

MEMBANGUN SISTEM INFORMASI

MANAJEMEN LABORATORIUM TERINTEGRASI

Penulis:

Arif Wibisono

Sudargo

Ika Menarianti

Wijayanto

**UNIVERSITAS PGRI
SEMARANG
PRESS**

Penerbit:

UPT Penerbitan Universitas PGRI Semarang Press

**Membangun Sistem Informasi
Manajemen Laboratorium Terintegrasi**

**Penulis:
Arif Wibisono
Sudargo
Ika Menarianti
Wijayanto**

Penerbit:

UPT Penerbitan Universitas PGRI Semarang Press



**Sanksi Pelanggaran Pasal 72
Undang-Undang Nomor 19 Tahun 2002**

1. Barangsiapa dengan sengaja dan tanpa hak melakukan perbuatan sebagaimana dimaksud dalam pasal 2 ayat (1) atau pasal 49 ayat (1) dan ayat (2) dipidana penjara paling singkat 1 (satu) bulan dan/atau denda paling sedikit Rp. 1.000.000,00- (satu juta rupiah) atau paling lama 7 (tujuh) tahun dan/atau denda paling banyak Rp.5.000.000.000,00 (lima miliar rupiah)
2. Barangsiapa dengan sengaja menyiarkan, memamerkan, mengedarkan, atau menjual kepada umum suatu ciptaan dan barang hasil pelanggaran hak cipta atau hak terkait, sebagaimana dimaksud ayat (1) dipidana dengan pidana paling lama 5 (lima) tahun dan/atau denda paling banyak Rp. 500.000.000,00 (lima ratus juta rupiah)

Dilarang keras memfotokopi atau memperbanyak sebagian atau
Seluruh buku ini tanpa seizing tertulis dari penerbit

Membangun Sistem Informasi Manajemen Laboratorium Terintegrasi
ISBN: 978-623-6602-78-2

Penulis:

Arif Wibisono
Sudargo
Ika Menarianti
Wijayanto

Penyunting: Fajar Setiawan

Perancang Sampul dan Penata Letak : Lontar Media

Penerbit:

UPT Penerbitan Universitas PGRI Semarang Press
Jl. Sidodadi Timur No 24, Dr. Cipto Semarang 50125 Jawa Tengah
Telepon: 085640369110

PRAKATA

Alhamdulillah penulis panjatkan ke ilahi robbi, karena atas berkat rahmat dan kuasa-nyalah penulis memiliki kekuatan pemikiran dan motivasi untuk menulis buku *Sistem Informasi Laboratorium Terintegrasi* ini, sehingga terbitan seri pertama dapat terselesaikan sesuai harapan.

Buku ini merupakan jawaban dari pengalaman penulis yang ingin berbagi ilmu dan pengetahuan seputar Pengembangan Sistem Informasi Menggunakan Kerangka Kerja Sistem Informasi yang dapat digunakan untuk mendukung pengelolaan organisasi, khususnya pengelolaan laboratorium. Dalam buku ini dibahas ulasan singkat mengenai Terminologi dan Definisi Sistem Informasi, Konsep PHP dan HTML, Fundamental Framework Codeigniter, Operasi MVC (Model, View dan Controller). Yang kesemuanya dapat digunakan untuk mendukung pengembangan sistem informasi secara cepat.

Semoga buku ini dapat menjadi penuntun bagi yang ingin memulai menekuni bidang sistem informasi utamanya bagi yang ingin memperdalam penguasaan pengembangan sistem informasi menggunakan kerangka kerja (framework). Tidak ada harapan yang lebih besar bagi penulis selain buku ini dapat menjadi manfaat bagi pembacanya.

Penulis menyadari bahwasannya buku ini memiliki kekurangan yang memerlukan penyempurnaan di masa yang akan datang. Oleh karena itu penulis mengharapkan kritik dan saran dari seluruh pembaca. Akhirul kata, semoga buku panduan bekerja ini dapat bermanfaat bagi pembaca secara umum dan penulis secara khusus.

Semarang, Januari 2022

Tim Penulis

DAFTAR ISI

PRAKATA.....	1
DAFTAR ISI.....	2
BAB 1. Mengenal Istilah dalam Sistem Informasi	5
1.1. Terminologi dan Definisi	5
1.1.1. URL.....	5
1.1.2. XAMPP.....	5
1.1.3. Apache	7
1.1.4. MySQL	7
1.1.5. PHP	7
BAB 2. Pengenalan konsep PHP dan HTML.....	9
2.1. Membuat file PHP.....	9
2.1.1. Sintaks Dasar dari PHP.....	9
2.1.2. Struktur Penulisan.....	10
2.2. Membuat file PHP dan HTML.....	11
BAB 3. Fundamental Framework Codeigniter	13
3.1. Instalasi Codeigniter di localhost	13
3.2. Mengenal Konsep MVC.....	15
BAB 4. Membuat Controller, View dan Model.....	18
4.1. Membuat Model di Codeigniter	18
4.1.1. Membuat View di Codeigniter.....	19
4.1.2. Membuat Controller di Codeigniter.....	20
4.2. Membuat dan Menggunakan Helper	22
4.3. Membuat dan Menggunakan Library	24
BAB 5. Upload File Menggunakan Codeigniter	27
5.1. Setting Database di Codeigniter.....	28
5.1.1. Buat Folder dengan nama uploads di project codeigniter.....	30
5.1.2. Membuat Controller Upload.....	30
5.2. Buat file view form_upload.php.....	31
5.2.1. Membuat Function Proses di Controller Upload	32
5.2.2. Menampilkan Data File yang diupload.....	34
5.2.3. Buat View dengan tampil_berkas	36
5.2.4. Buat Function Download pada controller Upload	37

BAB 6.	Cara Membuat Template di Codeigniter	40
6.1.	Mempersiapkan File codeigniter	41
6.1.1.	Setting Bagian Config.....	41
6.1.2.	Membuat Bagian Template.....	41
6.1.3.	Membuat Bagian Header	42
6.1.4.	Membuat bagian footer	44
6.1.5.	Mempersiapkan library JS dan CSS	44
6.1.6.	Buat Controller.....	45
6.1.7.	Membuat File View	46
6.1.8.	Testing tampilan template.....	47
6.2.	Cara Menghubungkan Codeigniter dengan Database MySQL	48
6.2.1.	Buat Database	49
6.2.2.	Setting koneksi di database	50
6.2.3.	Load Library database.....	51
6.2.4.	Buat Model.....	52
6.2.5.	Buat Controller.....	52
6.2.6.	Buat File View	53
BAB 7.	Penanganan Form & Form Validasi di Codeigniter	55
7.1.	Buat Sebuah Controller Siswa.....	55
7.1.1.	Membuat File View dengan nama formSiswa.php.....	55
7.1.2.	Mengakses Inputan Form di Controller	58
7.1.3.	MEMBUAT VALIDASI FORM.....	60
7.1.4.	Testing Form Validation	62
7.2.	Mengenal URI Segment di Codeigniter	63
7.3.	Menghilangkan Index.php pada Codeigniter	67
7.3.1.	Cara untuk menghilangkan index.php di codeigniter	69
7.4.	Cara Menerapkan Konsep HMVC pada Framework Codeigniter	70
7.4.1.	Install Extention Modular	71
7.4.2.	Buat Folder Modules di Folder Application	72
BAB 8.	Membuat CRUD (Create,Read, Update, Delete) di Codeigniter	79
8.1.	Mempersiapkan Database	79
8.1.1.	Mempersiapkan Project Codeigniter.....	83
8.1.2.	Setting File Config.....	84

8.1.3.	Load library dan Helper	85
8.1.4.	Mempersiapkan library CSS & JS untuk tampilan aplikasi.....	85
8.1.5.	Membuat Bagian Template Header & Footer.....	86
8.1.6.	Buat file header.php	86
8.1.7.	Buat File Footer.php	87
8.1.8.	Membuat Fitur Read Data.....	88
8.1.9.	Membuat Form untuk Menambahkan Data	92
8.1.10.	Membuat Method untuk memproses menambahkan data	96
8.1.11.	Membuat Form Edit Data.....	99
8.1.12.	Membuat Method Update untuk proses update data siswa	105
8.1.13.	Membuat Fitur Delete data siswa.....	109
8.1.14.	Hasil Kode Akhir	113
8.2.	Membuat Laporan PDF dengan DOMPdf di Codeigniter.....	120
8.2.1.	Install Dompdf dengan Composer	121
8.2.2.	Membuat Library untuk kebutuhan DomPDF	121
8.2.3.	Persiapkan database	123
8.2.4.	Setting Database di Codeigniter.....	127
8.2.5.	Membuat Model.....	128
8.2.6.	Membuat Controller.....	128
8.2.7.	Membuat file View	129
8.2.8.	Melakukan testing menampilkan laporan PDF	130
	DAFTAR PUSTAKA	131

BAB 1. Mengenal Istilah dalam Sistem Informasi

Teknologi dan manusia saat ini menjadi satu kesatuan dan menjadi dua hal yang tidak dapat dipisahkan. Dimana, kedua faktor tersebut saling berkaitan dan dapat digabung menjadi sebuah sistem. Istilah populer untuk saat ini dinamakan dengan sistem informasi.

1.1. Terminologi dan Definisi

Komputasi awan atau Cloud computing sendiri adalah penggabungan teknologi komputerisasi dan internet, dimana data mulai dari skala kecil hingga besar tersimpan di server internet, yang memungkinkan kita dapat mengakses data kita dari berbagai lokasi dan melalui berbagai platform (Sala-Zárate & Colombo-Mendoza, 2012).

Definisi dari sistem informasi (SI) secara umum adalah suatu sistem yang mengkombinasikan antara aktivitas manusia dan penggunaan teknologi untuk mendukung manajemen dan kegiatan operasional. Dimana, hal tersebut merujuk pada sebuah hubungan yang tercipta berdasarkan interaksi manusia, data, informasi, teknologi, dan algoritma.

Tujuan dari pengembangan sistem informasi adalah untuk menghasilkan sebuah produk yang berisi kumpulan informasi. Sebuah sistem tentunya melibatkan berbagai jenis dan tipe data yang mampu diolah agar dapat ditampilkan dengan mudah kepada pengguna (*user*).

1.1.1. URL

URL adalah singkatan dari Uniform Resource Locators yang berarti suatu “pathname” untuk mengidentifikasi sebuah dokumen di web. URL adalah pengidentifikasi unik yang digunakan untuk menemukan sumber daya di Internet. URL yang valid menunjuk ke sumber daya tertentu, sumber daya tersebut dapat berupa halaman HTML, dokumen CSS, gambar, dan lain sebagainya.

Isi dari URL adalah nama protokol yang diperlukan untuk mengakses sumber daya, serta nama sumber daya. Bagian pertama dari URL mengidentifikasi protokol apa yang digunakan sebagai media akses utama. Bagian kedua mengidentifikasi alamat IP atau nama domain (atau mungkin subdomain) tempat sumber daya berada.

Protokol URL adalah HTTP (Hypertext Transfer Protocol) dan HTTPS (HTTP Secure) untuk sumber daya web, mail to untuk alamat email, FTP untuk file pada server File Transfer Protocol (FTP). Sedangkan telnet untuk sesi saat mengakses komputer jarak jauh.

1.1.2. XAMPP

XAMPP merupakan sebuah perangkat lunak gratis sehingga bebas digunakan. *XAMPP* berfungsi sebagai *server* yang berdiri sendiri (*localhost*), yang terdiri dari *Apache HTTP Server*, *MySQL database* dan penerjemah bahasa yang ditulis dengan bahasa pemrograman PHP dan Perl. *XAMPP* dikembangkan oleh

perusahaan *apache friends* yang biasanya digunakan untuk simulasi pengembangan website.

Definisi sederhana dari Xampp adalah perangkat lunak berbasis web server yang bersifat open source (bebas), serta mendukung di berbagai sistem operasi, baik Windows, Linux, atau Mac OS. Xampp digunakan sebagai standalone server (berdiri sendiri) atau biasa disebut dengan localhost. Hal tersebut memudahkan dalam proses pengeditan, desain, dan pengembangan aplikasi.

Xampp tersusun atas kependekan dari beberapa kata berikut ini:

1. X (Cross Platform)

Maksudnya adalah, Xampp dapat dijalankan di berbagai perangkat sistem operasi yang ada, misalnya Windows, Linux, Mac OS, dan Solaris. Dari ke semua sistem operasi tersebut, software ini bersifat open source atau dapat digunakan secara gratis.

2. A (Apache)

Apache merupakan aplikasi web server yang bertugas untuk menciptakan halaman website yang benar berdasarkan kode program PHP yang ditulis oleh pengembang web (developer). Memungkinkan juga untuk mengakses sistem database terlebih dahulu untuk mendukung halaman situs yang dihasilkan.

3. M (MySQL / MariaDB)

MySQL merupakan salah satu aplikasi database server yang menerapkan bahasa pemrograman SQL (Structured Query Language). Fungsi dari MySQL sendiri adalah untuk mengelola dan membuat sistem basis data secara terstruktur dan sistematis.

4. P (PHP)

PHP adalah bahasa pemrograman khusus berbasis web untuk kebutuhan pada sisi server (back end). Sehingga, PHP sangat memungkinkan untuk membuat suatu halaman website menjadi lebih dinamis dengan menerapkan server-side scripting. PHP juga mendukung manajemen sistem pada Oracle, Postgresql, Microsoft Access, dan lain sebagainya.

5. P (Perl)

Perl merupakan bahasa pemrograman untuk segala kebutuhan (cross platform) yang berfungsi sebagai penunjuk eksistensi dari PHP. Perl biasanya banyak digunakan untuk website development pada sistem berbasis CMS (Content Management System) seperti WordPress.

1.1.3. Apache

Apache adalah aplikasi web server. Tugas utama Apache adalah menampilkan halaman web yang benar kepada user berdasarkan kode HTML, PHP atau yang lainnya. Server HTTP Apache atau Server Web/WWW Apache adalah server web yang dapat dijalankan di banyak sistem operasi yang berguna untuk melayani dan memfungsikan situs web. Protokol yang digunakan untuk melayani fasilitas web/www ini menggunakan HTTP.

1.1.4. MySQL

MySQL adalah aplikasi database server. *SQL* merupakan kependekan dari *Structured Query Language*. *SQL* merupakan bahasa terstruktur yang digunakan untuk mengolah database. *MySQL* dapat digunakan untuk membuat dan mengelola database beserta isinya. *Programmer* atau *user* dapat memanfaatkan *MySQL* untuk menambahkan (*Create*), membaca atau menampilkan (*Read*), mengubah (*Update*), dan menghapus (*Delete*) data yang berada dalam *database*.

MySQL adalah DBMS yang open source dengan dua bentuk lisensi, yaitu Free Software (perangkat lunak bebas) dan Shareware (perangkat lunak berpemilik yang penggunaannya terbatas). Jadi *MySQL* adalah database server yang gratis dengan lisensi GNU General Public License (GPL) sehingga dapat Anda pakai untuk keperluan pribadi atau komersil tanpa harus membayar lisensi yang ada.

MySQL masuk ke dalam jenis RDBMS (Relational Database Management System). Maka dari itu, istilah semacam baris, kolom, tabel, dipakai pada *MySQL*. Contohnya di dalam *MySQL* sebuah database terdapat satu atau beberapa tabel.

SQL sendiri merupakan suatu bahasa yang dipakai di dalam pengambilan data pada relational database atau database yang terstruktur. Jadi *MySQL* adalah database management system yang menggunakan bahasa *SQL* sebagai bahasa penghubung antara perangkat lunak aplikasi dengan database server.

1.1.5. PHP

PHP kependekatan dari *Hypertext Preprocessor* yaitu salah satu bahasa pemrograman web. Bahasa pemrograman *PHP* merupakan bahasa pemrograman untuk membuat web yang bersifat *server-side scripting*. *PHP* memungkinkan untuk membuat halaman web yang bersifat dinamis. Sistem manajemen basis data yang sering digunakan bersama *PHP* adalah *MySQL* namun *PHP* juga mendukung sistem manajemen database *Oracle*, *Microsoft Access*, *Interbase*, *d-base*, *PostgreSQL*, dan sebagainya.

PHP adalah singkatan rekursif untuk "PHP: Hypertext Preprocessor", yaitu bahasa pemrograman yang digunakan secara luas untuk penanganan pembuatan dan pengembangan sebuah situs web dan bisa digunakan bersamaan dengan HTML.

PHP adalah bagian dari bahasa skrip, seperti JavaScript dan Python. Mengutip modul Dasar-Dasar *PHP* dari dinus.ac.id, *PHP* merupakan bahasa script yang dijalankan

pada sisi server (SSS: Server Side Scripting). Database yang didukung PHP antara lain: MySQL, Informix, Oracle, Sybase, Solid, PostgreSQL, Generic ODBC. PHP adalah software Open Source, bebas untuk diunduh dan digunakan.

PHP juga banyak diaplikasikan untuk pembuatan program-program seperti misalnya sistem informasi klinik, rumah sakit, akademik, keuangan, manajemen aset, manajemen bengkel dan lain-lain. Penerapan PHP saat ini juga banyak ditemukan pada proyek-proyek pemerintah seperti e-budgetting, e-procurement, e-government dan e-e lainnya.

Menurut Kustiyaningsih (2011:114), “PHP (atau resminya PHP: Hypertext Preprocessor) adalah skrip bersifat server-side yang ditambahkan ke dalam HTML”. Pada prinsipnya, server akan bekerja apabila ada permintaan dari client. Dalam hal ini client menggunakan kode-kode PHP untuk mengirimkan permintaan ke server.

BAB 2. Pengenalan konsep PHP dan HTML

Dalam membuat halaman website, sebenarnya tidak terlalu membutuhkan bahasa pemrograman PHP. Cukup dengan menggunakan HTML dan CSS saja sudah bisa membuat tampilan web secara statis dan hanya tersusun konten saja. Namun, untuk pengembangan website yang membutuhkan integrasi dengan server serta kebutuhan database, maka PHP sangat dibutuhkan sebagai penghubung antara web statis dengan server dan database.

Website dinamis juga dapat digunakan untuk menyimpan data ke dalam database, selain itu juga dapat membuat halaman situs berubah sesuai dengan hasil input dari user. Jika anda sering menuliskan form pada halaman situs, maka hasil data yang telah dikirim akan diproses melalui PHP. Selain itu, bahasa pemrograman PHP biasanya juga dapat disisipkan ke dalam file atau dokumen HTML.

2.1. Membuat file PHP

Setelah mengetahui apa itu PHP, selanjutnya kita akan membahas terkait dengan sintaks dasar yang digunakan dalam menyusun algoritma dan struktur data dari PHP. Pertama, untuk melakukan proses coding, dengan menuliskan terlebih dahulu `<?php` dan ditutup dengan `?>`, sebagai contoh:

2.1.1. Sintaks Dasar dari PHP

```
<?php  
  
echo "Halo, Saya Arif Wibisono";  
  
?>
```

Jadi, fungsi dari echo adalah sebuah perintah untuk menampilkan teks. Selanjutnya, berkaitan dengan penggunaan variabel pada PHP dan bersifat case sensitive. Maksudnya, penggunaan dari huruf kapital dan kecil akan sangat berpengaruh pada output yang dihasilkan, sebagai contoh:

```
<?php  
  
$layanan = "Pengembangan Sistem Informasi";  
  
echo $layanan;  
  
?>
```

Dari kode program diatas, komponen terpenting untuk mendeklarasikan sebuah variabel adalah dengan menambahkan karakter \$ pada awal variabel. Dan pastikan untuk memanggil variabel tersebut sesuai dengan variabel yang ditulis. Apabila variabel yang anda tulis memiliki perbedaan dalam penulisan huruf kapital, maka akan berdampak pada tampilan error.

Dan sintaks dasar yang harus anda ketahui adalah penggunaan fitur komentar untuk memudahkan dalam proses pengembangan suatu program. Untuk menampilkan komentar pada PHP, anda perlu menuliskan karakter // atau /*. Tujuan dari penggunaan komentar agar kode program yang anda tulis tidak dijalankan atau diproses oleh sistem. Contoh penerapannya adalah sebagai berikut:

```
<?php  
  
// Ini contoh komentar sebaris  
  
echo "Selamat Pagi!";  
  
/*
```

Contoh komentar lebih dari

satu baris

```
*/  
  
?>
```

2.1.2. Struktur Penulisan

Untuk struktur penulisan dari PHP sendiri, dibagi menjadi dua jenis, pertama adalah PHP Native dan yang kedua adalah Framework PHP. Berikut merupakan beberapa penjelasan terkait dengan kedua struktur tersebut.

1. PHP Native

Native adalah bentuk penulisan kode menggunakan PHP yang dimulai dari nol untuk merancang sebuah halaman website. PHP Native biasanya sering digunakan oleh para developer yang mempunyai keahlian khusus dalam hal coding dan menyusun sebuah algoritma dan struktur data yang baik. Sehingga dapat memungkinkan untuk membuat alur kerangka kerja yang optimal dan memiliki fungsionalitas tinggi.

2. Framework PHP

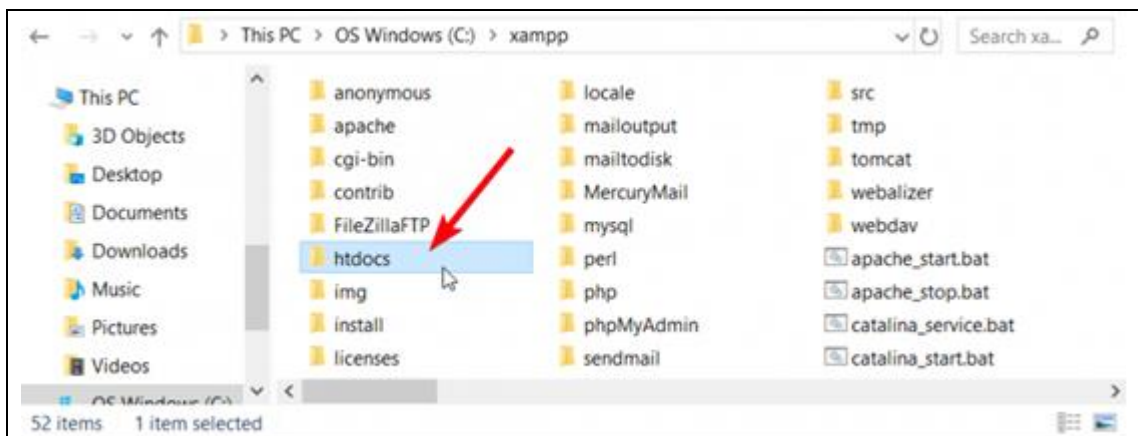
Ketika berbicara masalah framework, maka kita akan berfokus dalam bagaimana merancang sebuah alur atau kerangka kerja yang baik. Sehingga, dapat memungkinkan untuk bekerja secara lebih cepat, akurat, dan optimal. Tentunya, di dalam PHP sendiri juga memiliki framework khusus yang membantu tugas dari pengembang aplikasi berbasis web.

Beberapa framework PHP yang populer dan banyak digunakan saat ini adalah *CodeIgniter*, *Laravel*, *Symfony*, *Yii*, dan *Zend framework*. Apabila anda telah mahir dalam bidang native, alangkah lebih baik jika memulai bekerja dengan menggunakan sebuah framework. Dampak positif dari pengguna kerangka kerja

adalah proses pengembangan software dapat berjalan lebih cepat dan sistem keamanan yang lebih terjamin.

2.2. Membuat file PHP dan HTML

Agar dapat mengakses halaman PHP dari web browser, maka kita harus menempatkan file PHP dalam folder khusus yang merupakan folder home dari web server. Pada aplikasi XAMPP, folder tersebut adalah folder `htdocs` yang berada di `C:\xampp\htdocs`.



Gambar 1. Lokasi `htdocs` dalam Xampp Local Server

1. Cara Menulis File PHP

File PHP pada dasarnya merupakan file text biasa yang bisa dibuat menggunakan aplikasi notepad bawaan windows, namun untuk penggunaan yang lebih baik, kita bisa pakai aplikasi text editor khusus programming seperti Notepad++, Atom, VS Code atau Sublime Text.

Teks editor yang paling ringan adalah Notepad++, aplikasi ini hanya berukuran 4MB dan sudah cukup untuk membuat perintah PHP sederhana.

2. Praktik Menulis File PHP

Kembali ke folder belajar yang berada di `C:\xampp\htdocs\belajar`, kita akan membuat sebuah file baru. Untuk pembuatan file ini bisa menggunakan notepad bawaan Windows, Notepad++, maupun VS Code.

Untuk isi dari file tersebut, silahkan ketik kode berikut:

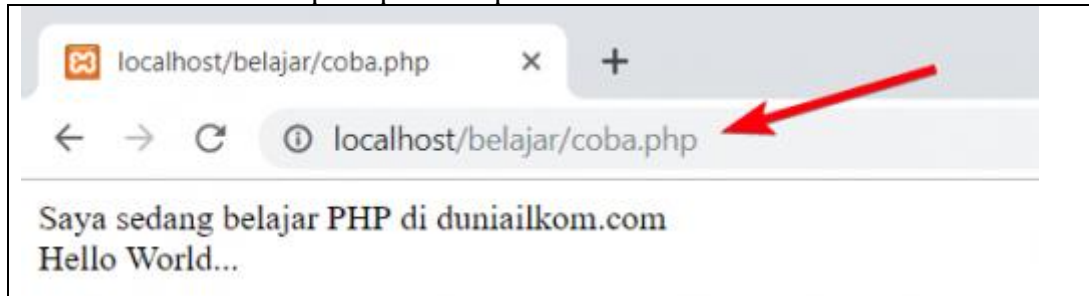
```
1 <?php
2     echo "Saya sedang belajar PHP di duniailkom.com";
3     echo "<br>";
4     echo "Hello World...";
5 ?>
```

Gambar 2. Pemrograman Menggunakan Teks Editor Notepad++

Lalu save sebagai coba.php. File tersebut akan berada pada alamat C:\xampp\htdocs\belajar\coba.php. Untuk menjalankannya, harus menggunakan web browser.

Lalu buka browser, dan ketik alamat berikut: localhost/belajar/coba.php pada address bar dan tekan Enter.

Apabila anda mengikutinya langkah-langkah di atas dengan benar, maka di dalam web browser akan tampil seperti tampilan di berikut:



Gambar 3. Tampilan pada Browser

BAB 3. Fundamental Framework Codeigniter

Framework adalah sebuah kerangka kerja yang digunakan untuk membantu developer dalam mengembangkan kode aplikasi secara konsisten.

Apa itu Codeigniter?

Codeigniter adalah salah satu framework untuk membuat website dengan bahasa pemrograman PHP.

Codeigniter terkenal dengan konsep MVC-nya. MVC merupakan singkatan dari *Model-View-Controller*.

3.1. Instalasi Codeigniter di localhost

Codeigniter merupakan sebuah framework php yang sangat sederhana, ringan, dan mudah untuk dipelajari, codeigniter sendiri telah mengadopsi pola MVC yaitu *Model-View-Controller* dimana kita dapat memisahkan logika dan tampilan sebuah website yang ingin dibangun dengan framework berbasis MVC ini.

Namun yang perlu diperhatikan dalam tahap dasar ini sebelum kita memulai seharusnya kita telah mengerti konsep *OOP (Object Oriented Programming)* agar lebih mudah memahami cara kerja sebuah framework.

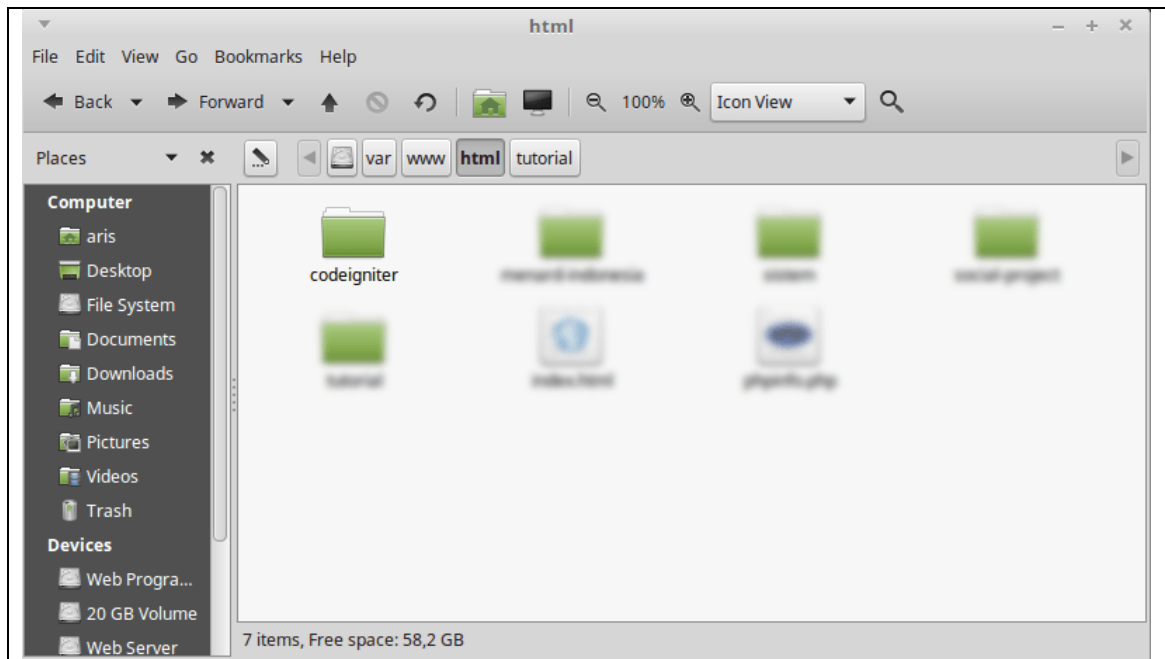
Seperti awal pembahasan bahwa codeigniter ini adalah sebuah framework yang sangat sederhana, sehingga cara install codeigniter sendiri tidak terlalu sulit bahkan sangat mudah, dimana kita hanya memerlukan sebuah web server pada komputer kita, berikut ini caranya :

Langkah 1. Download file Codeigniter

Silahkan klik link <https://codeigniter.com/> lalu download framework codeigniter terbaru, saat ini yang terbaru adalah versi 3.1.3, dan akan terus berlanjut ke versi berikutnya.

Langkah 2. Ekstrak dan Install Codeigniter Framework

Setelah anda download maka anda akan mendapatkan sebuah file bernama *CodeIgniter-3.1.3.zip*, silahkan ekstrak file tersebut setelah itu akan muncul folder bernama *CodeIgniter-3.1.3*, silahkan anda rename menjadi *codeigniter* lalu salin folder tersebut ke folder root anda, yaitu di *htdocs*, sedangkan linux di folder *www/html*.



Gambar 4. Peletakan File Folder Code Igniter

Langkah 3. Konfigurasi Base URL Codeigniter

Setelah anda selesai melakukan instalasi codeigniter maka selanjutnya melakukan konfigurasi base url yang terdapat di folder application/config/config.php silahkan anda buka dengan teks editor yang anda sukai. Lalu pada bagian kode dibawah ini:

```
$config['base_url'] = '';
```

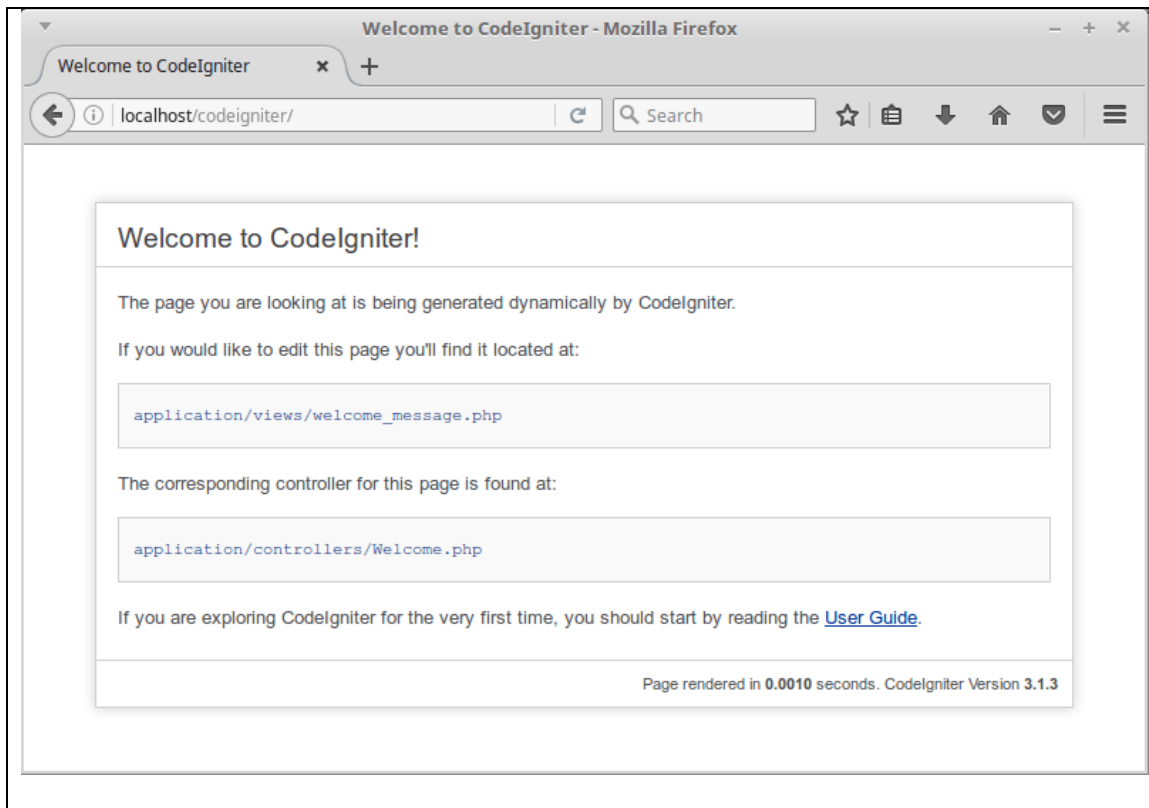
Ubahlah menjadi dibawah ini:

```
$config['base_url'] = 'http://localhost/codeigniter';
```

Nilai tersebut harus sesuai dengan alamat/nama folder yang anda tentukan ketika anda menyalin file codeigniter ke dalam folder root web server anda.

Langkah 4. Selesai

Setelah semua sudah dilakukan sesuai dengan instruksi diatas silahkan buka browser dan ketikkan url <http://localhost/codeigniter>, jika berhasil akan keluar tampilan seperti gambar dibawah ini:



Gambar 5. Tampilan Code Igniter yang telah berhasil terinstall di Xampp Local Server

3.2. Mengenal Konsep MVC

MVC atau Model View Controller merupakan pola perancangan perangkat lunak yang menitik beratkan terhadap pembagian kode berdasarkan fungsinya. Masing-masing kode sumber memiliki fungsinya masing-masing yang akan dikelompokkan terhadap tiga bagian itu.

Sejarah MVC

Konsep MVC diciptakan oleh programmer Smalltalk yang bernama Trygve Reenskaug. Beliau memperkenalkan konsep MVC ini pada paper yang dirilis tanggal 10 November 1979. Kemudian Jim Althoff dan yang lainnya mengimplementasikan sebuah versi dari MVC untuk Smalltalk-80 class library.

MVC Hari Ini

Meskipun pada dasarnya MVC dibangun untuk aplikasi Dekstop, tapi sesungguhnya MVC ini lebih dikembangkan oleh programmer web. Dimana aplikasi web ini terdiri dari berbagai bahasa pemrograman yang berbeda namun mengusung konsep yang sama. Beberapa kode sumber ada yang bersifat non-komersial ada juga yang dikomersilkan, namun tetap mengacup terhadap ciri khasnya masing-masing.

MVC Di Backend / Server

Beberapa bahasa pemrograman backend (server) masih mengusung konsep MVC diantaranya adalah Ruby on Rails, Django, ASP.NET MVC. Dimana semua framework

itu akan berlaku sebagai model dan controllernya. Sementara ini, framework seperti AngularJS, EmberJS, JavaScriptMVC and Backbone digunakan sebagai view.

Implementasi MVC dalam Pemrograman WEB

Beberapa implementasi MVC dalam pemrograman web ada didalam Framework PHP seperti Code Igniter, cakePHP, Laravel, dan SEME FRAMEWORK.

Fungsi Model dalam Konsep MVC

Model berfungsi untuk mengumpulkan semua logika yang berhubungan dengan database. Semua Query akan terkumpul yang dipisahkan berdasarkan masing-masing methodnya.

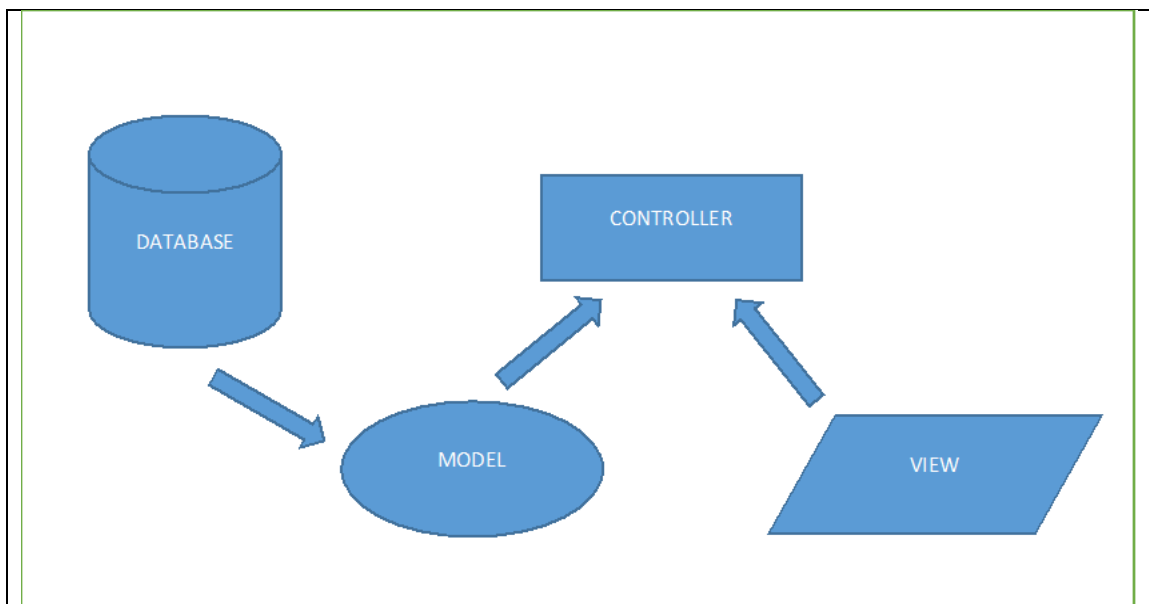
Fungsi View

View berfungsi menampung file-file HTML, javascript, meta header, dan lainya yang akan dipisah berdasarkan fungsi layoutnya. View ini nanti akan terdiri dari beberapa file yang akan dirangkai didalam controller. View ini juga yang nantinya akan mengendalikan kemampuan SEO dari sebuah WEB.

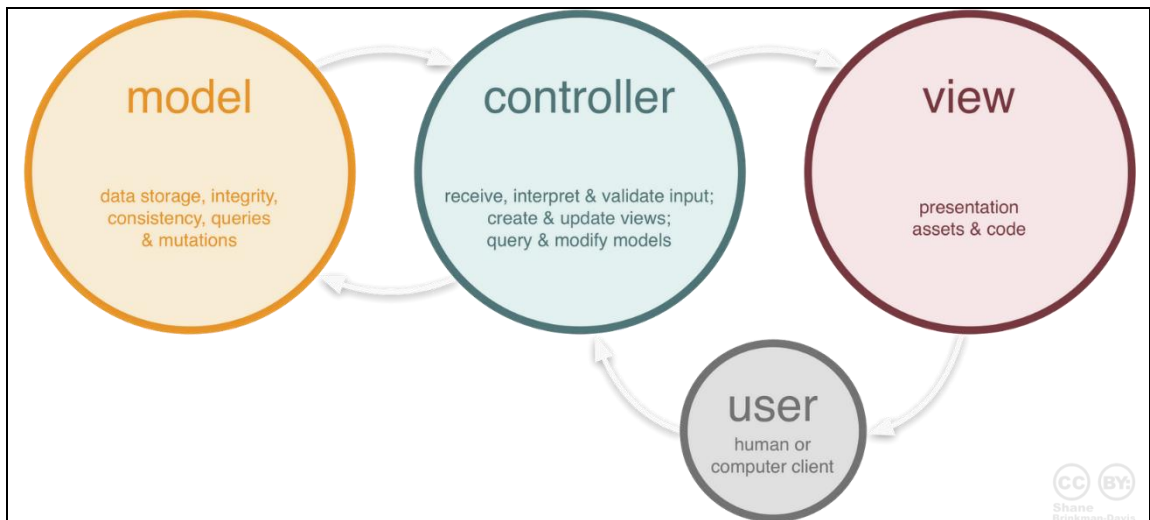
Fungsi Controller dalam Konsep MVC

Dalam konsep MVC, controller ini banyak sekali fungsinya, mulai dari pusat pengendali logika yang akan menggabungkan model dan view untuk diolah secara bersama-sama. Selain itu, controller juga berfungsi sebagai default routing untuk path request dalam beberapa framework. Sementara, untuk diframework laravel, fungsi controller sebagai routing engine sudah dipisah menjadi masing-masing lagi. Dan juga middleware atau API provider, dalam hal ini secara default dikelola oleh controller, tapi disebagian framework lagi hal ini dipisahkan.

Untuk lebih jelasnya bisa dilihat di ilustrasi ini.



Gambar 6. Ilustrasi MVC (Model, View & Controller)



Gambar 7. Fungsi tiap Blok MVC (Model, View & Controller)

BAB 4. Membuat Controller, View dan Model

4.1. Membuat Model di Codeigniter

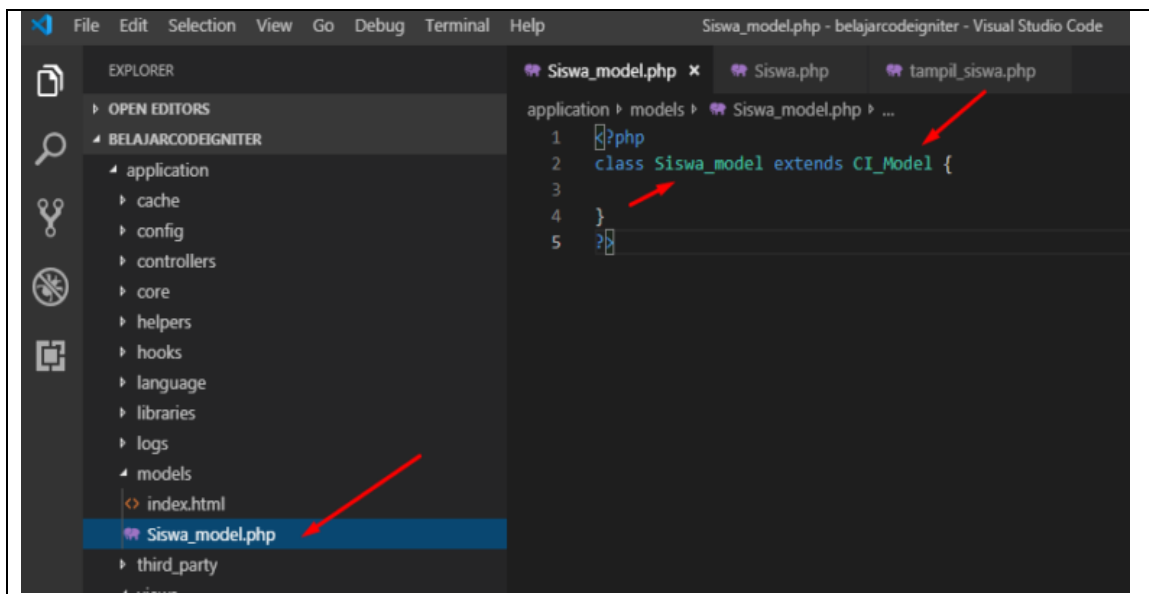
Seperti yang telah di singgung di pengenalan konsep MVC, Model ini adalah bagian yang digunakan untuk berkomunikasi dengan database.

Beberapa aturan dalam pembuatan Model antara lain :

File Model disimpan didalam folder application/models

Nama File, serta Class Model diawali oleh huruf besar / capital

Contohnya kita buat file dengan nama Siswa_model.php silahkan simpan di folder application/models.



Gambar 8. Pemrograman Model Menggunakan Visual Studio Code Text Editor

Codenya seperti berikut ini :

```
1 <?php
2 class Siswa_model extends CI_Model {
3
4 }
5 ?>
```

Keterangan :

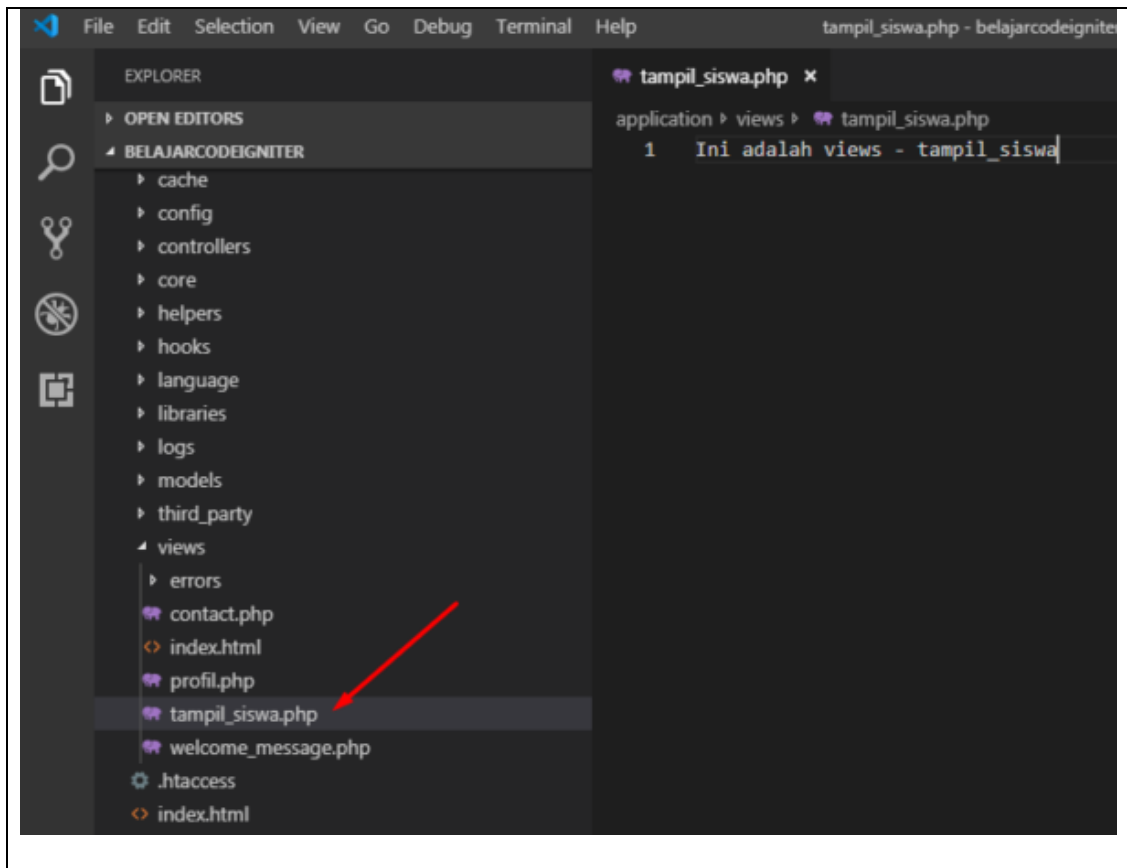
- File model disimpan di direktori : application/models
- Baik nama file model, dan class model harus diawali huruf besar
- Class model harus di extends dengan CI_Model

4.1.1. Membuat View di Codeigniter

Setelah anda belajar membuat controller, berikutnya kita akan belajar membuat sebuah view di codeigniter.

Anda bisa membuat view, didalam folder `application/views`

Pada contoh ini kita buat file dengan nama `tampil_siswa.php`, silahkan simpan di folder `application/views`



Untuk dapat memanggil view, kita perlu memanggilnya dibagian controller, contoh dibagian controller Siswa, kita akan load view `tampil_siswa.php` didalam function `index`, maka perintahnya adalah sebagai berikut :

```
$this->load->view('tampil_siswa');
```

Perintah tersebut artinya adalah mengakses view, dengan nama `tampil_siswa.php`, anda tidak perlu menuliskan `.php` didalam pemanggilan tersebut.

Sehingga jika kita letakkan pada function `index` di controller Siswa, perintahnya adalah sebagai berikut ini :

```
1 public function index(){
2     $this->load->view('tampil_siswa');
3 }
```

Dan ketika kita mengakses alamat : localhost/belajarcodigniter/index.php/siswa



Gambar 9. tampilan file view dengan nama tampil_siswa

4.1.2. Membuat Controller di Codeigniter

Controller merupakan bagian yang mengatur hubungan antara bagian model dan bagian view, sederhananya controller berfungsi untuk menerima request dan data dari user, berikutnya bagian controller akan menentukan proses apa yang akan dilakukan oleh aplikasi.

Baik kita akan membuat controllernya, anda bisa membuat controller di dalam folder : application/controllers

Contohnya kita akan buat controller dengan nama Siswa, silahkan buat file dengan nama Siswa.php didalam folder application/controllers

Dengan isi code seperti dibawah ini :

```
6 <?php
7 defined('BASEPATH') OR exit('No direct script access allowed');
8
9 class Siswa extends CI_Controller {
10
11     function __construct(){
12         parent::__construct();
13     }
14
15     public function index(){
16         echo "ini adalah method index pada controller Siswa";
17     }
18
19     public function profil(){
20         echo "ini adalah method profil pada controller Siswa";
21     }
22 }
```

Keterangan :

Untuk membuat sebuah controller, anda harus membuatnya didalam folder application/controllers, untuk nama filenya karakter awalnya gunakan huruf besar (jika tidak menggunakan huruf besar, pengalaman saya akan bermasalah jika dijalankan di server OS linux).

Nama file controller tidak mengandung karakter special seperti spasi, @, serta katakter special lainnya, kecuali underscore (_)

File Controller ini merupakan file php

```
defined('BASEPATH') OR exit('No direct script access allowed');
```

syntax ini memiliki fungsi untuk mencegah akses langsung ke file controller.

```
class Siswa extends CI_Controller {
```

Syntax ini digunakan untuk memberikan nama terhadap class, nama class harus diawali dengan huruf besar, nama class juga harus sesuai dengan nama file controller, karena nama file controller adalah Siswa, sehingga untuk nama class adalah Siswa, Class controller harus extends class CI_Controller.

```
1function __construct(){  
2    parent::__construct();  
3}
```

Pada method __construct digunakan untuk menjalankan fungsi yang anda inginkan pada saat controller diakses, pada bagian method __construct diletakkan fungsi untuk memanggil helper ataupun library, serta mendeklarasikan variable yang akan sering digunakan dibagian method pada controller itu sendiri

```
2public function index(){  
3    echo "ini adalah method index pada controller Siswa";  
4}  
5  
6public function profil(){  
7    echo "ini adalah method profil pada controller Siswa";  
8}
```

Dalam contoh ini kita membuat 2 method didalam class Siswa, yaitu index (merupakan method yang dijalankan ketika controller tersebut diakses di bagian URL), dan method profil

Setelah anda membuat controller jangan lupa disave ya, lalu kita akan coba mengakses controller tersebut.

Untuk mengakses controller tersebut silahkan akses alamat : localhost/belajarcodigniter/index.php/siswa



Untuk mengakses method profil di controller Siswa, anda bisa mengakses alamat : localhost/belajarcodigniter/index.php/siswa/profil



4.2. Membuat dan Menggunakan Helper

Codeigniter memiliki beberapa helper yang cukup berguna. Namun terkadang kita membutuhkan fungsi – fungsi lain yang tidak terdapat dalam native helper (helper asli) codeigniter. Misalnya untuk memformat tampilan tanggal dalam Bahasa Indonesia , seperti di bawah ini;

1 selasa, 15 september 2010

Tidak ada helper yang dapat kita pakai untuk memformat tanggal seperti di atas pada kondisi inilah, kita bisa membuat helper sendiri untuk memenuhi kebutuhan tersebut.

Untuk membuat helper sendiri langkah – langkah yang kita harus lakukan adalah :

Langkah 1.

Buat sebuah file bernama tanggal_helper.php, kemudian simpan di folder application/helpers. Folder application/helpers adalah lokasi untuk menyimpan costum helper (helper buatan sendiri). Sedangkan lokasi penyimpanan native helper di system /helpers. dengan demikian, antara native helper (helper asli) dan custom helper tidak bercampur. Jangan khawatir, meskipun lokasinya berbeda, tapi cara pemanggilannya sama. Codeigniter cukup pintar untuk mengetahui di mana lokasi file tersebut berada. Pada dasarnya codeigniter akan mencari file helper di system/helpers atau di application/helpers.

Berikut isi file tanggal_helper.php yang kita buat tadi:

```
<?php
function formatHariTanggal($waktu)
{
    $hari_array = [
        'Minggu',
        'Senin',
        'Selasa',
        'Rabu',
        'Kamis',
        'Jumat',
        'Sabtu'
    ];
```

```

        $hr = date('w', strtotime($waktu));
        $hari = $har_array[$hr];

        $tanggal = date('j', strtotime($waktu));

        $bulan_array = [
            1 => 'Januari',
            2 => 'February',
            3 => 'Maret',
            4 => 'April',
            5 => 'Mei',
            6 => 'Juni',
            7 => 'Juli',
            8 => 'Agustus',
            9 => 'September',
            10 => 'Oktober',
            11 => 'November',
            12 => 'Desember',
        ];

        $bl = date('n', strtotime($waktu));
        $bulan = $bulan_array[$bl];
        $tahun = date('Y', strtotime($waktu));

        return "$hari, $tanggal, $bulan, $tahun";
    }

```

Yang perlu diingat, Pemberian file helper harus diberikan akhiran `_helper` sebelum ekstensi `.php`, seperti `tanggal_helper.php` di atas. selanjutnya helper tersebut akan di panggil dengan perintah:

```
$this->load->helper('tanggal');
```

Bagian `_helper` tidak diikutkan pada pemanggilan helper.

Langkah 2.

Buat sebuah file controller bernama `Demo.php`, simpan di `application/controllers`. Isi dari controller `Demo.php` tersebut:

```

<?php

class Demo extends CI_Controller
{
    public function index()
    {
        $this->load->helper('tanggal');
    }
}

```

```
$waktu = date('Y-m-d');
echo formatHariTanggal($waktu);
}
}
```

4.3. Membuat dan Menggunakan Library

Pada codeigniter kita diperbolehkan untuk membuat library sendiri untuk keperluan kita dalam pengembangan aplikasi dengan menggunakan codeigniter, jadi ketika anda memerlukan pembuatan library yang anda gunakan berkali – kali, ada baiknya anda membuat library sendiri agar bisa digunakan di codeigniter.

Untuk membuat library di codeigniter, anda dapat menyimpan file library didalam folder yang telah disediakan oleh codeigniter, yaitu didalam folder application/libraries/ dalam contoh ini kita akan membuat library dengan nama ‘persegi’

silahkan buat file dengan nama persegi.php dan simpan didalam folder application/libraries

codenya adalah seperti berikut ini :

```
<?php
3
4 class Persegi{
5
6     function keliling($sisi){
7         $keliling = $sisi * 4;
8         echo "Keliling Persegi dengan sisi ".$sisi." adalah ".$keliling;
9     }
10
11     function luas($sisi){
12         $luas = $sisi * $sisi;
13         echo "Luas Persegi dengan sisi ".$sisi." adalah ".$luas;
14     }
15 }
```

Keterangan :

Karena library yang kita buat adalah dengan nama persegi, sehingga kita perlu perlu mendeklarasikan untuk nama classnya adalah Persegi perhatikan pada line 3

```
1class Persegi{
2
3}
```

Berikutnya kita buat 2 method yaitu :

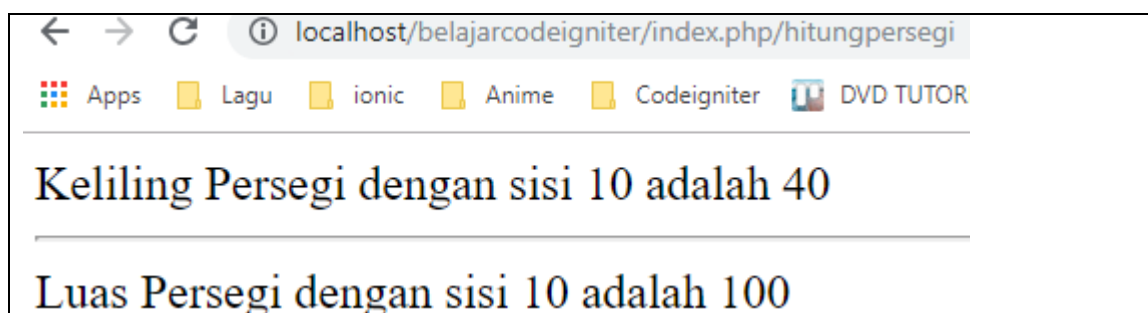
1. method keliling digunakan untuk menghitung keliling persegi, dengan parameternya adalah variabel sisi
2. method luas digunakan untuk menghitung luar persegi, dengan parameternya adalah variabel sisi

Sampai disini kita telah membuat library persegi dan siap untuk digunakan, dalam contoh ini kita mencontohkan library yang sederhana, anda bisa mengembangkan library sesuai kebutuhan aplikasi yang anda buat.

langkah selanjutnya adalah kita sudah bisa menggunakan library ini didalam controller, dalam contoh ini kita akan buat sebuah controller dengan nama `Hitungpersegi.php`

```
<?php
3 defined('BASEPATH') OR exit('No direct script access allowed');
4
5
6 class Hitungpersegi extends CI_Controller {
7
8     function __construct(){
9         parent::__construct();
10        $this->load->library('persegi');
11    }
12
13
14        function index(){
15            $this->persegi->keliling('10');
16            echo "<hr/>";
17            $this->persegi->luas('10');
18        }
19    }
```

Berikutnya kita akan coba mengaksesnya dengan alamat : <http://localhost/belajarcodigniter/index.php/hitungpersegi>



Keterangan :

Perhatikan line 8 kita load library persegi, kita load didalam function __construct agar bisa digunakan diseluruh controller Hitungpersegi

```
$this->load->library('persegi');
```

Pada line 12 kita mengakses function keliling di library persegi, dengan menyertakan nilai 10 dibagian parameter.

```
$this->persegi->keliling('10');
```

Sehingga saat diakses tertulis : Keliling Persegi dengan sisi 10 adalah 40

Pada line 14 kita mengakses function luas di library persegi, dengan menyertakan nilai 10 dibagian parameter.

```
$this->persegi->luas('10');
```

Sehingga saat diakses tertulis : Luas Persegi dengan sisi 10 adalah 100

BAB 5. Upload File Menggunakan Codeigniter

Karena nantinya kita akan membuat fitur upload dan download, dimana nama berkasnya akan kita simpan didatabase, sehingga kita perlu membuat databasenya.. dalam contoh ini kita buat databasenya dengan nama upload_db.

dalam database itu kita buat sebuah tabel dengan nama tb_berkas, dimana kolomnya adalah sebagai berikut :

1. kd_berkas (int) – Primary Key – Auto Increment
2. nama_berkas (varchar 255)
3. keterangan_berkas (varchar 255)
4. tipe_berkas (varchar 100)
5. ukuran_berkas (float)

Untuk memudahkan teman – teman, bisa menggunakan code SQL dibawah ini :

```
-- phpMyAdmin SQL Dump
-- version 4.9.0.1
-- https://www.phpmyadmin.net/
--
-- Host: 127.0.0.1
-- Waktu pembuatan: 01 Feb 2020 pada 06.34
-- Versi server: 10.4.6-MariaDB
-- Versi PHP: 7.3.9

SET SQL_MODE = "NO_AUTO_VALUE_ON_ZERO";
SET AUTOCOMMIT = 0;
START TRANSACTION;
SET time_zone = "+00:00";

/*!40101 SET @OLD_CHARACTER_SET_CLIENT=@@CHARACTER_SET_CLIENT
*/;
/*!40101 SET
@OLD_CHARACTER_SET_RESULTS=@@CHARACTER_SET_RESULTS */;
/*!40101 SET @OLD_COLLATION_CONNECTION=@@COLLATION_CONNECTION
*/;
/*!40101 SET NAMES utf8mb4 */;

--
-- Database: `upload_db`
--
CREATE DATABASE IF NOT EXISTS `upload_db` DEFAULT CHARACTER SET
latin1 COLLATE latin1_swedish_ci;
USE `upload_db`;
```

```

--
-- Struktur dari tabel `tb_berkas`
--
CREATE TABLE `tb_berkas` (
  `kd_berkas` int(11) NOT NULL,
  `nama_berkas` varchar(255) DEFAULT NULL,
  `keterangan_berkas` varchar(255) DEFAULT NULL,
  `tipe_berkas` varchar(100) DEFAULT NULL,
  `ukuran_berkas` float NOT NULL
) ENGINE=InnoDB DEFAULT CHARSET=latin1;

--
-- Indexes for dumped tables
--

-- Indeks untuk tabel `tb_berkas`
--
ALTER TABLE `tb_berkas`
  ADD PRIMARY KEY (`kd_berkas`);

--
-- AUTO_INCREMENT untuk tabel yang dibuang
--

--
-- AUTO_INCREMENT untuk tabel `tb_berkas`
--
ALTER TABLE `tb_berkas`
  MODIFY `kd_berkas` int(11) NOT NULL AUTO_INCREMENT,
  AUTO_INCREMENT=5;
COMMIT;

/*!40101 SET CHARACTER_SET_CLIENT=@OLD_CHARACTER_SET_CLIENT */;
/*!40101 SET CHARACTER_SET_RESULTS=@OLD_CHARACTER_SET_RESULTS
*/;
/*!40101 SET COLLATION_CONNECTION=@OLD_COLLATION_CONNECTION */;

```

5.1. Setting Database di Codeigniter

Silahkan buka file application/config/database.php, untuk bagian \$db['default'] rubah menjadi seperti berikut :

```

$db['default'] = array(
  'dsn' => "",
  'hostname' => 'localhost',
  'username' => 'root',
  'password' => "",

```

```

'database' => 'upload_db',
'dbdriver' => 'mysqli',
'dbprefix' => "",
'pconnect' => FALSE,
'db_debug' => (ENVIRONMENT !== 'production'),
'cache_on' => FALSE,
'cachedir' => "",
'char_set' => 'utf8',
'dbcollat' => 'utf8_general_ci',
'swap_pre' => "",
'encrypt' => FALSE,
'compress' => FALSE,
'stricton' => FALSE,
'failover' => array(),
'save_queries' => TRUE
);

```

Keterangan :

- Bagian host menjadi localhost
- Bagian username kita set menjadi root (setting default dari xampp)
- Bagian password kita set kosong (setting default dari xampp)
- Bagian database kita set dengan nama databasenya yaitu upload_db
- Bagian dbdriver kita gunakan mysqli

Load Library yang dibutuhkan

Kita akan tuliskan beberapa perintah untuk load library dan helper, silahkan bukan file application/config/autoload.php

tambahkan bagian libraries dengan library database

```
$autoload['libraries'] = array('database');
```

tambahkan bagian helper dengan helper url dan download

```
$autoload['helper'] = array('url', 'download');
```

Setting bagian base_url

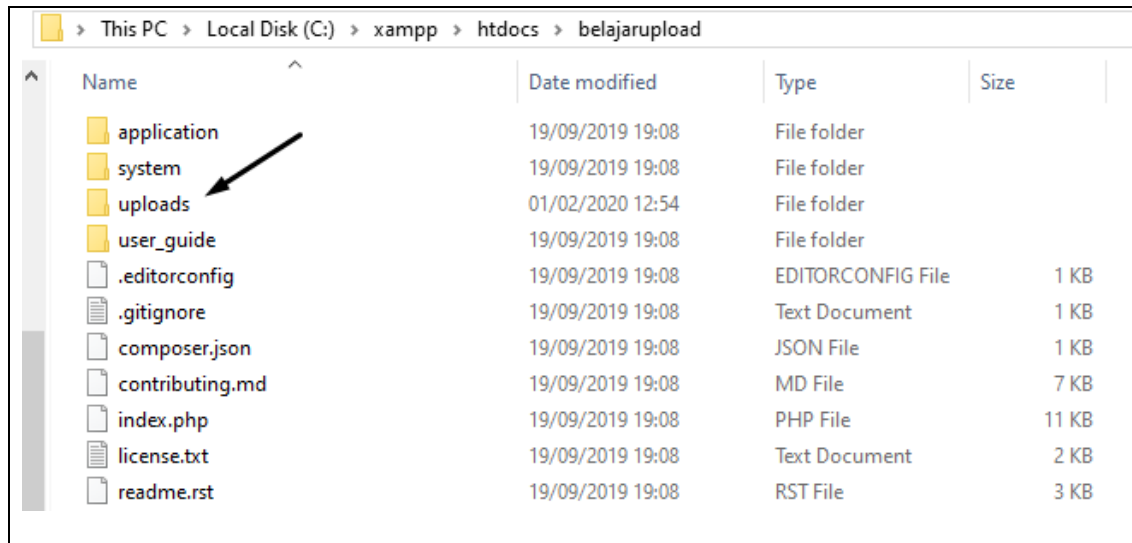
Berikutnya kita harus set bagian base url, buka file application/config/config.php

kita set bagian base_url, dalam contoh ini karena project kita berada pada folder belajarupload, sehingga pada bagian base_url kita tuliskan seperti dibawah ini, nanti bisa teman – teman sesuaikan dengan project punya teman – teman ya.

```
$config['base_url'] = 'http://localhost/belajarupload';
```


5.1.1. Buat Folder dengan nama uploads di project codeigniter

Kita perlu membuat folder yang digunakan untuk menyimpan berkas yang diupload, dalam contoh ini kita buat folder dengan nama uploads didalam folder project codeigniternya, untuk nama folder ini bisa disesuaikan dengan yang teman – teman inginkan, jadi tidak harus uploads.



5.1.2. Membuat Controller Upload

Berikutnya kita buat controller dengan nama Upload.php, posisi dari controller berada pada folder application/controllers.

Dalam Controller Upload, kita buat class Upload, dan kita buat function dengan nama create.

pada function create ini kita akan load view, dengan nama form_upload.php yang akan berisi form untuk mengupload berkas.

sehingga kurang lebih seperti berikut ini isi dari file Controller Upload.php

```
<?php
defined('BASEPATH') OR exit('No direct script access allowed');

class Upload extends CI_Controller {

    function create()
    {
        $this->load->view('form_upload');
    }
}
```

Keterangan :

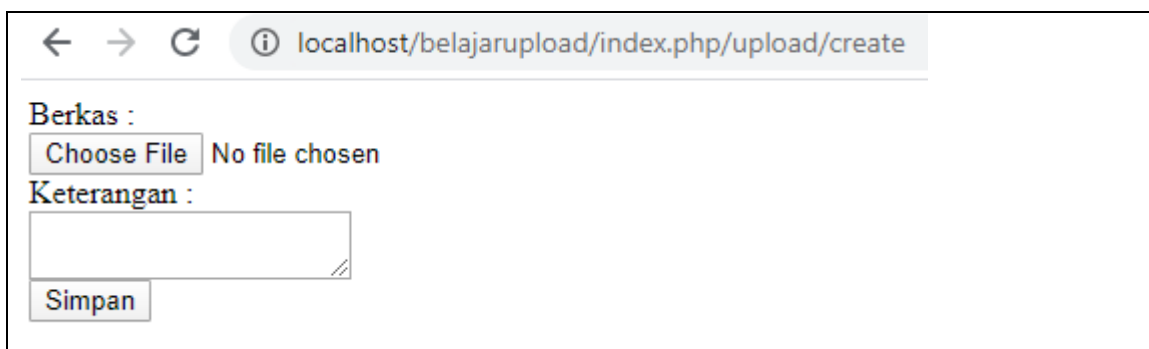
dengan perintah diatas, maka ketika function create diakses maka akan menampilkan file view dengan nama form_upload.

5.2. Buat file view form_upload.php

Berikutnya kita buat file view dengan nama form_upload.php, untuk kodenya adalah sebagai berikut :

```
<html>
  <head>
    <title> Belajar Upload di Codeigniter - Warung Belajar -</title>
  </head>
  <body>
    <?php
      if(isset($error))
      {
        echo "ERROR UPLOAD : <br/>";
        print_r($error);
        echo "<hr/>";
      }
    ?>
    <form method="post" enctype="multipart/form-data" action="<?php echo
base_url(); ?>index.php/upload/proses">
      <div>Berkas : </div>
      <div><input type="file" name="berkas"></div>
      <div>Keterangan : </div>
      <div><textarea name="keterangan_berkas"></textarea></div>
      <div><input type="submit" value="Simpan"/></div>
    </form>
  </body>
</html>
```

Untuk Mengakses form upload, anda bisa menggunakan alamat : localhost/belajarupload/index.php/upload/create



Keterangan :

Pada Line 7 kita menggunakan function isset untuk mengecek apakah memiliki variabel \$error, nah variabel \$error ini nantinya akan kita dapat saat proses upload terjadi error, pada bagian validasi form upload, nanti kita akan jelaskan lebih lanjut saat pembahasan proses uploadnya.

Pada Line 10 kita menampilkan variabel \$error yang berisi array dengan menggunakan perintah print_r, isi variabel \$error ini nantinya akan berisi pesan error saat upload file tidak berhasil.

Pada Line 14 kita gunakan tag <form> html dengan beberapa attribute didalamnya :

- kita gunakan method post.
- Karena kita membuat form upload, sehingga kita perlu menuliskan enctype="multipart/form-data"
- untuk bagian action kita menuliskan value "<?php echo base_url(); ?>index.php/upload/proses" karena nantinya proses upload akan di proses pada function proses di dalam controller Upload.

Pada bagian form kita membuat 2 inputan yaitu :

- inputan dengan tipe file dengan nama berkas
- textarea dengan nama keterangan_berkas

5.2.1. Membuat Function Proses di Controller Upload

Berikutnya kita buat function proses yang digunakan untuk memproses inputan dari bagian form upload, kurang lebih codenya adalah seperti ini :

```
<?php
defined('BASEPATH') OR exit('No direct script access allowed');

class Upload extends CI_Controller {

    function create()
    {
        $this->load->view('form_upload');
    }

    function proses()
    {
        $config['upload_path']     = './uploads/';
        $config['allowed_types']   = 'gif|jpg|png';
        $config['max_size']        = 100;
        $config['max_width']       = 1024;
        $config['max_height']      = 768;
        $config['encrypt_name']    = TRUE;
        $this->load->library('upload', $config);
        if ( ! $this->upload->do_upload('berkas'))
        {
            $error = array('error' => $this->upload-
>display_errors());
            $this->load->view('form_upload', $error);
        }
        else
        {
            $data['nama_berkas'] = $this->upload->data("file_name");
```

```

        $data['keterangan_berkas'] = $this->input-
>post('keterangan_berkas');
        $data['tipe_berkas'] = $this->upload->data('file_ext');
        $data['ukuran_berkas'] = $this->upload->data('file_size');
        $this->db->insert('tb_berkas',$data);
        redirect('upload');
    }
}
}

```

Keterangan :

Pada Line 11 kita buat function proses, untuk melakukan memproses bagian form upload.

Untuk proses upload, kita perlu membuat konfigurasi dari bagian upload, dalam contoh ini kita gunakan variabel \$config, beberapa settingannya adalah sebagai berikut :

```
$config['upload_path'] = './uploads/';
```

artinya untuk folder tempat menyimpan file hasil upload adalah folder uploads, folder uploads yang sebelumnya telah kita buat

```
$config['allowed_types'] = 'gif|jpg|png';
```

artinya untuk file extension yang di perbolehkan untuk diupload adalah gif, jpg,dan png

```
$config['max_size'] = 100;
```

artinya maksimal ukuran file yang boleh diupload adalah 100 Kb

```
$config['max_width'] = 1024;
```

artinya untuk ukuran lebar dari file adalah 1024 px, berlaku jika kita upload file gambar

```
$config['max_height'] = 768;
```

artinya untuk ukuran tinggi dari file adalah 768 px, berlaku jika kita upload file gambar

```
$config['encrypt_name'] = TRUE;
```

digunakan untuk memberikan nama file yang diupload, menjadi acak, sehingga membuat nama file 1 dan yang lain tidak mungkin akan sama.

```
$this->load->library('upload', $config);
```

Berikutnya kita setting konfigurasi ini kedalam library upload (Line 19).

```
if ( ! $this->upload->do_upload('berkas'))
```

Line 20 kita lakukan pengecekan apakah proses upload gagal ? untuk tulisan berkas disini adalah value name , dari bagian form yang digunakan untuk upload file, jika ternyata proses upload gagal maka akan menjalankan perintah di line 22, 23

```
$error = array('error' => $this->upload->display_errors());
```

Line 22 kita mengisi variabel \$error dengan informasi error yang terjadi saat proses upload.

```
$this->load->view('form_upload', $error);
```

Line 23 kita load view dengan nama form_upload, dengan mempassing variabel \$error, nah variabel ini yang sudah kita bahas pada file view dengan nama form_upload pada line 10

```
$data['nama_berkas'] = $this->upload->data("file_name");
```

Line 27 kita buat array \$data dengan elemen “nama_berkas” untuk nama berkasnya kita gunakan perintah \$this->upload->data(“file_name”) untuk bagian file name ini, akan bernilai hash, karena kita menggunakan perintah \$config[‘encrypt_name’] = TRUE;

```
$data['keterangan_berkas'] = $this->input->post('keterangan_berkas');
```

Line 28 kita buat array \$data dengan elemen “keterangan_berkas” untuk menyimpan nilai dari bagian form keterangan_berkas

```
$data['tipe_berkas'] = $this->upload->data('file_ext');
```

Line 29 kita buat array \$data dengan elemen “tipe_berkas” untuk menyimpan file extension dari file yang diupload dengan perintah \$this->upload->data(‘file_ext’)

```
$data['ukuran_berkas'] = $this->upload->data('file_size');
```

Line 30 kita buat array \$data dengan elemen “ukuran_berkas” untuk menyimpan ukuran berkas dari file yang diupload, dengan perintah \$this->upload->data(‘file_size’)

```
$this->db->insert('tb_berkas', $data);
```

Line 31 kita tuliskan perintah \$this->db->insert(‘tb_berkas’, \$data) yang artinya kita insert data ke dalam tabel tb_berkas dengan data pada array \$data.

```
redirect('upload');
```

Pada Line 32 kita redirect kedalam controller upload, ketika proses insert data ke tabel tb_berkas selesai.

5.2.2. Menampilkan Data File yang diupload

Berikutnya masih didalam controller Upload, kita menambahkan function index, yang nantinya digunakan untuk menampilkan view dengan nama tampil_berkas, dimana dalam view ini kita akan menampilkan data dari tabel tb_berkas.

```
<?php
defined('BASEPATH') OR exit('No direct script access allowed');

class Upload extends CI_Controller {
```

```

function create()
{
    $this->load->view('form_upload');
}

function proses()
{
    $config['upload_path']     = './uploads/';
    $config['allowed_types']   = 'gif|jpg|png';
    $config['max_size']        = 100;
    $config['max_width']       = 1024;
    $config['max_height']      = 768;
    $config['encrypt_name']    = TRUE;
    $this->load->library('upload', $config);
    if ( !$this->upload->do_upload('berkas'))
    {
        $error = array('error' => $this->upload-
>display_errors());
        $this->load->view('form_upload', $error);
    }
    else
    {
        $data['nama_berkas'] = $this->upload->data("file_name");
        $data['keterangan_berkas'] = $this->input-
>post('keterangan_berkas');
        $data['tipe_berkas'] = $this->upload->data('file_ext');
        $data['ukuran_berkas'] = $this->upload->data('file_size');
        $this->db->insert('tb_berkas',$data);
        redirect('upload');
    }
}

public function index()
{
    $data['berkas'] = $this->db->get('tb_berkas');
    $this->load->view('tampil_berkas',$data);
}
}

```

Jadi kurang lebih untuk controller Upload seperti gambar diatas, kita menambahkan function index dari line 36 – 40.

```

public function index()
{
    $data['berkas'] = $this->db->get('tb_berkas');
    $this->load->view('tampil_berkas',$data);
}

```

Keterangan :

Line 3 kita buat array variabel \$data, dengan elemen berkas dimana kita akan mengisi data record dari tabel tb_berkas.



Line 4 kita load view dengan nama tampil_berkas dengan passing data variabel \$data, yang berisi data dari tabel tb_berkas.

5.2.3. Buat View dengan tampil_berkas

Berikutnya kita buat view dengan nama tampil_berkas, untuk codenya adalah seperti berikut :

```
<html>
  <head>
    <title></title>
  </head>
  <body>
    <h3><center>Tampil Berkas</center></h3>
    <hr/>
    <a href="<?php echo base_url(); ?>index.php/upload/create">Tambah
Berkas</a>
    <hr/>
    <table border="1" width="75%" style="text-align:center;">
      <tr>
        <th>No</th>
        <th>Image</th>
        <th>Keterangan</th>
        <th>Action</th>
      </tr>
      <?php
        $no = 1;
        foreach($berkas->result() as $row)
        {
          ?>
          <tr>
            <td><?php echo $no++; ?></td>
            <td></td>
            <td><?php echo $row->keterangan_berkas; ?></td>
            <td><a href="<?php echo base_url();
?>index.php/upload/download/<?php echo $row->kd_berkas;
?>">Download</a></td>
          </tr>
          <?php
        }
      ?>
    </table>
  </body>
</html>
```

Untuk Mengakses Halaman ini anda bisa menggunakan alamat : localhost/belajarupload/index.php/upload

No	Image	Keterangan	Action
1		Donat Coklat	Download
2		Nastar Coklat	Download

Keterangan :

```
<a href="<?php echo base_url(); ?>index.php/upload/create">Tambah Berkas</a>
```

Line 8 kita buat link untuk mengakses function create pada controller upload.

```
foreach($berkas->result() as $row)
```

Line 19 kita buat perulangan untuk extract data dari variabel \$berkas yang telah berisi data dari tabel tb_berkas, dan setiap perulangannya diwakili dengan variabel \$row.

```
<td><?php echo $row->keterangan_berkas; ?></td>
```

Line 25 kita gunakan untuk menampilkan keterangan atas berkas, mengambil nilai dari kolom keterangan_berkas, pada tabel tb_berkas

```
<td><a href="<?php echo base_url(); ?>index.php/upload/download/<?php echo $row->kd_berkas; ?>">Download</a></td>
```

Line 26 kita buat kita tuliskan perintah untuk membuat link untuk mengakses function download didalam controller upload, dengan menyertakan nilai pada kolom kd_berkas dari masing – masing berkas.

Berikutnya kita buat function download yang digunakan untuk membuat fitur download dari berkas.

5.2.4. Buat Function Download pada controller Upload

Berikutnya kita buat function download didalam Controller Upload, sehingga untuk controller Upload menjadi seperti berikut ini :

```
<?php
defined('BASEPATH') OR exit('No direct script access allowed');

class Upload extends CI_Controller {

    function create()
    {
        $this->load->view('form_upload');
```



```

    }

    function proses()
    {
        $config['upload_path']      = './uploads/';
        $config['allowed_types']    = 'gif|jpg|png';
        $config['max_size']         = 100;
        $config['max_width']        = 1024;
        $config['max_height']       = 768;
        $config['encrypt_name']     = TRUE;
        $this->load->library('upload', $config);
        if ( ! $this->upload->do_upload('berkas'))
        {
            $error = array('error' => $this->upload-
>display_errors());
            $this->load->view('form_upload', $error);
        }
        else
        {
            $data['nama_berkas'] = $this->upload->data("file_name");
            $data['keterangan_berkas'] = $this->input-
>post('keterangan_berkas');
            $data['tipe_berkas'] = $this->upload->data('file_ext');
            $data['ukuran_berkas'] = $this->upload->data('file_size');
            $this->db->insert('tb_berkas',$data);
            redirect('upload');
        }
    }

    public function index()
    {
        $data['berkas'] = $this->db->get('tb_berkas');
        $this->load->view('tampil_berkas',$data);
    }

    function download($id)
    {
        $data = $this->db->get_where('tb_berkas',['kd_berkas'=>$id])->row();
        force_download('uploads/'.$data->nama_berkas,NULL);
    }
}

```

Keterangan :

Kita Menambahkan function download di line 43 – 47

```
function download($id)
```

Pada function download kita menyertakan variabel \$id sebagai parameter, variabel \$id berisi nilai dari kd_berkas, dari berkas yang akan didownload

```
$data = $this->db->get_where('tb_berkas', ['kd_berkas'=>$id])->row();
```

Line 45 kita membuat variabel \$data untuk menyimpan record hasil query pada tabel tb_berkas dengan kd_berkas sesuai dengan nilai dari variabel \$id

```
force_download('uploads/'. $data->nama_berkas, NULL);
```

Line 46 kita gunakan perintah force_download untuk melakukan proses download, function force_download adalah perintah bawaan dari codeigniter yang diperuntukkan untuk download file, untuk dapat menggunakan function force_download anda diharuskan load helper 'download'

pada function force_download terdapat 2 parameter yang dapat diisi yaitu :

1. nama file yang akan didownload, nama file disini harus dilengkapi dengan lokasi dari filenya
2. nama file yang akan diberikan saat proses download dilakukan.

nah dalam contoh ini, kita mengisikan nilai pada parameter pertama, dengan nama berkas yang akan didownload, nama berkas disini kita ambil dari nilai kolom nama_berkas dari tabel tb_berkas, tidak hanya itu kita juga menyertakan direktori dari tempat menyimpan berkas itu, yaitu pada folder uploads.

Pada parameter 2 kita berikan nilai NULL, karena kita akan memberikan nama pada file ini sesuai nama berkas yang terupload.

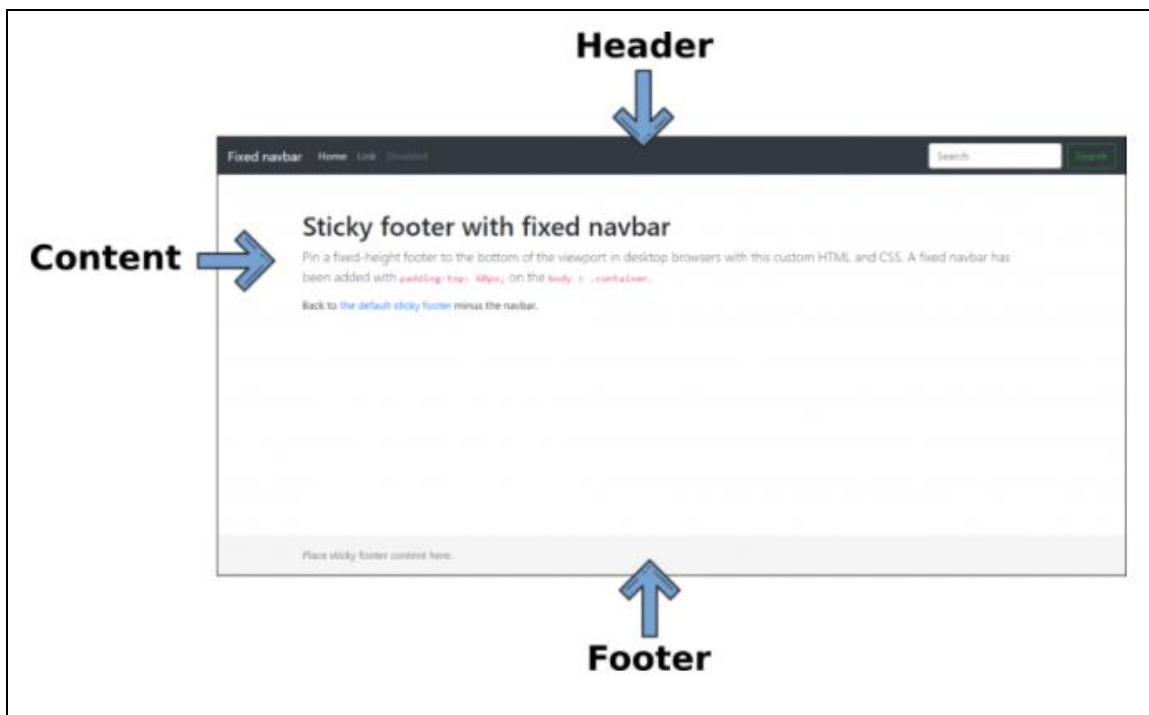
BAB 6. Cara Membuat Template di Codeigniter

Dalam proses pembuatan aplikasi, pastinya kita membutuhkan tampilan template, dimana pada bagian – bagian tertentu seperti contohnya header, menu, serta bagian footer akan bersifat statis (tidak berubah), sedangkan untuk bagian content akan bersifat dinamis karena akan menyesuaikan tampilan dari menu yang diakses.

Baik kita disini akan contohkan menggunakan template contoh dari bootstrap, bisa teman teman akses dalam alamat berikut ini :

<https://getbootstrap.com/docs/4.0/examples/sticky-footer-navbar/>

Untuk pembagian bagian templatanya adalah sebagai berikut :



Secara garis besar template diatas kita akan bagi menjadi 3 bagian ya :

- Header
- Content
- Footer

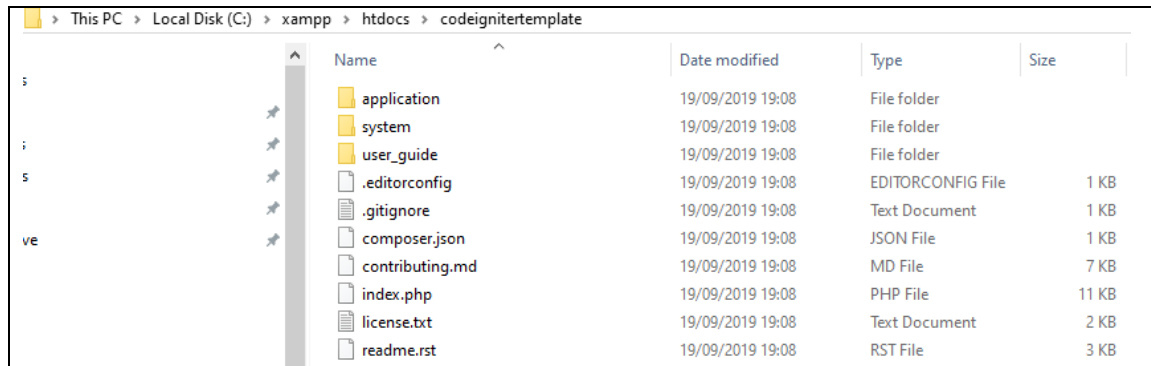
Untuk bagian header dan footer akan bersifat statis, dan untuk bagian content akan bersifat dinamis dimana akan menyesuaikan dari fitur yang sedang diakses.

Baik untuk langkah – langkahnya adalah sebagai berikut :

6.1. Mempersiapkan File codeigniter

disini kita telah mempersiapkan file framework codeigniter, yang telah kita letakkan di dalam direktori web server, karena kita menggunakan xampp, jadi kita letakkan di Drive C:/xampp/htdocs

saya simpan dalam folder dengan nama codeignitertemplate



6.1.1. Setting Bagian Config

Sebelum kita mengatur untuk bagian tampilan template, kita akan setting terlebih dahulu untuk file config, silahkan buka file application/config/config.php

Rubah bagian base_url

```
$config['base_url'] = 'http://localhost/codeignitertemplate';
```

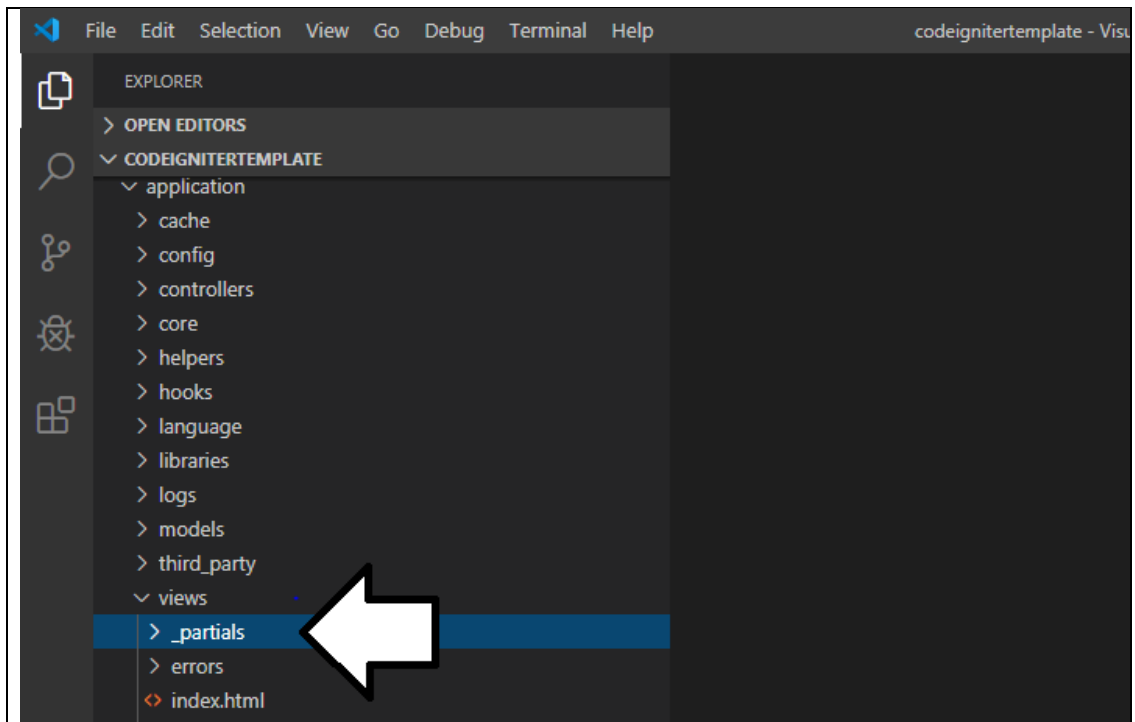
kita set menjadi http://localhost/codeignitertemplate karena kita akan sesuaikan dengan folder dari framework codeigniternya

```
$autoload['helper'] = array('url');
```

pada bagian autoload helper kita menuliskan helper url, agar dapat mengakses dari url dari aplikasinya.

6.1.2. Membuat Bagian Template

Berikutnya kita akan membuat bagian dari templatennya, silahkan buat folder dengan nama _partials didalam folder application/views



dalam folder `_partials` ini nanti kita akan simpan bagian dari templatnya, seperti header dan footer, untuk penamaan folder tidak selalu `_partials` anda bisa memberikan nama lainnya.

6.1.3. Membuat Bagian Header

buat file dengan nama `header.php` dan simpan pada folder `_partials`, dan untuk isi codenya adalah sebagai berikut :

```

<!doctype html>
<html lang="en">
  <head>
    <meta charset="utf-8">
    <meta name="viewport" content="width=device-width, initial-scale=1, shrink-to-fit=no">
    <meta name="description" content="">
    <meta name="author" content="">
    <link rel="icon" href="<?php echo base_url('assets/favicon.ico') ?>">

    <title>Sticky Footer Navbar Template for Bootstrap</title>

    <link rel="canonical" href="https://getbootstrap.com/docs/4.0/examples/sticky-footer-navbar/">

    <!-- Bootstrap core CSS -->
    <link href="<?php echo base_url('assets/bootstrap.min.css') ?>" rel="stylesheet">

    <!-- Custom styles for this template -->
    <link href="<?php echo base_url('assets/sticky-footer-navbar.css') ?>"
rel="stylesheet">

```

```

<!-- Bootstrap core JavaScript
===== -->
<!-- Placed at the end of the document so the pages load faster -->
<script src="<?php echo base_url('assets/jquery-3.2.1.slim.min.js') ?>"
integrity="sha384-
KJ3o2DKtIkvYIK3UEJmM7KCKRr/rE9/Qpg6aAZGJwFDMVNA/GpGFF93hXpG5KkN"
crossorigin="anonymous"></script>
<script src="<?php echo base_url('assets/popper.min.js') ?>"></script>
<script src="<?php echo base_url('assets/bootstrap.min.js') ?>"></script>
</head>

<body>

<header>
<!-- Fixed navbar -->
<nav class="navbar navbar-expand-md navbar-dark fixed-top bg-dark">
<a class="navbar-brand" href="#">Fixed navbar</a>
<button class="navbar-toggler" type="button" data-toggle="collapse" data-
target="#navbarCollapse" aria-controls="navbarCollapse" aria-expanded="false" aria-
label="Toggle navigation">
<span class="navbar-toggler-icon"></span>
</button>
<div class="collapse navbar-collapse" id="navbarCollapse">
<ul class="navbar-nav mr-auto">
<li class="nav-item active">
<a class="nav-link" href="<?php echo base_url();
?>index.php/page/index">Home<span class="sr-only">(current)</span></a>
</li>
<li class="nav-item">
<a class="nav-link" href="<?php echo base_url();
?>index.php/page/profil">Profil</a>
</li>
<li class="nav-item">
<a class="nav-link" href="<?php echo base_url();
?>index.php/page/about">About</a>
</li>
</ul>
<form class="form-inline mt-2 mt-md-0">
<input class="form-control mr-sm-2" type="text" placeholder="Search" aria-
label="Search">
<button class="btn btn-outline-success my-2 my-sm-0"
type="submit">Search</button>
</form>
</div>
</nav>
</header>

```

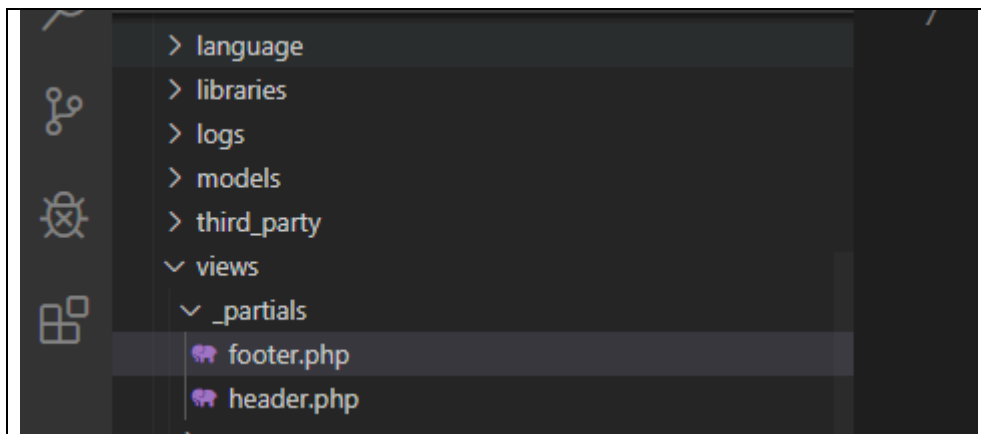
Untuk bagian menu dimulai dari line 40 kita menggunakan helper `base_url` serta memanggil controller `page`, disertai dengan function yang akan kita buat di controller `page` nantinya.

6.1.4. Membuat bagian footer

Berikutnya buat file dengan nama footer.php dengan isi code seperti berikut ini :

```
<footer class="footer">
  <div class="container">
    <span class="text-muted">Place sticky footer content here.</span>
  </div>
</footer>
</body>
</html>
```

Sehingga untuk isi dari folder _partials kurang lebih seperti berikut ini :

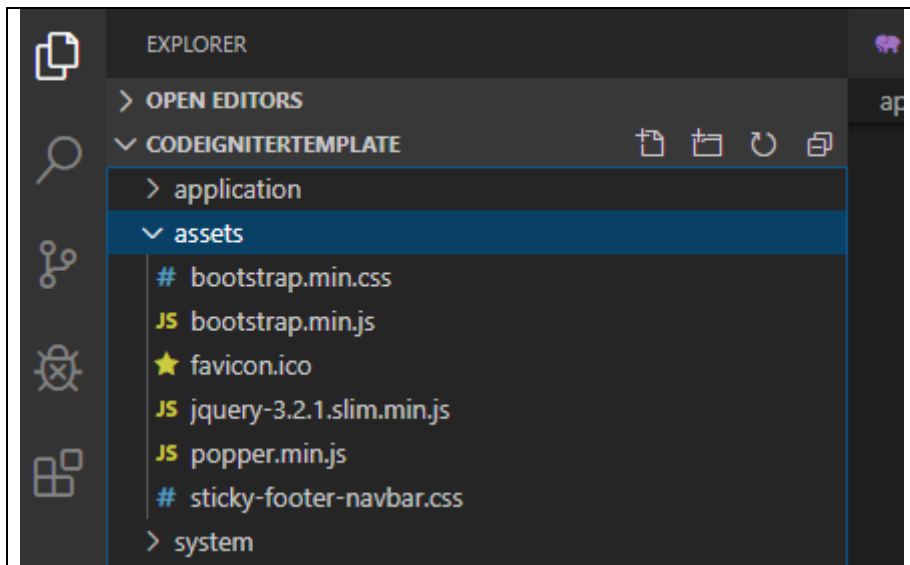


6.1.5. Mempersiapkan library JS dan CSS

Untuk dapat menampilkan template sesuai dari contoh, anda perlu mempersiapkan untuk library CSS dan JS, silahkan buat folder dengan nama assets didalam direktori project codeigniter.

Berikutnya silahkan download untuk file berikut yang berisi library css dan js untuk keperluan template, untuk download klik disini

setelah download dan extract, berikutnya letakkan file library css dan library js, ke dalam folder assets



6.1.6. Buat Controller

Berikutnya silahkan buat controller, dalam contoh ini saya buat controller dengan nama Page.

untuk isi codenya adalah sebagai berikut :

```
<?php
defined('BASEPATH') OR exit('No direct script access allowed');

class Page extends CI_Controller {

    public function index()
    {
        $this->load->view('_partials/header');
        $this->load->view('home');
        $this->load->view('_partials/footer');
    }

    public function profil()
    {
        $this->load->view('_partials/header');
        $this->load->view('profil');
        $this->load->view('_partials/footer');
    }

    public function about()
    {
        $this->load->view('_partials/header');
        $this->load->view('about');
        $this->load->view('_partials/footer');
    }

}
```

Keterangan :

kita buat 3 function yaitu :

- index
- profil
- about

dimana masing – masing function menuliskan 3 kali load view, yaitu :

- load view header untuk file berada di folder `_partials/header.php`
- load view yang berisi content, nah untuk bagian ini akan berbeda ditiap function, karena nantinya isinya juga dinamis disesuaikan dengan menu yang diakses.
- load view footer untuk file berada di folder `_partials/footer.php`

yang terpenting adalah load view content, setiap anda membuat menu baru, anda harus membuat view baru.

6.1.7. Membuat File View

Buat file `home.php` didalam folder `view`, dengan code seperti berikut ini :

```
<main role="main" class="container">
  <h1 class="mt-5">Home Page</h1>
  <p class="lead">Selamat Datang Pada Halaman Home</p>
  <p>Back to <a href=" ../sticky-footer/">the default sticky footer</a> minus the
  navbar.</p>
</main>
```

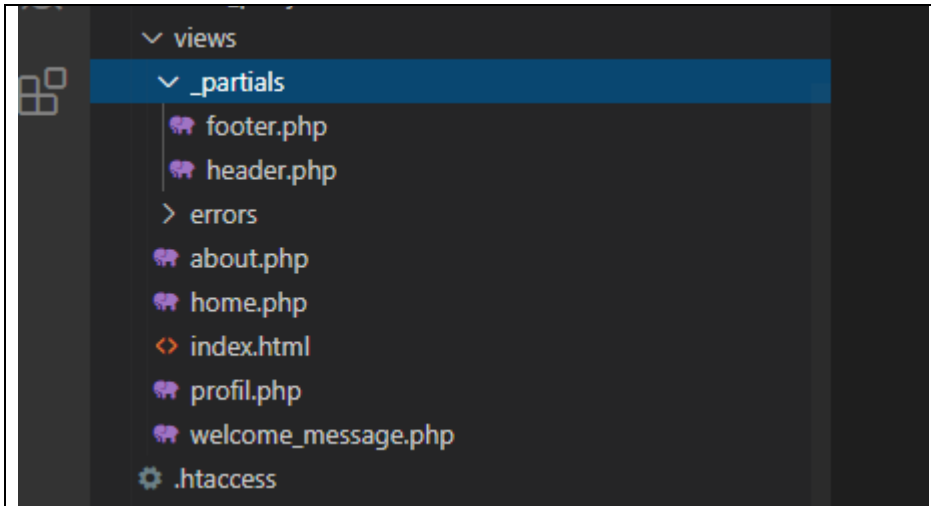
Buat file `profil.php` didalam folder `view`, dengan code seperti berikut ini :

```
<main role="main" class="container">
  <h1 class="mt-5">Profil Page</h1>
  <p class="lead">Selamat Datang Pada Halaman Profil</p>
  <p>Back to <a href=" ../sticky-footer/">the default sticky footer</a> minus the
  navbar.</p>
</main>
```

Buat file `about.php` didalam folder `view`, dengan code seperti berikut ini :

```
<main role="main" class="container">
  <h1 class="mt-5">Profil Page</h1>
  <p class="lead">Selamat Datang Pada Halaman Profil</p>
  <p>Back to <a href=" ../sticky-footer/">the default sticky footer</a> minus the
  navbar.</p>
</main>
```

Jadi kurang lebih seperti berikut ini :



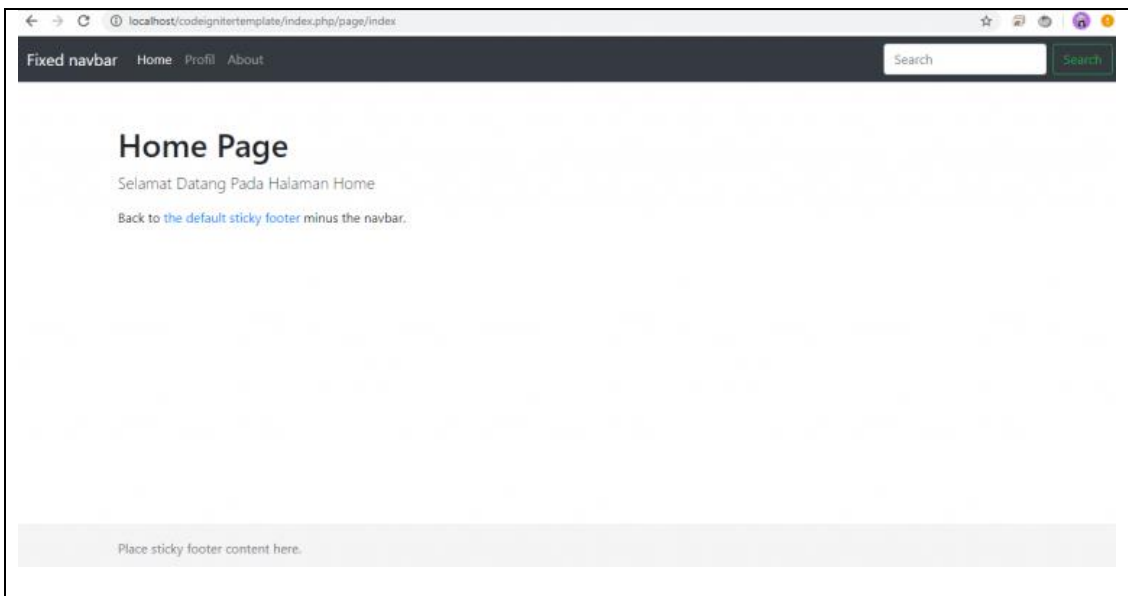
Keterangan :

Jadi kita buat 3 file view, dengan berbeda tampilan, harapannya nanti ketika kita mengakses halaman yang berbeda maka tampilannya akan juga berbeda.

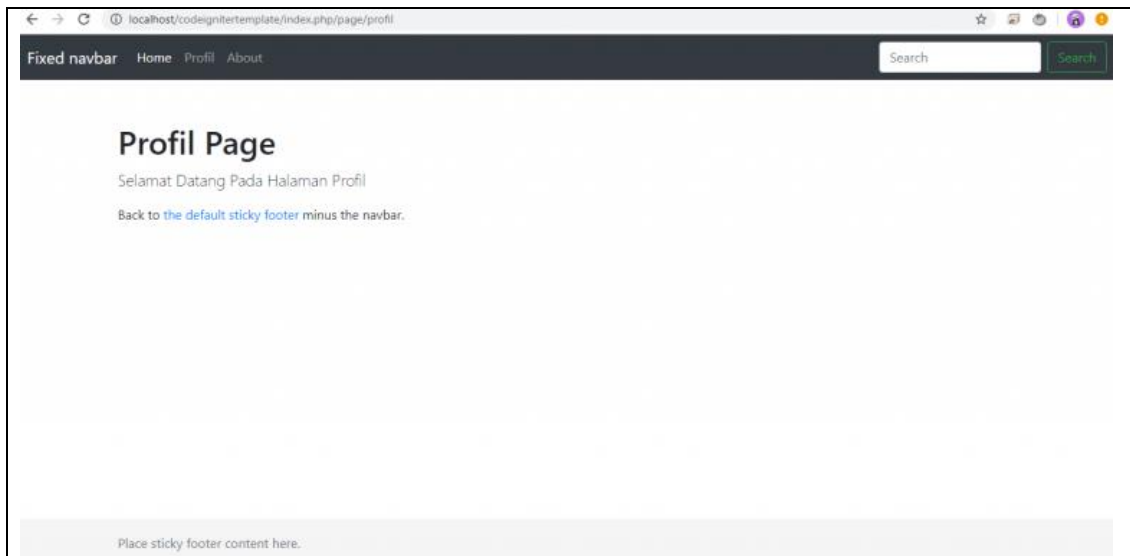
6.1.8. Testing tampilan template

Berikutnya kita akan coba untuk mengakses tampilan templatnya.

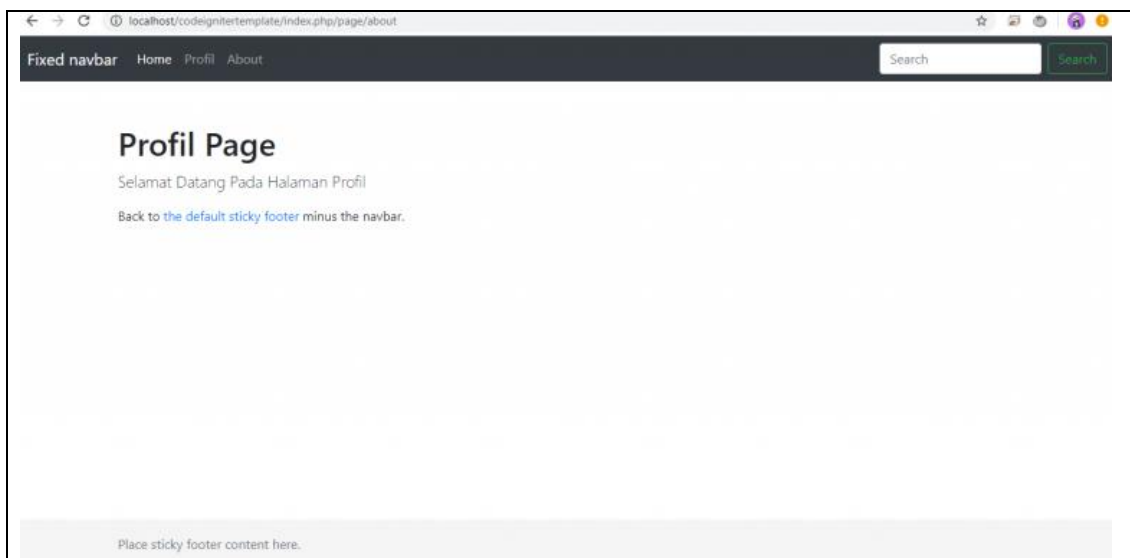
akses alamat : <http://localhost/codeignitertemplate/index.php/page>



akses alamat : <http://localhost/codeignitertemplate/index.php/page/profil>



akses alamat : <http://localhost/codeignitertemplate/index.php/page/about>



Jadi ketika kita mengakses dimenu yang berbeda yang berubah hanyalah tampilan content saja (bagian tengah) untuk bagian header dan footer tidak akan berubah.

6.2. Cara Menghubungkan Codeigniter dengan Database MySQL

Dalam Codeigniter anda sangat dipermudah untuk melakukan komunikasi dengan database,

Konsep komunikasi yang dilakukan codeigniter dengan database, adalah dengan mensetting file dengan nama `application/config/database.php`, setelah itu model akan bertugas berkomunikasi dengan tabel di database.. dan data tersebut akan dipassing ke controller dan bisa ditampilkan dibagian view.

6.2.1. Buat Database

Sebelum kita belajar untuk melakukan koneksi ke database, kita akan buat databasenya terlebih dahulu, dalam contoh ini kita buat database dengan nama warungbelajardb

```
--phpMyAdmin SQL Dump
-- version 4.9.0.1
-- https://www.phpmyadmin.net/
--
-- Host: 127.0.0.1
-- Waktu pembuatan: 19 Jan 2020 pada 15.07
-- Versi server: 10.4.6-MariaDB
-- Versi PHP: 7.3.9

SET SQL_MODE = "NO_AUTO_VALUE_ON_ZERO";
SET AUTOCOMMIT = 0;
START TRANSACTION;
SET time_zone = "+00:00";

/*!40101 SET @OLD_CHARACTER_SET_CLIENT=@@CHARACTER_SET_CLIENT */;
/*!40101 SET @OLD_CHARACTER_SET_RESULTS=@@CHARACTER_SET_RESULTS */;
/*!40101 SET @OLD_COLLATION_CONNECTION=@@COLLATION_CONNECTION */;
/*!40101 SET NAMES utf8mb4 */;

--
-- Database: `warungbelajardb`
--
-----

--
-- Struktur dari tabel `tb_pegawai`
--

CREATE TABLE `tb_pegawai` (
  `kd_pegawai` int(11) NOT NULL,
  `nama_pegawai` varchar(255) NOT NULL,
  `jenis_kelamin` enum('PRIA','WANITA') NOT NULL,
  `alamat_pegawai` varchar(255) NOT NULL
) ENGINE=InnoDB DEFAULT CHARSET=latin1;

--
-- Dumping data untuk tabel `tb_pegawai`
--

INSERT INTO `tb_pegawai` (`kd_pegawai`, `nama_pegawai`,
`jenis_kelamin`, `alamat_pegawai`) VALUES
(1, 'ANTON NUGRAHA', 'PRIA', 'JL.GELATIK NO. 150, RUNGKUT, SURABAYA'),
(2, 'DITA ANGGRAINI', 'WANITA', 'JL.BINTANG BERSINAR, NO. 141, GUNUNG
ANYAR, SURABAYA');
```

```

--
-- Indexes for dumped tables
--
--
-- Indeks untuk tabel `tb_pegawai`
--
ALTER TABLE `tb_pegawai`
  ADD PRIMARY KEY (`kd_pegawai`);

--
-- AUTO_INCREMENT untuk tabel yang dibuang
--
--
-- AUTO_INCREMENT untuk tabel `tb_pegawai`
--
ALTER TABLE `tb_pegawai`
  MODIFY `kd_pegawai` int(11) NOT NULL AUTO_INCREMENT,
  AUTO_INCREMENT=3;
COMMIT;

/*!40101 SET CHARACTER_SET_CLIENT=@OLD_CHARACTER_SET_CLIENT */;
/*!40101 SET CHARACTER_SET_RESULTS=@OLD_CHARACTER_SET_RESULTS */;
/*!40101 SET COLLATION_CONNECTION=@OLD_COLLATION_CONNECTION */;

```

Saya asumsikan teman – teman sudah membuat database diatas, dan juga sudah memiliki tabel tb_pegawai didalamnya.

6.2.2. Setting koneksi di database

Saya asumsikan teman – teman telah memiliki project codeigniter, dalam contoh ini project codeigniter saya beri nama “pegawai”.

Untuk proses instalasi dari codeigniter, bisa anda lihat di tutorial kita sebelumnya : Cara Menginstall Framework Codeigniter

Kita setting terlebih dahulu untuk file database.php

Buka file application/config/database.php

```

$db['default'] = array(
    'dsn' => "",
    'hostname' => 'localhost',
    'username' => 'root',
    'password' => "",
    'database' => 'warungbelajardb',
    'dbdriver' => 'mysqli',
    'dbprefix' => "",
    'pconnect' => FALSE,

```

```

'db_debug' => (ENVIRONMENT !== 'production'),
'cache_on' => FALSE,
'cachedir' => "",
'char_set' => 'utf8',
'dbcollat' => 'utf8_general_ci',
'swap_pre' => "",
'encrypt' => FALSE,
'compress' => FALSE,
'stricton' => FALSE,
'failover' => array(),
'save_queries' => TRUE
);

```

Keterangan :

- Silahkan setting pada bagian konfigurasi databasenya
- Pada Line 2 kita tuliskan hostname dari database, dalam contoh ini kita tuliskan localhost
- Pada Line 4 kita tuliskan username dari databasenya, dalam contoh ini kita tuliskan root
- Pada Line 5 kita tuliskan password dari username di database tersebut, dalam contoh ini saya biarkan kosong, karena settingan default dari xampp adalah kosong, nanti bisa teman – teman sesuaikan.
- Pada Line 6 kita tuliskan nama databasenya, dalam contoh ini kita tuliskan “warungbelajardb”
- Pada Line 7 kita tuliskan dbdriver, dalam contoh ini kita tuliskan mysqli karena kita akan mengkoneksikan ke database mysqli

6.2.3. Load Library database

Berikutnya kita akan load terlebih dahulu untuk library database di codeigniter, agar codeigniter dapat berkomunikasi dengan database.

Ada beberapa cara untuk load library database, yaitu dengan beberapa cara :

1. Menuliskan perintah load library database, pada function __construct di setiap controller.
2. Load Library dibagian file autoload

dalam contoh ini kita akan load library di file autoload, agar tidak perlu load di masing – masing controller, silahkan buka file application/config/autoload.php

```

$autoload['libraries'] = array('database');

```

Pada Line 61 tambahkan pada bagian libraries, tuliskan database pada bagian array.

Perintah ini digunakan untuk load library database, agar dapat digunakan di seluruh bagian controller.

6.2.4. Buat Model

Berikutnya kita akan buat model yang digunakan untuk berkomunikasi dengan database, dalam contoh ini kita buat model dengan nama M_pegawai.php

Untuk tutorial pembuatan model bisa dilihat di tutorial berikut ini : Cara Membuat Model, View, dan Controller

Posisi file model berada pada folder application/models

```
<?php
class M_pegawai extends CI_Model{
    function get_data(){
        return $this->db->get('tb_pegawai');
    }
}
```

Keterangan :

- Pada Line 2 kita buat class model dengan nama M_pegawai
- Pada Line 3 kita buat function dengan nama get_data,
- Pada Line 4 dalam function get_data kita menuliskan perintah untuk mengambil data pada tabel tb_pegawai
- untuk mendapatkan data pada tabel tb_pegawai kita cukup menuliskan

```
$this->db->get('tb_pegawai');
```

- kita tinggal menuliskan nama tabel didalam function get, karena dalam contoh ini kita mengambil data pada tabel tb_pegawai sehingga kita tuliskan tb_pegawai pada function get.
- pada line 4 kita menuliskan perintah return karena kita akan mengembalikan nilai hasil query untuk mendapatkan data di tabel pegawai.

Sehingga saat function get_data pada model M_pegawai akan dipanggil, maka akan mengembalikan nilai yang berisi data pada tabel tb_pegawai.

6.2.5. Buat Controller

Berikutnya kita akan buat controller, dalam contoh ini kita buat controller dengan nama Pegawai.php

posisi file controller berada folder application/controller/Pegawai.php

untuk isi dari file controller adalah sebagai berikut :

```
<?php
defined('BASEPATH') OR exit('No direct script access allowed');

class Pegawai extends CI_Controller {

    function __construct(){
```

```

        parent::__construct();
        $this->load->model('M_pegawai');
    }

    function index(){
        $data['pegawai'] = $this->M_pegawai->get_data();
        $this->load->view('vw_pegawai.php',$data);
    }
}

```

Keterangan :

- Pada Line 4 kita tuliskan class Pegawai sebagai nama class dari Controllernya
- Pada Line 6 kita buat function __construct dimana perintah yang kita tulis didalam function __construct akan otomatis dijalankan ketika class controller tersebut diakses.
- Pada Line 8 kita tuliskan perintah \$this->load->model('M_pegawai'); yang digunakan untuk load Model M_pegawai.
- Pada Line 11 kita tuliskan function index dimana didalamnya kita menuliskan perintah :

```
$data['pegawai'] = $this->M_pegawai->get_data();
```

- artinya kita mengakses function get_data didalam model M_pegawai, yang mana akan mengembalikan nilai yang berisi data pada tabel tb_pegawai, dan nilai tersebut akan disimpan di dalam element pegawai pada array \$data.
- Pada Line 13 kita load file view dengan nama vw_pegawai.php dan kita passing nilai pada variabel \$data, yang didalamnya telah berisi data dari tabel tb_pegawai.

6.2.6. Buat File View

Berikutnya kita buat file view dengan nama vw_pegawai.php, untuk codenya adalah sebagai berikut :

```

<!DOCTYPE html>
<html>
<head>
    <title>Belajar Codeigniter</title>
</head>
<body>
    <h1>Belajar Codeigniter di warungbelajar.com</h1>
    <table border="1">
        <tr>
            <th>Nama</th>
            <th>Jenis Kelamin</th>
            <th>Alamat</th>
        </tr>
        <?php
        foreach($pegawai->result() as $row)
        {
            ?>
        <tr>

```



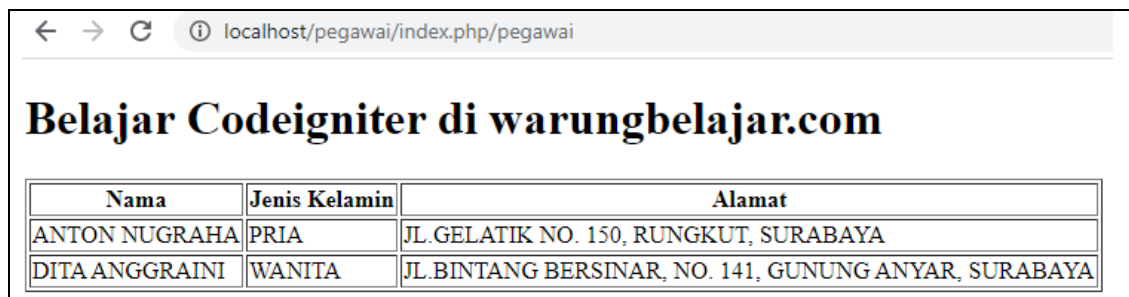
```

        <td><?php echo $row->nama_pegawai ?></td>
        <td><?php echo $row->jenis_kelamin ?></td>
        <td><?php echo $row->alamat_pegawai ?></td>
    </tr>
    <?php
    }
    ?>
</table>
</body>
</html>

```

Untuk mengakses tampilan view, kita akses dengan alamat : `http://localhost/pegawai/index.php/pegawai`

Maka tampilannya adalah sebagai berikut :



Nama	Jenis Kelamin	Alamat
ANTON NUGRAHA	PRIA	JL.GELATIK NO. 150, RUNGKUT, SURABAYA
DITA ANGGRAINI	WANITA	JL.BINTANG BERSINAR, NO. 141, GUNUNG ANYAR, SURABAYA

maka akan menampilkan data pegawai dalam bentuk tabel.

Keterangan :

- Kita buat tabel untuk menampilkan data pada line 8 – 26
- Pada line 15 kita gunakan perintah foreach untuk extract data yang pada variabel \$pegawai hasil passing data dari controller.

```
foreach($pegawai->result() as $row)
```

- kita extract data pada variabel \$pegawai dan setiap perulangannya kita gunakan variabel \$row untuk mewakili setiap datanya.
- Sehingga pada line 19 kita gunakan perintah \$row->nama_pegawai untuk menampilkan nilai pada kolom nama_pegawai.
- Pada line 20 kita gunakan perintah \$row->jenis_kelamin untuk menampilkan nilai pada kolom jenis_kelamin
- Pada line 21 kita gunakan perintah \$row->alamat_pegawai untuk menampilkan nilai pada kolom alamat_pegawai.

BAB 7. Penanganan Form & Form Validasi di Codeigniter

Penanganan form disini maksudnya adalah bagaimana cara codeigniter 4 ini dapat menangkap inputan dari sebuah form dan memprosesnya dibagian controller.

7.1. Buat Sebuah Controller Siswa

Berikutnya silahkan buat controller dengan nama Siswa.php dan simpan didalam folder app/Controllers, untuk isi controller Siswa.php adalah sebagai berikut ini :

```
<?php
namespace App\Controllers;

class Siswa extends BaseController
{
    public function index()
    {
        return view('formSiswa');
    }
    public function save()
    {
        echo "<pre>";
        var_dump($this->request->getVar());
        echo "</pre>";
    }
}
```

Keterangan :

- kita membuat controller dengan nama Siswa dimana controller ini memiliki method index.
- jika method index ini diakses maka akan mengakses file view dengan nama formSiswa.php
- berikutnya ada method save yang nantinya akan kita isi dengan perintah untuk menangkap inputan form

7.1.1. Membuat File View dengan nama formSiswa.php

Berikutnya kita akan membuat file view dengan nama formSiswa.php, file view ini akan berisi form, kita juga gunakan bootstrap untuk mengatur tampilannya, .. teman – teman bisa copy paste saja dengan code dibawah ini :

```
<!DOCTYPE html>
<html lang="en">
```

```

<head>
  <title>Full Width Form</title>
  <meta charset="utf-8">
  <meta name="viewport" content="width=device-width, initial-scale=1">
  <!-- CSS only -->
  <link rel="stylesheet"
href="https://stackpath.bootstrapcdn.com/bootstrap/4.5.2/css/bootstrap.min.css">

  <!-- JS, Popper.js, and jQuery -->
  <script src="https://code.jquery.com/jquery-3.5.1.slim.min.js"></script>
  <script
src="https://cdn.jsdelivr.net/npm/popper.js@1.16.1/dist/umd/popper.min.js"></scri
pt>
  <script
src="https://stackpath.bootstrapcdn.com/bootstrap/4.5.2/js/bootstrap.min.js"></scri
pt>
</head>

<body>
  <div class="container">
    <div class="card">
      <div class="card-header bg-primary text-white">Tambah Data Siswa</div>
      <div class="card-body">
        <?php if (!empty(session()->getFlashdata('error'))): ?>
          <div class="alert alert-danger alert-dismissible fade show" role="alert">
            <h4>Periksa Entrian Form</h4>
            <hr />
            <?php echo session()->getFlashdata('error'); ?>
            <button type="button" class="close" data-dismiss="alert" aria-
label="Close">
              <span aria-hidden="true">&times;</span>
            </button>
          </div>
        <?php endif; ?>
        <form action="<?= base_url('siswa/save'); ?>" method="post">
          <?= csrf_field(); ?>
          <div class="form-group">
            <label for="nama">Nama</label>
            <input type="text" name="nama" class="form-control" value="<?=
old('nama'); ?>" id="nama">
          </div>
          <div class="form-group">
            <label for="email">Email</label>
            <input type="email" name="email" class="form-control" id="email"
value="<?= old('email'); ?>">
          </div>

          <div class="form-group">
            <label for="password">Password</label>
            <input type="password" name="password" class="form-control"
id="password">
          </div>

          <div class="form-group">

```

```

        <label for="alamat">Alamat</label>
        <textarea name="alamat" id="alamat" class="form-control" cols="3"
rows="3"><?= old('alamat'); ?></textarea>
    </div>
    <button type="submit" class="btn btn-primary">Save</button>
</form>
</div>
</div>
</div>
</body>
</html>

```

Keterangan :

Bagian tampilan form ini kita menggunakan bootstrap, dan untuk library js dan css kita langsung mengaksesnya secara online, jadi kita sarankan untuk terkoneksi internet saat mengakses tampilan form.

Pada line 32 kita mengisi beberapa attribute di tag <form>, seperti bagian

```
action="<?= base_url('siswa/save'); ?>"
```

yang artinya ketika tombol save di klik maka entrian form akan di proses didalam method save pada controller Siswa

Pada Line 33 kita menuliskan perintah :

```
<?= csrf_field(); ?>
```

perintah diatas digunakan untuk memberikan token CSRF (Cross-Site Request Forgery) dibagian form, kami sarankan teman – teman menggunakan perlindungan CSRF ini.

Berikutnya kita buat beberapa inputan pada bagian form, seperti halnya kita membuat form di HTML, kita perlu memberikan value pada attribute name di masing – masing inputan form. (Line 36 name="nama"), (Line 40 name="email"), (Line 45 name="password"), (Line 50 name="alamat")

Tombol submit kita buat di line 52

Untuk mengakses tampilan form teman – teman bisa mengaktifkan local development server terlebih dahulu dengan perintah “php spark serve” lalu bisa mengakses url :

```
localhost:8080/siswa
```

Tampilannya adalah sebagai berikut :

7.1.2. Mengakses Inputan Form di Controller

Sebelumnya kita sudah membuat tampilan form, berikutnya kita akan belajar menuliskan perintah yang digunakan untuk mengakses inputan dibagian form.

Seperti yang telah kita tuliskan dibagian attribute action pada tag form, ketika tombol submit di form di klik, maka inputan form akan diproses dibagian method save di controller Siswa, perhatikan method save di controller Siswa kita menuliskan perintah sebagai berikut :

```
public function save()
{
    echo "<pre>";
    var_dump($this->request->getVar());
    echo "</pre>";
}
```

Keterangan :

Untuk mengakses seluruh inputan form kita bisa menggunakan perintah

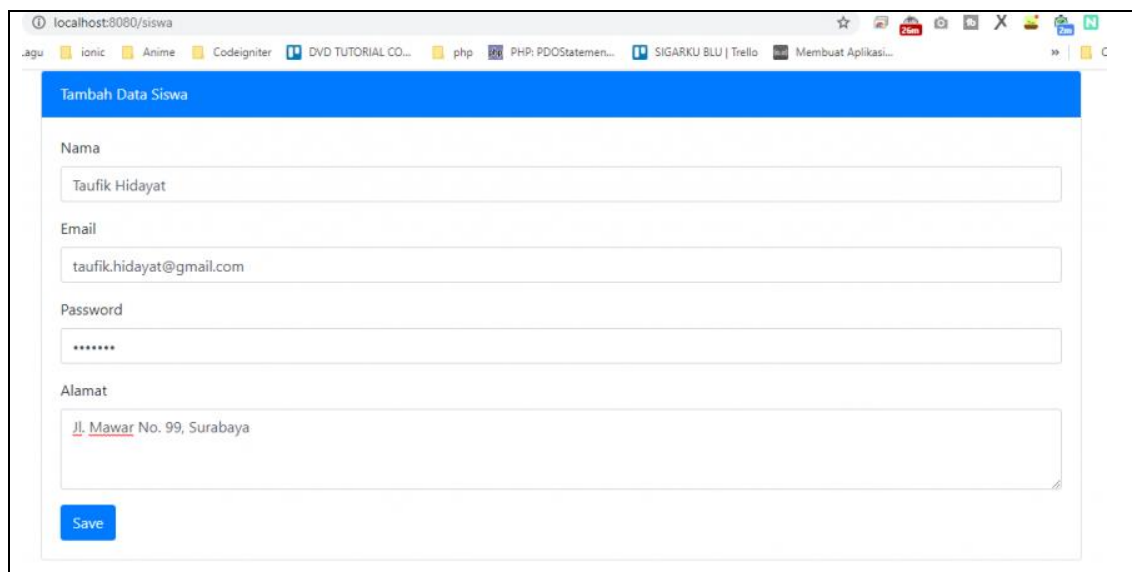
```
$this->request->getVar()
```

perintah diatas berlaku untuk inputan dengan tipe method post ataupun get, jika anda ingin mengakses inputan form bagian tertentu, bisa menuliskan value name dibagian input di method getVar, semisal dari form diatas kita ingin mengakses inputan email, maka bisa menggunakan perintah :

```
$this->request->getVar('email')
```

Berikutnya kita menggunakan perintah print_r kita gunakan untuk menampilkan hasil inputan form.

baik kita akan coba lakukan percobaan untuk entry form, silahkan akses form dengan alamat : localhost:8080/siswa, dan isikan bagian formnya lalu klik tombol save untuk submit form.

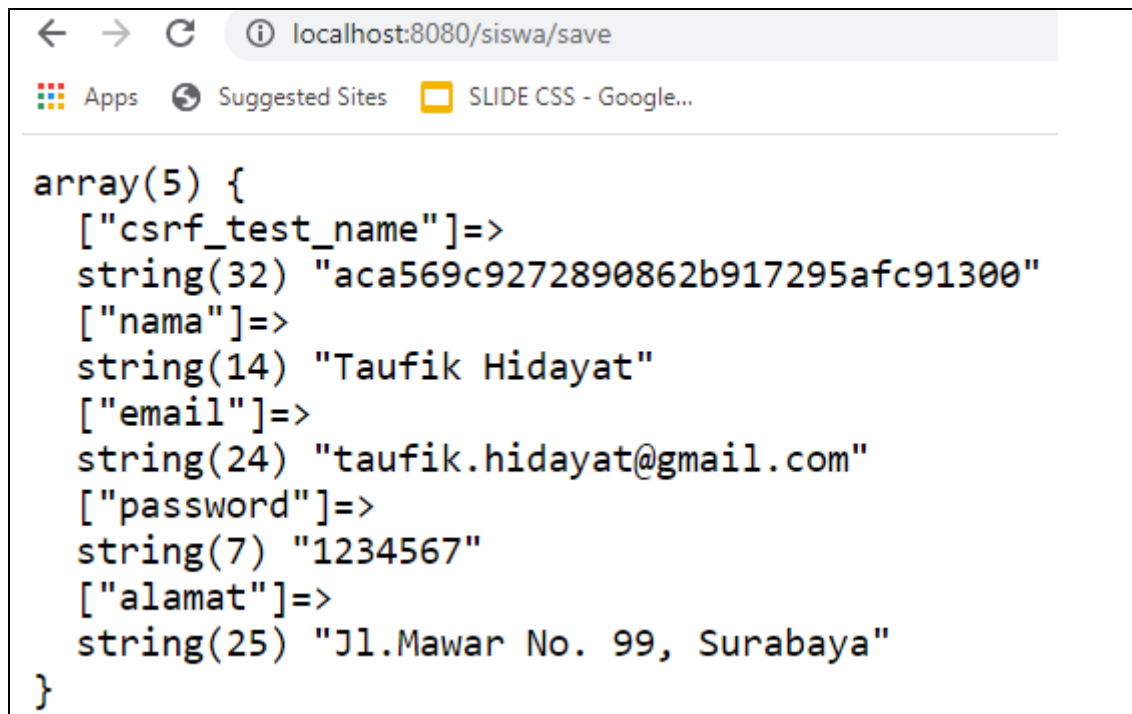


The screenshot shows a web browser window with the address bar displaying 'localhost:8080/siswa'. The page title is 'Tambah Data Siswa'. The form contains the following fields and values:

- Nama: Taufik Hidayat
- Email: taufik.hidayat@gmail.com
- Password: *****
- Alamat: Jl. Mawar No. 99, Surabaya

A blue 'Save' button is located at the bottom left of the form.

Ketika kita klik tombol save, maka akan mengakses method save di controller siswa, dan akan menampilkan seperti gambar dibawah.



The screenshot shows a web browser window with the address bar displaying 'localhost:8080/siswa/save'. The page content displays the following PHP array output:

```
array(5) {  
  ["csrf_test_name"]=>  
  string(32) "aca569c9272890862b917295afc91300"  
  ["nama"]=>  
  string(14) "Taufik Hidayat"  
  ["email"]=>  
  string(24) "taufik.hidayat@gmail.com"  
  ["password"]=>  
  string(7) "1234567"  
  ["alamat"]=>  
  string(25) "Jl.Mawar No. 99, Surabaya"  
}
```

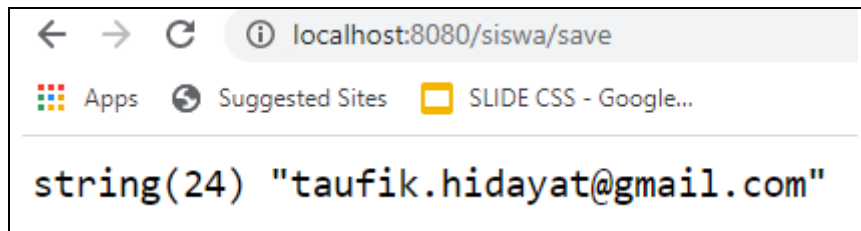
terlihat kita sudah berhasil menangkap inputan berdasarkan name dari inputan form.

Semisal kita akan menangkap salah satu dari inputan saja, semisal kita ingin menangkap inputan email saja, kita bisa merubah method save menjadi berikut ini :

```
public function save()
```

```
{
  var_dump($this->request->getVar('email'));
}
```

Maka saat kita entry bagian form lalu klik tombol save, hasilnya adalah sebagai berikut :



Hasilnya hanya akan menampilkan inputan dengan name email saja.

7.1.3. MEMBUAT VALIDASI FORM

Berikutnya kita akan belajar untuk membuat validasi form di codeigniter 4, validasi form ini sangat penting sekali untuk mengatur inputan form yang dilakukan oleh pengguna aplikasi, sehingga ketika inputan form akan di insert ke database sesuai dengan struktur tabel di database.

Untuk membuat validasi form, kita akan merubah method save di controller Siswa, sehingga method save menjadi seperti berikut ini :

```
public function save()
{
  if (!$this->validate([
    'nama' => [
      'rules' => 'required',
      'errors' => [
        'required' => '{field} Harus diisi'
      ]
    ],
    'email' => [
      'rules' => 'required|valid_email',
      'errors' => [
        'required' => '{field} Harus diisi',
        'valid_email' => 'Format Email Harus Valid'
      ]
    ],
    'password' => [
      'rules' => 'required',
      'errors' => [
        'required' => '{field} Harus diisi'
      ]
    ],
    'alamat' => [
      'rules' => 'required',
      'errors' => [
        'required' => '{field} Harus diisi'
      ]
    ]
  ]))
  {
    // ...
  }
}
```

```
    ]
  ]
  )) {
    session()->setFlashdata('error', $this->validator->listErrors());
    return redirect()->back()->withInput();
  } else {
    print_r($this->request->getVar());
  }
}
```

Keterangan :

Line 3 kita gunakan perintah IF dimana melakukan pengecekan dari penggunaan method \$this->validate.

method validate, pada method validate kita isi parameternya dengan array multidimensi, dimana bisa anda isi dengan rules (aturan) dan message error (pesan error)

Misal jika kita lihat lagi :

```
'nama' => [
  'rules' => 'required',
  'errors' => [
    'required' => '{field} Harus diisi'
  ]
],
```

artinya inputan nama, rulesnya 'required' (harus diisi), dan jika tidak diisi akan menampilkan pesan error "nama Harus diisi", {field} ini akan bernilai nama inputannya.

```
'email' => [
  'rules' => 'required|valid_email',
  'errors' => [
    'required' => '{field} Harus diisi',
    'valid_email' => 'Format Email Harus Valid'
  ]
],
```

artinya inputan email, rulesnya 'required' (harus diisi), dan jika tidak diisi akan menampilkan pesan error "email Harus diisi", selain itu terdapat rules valid_email artinya inputan harus menggunakan format email yang valid, jika tidak diisi maka akan menampilkan pesan error "Format Email Harus Valid"

Jika terdapat rules yang bernilai false (inputan form tidak sesuai aturan) maka akan menjalankan perintah di line 42 – 43.

Line 42 kita kirimkan error dengan menggunakan session flashdata, flashdata adalah istilah pengiriman data lewat session, tapi hanya satu kali pakai, biasanya digunakan untuk mengirimkan pesan / alert.

dalam contoh ini kita gunakan untuk mengirimkan pesan error dari form validation, setFlashdata memiliki 2 parameter yaitu :

- nama flashdata
- value flashdata

```
session()->setFlashdata('error', $this->validator->listErrors());
```

pada contoh diatas kita membuat flashdata dengan nama error yang berisi list error dari form validation

Line 43 kita tuliskan perintah :

```
return redirect()->back()->withInput();
```

Perintah diatas berfungsi untuk redirect ke halaman sebelumnya (form input), yang artinya adalah halaman formSiswa, dengan menyertakan inputan yang sudah dientrikan sebelumnya (withInput)

Line 45 akan dijalankan jika inputan form sudah sesuai rules.. kita tuliskan aja sebagai contoh menggunakan perintah dd untuk menampilkan isi entrian form.

7.1.4. Testing Form Validation

Setelah kita setting bagian form validation pada method save di controller Siswa, berikutnya kita akan coba lakukan testing, kita coba akses dulu untuk formnya :

```
localhost:8080/Siswa
```

Maka tampilannya adalah seperti berikut ini :

Keterangan :

semisal kita isi bagian form untuk nama dan email, tapi bagian password dan alamat kita kosongkan, lalu kita tekan tombol Save

Maka hasilnya adalah sebagai berikut :

<https://www.warungbelajar.com/penanganan-form-dan-form-validasi-di-codeigniter-4.html>

7.2. Mengetahui URI Segment di Codeigniter

URI adalah singkatan dari Uniform Resource Identifier , dimana dengan menggunakan URI kita dapat mendapatkan data melalui url, hal ini juga berlaku di Framework Codeigniter.

URI Segment ini memiliki urutan seperti segment 1, segment 2, segment 3 dan seterusnya.

Nah biar tidak bigung, perhatikan alamat URL dibawah ini :

```
http://localhost/belajarcodeigniter/index.php/produk/data/1
```

Keterangan :

http://localhost/belajarcodeigniter/index.php merupakan alamat utama dari aplikasi web yang sedang kita buat.

- segment 1 = produk
- segment 2 = data
- segment 3 = 1

jadi penggunaan segment dimulai setelah alamat utama dari aplikasi web yang kita buat, karena kita menggunakan framework codeigniter, sehingga segment dimulai dari setelah index.php

karena secara default itulah yang menjadi alamat utama dari aplikasi yang kita buat pakai codeigniter, meskipun nantinya untuk penggunaan index.php ini bisa kita hilangkan, nanti kita akan bahas di tutorial selanjutnya.

nah kita kembali ke pembahasan uri segment ya.

secara sederhana konsep penulisan URL di codeigniter adalah seperti ini :

```
http://localhost/belajarcodigniter/index.php/controller/method/id
```

Keterangan :

http://localhost/belajarcodigniter/index.php adalah alamat utamanya

- segment 1 dimulai dari controller, sehingga pada contoh diatas produk yang merupakan controller dijadikan segment 1
- segment 2 adalah method, sehingga pada contoh diatas data merupakan segment 2
- segment 3 adalah id, sehingga pada contoh diatas 1 merupakan segment 3

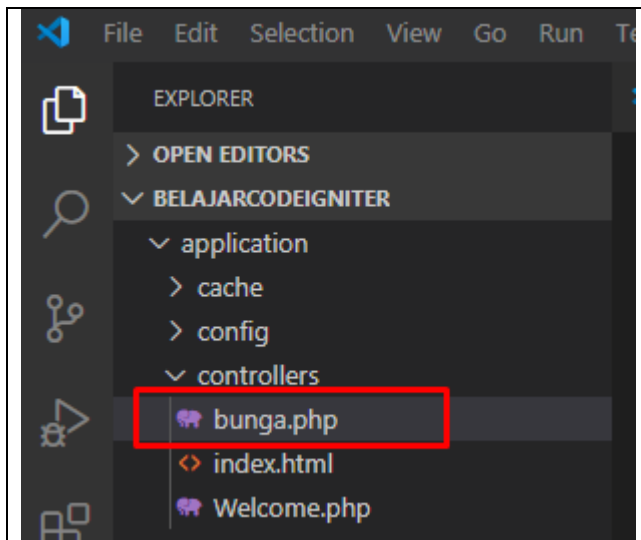
Oke gimana sudah mulai paham ya... baik biar gak bingung kebanyakan teori kita mulai praktekkan..

Kita buat dulu sebuah project codeigniter, untuk langkah – langkahnya silahkan bisa buka tutorial kita sebelumnya untuk proses instalasi dari codeigniter “Tutorial Instalasi Codeigniter”

Saya asumsikan kita sudah memiliki project codeigniter didalam folder belajarcodigniter

Silahkan buat Controller baru dengan nama bunga.php

file controller berada di application/controller/bunga.php



Isi Codenya adalah sebagai berikut :

```
<?php
defined('BASEPATH') OR exit('No direct script access allowed');

class Bunga extends CI_Controller {

    public function jenis(){
        echo "Bunga itu merupakan jenis " . $this->uri->segment('3');
    }
}
```

Silahkan disimpan, lalu coba kita mengaksesnya menggunakan url sebagai berikut :

```
http://localhost/belajarcodigniter/index.php/bunga/jenis/mawar
```

Maka tampilannya adalah sebagai berikut :



Keterangan :

Kita membuat controller dengan nama bunga, kita sesuaikan dengan nama filenya yaitu bunga.php (Line 4)

lalu kita buat method dengan nama jenis (Line 6)

didalam method jenis kita buat perintah untuk menampilkan text dengan menggunakan perintah echo

```
echo "Bunga itu merupakan jenis " . $this->uri->segment('3');
```

dalam perintah itu kita menuliskan perintah

```
$this->uri->segment('3');
```

yang artinya kita memerintahkan untuk mendapatkan nilai segment ketiga, sehingga menampilkan tulisan “Bunga merupakan jenis mawar”

ingat rumusnya :

```
http://localhost/belajarcodigniter/index.php/bunga/jenis/mawar
```

Segment 1 = bunga

Segment 2 = jenis

Segment 3 = mawar

Gimana Mudah kan... baik kita buat contoh lagi....

kita akan coba memanggil 5 segment sekaligus

Perhatikan perintah dibawah ini, kita membuat method baru untuk controller Bunga :

```
<?php
defined('BASEPATH') OR exit('No direct script access allowed');

class Bunga extends CI_Controller {

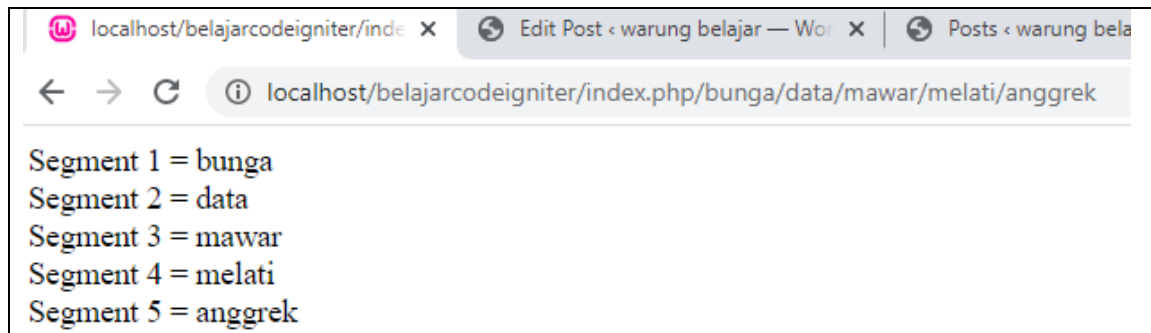
    public function jenis(){
        echo "Bunga itu merupakan jenis " . $this->uri->segment('3');
    }

    public function data(){
        echo "Segment 1 = ".$this->uri->segment('1');
        echo "<br/>";
        echo "Segment 2 = ".$this->uri->segment('2');
        echo "<br/>";
        echo "Segment 3 = ".$this->uri->segment('3');
        echo "<br/>";
        echo "Segment 4 = ".$this->uri->segment('4');
        echo "<br/>";
        echo "Segment 5 = ".$this->uri->segment('5');
    }
}
```

silahkan di simpan, lalu silahkan akses url sebagai berikut :

```
http://localhost/belajarcodigniter/index.php/bunga/data/mawar/melati/anggrek
```

Maka hasilnya adalah seperti berikut ini :



```
Segment 1 = bunga  
Segment 2 = data  
Segment 3 = mawar  
Segment 4 = melati  
Segment 5 = anggrek
```

Keterangan :

- Kita menambahkan method data pada line 11
- didalam method data kita menuliskan perintah echo untuk menampilkan text.
- Line 12 kita menuliskan perintah `$this->uri->segment('1')` maka akan menampilkan nilai dari segment 1 yaitu bunga yang merupakan nama dari controller
- Line 14 kita menuliskan perintah `$this->uri->segment('2')` maka akan menampilkan nilai dari segment 2 yaitu data yang merupakan nama dari method
- Line 16 kita menuliskan perintah `$this->uri->segment('3')` maka akan menampilkan nilai dari segment 3 yaitu mawar
- Line 18 kita menuliskan perintah `$this->uri->segment('4')` maka akan menampilkan nilai dari segment 4 yaitu melati
- Line 20 kita menuliskan perintah `$this->uri->segment('5')` maka akan menampilkan nilai dari segment 5 yaitu anggrek

Jadi kurang lebih seperti itu peran dari URI Segment yang dari digunakan untuk mendapatkan data dari URL Codeigniter.

7.3. Menghilangkan Index.php pada Codeigniter

Secara default ketika kita mengakses alamat dari web dengan codeigniter, anda akan melihat index.php pada alamat tersebut.

Contohnya seperti ini, kita memiliki project codeigniter dengan folder belajarcodigniter, dan kita akan mengakses controller dengan nama belajar, maka alamat yang kita gunakan adalah

```
http://localhost/belajarcodigniter/index.php/belajar
```

perhatikan kita menuliskan index.php sebelum kita mengakses nama controller, akan lebih baik jika mengaksesnya dengan menghilangkan index.php pada alamat, sehingga alamatnya adalah sebagai berikut :

```
http://localhost/belajarcodigniter/belajar
```

Nah dalam tutorial ini kita akan bahas cara menghilangkan index.php pada codeigniter, nah saya asumsikan teman – teman sudah memiliki project codeigniter, dan foldernya bernama belajarcodigniter.

Tutorial untuk menginstall Codeigniter bisa dilihat di tutorial kita sebelumnya : Tutorial Cara Menginstall Codeigniter

setelah itu silahkan buat controller dengan nama belajar.php, posisi file controller :

```
application/controllers/Belajar.php
```

untuk isi controller Belajar.php adalah sebagai berikut :

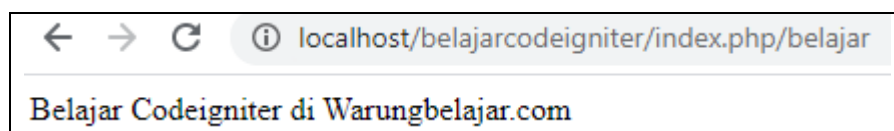
```
<?php
defined('BASEPATH') OR exit('No direct script access allowed');

class Belajar extends CI_Controller {

    public function index()
    {
        echo 'Belajar Codeigniter di Warungbelajar.com';
    }
}
```

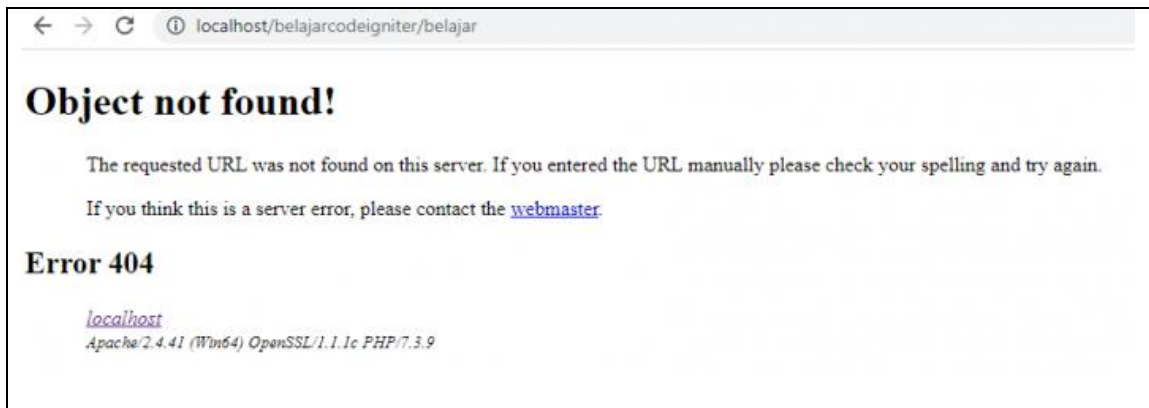
Nah untuk mengakses controller belajar, kita menggunakan alamat :

```
http://localhost/belajarcodigniter/index.php/belajar
```



kita akan coba mengaksesnya tanpa menggunakan index.php, kita mengakses dengan alamat :

```
http://localhost/belajarcodigniter/belajar
```



maka hasilnya akan menampilkan pesan error, karena kita menghilangkan index.php pada alamat saat mengakses controller.

7.3.1. Cara untuk menghilangkan index.php di codeigniter

Untuk menghilangkan index.php di codeigniter, langkah awal silahkan buka file dengan nama config.php didalam folder

application/config/config.php

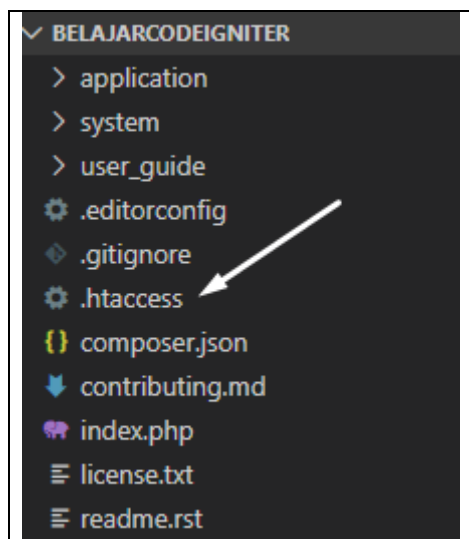
cari code seperti berikut ini, kurang lebih berada pada line 38 :

```
$config['index_page'] = 'index.php';
```

hapus bagian index.php, sehingga hasilnya adalah seperti berikut ini :

```
$config['index_page'] = '';
```

Langkah berikutnya silahkan buat file dengan nama .htaccess, didalam folder project codeigniter anda.

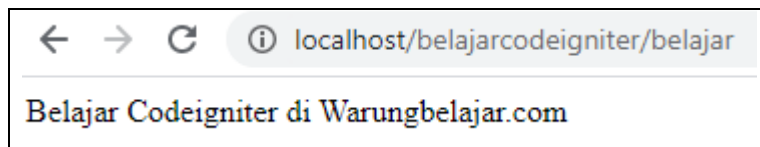


Untuk code didalam file .htaccess adalah sebagai berikut :


```
RewriteEngine On
RewriteCond %{REQUEST_FILENAME} !-f
RewriteCond %{REQUEST_FILENAME} !-d
RewriteRule ^(.*)$ index.php/$1 [L]
```

baik saya asumsikan teman – teman sudah memperbaiki settingan dibagian config.php dan membuat file .htaccess, sekarang kita coba mengakses controller, tanpa menggunakan index.php hasilnya adalah sebagai berikut :

```
http://localhost/belajarcodigniter/belajar
```



Maka hasilnya kita dapat mengakses controller tanpa perlu menuliskan index.php pada alamat url.

7.4. Cara Menerapkan Konsep HMVC pada Framework Codeigniter

Konsep HMVC ini akan memudahkan anda dalam melakukan management file model view dan controller, dengan membaginya menjadi folder – folder berdasarkan module dari aplikasi yang sedang anda buat.

HMVC sangat membantu ketika aplikasi yang anda buat memiliki banyak modules, sehingga anda dapat mengelompokkan masing – masing module itu menjadi beberapa folder yang namanya disesuaikan dengan modules, sehingga proses management filenya seperti model, view, dan controller dapat lebih mudah, karena terbagi menjadi folder berdasarkan modules

Saya akan bahas terlebih dahulu perbedaan penggunaan HMVC dan MVC biasa.

Jika anda menggunakan MVC biasa, maka anda diharuskan untuk meletakkan file Model, View, dan Controller di folder yang telah disediakan oleh Codeigniter.

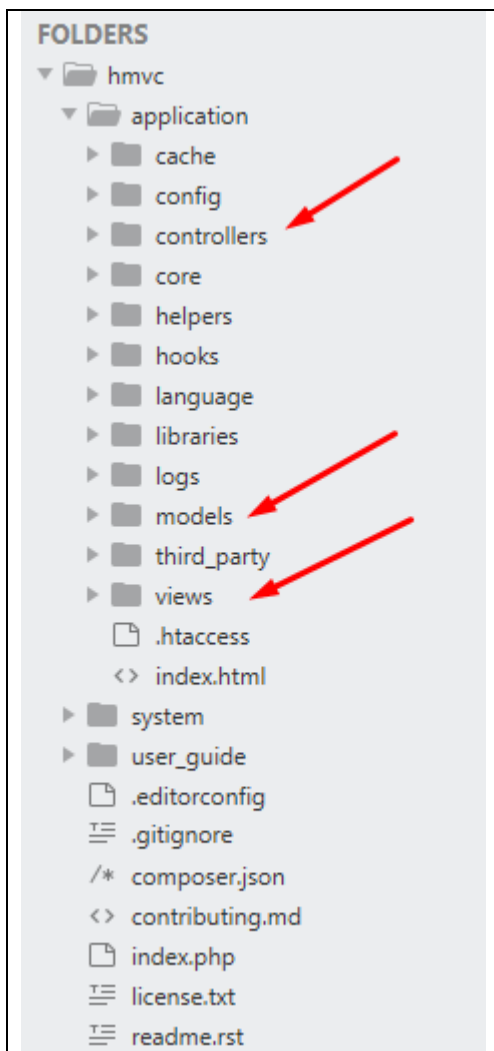
- Jika anda belum memahami konsep MVC bisa baca tutorial ini : Mengetahui Konsep MVC di Codeigniter
- Untuk Cara Membuat Controller, Model dan View bisa baca di tutorial ini : Membuat Model, View dan Controller di Codeigniter

Saya asumsikan anda telah memahami, konsep mvc, dan peran model, view, dan controller

secara default posisi dari tempat menyimpan file model, view dan controller adalah :

- Model : application/models
- Controller : application/controllers

- Views : application/views



Nah Kalau HMVC posisi tempat menyimpan Model, View, dan Controller akan terbagi menjadi beberapa folder, disesuaikan dengan nama module, nah biar tidak bingung kita langsung praktekkan aja ya, langkah – langkah untuk membuat HMVC di Codeigniter adalah sebagai berikut.

7.4.1. Install Extention Modular

Saya asumsikan teman – teman sudah menginstall framework Codeigniter, untuk tutorial instalasi codeigniter bisa dilihat di : Cara Menginstall Framework Codeigniter

dalam contoh ini kita memiliki project codeigniter dengan nama folder adalah hmvc.

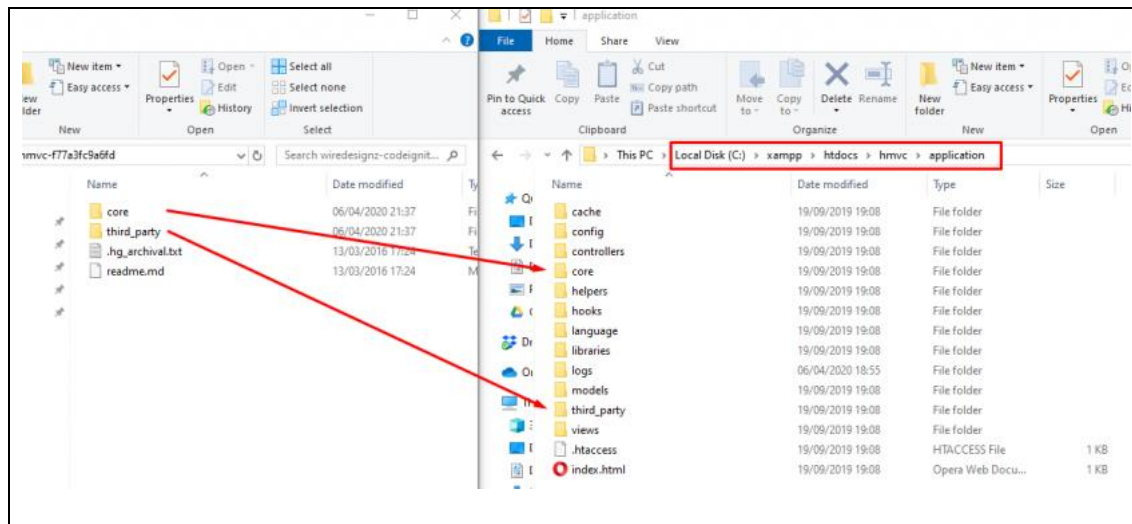
Untuk contoh ini versi codeigniter yang saya gunakan adalah Codeigniter 3.1.11, dan versi PHP 7.3.9

Untuk langkah awal kita harus menginstall extention Modular, agar dapat menggunakan konsep HMVC, anda bisa mendownload

di <https://bitbucket.org/wiredesignz/codeigniter-modular-extensions-hmvc/downloads/>

Setelah download extension modular, silahkan extract file tersebut,

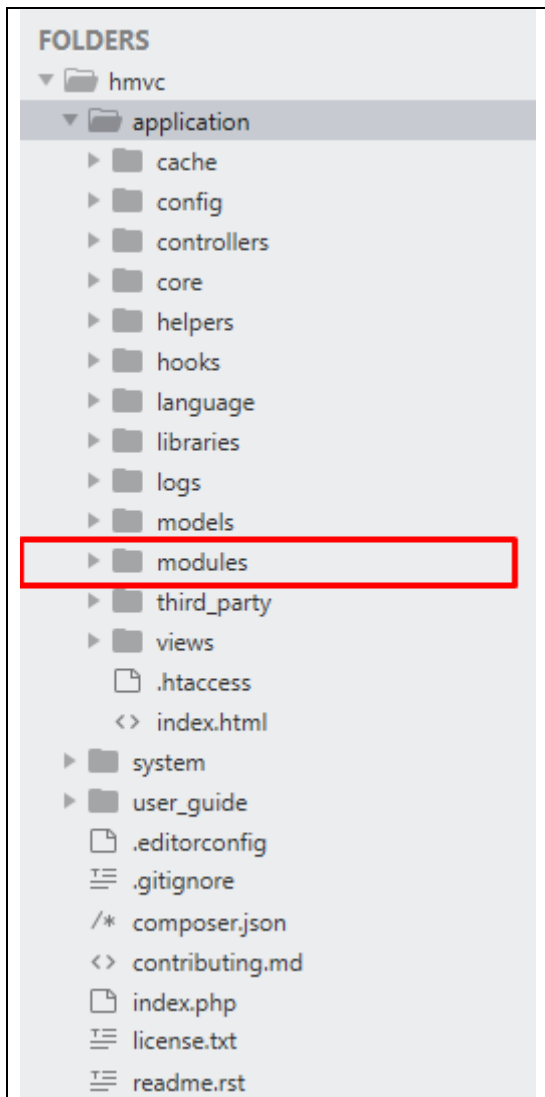
didalam folder extension modular itu ada folder core, dan third_party, silahkan pindahkan 2 folder tersebut ke dalam folder project codeigniter anda pada folder application.



silahkan replace seluruh file untuk folder core dan third_party di folder application di project codeigniter anda.

7.4.2. Buat Folder Modules di Folder Application

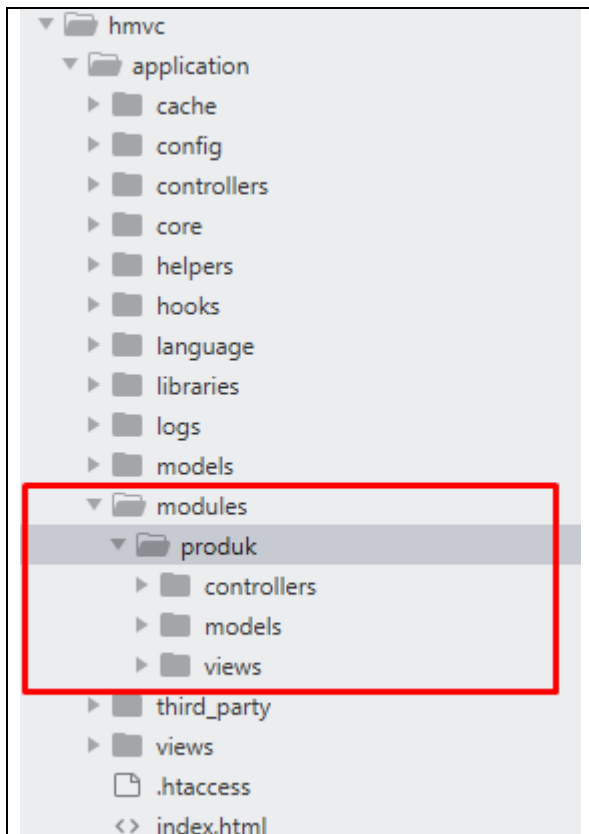
Berikutnya didalam folder application, buat folder dengan nama modules.



Lalu didalam folder modules, kita akan buat folder baru dengan nama produk

didalam folder produk, kita buat 3 folder didalamnya yaitu :

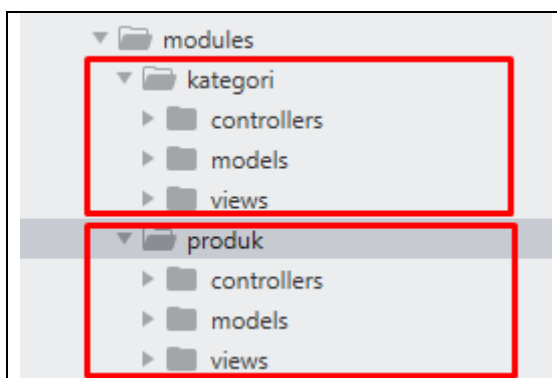
1. controllers
2. models
3. views



Keterangan :

Produk ini adalah nama modules, dan didalamnya terdapat folder controller, model, dan view, sehingga untuk membuat controller, model, dan view untuk keperluan modules produk, anda harus menyimpan file tersebut didalam folder produk ini.

Hal ini berlaku di module lain, contohnya anda akan buat modules kategori ya silahkan buat folder kategori didalam folder modules dan didalam folder kategori, buat folder controller, models, dan views, jadi masing masing modules, memiliki folder controller, models, dan views di masing masing folder nama modules



Oke kita kembali ke ke modules produk, silahkan buat file controller dengan nama Produk.php didalam folder application/modules/produk/controllers

```
<?php
```

```

defined('BASEPATH') OR exit('No direct script access allowed');

class Produk extends CI_Controller {

    public function index()
    {
        $this->load->view('vw_produk');
    }

}

```

Keterangan :

Perintah controller yang kita tulis ini sama dengan perintah menulis controller biasanya.

pada line 8 kita mengakses file view dengan nama vw_produk, nah yang berbeda dari konsep hmvc disini, posisi file vw_produk.php adalah didalam folder application/modules/produk/views, sehingga bisa kita simpulkan kalau kita buat controller didalam folder application/modules/produk/controllers, ya maka viewnya ada di folder application/modules/produk/views

jadi posisi file controller, model, dan view akan berada di masing masing folder pada bagian folder modules masing – masing

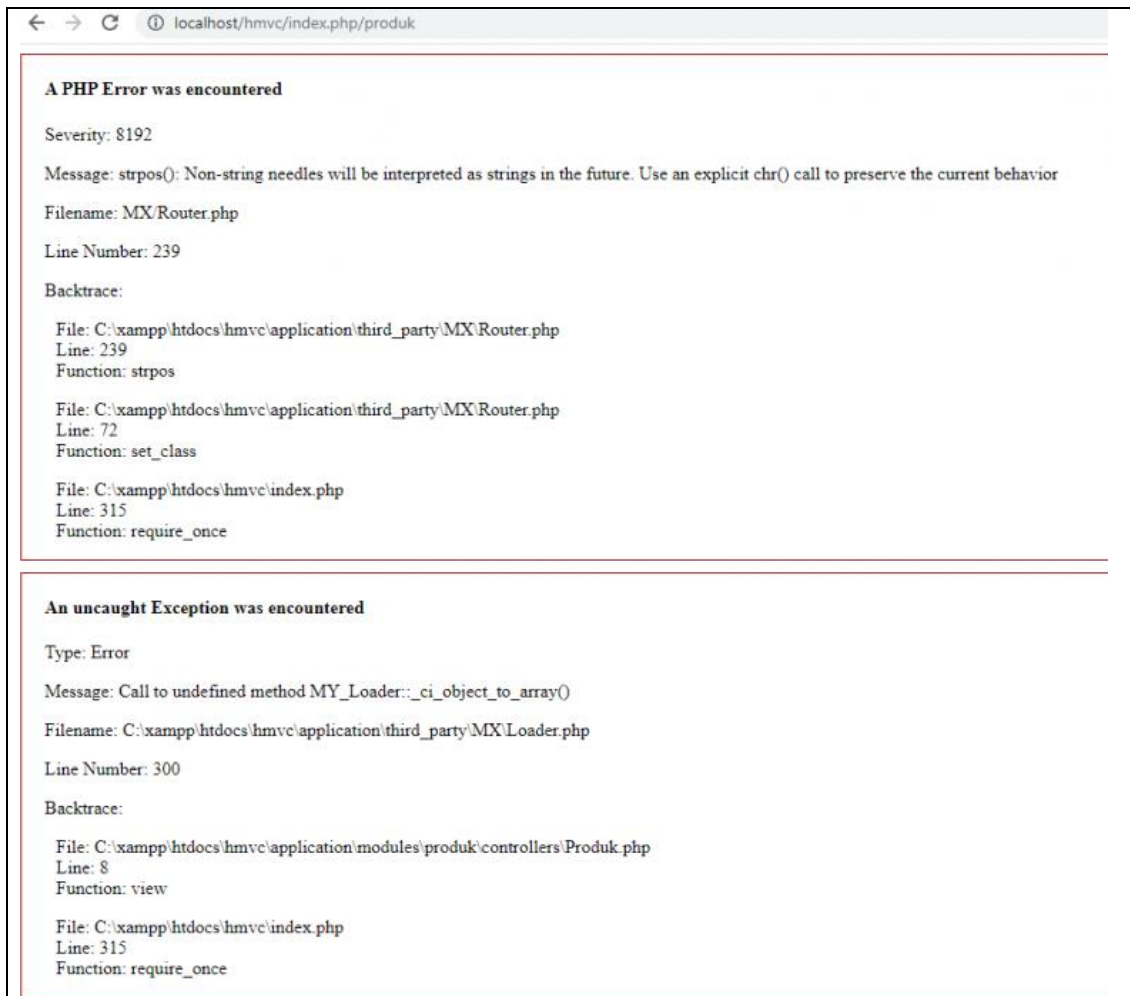
Berikutnya buat file view dengan nama vw_produk.php didalam folder application/modules/produk/views

```

<span style="color: #ff0000;"><html></span>
<head>
    <meta charset="UTF-8">
    <title>Belajar HMVC di Codeigniter</title>
</head>
<body>
    <h1>Ini adalah halaman module Produk</h1>
</body>
</html>

```

Berikutnya kita coba untuk mengaksesnya dengan alamat : localhost/hmvc/index.php/produk



hasilnya akan menampilkan error seperti gambar diatas, hal ini dikarenakan konsep HMVC di Codeigniter 3, dan kita menggunakan versi php 7, jadi kita perlu melakukan beberapa perbaikan code pada core dari codeigniter 3.

Perbaiki Error Non-string needles will be interpreted as strings in the future. Use an explicit chr() call to preserve the current behavior

Cara Memperbaikinya adalah sebagai berikut :

1. buka folder application/third_party/MX/Router.php
2. Berikutnya cari code function set_class, anda bisa menggunakan fasilitas pencarian di text editor yang anda gunakan, dengan menekan tombol CTRL + F, jika sudah ketemu, silahkan ubah codenya seperti dibawah ini

code sebelumnya :

```
public function set_class($class)
{
    $suffix = $this->config->item('controller_suffix');
    if (strpos($class, $suffix) === FALSE)
    {
        $class .= $suffix;
    }
}
```

```

    }
    parent::set_class($class);
}

```

Ubah menjadi seperti berikut ini :

```

public function set_class($class)
{
    $suffix = $this->config->item('controller_suffix');
    if( $suffix && strpos($class, $suffix) === FALSE) //kode perubahan
yang benar
    {
        $class .= $suffix;
    }
    parent::set_class($class);
}

```

Lalu silahkan simpan perubahannya.

Perbaiki Call to undefined method MY_Loader::_ci_object_to_array()

Berikutnya kita akan perbaiki error Call to undefined method MY_Loader::_ci_object_to_array(), langkah – langkahnya adalah sebagai berikut :

Silahkan buka file application/third_party/MX/Loader.php

Berikutnya cari function view (kurang lebih di line 290), lalu ubah code didalam function tersebut

Code Sebelumnya :

```

public function view($view, $vars = array(), $return = FALSE)
{
    list($path, $_view) = Modules::find($view, $this->_module, 'views/');

    if ($path != FALSE)
    {
        $this->_ci_view_paths = array($path => TRUE) + $this->_ci_view_paths;
        $view = $_view;
    }

    return $this->_ci_load(array('_ci_view' => $view, '_ci_vars' => $this->_ci_object_to_array($vars), '_ci_return' => $return));
}

```

Menjadi seperti berikut ini :

```

public function view($view, $vars = array(), $return = FALSE)
{
    list($path, $_view) = Modules::find($view, $this->_module, 'views/');

```



```
        if ($path != FALSE)
        {
            $this->_ci_view_paths = array($path => TRUE) + $this-
>_ci_view_paths;
            $view = $_view;
        }

        if (method_exists($this, '_ci_object_to_array'))
        {
            return $this->_ci_load(array('_ci_view' => $view,
'_ci_vars' => $this->_ci_object_to_array($vars), '_ci_return' => $return));
        } else {
            return $this->_ci_load(array('_ci_view' => $view,
'_ci_vars' => $this->_ci_prepare_view_vars($vars), '_ci_return' => $return));
        }
    }
}
```

Silahkan simpan perubahannya.

Berikutnya kita akan coba lagi mengakses alamat : localhost/hmvc/index.php/produk



Keterangan :

yang tampil adalah tulisan “Ini adalah halaman module Produk” yang merupakan isi dari bagian view vw_produk.

Jadi kurang lebih seperti itu konsep HMVC yang bisa kita terapkan di Codeigniter 3

BAB 8. Membuat CRUD (Create,Read, Update, Delete) di Codeigniter

kita akan membahas bagaimana cara membuat crud (Create, Read, Update, Delete) di codeigniter, harapannya anda dapat melakukan komunikasi antara project codeigniter anda dengan database, setidaknya di operasional standart seperti Create, Read, Update, dan Delete.

Crud yang kita buat disini dengan study kasusnya adalah data siswa, untuk datanya antara lain menyimpan data :

- nama
- jenis_kelamin
- tempat_lahir
- tanggal_lahir
- no_telp
- alamat

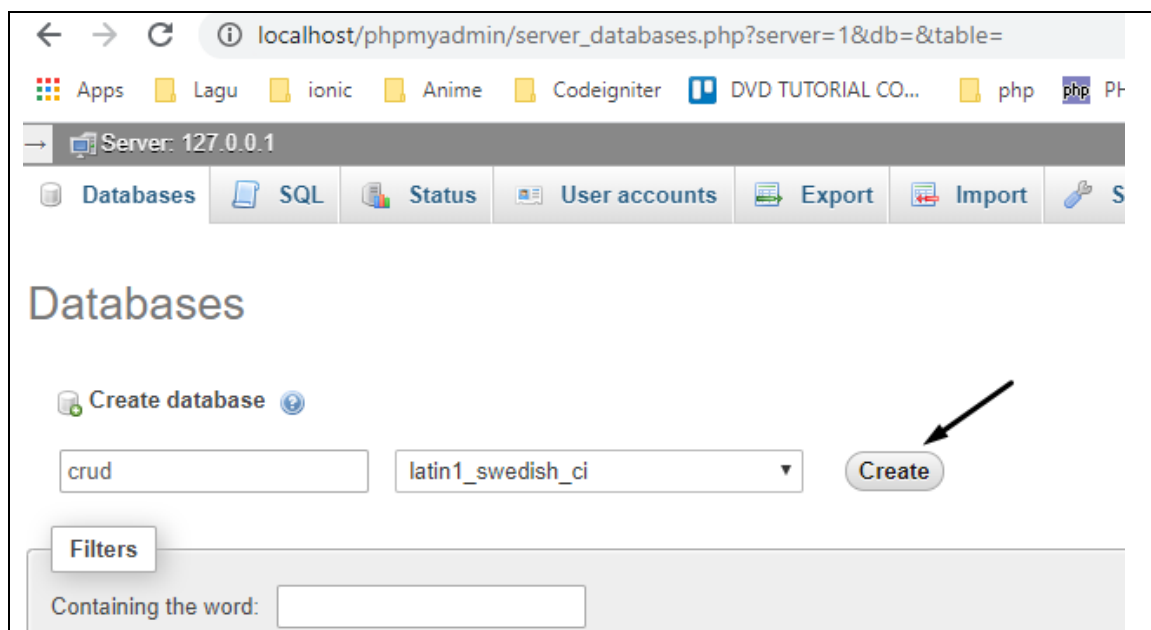
Beberapa hal yang akan kita gunakan dalam tutorial ini antara lain adalah :

1. PHP 7.3.9
2. MySQL
3. Visual Studio Code (Text Editor)
4. Bootstrap 4

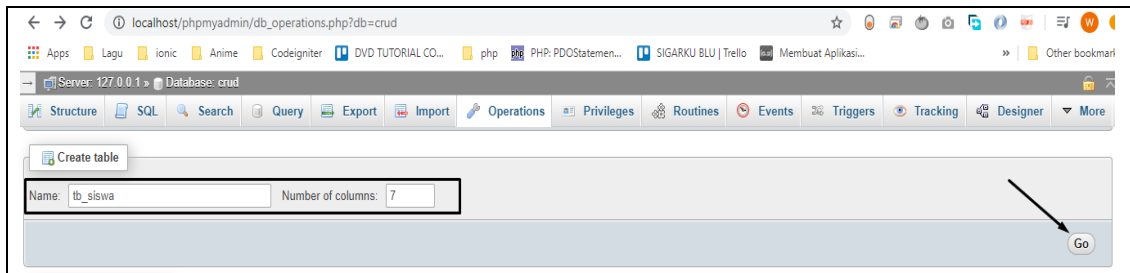
8.1. Mempersiapkan Database

Baik untuk langkah awal kita akan buat terlebih dahulu databasenya, anda bisa menggunakan phpmyadmin untuk memudahkan anda dalam membuat database dan tabelnya, silahkan buka localhost/phpmyadmin untuk membuka phpmyadmin

lalu dibagian database kita buat database baru, semisal dengan nama crud, lalu klik tombol create

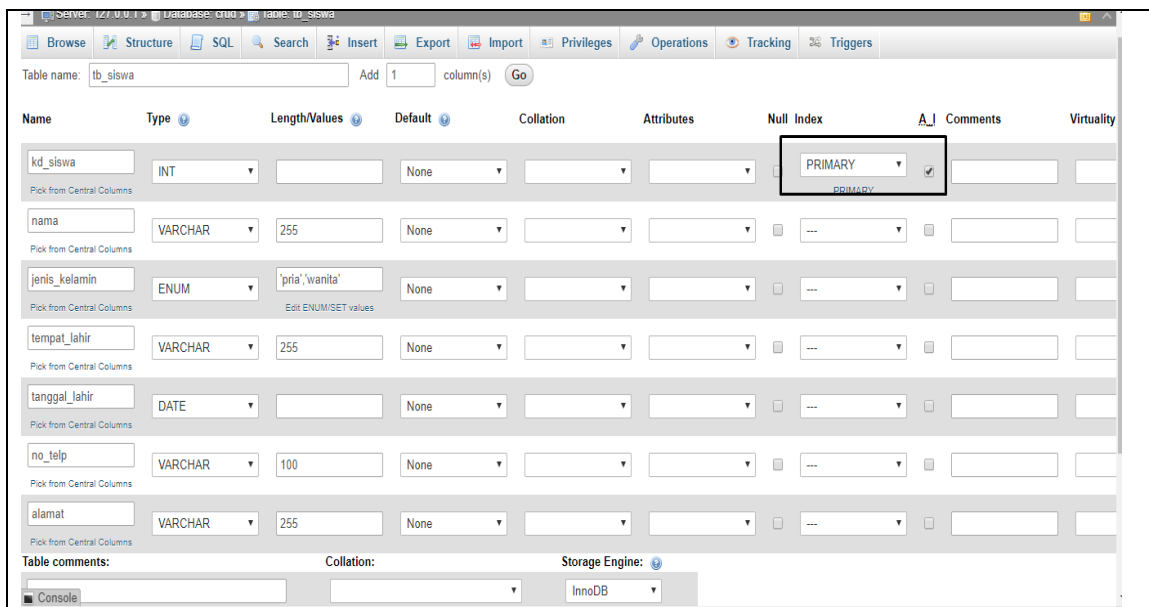


Berikutnya buat tabel dengan nama tb_siswa dengan jumlah kolom adalah 7, lalu klik Go



Berikutnya buat beberapa kolom dengan struktur seperti berikut ini :

- kd_siswa : (int) Primary Key, Auto Increment
- nama : Varchar(255)
- jenis_kelamin : Enum('pria', 'wanita')
- tempat_lahir : Varchar(255)
- tanggal_lahir : date
- no_telp : Varchar(100)
- alamat : Varchar(100)



setelah itu silahkan klik save.

Nah dalam step ini kita sudah membuat database, beserta tabelnya, nah karena untuk bagian awal nanti kita akan belajar menampilkan data terlebih dahulu sehingga kita perlu menambahkan datanya terlebih dahulu, dalam contoh ini kita akan menambahkan 2 data.

Untuk menambahkan data silahkan klik menu Insert

Server: 127.0.0.1 » Database: crud » Table: tb_siswa

Browse Structure SQL Search Insert Export Import Privilege

Column	Type	Function	Null	Value
kd_siswa	int(11)			
nama	varchar(255)			Anton Nugraha
jenis_kelamin	enum	--		pria
tempat_lahir	varchar(255)			Sidoarjo
tanggal_lahir	date			1996-04-04
no_telp	varchar(100)			0813324354
alamat	varchar(255)			Jl. Gelatik No. 76, Surabaya

Go

Server: 127.0.0.1 » Database: crud » Table: tb_siswa

Browse Structure SQL Search Insert Export Import Privilege

Column	Type	Function	Null	Value
kd_siswa	int(11)			
nama	varchar(255)			Budi Santoso
jenis_kelamin	enum	--		pria
tempat_lahir	varchar(255)			Surabaya
tanggal_lahir	date			1996-01-01
no_telp	varchar(100)			08568976545
alamat	varchar(255)			Jl. Bima Sakti 90 Sidoarjo

Go

Lalu silahkan klik tombol Go, sehingga kita memilih 2 data siswa.

Untuk melihat data anda bisa klik menu browse

Showing rows 0 - 1 (2 total, Query took 0.0020 seconds.)

```
SELECT * FROM `tb_siswa`
```

Number of rows: 25 | Filter rows: Search this table | Sort by key: None

	kd_siswa	nama	jenis_kelamin	tempat_lahir	tanggal_lahir	no_telp	alamat
<input type="checkbox"/>	1	Anton Nugraha	pria	Sidoarjo	1996-04-04	0813324354	Jl. Gelatik No. 76, Surabaya
<input type="checkbox"/>	2	Budi Santoso	pria	Surabaya	1996-01-01	08568976545	Jl. Bima Sakti 90 Sidoarjo

Selain cara diatas anda bisa membuat database, tabel serta mengisi data siswa menggunakan query SQL dibawah ini :

```
-- phpMyAdmin SQL Dump
-- version 4.9.0.1
-- https://www.phpmyadmin.net/
--
-- Host: 127.0.0.1
-- Generation Time: Apr 22, 2020 at 02:49 PM
-- Server version: 10.4.6-MariaDB
-- PHP Version: 7.3.9

SET SQL_MODE = "NO_AUTO_VALUE_ON_ZERO";
SET AUTOCOMMIT = 0;
START TRANSACTION;
SET time_zone = "+00:00";

/*!40101 SET @OLD_CHARACTER_SET_CLIENT=@@CHARACTER_SET_CLIENT */;
/*!40101 SET @OLD_CHARACTER_SET_RESULTS=@@CHARACTER_SET_RESULTS */;
/*!40101 SET @OLD_COLLATION_CONNECTION=@@COLLATION_CONNECTION */;
/*!40101 SET NAMES utf8mb4 */;

--
-- Database: `crud`
--
CREATE DATABASE IF NOT EXISTS `crud` DEFAULT CHARACTER SET latin1 COLLATE latin1_swedish_ci;
USE `crud`;

--
-- Table structure for table `tb_siswa`
--
CREATE TABLE `tb_siswa` (
  `kd_siswa` int(11) NOT NULL,
  `nama` varchar(255) NOT NULL,
  `jenis_kelamin` enum('pria','wanita') NOT NULL,
  `tempat_lahir` varchar(255) NOT NULL,
  `tanggal_lahir` date NOT NULL,
```

```

`no_telp` varchar(100) NOT NULL,
`alamat` varchar(255) NOT NULL
) ENGINE=InnoDB DEFAULT CHARSET=latin1;

--
-- Dumping data for table `tb_siswa`
--

INSERT INTO `tb_siswa` (`kd_siswa`, `nama`, `jenis_kelamin`, `tempat_lahir`,
`tanggal_lahir`, `no_telp`, `alamat`) VALUES
(1, 'Anton Nugraha', 'pria', 'Sidoarjo', '1996-04-04', '0813324354', 'Jl. Gelatik No. 76,
Surabaya'),
(2, 'Budi Santoso', 'pria', 'Surabaya', '1996-01-01', '08568976545', 'Jl. Bima Sakti 90
Sidoarjo'),
(3, 'cika', 'wanita', 'sidoarjo', '2020-05-01', '1231231', 'sedati');

--
-- Indexes for dumped tables
--

--
-- Indexes for table `tb_siswa`
--
ALTER TABLE `tb_siswa`
  ADD PRIMARY KEY (`kd_siswa`);

--
-- AUTO_INCREMENT for dumped tables
--

--
-- AUTO_INCREMENT for table `tb_siswa`
--
ALTER TABLE `tb_siswa`
  MODIFY `kd_siswa` int(11) NOT NULL AUTO_INCREMENT, AUTO_INCREMENT=4;
COMMIT;

/*!40101 SET CHARACTER_SET_CLIENT=@OLD_CHARACTER_SET_CLIENT */;
/*!40101 SET CHARACTER_SET_RESULTS=@OLD_CHARACTER_SET_RESULTS */;
/*!40101 SET COLLATION_CONNECTION=@OLD_COLLATION_CONNECTION */;

```

Baik asumsi saya anda sudah selesai mempersiapkan databasenya, berikutnya kita mempersiapkan project codeigniter, serta melakukan settingan beberapa bagian.

8.1.1. Mempersiapkan Project Codeigniter

Berikutnya silahkan teman – teman download dan install codeigniter 3, dalam contoh ini nama folder projectnya adalah crud, untuk tutorial langkah – langkah instalasi codeigniter anda bisa melihat tutorial kita sebelumnya Tutorial Cara Menginstall Codeigniter

Setting Konfigurasi Database

Berikutnya kita perlu setting database terlebih dahulu, silahkan buka file application/config/database.php, lalu berikutnya silahkan setting bagian database dengan settingan kurang lebih seperti berikut ini :

```
$db['default'] = array(  
    'dsn' => "",  
    'hostname' => 'localhost',  
    'username' => 'root',  
    'password' => "",  
    'database' => 'crud',  
    'dbdriver' => 'mysqli',  
    'dbprefix' => "",  
    'pconnect' => FALSE,  
    'db_debug' => (ENVIRONMENT !== 'production'),  
    'cache_on' => FALSE,  
    'cachedir' => "",  
    'char_set' => 'utf8',  
    'dbcollat' => 'utf8_general_ci',  
    'swap_pre' => "",  
    'encrypt' => FALSE,  
    'compress' => FALSE,  
    'stricton' => FALSE,  
    'failover' => array(),  
    'save_queries' => TRUE  
);
```

yang perlu disetting antara lain :

- hostname : localhost
- username : root (menyesuaikan username dari databasenya)
- password : " biarkan kosong menyesuaikan password databasenya
- database : 'crud' (merupakan nama dari databasenya)
- dbdriver : 'mysqli'

8.1.2. Setting File Config

Berikutnya silahkan buka file application/config/config.php

Ubah bagian

```
$config['base_url'] = '';
```

Menjadi

```
$config['base_url'] = 'http://localhost/crud/';
```

Ubah bagian

```
$config['index_page'] = 'index.php';
```

Menjadi

```
$config['index_page'] = '';
```

hal ini kita seting karena kita akan menghilangkan index.php

Buat file .htaccess dibagian root folder dan isi codenya adalah seperti berikut ini :

```
RewriteEngine On  
RewriteCond %{REQUEST_FILENAME} !-f  
RewriteCond %{REQUEST_FILENAME} !-d  
RewriteRule ^(.*)$ index.php/$1 [L]
```

untuk tutorial lebih lengkapnya bisa dilihat di tutorial kita sebelumnya : cara menghilangkan index.php di codeigniter

8.1.3. Load library dan Helper

Berikutnya kita akan load beberapa library , dan helper, silahkan buka file dengan nama application/config/autoload.php

ubah bagian

```
$autoload['libraries'] = array('');
```

menjadi

```
$autoload['libraries'] = array('database','session');
```

ubah bagian

```
$autoload['helper'] = array('');
```

Menjadi

```
$autoload['helper'] = array('url');
```

kita load helper url, karena kita akan menggunakan perintah base_url nantinya

8.1.4. Mempersiapkan library CSS & JS untuk tampilan aplikasi

agar tampilan aplikasinya lebih menarik kita akan buat tampilannya dengan bootstrap, serta kita juga akan menggunakan datepicker untuk memilih tanggal lahir siswa.

Beberapa library CSS yang perlu anda download linknya ada dibawah ini :

- <https://maxcdn.bootstrapcdn.com/bootstrap/4.0.0/css/bootstrap.min.css>
- <https://unpkg.com/bootstrap-datepicker@1.9.0/dist/css/bootstrap-datepicker3.min.css>

Beberapa library JS yang perlu anda download linknya ada dibawah ini :

- <https://stackpath.bootstrapcdn.com/bootstrap/4.4.1/js/bootstrap.min.js>
- <https://code.jquery.com/jquery-3.5.0.min.js>
- <https://unpkg.com/bootstrap-datepicker@1.9.0/dist/js/bootstrap-datepicker.min.js>

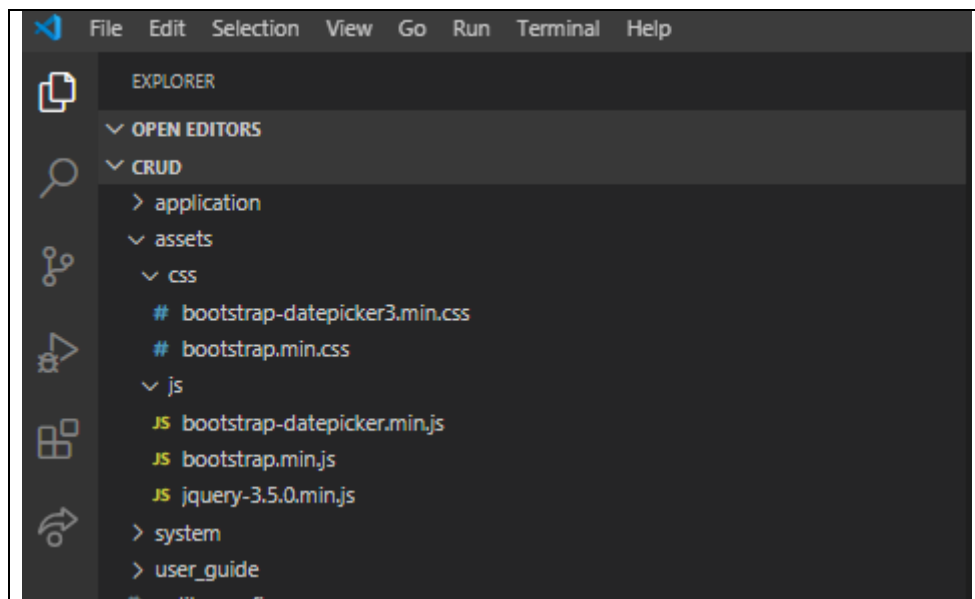
Berikutnya buat folder dengan nama assets, didalam folder assets kita buat 2 folder yaitu

1. css
2. js

hasil download library css kita letakkan di folder assets/css

hasil download library js kita letakkan di folder assets/js

jadi kurang lebih seperti berikut ini :



8.1.5. Membuat Bagian Template Header & Footer

Berikutnya kita akan buat file untuk keperluan template, bagian template ini kita akan bagi menjadi 3 bagian :

1. header
2. content
3. footer

8.1.6. Buat file header.php

kita buat file header.php didalam folder application/views/template/header.php

```

<!doctype html>
<html lang="en">
  <head>
    <title>Starter Template for Bootstrap</title>
    <link href="<?php echo base_url(); ?>assets/css/bootstrap.min.css"
rel="stylesheet">
    <link href="<?php echo base_url(); ?>assets/css/bootstrap-
datepicker3.min.css" rel="stylesheet">
    <script src="<?php echo base_url(); ?>assets/js/jquery-
3.5.0.min.js"></script>
    <script src="<?php echo base_url();
?>assets/js/bootstrap.min.js"></script>
    <script src="<?php echo base_url(); ?>assets/js/bootstrap-
datepicker.min.js"></script>

  </head>
  <body>

```

kita tuliskan kode untuk mengakses library css dan js yang sebelumnya telah kita download :

- line 5 – 6 load library css
- line 7 – 9 load library js

8.1.7. Buat File Footer.php

berikutnya kita buat file dengan nama footer.php didalam folder application/views/template/footer.php, untuk isi codenya adalah sebagai berikut :

```

<footer class="footer">
  <div class="container" style="text-align:center;">
    <hr/>
    <span>Warung Belajar @<?php echo date('Y'); ?></span>
  </div>
</footer>
</body>
</html>

```

baik kebutuhan untuk template sudah selesai kita akan mulai membuat CRUD, kita akan mulai dengan dengan membuat halaman untuk read data.

8.1.8. Membuat Fitur Read Data

Untuk langkah awal kita akan membuat fitur untuk menampilkan data siswa, kita perlu membuat controller terlebih dahulu, dalam contoh ini kita buat dengan nama Siswa.php

posisi file controller berada pada application/controller/Siswa.php

```
<?php
defined('BASEPATH') OR exit('No direct script access allowed');

class Siswa extends CI_Controller {

    public function __construct()
    {
        parent::__construct();
        $this->load->model("siswa_model");
        $this->load->library('form_validation');
    }

    public function index()
    {
        $data['siswa'] = $this->siswa_model->getAll();
        $this->load->view('template/header');
        $this->load->view('siswa/index',$data);
        $this->load->view('template/footer');
    }
}
```

Berikutnya kita buat file model dengan nama Siswa_model.php, didalam folder application/models/Siswa_model.php

```
<?php defined('BASEPATH') OR exit('No direct script access allowed');

class Siswa_model extends CI_Model
{
    private $table = "tb_siswa";

    public function getAll()
    {
        return $this->db->get($this->table)->result();
    }
}
```

Berikutnya kita buat file view dengan nama index.php, posisi file view berada pada application/siswa/index.php, isinya adalah seperti berikut ini :

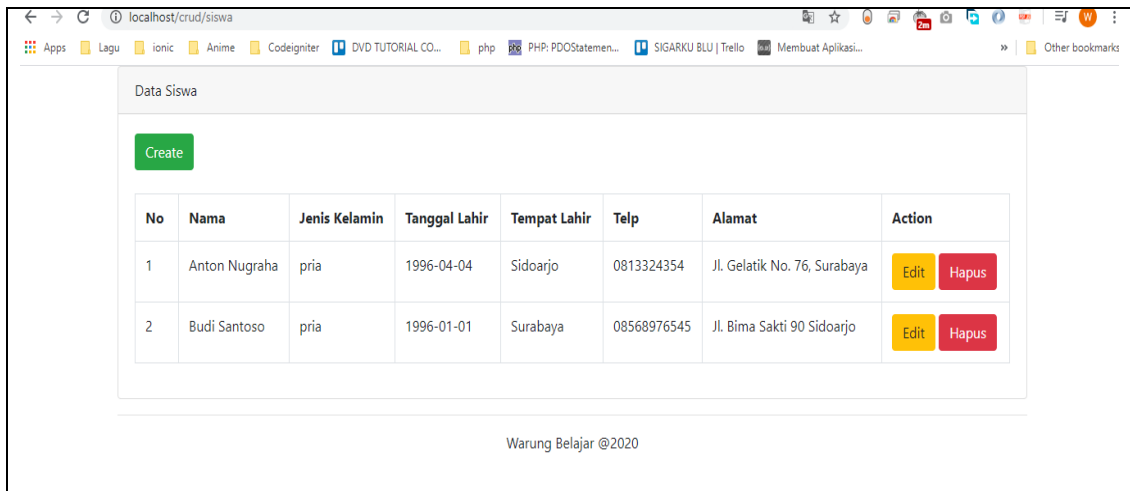
```
<main role="main" class="container">
    <div class="card">
        <div class="card-header">Data Siswa</div>
```

```

        <div class="card-body">
            <a href="<?php echo base_url(); ?>siswa/create"
class="btn btn-success">Create</a>
            <br/>
            <br/>
            <table class="table table-bordered">
                <tr>
                    <th width="5%">No</th>
                    <th>Nama</th>
                    <th>Jenis Kelamin</th>
                    <th>Tanggal Lahir</th>
                    <th>Tempat Lahir</th>
                    <th>Telp</th>
                    <th>Alamat</th>
                    <th>Action</th>
                </tr>
                <?php
                $no = 1;
                foreach($siswa as $row)
                {
                    <?>
                    <tr>
                        <td width="5%"><?php echo
                $no++; ?></td>
                        <td><?php echo $row->nama;
                ?></td>
                        <td><?php echo $row-
                >jenis_kelamin; ?></td>
                        <td><?php echo $row-
                >tanggal_lahir; ?></td>
                        <td><?php echo $row-
                >tempat_lahir; ?></td>
                        <td><?php echo $row->no_telp;
                ?></td>
                        <td><?php echo $row->alamat;
                ?></td>
                        <td>
                            <a href="<?php echo base_url();
                ?>siswa/edit/<?php echo $row->kd_siswa; ?>" class="btn btn-warning">Edit</a>
                            <a href="<?php echo base_url();
                ?>siswa/delete/<?php echo $row->kd_siswa; ?>" class="btn btn-danger">Hapus</a>
                        </td>
                    </tr>
                <?php
                }
                <?>
            </table>
        </div>
    </div>
</main>

```

Nah berikutnya kita akan coba mengakses, dengan alamat : localhost/crud/siswa



Penjelasan :

Kita Mulai dari file Controller Siswa.php, dibagian function `__construct` kita load model dengan nama `siswa_model.php` agar data berkomunikasi dengan tabel siswa di database, selain itu kita load library `form_validation` untuk keperluan validasi form.

```
public function __construct()
{
    parent::__construct();
    $this->load->model("siswa_model");
    $this->load->library('form_validation');
}
```

berikutnya pada method index kita menuliskan beberapa perintah antara lain :

```
$data['siswa'] = $this->siswa_model->getAll();
```

digunakan untuk mengambil data siswa, kita mengakses method `getAll` di model `siswa_model`, data tersebut kita simpan di array `$data` dengan elemen `siswa`

```
public function getAll()
{
    return $this->db->get($this->table)->result();
}
```

ini merupakan method `getAll` di model `siswa_model.php`, dimana kita menuliskan perintah untuk mengambil data di table `siswa`, disitu kita menuliskan `$this->table` yang berisi nilai `'tb_siswa'` (perhatikan line 5 di `siswa_model.php`)

```
$this->load->view('template/header');
$this->load->view('siswa/index',$data);
$this->load->view('template/footer');
```

ini adalah perintah di controller `siswa.php`, untuk menampilkan view, kita mengakses 3 view sekaligus, kita menatanya sesuai template yaitu :

- header (mengakses file view template/header.php)
- content (mengakses file view siswa/index.php) kita juga passing data berupa array \$data yang berisi data siswa hasil query dari method getAll di model siswa_model.php
- footer (mengakses file view template/footer.php)

Berikutnya kita beralih ke file view siswa/index.php

perhatikan beberapa kode dibawah ini :

Kita buat tombol untuk menambah data, dimana jika tombol ini di klik akan mengakses alamat <http://localhost/crud/siswa/create>, perhatikan kode line 5

```
<a href="<?php echo base_url(); ?>siswa/create" class="btn btn-success">Create</a>
```

kita lakukan extract variabel \$siswa yang merupakan array yang dipassing dari controller, kita gunakan perintah foreach (Line 21)

```
foreach($siswa as $row)
{
?>
    <tr>
    <td width="5%"><?php echo $no++; ?></td>
    <td><?php echo $row->nama; ?></td>
    <td><?php echo $row->jenis_kelamin; ?></td>
    <td><?php echo $row->tanggal_lahir; ?></td>
    <td><?php echo $row->tempat_lahir; ?></td>
    <td><?php echo $row->no_telp; ?></td>
    <td><?php echo $row->alamat; ?></td>
    <td>
    <a href="<?php echo base_url(); ?>siswa/edit/<?php echo $row->kd_siswa; ?>" class="btn btn-warning">Edit</a>
    <a href="<?php echo base_url(); ?>siswa/delete/<?php echo $row->kd_siswa; ?>" class="btn btn-danger">Hapus</a>
    </td>
    </tr>
    <?php
}
```

kita menampilkan setiap data siswanya dengan perpaduan perintah html <tr></td> dan <td></td>

untuk menampilkan data kita bisa menuliskan perintah \$row->nama_kolom, semisal menampilkan nilai dari kolom alamat maka menggunakan perintah \$row->alamat

bagian tombol edit akan mengakses alamat : http://localhost/crud/siswa/edit/kd_siswa, karena kd_siswa ada dalam kolom kd_siswa sehingga kita menggunakan perintah \$row->kd_siswa

bagian tombol hapus akan mengakses alamat : http://localhost/crud/siswa/delete/kd_siswa, karena kd_siswa ada dalam kolom kd_siswa sehingga kita menggunakan perintah \$row->kd_siswa

8.1.9. Membuat Form untuk Menambahkan Data

Kembali ke controller Siswa.php, lalu tambahkan function create, sehingga kode di controller Siswa.php menjadi seperti berikut ini :

```
<?php
defined('BASEPATH') OR exit('No direct script access allowed');

class Siswa extends CI_Controller {

    public function __construct()
    {
        parent::__construct();
        $this->load->model("siswa_model");
        $this->load->library('form_validation');
    }

    public function index()
    {
        $data['siswa'] = $this->siswa_model->getAll();
        $this->load->view('template/header');
        $this->load->view('siswa/index',$data);
        $this->load->view('template/footer');
    }

    public function create()
    {
        $this->load->view('template/header');
        $this->load->view('siswa/create');
        $this->load->view('template/footer');
    }
}
```

Keterangan :

Perintah dibagian method create akan menampilkan file view dengan nama create.php di dalam folder application/views/siswa/create.php (Line 24)

tidak hanya itu kita juga mengakses file view header.php (Line 23), dan view footer.php (Line 24) untuk keperluan template.

Berikutnya kita buat file view dengan nama create.php, posisi file view berada pada folder application/create.php, isi kodenya adalah sebagai berikut :

```

<div class="container">
  <div class="card">
    <div class="card-header">Create Siswa</div>
    <div class="card-body">
      <?php
      if(validation_errors() != false)
      {
        ?>
        <div class="alert alert-danger" role="alert">
          <?php echo validation_errors(); ?>
        </div>
      <?php
      }
      ?>
      <form method="post" action="<?php echo base_url();
?>siswa/save">
        <div class="form-group">
          <label for="nama">Nama</label>
          <input type="text" class="form-control"
id="nama" name="nama">
        </div>

        <div class="form-group">
          <label for="jenis_kelamin">Jenis
Kelamin</label>
          <select name="jenis_kelamin"
id="jenis_kelamin" class="form-control">
            <option value="pria">Pria</option>
            <option
value="wanita">Wanita</option>
          </select>
        </div>

        <div class="form-group">
          <label for="tempat_lahir">Tempat
Lahir</label>
          <input type="text" class="form-control"
id="tempat_lahir" name="tempat_lahir">
        </div>

        <div class="form-group">
          <label for="tgl_lahir">Tanggal Lahir</label>
          <input type="text" class="form-control
datepicker" id="tanggal_lahir" name="tanggal_lahir">
        </div>

        <div class="form-group">
          <label for="no_telp">No Telp</label>
          <input type="number" class="form-control"

```



```

id="no_telp" name="no_telp">
    </div>

    <div class="form-group">
        <label for="alamat">Alamat</label>
        <textarea class="form-control"
name="alamat" id="alamat"></textarea>
    </div>

    <button type="submit" class="btn btn-
primary">Submit</button>
    </form>
    </div>
</div>

```

Untuk menampilkan form tambah data, anda bisa menggunakan alamat : <http://localhost/crud/siswa/create>

Keterangan :

Pada Line 6 kita menuliskan perintah untuk melakukan pengecekan apakah nantinya entrian form sudah valid sesuai form validation, jika tidak valid maka akan menampilkan pesan error berisi informasi entrian form yang tidak valid

```

<?php
    if(validation_errors() != false)
    {
        ?>
        <div class="alert alert-danger" role="alert">
            <?php echo validation_errors(); ?>
        </div>
    }
<?php

```

```
}  
?>
```

Line 15 untuk bagian form kita menggunakan method post, dan entrian form akan di proses di method save pada controller Siswa

```
<form method="post" action="<?php echo base_url();  
?>siswa/save">
```

Line 18 kita buat inputan form dengan name="nama"

```
<input type="text" class="form-control" id="nama"  
name="nama">
```

Line 23 – 26 kita buat pilihan dengan menggunakan selectbox dengan name="jenis_kelamin"

```
<select name="jenis_kelamin" id="jenis_kelamin" class="form-control">  
  <option value="pria">Pria</option>  
  <option value="wanita">Wanita</option>  
</select>
```

Line 31 kita buat inputan dengan name = "tempat_lahir"

```
<input type="text" class="form-control"  
id="tempat_lahir" name="tempat_lahir">
```

Line 36 kita buat inputan dengan name="tanggal_lahir", disini kita juga tambahkan class datepicker, karena kita akan menggunakan datepicker untuk memilih tanggalnya.

```
<input type="text" class="form-control datepicker"  
id="tanggal_lahir" name="tanggal_lahir">
```

Line 41 kita buat inputan dengan name ="no_telp"

```
<input type="number" class="form-control" id="no_telp"  
name="no_telp">
```

Line 46 kita buat textarea dengan name="alamat"

```
<textarea class="form-control" name="alamat"  
id="alamat"></textarea>
```

Line 49 kita buat tombol submit untuk submit form

```
<button type="submit" class="btn btn-  
primary">Submit</button>
```

8.1.10. Membuat Method untuk memproses menambahkan data

Untuk melakukan proses terhadap inputan form tambah data, kita perlu membuat method dengan nama save di controller Siswa.php, sehingga controller Siswa isinya menjadi seperti berikut :

```
<?php
defined('BASEPATH') OR exit('No direct script access allowed');

class Siswa extends CI_Controller {

    public function __construct()
    {
        parent::__construct();
        $this->load->model("siswa_model");
        $this->load->library('form_validation');
    }

    public function index()
    {
        $data['siswa'] = $this->siswa_model->getAll();
        $this->load->view('template/header');
        $this->load->view('siswa/index',$data);
        $this->load->view('template/footer');
    }

    public function create()
    {
        $this->load->view('template/header');
        $this->load->view('siswa/create');
        $this->load->view('template/footer');
    }

    public function save()
    {
        $this->form_validation->set_rules('nama','Nama','required');
        $this->form_validation->set_rules('jenis_kelamin','Jenis
Kelamin','required');
        $this->form_validation->set_rules('tempat_lahir','Tempat
Lahir','required');
        $this->form_validation->set_rules('tanggal_lahir','Tanggal
Lahir','required');
        $this->form_validation->set_rules('no_telp','Nomor
Telepon','required');
        $this->form_validation->set_rules('alamat','Alamat','required');
        if ($this->form_validation->run()==true)
        {
            $data['nama'] = $this->input->post('nama');
            $data['jenis_kelamin'] = $this->input->post('jenis_kelamin');
            $data['tempat_lahir'] = $this->input->post('tempat_lahir');
            $data['tanggal_lahir'] = $this->input->post('tanggal_lahir');
            $data['no_telp'] = $this->input->post('no_telp');
            $data['alamat'] = $this->input->post('alamat');
```

```

        $this->siswa_model->save($data);
        redirect('siswa');
    }
    else
    {
        $this->load->view('template/header');
        $this->load->view('siswa/create');
        $this->load->view('template/footer');
    }
}
}
}

```

Keterangan :

Pada line 28 kita buat function save untuk memproses hasil inputan dari form tambah data.

Kita buat validasi untuk inputan form, dimana inputan dengan name (nama,jenis_kelamin,tempat_lahir,tanggal_lahir, no_telp, alamat) bersifat required yang artinya harus diisi inputan formnya.

```

$this->form_validation->set_rules('nama','Nama','required');
$this->form_validation->set_rules('jenis_kelamin','Jenis Kelamin','required');
$this->form_validation->set_rules('tempat_lahir','Tempat Lahir','required');
$this->form_validation->set_rules('tanggal_lahir','Tanggal Lahir','required');
$this->form_validation->set_rules('no_telp','Nomor Telepon','required');
$this->form_validation->set_rules('alamat','Alamat','required');
if ($this->form_validation->run()==true)

```

untuk tutorial lebih lanjut mengenai form validasi bisa lihat tutorial kita sebelumnya : Penanganan Form dan Form validasi di Codeigniter

Jika validasi form bernilai true atau bisa dikatakan inputan formnya valid seluruhnya sesuai form validasi, maka akan menjalankan code dibaris 38 – 45.

line 38 – 43 kita menangkap inputan form dan disimpan dalam array \$data, kita menangkap inputan form dengan menggunakan method post sehingga perintahnya adalah \$this->input->post('nama_inputan')

```

$data['nama'] = $this->input->post('nama');
$data['jenis_kelamin'] = $this->input->post('jenis_kelamin');
$data['tempat_lahir'] = $this->input->post('tempat_lahir');
$data['tanggal_lahir'] = $this->input->post('tanggal_lahir');
$data['no_telp'] = $this->input->post('no_telp');
$data['alamat'] = $this->input->post('alamat');

```

Line 44 kita mengakses method save di model Siswa_model, dengan mengirimkan parameter variabel \$data yang telah berisi inputan form.

```

$this->siswa_model->save($data);

```

Line 45 kita lakukan redirect ke dalam alamat : <http://localhost/crud/siswa>

```
redirect('siswa');
```

line 49 – 51 akan dijalankan jika inputan form ada yang tidak valid sesuai form validasi, maka akan menampilkan view dengan nama create didalam folder application/views/siswa/create.php atau secara sederhana kembali menampilkan form inputan.

```
$this->load->view('template/header');  
$this->load->view('siswa/create');  
$this->load->view('template/footer');
```

Menambahkan method save di model Siswa_model.php

silahkan buka file Siswa_model.php dan tambahkan method save, sehingga kode file Siswa_model.php menjadi berikut ini :

```
<?php defined('BASEPATH') OR exit('No direct script access allowed');  
  
class Siswa_model extends CI_Model  
{  
    private $table = "tb_siswa";  
  
    public function getAll()  
    {  
        return $this->db->get($this->table)->result();  
    }  
  
    public function save($data)  
    {  
        return $this->db->insert($this->table, $data);  
    }  
}
```

Keterangan :

pada line 12 kita membuat method save, dengan memiliki satu parameter yaitu variabel \$data, parameter ini adalah variabel array yang berisi inputan form yang dikirimkan dari controller Siswa.php pada method save

disini kita akan insert data ke dalam tabel tb_siswa, perintahnya adalah :

```
return $this->db->insert($this->table, $data);
```

jadi perintah insert disini memiliki 2 parameter yaitu :

nama tabel yang akan kita insert data, maka dalam contoh ini kita tuliskan \$this->table karena bagian class variabel \$table sudah kita berikan nilai “tb_siswa”

parameter yang kedua adalah data yang akan diinsert di tabel yang ditulis di parameter pertama, dalam contoh ini adalah variabel \$data yang berisi array hasil inputan form

8.1.11. Membuat Form Edit Data

Berikutnya kita akan melanjutkan ke proses edit data, bagian ini kita akan bagian menjadi 2 yaitu, pembuatan form editnya, dan proses updatenya.

kita akan mulai dibagian form editnya terlebih dahulu, perhatikan dibagian file view application/views/siswa/index.php

perhatikan pada line 33 kita membuat tombol edit :

```
<a href="<?php echo base_url(); ?>siswa/edit/<?php echo $row->kd_siswa; ?>"  
class="btn btn-warning">Edit</a>
```

Keterangan :

tombol edit ini jika di klik akan mengakses alamat : localhost/crud/siswa/edit/kd_siswa

sehingga bisa disimpulkan untuk menampilkan form edit kita perlu mengakses method edit didalam controller siswa, dan juga menyertakan nilai kolom kd_siswa dari siswa yang akan diedit.

Nah untuk langkah awal kita akan tambahkan method edit didalam controller siswa, silahkan buka file controller application/controller/Siswa.php, lalu tambahkan bagian method edit, sehingga hasilnya akan seperti berikut ini :

```
<?php  
defined('BASEPATH') OR exit('No direct script access allowed');  
  
class Siswa extends CI_Controller {  
  
    public function __construct()  
    {  
        parent::__construct();  
        $this->load->model("siswa_model");  
        $this->load->library('form_validation');  
    }  
  
    public function index()  
    {  
        $data['siswa'] = $this->siswa_model->getAll();  
        $this->load->view('template/header');  
        $this->load->view('siswa/index',$data);  
        $this->load->view('template/footer');  
    }  
}
```

```

public function create()
{
    $this->load->view('template/header');
    $this->load->view('siswa/create');
    $this->load->view('template/footer');
}

public function save()
{
    $this->form_validation->set_rules('nama','Nama','required');
    $this->form_validation->set_rules('jenis_kelamin','Jenis
Kelamin','required');
    $this->form_validation->set_rules('tempat_lahir','Tempat
Lahir','required');
    $this->form_validation->set_rules('tanggal_lahir','Tanggal
Lahir','required');
    $this->form_validation->set_rules('no_telp','Nomor
Telepon','required');
    $this->form_validation->set_rules('alamat','Alamat','required');
    if ($this->form_validation->run()==true)
    {
        $data['nama'] = $this->input->post('nama');
        $data['jenis_kelamin'] = $this->input-
>post('jenis_kelamin');
        $data['tempat_lahir'] = $this->input->post('tempat_lahir');
        $data['tanggal_lahir'] = $this->input->post('tanggal_lahir');
        $data['no_telp'] = $this->input->post('no_telp');
        $data['alamat'] = $this->input->post('alamat');
        $this->siswa_model->save($data);
        redirect('siswa');
    }
    else
    {
        $this->load->view('template/header');
        $this->load->view('siswa/create');
        $this->load->view('template/footer');
    }
}

function edit($kd_siswa)
{
    $data['siswa'] = $this->siswa_model->getById($kd_siswa);
    $this->load->view('template/header');
    $this->load->view('siswa/edit',$data);
    $this->load->view('template/footer');
}
}

```

Berikutnya buka file model Siswa_model.php dalam folder application/models/Siswa_model.php, lalu tambahkan method getById, sehingga hasil dari Siswa_model.php adalah sebagai berikut :

```
<?php defined('BASEPATH') OR exit('No direct script access allowed');

class Siswa_model extends CI_Model
{
    private $table = "tb_siswa";

    public function getAll()
    {
        return $this->db->get($this->table)->result();
    }

    public function save($data)
    {
        return $this->db->insert($this->table, $data);
    }

    public function getById($id)
    {
        return $this->db->get_where($this->table, ["kd_siswa" => $id])->row();
    }
}
```

Keterangan :

Kita menambahkan method edit di controller Siswa di line 55 – 61, pada method edit ini kita menambahkan 1 parameter yaitu \$kd_siswa

Pada line 57 di controller siswa kita mengakses method getById didalam siswa_model, dengan menyertakan parameter yang dibutuhkan yaitu \$kd_siswa, dimana nanti hasilnya akan kita simpan pada array \$data dengan nama key/elemen yaitu siswa

```
$data['siswa'] = $this->siswa_model->getById($kd_siswa);
```

Pada line 17 di model Siswa_model kita buat method dengan nama getById dan memiliki parameter \$id, dalam method ini kita buat perintah query untuk mencari data siswa berdasarkan kd_siswa yang sama dengan nilai dari variabel \$id.

```
public function getById($id)
{
    return $this->db->get_where($this->table, ["kd_siswa" => $id])->row();
}
```

kita kembali ke bagian controller siswa.php, pada method edit setelah kita mengambil data siswa berdasarkan kd_siswa, berikutnya kita akan load view header, edit, dan footer.

kita fokus load view edit.php pada line 59, kita juga passing data yang berupa variabel array \$data yang berisi data siswa sesuai kd_siswa yang diinginkan.

```
$this->load->view('template/header');
$this->load->view('siswa/edit',$data);
$this->load->view('template/footer');
```

berikutnya kita buat file view dengan nama edit.php, kita buat di folder application/views/siswa/edit.php, untuk isi kodenya seperti berikut :

```
<div class="container">
    <div class="card">
        <div class="card-header">Edit Siswa</div>
        <div class="card-body">
            <?php
            if(validation_errors() != false)
            {
                ?>
                <div class="alert alert-danger" role="alert">
                    <?php echo validation_errors(); ?>
                </div>
                <?php
            }
            ?>
            <form method="post" action="<?php echo base_url();
?>siswa/update">
                <input type="hidden" name="kd_siswa" id="kd_siswa"
value="<?php echo $siswa->kd_siswa; ?>"/>
                <div class="form-group">
                    <label for="nama">Nama</label>
                    <input type="text" value="<?php echo $siswa-
>nama; ?>" class="form-control" id="nama" name="nama">
                </div>

                <div class="form-group">
                    <label for="jenis_kelamin">Jenis
Kelamin</label>
                    <select name="jenis_kelamin"
id="jenis_kelamin" class="form-control">
                        <option value="pria" <?php echo
($siswa->jenis_kelamin ? 'pria' : 'selected') ; ?> >Pria</option>
                        <option value="wanita" <?php echo
($siswa->jenis_kelamin ? 'wanita' : 'selected') ; ?> >Wanita</option>
                    </select>
                </div>

                <div class="form-group">
                    <label for="tempat_lahir">Tempat Lahir</label>
                    <input type="text" class="form-control"
id="tempat_lahir" name="tempat_lahir" value="<?php echo $siswa->tempat_lahir; ?>">
                </div>

                <div class="form-group">
```

```

        <label for="tgl_lahir">Tanggal Lahir</label>
        <input type="text" class="form-control
datepicker" readonly id="tanggal_lahir" name="tanggal_lahir" value="<?php echo
$siswa->tanggal_lahir; ?>">
    </div>

    <div class="form-group">
        <label for="no_telp">No Telp</label>
        <input type="number" class="form-control"
value="<?php echo $siswa->no_telp; ?>" id="no_telp" name="no_telp">
    </div>

    <div class="form-group">
        <label for="alamat">Alamat</label>
        <textarea class="form-control" name="alamat"
id="alamat"><?php echo $siswa->alamat; ?></textarea>
    </div>

    <button type="submit" class="btn btn-
primary">Update</button>
    </form>
    </div>
</div>

```

Keterangan :

Line 5 – 14 kita tuliskan perintah untuk melakukan pengecekan terhadap error validasinya, nantinya jika ada error terhadap form validasi maka akan menampilkan error.

```

<?php
if(validation_errors() != false)
{
    ?>
    <div class="alert alert-danger" role="alert">
    <?php echo validation_errors(); ?>
    </div>
    <?php
}
?>

```

Line 15 kita buat form dengan perintah HTML, dengan method = “post” dan bagian action akan mengakses alamat localhost/crud/siswa/update untuk memproses form update.

```

<form method="post" action="<?php echo base_url(); ?>siswa/update">

```

jadi bisa kita simpulkan form update akan di proses dibagian method update di controller siswa.

Line 16 kita buat inputan dengan type hidden dengan name kd_siswa, dan value kita set \$siswa->kd_siswa; yang artinya mengambil nilai pada kolom kd_siswa dari variabel \$siswa, variabel \$siswa adalah nilai yang dipassing dari controller ke bagian view.

```
<input type="hidden" name="kd_siswa" id="kd_siswa" value="<?php echo $siswa->kd_siswa; ?>" />
```

Line 24 – 27 kita buat selectbox dengan nama jenis_kelamin, dan untuk option yang terpilih kita sesuai dengan nilai jenis_kelamin dari data siswa

```
<select name="jenis_kelamin" id="jenis_kelamin" class="form-control">
  <option value="pria" <?php echo ($siswa->jenis_kelamin ? 'pria' : 'selected' );
  ?> >Pria</option>
  <option value="wanita" <?php echo ($siswa->jenis_kelamin ? 'wanita' :
  'selected' ); ?> >Wanita</option>
</select>
```

Line 32 kita buat input type text dengan name tempat_lahir, yang bagian value kita isi dengan nilai kolom tempat_lahir di data siswa

```
<input type="text" class="form-control" id="tempat_lahir" name="tempat_lahir" value="<?php echo $siswa->tempat_lahir; ?>">
```

Line 37 kita buat input type text dengan name tanggal_lahir, yang bagian value kita isi dengan nilai kolom tanggal_lahir di data siswa

```
<input type="text" class="form-control datepicker" readonly id="tanggal_lahir" name="tanggal_lahir" value="<?php echo $siswa->tanggal_lahir; ?>">
```

Line 42 kita buat input type text dengan name no_telp, yang bagian value kita isi dengan nilai kolom no_telp di data siswa

```
<input type="number" class="form-control" value="<?php echo $siswa->no_telp; ?>" id="no_telp" name="no_telp">
```

Line 47 kita buat textarea dengan name alamat, yang bagian value kita isi dengan nilai kolom alamat di data siswa

```
<textarea class="form-control" name="alamat" id="alamat"><?php echo $siswa->alamat; ?></textarea>
```

Line 50 kita buat tombol update, yang nantinya akan ada tulisan Update.

```
<button type="submit" class="btn btn-primary">Update</button>
```

Kita contohkan kita coba edit salah satu data siswa, semisal kita edit data yang pertama, alamatnya seperti berikut ini : localhost/crud/siswa/edit/1

The screenshot shows a web browser window with the address bar displaying 'localhost/crud/siswa/edit/1'. The page content is a form titled 'Edit Siswa'. The form has the following fields and values:

- Nama: Anton Nugraha
- Jenis Kelamin: Pria
- Tempat Lahir: Sidoarjo
- Tanggal Lahir: 1996-04-04
- No Telp: 0813324354
- Alamat: Jl. Gelatik No. 76, Surabaya

At the bottom of the form is a blue 'Update' button. The footer of the page reads 'Warung Belajar @2020'.

8.1.12. Membuat Method Update untuk proses update data siswa

Berikutnya kita buat method update didalam controller siswa.php, sehingga kodenya menjadi seperti berikut ini :

```

<?php
defined('BASEPATH') OR exit('No direct script access allowed');

class Siswa extends CI_Controller {

    public function __construct()
    {
        parent::__construct();
        $this->load->model("siswa_model");
        $this->load->library('form_validation');
    }

    public function index()
    {
        $data['siswa'] = $this->siswa_model->getAll();
        $this->load->view('template/header');
        $this->load->view('siswa/index',$data);
        $this->load->view('template/footer');
    }

    public function create()
    {
        $this->load->view('template/header');
        $this->load->view('siswa/create');
        $this->load->view('template/footer');
    }

    public function save()

```

```

        {
            $this->form_validation->set_rules('nama','Nama','required');
            $this->form_validation->set_rules('jenis_kelamin','Jenis
Kelamin','required');
            $this->form_validation->set_rules('tempat_lahir','Tempat
Lahir','required');
            $this->form_validation->set_rules('tanggal_lahir','Tanggal
Lahir','required');
            $this->form_validation->set_rules('no_telp','Nomor
Telepon','required');
            $this->form_validation->set_rules('alamat','Alamat','required');
            if ($this->form_validation->run()==true)
            {
                $data['nama'] = $this->input->post('nama');
                $data['jenis_kelamin'] = $this->input->post('jenis_kelamin');
                $data['tempat_lahir'] = $this->input->post('tempat_lahir');
                $data['tanggal_lahir'] = $this->input->post('tanggal_lahir');
                $data['no_telp'] = $this->input->post('no_telp');
                $data['alamat'] = $this->input->post('alamat');
                $this->siswa_model->save($data);
                redirect('siswa');
            }
            else
            {
                $this->load->view('template/header');
                $this->load->view('siswa/create');
                $this->load->view('template/footer');
            }
        }

        function edit($kd_siswa)
        {
            $data['siswa'] = $this->siswa_model->getById($kd_siswa);
            $this->load->view('template/header');
            $this->load->view('siswa/edit',$data);
            $this->load->view('template/footer');
        }

        public function update()
        {
            $this->form_validation->set_rules('nama','Nama','required');
            $this->form_validation->set_rules('jenis_kelamin','Jenis
Kelamin','required');
            $this->form_validation->set_rules('tempat_lahir','Tempat
Lahir','required');
            $this->form_validation->set_rules('tanggal_lahir','Tanggal
Lahir','required');
            $this->form_validation->set_rules('no_telp','Nomor
Telepon','required');
            $this->form_validation->set_rules('alamat','Alamat','required');
            if ($this->form_validation->run()==true)
            {
                $kd_siswa = $this->input->post('kd_siswa');
                $data['nama'] = $this->input->post('nama');
            }
        }
    }
}

```

```

        $data['jenis_kelamin'] = $this->input->post('jenis_kelamin');
        $data['tempat_lahir'] = $this->input->post('tempat_lahir');
        $data['tanggal_lahir'] = $this->input->post('tanggal_lahir');
        $data['no_telp'] = $this->input->post('no_telp');
        $data['alamat'] = $this->input->post('alamat');
        $this->siswa_model->update($data,$kd_siswa);
        redirect('siswa');
    }
    else
    {
        $kd_siswa = $this->input->post('kd_siswa');
        $data['siswa'] = $this->siswa_model->getById($kd_siswa);
        $this->load->view('template/header');
        $this->load->view('siswa/edit',$data);
        $this->load->view('template/footer');
    }
}
}

```

Berikutnya kita akan menambahkan method update didalam file model Siswa_model.php, method update ini memiliki 2 parameter, yaitu variabel \$data dan variabel \$id, sehingga file model Siswa_model.php berisi kode seperti ini

```

<?php defined('BASEPATH') OR exit('No direct script access allowed');

class Siswa_model extends CI_Model
{
    private $table = "tb_siswa";

    public function getAll()
    {
        return $this->db->get($this->table)->result();
    }

    public function save($data)
    {
        return $this->db->insert($this->table, $data);
    }

    public function getById($id)
    {
        return $this->db->get_where($this->table, ["kd_siswa" => $id])->row();
    }

    public function update($data,$id)
    {
        return $this->db->update($this->table, $data, array('kd_siswa' => $id));
    }
}

```

Keterangan :

kita mulai dari file controller siswa.php, line 65 – 70 kita tuliskan perintah untuk melakukan validasi inputan form, jika inputan form valid sesuai validasi form maka akan menjalankan line 73 – 81, tetapi jika tidak valid akan menjalankan line 85 – 89

```
$this->form_validation->set_rules('nama','Nama','required');
$this->form_validation->set_rules('jenis_kelamin','Jenis Kelamin','required');
$this->form_validation->set_rules('tempat_lahir','Tempat Lahir','required');
$this->form_validation->set_rules('tanggal_lahir','Tanggal Lahir','required');
$this->form_validation->set_rules('no_telp','Nomor Telepon','required');
$this->form_validation->set_rules('alamat','Alamat','required');
```

Line 73 kita simpan inputan form kd_siswa kedalam variabel \$kd_siswa

```
$kd_siswa = $this->input->post('kd_siswa');
```

Line 74 – 79 kita simpan inputan form (nama, jenis_kelamin, tempat_lahir, tanggal_lahir, no_telp, alamat) inputan form ini akan disimpan kedalam variabel \$data yang merupakan array.

```
$data['nama'] = $this->input->post('nama');
$data['jenis_kelamin'] = $this->input->post('jenis_kelamin');
$data['tempat_lahir'] = $this->input->post('tempat_lahir');
$data['tanggal_lahir'] = $this->input->post('tanggal_lahir');
$data['no_telp'] = $this->input->post('no_telp');
$data['alamat'] = $this->input->post('alamat');
```

Pada line 80 kita mengakses method update, didalam model siswa_model, kita juga menyertakan 2 parameter yaitu variabel \$data, dan variabel \$kd_siswa, dimana variabel \$data berisi inputan dari form update, dan \$kd_siswa adalah nilai kd_siswa yang akan diupdate.

```
$this->siswa_model->update($data,$kd_siswa);
```

Berikutnya kita buat beralih ke file siswa_model.php, pada line 22 kita method update, dengan 2 parameter juga yaitu variabel \$data, dan variabel \$id.

Pada method update dibagian siswa_model ini kita tuliskan query untuk update data, untuk query update data kita perlu menuliskan 3 bagian penting, yaitu :

1. nama tabelnya dalam hal ini kita tuliskan \$this->table, yang memiliki nilai 'tb_siswa'
2. nilai yang diupdate, dalam hal ini kita tuliskan variabel \$data yang telah berisi hasil inputan form update dari bagian controller
3. kd_siswa dari data siswa yang akan diupdate, dalam ini ini tuliskan variabel \$id yang berisi kode siswa dari nilai siswa yang akan diupdate.

```
public function update($data,$id)
{
    return $this->db->update($this->table, $data, array('kd_siswa' => $id));
}
```

Jika proses update berhasil akan diarahkan kedalam controller siswa pada method index (Line 81)

```
redirect('siswa');
```

Kita lanjut ke pembahasan dibagian controller siswa, jika inputan form tidak valid, maka akan menjalankan line 85 – 89, dimana bagian ini line 85 akan mendapatkan data kd_siswa, line 86 akan mendapatkan data siswa berdasarkan kd_siswa, lalu akan diarahkan untuk mengakses view bagian form edit, dengan passing variabel \$data (Line 88)

```
$kd_siswa = $this->input->post('kd_siswa');  
$data['siswa'] = $this->siswa_model->getById($kd_siswa);  
$this->load->view('template/header');  
$this->load->view('siswa/edit',$data);  
$this->load->view('template/footer');
```

8.1.13. Membuat Fitur Delete data siswa

Berikutnya kita akan melanjutkan untuk membuat fitur delete data siswa, jika kita perhatikan kita telah membuat tombol delete dibagian file view index.

coba buka file view dengan nama index.php didalam folder application/views/siswa/index.php, perhatikan pada line 34.

```
<a href="<?php echo base_url(); ?>siswa/delete/<?php echo $row->kd_siswa;  
?" class="btn btn-danger">Hapus</a>
```

Keterangan :

tombol delete itu jika di klik maka akan mengakses localhost/crud/siswa/delete/kd_siswa.

jadi bisa disimpulkan proses delete data akan dilakukan di method delete pada controller siswa, dengan mengirimkan 1 parameter kd_siswa

berikutnya kita buka file controller siswa, lalu kita buat method delete, yang memiliki parameter \$kd_siswa, hasilnya controller siswa menjadi seperti ini :

```
<?php  
defined('BASEPATH') OR exit("No direct script access allowed");  
  
class Siswa extends CI_Controller {  
  
    public function __construct()  
    {  
        parent::__construct();  
        $this->load->model("siswa_model");  
        $this->load->library("form_validation");  
    }  
  
}
```



```

public function index()
{
    $data['siswa'] = $this->siswa_model->getAll();
    $this->load->view('template/header');
    $this->load->view('siswa/index',$data);
    $this->load->view('template/footer');
}

public function create()
{
    $this->load->view('template/header');
    $this->load->view('siswa/create');
    $this->load->view('template/footer');
}

public function save()
{
    $this->form_validation->set_rules('nama','Nama','required');
    $this->form_validation->set_rules('jenis_kelamin','Jenis
Kelamin','required');
    $this->form_validation->set_rules('tempat_lahir','Tempat
Lahir','required');
    $this->form_validation->set_rules('tanggal_lahir','Tanggal
Lahir','required');
    $this->form_validation->set_rules('no_telp','Nomor
Telepon','required');
    $this->form_validation->set_rules('alamat','Alamat','required');
    if ($this->form_validation->run()==true)
    {
        $data['nama'] = $this->input->post('nama');
        $data['jenis_kelamin'] = $this->input->post('jenis_kelamin');
        $data['tempat_lahir'] = $this->input->post('tempat_lahir');
        $data['tanggal_lahir'] = $this->input->post('tanggal_lahir');
        $data['no_telp'] = $this->input->post('no_telp');
        $data['alamat'] = $this->input->post('alamat');
        $this->siswa_model->save($data);
        redirect('siswa');
    }
    else
    {
        $this->load->view('template/header');
        $this->load->view('siswa/create');
        $this->load->view('template/footer');
    }
}

function edit($kd_siswa)
{
    $data['siswa'] = $this->siswa_model->getById($kd_siswa);
    $this->load->view('template/header');
    $this->load->view('siswa/edit',$data);
    $this->load->view('template/footer');
}

```

```

public function update()
{
    $this->form_validation->set_rules('nama','Nama','required');
    $this->form_validation->set_rules('jenis_kelamin','Jenis
Kelamin','required');
    $this->form_validation->set_rules('tempat_lahir','Tempat
Lahir','required');
    $this->form_validation->set_rules('tanggal_lahir','Tanggal
Lahir','required');
    $this->form_validation->set_rules('no_telp','Nomor
Telepon','required');
    $this->form_validation->set_rules('alamat','Alamat','required');
    if ($this->form_validation->run()==true)
    {
        $kd_siswa = $this->input->post('kd_siswa');
        $data['nama'] = $this->input->post('nama');
        $data['jenis_kelamin'] = $this->input->post('jenis_kelamin');
        $data['tempat_lahir'] = $this->input->post('tempat_lahir');
        $data['tanggal_lahir'] = $this->input->post('tanggal_lahir');
        $data['no_telp'] = $this->input->post('no_telp');
        $data['alamat'] = $this->input->post('alamat');
        $this->siswa_model->update($data,$kd_siswa);
        redirect('siswa');
    }
    else
    {
        $kd_siswa = $this->input->post('kd_siswa');
        $data['siswa'] = $this->siswa_model->getById($kd_siswa);
        $this->load->view('template/header');
        $this->load->view('siswa/edit',$data);
        $this->load->view('template/footer');
    }
}

function delete($kd_siswa)
{
    $this->siswa_model->delete($kd_siswa);
    redirect('siswa');
}
}

```

Berikutnya kita buka file model Siswa_model.php, kita tambahkan method delete yang memiliki 1 parameter yaitu variabel \$id

```

<?php defined('BASEPATH') OR exit('No direct script access allowed');

class Siswa_model extends CI_Model
{
    private $table = "tb_siswa";

    public function getAll()
    {

```

```

return $this->db->get($this->table)->result();
    }

    public function save($data)
    {
        return $this->db->insert($this->table, $data);
    }

    public function getById($id)
    {
        return $this->db->get_where($this->table, ["kd_siswa" => $id])->row();
    }

    public function update($data,$id)
    {
        return $this->db->update($this->table, $data, array('kd_siswa' => $id));
    }

    public function delete($id)
    {
        return $this->db->delete($this->table, array("kd_siswa" => $id));
    }
}

```

Keterangan :

Controller Siswa, Line 93 kita membuat method delete, dimana memiliki 1 parameter \$kd_siswa, line 95 kita akan mengakses method delete yang ada dalam model siswa_model dan juga mengirimkan parameter variabel \$kd_siswa

```

function delete($kd_siswa)
{
    $this->siswa_model->delete($kd_siswa);
    redirect('siswa');
}

```

Model Siswa_model Line 27, kita membuat method delete dengan memiliki 1 parameter yaitu variabel \$id, pada method delete ini kita tuliskan perintah query untuk delete data, dimana kita perlu menuliskan 2 bagian ketika menuliskan query delete yaitu :

1. nama tabel yang datanya akan dihapus, dalam contoh ini kita tuliskan \$this->table yang bernilai 'tb_siswa'
2. kondisi data mana yang akan dihapus, kita menuliskan array kd_siswa dengan value variabel \$id, yang artinya menghapus data siswa berdasarkan kd_siswa yang nilainya sama dengan variabel \$id

```

public function delete($id)
{
    return $this->db->delete($this->table, array("kd_siswa" => $id));
}

```

8.1.14. Hasil Kode Akhir

File Controller **Siswa.php**

```
<?php
defined('BASEPATH') OR exit('No direct script access allowed');

class Siswa extends CI_Controller {

    public function __construct()
    {
        parent::__construct();
        $this->load->model("siswa_model");
        $this->load->library('form_validation');
    }

    public function index()
    {
        $data['siswa'] = $this->siswa_model->getAll();
        $this->load->view('template/header');
        $this->load->view('siswa/index',$data);
        $this->load->view('template/footer');
    }

    public function create()
    {
        $this->load->view('template/header');
        $this->load->view('siswa/create');
        $this->load->view('template/footer');
    }

    public function save()
    {
        $this->form_validation->set_rules('nama','Nama','required');
        $this->form_validation->set_rules('jenis_kelamin','Jenis
Kelamin','required');
        $this->form_validation->set_rules('tempat_lahir','Tempat
Lahir','required');
        $this->form_validation->set_rules('tanggal_lahir','Tanggal
Lahir','required');
        $this->form_validation->set_rules('no_telp','Nomor
Telepon','required');
        $this->form_validation->set_rules('alamat','Alamat','required');
        if ($this->form_validation->run()==true)
        {
            $data['nama'] = $this->input->post('nama');
            $data['jenis_kelamin'] = $this->input->post('jenis_kelamin');
            $data['tempat_lahir'] = $this->input->post('tempat_lahir');
            $data['tanggal_lahir'] = $this->input->post('tanggal_lahir');
            $data['no_telp'] = $this->input->post('no_telp');
            $data['alamat'] = $this->input->post('alamat');
            $this->siswa_model->save($data);
            redirect('siswa');
```

```

    }
    else
    {
        $this->load->view('template/header');
        $this->load->view('siswa/create');
        $this->load->view('template/footer');
    }
}

function edit($kd_siswa)
{
    $data['siswa'] = $this->siswa_model->getById($kd_siswa);
    $this->load->view('template/header');
    $this->load->view('siswa/edit',$data);
    $this->load->view('template/footer');
}

public function update()
{
    $this->form_validation->set_rules('nama','Nama','required');
    $this->form_validation->set_rules('jenis_kelamin','Jenis
Kelamin','required');
    $this->form_validation->set_rules('tempat_lahir','Tempat
Lahir','required');
    $this->form_validation->set_rules('tanggal_lahir','Tanggal
Lahir','required');
    $this->form_validation->set_rules('no_telp','Nomor
Telepon','required');
    $this->form_validation->set_rules('alamat','Alamat','required');
    if ($this->form_validation->run()==true)
    {
        $kd_siswa = $this->input->post('kd_siswa');
        $data['nama'] = $this->input->post('nama');
        $data['jenis_kelamin'] = $this->input->post('jenis_kelamin');
        $data['tempat_lahir'] = $this->input->post('tempat_lahir');
        $data['tanggal_lahir'] = $this->input->post('tanggal_lahir');
        $data['no_telp'] = $this->input->post('no_telp');
        $data['alamat'] = $this->input->post('alamat');
        $this->siswa_model->update($data,$kd_siswa);
        redirect('siswa');
    }
    else
    {
        $kd_siswa = $this->input->post('kd_siswa');
        $data['siswa'] = $this->siswa_model->getById($kd_siswa);
        $this->load->view('template/header');
        $this->load->view('siswa/edit',$data);
        $this->load->view('template/footer');
    }
}

function delete($kd_siswa)
{
    $this->siswa_model->delete($kd_siswa);
}

```

```

        redirect('siswa');
    }
}

```

File Model **Siswa_model.php**

```

<?php defined('BASEPATH') OR exit('No direct script access allowed');

class Siswa_model extends CI_Model
{
    private $table = "tb_siswa";

    public function getAll()
    {
        return $this->db->get($this->table)->result();
    }

    public function save($data)
    {
        return $this->db->insert($this->table, $data);
    }

    public function getById($id)
    {
        return $this->db->get_where($this->table, ["kd_siswa" => $id])->row();
    }

    public function update($data,$id)
    {
        return $this->db->update($this->table, $data, array('kd_siswa' => $id));
    }

    public function delete($id)
    {
        return $this->db->delete($this->table, array("kd_siswa" => $id));
    }
}

```

File **Header.php** di folder **application/views/template/header.php**

```

<!doctype html>
<html lang="en">
<head>
<title>Belajar Crud di Codeigniter</title>
<link href="<?php echo base_url(); ?>assets/css/bootstrap.min.css"
rel="stylesheet">
<link href="<?php echo base_url(); ?>assets/css/bootstrap-
datepicker3.min.css" rel="stylesheet">
<script src="<?php echo base_url(); ?>assets/js/jquery-3.5.0.min.js"></script>
<script src="<?php echo base_url(); ?>assets/js/bootstrap.min.js"></script>
<script src="<?php echo base_url(); ?>assets/js/bootstrap-
datepicker.min.js"></script>

```

```
</head>
<body>
```

File **index.php** di folder `application/views/siswa/index.php`

```
<main role="main" class="container">
    <div class="card">
        <div class="card-header">Data Siswa</div>
        <div class="card-body">
            <a href="<?php echo base_url(); ?>siswa/create"
class="btn btn-success">Create</a>
            <br/>
            <br/>
            <table class="table table-bordered">
                <tr>
                    <th width="5%">No</th>
                    <th>Nama</th>
                    <th>Jenis Kelamin</th>
                    <th>Tanggal Lahir</th>
                    <th>Tempat Lahir</th>
                    <th>Telp</th>
                    <th>Alamat</th>
                    <th>Action</th>
                </tr>
                <?php
                $no = 1;
                foreach($siswa as $row)
                {
                    <?>
                    <tr>
                        <td width="5%"><?php echo
$no++; ?></td>
                        <td><?php echo $row->nama;
?></td>
                        <td><?php echo $row-
>jenis_kelamin; ?></td>
                        <td><?php echo $row-
>tanggal_lahir; ?></td>
                        <td><?php echo $row-
>tempat_lahir; ?></td>
                        <td><?php echo $row->no_telp;
?></td>
                        <td><?php echo $row->alamat;
?></td>
                        <td>
                            <a href="<?php echo base_url();
?>siswa/edit/<?php echo $row->kd_siswa; ?>" class="btn btn-warning">Edit</a>
                            <a href="<?php echo base_url();
?>siswa/delete/<?php echo $row->kd_siswa; ?>" class="btn btn-danger">Hapus</a>
                        </td>
                    </tr>
                <?php
```

```

    }
    ?>
  </table>
</div>
</div>
</main>

```

File **Footer.php** di folder **application/views/template/footer.php**

```

<footer class="footer">
  <div class="container" style="text-align:center;">
    <hr/>
    <span>Warung Belajar @<?php echo date('Y'); ?></span>
  </div>
</footer>
</body>
</html>

```

File **index.php** di folder **application/views/siswa/index.php**

```

<main role="main" class="container">
  <div class="card">
    <div class="card-header">Data Siswa</div>
    <div class="card-body">
      <a href="<?php echo base_url(); ?>siswa/create"
class="btn btn-success">Create</a>
      <br/>
      <br/>
      <table class="table table-bordered">
        <tr>
          <th width="5%">No</th>
          <th>Nama</th>
          <th>Jenis Kelamin</th>
          <th>Tanggal Lahir</th>
          <th>Tempat Lahir</th>
          <th>Telp</th>
          <th>Alamat</th>
          <th>Action</th>
        </tr>
        <?php
        $no = 1;
        foreach($siswa as $row)
        {
          ?>
          <tr>
            <td width="5%"><?php echo
            $no++; ?></td>
            <td><?php echo $row->nama;
            ?></td>
            <td><?php echo $row-
            >jenis_kelamin; ?></td>
            <td><?php echo $row-
            >tanggal_lahir; ?></td>

```



```

        <td><?php echo $row-
>tempat_lahir; ?></td>
        <td><?php echo $row->no_telp;
?></td>
        <td><?php echo $row->alamat;
?></td>
        <td>
        <a href="<?php echo base_url();
?>siswa/edit/<?php echo $row->kd_siswa; ?>" class="btn btn-warning">Edit</a>
        <a href="<?php echo base_url();
?>siswa/delete/<?php echo $row->kd_siswa; ?>" class="btn btn-danger">Hapus</a>
        </td>
    </tr>
    <?php
    }
    ?>
</table>
</div>
</div>
</main>

```

File **create.php** di folder **application/views/siswa/create.php**

```

<div class="container">
    <div class="card">
        <div class="card-header">Create Siswa</div>
        <div class="card-body">
            <?php
            if(validation_errors() != false)
            {
                ?>
                <div class="alert alert-danger" role="alert">
                    <?php echo validation_errors(); ?>
                </div>
            }
            ?>
            <form method="post" action="<?php echo base_url();
?>siswa/save">
                <div class="form-group">
                    <label for="nama">Nama</label>
                    <input type="text" class="form-control"
id="nama" name="nama">
                </div>
                <div class="form-group">
                    <label for="jenis_kelamin">Jenis
Kelamin</label>
                    <select name="jenis_kelamin"
id="jenis_kelamin" class="form-control">
                        <option value="pria">Pria</option>
                        <option
value="wanita">Wanita</option>

```

```

        </select>
    </div>

    <div class="form-group">
        <label for="tempat_lahir">Tempat Lahir</label>
        <input type="text" class="form-control"
id="tempat_lahir" name="tempat_lahir">
    </div>

    <div class="form-group">
        <label for="tgl_lahir">Tanggal Lahir</label>
        <input type="text" class="form-control
datepicker" id="tanggal_lahir" name="tanggal_lahir">
    </div>

    <div class="form-group">
        <label for="no_telp">No Telp</label>
        <input type="number" class="form-control"
id="no_telp" name="no_telp">
    </div>

    <div class="form-group">
        <label for="alamat">Alamat</label>
        <textarea class="form-control" name="alamat"
id="alamat"></textarea>
    </div>

    <button type="submit" class="btn btn-
primary">Submit</button>
</form>
</div>
</div>
</div>

```

File **edit.php** di folder **application/views/siswa/edit.php**

```

<div class="container">
    <div class="card">
        <div class="card-header">Edit Siswa</div>
        <div class="card-body">
            <?php
            if(validation_errors() != false)
            {
                <?>
                <div class="alert alert-danger" role="alert">
                    <?php echo validation_errors(); <?>
                </div>
                <?php
            }
            <?>
            <form method="post" action="<?php echo base_url();
?>siswa/update">
                <input type="hidden" name="kd_siswa" id="kd_siswa"
value="<?php echo $siswa->kd_siswa; ?>" />

```

```

        <div class="form-group">
            <label for="nama">Nama</label>
            <input type="text" value="<?php echo $siswa-
>nama; ?>" class="form-control" id="nama" name="nama">
        </div>

        <div class="form-group">
            <label for="jenis_kelamin">Jenis
Kelamin</label>
            <select name="jenis_kelamin"
id="jenis_kelamin" class="form-control">
                <option value="pria" <?php echo
($siswa->jenis_kelamin ? 'pria' : 'selected' ); ?> >Pria</option>
                <option value="wanita" <?php echo
($siswa->jenis_kelamin ? 'wanita' : 'selected' ); ?>>Wanita</option>
            </select>
        </div>

        <div class="form-group">
            <label for="tempat_lahir">Tempat Lahir</label>
            <input type="text" class="form-control"
id="tempat_lahir" name="tempat_lahir" value="<?php echo $siswa->tempat_lahir; ?>">
        </div>

        <div class="form-group">
            <label for="tgl_lahir">Tanggal Lahir</label>
            <input type="text" class="form-control
datepicker" readonly id="tanggal_lahir" name="tanggal_lahir" value="<?php echo
$siswa->tanggal_lahir; ?>">
        </div>

        <div class="form-group">
            <label for="no_telp">No Telp</label>
            <input type="number" class="form-control"
value="<?php echo $siswa->no_telp; ?>" id="no_telp" name="no_telp">
        </div>

        <div class="form-group">
            <label for="alamat">Alamat</label>
            <textarea class="form-control" name="alamat"
id="alamat"><?php echo $siswa->alamat; ?></textarea>
        </div>

        <button type="submit" class="btn btn-
primary">Update</button>
    </form>
</div>
</div>

```

8.2. Membuat Laporan PDF dengan DOMPDP di Codeigniter

Kita akan menginstall Dompdf dengan menggunakan Composer, karena itu kita perlu mengaktifkan autoload untuk composer.

Silahkan buka file dengan nama application/config/config.php, kemudian isi bagian autoload composer menjadi seperti dibawah ini :

```
$config['composer_autoload'] = "vendor/autoload.php";
```

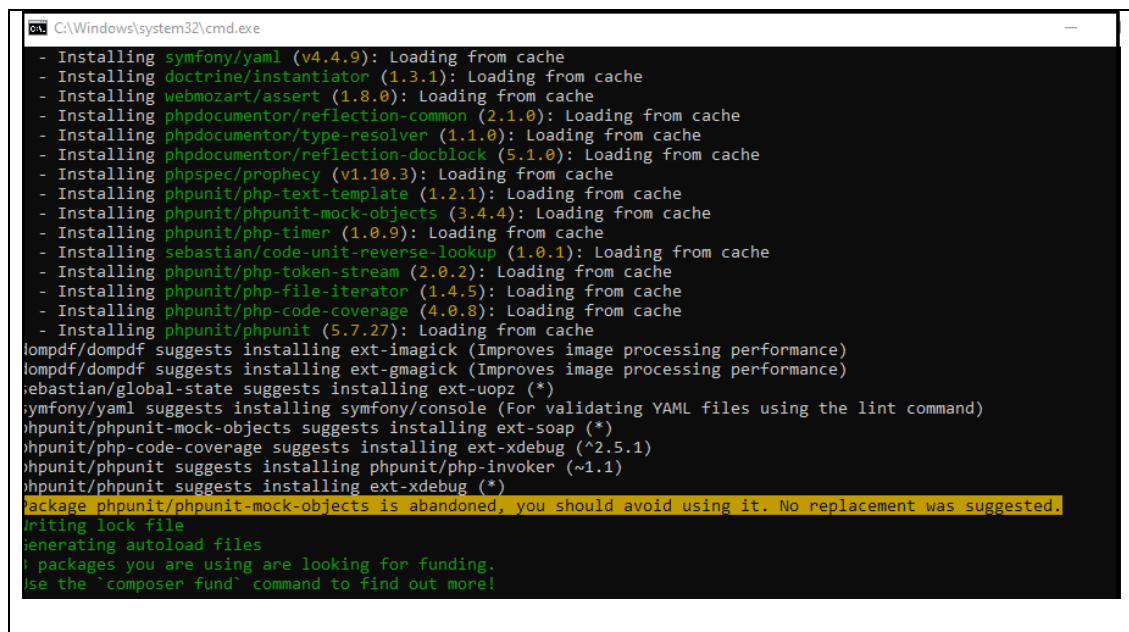
8.2.1. Install Dompdf dengan Composer

Pastikan anda sudah menginstall Composer, untuk instalasi Composer bisa download disini : [Klik Disini](#)

Berikutnya buka Command Prompt / Terminal, dan akses folder project belajarcodeigniter, lalu tuliskan perintah seperti berikut ini :

```
composer require dompdf/dompdf
```

Tunggu proses download dompdf melalui composer.



8.2.2. Membuat Library untuk kebutuhan DomPDF

Berikutnya kita akan membuat library sendiri untuk kebutuhan dompdf, agar dompdf dapat cepat digunakan saat dibutuhkan.

Silahkan buat file dengan nama pdf.php didalam folder application/libraries

untuk isi codenya seperti berikut ini :

```
<?php defined('BASEPATH') OR exit('No direct script access allowed');

/**

 * CodeIgniter DomPDF Library
```

```

*
* Generate PDF's from HTML in CodeIgniter
*
* @package      CodeIgniter
* @subpackage   Libraries
* @category     Libraries
* @author       Ardianta Pargo
* @license      MIT License
* @link         https://github.com/ardianta/codeigniter-dompdf
*/

use Dompdf\Dompdf;

class Pdf extends Dompdf{

    /**
     * PDF filename
     * @var String
     */
    public $filename;

    public function __construct(){
        parent::__construct();

        $this->filename = "laporan.pdf";
    }

    /**
     * Get an instance of CodeIgniter
     *
     * @access      protected
     * @return     void
     */
}

```

```

protected function ci()
{
    return get_instance();
}

/**
 * Load a CodeIgniter view into domPDF
 *
 * @access    public
 * @param    string    $view The view to load
 * @param    array    $data The view data
 * @return    void
 */
public function load_view($view, $data = array()){
    $html = $this->ci()->load->view($view, $data, TRUE);
    $this->load_html($html);
    // Render the PDF
    $this->render();
    // Output the generated PDF to Browser
    $this->stream($this->filename, array("Attachment" =>
false));
}
}

```

Kode diatas kita membuat class PDF dengan extends di class Dompdf.

8.2.3. Persiapkan database

Dalam contoh ini kita akan belajar untuk menampilkan data dari database, dan ditampilkan dalam bentuk PDF, sehingga kita perlu mempersiapkan database terlebih dahulu.

Dalam contoh ini kita buat database dengan nama db_siswa, dan kita juga membuat tabel siswa didalamnya dengan kolom sebagai berikut :

- kd_siswa (Primary Key, Int, Auto Increment)
- nama (Varchar 255)
- kelas (Varchar 5)
- jenis_kelamin (Enum ('PRIA', 'WANITA'))
- alamat (Varchar 255)

Berikutnya kita akan isi tabel dengan data sebagai berikut :

- Data 1

kd_siswa(1), nama (Budi Hermawan), kelas (3MM1), jenis_kelamin(PRIA), alamat (Wonocolo, Surabaya)

- Data 2

kd_siswa(2), nama (Cika Larasati), kelas (3MM1), jenis_kelamin(WANITA), alamat (Sedati, Sidoarjo)

disini kita tidak membahas detail teknis bagaimana membuat database dan tabel didalamnya ya.. anda bisa gunakan Query SQL berikut ini untuk dapat membuat database disertai tabelnya :

```
-- phpMyAdmin SQL Dump
-- version 5.0.2
-- https://www.phpmyadmin.net/
--
-- Host: 127.0.0.1
-- Generation Time: Jun 11, 2020 at 01:01 PM
-- Server version: 10.4.11-MariaDB
-- PHP Version: 7.4.5

SET SQL_MODE = "NO_AUTO_VALUE_ON_ZERO";
START TRANSACTION;
SET time_zone = "+00:00";

/*!40101 SET @OLD_CHARACTER_SET_CLIENT=@@CHARACTER_SET_CLIENT */;
```

```

/*!40101 SET @OLD_CHARACTER_SET_RESULTS=@@CHARACTER_SET_RESULTS */;

/*!40101 SET @OLD_COLLATION_CONNECTION=@@COLLATION_CONNECTION */;

/*!40101 SET NAMES utf8mb4 */;

--

-- Database: `db_siswa`

--

CREATE DATABASE IF NOT EXISTS `db_siswa` DEFAULT CHARACTER SET
utf8mb4 COLLATE utf8mb4_general_ci;

USE `db_siswa`;

-----

--

-- Table structure for table `siswa`

--

CREATE TABLE `siswa` (
  `kd_siswa` int(11) NOT NULL,
  `nama` varchar(255) NOT NULL,
  `kelas` varchar(5) NOT NULL,
  `jenis_kelamin` enum('PRIA','WANITA') NOT NULL,
  `alamat` varchar(255) NOT NULL
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4;

--

-- Dumping data for table `siswa`

--

```



```

INSERT INTO `siswa` (`kd_siswa`, `nama`, `kelas`, `jenis_kelamin`,
`alamat`) VALUES

(1, 'Budi Hermawan', '3MM1', 'PRIA', 'Wonocolo, Surabaya'),

(2, 'Cika Larasati', '3MM1', 'WANITA', 'Sedati, Sidoarjo');

--

-- Indexes for dumped tables

--

--

-- Indexes for table `siswa`

--

ALTER TABLE `siswa`

  ADD PRIMARY KEY (`kd_siswa`);

--

-- AUTO_INCREMENT for dumped tables

--

--

-- AUTO_INCREMENT for table `siswa`

--

ALTER TABLE `siswa`

  MODIFY `kd_siswa` int(11) NOT NULL AUTO_INCREMENT,
  AUTO_INCREMENT=3;

COMMIT;

```

```
/*!40101 SET CHARACTER_SET_CLIENT=@OLD_CHARACTER_SET_CLIENT */;  
/*!40101 SET CHARACTER_SET_RESULTS=@OLD_CHARACTER_SET_RESULTS */;  
/*!40101 SET COLLATION_CONNECTION=@OLD_COLLATION_CONNECTION */;
```

8.2.4. Setting Database di Codeigniter

Berikutnya kita akan setting codeigniter agar dapat mengakses database, silahkan buka file application/config/autoload.php, lalu ubah code dibawah ini :

```
$autoload['libraries'] = array();
```

Menjadi

```
$autoload['libraries'] = array('database');
```

Berikutnya silahkan buka file application/config/database.php

lalu konfigurasi setingan database, sebagai contoh seperti berikut ini :

```
$db['default'] = array(  
    'dsn' => "",  
    'hostname' => 'localhost',  
    'username' => 'root',  
    'password' => "",  
    'database' => 'db_siswa',  
    'dbdriver' => 'mysqli',  
    'dbprefix' => "",  
    'pconnect' => FALSE,  
    'db_debug' => (ENVIRONMENT !== 'production'),  
    'cache_on' => FALSE,  
    'cachedir' => "",  
    'char_set' => 'utf8',  
    'dbcollat' => 'utf8_general_ci',  
    'swap_pre' => "",  
    'encrypt' => FALSE,  
    'compress' => FALSE,  
    'stricton' => FALSE,  
    'failover' => array(),  
    'save_queries' => TRUE  
);
```

dalam contoh diatas saya hanya lakukan konfigurasi pada :

- hostname : localhost
- username : root
- password : saya kosongi
- database : db_siswa

8.2.5. Membuat Model

Karena nanti kita melakukan komunikasi dengan database sehingga kita perlu membuat model, silahkan buat model dengan nama Siswa_model.php didalam folder application/models untuk codenya adalah sebagai berikut :

```
<?php
Class Siswa_model extends CI_Model{

    function getData()
    {
        $data_siswa = $this->db->get('siswa');
        return $data_siswa->result();
    }
}
?>
```

Keterangan :

Kita buat class dengan nama Siswa_model dengan extends class CI_Model.

lalu didalamnya kita buat method getData, didalamnya ada perintah untuk get data ditabel siswa (Line 6) dan kita return hasil data tersebut (Line 7)

8.2.6. Membuat Controller

Berikutnya kita akan membuat controller dimana didalamnya kita akan buat method yang nantinya akan kita akses di browser untuk menampilkan laporan PDF, dengan menggunakan Library DomPDF

buat controller dengan nama Siswa.php didalam folder application/controllers/Siswa.php, dengan code seperti berikut ini :

```
<?php
defined('BASEPATH') OR exit('No direct script access allowed');

class Siswa extends CI_Controller {

    public function index()
    {
        $this->load->model('siswa_model');
        $data['siswa'] = $this->siswa_model->getData();
        $this->load->library('pdf');
        $this->pdf->setPaper('A4', 'potrait');
        $this->pdf->filename = "laporan-data-siswa.pdf";
        $this->pdf->load_view('laporan_siswa', $data);
    }
}
```

Keterangan :

- Kita buat class dengan nama Siswa yang extends di class CI_Controller

- didalamnya kita buat method index (Line 6)
- Kita load model siswa_model (Line 8)
- kita mengakses method getData didalam model siswa_model, untuk mendapatkan data siswa, dan hasilnya kita simpan pada array \$data di elemen siswa (Line 9)
- kita load library pdf, dimana library ini merupakan library yang sudah extends dari library dompdf (Line 10)
- kita set kertasnya menggunakan ukuran A4, dengan mode potrait (Line 11)
- kita set nama file pdf dengan nama “laporan-data-siswa.pdf” (Line 12)
- kita set view untuk membuat report file PDF, untuk nama file view laporan_siswa, dan passing variabel \$data untuk digunakan di view laporan_siswa (Line 13)

8.2.7. Membuat file View

Berikutnya kita buat file view, dengan nama laporan_siswa, file view berada application/views

untuk isi codenya adalah sebagai berikut :

```
<h2><center>Data Siswa</center></h2>
<hr/>
<table border="1" width="100%" style="text-align:center;">
  <tr>
    <th>No</th>
    <th>Nama</th>
    <th>Kelas</th>
    <th>Jenis Kelamin</th>
    <th>Alamat</th>
  </tr>
  <?php
  $no = 1;
  foreach($siswa as $row)
  {
    ?>
    <tr>
      <td><?php echo $no++; ?></td>
      <td><?php echo $row->nama; ?></td>
      <td><?php echo $row->kelas; ?></td>
      <td><?php echo $row->jenis_kelamin; ?></td>
      <td><?php echo $row->alamat; ?></td>
    </tr>
  <?php
  }
  ?>
</table>
```

Keterangan :

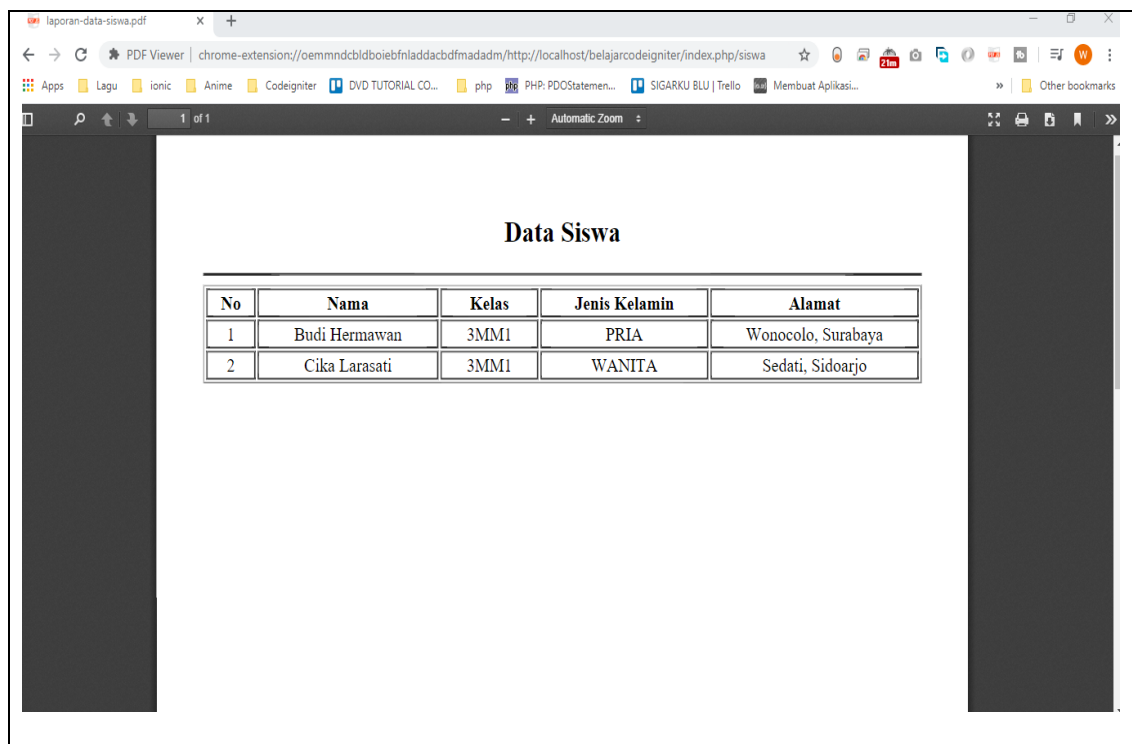
- untuk membuat tampilan report dalam bentuk PDF, kita cukup menuliskan perintah HTML, disertai PHP untuk menampilkan data, dimana akan otomatis akan dikonversi menjadi bentuk PDF.

- kita ingin menampilkan laporan PDF dalam bentuk tabel, maka kita bisa menggunakan perintah HTML untuk membuat tabel seperti yang kita tuliskan di line 3 – 26
- Line 14 kita menggunakan perintah foreach untuk extract variabel \$siswa, dimana setiap data siswa akan diwakili dengan variabel \$row.
- Line 17 kita menampilkan nomor dengan variabel \$no
- Line 18 kita menampilkan isi kolom nama
- Line 19 kita menampilkan isi kolom kelas
- Line 29 kita menampilkan isi kolom jenis_kelamin
- Line 30 kita menampilkan isi kolom alamat

8.2.8. Melakukan testing menampilkan laporan PDF

Berikutnya kita akan coba mengakses method di controller siswa untuk menampilkan laporan PDF, dengan alamat : localhost/belajarcodigniter/index.php/siswa

maka tampilannya adalah sebagai berikut ini :



The screenshot shows a PDF viewer displaying a table titled "Data Siswa". The table has five columns: "No", "Nama", "Kelas", "Jenis Kelamin", and "Alamat". There are two rows of data in the table.

No	Nama	Kelas	Jenis Kelamin	Alamat
1	Budi Hermawan	3MM1	PRIA	Wonocolo, Surabaya
2	Cika Larasati	3MM1	WANITA	Sedati, Sidoarjo

maka akan ditampilkan laporan berupa file PDF, dimana bentuk laporan ini adalah tabel HTML yang dikonversi menjadi bentuk PDF .

DAFTAR PUSTAKA

Sala-Zárate, M., & Colombo-Mendoza, L. (2012). {CLOUD} {COMPUTING}: A {REVIEW} {OF} {PAAS}, {IAAS}, {SAAS} {SERVICES} {AND} {PROVIDERS}. *Lámpsakos*, 7, 47. <https://doi.org/10.21501/21454086.844>

<https://www.warungbelajar.com/category/tutorial-codeigniter-3>

<https://www.warungbelajar.com/category/tutorial-codeigniter-4>

diolah dari berbagai sumber

ISBN 978-623-6602-78-2



9 786236 602782