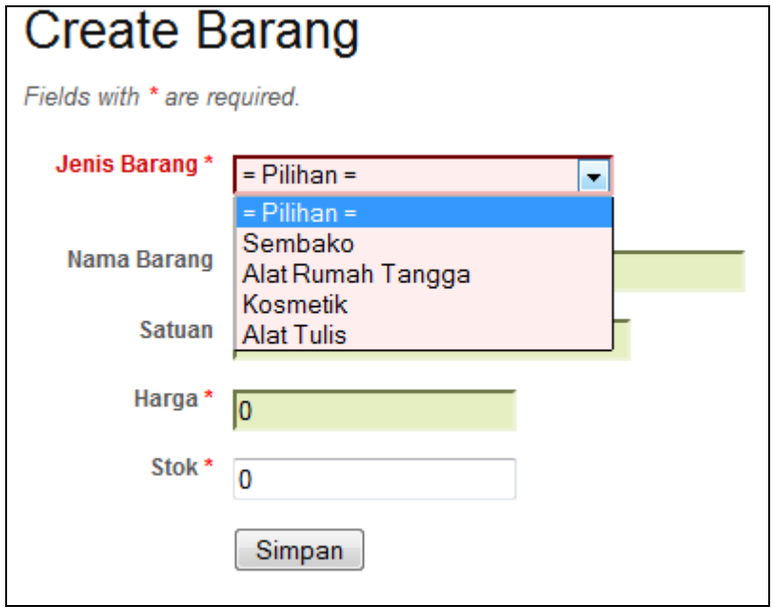
```
class Barang extends CActiveRecord
{
   /**
   * @return string the associated database table name
   */
  public function tableName()
   {
    return 'barang';
   }

      /*****************************************************
       * @return array validation rules for model attributes.
       *****************************************************/
   public function rules()
    {

      return array(
            array('jenis_barang_id, harga, stok','required'),

      array('jenis_barang_id,harga,stok','numerical','integerOnly'=>true),
                  array('nama_barang', 'length', 'max'=>40),
                  array('satuan', 'length', 'max'=>30),
                  array('id,  jenis_barang_id,  nama_barang,  satuan,  harga,
stok',
                        'safe', 'on'=>'search'),
            );
      }
……..
```

akhir file Barang.php



2. Lakukan pengubahan bagian Controller BaranController.php
   Buka file BaranController.php, aktifkan bagian
   `$this->performAjaxValidation($model);`

skrip function actionCreate()

```
public function actionCreate()
```

```
{
 $model=new Barang;

// untuk mengaktifkan ajax ubah seperti dibawah ini
   $this->performAjaxValidation($model);

   if(isset($_POST['Barang']))
       {
            $model->attributes=$_POST['Barang'];
            if($model->save())
               $this->redirect(array('view','id'=>$model->id));
        }

       $this->render('create',array(
                  'model'=>$model,));
    }
```

akhir skrip function actionCreate()

3. Lakukan pengubahan bagian view /protected/views/barang/_form.php

file _form.php

```
<?php
/* @var $this BarangController */
/* @var $model Barang */
/* @var $form CActiveForm */
?>

<div class="wide form">

<?php $form=$this->beginWidget('CActiveForm', array(
      'id'=>'barang-form',
      'enableAjaxValidation'=>true,
      'enableClientValidation'=>true,
      'clientOptions'=>array(
             'validateOnType'=>true,
             'validateOnChange' => true,
            ),
)); ?>

      <p class="note">Fields with
      <span class="required">*</span> are required.</p>

      <?php echo $form->errorSummary($model); ?>

      <div class="row">
            <?php echo $form->labelEx($model,'jenis_barang_id'); ?>
            <?php echo $form->dropDownList($model,'jenis_barang_id',
                    CHtml::listData(JenisBarang::model()->findAll(),
                          'id','jenis_barang'),
              array('prompt'=>'= Pilihan =','style'=>'width:200px;')); ?>
            <?php echo $form->error($model,'jenis_barang_id'); ?>
      </div>

      <div class="row">
            <?php echo $form->labelEx($model,'nama_barang'); ?>
            <?php echo $form->textField($model,'nama_barang',
                  array('size'=>40,'maxlength'=>40)); ?>
            <?php echo $form->error($model,'nama_barang'); ?>
      </div>

      <div class="row">
            <?php echo $form->labelEx($model,'satuan'); ?>
            <?php echo $form->textField($model,'satuan',
                  array('size'=>30,'maxlength'=>30)); ?>
            <?php echo $form->error($model,'satuan'); ?>
```

```
	</div>

	<div class="row">
		<?php echo $form->labelEx($model,'harga'); ?>
		<?php echo $form->textField($model,'harga'); ?>
		<?php echo $form->error($model,'harga'); ?>
	</div>

	<div class="row">
		<?php echo $form->labelEx($model,'stok'); ?>
		<?php echo $form->textField($model,'stok'); ?>
		<?php echo $form->error($model,'stok'); ?>
	</div>

	<div class="row buttons">
		<?php echo CHtml::submitButton($model->isNewRecord ?
				'Simpan' : 'UBah'); ?>
	</div>

<?php $this->endWidget(); ?>

</div><!-- form -->
```

akhir file _form.php

4. Tambahkan properti $_existed pada Barang.php

```
class Barang extends CActiveRecord
{
    private $_existed;

    ……..
}
```

5. Membuat Validasi dengan menggunakan

Buka dan tambahkan file `/protected/models/Barang.php` seperti pada skrip dibawah:

skrip function rules()

```
public function rules()
	{
 return array(
	array('nama_barang','validasiNama'),
	// validasi nama kembar jika digunakan
……..
);
}
```

```
public function validasiNama($attribute, $params)
    {
      $this->_existed = self::model()->exists('nama_barang=:nm_brg', array(
                ':nm_brg'=>$this->nama_barang, ));
      if($this->_existed)
            $this->addError('nama_barang','Nama yang sama sudah ada..!');
    }
```

skrip function rules()

Uji hasilnya, seperti pada gambar berikut :

D. LATIHAN

1. Buatlah tabel user dengan struktur tabel berikut :

```
CREATE TABLE user (
 id int(11) NOT NULL AUTO_INCREMENT,
 full_name char(50) CHARACTER SET utf8 NOT NULL,
 username varchar(128) CHARACTER SET utf8 NOT NULL,
 password varchar(128) CHARACTER SET utf8 NOT NULL,
 email varchar(128) CHARACTER SET utf8 NOT NULL,
 active tinyint(1) NOT NULL DEFAULT '1',
 PRIMARY KEY (`id`)
) ENGINE=InnoDB
```

2. Buatlah model tabel user
3. Buatlah Controroller dan view untuk memasukkan data user
4. Validasi username tidak boleh sama
5. Validasi format penulisan email
6. Validasi tidak boleh sama
7. Panjang password minimal 6 karakter dan maksimal 12 karakter
8. Menyimpan password ter enkripsi dengan MD5,
9. Membuat form masukan dan validasi pengisian password 2 kali
10. Membuat form mengganti password
11. Amati hasilnya

.

E. TUGAS

Apa hasil model generator, dalam model user pada metode rules() jika field-field dalam tabel user didefinisikan validasi dan tipe datanya seperti struktur di bawah:

```
id int(11) NOT NULL AUTO_INCREMENT,
full_name char(50) CHARACTER SET utf8 NOT NULL,
username varchar(128) CHARACTER SET utf8 NOT NULL,
password varchar(128) CHARACTER SET utf8 NOT NULL,
email varchar(128) CHARACTER SET utf8 NOT NULL,
active tinyint(1) NOT NULL DEFAULT '1',
```