

Tutorial Dasar

Yii Framework 2

**Membuat aplikasi sederhana dengan
Yii 2 Advanced**

Dida Nurwanda
www.didanurwanda.com
November 2015

Kata Pengantar

Puji dan syukur saya panjatkan kehadirat Allah SWT yang telah memberikan rahmat-Nya sehingga saya dapat menyelesaikan ebook ini. Saya Alhamdulillah baru saja menyelesaikan S1 dan di wisuda tanggal Minggu 15 November 2015 kemarin, karena masih belum memiliki kesibukan akhirnya saya berkesempatan menulis ebook yang sangat sederhana ini. Ebook ini merupakan ebook pertama saya yang memang masih jauh dari kata bagus. Begitu banyak kekurangan dari isi materi pada ebook ini telah saya sadari, saya minta maaf jika ada kesalahan dalam menjelaskan pada ebook ini.

Pada kesempatan berikutnya, insyaallah saya akan menulis ebook-ebook yang membahas beberapa bahasa pemrograman ataupun php framework. Anda bisa memberi masukan atau meminta isi materi dengan cara mengirim email ke saya secara langsung.

Ebook ini saya bagikan secara gratis beserta source codenya yang dapat anda download di website saya. Saya berterimakasih kepada Mas Sabit Huraira yang dimana sebagian isi ebook ini saya ambil dari ebook yang beliau tulis. Saya juga berterimakasih kepada group Yii Framework Indonesia yang menginspirasi saya menulis ebook ini.

Akhir kata, semoga ebook ini dapat membantu kita semua dalam belajar.

Pandeglang, November 2015

Dida Nurwanda

Daftar Isi

Kata Pengantar	i
Daftar Isi	ii
PART I : INSTALASI	1
A. Instalasi dengan Composer	1
B. Instalasi dengan Download Arsip	2
PART II : Berkenalan	3
A. Apa itu MVC ?	3
B. Struktur Folder	4
C. Pengaturan Dasar	6
D. Perbedaan Basic dan Advanced	11
PART III : DATABASE DAN GII	12
A. Persiapan Database	12
B. Konfigurasi Database Yii 2	15
C. Berkenalan dengan Gii	16
D. CRUD dengan Gii	17
PART IV : TOPIK SPESIAL	21
A. Menenal Model	21
B. Bekerja dengan Database	25
C. Data Formatter	35
PART V : MEMBUAT BLOG SEDERHANA	37

PART I

INSTALASI

Yii Framework atau lebih dikenal dengan sebutan Yii merupakan kerangka kerja open source berbasis PHP. Nama Yii merupakan singkatan dari “Yes It Is!”. Seperti PHP Framework pada umumnya, Yii juga mengadopsi konsep MVC (Model – View – Controller).

Pada perkembangannya, Yii telah memasuki generasi ke-3, dimana perkembangannya di tandai dengan nama versi 1.0.x untuk generasi pertama (3 Desember 2008), 1.1.x untuk generasi kedua (10 Januari 2010), dan 2.0.x untuk generasi ketiga (12 Oktober 2014). Yii versi 2.0.x inilah yang akan kita bahas pada ebook ini.

Pada Yii Framework 2, proses instalasi lazimnya dapat dilakukan dengan dua cara. Cara yang pertama yaitu dengan menginstal secara online dengan bantuan Composer, dan yang kedua yaitu dengan mendownload arsip yang telah di sediakan pada website www.yiiframework.com. Berikut akan dijelaskan kedua proses instalasi Yii Framework 2 tersebut.

A. Instalasi dengan Composer

Untuk dapat melakukan instalasi dengan menggunakan Composer, tentu Anda diwajibkan menginstal Composer terlebih dahulu. Disini saya tidak menjelaskan cara instal dan penggunaan Composer, silahkan Anda pelajari pada website www.getcomposer.org

Jika Anda telah menginstal Composer, maka syarat kedua adalah Anda memiliki koneksi internet. Saya anggap Anda telah terkoneksi dengan internet, maka ikuti langkah berikut.

Buka terminal atau command line, kemudian ketik perintah berikut :

```
php composer.phar global require "fxp/composer-asset-plugin:~1.0.3"
```

Kemudian ketikkan perintah berikut :

```
php composer.phar create-project yiisoft/yii2-app-basic  
c:/xampp/htdocs/yiibasic 2.0.6
```

atau jika ingin menggunakan template advanced :

```
php composer.phar create-project yiisoft/yii2-app-advanced  
c:/xampp/htdocs/yiiadvanced 2.0.6
```

Keterangan :

- Pada perintah *c:/xampp/htdocs/yiibasic* atau *c:/xampp/htdocs/yiiadvanced* merupakan tempat dimana Anda ingin meletakkan atau menyimpan lokasi Yii Framework.
- Pada perintah *2.0.6* merupakan versi Yii Framework 2 yang terakhir di rilis pada pembuatan buku ini. Untuk mendapatkan versi terbaru silahkan Anda sesuaikan dengan versi yang terdapat pada website www.yiiframework.com/download.

B. Instalasi dengan Download Arsip

Instalasi Yii Framework 2 yang lebih mudah adalah dengan mendownload arsip yang telah di sediakan. Anda 2ias mendapatkan arsip tersebut pada website www.yiiframework.com/download. Pada website tersebut Anda akan dihadapkan dua pilihan Yii, yaitu *Yii 2 with basic application template* dan *Yii 2 with advanced application template*. Untuk penjelasan perbedaan antara kedua jenis tersebut, akan dibahas pada BAB berikutnya.

PART II

BERKENALAN

Mungkin Anda bertanya-tanya, kenapa Part II ini diberi judul “Berkenalan”. Berdasarkan judul tersebut, pembahasan pada part ini yaitu mengenal lebih jauh Yii Framework 2, mulai dengan memahami konsep MVC, mengenal struktur folder, pengaturan-pengaturan dasar, dan perbedaan antara Basic Template dan Advanced Template.

A. Apa itu MVC ?

Yii Framework hadir dengan konsep PHP Framework pada umumnya, yaitu dengan pola desain MVC. MVC merupakan singkatan dari Model View dan Controller. MVC merupakan sebuah pattern pemrograman yang memisahkan antara bisnis logic, data logic, dan presentation logic. Secara sederhana, MVC memisahkan antara desain, data, dan proses. Penggunaan MVC pada dasarnya digunakan untuk mempermudah penembang aplikasi dalam mengubah suatu bagian pada aplikasi tanpa harus mengubah bagian lainnya.

1). Model

Dalam MVC, model bertugas dalam menggambarkan suatu informasi atau data disertai dengan aturan bisnisnya. Aturan tersebut meliputi validasi, hubungan antar tabel, dan lain-lain.

2). View

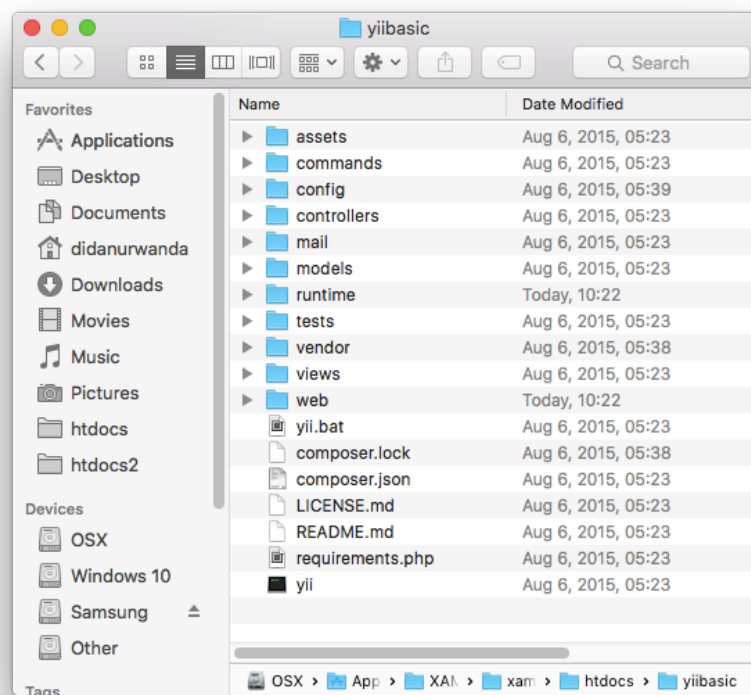
View berhubungan dengan segala sesuatu yang akan di tampilkan pada end-user. Bisa berupa halaman web, rss, javascript, dan lain-lain. Dalam konsep MVC, sebisa mungkin Anda harus menghindari adanya logika pemrosesan yang di simpan dalam view.

3). Controller

Controller merupakan jembatan komunikasi antara Model dengan View. Pada Controller, Anda sebaiknya hindari kode-kode yang bertugas untuk mengakses data secara langsung.

B. Struktur Folder

1). Yii 2 Basic Template

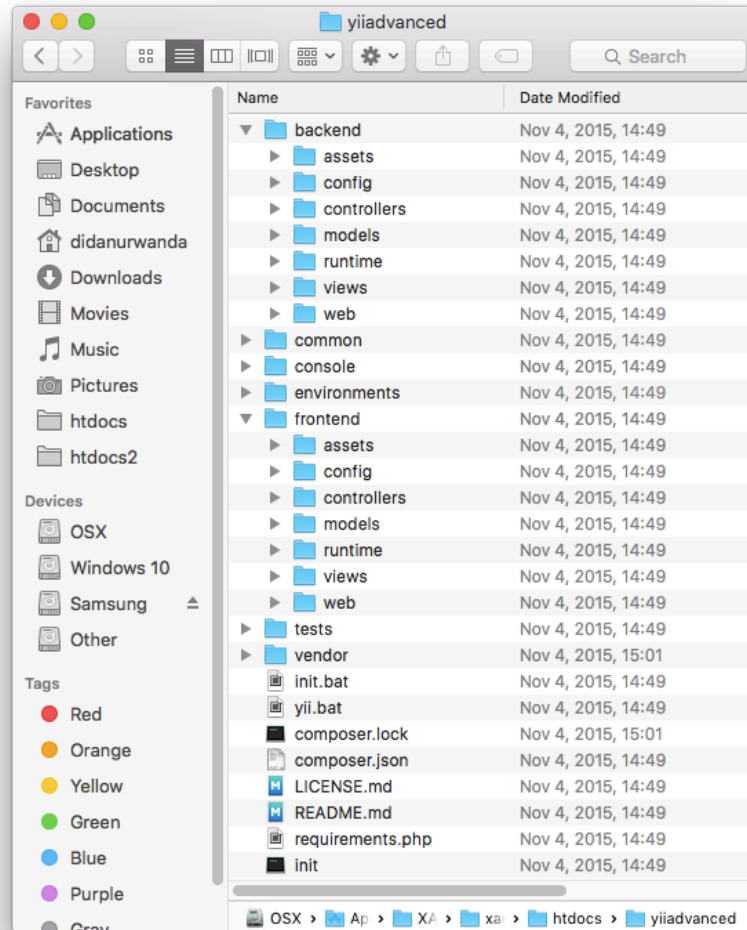


Gambar 2.1

Struktur Folder Yii 2 Basic

Pada Yii 2 Basic, Anda dapat langsung menemukan folder Controller, Model, dan View pada directori root. Folder *assets* berfungsi untuk menyimpan class Asset yang digunakan untuk mendefinisikan file css, js, dan lain-lain yang diperlukan oleh view. Folder *web* berisi file index aplikasi, ini merupakan folder root pada webserver (*public_html*).

2). Yii 2 Advanced Template



Gambar 2.2

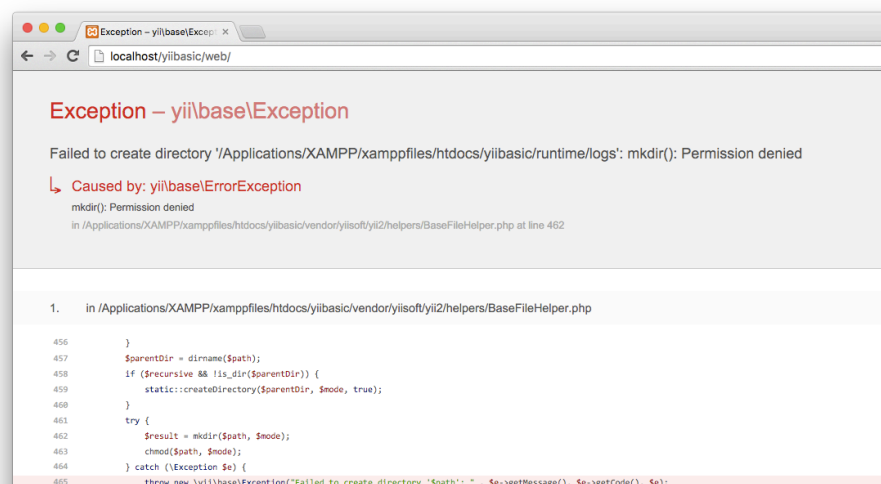
Struktur Folder Yii 2 Advanced

Pada Yii 2 Advanced, Anda akan menemukan folder yang berbeda dengan Yii 2 Basic. Perbedaan mendasar adalah adanya folder *backend* dan *frontend* serta tidak adanya folder Controller, Model, dan View pada halaman root. Jadi pada dasarnya Yii 2 Advanced merupakan dua aplikasi yang berbeda yang memisahkan antara website public dan website administrator.

C. Pengaturan Dasar

1). Menjalankan Yii 2 Basic

Sebelum memulai menjalankannya, pastikan Anda telah menginstal Yii 2 Basic Template pada folder htdocs Anda. Sebagai contoh, disini saya install Yii 2 Basic dengan nama yiibasic. Kemudian buka browser dan ketik *http://localhost/yiibasic/web*. Pada sistem operasi Linux atau OS X, kemungkinan besar akan muncul error seperti berikut.

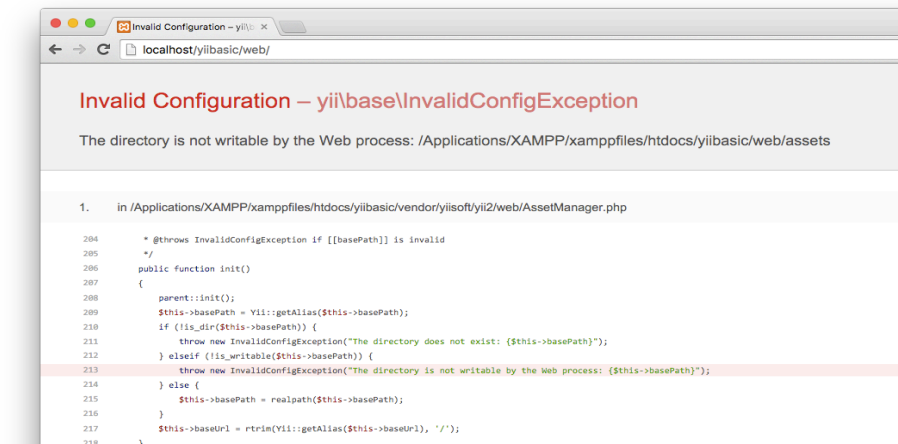


```
1. in /Applications/XAMPP/xamppfiles/htdocs/yiibasic/vendor/yiisoft/yii2/helpers/BaseFileHelper.php

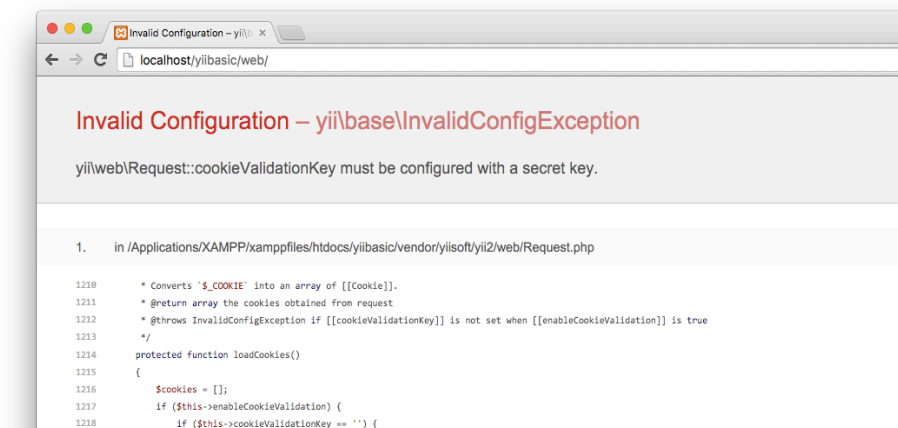
456     }
457     $parentDir = dirname($path);
458     if ($recursive && !is_dir($parentDir)) {
459         static::createDirectory($parentDir, $mode, true);
460     }
461     try {
462         $result = mkdir($path, $mode);
463         chmod($path, $mode);
464     } catch (\Exception $e) {
465         throw new \yii\base\Exception("Failed to create directory '$path': " . $e->getMessage(), $e->getCode(), $e);
```

Gambar 2.3

Permission denied pada folder runtime



Gambar 2.4
Permission denied pada folder web/assets



Gambar 2.5
Error dikarenakan *cookieValidationKey* masih kosong.

Untuk mengatasi masalah-masalah di atas, silahkan Anda ubah permission pada folder runtime dan folder web/assets. Jika sudah, kemudian buka file config/web.php kemudian isi *cookieValidationKey* dengan kata sembarang.

```

<?php

$params = require(__DIR__ . '/params.php');

$config = [
    'id' => 'basic',
    'basePath' => dirname(__DIR__),
    'bootstrap' => ['log'],
    'components' => [

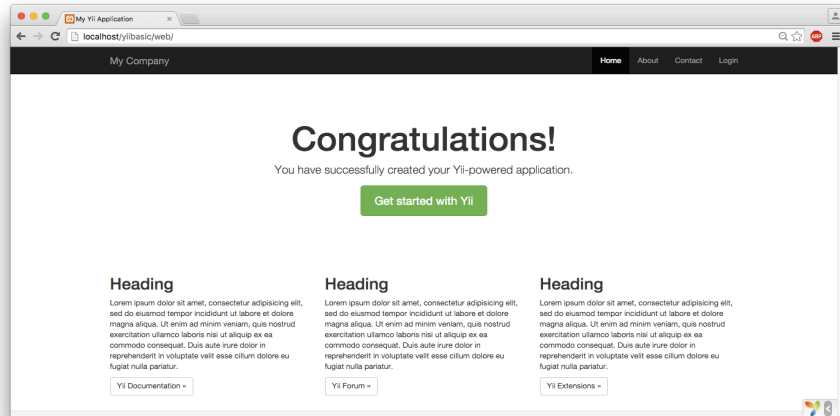
        'request' => [
            // !!! insert a secret key in the following (if
            // it is empty) - this is required by cookie
            // validation
            'cookieValidationKey' => 'didacookie',
        ],

        ...

    ],
    'params' => $params,
];

```

Jika semua telah di lakukan, silahkan reload browser Anda dan berikut adalah hasilnya.



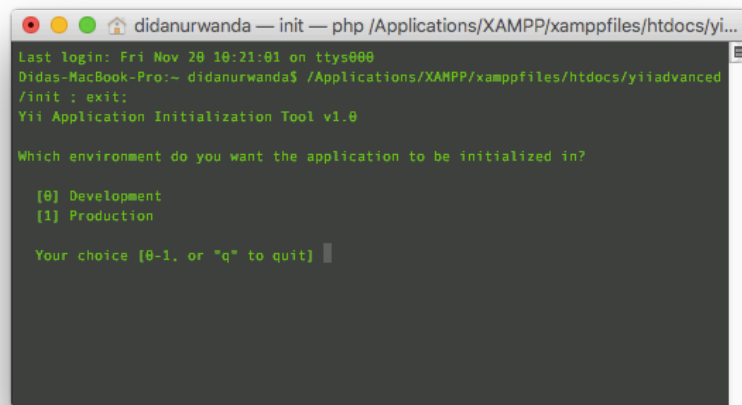
Gambar 2.6

Tampilan Yii 2 Basic Sukses Terpasang

2). Menjalankan Yii 2 Advanced

Sebelum melangkah lebih jauh, pastikan Anda telah selesai menginstal Yii 2 Advanced pada folder htdocs dengan cara yang telah di bahas pada Part I. Untuk contoh, disini saya menginstal Yii 2 Advanced dengan nama folder yiiadvanced.

Pada Yii 2 Advanced memiliki sedikit perbedaan dalam tatacara penggunaan awal. Yaitu harus mendefinisikan terlebih dahulu environment aplikasi, Apakah Development atau Production. Cara mendefinisikannya adalah dengan menjalankan file init.bat (Windows) atau init (Linux dan OS X) sampai muncul tampilan seperti berikut.



```
didanurwanda — init — php /Applications/XAMPP/xamppfiles/htdocs/yi...
Last login: Fri Nov 20 18:21:01 on ttys000
Didas-MacBook-Pro:~ didanurwanda$ /Applications/XAMPP/xamppfiles/htdocs/yiiadvanced
/init : exit:
Yii Application Initialization Tool v1.0

Which environment do you want the application to be initialized in?

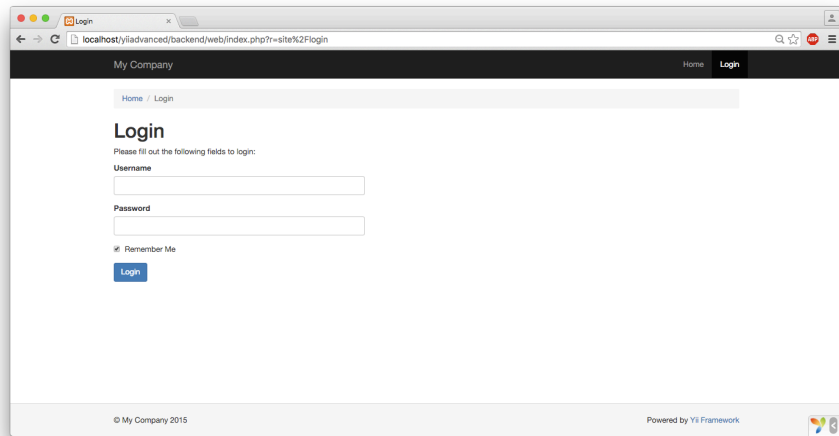
[0] Development
[1] Production

Your choice [0-1, or "q" to quit]
```

Gambar 2.7

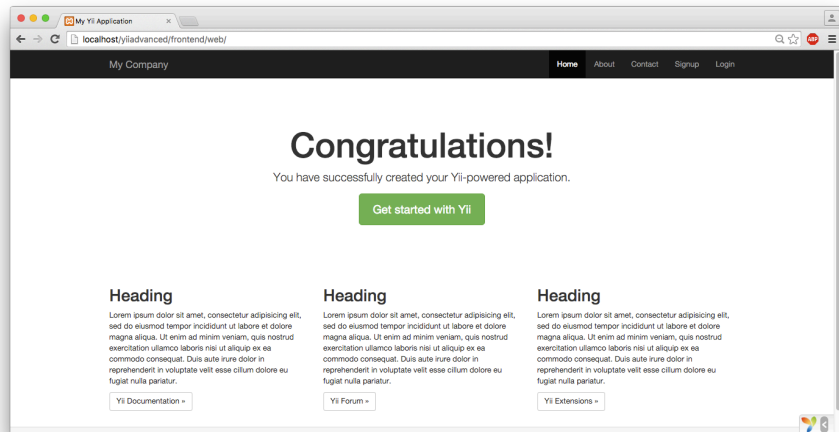
Gambar terminal menjalankan file init

Kemudian anda pilih 0 untuk pengembangan/development, 1 untuk production, atau q untuk keluar. Karena kita di sini masih tahap development, maka pilih 0 dan Enter. Pada Yii 2 Advanced Anda tidak harus mengubah permission pada folder runtime dan assets, sebab telah secara otomatis di ubah saat kita menjalankan file init tersebut. Selanjutnya jika sudah, silahkan Anda buka browser dan masuk ke link <http://localhost/yiiadvanced/backend/web> maka akan muncul tampilan seperti berikut.



Gambar 2.8
Tampilan halaman awal *backend*

Kemudian Anda coba buka halaman frontend dengan masuk ke link <http://localhost/yiiadvanced/frontend/web> maka akan muncul tampilan seperti berikut.



Gambar 2.9
Tampilan Halaman Frontend

D. Perbedaan Basic dan Advanced

Yii 2 hadir dengan dua pilihan konsep, yaitu Basic dan Advanced. Kedua konsep tersebut merupakan pemisahan dari kebutuhan aplikasi, dimana aplikasi yang mengharuskan hak akses secara penuh dapat menggunakan Yii 2 Basic. Sedangkan aplikasi yang bekerja di dua habitat yang berbeda yaitu memiliki akses publik dan administrator akan lebih mudah menggunakan Yii 2 Advanced. Namun, bukan suatu keharusan Anda mengikuti acuan tersebut, sebab Yii 2 memiliki fleksibilitas untuk dapat bekerja di kedua penerapan tersebut.

Pada Yii 2 Advanced, anda akan menemukan folder backend dan frontend, yaitu berfungsi untuk memisahkan antara aplikasi publik dengan administrator. Konsep ini hadir dengan menerapkan dua website berbeda dalam satu aplikasi. Untuk kalangan publik, lazimnya Anda dapat menggunakan frontend dan untuk administrator Anda dapat menggunakan backend. Namun ini bukan berarti tindakan Login atau hak akses hanya berlaku untuk backend saja, anda tetap bias menggunakan hak akses pada frontend tergantung kebutuhan. Bingung ? Pada Part V akan dibahas contoh pembuatan aplikasi dengan menerapkan Yii 2 Advanced.

PART III

DATABASE DAN GUI

A. Persiapan Database

Untuk dapat mempermudah pembelajaran pada buku ini, mari kita seragamkan skema database yang akan kita gunakan pada aplikasi kita. Saya disini menggunakan database MySQL, saya sarankan Anda menggunakan database yang sama dengan skema yang sama. Akan tetapi, tidak masalah jika Anda ingin menggunakan database lain. Sekarang silahkan Anda buat database dengan nama “yiiblog” kemudian buat table dengan nama “user” dengan skema berikut.

Column	Type	Comment
id	int(11) <i>Auto Increment</i>	
first_name	varchar(100)	
last_name	varchar(100)	
username	varchar(255)	
auth_key	varchar(32)	
password_hash	varchar(255)	
password_reset_token	varchar(255) <i>NULL</i>	
email	varchar(255)	
role	smallint(6) [10]	
status	smallint(6) [10]	
created_at	int(11)	
updated_at	int(11)	

Gambar 3.1
Skema Tabel User

Selanjutnya Anda buat table lagi dengan nama “kategori” dan “artikel”. Tabel tersebut Akan terus kita gunakan untuk membuat blog sederhana dengan Yii 2 Advanced pada Part V. Berikut skema dari kedua tabel tersebut.

Column	Type	Comment
id_kategori	int(3) <i>Auto Increment</i>	
nama_kategori	varchar(200)	

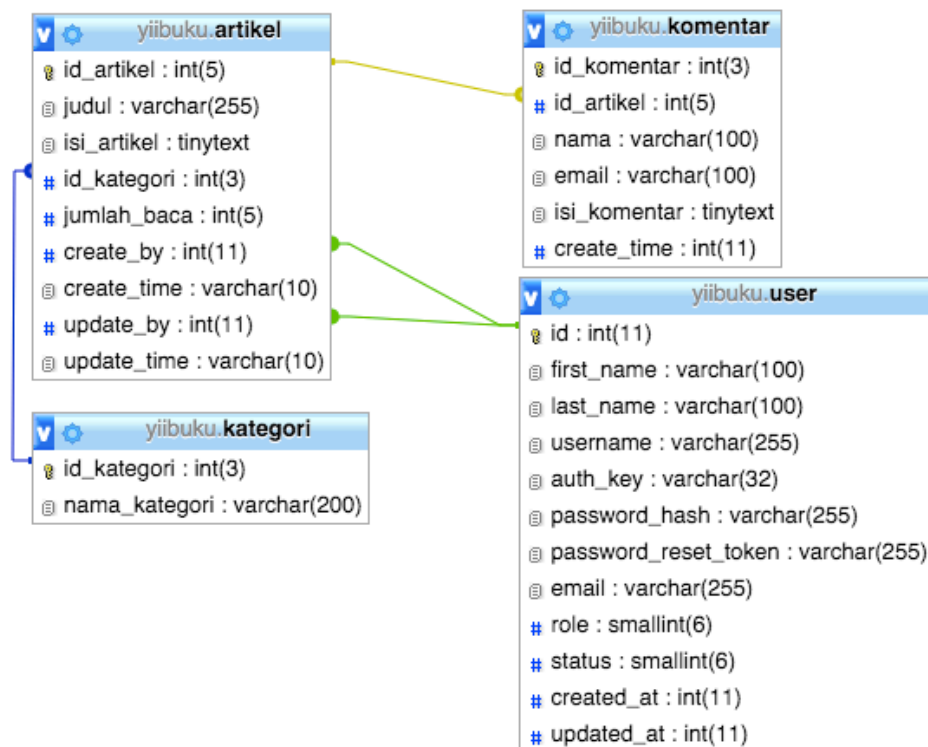
Gambar 3.2
Skema Tabel Kategori

Column	Type	Comment
id_artikel	int(5) <i>Auto Increment</i>	
judul	varchar(255)	
isi_artikel	tinytext	
id_kategori	int(3)	
jumlah_baca	int(5)	
create_by	int(11)	
create_time	varchar(10)	
update_by	int(11)	
update_time	varchar(10)	

Gambar 3.3
Skema Tabel Artikel

Column	Type	Comment
id_komentar	int(3) <i>Auto Increment</i>	
id_artikel	int(5)	
nama	varchar(100)	
email	varchar(100)	
isi_komentar	tinytext	
create_time	int(11)	

Gambar 3.4
Skema Tabel Komentar



Gambar 3.5
Skema Relasi Pada Database yiiblog

Jika semua tabel telah di buat, selanjutnya kita akan mengkoneksikan Yii 2 dengan database tersebut.

B. Konfigurasi Database Yii 2

Untuk melakukan konfigurasi koneksi database, Yii 2 Basic dan Yii 2 Advanced memiliki struktur file konfigurasi yang berbeda. Pada Yii 2 Basic kita bisa melakukan konfigurasi dengan mengubah file *config/db.php* dan pada Yii 2 Advanced bisa dilakukan dengan membuka file *common/config/main-local.php*. Meski struktur file konfigurasi berbeda, tapi konfigurasinya tetap sama.

```
<?php  
  
return [  
    'class' => 'yii\db\Connection',  
    'dsn' => 'mysql:host=localhost;dbname=dbname',  
    'username' => 'root',  
    'password' => '',  
    'charset' => 'utf8',  
];
```

Class merupakan komponen koneksi, biarkan default jika Anda tidak membuat atau menginginkan penggunaan class baru sebagai core koneksi. Pada *dsn* secara terdapat konfigurasi mysql, disini kita di hadapkan dengan konfigurasi host dan nama database, ubah sesuai dengan server Anda, Terdapat pula username, password dan charset tentu saja isi sesuai dengan server yang Anda gunakan.

Sebagai bahan pembelajaran Part III, disini kita akan bahas contoh penggunaan Yii 2 Basic. Kita akan menggunakan yiibasic yang telah Anda install sebelumnya. Kemudian buka file *config/db.php* dan ubah seperti gambar di bawah ini, jika terdapat perbedaan konfigurasi dengan server atau database yang Anda gunakan, silahkan sesuaikan konfigurasinya.

```
<?php  
  
return [  
    'class' => 'yii\db\Connection',  
    'dsn' => 'mysql:host=localhost;dbname=yiiblog',
```

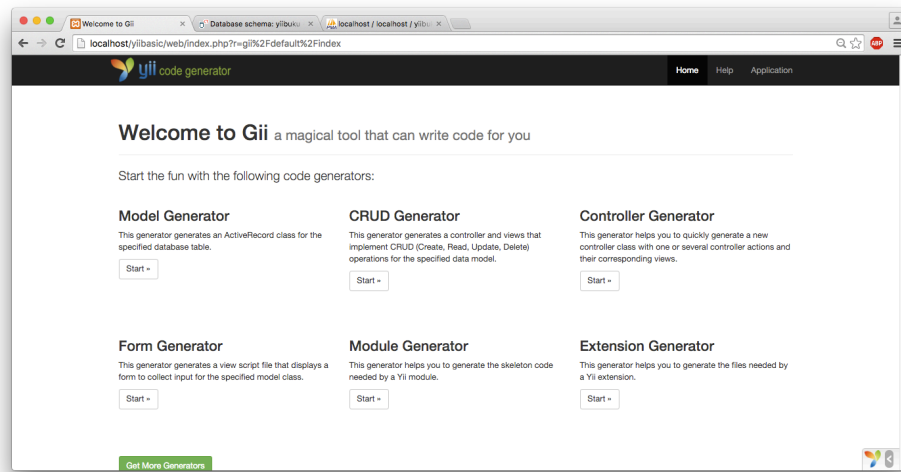
```
'username' => 'root',  
'password' => '',  
'charset' => 'utf8',  
];
```

C. Berkenalan Dengan Gii

Gii merupakan modul yang telah disediakan oleh Yii. Gii merupakan fitur unggulan Yii yang berfungsi untuk menggenerate atau menciptakan file Model, Controller, Module, dan bahkan kode lengkap untuk melakukan CRUD (Create, Read, Update, Delete).

Untuk dapat melakukan generate CRUD dengan Gii kita disyaratkan untuk membuat table yang memiliki primary key. Sebelum menggenerate CRUD dengan Gii, Anda terlebih dahulu harus menciptakan/menggenerate Model yang mewakili satu table. Model yang di generate oleh Gii akan secara otomatis dapat mengakses atau melakukan join data dengan tabel lain dengan catatan Anda telah melakukan relasi pada saat pembuatan tabel.

Jika ingin membuka halaman Gii, silahkan Anda buka browser dan masuk ke halaman <http://localhost/yiibase/web/index.php?r=gii> pada Yii 2 Basic dan <http://localhost/yiiadvanced/backend/web/index.php?r=gii> atau <http://localhost/yiiadvanced/frontend/web/index.php?r=gii> pada Yii 2 Advanced.



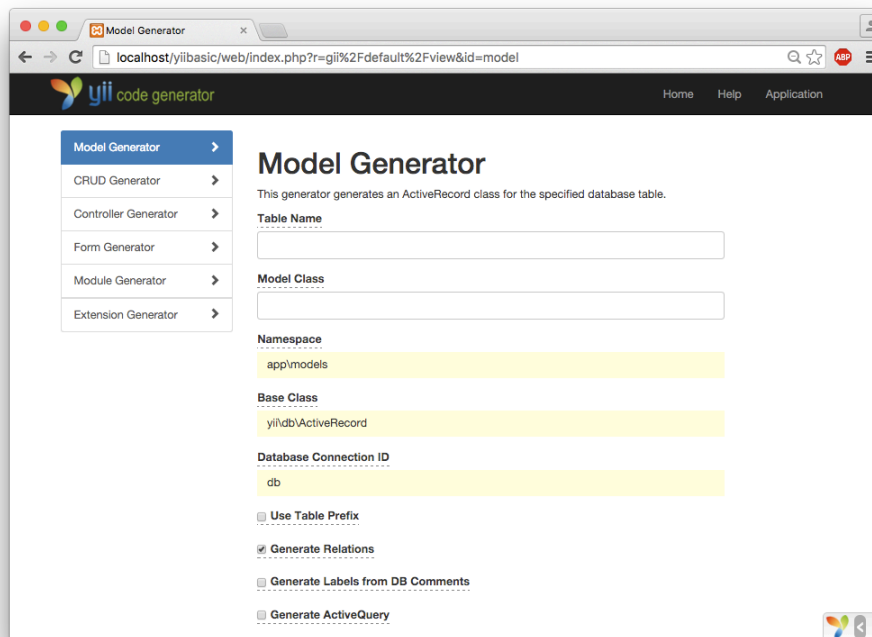
Gambar 3.6
Halaman Depan Gii

D. CRUD dengan Gii

Agar lebih akrab dengan Gii, kita akan membuat contoh CRUD pada tabel “artikel” dan “kategori” yang telah kita buat. Sesuai dengan tulisan saya di atas, disini kita akan membuat CRUD dengan Yii 2 Basic dengan nama project *yiibasic*.

Pastikan Anda telah mengikuti langkah pembuatan tabel dan konfigurasi database pada pembahasan di atas, kemudian buka browser dan buka link <http://localhost/yiibase/web/index.php?r=gii>.

Pertama kita generate Model terlebih dahulu. Jika link di atas telah di buka, lihat pada browser akan terdapat menu “Model Generator” kemudian klik “Start” maka akan muncul tampilan seperti berikut.

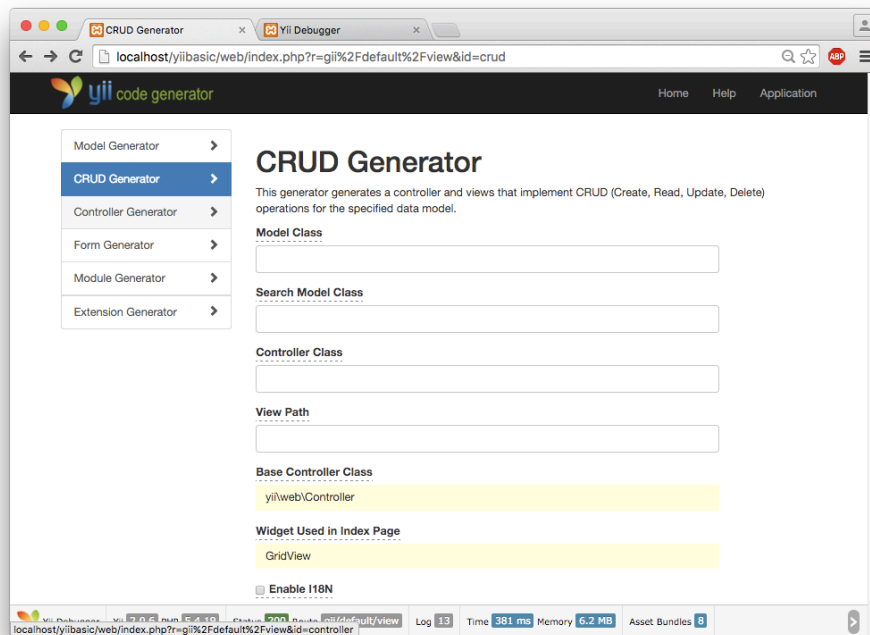


Gambar 3.7
Tampilan Model Generator pada Gii

Terdapat dua kolom utama, yaitu “Table Name” dan “Model Class”. Jika konfigurasi database telah benar, ketika anda mengetik nama tabel pada kolom “Table” maka akan secara otomatis muncul autocomplete yang berisi nama tabel tersebut.

Silahkan anda masukkan nama tabel “*kategori*” pada kolom *Table Name*, kemudian klik atau pilih kolom *Class Name* maka akan terisi secara otomatis dengan nama “*Kategori*” dan tekan Preview. Setelah itu akan muncul tombol Generate kemudian klik. Jika muncul pesan “*There was something wrong when generating the code. Please check the following messages.*” Ini berarti proses generate gagal, silahkan Anda ubah permission pada folder models. Jika sudah, kemudian lakukan ulang langkah di atas. Lakukan hal yang sama juga untuk tabel “artikel” dan tabel “komentar”. Jika proses generate berhasil maka akan muncul file baru pada folder models dengan nama *Kategori.php* dan *Artikel.php*

Setelah proses generate Model selesai, selanjutnya kita akan generate CRUD. Silahkan Anda buka menu CRUD Generator pada Gii maka akan muncul tampilan seperti berikut.



Gambar 3.8

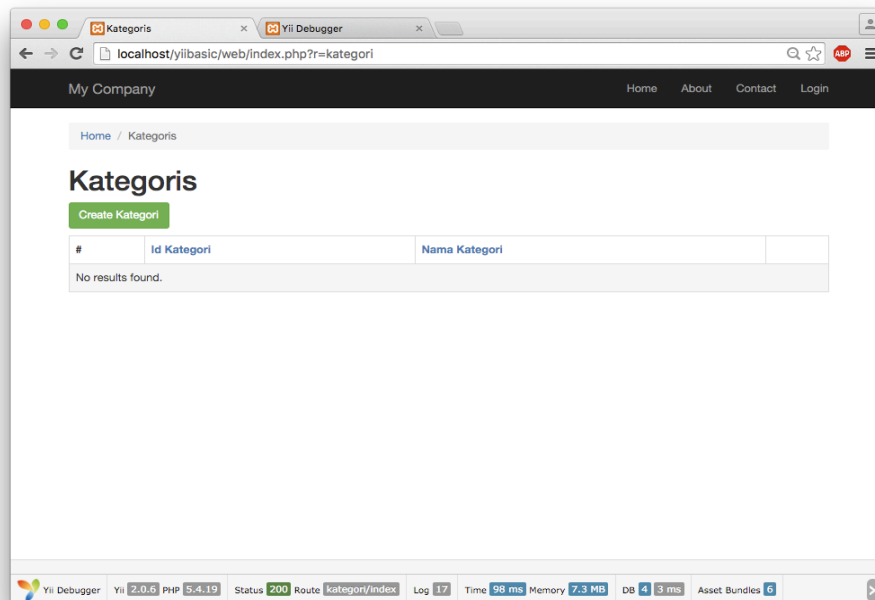
Tampilan CRUD Generator pada Gii

Pada CRUD Generator terdapat empat kolom utama, namun yang diwajibkan untuk di tambahkan adalah kolom “Model Class” dan “Controller Class”. Untuk mengisi nama model dan nama controller ini sedikit berbeda dengan mengisi nama tabel dan model pada Model Generator yaitu Anda harus memasukkan namespace.

Sebagai contoh, pertama kita akan generate CRUD untuk model “Kategori”. Silahkan input namespace model kategori pada kolom “Model Class” seperti berikut *“app/models/Kategori”* dan kemudian input namespace controller pada kolom “Controller Class” seperti berikut *“app/contollers/KategoriController”* kemudian klik Preview. Jika muncul kembali pesan *“There was something wrong when generating the code.*

Please check the following messages.” Ini berarti proses generate gagal. Silahkan ubah permission pada folder “*controllers*” dan folder “*views*” kemudian ulangi langkah CRUD Controller di atas. Jika proses generate berhasil maka ulang kembali CRUD Controller untuk model Artikel dan Komentar.

Setelah berhasil melakukan generate Model dan CRUD, untuk mencobanya silahkan anda buka link <http://localhost/yiibasic/web/index.php?r=kategori> jika tidak terjadi kesalahan saat proses generate di atas, maka akan muncul tampilan berikut.



Gambar 3.9
Tampilan hasil CRUD Generator

Selamat, secara otomatis aplikasi Anda telah separuh selesai. Silahkan Anda coba input data dengan membuka menu “Create Kategori” dan input nama kategori sesuai keinginan Anda.

PART IV

TOPIK SPESIAL

A. Mengetahui Model

Model bertugas menggambarkan informasi atau data beserta aturan bisnisnya seperti validasi, relasi, tipe data dan lain sebagainya. Dalam Yii terdapat dua jenis model, yaitu model yang bertugas mengelola data dari database, dan yang kedua adalah model yang bertugas hanya sebagai pengatur bisnis untuk form.

Penggunaan model yang bertugas mengelola database didefinisikan dengan menurunkan class `\yii\db\ActiveRecord` sedangkan model yang hanya bertugas mengelola form didefinisikan dengan menurunkan class `\yii\base\Model`.

Sebagai contoh, model yang hanya bertugas mengelola form adalah seperti berikut.

```
<?php

namespace app\models;

use Yii;
use yii\base\Model;

class LoginForm extends Model
{
    public $username;
    public $password;
    public $rememberMe = true;

    private $_user = false;

    public function rules()
    {
        return [
            // username dan password tidak boleh kosong
            [['username', 'password'], 'required'],
            // rememberMe harus berupa boolean
            ['rememberMe', 'boolean'],
        ];
    }
}
```



```

        // username dan password menggunakan custom
        validation
        ['username', 'validateUsername'],
        ['password', 'validatePassword'],
    ];
}

public function validateUsername($attribute, $params)
{
    if (!$this->hasErrors()) {
        if ($this->username != 'admin') {
            $this->addError($attribute, 'Username yang
Anda masukan aslah.');
```

Diatas merupakan contoh Model yang digunakan untuk login. Kode tersebut juga tidak terhubung langsung ke database. Disana juga telah didefinisikan variable public “username”, “password”, dan “rememberMe”. Selain property, di model juga terdapat baris kode yang digunakan untuk melakukan validasi pada masing-masing variabel. Dimana validasinya adalah di atur oleh fungsi *rules()*.

Jika di perhatikan lebih jelas lagi, pada bagian validasi password, terdapat bentuk validasinya yang di buat sendiri, dimana jika password yang di input pada form dengan value selain “admin” maka akan muncul pesan error dan login gagal.

Sementara pada kasus lain, jika kita ingin menginput data ke database melalui form, maka buat Model yang mendefinisikan atau mewakili tabel yang akan kita kelola. Model ini harus di turunkan dari

\yii\db\ActiveRecord. Selain mengelola data, model ini juga dapat bertindak membuat validasi atau aturan bisnis lainnya. Berikut adalah contoh model yang digunakan untuk mengelola database.

```
<?php

namespace app\models;

use Yii;

/**
 * This is the model class for table "artikel".
 *
 * @property integer $id_artikel
 * @property string $judul
 * @property string $isi_artikel
 * @property integer $id_kategori
 * @property integer $create_by
 * @property string $create_time
 * @property integer $update_by
 * @property string $update_time
 *
 * @property Kategori $idKategori
 * @property User $createBy
 * @property User $updateBy
 * @property Kategori $idKategori0
 */
class Artikel extends \yii\db\ActiveRecord
{
    /**
     * @inheritdoc
     */
    public static function tableName()
    {
        return 'artikel';
    }

    /**
     * @inheritdoc
     */
    public function rules()
    {
        return [
            [['judul', 'isi_artikel', 'id_kategori',
'create_by', 'create_time', 'update_by', 'update_time'],
'required'],
            [['isi_artikel'], 'string'],
            [['id_kategori', 'create_by', 'update_by'],
'integer'],
            [['judul'], 'string', 'max' => 255],
            [['create_time', 'update_time'], 'string', 'max'
=> 10]
        ];
    }
}
```

```

/**
 * @inheritdoc
 */
public function attributeLabels()
{
    return [
        'id_artikel' => 'Id Artikel',
        'judul' => 'Judul',
        'isi_artikel' => 'Isi Artikel',
        'id_kategori' => 'Id Kategori',
        'create_by' => 'Create By',
        'create_time' => 'Create Time',
        'update_by' => 'Update By',
        'update_time' => 'Update Time',
    ];
}

/**
 * @return \yii\db\ActiveQuery
 */
public function getIdKategori()
{
    return $this->hasOne(Kategori::className(), [
        'id_kategori' => 'id_kategori'
    ]);
}

/**
 * @return \yii\db\ActiveQuery
 */
public function getCreateBy()
{
    return $this->hasOne(User::className(), [
        'id' => 'create_by'
    ]);
}

/**
 * @return \yii\db\ActiveQuery
 */
public function getUpdateBy()
{
    return $this->hasOne(User::className(), [
        'id' => 'update_by'
    ]);
}

/**
 * @return \yii\db\ActiveQuery
 */
public function getIdKategori0()
{
    return $this->hasOne(Kategori::className(), [
        'id_kategori' => 'id_kategori'
    ]);
}

```

```
}  
}
```

Di atas merupakan contoh model yang mewakili tabel “artikel”. Tabel tersebut memiliki tiga field yaitu *id_artikel*, *judul*, *isi_artikel* dan lainnya. Pada bagian rules, penggunaannya sama dengan Model yang diturunkan dari `\yii\base\Model` yang telah kita bahas di atas.

Pada fungsi *attributeLabels()* digunakan untuk mendefinisikan label dari tiap variabel atau property yang bertugas mewakili field pada tabel. Sebab pada Yii, pembuatan form juga tidak dibuat secara manual atau dengan mengetikkan kode html. Melainkan telah disediakan class kusus yang akan mengelola penggunaannya.

Untuk cara penggunaan atau pengelolaan model ActiveRecord, kita akan bahas tepat di bawah ini.

B. Bekerja dengan Database

1). Konfigurasi

Yii 2 hadir dengan akses database dengan menggunakan driver PDO. Secara default Yii support untuk beberapa platform database, yaitu :

- MySQL
- MariaDB
- SQLite
- PostgreSQL
- CUBIRD (versi 9.1 ke atas)
- Oracle
- MSSQL (versi 2005 ke atas)

Untuk dapat melakukan konfigurasi database pada Yii 2, Anda dapat membuka file *config/web.php* pada Yii 2 Basic Template dan *common/config/main-local.php* pada Yii 2 Advanced Template.

```

<?php

return [
    'class' => 'yii\db\Connection',

    'dsn' => 'mysql:host=localhost;dbname=mydatabase',
    //'dsn' => 'sqlite:/path/to/database/file',
    //'dsn' => 'pgsql:host=localhost;port=5432;dbname=
        mydatabase',
    //'dsn' => 'cubird:dbname=mydatabase;host=localhost;
        port=33000',
    //'dsn' => 'sqlsrv:Server=localhost;Database=
        mydatabase',
    //'dsn' => 'dblib:host=localhost;dbname=mydatabase',
    //'dsn' => 'mssql:host=localhost;dbname=mydatabase',
    //'dsn' => 'oci:dbname=//localhost:1521/mydatabase',

    'username' => 'root',
    'password' => '',
    'charset' => 'utf8',
];

```

Terlihat pada gambar di atas bagaimana cara penggunaan atau konfigurasi berbagai database yang di dukung oleh Yii. Anda tidak harus menuliskan semua jenis *dsn* seperti di atas, melainkan memilih salah satu sesuai dengan database yang digunakan.

Selain dengan cara di atas, Anda juga dapat melakukan konfigurasi atau koneksi database langsung seperti berikut.

```

<?php

$connection = new \yii\db\Connection([
    'dsn' =>
    'mysql:host=localhost;dbname=mydatabase',
    'username' => 'root',
    'password' => ''
]);

$connection->open();

```

2). Data Access Object (DAO)

Yii memberikan Anda cara untuk melakukan query secara tradisional. Hal ini dapat dilakukan dengan adanya DAO. DAO sendiri dibangun atas PHP Data Objects (PDO), ini berarti Anda harus

memastikan extension PDO untuk database tertentu telah terinstal. Berikut adalah contoh penggunaannya.

SELECT

Jika ingin mengambil banyak record

```
$connection = Yii::$app->db;
$command = $connection->createCommand('SELECT * FROM
artikel');
$artikel = $command->queryAll();
```

Jika ingin mengambil satu record

```
$connection = Yii::$app->db;
$command = $connection->createCommand('SELECT * FROM
artikel WHERE id_artikel=1');
$artikel = $command->queryOne();
```

Jika ingin mengambil banyak record dengan column yang sama

```
$connection = Yii::$app->db;
$command = $connection->createCommand('SELECT judul FROM
artikel');
$judul = $command->queryColumn();
```

Jika ingin mengambil jumlah record

```
$connection = Yii::$app->db;
$command = $connection->createCommand('SELECT count(*)
FROM artikel');
$artikelCount = $command->queryScalar();
```

INSERT, UPDATE, DELETE dan lain-lain

Jika tujuan anda untuk mengeksekusi suatu query tanpa ingin mengambil datanya, seperti proses insert, update atau delete, berikut adalah caranya.

```
$connection = Yii::$app->db;
$command = $connection->createCommand('UPDATE artikel
SET id_kategori=2 WHERE id_artikel=1');
$artikel = $command->execute();
```

ALTERNATIVE SIMPLE

Berikut adalah alternative untuk berbagai kegiatan yang mungkin akan Anda temui suatu waktu.

```

$connection = Yii::$app->db;

// Insert
$connection->createCommand()->insert('user', [
    'nama' => 'Dida Nurwanda',
    'umur' => 22
])->execute();

// Multi Insert
$connection->createCommand()->insert('user', ['nama',
'umur'], [
    ['Dida Nurwanda', 22],
    ['Steve Jobs', 54]
])->execute();

// Update
$connection->createCommand()->update('user', ['umur' =>
23], 'id_user=1')->execute();

// Delete
$connection->createCommand()->delete('user', 'umur >
24')->execute();

```

3). Query Builder dan Query

Penggunaan query tradisional mungkin bisa saja di anggap membosankan dan rawan kesalahan dalam penulisannya, maka dari itu Yii menyediakan alternative dalam penggunaan query, yaitu dengan bantuan Query Builder. Seperti DAO, Query Builder di bangun di atas PDO. Perbedaanya antara DAO dengan Query Builder adalah cara mendefinisikan query yang berbeda. Dimana DAO mendefinisikan suatu sintaks SQL secara langsung dalam satu baris query, sedangkan dengan Query Builder kita mendefinisikan sintaks SQL secara procedural menggunakan property dan method yang tersedia. Berikut adalah contoh penggunaannya.

```

<?php

$rows = (new \yii\db\Query())
    ->select('id, nama')
    ->from('user')
    ->limit(10)
    ->all();

```

Pada contoh di atas, itu berarti Anda mengambil id dan nama dari tabel user dengan jumlah record 10 dan di ambil semua. Sebenarnya, Query Builder tidak hanya menyediakan metode pengembalian semua, melainkan terdapat beberapa metode pengembalian data, yaitu sebagai berikut.

- **yii\db\Query::all()** : *Membangun query, mengeksekusinya dan mengembalikan semua hasil sebagai array.*
- **yii\db\Query::one()** : *Mengembalikan baris pertama pada hasilnya.*
- **yii\db\Query::column()** : *Mengembalikan kolom pertama pada hasilnya.*
- **yii\db\Query::scalar()** : *Mengembalikan kolom pertama pada baris pertama.*
- **yii\db\Query::Exists()** : *Mengembalikan nilai yang menunjukkan apakah hasil query pada apapun.*
- **yii\db\Query::Count()** : *Mengembalikan hasil count. Metode yang serupa lainnya termasuk SUM(\$q), AVERAGE(\$q), MAX(\$q), MIN(\$q).*

SELECT

Berikut beberapa contoh penggunaan SELECT dengan Query Builder.

```
<?php
$query = (new \yii\db\Query())
    ->select('id, nama')
    ->from('user');

// select dapat juga di tulis dengan array
$query = (new \yii\db\Query())
    ->select(['id', 'nama'])
    ->from('user');

// penggunaan distinct()
$query = (new \yii\db\Query())
    ->select('*')
    ->distinct()
    ->from('user');
```


FROM

Berikut beberapa contoh penggunaan FROM pada Query Builder.

```
<?php

$query = (new \yii\db\Query())
    ->select('*')
    ->from('user');

// penggunaan prefix
$query = (new \yii\db\Query())
    ->select('u.*, a.*')
    ->from(['user u', 'artikel a']);

// sub-query
$subQuery = (new \yii\db\Query())->select('id')
    ->from('user')->where('status=1');
$query = (new \yii\db\Query())->select('*')
    ->from(['u' => $subQuery]);
```

WHERE

Berikut beberapa contoh penggunaan WHERE pada Query Builder.

```
// binder
$query->where('status=:status', [':status' => $status]);

// RAWAN !! Tidak disarankan
$query->where("status=$status");

// Menggunakan params terpisah
$query->where('status=:status');
$query->addParams([':status' => $status]);

// Multiple
$query->where([
    'status'    => 10,
    'type'      => 2,
    'id'        => [4, 5, 6, 9]
]);

// NULL
$query->where(['status' => NULL]);

// sub-query
$subQuery = (new \yii\db\Query())->select('id')-
>from('user');
$query->where(['id' => $subQuery]);

// like, or like, not like, or not like, exists, not
exists
$query->where(['status' => $status]);
if (!empty($search)) {
    $query->addWhere('like', 'name', $search);
}
```

ORDER BY, GROUP BY, HAVING, LIMIT dan OFFSET

Berikut beberapa contoh penggunaan pada Query Builder.

```
// Order By
$query->orderBy([
    'id' => SORT_ASC,
    'name' => SORT_DESC
]);

// Group By
$query->groupBy('id, status');

// Menambahkan Group By jika sebelumnya telah
ditambahkan
$query->addGroupBy(['created_at', 'updated_at']);

// Having
$query->having(['status' => $status]);

// limit
$query->limit(10);

// offset
$query->offset(100);
```

JOIN dan UNION

Berikut adalah beberapa contoh penggunaan JOIN dan UNION pada Query Builder.

```
// Left Join
$query->select(['user.nama as author', 'artikel.judul as
title'])
->from('user')
->leftJoin('artikel', 'artikel.user_id = user.id');

// Join
$query->join('FULL OUTER JOIN', 'artikel',
'artikel.user_id = user.id');

// Sub-Query
$query->leftJoin(['u' => $subQuery, 'u.id=another_id']);

// Union
$query = (new \yii\db\Query())
->select('id, category_id as type, name')
->from('post')
->limit(10);
$anotherQuery = (new \yii\db\Query())
->select('id, type, name')
->from('user')
->limit(10);
$query->union($anotherQuery);
```

4). Active Record (AR)

Active Record (AR) merupakan teknik populer dari Object-Relational Mapping (ORM). Setiap kelas AR akan mewakili satu tabel pada database dan propertynya mewakili attribute dari tabel tersebut. Pada Yii, AR merupakan model yang didefinisikan agar suatu class dapat terkoneksi dengan database. Jadi jika ingin menggunakan AR, maka kita harus mendefinisikan model menggunakan AR. Dengan kata lain, AR bisa disebut juga model.

Penggunaan AR sangat membantu mengurangi waktu dalam penulisan sintaks-sintaks SQL. Berikut contoh penggunaan AR pada kasus menyimpan data.

```
$kategori = new \app\models\Kategori();
$kategori->nama_kategori = 'Yii Framework';
$kategori->save();
```

Selain penggunaannya mempermudah dalam penyimpanan data, AR juga sangat mempermudah dalam pengambilan data. Berikut beberapa contoh pengambilan data menggunakan AR.

```
// mengambil seluruh record
$post = \app\models\Artikel::find()->all();

// mengambil satu record
$post = \app\models\Artikel::find()->where(['id' => 1])
->one();

// mengambil jumlah record
$count = \app\models\Artikel::find()->count();

// mengambil satu record berdasarkan primary key
$post = \app\models\Artikel::findOne(1);

// mengambil satu record berdasarkan kriteria
$post = \app\models\Artikel::findOne([
    'create_by' => 1
]);

// mengambil banyak record berdasarkan primary key
$post = \app\models\Artikel::findAll([1, 2, 4, 7]);

// mengambil banyak record berdasarkan kriteria
$post = \app\models\Artikel::findAll([
    'create_by' => 2
]);
```

Manipulasi Data pada Database

Manipulasi data pada database dengan menggunakan AR sangatlah mudah. Berbagai kegiatan seperti menambahkan data baru, mengubah data, menghapus data dapat dilakukan tanpa menyentuh sintaks SQL secara langsung. Berikut adalah metode yang disediakan oleh AR untuk mengolah satu baris record :

- yii\db\ActiveRecord::save()
- yii\db\ActiveRecord::insert()
- yii\db\ActiveRecord::update()
- yii\db\ActiveRecord::delete()

AR juga menyediakan metode statis yang berlaku untuk seluruh tabel yang terkait dengan kelas AR. Berhati-hatilah dalam menggunakan metode ini karena dapat mempengaruhi seluruh tabel. Misal deleteAll() akan menghapus semua baris dalam tabel.

- yii\db\ActiveRecord::updateCounters()
- yii\db\ActiveRecord::updateAll()
- yii\db\ActiveRecord::updateAllCounters()
- yii\db\ActiveRecord::deleteAll()

```
// insert data baru pada tabel customer
$customer = new \app\models\Customer();
$customer->name = 'Dida Nurwanda';
$customer->email = 'didanurwanda@gmail.com';
$customer->save();

// update satu data pada tabel customer
$customer = \app\models\Customer::findOne($id);
$customer->email = 'dida_n@ymail.com';
$customer->save();

// delete satu data pada tabel customer
$customer = \app\models\Customer::findOne($id);
$customer->delete();

// delete banyak data berdasarkan berdasarkan kriteria
\app\models\Customer::deleteAll('age > :age AND gender =
:gender', [
    ':age' => 24,
    ':gender' => 'M'
]);
```

Bekerja dengan Relasi Data

Anda dapat menggunakan AR untuk melakukan query relasi antar table. Berterima kasihlah kepada AR, sebab penggunaan data relasi dengan menggunakan AR benar-benar dipermudah. Secara default, Yii akan secara otomatis mendefinisikan teknik atau kode untuk relasi antar table dengan AR, dengan catatan pada saat pembuatan atau perancangan database anda telah melakukan relasi terlebih dahulu.

```
class Artikel extends \yii\db\ActiveRecord
{
    public static function tableName()
    {
        return 'artikel';
    }

    public function getUser()
    {
        // Artikel has-one User dengan user.id =
        // artikel.create_by
        return $this->hasOne(User::className(), [
            'id' => 'create_by'
        ]);
    }
}

class User extends \yii\db\ActiveRecord
{
    public static function tableName()
    {
        return 'user';
    }

    public function getArtikels()
    {
        // User has-many Artikel dengan user.id =
        // artikel.create_by
        return $this->hasMany(Artikel::className(), [
            'create_by' => 'id'
        ]);
    }
}
```

Dengan menggunakan script kedua model seperti diatas, anda dapat mengambil data user mana yang membuat artikel, dan anda juga

bisa mencari tahu artikel mana saja yang di buat oleh user. Berikut contohnya.

```
// mengembalikan record dari model User
// mengambil data user mana yang membuat artikel dengan
id_artikel = 1
$pebuat_artikel = \app\models\Artikel::findOne(1)
                ->user;

// mengembalikan record dari model Artikel
// mengambil data artikel mana yang dibuat oleh user
dengan id = 1
$daftar_artikel = \app\models\User::findOne(1)->user;
```

C. Data Formatter

Untuk format output, Yii telah menyediakan class formatter yang berfungsi untuk membuat data lebih mudah dibaca oleh pengguna. `\yii\i18n\Formatter` adalah kelas pembantu yang terdaftar sebagai komponen aplikasi. Ini menyediakan satu set metode untuk daa format tujuan, seperti tanggal, nilai waktu, angka dan format lain yang umum digunakan. Formatter dapat digunakan dengan dua cara yang berbeda, berikut adalah cotoh penggunaan formatter dengan method yang telah di definisikan.

```
// untuk memformat tanggal
// short, medium, long, full
echo Yii::$app->formatter->asDate('2015-11-23', 'long');

// untuk memformat time
// short, medium, long, full
echo Yii::$app->formatter->asTime(time(), 'long');
echo Yii::$app->formatter->asDatetime(time(), 'long');

// untuk memformat angka desimal menjadi persen
// angka 2 pada contoh merupakan jumlah desimal
echo Yii::$app->formatter->asPercent(0.80, 2);

// untuk memformat email menjadi mailto(link)
echo Yii::$app->formatter->asEmail('didanurwanda@gmail.com');

// untuk memformat boolean
echo Yii::$app->formatter->asBoolean(true);

// untuk memformat text
echo Yii::$app->formatter->asRaw($value);
echo Yii::$app->formatter->asText($value);
echo Yii::$app->formatter->asNtext($value);
echo Yii::$app->formatter->asParagraphs($value);
echo Yii::$app->formatter->asHtml($value);
```

```
echo Yii::$app->formatter->asImage($path, $options);  
echo Yii::$app->formatter->asUrl($link, $options);
```

Anda juga dapat menggunakan formatter dengan format penulisan seperti berikut.

```
echo Yii::$app->formatter->format('2015-11-23', 'date');  
echo Yii::$app->formatter->format('0.75', 'percent');  
echo Yii::$app->formatter->format(time(), 'date');  
echo Yii::$app->formatter->format(time(), 'time');  
echo Yii::$app->formatter->format(time(), 'datetime');  
echo Yii::$app->formatter->format('didanurwanda@gmail.com',  
'email');  
echo Yii::$app->formatter->format($value, 'raw');  
echo Yii::$app->formatter->format($value, 'text');  
echo Yii::$app->formatter->format($value, 'ntext');  
echo Yii::$app->formatter->format($value, 'paragraphs');  
echo Yii::$app->formatter->format($value, 'html');  
echo Yii::$app->formatter->format($path, 'image');  
echo Yii::$app->formatter->format($link, 'url');
```

Anda juga dapat menambahkan terjemahan atau output hasil format sesuai dengan bahasa tertentu.

```
echo Yii::$app->formatter->locale = 'en-US';
```

PART V

MEMBUAT BLOG SEDERHANA

Setelah berkuat dengan materi-materi di atas, mungkin Anda juga masih merasa bingung karena penjelasannya terlalu berbelit dan karena memang saya sendiri masih belajar sebagai penulis.

Langsung saja, pada pembuatan aplikasi sederhana ini kita akan menggunakan database yang telah di buat sebelumnya yang di bahas di Part III. Jika Anda belum membuatnya, silahkan baca kembali Part III dan praktekan. Aplikasi ini juga dibangun berdasarkan Yii 2 Advanced.

Jika database telah dibuat, selanjutnya Anda instal Yii 2 Advanced Template dengan nama “yiiblog”. Cara instal telah di bahas di atas. Kemudian ubah koneksinya dan sesuaikan dengan database yang telah Anda buat. Untuk melakukan konfigurasi database, lokasi file konfigurasinya berada pada direktori *common/config/main-local.php*. Jika Anda masih bingung, berikut merupakan konfigurasi yang saya pakai. Disana terdapat pula tambahan baris kode untuk mengubah url agar lebih mudah di baca.

```
<?php
return [
    'name' => 'Yii Blog !',
    'components' => [
        'urlManager' => [
            'enablePrettyUrl' => true
        ],
        'db' => [
            'class' => 'yii\db\Connection',
            'dsn' => 'mysql:host=localhost;dbname=yiiblog',
            'username' => 'root',
            'password' => '',
            'charset' => 'utf8',
        ],
        'mailer' => [
            'class' => 'yii\swiftmailer\Mailer',
            'viewPath' => '@common/mail',
            'useFileTransport' => true,
        ],
    ],
];
```



```

    ],
    ],
];

```

Jika konfigurasi database telah selesai, kemudaian kita akan membuat CRUD dengan bantuan Gii. Oleh karena itu, silahkan Anda buka browser kesayangan Anda, kemudian ketikkan alamat <http://localhost/yiiblog/backend/web/index.php/gii>. Untuk pertama yang dilakukan, kita akan generate Model terlebih dahulu. Karena Model yang akan kita gunakan dipakai di backend dan frontend, maka untuk memudahkan Model akan di simpan di direktori *common/models*.

Buka menu Model Generator, kemudian ketikkan perintah-perintah berikut pada kolom yang telah tersedia.

Table Name	Class Name	Namespace
artikel	Artikel	common/models/Artikel
kategori	Kategori	common/models/Kategori
komentar	Komentar	common/models/Komentar

Pada tabel di atas, mungkin Anda bertanya-tanya, kenapa tidak ada tabel user. Jika Anda perhatikan struktur file dan folder pada Yii 2, sebenarnya telah tersimpan model User secara default. File model tersebut terdapat pada directory *common/models/User.php* untuk Yii 2 Advanced, dan *models/User.php* untuk Yii 2 Basic. Disini kita akan memanfaatkan model yang telah ada tersebut, dan pada saat pembuatan tabel, kita membuat tabel sesuai dengan isi dari tabel User, namun sedikit ditambah field yaitu *first_name* dan *last_name*, maka dari itu kita akan sedikit memodifikasi model User tersebut.

Jika proses generate model telah selaesai, selanjutnya kita akan menggenerate CRUD. Buka menu CRUD Generator, kemudian lakukan generator

CRUD untuk model Artikel, Kategori, dan Komentar. Ketikkan perintah berikut pada kolom-kolom yang tersedia pada CRUD generator.

Model Class	Controller Class
common/models/Artikel	backend/controllers/ArtikelController
common/models/Kategori	backend/controllers/KategoriController
common/models/Komentar	backend/controllers/KomentarController
common/models/User	backend/controllers/UserController

Pastikan semua proses generate telah berhasil, jika gagal ada kemungkinan Anda salah dalam mengetik atau bisa juga direktori *common/models*, *backend/controllers* dan *backend/views* belum di ubah hak aksesnya.

Jika semua proses diatas telah selesai, hal yang pertama kita ubah adalah mengubah tampilan halaman web frontend. Silahkan Anda buka file *frontend/views/layouts/main.php* dan ubah seperti berikut.

```
<?php

use yii\helpers\Html;
use yii\bootstrap\Nav;
use yii\bootstrap\NavBar;
use frontend\assets\AppAsset;

AppAsset::register($this);
?>
<?php $this->beginPage() ?>
<!DOCTYPE html>
<html lang="<? = Yii::$app->language ?>">
<head>
    <meta charset="<? = Yii::$app->charset ?>">
    <meta name="viewport" content="width=device-width, initial-
scale=1">
    <? = Html::csrfMetaTags() ?>
    <title><? = Html::encode($this->title) ?></title>
    <?php $this->head() ?>

    <style type="text/css">
        body {
            background: #f0f0f0;
```

```

    }

    .content {
        margin-bottom: 30px;
        width: 100%;
        background: #fbfbfb;
        border-radius: 5px;
        padding: 10px;
    }

    .content:hover {
        background: #f5f5f5;
    }

    .content-title a {
        font-size: 18px;
        font-color: #333;
        width: 100%;
        border-bottom: 1px dotted #ccc;
    }

    .content-detail {
        font-size: 10px;
        width: 100%;
        color: blue;
        margin-bottom: 10px;
    }

    .content-fill {
        width: 100%;
        font-size: 12px;
    }
</style>
</head>
<body>
<?php $this->beginBody() ?>

<div class="wrap">
    <?php NavBar::begin([
        'brandLabel' => Yii::$app->name,
        'brandUrl' => Yii::$app->homeUrl,
        'options' => [
            'class' => 'navbar-default navbar-fixed-top',
        ],
    ]);

    echo Nav::widget([
        'options' => ['class' => 'navbar-nav navbar-right'],
        'items' => [
            ['label' => 'Home', 'url' => ['/site/index']],

            // ['label' => 'Kategori', 'items' =>
common\models\Kategori::getKategoriMenu()],

            ['label' => 'Signup', 'url' => ['/site/signup']],

```

```

'visible' => Yii::$app->user->isGuest],
    ['label' => 'Login', 'url' => ['/site/login']],
'visible' => Yii::$app->user->isGuest],
    [
        'label' => Logout,
        'url' => ['/site/logout'],
        'visible' => !Yii::$app->user->isGuest,
        'linkOptions' => ['data-method' => 'post']
    ]
    ],
]);
NavBar::end();
?>

<div class="container-fluid" style="background: #fff; width:
960px; margin-top: 50px">

    <div class="row-fluid">
        <div class="col-md-12">
            <div class="jumbotron" style="background: #f0f0f0;
margin-top: 20px; padding-top: 10px; padding-bottom: 10px">
                <h1>Yii Blog!</h1>
                <p class="lead">Ini merupakan contoh aplikasi
yang sangat sederhana, dibuat dengan menggunakan Yii Framework 2
Advanced Template.</p>
                <p><a class="btn btn-lg btn-success"
href="http://www.didanurwanda.com">http://blog.didanurwanda.com</a
></p>
            </div>
        </div>
    </div>

    <div class="row-fluid">
        <div class="col-md-8">
            <?= $content ?>
        </div>

        <div class="col-md-4">
            <div class="panel panel-default">
                <div class="panel-heading">Top Artikel</div>
                <div class="panel-body">
                    <ul>
                        <?php
                        /*

                        <?php
foreach(common\models\Artikel::topArtikel() as $row): ?>
                            <li><?= Html::a($row->judul .'
('.$row->jumlah_baca.'), ['view', 'id' => $row->id_artikel])
?></li>

                            <?php endforeach; ?>
                        </ul>

                        */
                        ?>
                    </div>
                </div>
            </div>
        </div>
    </div>

```

```

        </div>
        <div class="panel panel-default">
            <div class="panel-heading">Komentar
Terbanyak</div>
            <div class="panel-body">
                <ul>
                    <?php
                        /*

                            <?php
                                foreach(common\models\Artikel::topKomentar() as $row): ?>
                                    <li><?= Html::a($row->judul .' ('.
                                        count($row->komentars).')', ['view', 'id' => $row->id_artikel])
                                        ?></li>
                                    <?php endforeach; ?>
                                */

                            ?>
                        </ul>
                    </div>
                </div>
            </div>

        </div>
    </div>
</div>
</div>

<footer class="footer">
    <div class="container">
        <p class="pull-left">
            <?= Yii::$app->name ?> &copy; <?= date('Y') ?> Dida
Nurwanda
            &nbsp; | &nbsp;
            <?= Html::a('www.didanurwanda.com',
                'http://www.didanurwanda.com') ?>
        </p>
        <p class="pull-right"><?= Yii::powered() ?></p>
    </div>
</footer>

<?php $this->endBody() ?>
</body>
</html>
<?php $this->endPage() ?>

```

Anda pasti berfikir kalo kode di atas banyak sekali, wajar saja sebab kode diatas merupakan template dari aplikasi yang kita buat. Selanjutnya kita akan membuat atau mengaktifkan fitur signup, ini akan berfungsi untuk Anda melakukan registrasi ke aplikasi dan dapat digunakan untuk login. Pertama kita akan membuat tampilan signup terlebih dahulu. Sebenarnya proses signup telah

disediakan oleh Yii secara default, kita hanya memodifikasinya saja. Silahkan Anda buka file *frontend/views/site/signup.php* dan ubah menjadi seperti berikut.

```
<?php

use yii\helpers\Html;
use yii\bootstrap\ActiveForm;

$this->title = 'Signup';
$this->params['breadcrumbs'][] = $this->title;
?>
<div class="site-signup">
    <div class="panel panel-info">
        <div class="panel-heading">Signup</div>
        <div class="panel-body">
            <p>Please fill out the following fields to signup:</p>
            <?php $form = ActiveForm::begin(['id' => 'form-
signup']); ?>

                <?= $form->field($model, 'first_name') ?>
                <?= $form->field($model, 'last_name') ?>
                <?= $form->field($model, 'username') ?>
                <?= $form->field($model, 'email') ?>
                <?= $form->field($model, 'password')-
                >passwordInput() ?>

                <div class="form-group">
                    <?= Html::submitButton('Signup', ['class' =>
'btn btn-primary', 'name' => 'signup-button']) ?>
                </div>

            <?php ActiveForm::end(); ?>
        </div>
    </div>
</div>
```

Sebenarnya jika Anda perhatikan, kita hanya menambahkan baris kode untuk *first_name* dan *last_name* saja. Jika Anda berfikir terjadi banyak perubahan, itu hanya untuk mengubah tampilan saja agar berbeda dari keadaan default. Selanjutnya buka file *frontend/models/SignupForm.php* dan tambahkan atau ubah baris kode berikut pada fungsi *signup()*

```
public function signup()
{
    if ($this->validate()) {
        $user = new User();

        $user->first_name = $this->first_name;
        $user->last_name = $this->last_name;
```

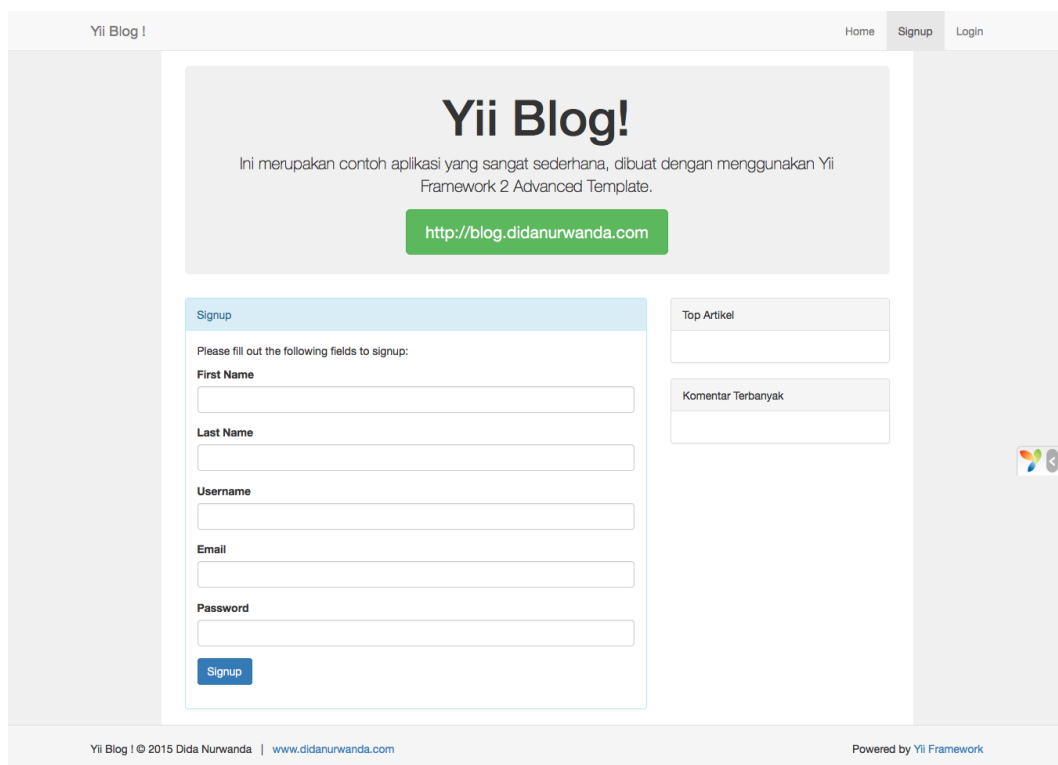
```

        $user->username = $this->username;
        $user->email = $this->email;
        $user->setPassword($this->password);
        $user->generateAuthKey();
        if ($user->save()) {
            return $user;
        }
    }

    return null;
}

```

Jika sudah, silahkan Anda buka browser dan masuk ke halaman <http://localhost/yiiblog/frontend/web> dan pilih menu signup, maka akan muncul tampilan seperti berikut.



Gambar 5.1

Halaman Signup pada frontend

Silahkan Anda isi form tersebut dan harap ingat *username* dan *passwordnya*. Jika proses signup telah berhasil, secara otomatis berarti Anda juga

telah melakukan login. Tanda ini terlihat jelas pada tombol Login yang berganti jadi Logout.

Tahap berikutnya kita akan mengelola halaman Admin, dimana disini kita hanya mengelola dua kegiatan, yaitu Artikel dan Kategori Artikel. Disini kita akan memodifikasi SiteController yang berada pada direktori *backend/controllers/SiteControllers.php* dan pada bagian fungsi *actionIndex()* ubah menjadi seperti berikut.

```
public function actionIndex()
{
    // return $this->render('index');
    return $this->redirect(['/artikel/index']);
}
```

Baris kode di atas hanya menambahkan atau mengganti fungsi render dengan redirect. Ini dimaksudkan agar ketika kita buka halaman *site/index* secara otomatis kita dialihkan ke halaman *artikel/index* dimana terdapat daftar artikel yang kita buat. Selanjutnya silahkan Anda buka *backend/controllers/ArtikelController.php* dan ubah seperti berikut.

```
<?php

namespace backend\controllers;

use Yii;
use common\models\Artikel;
use common\models\Kategori;
use yii\data\ActiveDataProvider;
use yii\web\Controller;
use yii\web\NotFoundHttpException;
use yii\filters\VerbFilter;
use yii\filters\AccessControl;

class ArtikelController extends Controller
{
    public function behaviors()
    {
        return [
            'access' => [
                'class' => AccessControl::className(),
                'rules' => [
                    [
                        'actions' => ['index', 'create', 'update',
'delete'],
                        'allow' => true,
                        'roles' => ['@'],

```



```

        ],
    ],
    'verbs' => [
        'class' => VerbFilter::className(),
        'actions' => [
            'delete' => ['post'],
        ],
    ],
];
}

public function actionIndex()
{
    $dataProvider = new ActiveDataProvider([
        'query' => Artikel::find(),
        'sort' => [
            'defaultOrder' => [
                'id_artikel' => SORT_DESC
            ]
        ],
    ]);
    return $this->render('index', [
        'dataProvider' => $dataProvider
    ]);
}

public function actionCreate()
{
    $model = new Artikel();

    if ($model->load(Yii::$app->request->post()) && $model->save()) {
        return $this->redirect(['index']);
    } else {
        return $this->render('create', [
            'model' => $model,
        ]);
    }
}

public function actionUpdate($id)
{
    $model = $this->findModel($id);

    if ($model->load(Yii::$app->request->post()) && $model->save()) {
        return $this->redirect(['index']);
    } else {
        return $this->render('update', [
            'model' => $model,
        ]);
    }
}

public function actionDelete($id)

```

```

    {
        $this->findModel($id)->delete();

        return $this->redirect(['index']);
    }

    protected function findModel($id)
    {
        if (($model = Artikel::findOne($id)) !== null) {
            return $model;
        } else {
            throw new NotFoundHttpException('The requested page
does not exist.');
```

Pada kode di atas, jika Anda perhatikan kita hanya sedikit mengubah atau menambahkan beberapa kode, seperti pada bagian *behaviors()* diatas kita menambahkan mode akses, ini bertujuan untuk melarang orang membuka halaman tersebut jika belum login. Dan pada *actionIndex()* kita juga hanya sedikit menambahkan fungsi sort agar artikel yang paling baru di buat muncul di atas. Pada *actionCreate()* dan *actionUpdate()* kita hanya mengubah alamat redirect, yaitu langsung kembali ke halaman *actionIndex()*.

Setelah itu, kita ubah halaman index untuk controller artikel, silahkan Anda buka file *backend/views/artikel/index.php* dan ubah menjadi seperti berikut.

```

<?php

use yii\helpers\Html;
use yii\grid\GridView;

$this->title = 'Artikel';
$this->params['breadcrumbs'][] = $this->title;
?>

<div class="artikel-index panel panel-info">

    <div class="panel-heading">
        <h4><?= Html::encode($this->title) ?>
        <span class="pull-right">
            <?= Html::a('Tambah Artikel', ['create'], ['class' =>
'btn btn-primary btn-sm']) ?>
            <?= Html::a('Kategori', ['/kategori'], ['class' =>
'btn btn-danger btn-sm']) ?>
        </span>
    </h4>
    </div>

```

```

<div class="panel-body">
    <?php GridView::widget([
        'dataProvider' => $dataProvider,
        'columns' => [
            ['class' => 'yii\grid\SerialColumn'],
            'judul',
            'idKategori.nama_kategori',
            'jumlah_baca',
            'create_time:date',
            [
                'class' => 'yii\grid\ActionColumn',
                'template' => '{update} {delete}'
            ],
        ],
    ]); ?>
</div>
</div>

```

Selanjutnya kita akan mengubah form agar lebih menarik, silahkan buka *backend/views/artikel/_form.php* dan ubah menjadi seperti berikut.

```

<?php

use yii\helpers\Html;
use yii\helpers\ArrayHelper;
use yii\widgets\ActiveForm;

?>

<div class="artikel-form">
    <div class="panel panel-info">
        <div class="panel-heading">
            <h4>
                <?php $this->title ?>
                <span class="pull-right">
                    <?php Html::a('Kembali', ['index'], ['class' =>
'btn btn-danger btn-sm']) ?>
                </span>
            </h4>
        </div>

        <div class="panel-body">
            <?php $form = ActiveForm::begin([
                'options' => [
                    'class' => 'col-md-5'
                ]
            ]); ?>
            <?php $form->field($model, 'judul')-
>textInput(['maxlength' => true]) ?>
            <?php $form->field($model, 'isi_artikel')-
>textarea(['rows' => 6]) ?>
            <?php $form->field($model, 'id_kategori')->dropDownList(
                ArrayHelper::map(

```

```

        common\models\Kategori::find()->all(),
        'id_kategori',
        'nama_kategori'
    )
) ?>
<div class="form-group">
    <?= Html::submitButton($model->isNewRecord ?
'Create' : 'Update', ['class' => $model->isNewRecord ? 'btn btn-
success' : 'btn btn-primary']) ?>
</div>
<?php ActiveForm::end(); ?>
</div>
</div>
</div>

```

Kemudian ubah juga file create yang berada pada direktori *backend/views/artikel/create.php* dan ikuti baris kode berikut.

```

<?php

use yii\helpers\Html;

$this->title = 'Tambah Artikel';
$this->params['breadcrumbs'][] = ['label' => 'Artikel', 'url' =>
['index']];
$this->params['breadcrumbs'][] = $this->title;
?>

<div class="artikel-create">
    <?= $this->render('_form', [
        'model' => $model,
    ]) ?>
</div>

```

Dan terakhir ubah file update yang berada pada direktori *backend/views/artikel/update.php* lalu ikuti kode berikut.

```

<?php

use yii\helpers\Html;

$this->title = 'Update Artikel';
$this->params['breadcrumbs'][] = ['label' => 'Artikel', 'url' =>
['index']];
$this->params['breadcrumbs'][] = ['label' => $model->judul, 'url'
=> ['index']];
$this->params['breadcrumbs'][] = 'Update';
?>

<div class="artikel-update">
    <?= $this->render('_form', [
        'model' => $model,
    ]) ?>
</div>

```

```
    ])?>  
</div>
```

Tahap terakhir adalah kita mengubah logika dari model artikel, yaitu yang berada pada direktori *common/models/Artikel.php*. Pada bagian *rules()* silahkan Anda ubah sesuai dengan kode berikut.

```
public function rules()  
{  
    return [  
        [['judul', 'isi_artikel', 'id_kategori'], 'required'],  
        [['isi_artikel'], 'string'],  
        [['id_kategori', 'jumlah_baca', 'create_by', 'update_by'],  
        'integer'],  
        [['judul'], 'string', 'max' => 255],  
        [['create_time', 'update_time'], 'string', 'max' => 10]  
    ];  
}
```

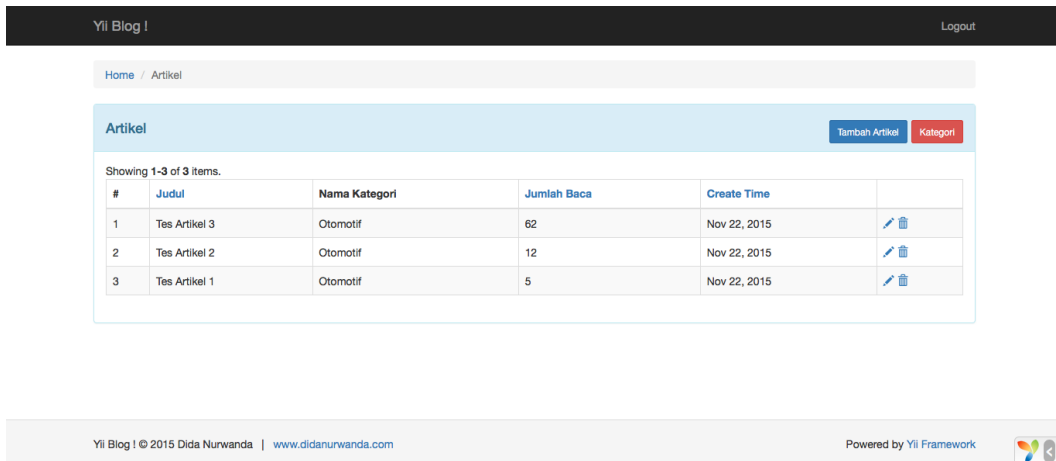
Kemudian tambahkan juga fungsi *beforeSave()* pada baris terakhir class. Ini bertujuan untuk membuat kondisi dimana kita melakukan beberapa tindakan sesaat sebelum proses penyimpanan data pada database.

```
public function beforeSave($insert)  
{  
    parent::beforeSave($insert);  
    if ($this->isNewRecord)  
    {  
        $this->jumlah_baca = 0;  
        $this->create_by = Yii::$app->user->id;  
        $this->update_by = Yii::$app->user->id;  
        $this->create_time = time();  
        $this->update_time = time();  
    }  
    else  
    {  
        $this->update_by = Yii::$app->user->id;  
        $this->update_time = time();  
    }  
    return true;  
}
```

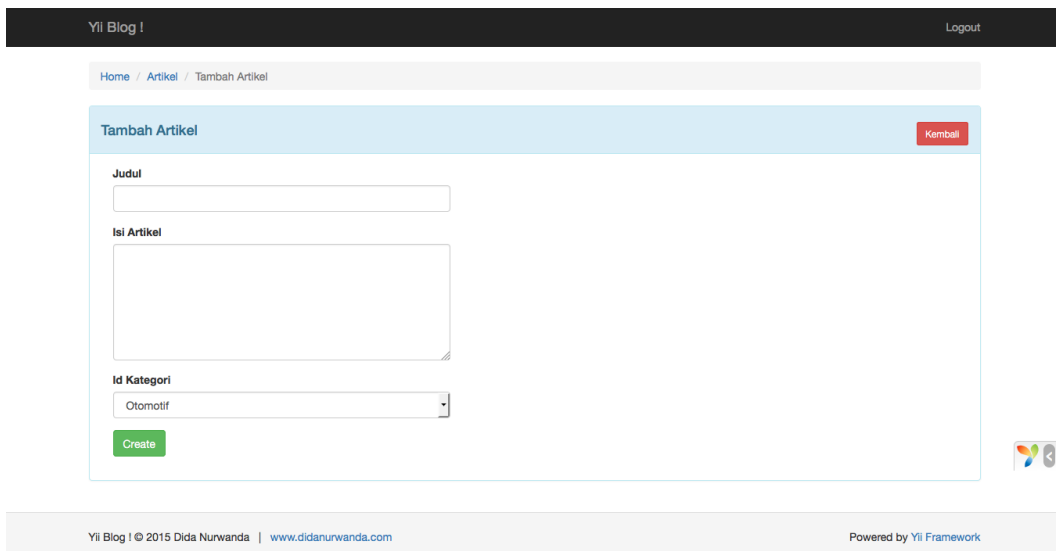
Untuk artikel kita telah selesai, jika Anda ingin mencobanya silahkan buka <http://localhost/yiiblog/backend/web> dan pastikan anda telah login, jika belum silahkan Anda login pada kolom yang telah disediakan. Anda masih belum bisa

menambahkan artikel, sebab artikel bisa dibuat jika ada kategori untuk artikel. Maka dari itu kita akan memodifikasi kategori yang telah dibuat tadi melalui Gii.

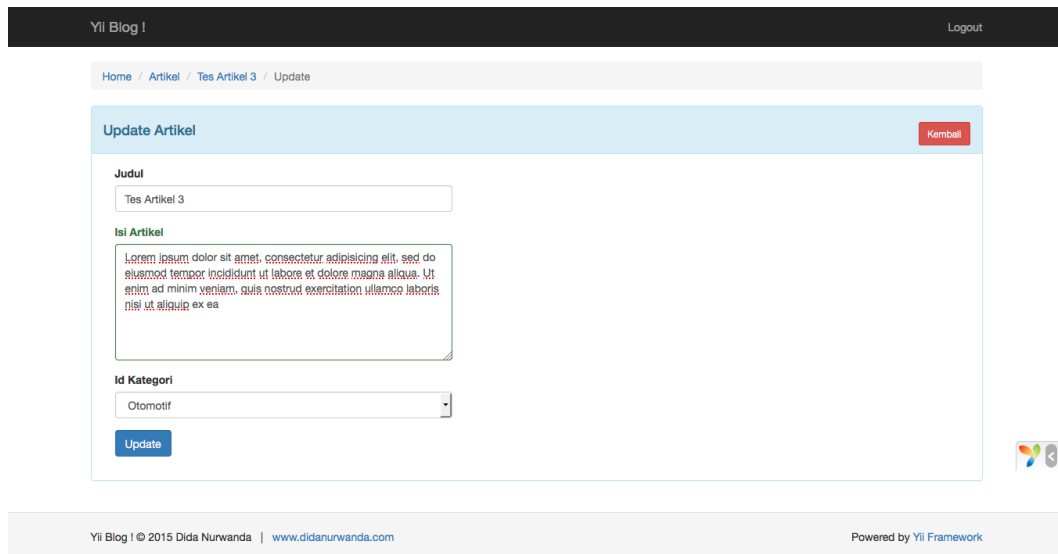
Namun berikut beberapa tampilan halaman backend untuk artikel jika proses modifikasi yang Anda lakukan telah berhasil.



Gambar 5.2
Halaman daftar artikel



Gambar 5.3
Halaman tambah artikel



Gambar 5.4
Halaman update artikel

Jika semua kegiatan diatas telah dilakukan, selanjutnya kita akan memodifikasi *KategoriController*. Silahkan anda buka file *backend/controllers/KategoriController.php* dan ubah menjadi seperti berikut.

```
<?php

namespace backend\controllers;

use Yii;
use common\models\Kategori;
use yii\data\ActiveDataProvider;
use yii\web\Controller;
use yii\web\NotFoundHttpException;
use yii\filters\VerbFilter;
use yii\filters\AccessControl;

class KategoriController extends Controller
{
    public function behaviors()
    {
        return [
            'access' => [
                'class' => AccessControl::className(),
                'rules' => [
                    [
                        'actions' => ['index', 'create', 'update',
'delete'],
                        'allow' => true,
                        'roles' => ['@'],

```

```

        ],
    ],
    'verbs' => [
        'class' => VerbFilter::className(),
        'actions' => [
            'delete' => ['post'],
        ],
    ],
];
}

public function actionIndex()
{
    $dataProvider = new ActiveDataProvider([
        'query' => Kategori::find(),
        'sort' => [
            'defaultOrder' => [
                'nama_kategori' => SORT_ASC
            ]
        ]
    ]);

    return $this->render('index', [
        'dataProvider' => $dataProvider,
    ]);
}

public function actionCreate()
{
    $model = new Kategori();

    if ($model->load(Yii::$app->request->post()) && $model->save()) {
        return $this->redirect(['index']);
    } else {
        return $this->render('create', [
            'model' => $model,
        ]);
    }
}

public function actionUpdate($id)
{
    $model = $this->findModel($id);

    if ($model->load(Yii::$app->request->post()) && $model->save()) {
        return $this->redirect(['index']);
    } else {
        return $this->render('update', [
            'model' => $model,
        ]);
    }
}
}

```



```

public function actionDelete($id)
{
    $this->findModel($id)->delete();

    return $this->redirect(['index']);
}

protected function findModel($id)
{
    if (($model = Kategori::findOne($id)) !== null) {
        return $model;
    } else {
        throw new NotFoundHttpException('The requested page
does not exist.');
```

Jika Anda perhatikan lagi, modifikasi pada KategoriController juga hamper sama dengan ArtikelController. Kita hanya menambahkan hak akses, mengubah urutan kategori, dan mengubah halaman redirect pada saat create dan update kategori selesai.

Selanjutnya, masih sama dengan artikel tadi. Kali ini kita ubah tampilan index dengan cara memodifikasi file *backend/views/kategori/index.php* dan ikuti kode berikut.

```

<?php

use yii\helpers\Html;
use yii\grid\GridView;

$this->title = 'Kategori';
$this->params['breadcrumbs'][] = $this->title;
?>

<div class="kategori-index panel panel-info">
    <div class="panel-heading">
        <h4><?= Html::encode($this->title) ?>
        <span class="pull-right">
            <?= Html::a('Tambah Kategori', ['create'], ['class' =>
'btn btn-primary btn-sm']) ?>
            <?= Html::a('Artikel', ['/artikel'], ['class' => 'btn
btn-danger btn-sm']) ?>
        </span>
    </h4>
    </div>

    <div class="panel-body">
        <?= GridView::widget([
```

```

        'dataProvider' => $dataProvider,
        'columns' => [
            ['class' => 'yii\grid\SerialColumn'],
            'nama_kategori',
            [
                'class' => 'yii\grid\ActionColumn',
                'template' => '{update} {delete}'
            ],
        ],
    ]; ?>
</div>
</div>

```

Kemudian ubah juga tampilan form dengan memodifikasi file *backend/views/kategori/_form.php* dan ikuti baris kode berikut.

```

<?php

use yii\helpers\Html;
use yii\widgets\ActiveForm;

?>

<div class="kategori-form">
    <div class="panel panel-info">
        <div class="panel-heading">
            <h4>
                <?= $this->title ?>
                <span class="pull-right">
                    <?= Html::a('Kembali', ['index'], ['class' =>
'btn btn-danger btn-sm']) ?>
                </span>
            </h4>
        </div>
        <div class="panel-body">
            <?php $form = ActiveForm::begin([
                'options' => [
                    'class' => 'col-md-4'
                ]
            ]); ?>

            <?= $form->field($model, 'nama_kategori')->
textInput(['maxlength' => true]) ?>

            <div class="form-group">
                <?= Html::submitButton($model->isNewRecord ?
'Create' : 'Update', ['class' => $model->isNewRecord ? 'btn btn-
success' : 'btn btn-primary']) ?>
            </div>
            <?php ActiveForm::end(); ?>
        </div>
    </div>
</div>

```

Ubah juga file file create yang berada pada direktori *backend/views/kategori/create.php*.

```
<?php

use yii\helpers\Html;

$this->title = 'Tambah Kategori';
$this->params['breadcrumbs'][] = ['label' => 'Kategori', 'url' =>
['index']];
$this->params['breadcrumbs'][] = $this->title;
?>

<div class="kategori-create">
    <?=$this->render('_form', [
        'model' => $model,
    ]) ?>
</div>
```

Terakhir ubah file update yang terdapat pada direktori *backend/views/kategori/update.php*.

```
<?php

use yii\helpers\Html;

$this->title = 'Update Kategori';
$this->params['breadcrumbs'][] = ['label' => 'Kategori', 'url' =>
['index']];
$this->params['breadcrumbs'][] = ['label' => $model->
nama_kategori, 'url' => ['index']];
$this->params['breadcrumbs'][] = 'Update';

?>

<div class="kategori-update">
    <?=$this->render('_form', [
        'model' => $model,
    ]) ?>
</div>
```

Selesai, agar halaman admin lebih rapi lagi. Kita ubah file main.php yang berada pada direktori *backend/views/layouts/main.php*. File ini bisa di bilang file template kita, atau file layout, tema atau sejenisnya.

```
<?php

use backend\assets\AppAsset;
```

```

use yii\helpers\Html;
use yii\bootstrap\Nav;
use yii\bootstrap\NavBar;
use yii\widgets\Breadcrumbs;
use common\widgets\Alert;

AppAsset::register($this);
?>
<?php $this->beginPage() ?>

<!DOCTYPE html>
<html lang="<? = Yii::$app->language ?>">
<head>
    <meta charset="<? = Yii::$app->charset ?>">
    <meta name="viewport" content="width=device-width, initial-
scale=1">
    <? = Html::csrfMetaTags() ?>
    <title><? = Html::encode($this->title) ?></title>
    <?php $this->head() ?>
</head>

<body>
<?php $this->beginBody() ?>

<div class="wrap">
    <?php NavBar::begin([
        'brandLabel' => Yii::$app->name,
        'brandUrl' => Yii::$app->homeUrl,
        'options' => [
            'class' => 'navbar-inverse navbar-fixed-top',
        ],
    ]); ?>

    <? = Nav::widget([
        'options' => ['class' => 'navbar-nav navbar-right'],
        'items' => [
            [
                'label' => 'Logout',
                'visible' => !Yii::$app->user->isGuest,
                'url' => ['/site/logout'],
                'linkOptions' => ['data-method' => 'post']
            ]
        ],
    ]); ?>

    <?php NavBar::end(); ?>

    <div class="container">
        <? = Breadcrumbs::widget([
            'links' => isset($this->params['breadcrumbs']) ?
$this->params['breadcrumbs'] : [],
        ]) ?>
        <? = Alert::widget() ?>
        <? = $content ?>
    </div>
</div>

```

```

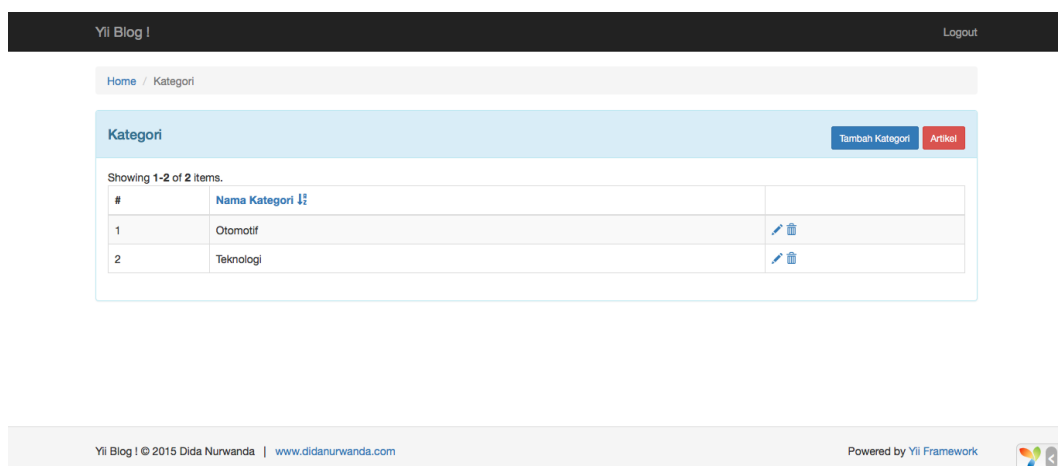
<footer class="footer">
  <div class="container">
    <p class="pull-left">
      <?= Yii::$app->name ?> &copy; <?= date('Y') ?> Dida
      Nurwanda
      &nbsp; | &nbsp;
      <?= Html::a('www.didanurwanda.com',
'http://www.didanurwanda.com') ?>
    </p>

    <p class="pull-right"><?= Yii::powered() ?></p>
  </div>
</footer>

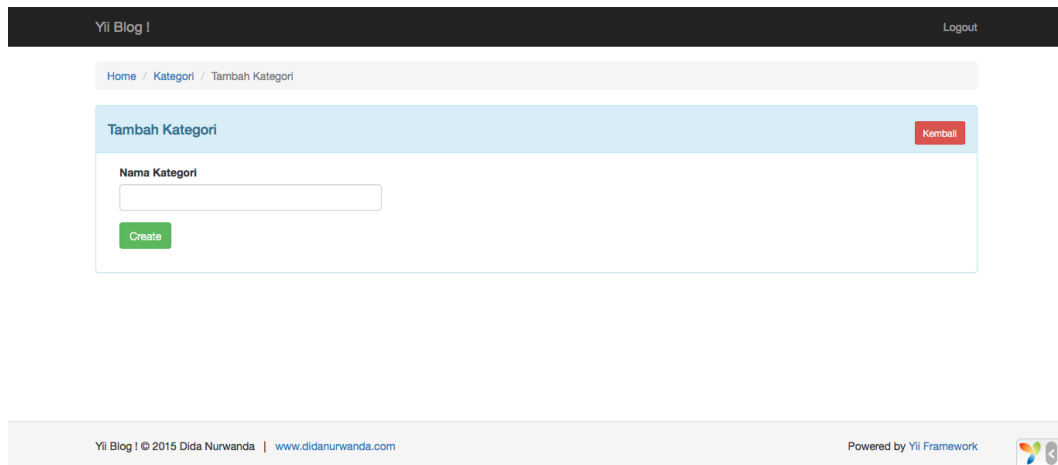
<?php $this->endBody() ?>
</body>
</html>
<?php $this->endPage() ?>

```

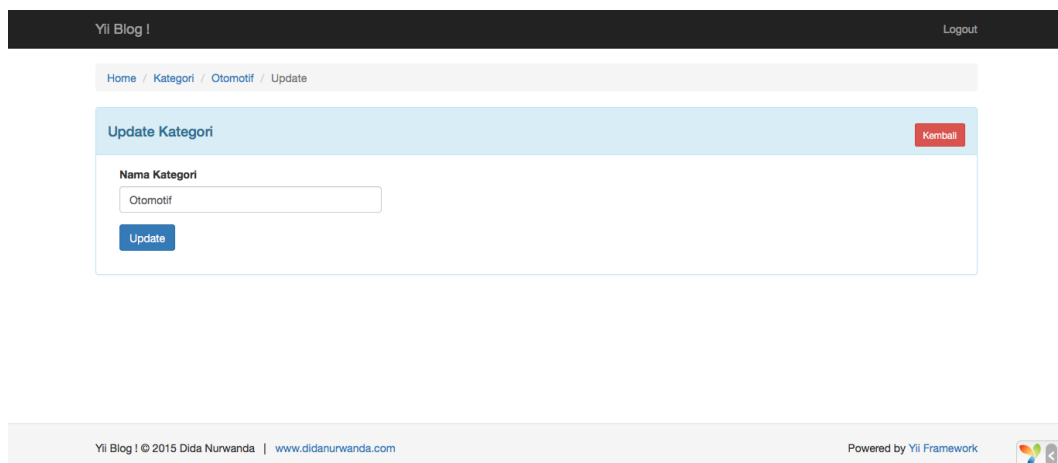
Silahkan Anda buka kembali browser anda dan buka halaman backend. Kemudian tambahkan kategori artikel sesuai dengan keinginan Anda. Input beberapa agar ada pilihan nantinya. Jika proses input kategori telah selesai, Anda juga silahkan input beberapa artikel. Berikut adalah beberapa tampilan untuk halaman kategori.



Gambar 5.5
Halaman daftar kategori



Gambar 5.6
Halaman tambah kategori



Gambar 5.7
Halaman update kategori

Setelah semua yang di atas telah Anda praktikan, tahap selanjutnya adalah membuat halaman blognya. Sederhana memang tapi setidaknya membantu untuk memahami atau mempelajari Yii Framework 2 ini. Langsung saja, sekarang kita beralih ke frontend. Hal yang pertama kita lakukan adalah membuat halaman beranda/home yang berisi daftar berita atau artikel.

Pada contoh aplikasi ini, kita hanya menggunakan satu controller saja untuk frontend. Tapi tentu hal ini jangan selamanya di ikuti, pembuatan controller tentu

sesuai kebutuhan. Silahkan Anda buka file SiteController yang berada pada direktori *frontend/controllers/SiteControllers.php* dan pada bagian *actionIndex()* ubah menjadi seperti berikut.

```
public function actionIndex()
{
    $dataProviderArtikel = new ActiveDataProvider([
        'query' => Artikel::find()
            ->where('id_kategori != :kategori', [
                ':kategori' => isset($_GET['kategori']) ?
$_GET['kategori'] : 'NULL'
            ]),
        'sort' => [
            'defaultOrder' => [
                'id_artikel' => SORT_DESC
            ]
        ]
    ]);
}
```

Diatas kita membuat Data Provider yang digunakan untuk menampilkan daftar artikel yang telah di buat. Artikel diurutkan dimana artikel yang baru dibuat akan tampil lebih dahulu. Kemudian kita ubah file *frontend/views/site/index.php* dan ubah menjadi seperti berikut.

```
<?php
use yii\widgets\ListView;
$this->title = Yii::$app->name;

echo ListView::widget([
    'dataProvider' => $dataProviderArtikel,
    'layout' => "{items}\n{pager}",
    'itemOptions' => ['class' => 'item'],
    'itemView' => '_itemArtikel'
]);
```

Dan kemudaian buat file dengan nama “_itemArtikel.php” pada direktori *frontend/views/site/_itemArtikel.php*. File ini bertugas menampilkan isi highlight artikel secara *looping*.

```
<?php
use yii\helpers\Html;
?>

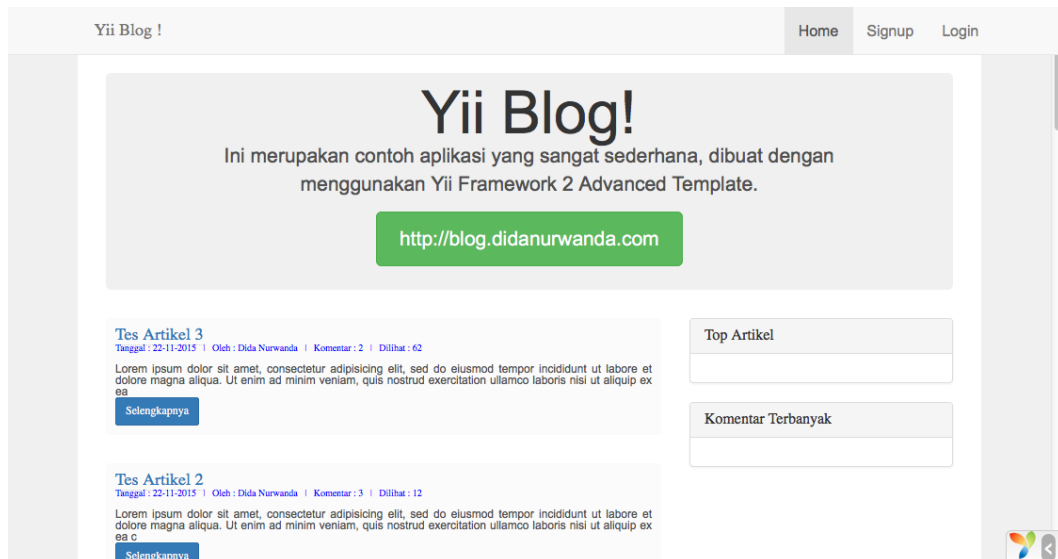
<div class="content">
    <div class="content-title"><?= Html::a($model->judul, ['view',
'id' => $model->id_artikel]) ?></div>
    <div class="content-detail">
```

```

        Tanggal : <strong><?= date('d-m-Y', $model->create_time)
?></strong>
        &nbsp; | &nbsp;
        Oleh : <strong><?= $model->createBy->first_name . ' '.
$model->createBy->last_name ?></strong>
        &nbsp; | &nbsp;
        Komentar : <strong><?= count($model->komentars)
?></strong>
        &nbsp; | &nbsp;
        Dilihat : <strong><?= $model->jumlah_baca ?></strong>
        &nbsp; | &nbsp;
        Kategori : <strong><?= $model->idKategori->nama_kategori
?></strong>
    </div>
    <div class="content-fill">
        <p style="text-align: justify"><?=
substr(strip_tags($model->isi_artikel), 0, 300) ?></p>
        <?= Html::a('Selengkapnya', ['view', 'id'=>$model-
>id_artikel], ['class' => 'btn btn-sm btn-primary']) ?>
    </div>
</div>

```

Jika sudah, silahkan Anda buka browser dan masukan ke link <http://localhost/yiiblog/frontend/web> maka Akan tampil tampilan seperti berikut.



Gambar 5.8
Halaman depan

Selanjutnya kita akan buat halaman detail artikel dan komentar. Pertama buat fungsi atau method berikut pada SiteController.


```

public function actionView($id)
{
    $model = Artikel::findOne($id);

    // menambahkan jumlah baca 1
    $model->updateCounters(['jumlah_baca' => 1]);

    // form komentar
    $komentarForm = new Komentar();
    if ($komentarForm->load(Yii::$app->request->post()) &&
    $komentarForm->save()) {
        return $this->redirect(['view', 'id' => $id]);
    }

    // data provider komentar
    $dataProviderKomentar = new ActiveDataProvider([
        'query' => Komentar::find()->where(['id_artikel' => $id]),
        'sort' => [
            'defaultOrder' => [
                'id_komentar' => SORT_DESC
            ]
        ]
    ]);

    return $this->render('view', [
        'model' => $model,
        'komentarForm' => $komentarForm,
        'dataProviderKomentar' => $dataProviderKomentar
    ]);
}

```

Pada baris kode diatas, terdapat beberapa fungsi untuk menambahkan jumlah baca, menyimpan komentar, menampilkan artikel, dan menampilkan komentar. Di Yii semua kegiatan tersebut dapat dengan mudah dan sederhana di buat. Kemudian buat file *frontend/views/site/view.php* dan isi seperti berikut.

```

<?php

use yii\widgets\ActiveForm;
use yii\helpers\Html;
use yii\widgets\ListView;
$this->title = Yii::$app->name . ' - '. $model->judul;

?>

<div class="content">
    <div class="content-title"><?= Html::a($model->judul, ['view',
'id' => $model->id_artikel]) ?></div>
    <div class="content-detail">
        Tanggal : <strong><?= date('d-m-Y', $model->create_time)
?></strong>
        &nbsp; | &nbsp;

```

```

        Oleh : <strong><?=$model->createBy->first_name .' '.
$model->createBy->last_name ?></strong>
        &nbsp; | &nbsp;
        Komentar : <strong><?=$count($model->komentars)
?></strong>
        &nbsp; | &nbsp;
        Dilihat : <strong><?=$model->jumlah_baca ?></strong>
        &nbsp; | &nbsp;
        Kategori : <strong><?=$model->idKategori->nama_kategori
?></strong>

    </div>
    <div class="content-fill"><?=$nl2br($model->isi_artikel)
?></div>
</div>

<br />
<br />

<div class="panel panel-info">
    <div class="panel-heading">Komentar</div>
    <div class="panel-body">
        <?php $form = ActiveForm::begin([
            'options' => [
                'style' => 'font-size: 80%'
            ]
        ]); ?>

        <?=$form->field($komentarForm, 'id_artikel')-
>hiddenInput(['value' => $model->id_artikel])->label(false) ?>

        <?=$form->field($komentarForm, 'nama')-
>textInput(['maxlength' => true]) ?>

        <?=$form->field($komentarForm, 'email')-
>textInput(['maxlength' => true]) ?>

        <?=$form->field($komentarForm, 'isi_komentar')-
>textarea(['rows' => 6]) ?>

        <div class="form-group">
            <?=$Html::submitButton('Kirim', ['class' => 'btn btn-
success']) ?>
        </div>

        <?php ActiveForm::end(); ?>
    </div>
</div>

<hr />
<h3><i>Komentar</i></h3>

<?=$ListView::widget([
    'dataProvider' => $dataProviderKomentar,
    'layout' => "{items}\n{pager}",

```

```

        'itemOptions' => ['class' => 'item'],
        'itemView' => '_itemKomentar'
    ]) ?>

```

Dan buat juga file dengan nama “_itemKomentar” pada direktori *frontend/views/site/_itemKomentar.php*. Ini tujuannya sama dengan *_itemArtikel.php* namun untuk menampilkan komentar.

```

<?php
use yii\helpers\Html;

?>

<div class="panel panel-success">
    <div class="panel-heading"><?= $model->nama ?></div>
    <div class="panel-body">
        <?= Html::encode($model->isi_komentar) ?>
    </div>
</div>

```

Terakhir kita sedikit mengubah model Komentar yang terdapat pada direktori *common/models/Komentar.php* kemudian ubah dan sesuaikan menjadi dengan baris kode berikut.

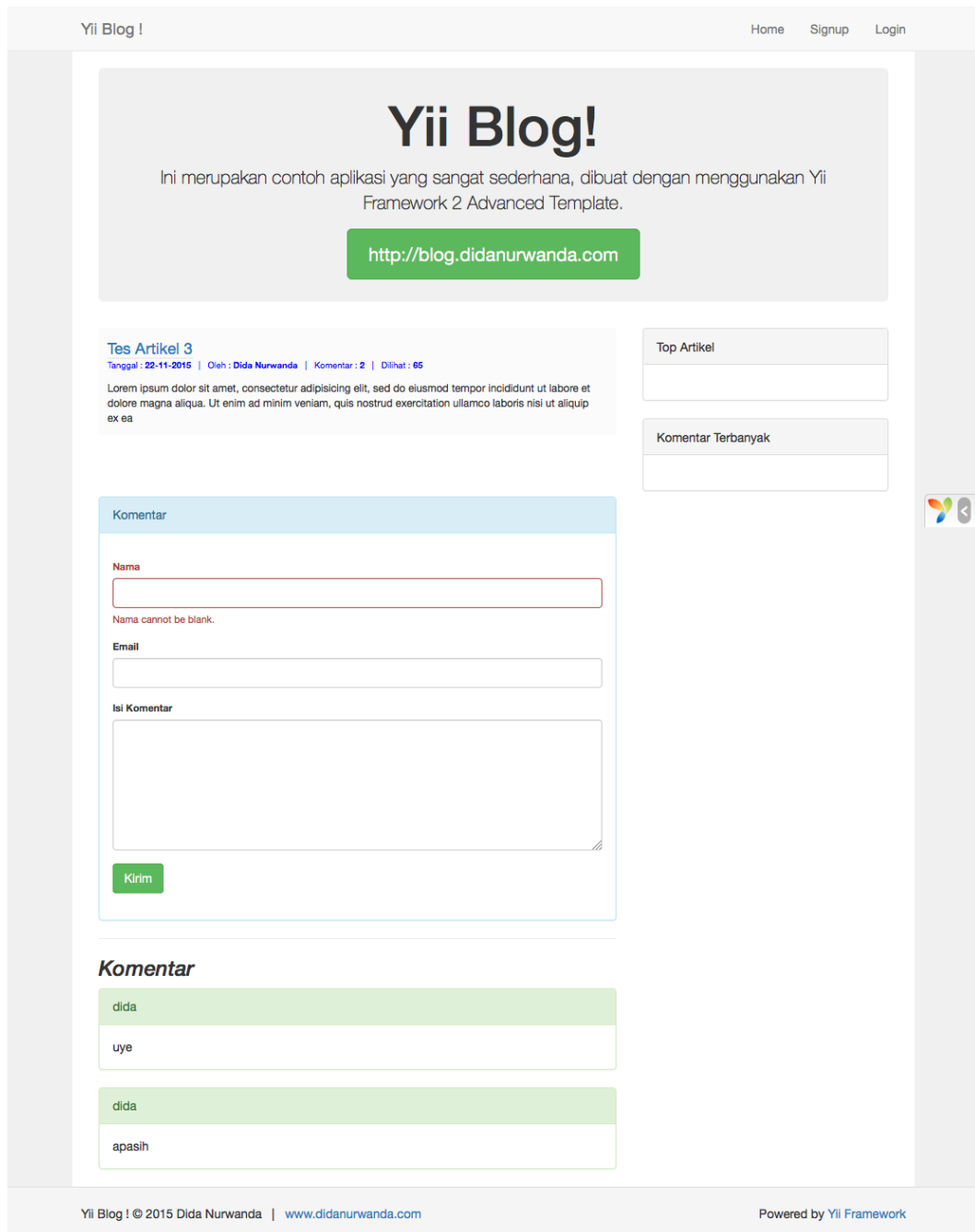
```

public function rules()
{
    return [
        [['id_artikel', 'nama', 'email', 'isi_komentar'],
        'required'],
        [['id_artikel', 'create_time'], 'integer'],
        [['email'], 'email'],
        [['isi_komentar'], 'string'],
        [['nama', 'email'], 'string', 'max' => 100]
    ];
}

public function beforeSave($insert)
{
    parent::beforeSave($insert);
    if ($this->isNewRecord)
    {
        $this->create_time = time();
    }
    return true;
}

```

Selesai, silahkan Anda buka browser Anda dan pilih artikel yang dan coba-coba isi komentar. Berikut tampilan detail artikel.



Gambar 5.9
Halaman detail artikel

Tahap terakhir dalam pembuatan aplikasi ini adalah mengaktifkan beberapa baris kode yang di blok oleh komentar yang terdapat pada file *frontend/views/layouts/main.php*. Baris tersebut merupakan menu kategori, artikel paling banyak di baca dan komentar paling banyak. Silahkan Anda hapus blok komentar pada baris kode berikut.

```
// ['label' => 'Kategori', 'items' =>
common\models\Kategori::getKategoriMenu()],

<div class="panel panel-default">
  <div class="panel-heading">Top Artikel</div>
  <div class="panel-body">
    <ul>
      <?php
      /*
      <?php foreach(common\models\Artikel::topArtikel() as
$row): ?>
        <li><?= Html::a($row->judul .' ('.$row-
>jumlah_baca.)', ['view', 'id' => $row->id_artikel]) ?></li>
        <?php endforeach; ?>

      */
      ?>
    </ul>
  </div>
</div>

<div class="panel panel-default">
  <div class="panel-heading">Komentar Terbanyak</div>
  <div class="panel-body">
    <ul>
      <?php
      /*
      <?php foreach(common\models\Artikel::topKomentar() as
$row): ?>
        <li><?= Html::a($row->judul .' ('. count($row-
>komentars).')', ['view', 'id' => $row->id_artikel]) ?></li>
        <?php endforeach; ?>
      */
      ?>
    </ul>
  </div>
</div>
```

Setelah disimpan dan Anda reload browser Anda maka akan muncul error. Itu disebabkan kita belum membuat fungsi dari *getKategoriMenu()*, *topArtikel()*, dan *topKomentar()*. Maka dari itu silahkan Anda tambahkan baris kode berikut pada model Kategori.

```

public function getKategoriMenu()
{
    $ar = [];
    foreach(Kategori::find()->all() as $row)
    {
        $ar[] = ['label' => $row->nama_kategori, 'url' =>
['/site/index', 'kategori' => $row->id_kategori]];
    }

    return $ar;
}

```

Kemudian, tambahkan juga baris kode berikut pada model Artikel.

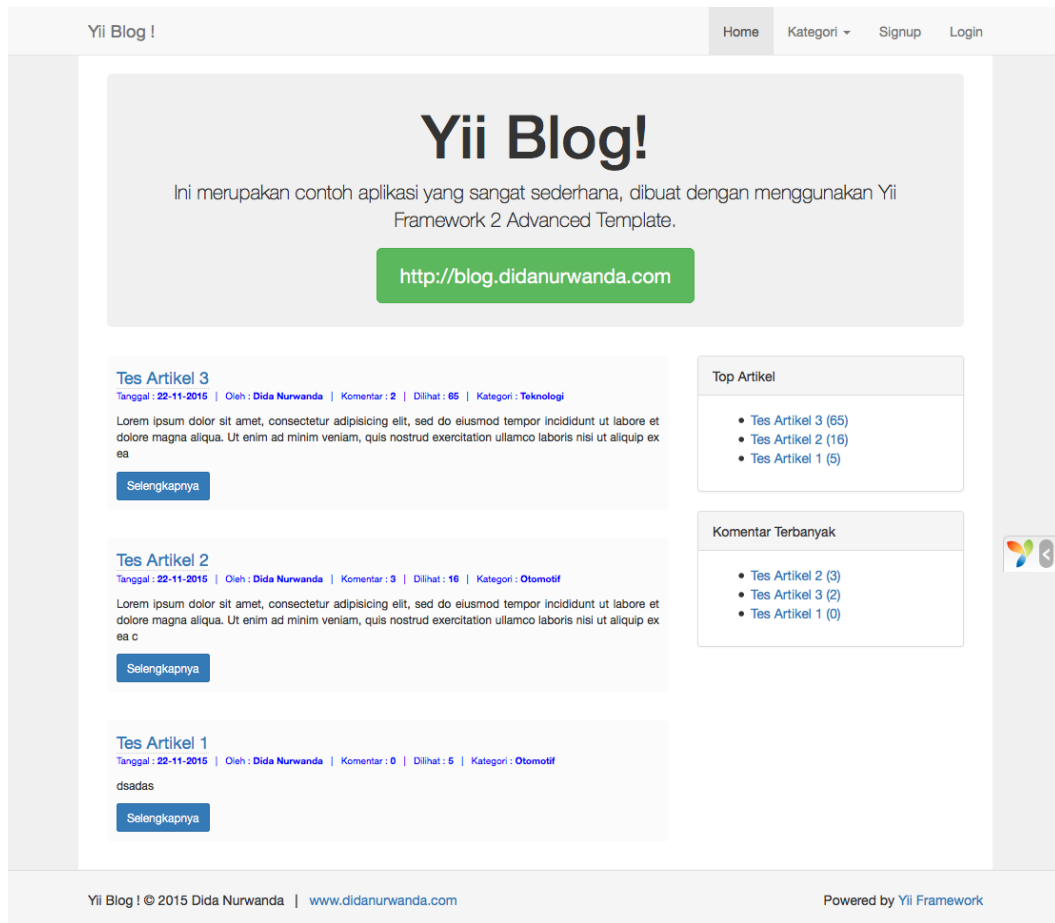
```

public static function topArtikel()
{
    return self::find()
        ->orderBy('jumlah_baca DESC')
        ->limit(10)
        ->all();
}

public function topKomentar()
{
    return Artikel::findBySql("SELECT a . * , COUNT(
k.id_komentar ) AS jumlah
FROM artikel a
LEFT JOIN komentar k ON ( k.id_artikel =
a.id_artikel )
GROUP BY a.id_artikel
ORDER BY `jumlah` DESC
LIMIT 0 , 10")
        ->all();
}

```

Jika telah di tambahkan, silahkan anda reload browser Anda. Maka tampilan akan menjadi seperti berikut.



Gambar 5.10
Halaman depan

Itulah aplikasi yang kita buat, sangat sederhana. Silahkan Anda kembangkan lebih jauh lagi. Untuk kedepannya silahkan Anda request ebook yang membahas tentang programming ke email didanurwanda@gmail.com.

