

## Dokumentasi CodeIgniter 4 Unpas.

### ✚ Persiapan.

- ✓ Install XAMPP.
- ✓ Install VS Code.
  - Install intelephense, digunakan u/ mempermudah dalam pen-kodingan php. Untuk aktivasi caranya ke *extension* → *search* → *@builtin php* → *php language feature* → *disable*.
  - Install *prettier*, digunakan u/ *text formatter*. u/ aktivasi pergi ke *preferences* → *format with save* → *check*.
  - Buat juga *snippets*, yaitu *shortcut* kode yang dapat digunakan u/ pemrograman php nantinya. Ke *preferences* → *user snippets* → *new snippets* → *html.json*.

```
{
  "PHP Tag" :{
    "prefix": "php",
    "body": "<?php $1 ?>"
  },
  "Inline Echo":{
    "prefix": "php",
    "body": "<?= $$1; ?>"
  }
}
```

- ✓ Install Composer.
- ✓ Install GitBash.
- ✓ Install CI4. Pergi ke CI4 web → gunakan perintah install dengan GitBash dan Composer, copy syntax → jalankan GitBash cmd pada folder htdocs dan folder project → paste, jangan lupa juga untuk memberi nama file project nya (ci4app pada gambar).



```
MINGW64:/d/xampp/htdocs/Project-ku
FX505DT@LAPTOP-UCM6J2T0 MINGW64 /d/xampp/htdocs/Project-ku
$ composer create-project codeigniter4/appstarter ci4app --no-dev
```

\*NB : Jangan lupa juga untuk *enable intl extension* pada file php XAMPP, jika ini *disable* maka akan menyebabkan *error* pada saat instalasi CI4.

```
928 | extension=intl
```

- ✓ Untuk menjalankan CI4, masuk ke folder project yang sudah dilakukan instalasi → GitBash here → ketik perintah `php spark serve`.

### ✚ Environment dan struktur folder.

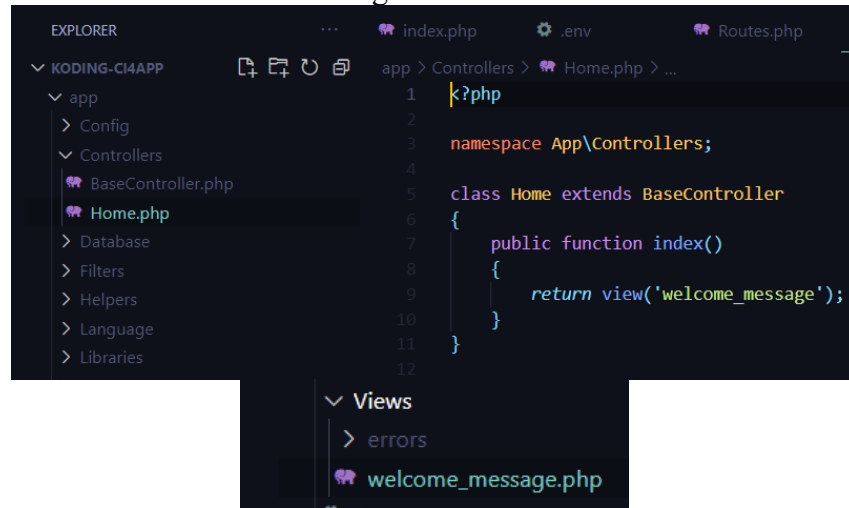
- ✓ Folder Apps, ini adalah folder inti kita. Aplikasi yang kita buat akan disimpan di dalam folder ini. Di dalam folder ini terdapat banyak sekali folder yang akan dipakai, diantaranya yang akan kita sering gunakan.
  - Controller, folder ini berisi seluruh controller dari semua Class. Semua class controller akan disimpan pada folder ini.
  - Model, salah satu dari fungsi folder ini ialah untuk mengatur *database*.
  - View, folder ini digunakan untuk menyimpan interface/frontend dari web yang kita buat.

- Config, di dalam folder ini kita perlu mengubah beberapa file yang terdapat didalamnya, seperti config, database dll.
  - Di dalam config juga kita dapat melakukan modifikasi seperti deteksi database yang digunakan pada file database.php. Kita dapat menentukan hostnamen, username, hingga password database yang digunakan atau juga kita dapat mengubahnya pada file .env .
  - Di dalam config juga kita akan mengubah konfigurasi file Routes.php. File ini akan sering sekali kita kunjungi pada saat pembuatan project WEB. File ini juga berfungsi sebagai penentu jalur yang akan dikunjungi oleh pengunjung WEB.
- ✓ Folder public, folder ini digunakan untuk menyimpan folder-folder assets (CSS, Images dll) kita nantinya. Pada saat kita pertama kali install CI4, kita akan memodifikasi folder ini,
  - .htaccess, diantara fungsi dari file ini ialah untuk membuat URL yang rapi. URL kita dapat rapi tanpa adanya nama-nama files yang lain.
  - Favicon.ico , file ini dapat digunakan untuk membuat icon pada tab browser WEB kita.
  - Index.php ,file ini akan menjalankan seluruh perintah CI4 mu, jadi pastikan jangan mengubah apapun pada file ini. `$app->run();` satu baris perintah tersebut dapat menjalankan seluruh perintah/program di dalam folder Apps.
- ✓ Test, folder ini digunakan pada saat kita akan melakukan testing pada WEB apps kita.
- ✓ Vendor, folder ini digunakan untuk dependency di dalam aplikasi kita. Isi dari file ini diatur oleh Composer.json file. Contoh fungsi dari file ini, pada saat kita membutuhkan plugin dan kita install menggunakan Composer, file ini akan dipenuhi oleh syntax instalasi tersebut. Atau dengan kata lain file ini secara otomatis akan diisi oleh Composer sehingga kita tidak perlu lagi mengisinya secara manual.
- ✓ env , file ini adalah template konfigurasi environment. Untuk membuat file ini menjadi konfigurasi environment aplikasi kita perlu mengaktifkannya dengan cara mengubah nama menjadi .env . saran dari dosen unpas, lebih baik kita men-copy file ini sebelum mengubahnya, sehingga kita tetap memiliki defaultnya. Ada beberapa syntax yang perlu diaktifkan di dalam .env ini, caranya dengan menghapus # di depan syntax, sebagai berikut.
  - CI\_ENVIRONMENT = development/testing/production. Perbedaan dari setiap valuenya adalah Ketika kita memakai environment development Ketika kita melakukakn kode program maka Ketika ada kode yang error akan ditampilkan, jika menggunakan produksi akan disembunyikan.
  - App.baseURL = 'http://localhost:8080'

#### Routes dan Controllers.

- ✓ Routes, adalah komponen yang akan diakses Ketika kita mengetikkan perintah pada URL WEB. Routes akan menentukan Controllers mana yang akan dieksekusi, kemudian Controllers akan menentukan kemana kah data yang diperintahkan, akan menuju View kemudian menampilkannya atau menuju Model untuk mengambil data dan menampilkannya. `$routes->get('/', 'Home::index');` kita dapat membacanya CI4 buat jalur Ketika ada akses yang request methodnya adalah get dan '/' sebagai alamat rootnya kemudian arahkan ke Controller Home dengan nama method Index.

Controller Home berasosiasi dengan file yang berada di folder Controller → home.php yang berisikan method index, dan ini berisikan parameter view yang akan mengembalikan nilai 'welcome messages' dari folder view.



The screenshot shows the VS Code interface. On the left, the Explorer sidebar shows the project structure with folders like Config, Controllers, Database, Filters, Helpers, Language, and Libraries. The Controllers folder is expanded, showing BaseController.php and Home.php. The Home.php file is selected, and its code is displayed in the editor. The code defines a class Home that extends BaseController and has an index() method that returns the view 'welcome\_message'. Below the editor, a snippet of the Views folder is visible, showing errors and welcome\_message.php.

```
1 <?php
2
3 namespace App\Controllers;
4
5 class Home extends BaseController
6 {
7     public function index()
8     {
9         return view('welcome_message');
10    }
11 }
12
```

- ✓ Ketika kita ingin menangani/memanipulasi URL untuk mencegah Ketika user mengetikkan perintah sembarang di URL WEB, kita dapat menggunakan perintah berikut.

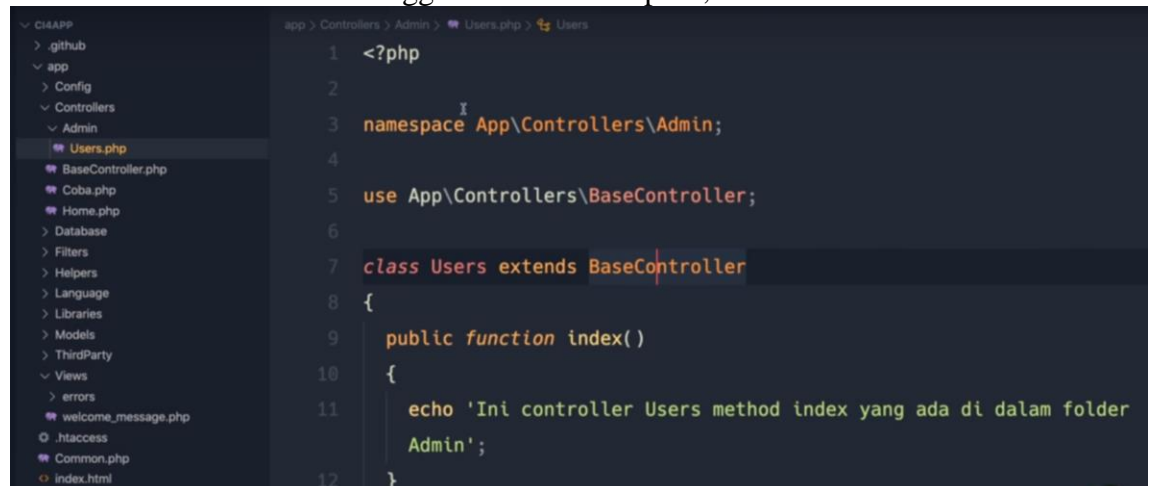
```
$routes->get('/coba/(:any)', 'Coba::about/$1');
```

kita dapat membacanya Ketika user mengetik secara random di URL tolong CI4 buatkan route yang mengambil nilai dari URL (dengan method get) dan berikan arah ke baseURL (dalam kasus ini coba) dan ambil nilai apapun darinya (:any) kemudian arahkan ke Controller Coba dan method about.

```
$routes->get('/coba/index', 'Coba::index');
$routes->get('/coba/about', 'Coba::about');
$routes->get('/coba/(:any)', 'Coba::about/$1');
```

full screenshoot →

- ✓ Membuat Controllers menggunakan namespace, contoh untuk admin.

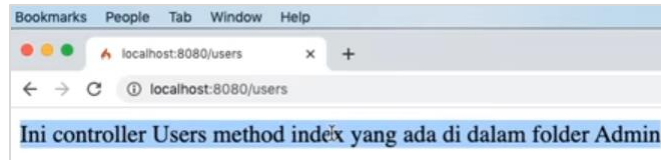


The screenshot shows the VS Code interface. On the left, the Explorer sidebar shows the project structure with folders like Config, Controllers, Database, Filters, Helpers, Language, Libraries, Models, ThirdParty, and Views. The Controllers folder is expanded, showing Admin, BaseController.php, Coba.php, Home.php, and Users.php. The Admin folder is expanded, showing Users.php. The Users.php file is selected, and its code is displayed in the editor. The code defines a class Users that extends BaseController and has an index() method that echoes a message. The file is named Users.php but is located in the Admin folder.

```
1 <?php
2
3 namespace App\Controllers\Admin;
4
5 use App\Controllers\BaseController;
6
7 class Users extends BaseController
8 {
9     public function index()
10    {
11        echo 'Ini controller Users method index yang ada di dalam folder Admin';
12    }
13 }
```

Kita juga perlu membuat routesnya seperti yang ditampilkan di bawah.

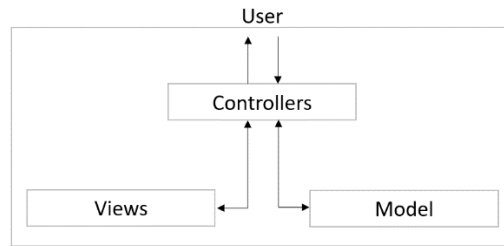
```
$routes->get('/users', 'Admin\Users::index');
```



sehingga hasilnya adalah seperti gambar disamping. (untuk lebih lanjutnya dapat disimak pada video CI4 unpas episode routes & controllers).

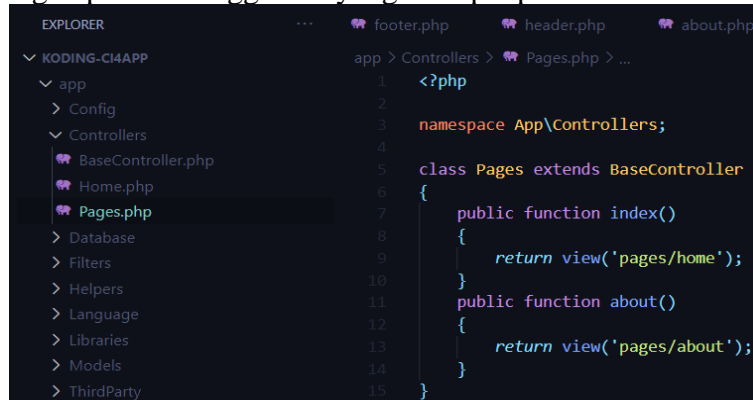
- ✓ Controllers, pastikan Ketika anda membuat Controllers nama Class dan filenya sama.

## Views.

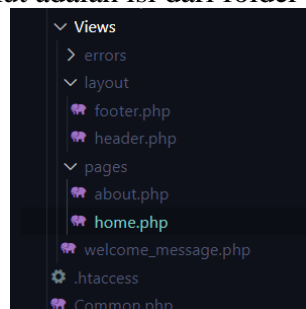


MVC adalah sebuah metode yang digunakan oleh CI guna memproses dan menampilkan halaman WEB. MVC dimulai dari inputan yang diberikan oleh user, kemudian akan diproses oleh Controllers. Controllers akan memproses perintah dari user dan akan menampilkan kepada pengguna halaman WEB tergantung dari kebutuhan pengguna, perintah untuk menampilkan Views atau mengambil data di Model.

- ✓ Untuk dapat menampilkan berbagai halaman WEB kita, kita dapat membuat Controllers yang dapat memanggil file yang terdapat pada folder Views.



Dari gambar dapat dibaca/dideskripsikan folder Controllers yang berisikan file Page.php diperintahkan untuk menampilkan halaman WEB dari folder Views → folder pages → home.php. Dan berikut adalah isi dari folder Views.



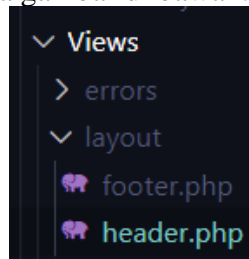
- ✓ Pastikan Ketika anda membuat sebuah file Controller baru, ganti nama Class sama dengan nama file dan untuk huruf depan dibuat Kapital.

```

File Edit Selection View Go Run Terminal Help
EXPLORER
KODING-CI4APP
  app
    Config
    Controllers
      BaseController.php
      Home.php
      Pages.php
    Database
    Filters
    Helpers
    Language
    Libraries
    Models
    ThirdParty
Pages.php
app > Controllers > Pages.php > Pages
1 <?php
2
3 namespace App\Controllers;
4
5 class Pages extends BaseController
6 {
7     public function index()
8     {
9         return view('pages/home');
10    }
11    public function about()
12    {
13        return view('pages/about');
14    }
15 }

```

- ✓ Kita juga dapat membuat header dan footer WEB secara terpisah, Teknik ini disebut multiple layout/views. Lihat pada gambar di bawah.



Kita dapat membuat folder layout di dalam folder views untuk menyimpan header dan footer.

```

> Views > layout > header.php > html > body
<!doctype html>
<html lang="en">

<head>
  <!-- Required meta tags -->
  <meta charset="utf-8">
  <meta name="viewport" content="width=device-width, initial-scale=1">

  <!-- Bootstrap CSS -->
  <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.1.0/dist/css/bootstrap.min.css" rel="stylesheet" integrity="sha384-KyZXEAg3QhqLM
  <title>About me</title>
</head>

<body>

```

(header.php)

```

app > Views > layout > footer.php > ...
1 <!-- Optional JavaScript; choose one of the two! -->
2
3 <!-- Option 1: Bootstrap Bundle with Popper -->
4 <script src="https://cdn.jsdelivr.net/npm/bootstrap@5.1.0/dist/js/bootstrap.bundle.min.js" integrity="sha384-U1DAWAznBHeqEITVSCgzq+c9gqGA
5
6 <!-- Option 2: Separate Popper and Bootstrap JS -->
7 <!--
8 <script src="https://cdn.jsdelivr.net/npm/@popperjs/core@2.9.3/dist/umd/popper.min.js" integrity="sha384-eMNCoe7TC1dohpgowe/6oMvmdAV
9 <script src="https://cdn.jsdelivr.net/npm/bootstrap@5.1.0/dist/js/bootstrap.min.js" integrity="sha384-cn7L7gDp0eyniUwaAZgrZD06ekc/tfTF
10 -->
11 </body>
12
13 </html>

```

(footer.php)

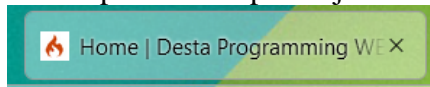
```

app > Controllers > Pages.php > ...
1  <?php
2
3  namespace App\Controllers;
4
5  class Pages extends BaseController
6  {
7      public function index()
8      {
9          echo view('layout/header');
10         return view('pages/home');
11         echo view('layout/footer');
12     }
13     public function about()
14     {
15         echo view('layout/header');
16         echo view('pages/about');
17         echo view('layout/footer');
18     }
19 }

```

kemudian kita dapat memanggilnya melalui Routes dengan cara di atas. Namun teknik ini merupakan teknik dasar, CI4 memiliki teknik tingkat lanjut untuk membuat multiple layout/views ini.

- ✓ Kita dapat memanipulasi judul tab WEB kita.



Caranya kita dapat memanipulasi isi dari Controller seperti berikut, dengan memanfaatkan Array asosiatif.

```

Terminal  Help  Pages.php - Kodig-ci4app - Visual Stu
header.php  footer.php  Pages.php X  Routes.php
app > Controllers > Pages.php > ...
1  <?php
2
3  namespace App\Controllers;
4
5  class Pages extends BaseController
6  {
7      public function index()
8      {
9          $data = [
10             'title' => 'Home | Desta Programming WEB'
11          ];
12          echo view('layout/header', $data);
13          return view('pages/home');
14          echo view('layout/footer');
15      }
16      public function about()
17      {
18          $data = [
19             'title' => 'About | Desta Programming WEB'
20          ];
21          echo view('layout/header', $data);
22          echo view('pages/about');
23          echo view('layout/footer');
24      }
25  }
26

```

Kemudian disetiap header kita tangkap nilai dengan menambahkan ( , \$data). Kemudian kita tangkap data dari array pada header, seperti berikut.

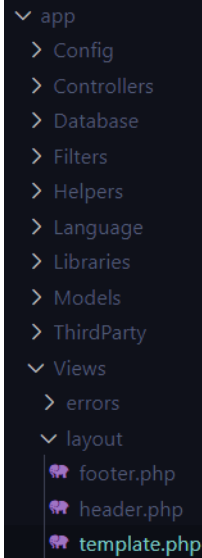
```
<title><?= $title; ?></title>
```

variable title diambil dari Qarray.

## Views layout.

- ✓ Jika sebelumnya kita telah membuat sebuah metode untuk memanggil header dan footer secara terpisah, pada kali ini ada metode yang lebih rapih yaitu disebut dengan templating header-footer.

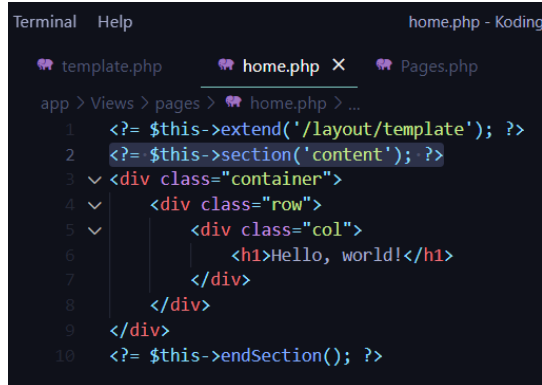
Caranya buka folder Apps → folder Views → folder layout → buat sebuah file bernama template.php, file ini berisi isi dari file header.php dan footer.php, di tengah file template.php antara header dengan footer kita juga buat syntax php untuk menjalankan setiap konten halaman. Dengan cara ini diharapkan bagian header dan footer akan lebih rapih nantinya. Berikut penjelasan dengan gambar.



```
template.php X footer.php header.php
app > Views > layout > template.php > html > body
11 <title><?= $title; ?></title>
12 </head>
13
14 <body>
15 <nav class="navbar navbar-expand-lg navbar-light bg-light">
16 <div class="container">
17 <div class="container-fluid">
18 <a class="navbar-brand" href="#">Desta MY</a>
19 <button class="navbar-toggler" type="button" data-bs-toggle="collapse" data-bs-target="#navbarNavAltMarkup" aria-controls
20 <span class="navbar-toggler-icon"></span>
21 </button>
22 <div class="collapse navbar-collapse" id="navbarNavAltMarkup">
23 <div class="navbar-nav">
24 <a class="nav-link active" aria-current="page" href="/">Home</a>
25 <a class="nav-link" href="/pages/about">About</a>
26 <a class="nav-link" href="#">Contact</a>
27 </div>
28 </div>
29 </div>
30 </nav>
31 <?= $this->renderSection('content'); ?>
32
33 <!-- Optional JavaScript; choose one of the two! -->
34
35 <!-- Option 1: Bootstrap Bundle with Popper -->
36 <script src="https://cdn.jsdelivr.net/npm/bootstrap@5.1.0/dist/js/bootstrap.bundle.min.js" integrity="sha384-U1DAWAznBHeqE11VSCgzq+c9
37
38 <!-- Option 2: Separate Popper and Bootstrap JS -->
39 <!--
40 <script src="https://cdn.jsdelivr.net/npm/@popperjs/core@2.9.3/dist/umd/popper.min.js" integrity="sha384-eMNCoe7tC1dohPgowe/6oMvmdAV
41 <script src="https://cdn.jsdelivr.net/npm/bootstrap@5.1.0/dist/js/bootstrap.min.js" integrity="sha384-cn7L7gDp0eyniUmwAZgrzD06kc/tftF
42 -->
43 </body>
44 </html>
```

Jika sudah, selanjutnya kita juga perlu ubah controller dan view dari halaman.

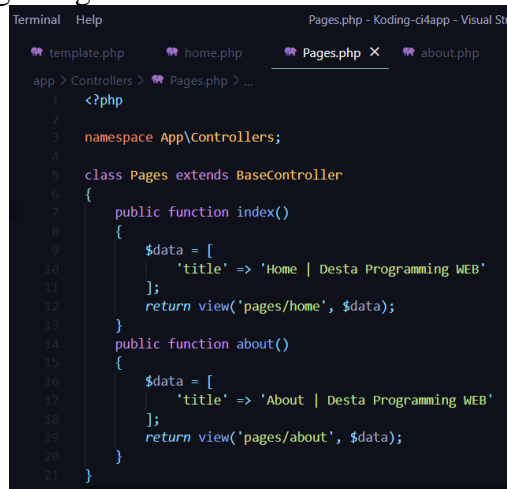
- 1) Kita perlu ubah isi dari halaman konten. Sebagai contoh adalah konten pada halaman home.php. Di dalam file home.php kita perlu menuliskan perintah yang akan memberitahu kepada program setiap element sectionnya. Berikut penjelasan dengan gambar.



```
1 <?=$this->extend('/layout/template'); ?>
2 <?=$this->section('content'); ?>
3 <div class="container">
4   <div class="row">
5     <div class="col">
6       <h1>Hello, world!</h1>
7     </div>
8   </div>
9 </div>
10 <?=$this->endSection(); ?>
```

Sebagai catatan, perintah `<?=$this->section('content'); ?>` berasosiasi dengan perintah `<?=$this->renderSection('content'); ?>` yang terdapat pada template.php

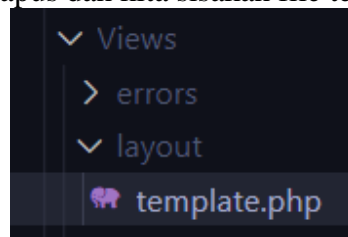
- 2) Kita juga perlu ubah syntax program pada controllers home. Keunggulan dengan teknik ini kita tidak perlu lagi menggunakan cara pemanggilan header dan footer secara berulang sehingga kita dapat menggunakan perintah return kembali, yang jika sebelumnya yang kita gunakan ialah echo.



```
1 <?php
2
3 namespace App\Controllers;
4
5 class Pages extends BaseController
6 {
7     public function index()
8     {
9         $data = [
10             'title' => 'Home | Desta Programming WEB'
11         ];
12         return view('pages/home', $data);
13     }
14     public function about()
15     {
16         $data = [
17             'title' => 'About | Desta Programming WEB'
18         ];
19         return view('pages/about', $data);
20     }
21 }
```

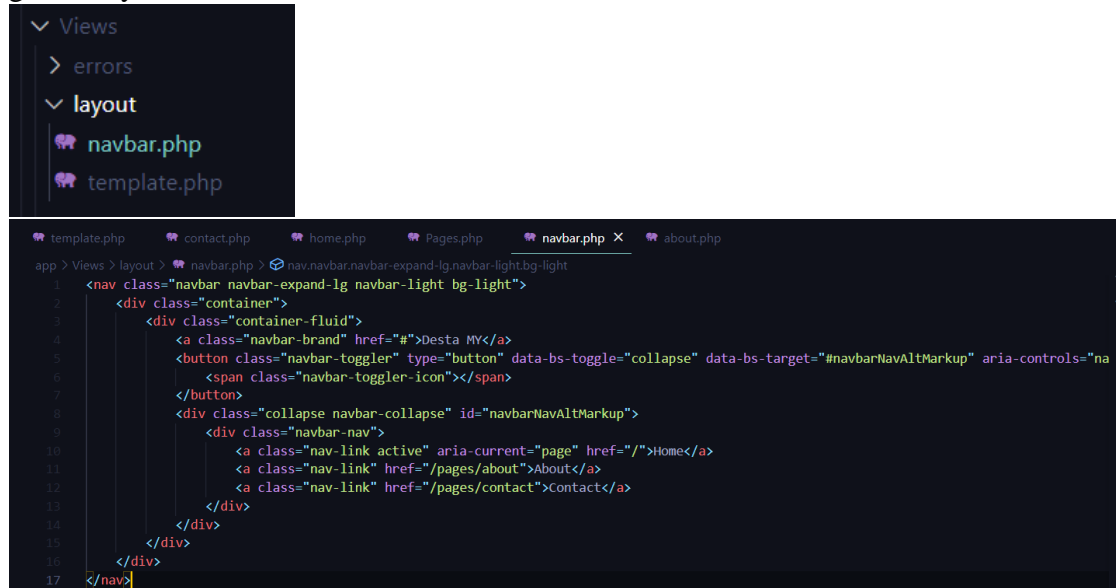
Sehingga Controllers nya akan menjadi seperti di atas.

Setelah kita menggunakan metode templating header-footer di atas, file header.php dan footer.php dapat kita hapus dan kita sisakan file template.php saja.





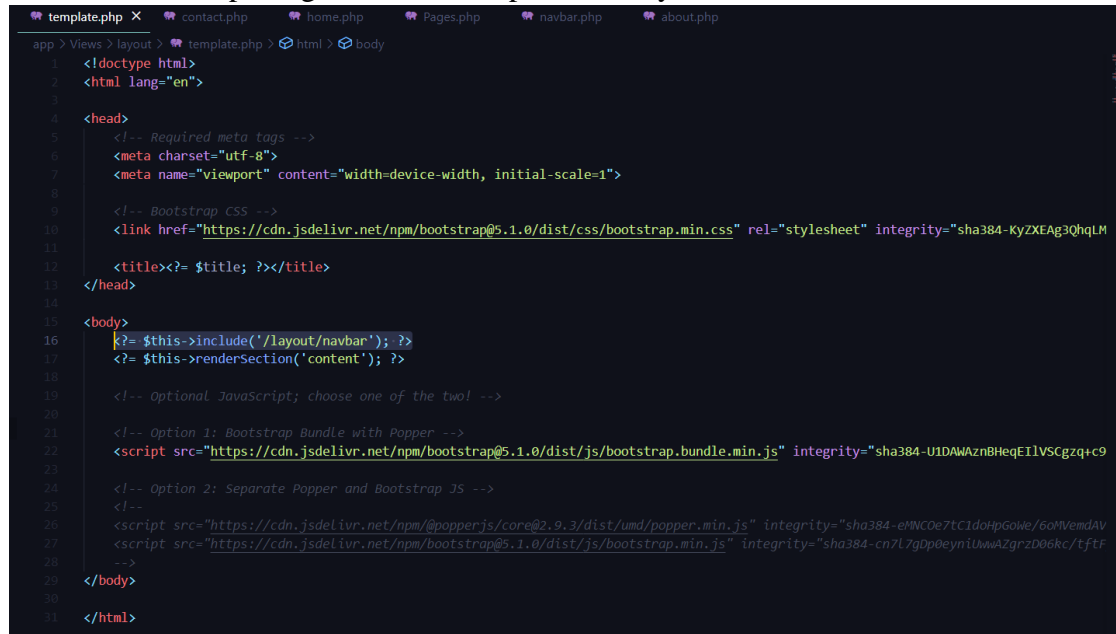
- ✓ Jika sebelumnya di dalam template yang berisikan header & footer website juga berisikan navbar, kita juga dapat memisahkannya untuk bertujuan mudah dalam mengelolanya. Caranya kita masuk folder layout → buat file bernama navbar.php. File ini nanti akan kita asosiasikan dengan file template. Berikut penjelasan dengan gambarnya.



```
Views
├── errors
└── layout
    ├── navbar.php
    └── template.php
```

```
template.php contact.php home.php Pages.php navbar.php X about.php
app > Views > layout > navbar.php > nav.navbar.navbar-expand-lg.navbar-light.bg-light
1 <nav class="navbar navbar-expand-lg navbar-light bg-light">
2   <div class="container">
3     <div class="container-fluid">
4       <a class="navbar-brand" href="#">Desta MY</a>
5       <button class="navbar-toggler" type="button" data-bs-toggle="collapse" data-bs-target="#navbarNavAltMarkup" aria-controls="na
6       <span class="navbar-toggler-icon"></span>
7     </button>
8     <div class="collapse navbar-collapse" id="navbarNavAltMarkup">
9       <div class="navbar-nav">
10        <a class="nav-link active" aria-current="page" href="/">Home</a>
11        <a class="nav-link" href="/pages/about">About</a>
12        <a class="nav-link" href="/pages/contact">Contact</a>
13      </div>
14    </div>
15  </div>
16 </div>
17 </nav>
```

Kemudian jika sudah kita juga perlu menasosiasikan file template dengan navbar tersebut untuk dapat digunakan di setiap halamannya.

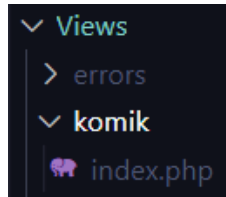


```
template.php X contact.php home.php Pages.php navbar.php about.php
app > Views > layout > template.php > html > body
1 <!doctype html>
2 <html lang="en">
3
4 <head>
5   <!-- Required meta tags -->
6   <meta charset="utf-8">
7   <meta name="viewport" content="width=device-width, initial-scale=1">
8
9   <!-- Bootstrap CSS -->
10  <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.1.0/dist/css/bootstrap.min.css" rel="stylesheet" integrity="sha384-KyZXEAg3QhqlM
11
12  <title><?= $title; ?></title>
13 </head>
14
15 <body>
16   <?= $this->include('/layout/navbar'); ?>
17   <?= $this->renderSection('content'); ?>
18
19   <!-- Optional JavaScript; choose one of the two! -->
20
21   <!-- Option 1: Bootstrap Bundle with Popper -->
22   <script src="https://cdn.jsdelivr.net/npm/bootstrap@5.1.0/dist/js/bootstrap.bundle.min.js" integrity="sha384-U1DAWAznBHeqEITlVSCgzc+c9
23
24   <!-- Option 2: Separate Popper and Bootstrap JS -->
25   <!--
26   <script src="https://cdn.jsdelivr.net/npm/@popperjs/core@2.9.3/dist/umd/popper.min.js" integrity="sha384-mNCCOe7tC1dHqGowe/6oWVemdAV
27   <script src="https://cdn.jsdelivr.net/npm/bootstrap@5.1.0/dist/js/bootstrap.min.js" integrity="sha384-cn7L7gDp0eyniUwwA2grzD06kc/tftf
28   -->
29 </body>
30
31 </html>
```

Perintah `<?= $this->include('/layout/navbar'); ?>` digunakan untuk mengarahkan ke file navbar. Dengan teknik ini diharapkan pembuatan navbar akan lebih efisien dan efektif terhadap semua halaman web.

## Model.

- ✓ Models di dalam CI4 menyediakan sebuah cara untuk berinteraksi dengan data yang terdapat di dalam database anda. Cara yang disediakan dapat untuk Create, Read, Update dan Delete data (CRUD).
- ✓ Praktik WEB Unpas.
  1. Buat file di dalam folder views/folder komik bernama index.php.



Berikut adalah isi file Views/komik/index.php

```
Komik.php Pages.php template.php index.php X navbar.php
app > Views > komik > index.php > ...
1 <?=$this->extend('/layout/template'); ?>
2 <?=$this->section('content'); ?>
3 <div class="container">
4   <div class="row">
5     <div class="col">
6       <table class="table">
7         <thead>
8           <tr>
9             <th scope="col">#</th>
10            <th scope="col">First</th>
11            <th scope="col">Last</th>
12            <th scope="col">Handle</th>
13          </tr>
14        </thead>
15        <tbody>
16          <tr>
17            <th scope="row">1</th>
18            <td>Mark</td>
19            <td>Otto</td>
20            <td>@mdo</td>
21          </tr>
22          <tr>
23            <th scope="row">2</th>
24            <td>Jacob</td>
25            <td>Thornton</td>
26            <td>@fat</td>
27          </tr>
28          <tr>
29            <th scope="row">3</th>
30            <td colspan="2">Larry the Bird</td>
31            <td>@twitter</td>
32          </tr>
33        </tbody>
34      </table>
35    </div>
36  </div>
37 </div>
38 <?=$this->endSection(); ?>
```

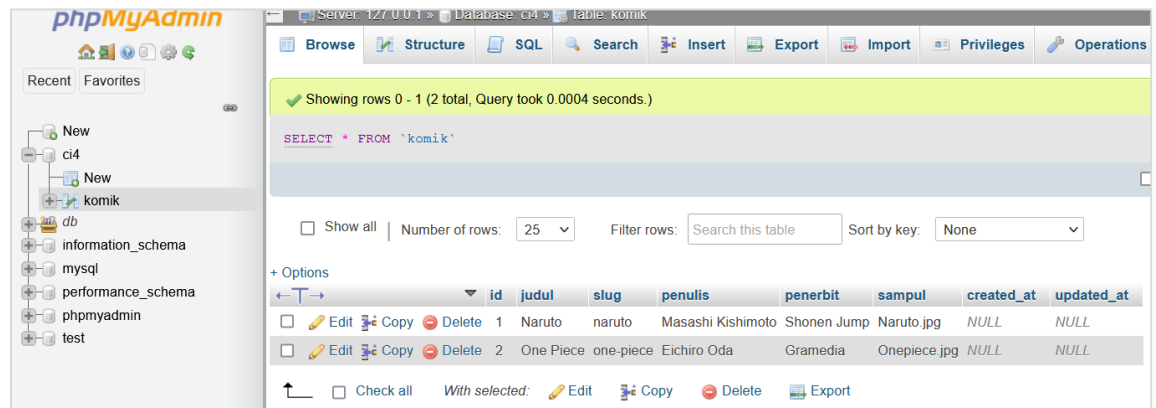
2. Buat file controller di dalam folder Controllers/Komik.php  
Berikut adalah isi dari file Komik.php

```
EXPLORER
KODING-CI4APP
  app
    Config
    Controllers
      BaseController.php
      Home.php
      Komik.php
      Pages.php

app > Controllers > Komik.php > ...
1 <?php
2
3 namespace App\Controllers;
4
5 class Komik extends BaseController
6 {
7   public function index()
8   {
9     $data = [
10       'title' => 'Daftar Komik '
11     ];
12     return view('komik/index', $data);
13   }
14 }
```

### 3. Bekerja dengan database.

- ✓ Buat sebuah database.



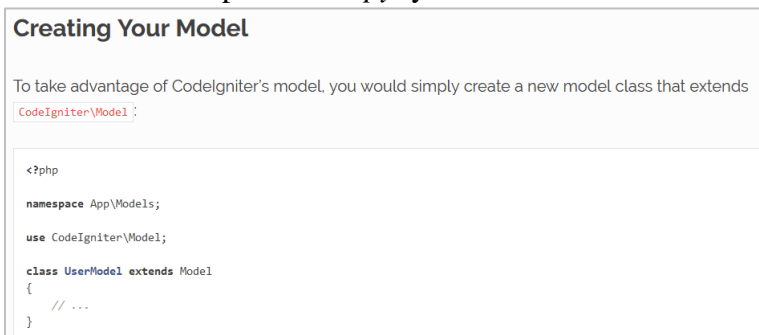
- ✓ Konfigurasi database pada file .env

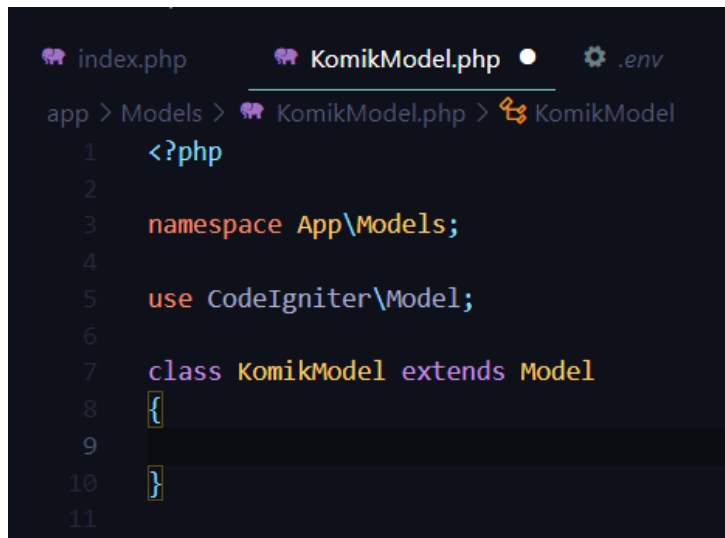
```
36 #-----
37 # DATABASE
38 #-----
39
40 database.default.hostname = localhost
41 database.default.database = ci4
42 database.default.username = root
43 database.default.password =
44 database.default.DBDriver = MySQLi
45 database.default.DBPrefix =
46
47 # database.tests.hostname = localhost
48 # database.tests.database = ci4
49 # database.tests.username = root
50 # database.tests.password = root
51 # database.tests.DBDriver = MySQLi
52 # database.tests.DBPrefix =
53
```

- ✓ Buat model untuk merepresentasikan database.

Cara buatnya pada folder app/models/KomikModel.php

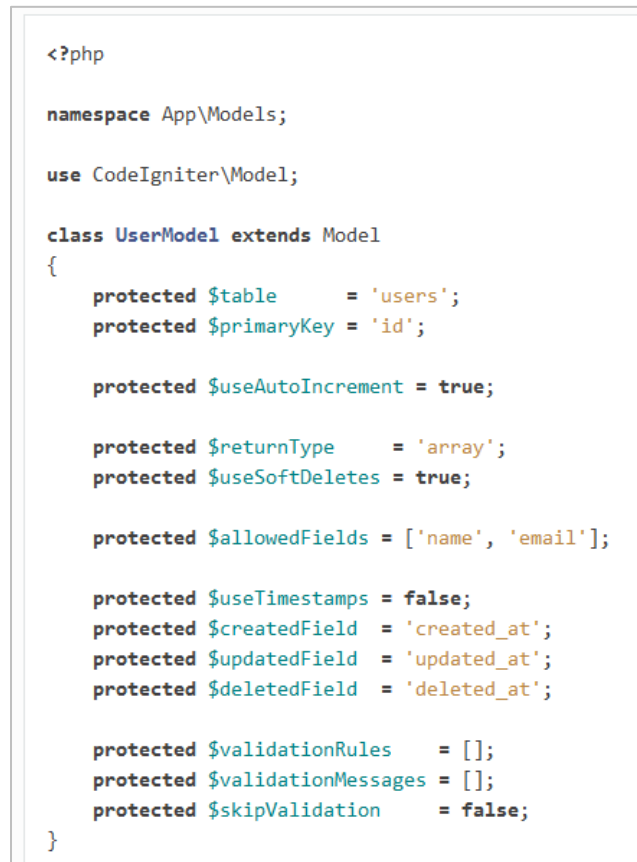
Untuk template isian awal kita dapat lihat pada dokumentasi WEB CI4/Models, kemudian kita dapat men-copynya.





```
index.php KomikModel.php .env
app > Models > KomikModel.php > KomikModel
1 <?php
2
3 namespace App\Models;
4
5 use CodeIgniter\Model;
6
7 class KomikModel extends Model
8 {
9
10 }
11
```

Jika dilihat isi dari model masih kosong, apa saja yang kiranya dapat diisikan ke dalam model?. Kita dapat melihatnya pada file dokumentasi Models di WEB CI4.



```
<?php

namespace App\Models;

use CodeIgniter\Model;

class UserModel extends Model
{
    protected $table = 'users';
    protected $primaryKey = 'id';

    protected $useAutoIncrement = true;

    protected $returnType = 'array';
    protected $useSoftDeletes = true;

    protected $allowedFields = ['name', 'email'];

    protected $useTimestamps = false;
    protected $createdField = 'created_at';
    protected $updatedField = 'updated_at';
    protected $deletedField = 'deleted_at';

    protected $validationRules = [];
    protected $validationMessages = [];
    protected $skipValidation = false;
}
```

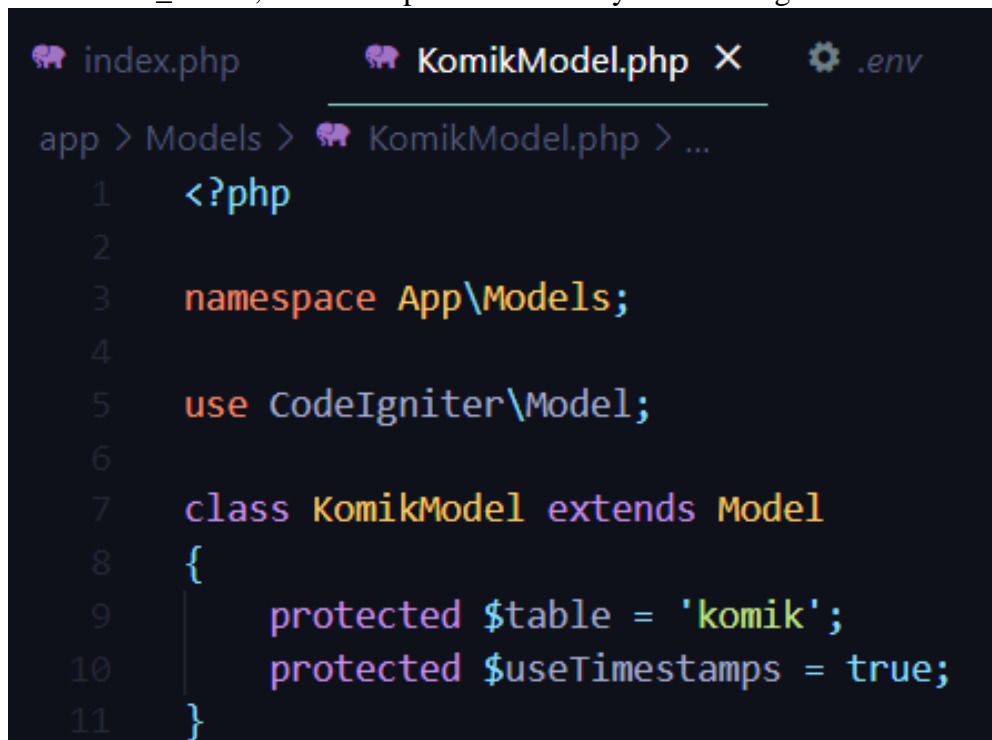
Isian tersebut menunjukkan syntax konfigurasi yang harus diisikan, namun tidak semuanya karena jika tidak diisikanpun tidak apa asalkan masih ada nilai defaultnya misal `protected $primaryKey = 'id'`. Namun jika kita konfigurasi dengan database yang kita buat maka nama dari setiap syntac default harus kita timpa dengan nama file kita. Berikut gambar lebih terperinci.

```

1  <?php
2
3  namespace App\Models;
4
5  use CodeIgniter\Model;
6
7  class KomikModel extends Model
8  {
9      protected $table = 'komik';
10     /*nama table harus disamakan dengan yg ada
11     | pada db kita, huruf besar kecilnya juga*/
12 }

```

Selanjutnya kita juga perlu lihat file parents dari extends Model. Untuk mencarinya kita dapat menekan tombol kombinasi ctrl+p, lalu ketik Model.php. Di file Model.php kita hanya untuk melihat mana file yang harus kita isi dan mana yang boleh dibiarkan, jangan ubah apapun!. Jika tidak ada nilai defaultnya maka harus kita isikan secara manual. Jika terdapat nilai defaultnya, namun berbeda dengan nama yang terdapat pada db kita maka kita mau tidak mau harus merubahnya contoh `protected $primaryKey = 'id'` sedangkan punya kita bernama `id_komik`, maka kita perlu membuatnya di Model gambar di atas.

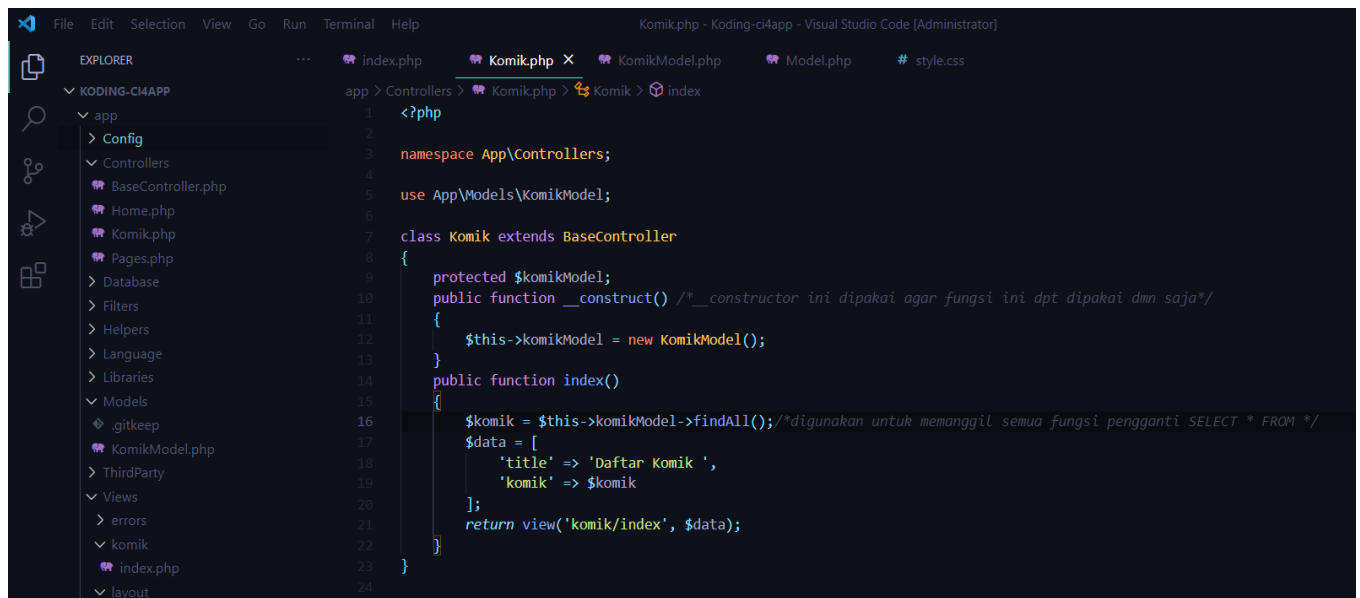


```

index.php  KomikModel.php X  .env
app > Models > KomikModel.php > ...
1  <?php
2
3  namespace App\Models;
4
5  use CodeIgniter\Model;
6
7  class KomikModel extends Model
8  {
9      protected $table = 'komik';
10     protected $useTimestamps = true;
11 }

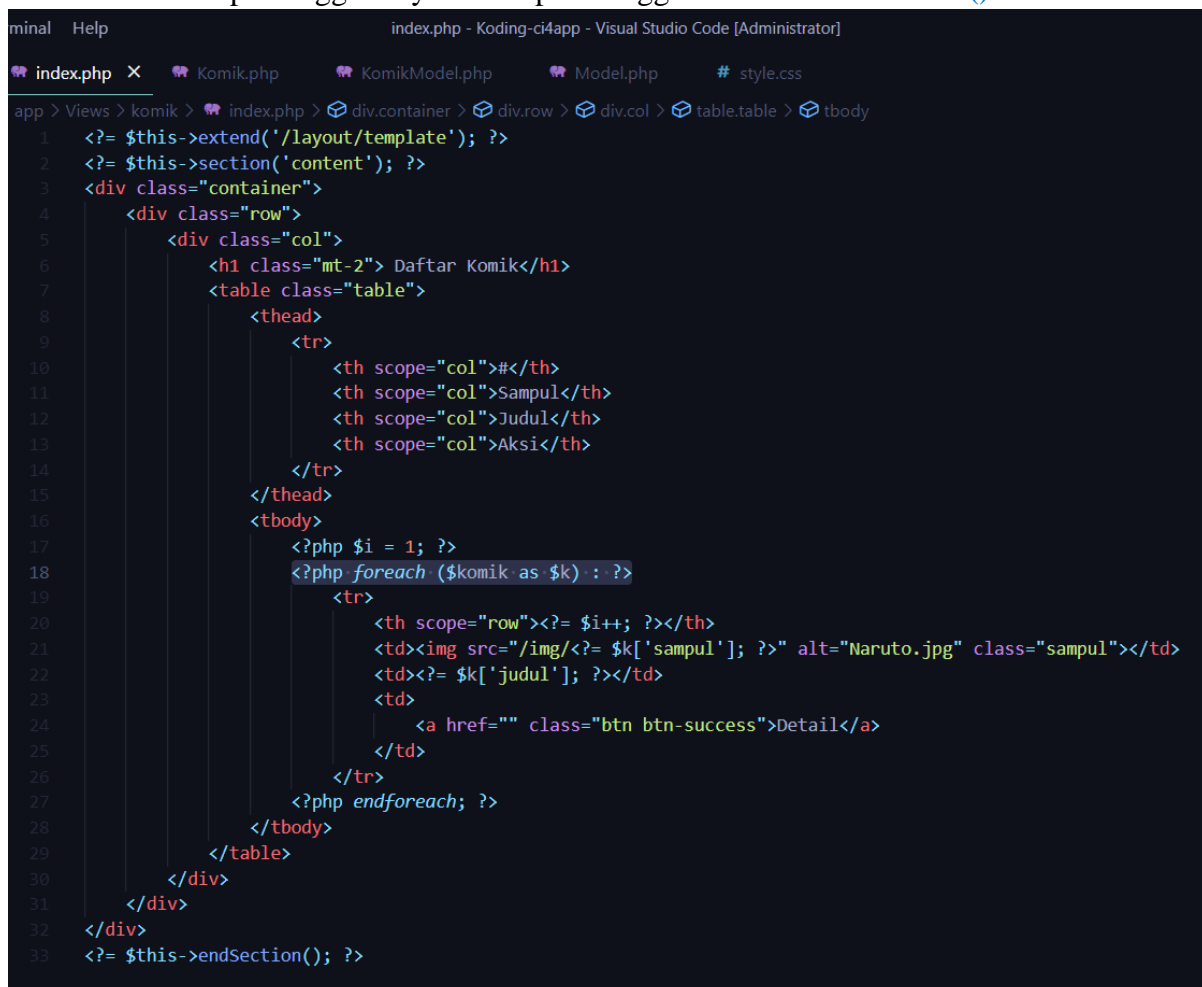
```

- ✓ Menghubungkan database melalui controllers.



```
1 <?php
2
3 namespace App\Controllers;
4
5 use App\Models\KomikModel;
6
7 class Komik extends BaseController
8 {
9     protected $komikModel;
10    public function __construct() /* __constructor ini dipakai agar fungsi ini dpt dipakai dmnn saja */
11    {
12        $this->komikModel = new KomikModel();
13    }
14    public function index()
15    {
16        $komik = $this->komikModel->findAll(); /* digunakan untuk memanggil semua fungsi pengganti SELECT * FROM */
17        $data = [
18            'title' => 'Daftar Komik ',
19            'komik' => $komik
20        ];
21        return view('komik/index', $data);
22    }
23 }
```

Kemudian untuk pemanggilannya kita dapat menggunakan metode `foreach()`.



```
1 <?php $this->extend('/layout/template'); ?>
2 <?php $this->section('content'); ?>
3 <div class="container">
4     <div class="row">
5         <div class="col">
6             <h1 class="mt-2">Daftar Komik</h1>
7             <table class="table">
8                 <thead>
9                     <tr>
10                        <th scope="col">#</th>
11                        <th scope="col">Sampul</th>
12                        <th scope="col">Judul</th>
13                        <th scope="col">Aksi</th>
14                    </tr>
15                </thead>
16                <tbody>
17                    <?php $i = 1; ?>
18                    <?php foreach ($komik as $k) : ?>
19                        <tr>
20                            <th scope="row"><?php $i++; ?></th>
21                            <td></td>
22                            <td><?php $k['judul']; ?></td>
23                            <td>
24                                <a href="" class="btn btn-success">Detail</a>
25                            </td>
26                        </tr>
27                    <?php endforeach; ?>
28                </tbody>
29            </table>
30        </div>
31    </div>
32 </div>
33 <?php $this->endSection(); ?>
```

- ✓ Manipulasi URL WEB yang diakses agar menyembunyikan routesnya.

Sebelum dimanipulasi bentuknya ialah seperti ini,

localhost:8080/komik/detail/naruto

Kita ingin mengubahnya menjadi cukup seperti ini,

localhost:8080/komik/naruto

- ✓ Maka, kita dapat membajak Routes nya seperti berikut,

```
$routes->get('/', 'Pages::index');  
$routes->get('/komik/(:segment)', 'Komik::detail/$1');
```

dapat dibaca, CI4 tolong

buatkan routes dengan method get jika ada user yang mengakses /komik/ apapun (kita ambil nilai apapun itu dengan segment, supaya slashnya tidak terbawa) kemudian nilai si apapun itu kita ambil dan arahkan ke Controller Komik Methodnya detail dengan mengirimkan segment tadi ke \$1.

- ✓ Buat method baru di dalam Controllers Komik.php.

Method baru Bernama detail yang akan menerima parameter \$slug.

```
Routes.php Komik.php X detail.php KomikModel.php index.php  
app > Controllers > Komik.php > ...  
1 <?php  
2  
3 namespace App\Controllers;  
4  
5 use App\Models\KomikModel;  
6  
7 class Komik extends BaseController  
8 {  
9     protected $komikModel;  
10    public function __construct() /*__constructor ini dipakai agar fungsi ini dpt dipakai dmn saja*/  
11    {  
12        $this->komikModel = new KomikModel();  
13    }  
14  
15    public function index()  
16    {  
17        //$komik = $this->komikModel->findAll();/*digunakan untuk memanggil semua fungsi pengganti SELECT * FROM */  
18        $data = [  
19            'title' => 'Daftar Komik ',  
20            'komik' => $this->komikModel->getKomik() /*Panggil komikModel lalu panggil-  
21            method getKomik yang terdapat di dalamnya. getKomik tidak memakai parameter karena akan difindAll*/  
22        ];  
23        return view('komik/index', $data);  
24    }  
25  
26    public function detail($slug)  
27    {  
28        $data = [  
29            'title' => 'Detail Komik',  
30            'komik' => $this->komikModel->getKomik($slug)/*ini juga sama seperti index di atas-  
31            namun getKomik pakai parameter karena hanya akan menampilkan slug*/  
32        ];  
33        return view('komik/detail', $data);  
34    }  
35 }  
36
```

\*Fokus pada public function index dan detail.

- ✓ Buat method baru di dalam models untuk menghubungkan ke database dan mengambil seluruh data berdasarkan slug.

```
app > Models > KomikModel.php > ...
1  <?php
2
3  namespace App\Models;
4
5  use CodeIgniter\Model;
6
7  class KomikModel extends Model
8  {
9      protected $table = 'komik';
10     protected $useTimestamps = true;
11
12     public function getKomik($slug = false) /*default parameter false, jika ada parameter cari pakai-
13     yang where jika tidak ambil semua data komik*/
14     {
15         if ($slug == false) {
16             return $this->findAll();
17         } /*jika slug false, maka findall*/ else
18             return $this->where(['slug' => $slug])->first();
19     } /*jika ada slug tampilkan*/
20 }
21
```

- ✓ Buat view dengan nama detail.php

```
EXPLODER
KODING-CI4APP
app
  Config
  Controllers
    BaseController.php
    Home.php
    Komik.php
    Pages.php
  Database
  Filters
  Helpers
  Language
  Libraries
  Models
    KomikModel.php
  ThirdParty
  Views
    errors
    komik
      detail.php
      index.php
    layout
      navbar.php
      template.php

app > Views > komik > detail.php > div.container > div.row > div.col > div.card.mb-3 > div.row.g-0 > div.col-md-8 > div.card-body > br
1  <?= $this->extend('/layout/template'); ?>
2  <?= $this->section('content'); ?>
3  <div class="container">
4      <div class="row">
5          <div class="col">
6              <h2 class="mt-2">Detail Komik</h2>
7              <div class="card mb-3" style="max-width: 540px;">
8                  <div class="row g-0">
9                      <div class="col-md-4">
10                     
11                     </div>
12                     <div class="col-md-8">
13                         <div class="card-body">
14                             <h5 class="card-title"><?= $komik['judul']; ?></h5>
15                             <p class="card-text"><b>Penulis : <?= $komik['penulis']; ?></b></p>
16                             <p class="card-text"><small class="text-muted"><b>Penerbit : <?= $komik['penerbit']; ?></b></small></p>
17                             <a href="" class="btn btn-warning">Edit</a>
18                             <a href="" class="btn btn-danger">Delete</a><br><br>
19                             <a href="/komik">Kembali ke daftar komik</a>
20                         </div>
21                     </div>
22                 </div>
23             </div>
24         </div>
25     </div>
26 </div>
27 <?= $this->endSection(); ?>
```

\*terdapat perintah `<?= $komik['...']; ?>` , itu dapat langsung dipakai karena sebelumnya sudah dideklarasikan di dalam file controller Komik.php

```
25
26 public function detail($slug)
27 {
28     $data = [
29         'title' => 'Detail Komik',
30         'komik' => $this->komikModel->getKomik($slug) /*ini juga sama seperti index di atas-
31         namun getKomik pakai parameter karena hanya akan menampilkan slug*/
32     ];
33     return view('komik/detail', $data);
34 }
35
```



## ✚ Insert Data.

- ✓ Pertama kita buat sebuah Controllers yang akan mengarahkan ke halaman Create jika ada user yang mengakses halaman tambah data komik melalui halaman views/komik/index.php .
  - Controllers/Komik.php kita buat controller

```
35
36     public function create()
37     {
38         $data = [
39             'title' => 'Form Tambah Data Komik'
40         ];
41         return view('komik/create', $data);
42     }
43 }
```

- ✓ Kemudian pada views/komik/index.php kita buat sebuah :a href yang akan mengarahkan ke halaman views/komik/create.php .

```
<a href="/komik/create" class="btn btn-primary mb-3">Tambah Data Komik</a>
```

- ✓ Buat halaman create.php

```
app > Views > komik > create.php > div.container > div.col-8 > div.row > form
1 <?= $this->section('content'); ?>
2 <div class="container">
3     <div class="col-8">
4         <div class="row">
5             <h2 class="my-3">Form Tambah Data Komik</h2>
6             <form action="/komik/save" method="POST">
7                 <?= csrf_field(); /*ini adalah fitur keamanan terbaru dari CI4 berfungsi u/ menghindari pembajakan input form dr hacker-
8                 jadi form ini hanya akan dapat diisi dari form ini saja tdk dpt dr hal lain. */ ?>
9                 <div class="row mb-3">
10                     <label for="judul" class="col-sm-2 col-form-label">Judul</label>
11                     <div class="col-sm-10">
12                         <input type="text" class="form-control" id="judul" name="judul" autofocus>
13                     </div>
14                 </div>
15                 <div class="row mb-3">
16                     <label for="penulis" class="col-sm-2 col-form-label">Penulis</label>
17                     <div class="col-sm-10">
18                         <input type="text" class="form-control" id="penulis" name="penulis">
19                     </div>
20                 </div>
21                 <div class="row mb-3">
22                     <label for="penerbit" class="col-sm-2 col-form-label">Penerbit</label>
23                     <div class="col-sm-10">
24                         <input type="text" class="form-control" id="penerbit" name="penerbit">
25                     </div>
26                 </div>
27                 <div class="row mb-3">
28                     <label for="sampul" class="col-sm-2 col-form-label">Sampul</label>
29                     <div class="col-sm-10">
30                         <input type="text" class="form-control" id="sampul" name="sampul">
31                     </div>
32                 </div>
33                 <button type="submit" class="btn btn-primary">Tambah Data</button>
34             </form>
35         </div>
36     </div>
37 </div>
38 <?= $this->endSection(); ?>
```

- ✓ Buat juga routes yang akan mengarahkan ke halaman tambah data/create.php tsb. Pada kasus tertentu jika ini tidak dibuat maka halaman tidak akan load menu tambah data dan malah akan megkases halaman detail komik.

```
36 $routes->get('/', 'Pages::index');
37 $routes->get('/komik/create', 'komik::create');
38 $routes->get('/komik/{segment}', 'Komik::detail/{1}');
```

- ✓ Selanjutnya kita buat sebuah controller baru yaitu untuk menyimpan record data di Controllers/Komik.php.

```
47
48     public function save()
49     {
50         /**dd($this->request->getVar()); */method baru ci4 yg akan mengambil
51         data apapun yang akan diambil dari form */
52         //perintah berikut digunakan untuk mengolah slug (One Piece = one-piece)
53         $slug = url_title($this->request->getVar('judul'), '-', true);
54         //berikut adalah cara insert ke db dengan konfigurasi model
55         $this->komikModel->save([
56             'judul' => $this->request->getVar('judul'),
57             'slug' => $slug,
58             'penulis' => $this->request->getVar('penulis'),
59             'penerbit' => $this->request->getVar('penerbit'),
60             'sampul' => $this->request->getVar('sampul')
61         ]);
62         /*Flashdata = pesan yg sekali muncul setelah tambah data */
63         session()->setFlashdata('pesan', 'Data berhasil ditambahkan');
64         return redirect()->to('/komik'); //perintah untuk redirect halaman
65     }
66 }
67
```

Pada controller detail kita juga buat sebuah pengkondisian yang akan memproses ke halaman 404 Page Not Found.

```
25
26     public function detail($slug)
27     {
28         $data = [
29             'title' => 'Detail Komik',
30             'komik' => $this->komikModel->getKomik($slug) /*ini juga sama seperti index di atas-
31             namun getKomik pakai parameter karena hanya akan menampilkan slug*/
32         ];
33         //jika komik(slug) tidak ada di tabel
34         if (empty($data['komik'])) {
35             throw new \CodeIgniter\Exceptions\PageNotFoundException('Judul komik ' . $slug . ' tidak ditemukan');
36         }
37         return view('komik/detail', $data);
38     }

```

- ✓ Selanjutnya kita modifikasi isi model/komikModel.php sbb.

```
Komik.php  KomikModel.php X  detail.php  create.php  index.php
app > Models > KomikModel.php > ...
1  <?php
2
3  namespace App\Models;
4
5  use CodeIgniter\Model;
6
7  class KomikModel extends Model
8  {
9      protected $table = 'komik';
10     protected $useTimestamps = true;
11     //berikut adalah perintah yg memberi tahu isi database mana saja yg kita boleh isi manual
12     protected $allowedFields = ['judul', 'slug', 'penulis', 'penerbit', 'sampul'];
13
14     public function getKomik($slug = false) /*default parameter false, jika ada parameter cari pakai-
15     yang where jika tidak ambil semua data komik*/
16     {
17         if ($slug == false) {
18             return $this->findAll();
19         } /*jika slug false, maka findall*/ else
20         return $this->where(['slug' => $slug])->first();
21     } /*jika ada slug tampilkan*/
22 }
23
```

- ✓ Terakhir di dalam views/komik/index.php kita akan membuat sebuah pesan flash yang apabila kita tambah sebuah data ke dlm db akan muncul pesan berhasil.

```

7      <h1 class="mt-2"> Daftar Komik</h1>
8      <?php if (session()->getFlashdata('pesan')) : ?>
9          <div class="alert alert-success" role="alert">
10             <?= session()->getFlashdata('pesan'); ?>
11          </div>
12      <?php endif; ?>
13      <table class="table">

```

Dengan menghubungkannya pada syntax program session pada file controllers Komik.php berikut.. (lihat pada pointer)

```

47
48     public function save()
49     {
50         /*//dd($this->request->getVar()); /*method baru ci4 yg akan mengambil
51         data apapun yang akan diambil dari form */
52         //perintah berikut digunakan untuk mengolah slug (One Piece = one-piece)
53         $slug = url_title($this->request->getVar('judul'), '-', true);
54         //berikut adalah cara insert ke db dengan konfigurasi model
55         $this->komikModel->save([
56             'judul' => $this->request->getVar('judul'),
57             'slug' => $slug,
58             'penulis' => $this->request->getVar('penulis'),
59             'penerbit' => $this->request->getVar('penerbit'),
60             'sampul' => $this->request->getVar('sampul')
61         ]);
62         /*Flashdata = pesan yg sekali muncul setelah tambah data */
63         session()->setFlashdata('pesan', 'Data berhasil ditambahkan');
64         return redirect()->to('/komik'); //perintah untuk redirect halaman
65     }
66 }

```

## Validations.

Digunakan untuk memvalidasi data inputan agar tidak terjadi kesamaan data.

- ✓ Pertama controller save kita modifikasi seperti berikut.

```

50     public function save()
51     {
52         //Validasi inputan
53         //berikut adl. syntax rule u/ pengkondisian apabila terdpt data yg sama diinputkan
54         if (!$this->validate([
55             //--> validasi sederhana 'judul' => 'required|is_unique[komik.judul]' /*berarti judul wajib diisi */
56             //validasi lebih kompleks
57             'judul' => [
58                 'rules' => 'required|is_unique[komik.judul]',
59                 'errors' => [
60                     'required' => '{field} komik harus diisi',
61                     'is_unique' => '{field} komik sudah terdaftar'
62                 ]
63             ]
64         ])) {
65             //menampilkan hasil rule
66             $validation = \Config\Services::validation();
67             return redirect()->to('komik/create')->withInput()->with('validation', $validation);
68         }
69
70         /*//dd($this->request->getVar()); /*method baru ci4 yg akan mengambil
71         data apapun yang akan diambil dari form */
72
73         //perintah berikut digunakan untuk mengolah slug (One Piece = one-piece)
74         $slug = url_title($this->request->getVar('judul'), '-', true);
75
76         //berikut adalah cara insert ke db dengan konfigurasi model
77         $this->komikModel->save([
78             'judul' => $this->request->getVar('judul'),
79             'slug' => $slug,
80             'penulis' => $this->request->getVar('penulis'),
81             'penerbit' => $this->request->getVar('penerbit'),
82             'sampul' => $this->request->getVar('sampul')
83         ]);
84     }

```

- ```
39  
40 public function create()  
41 {  
42     //session();  
43     $data = [  
44         'title' => 'Form Tambah Data Komik',  
45         'validation' => \Config\Services::validation()  
46     ];  
47     return view('komik/create', $data);  
48 }  
49
```

```

48 public function initController(RequestInterface $request, ResponseInterface $response, LoggerInterface $logger)
49 {
50     // Do Not Edit This Line
51     parent::initController($request, $response, $logger);
52     session();
53     //-----

```

- ```

Komik.php BaseController.php create.php X
app > Views > komik > create.php > div.container > div.col-8 > div.row > form
4
5 ss="row">
6 class="my-3">Form Tambah Data Komik</h2>
7 action="/komik/save" method="POST">
8 <?= csrf_field(); /*ini adalah fitur keamanan terbaru dari CI4 berfungsi w/ menghindari pembajakan input form dr hacker-
9 jadi form ini hanya akan dapat diisi dari ini saja tdk dpt dr hal lain. */ ?>
10 <div class="row mb-3">
11 ... <label for="judul" class="col-sm-2 col-form-label">Judul</label>
12 ... <div class="col-sm-10">
13 ... <input type="text" class="form-control" <?=( $validation->hasError('judul')) ? 'is-invalid' : ''; ?> id="judul" name="judul" autofocus value=<?= old('judul'); ?>
14 ... <div id="validationServer05feedback" class="invalid-feedback">
15 ... <?=$validation->getError('judul'); ?>
16 ... </div>
17 ... </div>
18 </div>
19 <div class="row mb-3">
20 <label for="penulis" class="col-sm-2 col-form-label">Penulis</label>
21 <div class="col-sm-10">
22 | <input type="text" class="form-control" id="penulis" name="penulis" value=<?= old('penulis'); ?>
23 </div>
24 </div>
25 <div class="row mb-3">
26 <label for="penerbit" class="col-sm-2 col-form-label">Penerbit</label>
27 <div class="col-sm-10">
28 | <input type="text" class="form-control" id="penerbit" name="penerbit" value=<?= old('penerbit'); ?>
29 </div>
30 </div>
31 <div class="row mb-3">
32 <label for="sampul" class="col-sm-2 col-form-label">Sampul</label>
33 <div class="col-sm-10">
34 | <input type="text" class="form-control" id="sampul" name="sampul" value=<?= old('sampul'); ?>
35 </div>
36 </div>
37 <button type="submit" class="btn btn-primary">Tambah Data</button>
38 rm>

```

## ✚ Delete & Edit data.

### A. Delete.

- ✓ Pertama kita buat controllers delete terlebih dahulu.

```
90 public function delete($id)
91 {
92     $this->komikModel->delete($id);
93     session()->setFlashdata('pesan', 'Data berhasil dihapus');
94     return redirect()->to('/komik');
95 }
```

Tunjuk modelnya kemudian delete berdasarkan parameter.

- ✓ Selanjutnya di dalam views detail.php kita buat method delete. Method ini menggunakan teknik yang dinamakan http spoofing, teknik ini digunakan untuk menghindari delete random berdasarkan alamat yang dimunculkan di URL.

```
18 <!-- HTTP Spoofing -->
19 <form action="/komik/<?= $komik['id']; ?>" method="post" class="d-inline">
20     <?= csrf_field(); ?>
21     <input type="hidden" name="_method" value="DELETE">
22     <button type="submit" class="btn btn-danger" onclick="return confirm('Apakah anda yakin?');">Delete</butt
23 </form><br>
24 <!-- Sampai sini -->
```

- ✓ Buat routes baru yang akan mengarahkan apabila ada proses penghapusan data.

```
35 // $routes->get('/', 'Home::index'); --> default
36 $routes->get('/', 'Pages::index');
37 $routes->get('/komik/create', 'komik::create');
38 $routes->delete('/komik/(:num)', 'Komik::delete/$1');
39 $routes->get('/komik/(:any)', 'Komik::detail/$1');
40 /*
```

Perintah di atas dapat dibaca, CI4 tolong buatlah sebuah route atau jalur khusus yang methodnya requestnya delete (method delete ini didapat dari HTTP spoofing di atas) kemudian arahkan ke views komik ambil nilai yang terdapat di URL khusus yang dalam bentuk angka lalu arahkan ke Controllers Komik methodnya delete dan ambil id nya (\$1).

### B. Edit.

- ✓ Hyperlink ke halaman detail ke views/menu edit.php

```
<p class="card-text"><small class="text-muted"><b>Penerbit : <?= $komik['penerbit']; ?></b></small></p>
<a href="/komik/edit/<?= $komik['slug']; ?>" class="btn btn-warning">Edit</a>
<!-- HTTP Spoofing -->
```

Menu tersebut kita juga arahkan ke slug dari item yang dipilih, sehingga dibelakang syntax terdapat php.

- ✓ Buat sebuah controller yang bernama edit.

```
96 public function edit($slug)
97 {
98     $data = [
99         'title' => 'Form Ubah Data Komik',
100         'validation' => \Config\Services::validation(),
101         'komik' => $this->komikModel->getKomik($slug)
102     ];
103     return view('komik/edit', $data);
104 }
```

Controller ini akan menerima parameter dari slug yang terdapat pada database.

- ✓ Buat views untuk menu edit ini.

```

Komik.php KomikModel.php edit.php detail.php Routes.php
D:\xampp\htdocs> Project-ku > Kodiny-CI4App > app > Views > komik > edit.php
1 <?=$this->extend('layout/template');?>
2 <?=$this->section('content');?>
3 <div class="container">
4 <div class="col-8">
5 <div class="row">
6 <div class="my-3">Form Ubah Data Komik</div>
7 <form action="/komik/update/<?=$komik['id'];?>" method="POST">
8 <?=$csrf_field();?>
9 <input type="hidden" name="slug" value="<?=$komik['slug'];?>">
10 <div class="row mb-3">
11 <div class="col-sm-10">
12 <label for="judul" class="col-sm-2 col-form-label">Judul</label>
13 <div class="col-sm-10">
14 <input type="text" class="form-control" <?=$validation->hasError('judul')? 'is-invalid' : ''> id="judul" name="judul" autofocus value="<?=$komik['judul'];?>">
15 <div id="validationServer05feedback" class="invalid-feedback">
16 <?=$validation->getError('judul');?>
17 </div>
18 </div>
19 </div>
20 <div class="row mb-3">
21 <label for="penulis" class="col-sm-2 col-form-label">Penulis</label>
22 <div class="col-sm-10">
23 <input type="text" class="form-control" id="penulis" name="penulis" value="<?=(old('penulis'))? old('penulis') : $komik['penulis'];?>">
24 </div>
25 </div>
26 <div class="row mb-3">
27 <label for="penerbit" class="col-sm-2 col-form-label">Penerbit</label>
28 <div class="col-sm-10">
29 <input type="text" class="form-control" id="penerbit" name="penerbit" value="<?=(old('penerbit'))? old('penerbit') : $komik['penerbit'];?>">
30 </div>
31 </div>
32 <div class="row mb-3">
33 <label for="sampul" class="col-sm-2 col-form-label">Sampul</label>
34 <div class="col-sm-10">
35 <input type="text" class="form-control" id="sampul" name="sampul" value="<?=(old('sampul'))? old('sampul') : $komik['sampul'];?>">
36 </div>
37 </div>
38 <button type="submit" class="btn btn-primary">Simbah Data</button>
39 </div>
40 </div>

```

Di form ini kita juga atur untuk mengarahkan ke controllers komik yang methodnya adalah update dan kite juga kasih tau ke si program agar mengambil parameter indexnya.

```
<form action="/komik/update/<?=$komik['id'];?>" method="POST">
```

- ✓ Buat controllers Bernama update yang akan memproses update data.

```

Komik.php KomikModel.php edit.php detail.php Routes.php
app > Controllers > Komik.php Komik > update
1095 public function update($id)
1096 {
1097     $komikLama = $this->komikModel->getKomik($this->request->getVar('slug'));
1098     if ($komikLama['judul'] == $this->request->getVar('judul')) {
1099         $rule_judul = 'required';
1100     } else {
1101         $rule_judul = 'required|is_unique[komik.judul]';
1102     }
1103     if (!$this->validate([
1104         //-> validasi sederhana 'judul' => 'required|is_unique[komik.judul]' /*berarti judul wajib diisi */
1105         //validasi lebih kompleks
1106         'judul' => [
1107             'rules' => $rule_judul,
1108             'errors' => [
1109                 'required' => '{field} komik harus diisi',
1110                 'is_unique' => '{field} komik sudah terdaftar'
1111             ]
1112         ]
1113     ])) {
1114         //menampilkan hasil rule
1115         $validation = \Config\Services::validation();
1116         return redirect()->to('komik/edit/' . $this->request->getVar('slug'))->withInput()->with('validation', $validation);
1117     }
1118
1119     $slug = url_title($this->request->getVar('judul'), '-', true);
1120     $this->komikModel->save([
1121         'id' => $id,
1122         'judul' => $this->request->getVar('judul'),
1123         'slug' => $slug,
1124         'penulis' => $this->request->getVar('penulis'),
1125         'penerbit' => $this->request->getVar('penerbit'),
1126         'sampul' => $this->request->getVar('sampul')
1127     ]);
1128     session()->setFlashdata('pesan', 'Data berhasil diubah');
1129     return redirect()->to('komik');
1130 }

```

- ✓ Buat routes yang akan mengarahkan ke menu edit.

```

34
35 // $routes->get('/', 'Home::index'); --> default
36 $routes->get('/', 'Pages::index');
37 $routes->get('/komik/edit/{:segment}', 'Komik::edit/{:1}');
38 $routes->get('/komik/create', 'komik::create');
39 $routes->delete('/komik/{:num}', 'Komik::delete/{:1}');
40 $routes->get('/komik/{:any}', 'Komik::detail/{:1}');
41 /*

```

#### Upload File.

- ✓ Pada file views/create.php kita ubah pada class sampul untuk menjadi file browser. Dan edit beberapa penamaan file persis seperti berikut.

```

<div class="row mb-3">
<div class="col-sm-2 col-form-label">Sampul</div>
<div class="col-sm-10">
<div class="custom-file">
<input type="file" class="custom-file-input" <?=( $validation->hasError('sampul')) ? 'is-invalid' : '' ; > id="sampul" name="sampul">
<div class="invalid-feedback">
<?=( $validation->getError('sampul')) ; >
</div>
<label class="custom-file-label" for="Sampul">Pilih gambar</label>
</div>
</div>
</div>

```

Kita juga wajib menambahkan syntax enctype berikut yang berfungsi sebagai, ini supaya file kita dapat bekerja dengan inputan biasa dengan inputan gambar.

```
<form action="/komik/save" method="POST" enctype="multipart/form-data">
```

- ✓ Modify controllers public function save.

```

50 public function save()
51 {
52     //validasi inputan
53     //berikut adl syntax rule u/ pengkondisian apabila terdpt data yg sama diinputkan
54     if (!$this->validate([
55         //--> validasi sederhana 'judul' => 'required|is_unique[komik.judul]' /*berarti judul wajib diisi */
56         //validasi lebih kompleks
57         'judul' => [
58             'rules' => 'required|is_unique[komik.judul]',
59             'errors' => [
60                 'required' => '{field} komik harus diisi',
61                 'is_unique' => '{field} komik sudah terdaftar'
62             ]
63         ],
64         'sampul' => [
65             'rules' => 'uploaded[sampul]|max_size[sampul,1024]|is_image[sampul]|mime_in[sampul,image/jpeg,image/png]',
66             'errors' => [
67                 'uploaded' => 'Pilih gambar sampul terlebih dahulu',
68                 'max_size' => 'Ukuran gambar terlalu besar',
69                 'is_image' => 'Yang anda pilih bukan gambar',
70                 'mime_in' => 'Yang anda pilih bukan gambar'
71             ]
72         ]
73     ))) {
74         //menampilkan hasil rule
75         // $validation = \Config\Services::validation();
76         // return redirect()->to('komik/create')->withInput()->with('validation', $validation);
77         return redirect()->to('komik/create')->withInput();
78     }
79
80     //syntax kelola gambar
81     $fileSampul = $this->request->getFile('sampul');
82     //pindahkan file ke folder img
83     $fileSampul->move('img');
84     //ambil nama sampul
85     $namaSampul = $fileSampul->getName();
86
87     // $this->save($namaSampul, $judul); //method baru rdt dan akan menambahkan

```

```

Komik.php X create.php
app > Controllers > Komik.php > Komik > save
72     }
73     ))) {
74         //menampilkan hasil rule
75         // $validation = \Config\Services::validation();
76         // return redirect()->to('komik/create')->withInput()->with('validation', $validation);
77         return redirect()->to('/komik/create')->withInput();
78     }
79
80     //syntax kelola gambar
81     $fileSampul = $this->request->getFile('sampul');
82     //pindahkan file ke folder img
83     $fileSampul->move('img');
84     //ambil nama sampul
85     $namaSampul = $fileSampul->getName();
86
87     /*//dd($this->request->getVar()); //method baru ci4 yg akan mengambil
88     data apapun yang akan diambil dari form */
89
90     //perintah berikut digunakan untuk mengolah slug (One Piece = one-piece)
91     $slug = url_title($this->request->getVar('judul'), '-', true);
92
93     //berikut adalah cara insert ke db dengan konfigurasi model
94     $this->komikModel->save([
95         'judul' => $this->request->getVar('judul'),
96         'slug' => $slug,
97         'penulis' => $this->request->getVar('penulis'),
98         'penerbit' => $this->request->getVar('penerbit'),
99         'sampul' => $namaSampul
100     ]);
101
102     /*Flashdata = pesan yg sekali muncul setelah tambah data */
103     session()->setFlashdata('pesan', 'Data berhasil ditambahkan');
104     return redirect()->to('/komik'); //perintah untuk redirect halaman
105 }
106
107 public function delete($id)
108 {
109     $this->komikModel->delete($id);

```