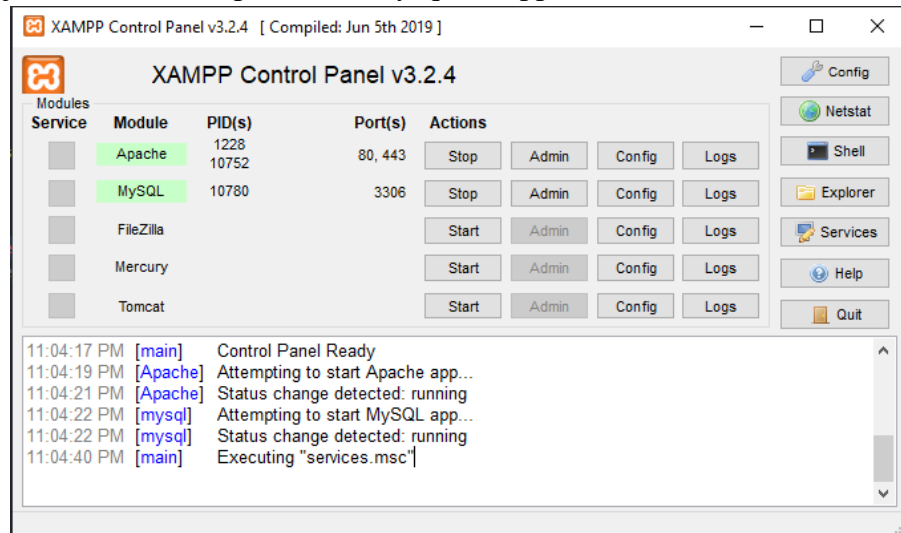


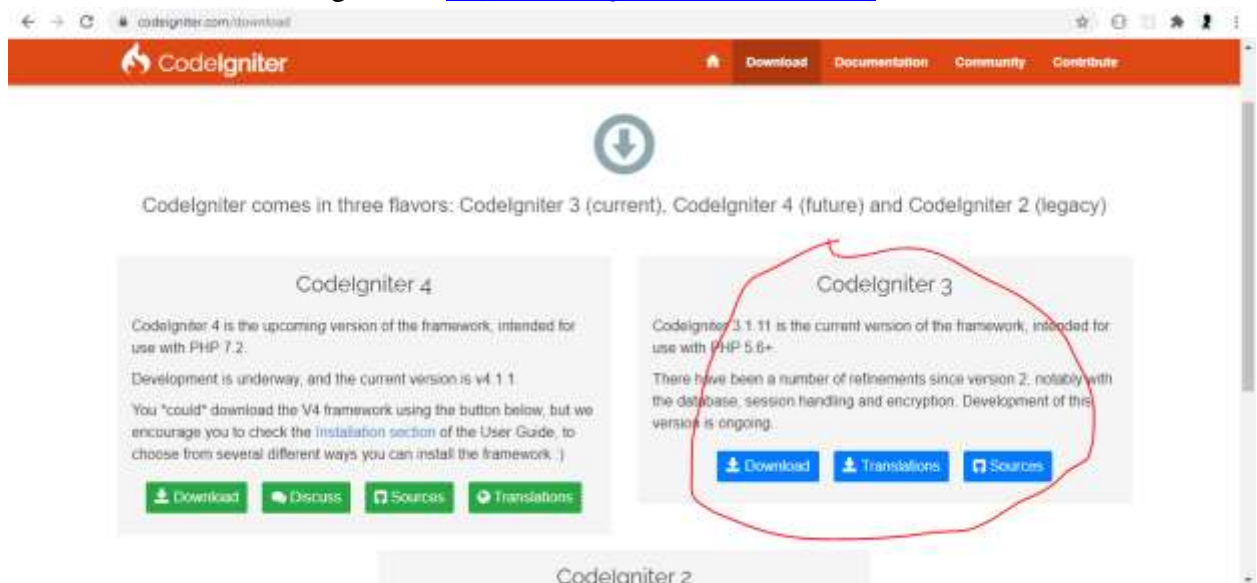
Pertemuan 2 Konfigurasi Dasar Codeigniter, Konsep MVC, Pemanggilan library, Pengenalan Function

1. Instalasi Codeigniter

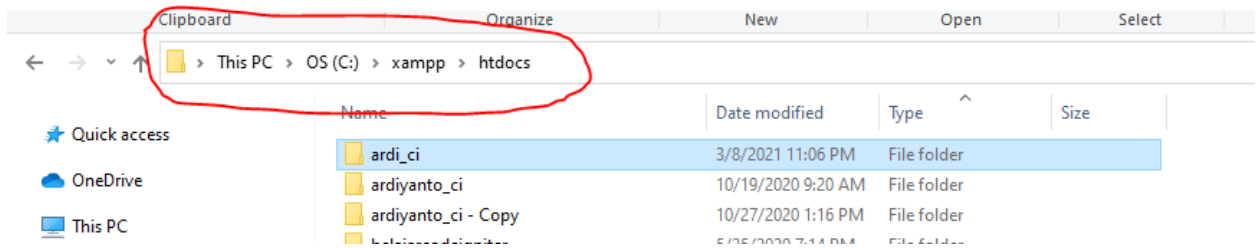
- a. Silahkan jalankan service apache dan mysql xampp



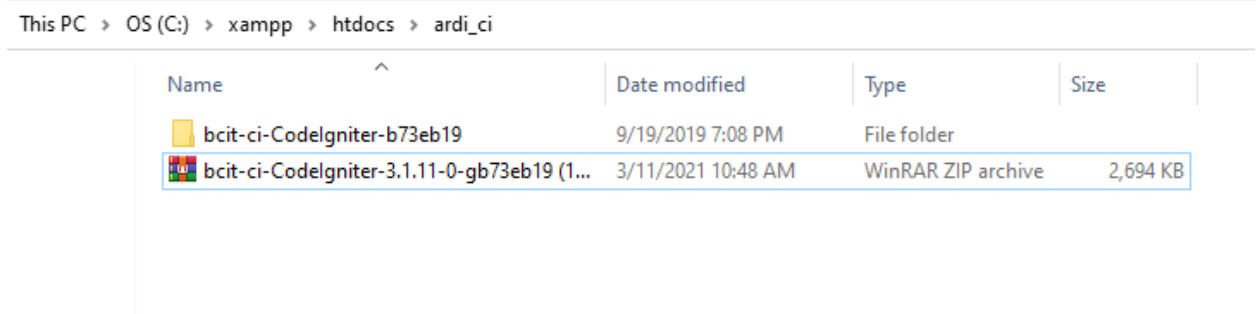
- b. Silahkan download codeigniter di <https://codeigniter.com/download>



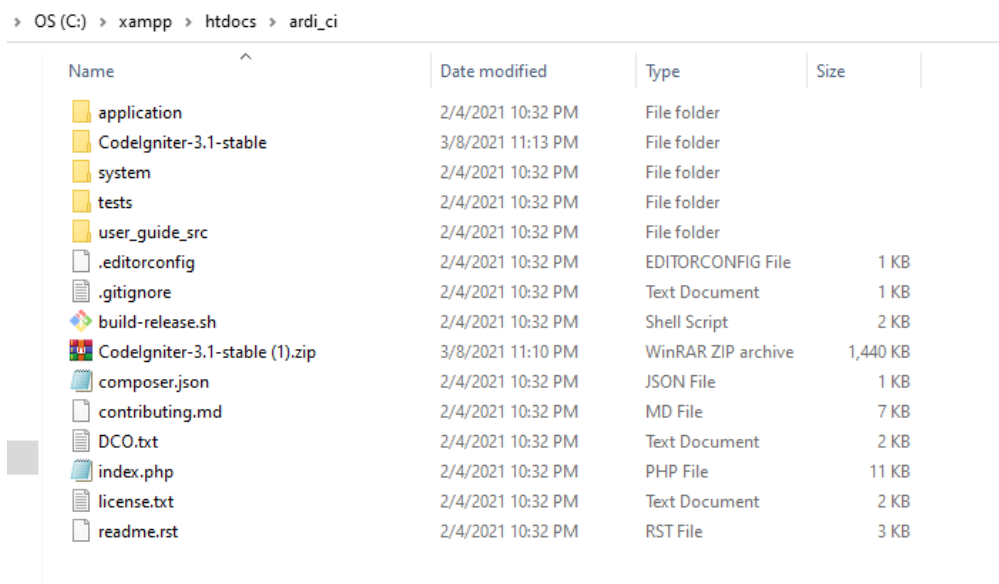
- c. Buat Folder dengan format nama_ci di dalam web server atau di c:/xampp/htdocs (xampp), Application/mamp/htdocs/(mamp)



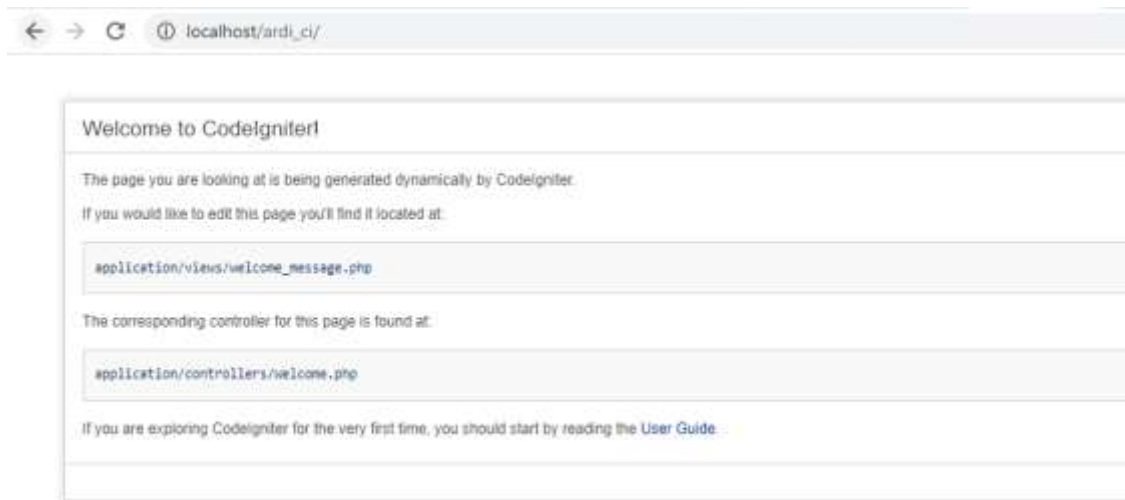
- d. Extract file yang sudah kita download ke dalam folder yang sudah dibuat



- e. keluarkan semua file yang ada didalam folder **bcit-ci-odeingniter** yang terdiri dari (Application, system, dll)



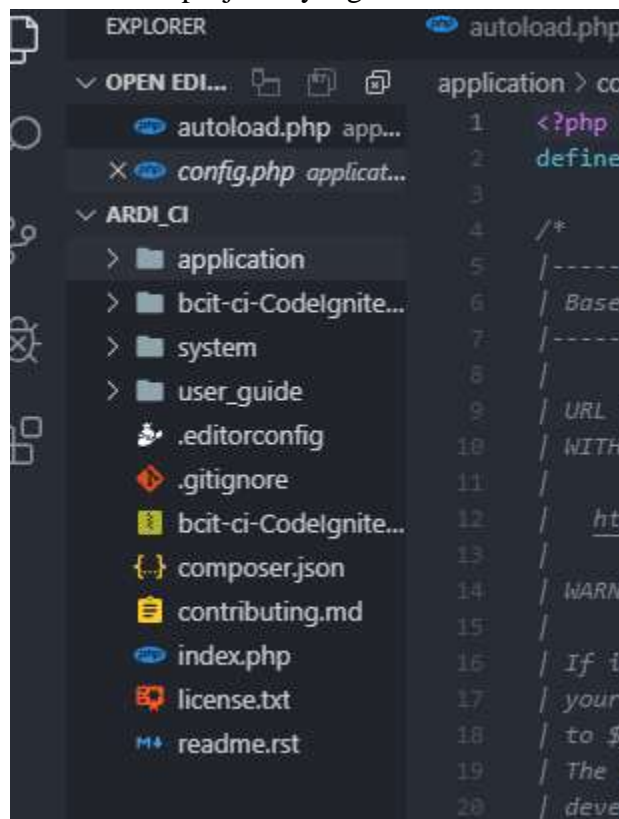
- f. Cek instalasi dari codeigniter dengan membuka browser dan masukkan url http://localhost/nama_proyek/ jika berhasil maka akan muncul tampilan seperti berikut



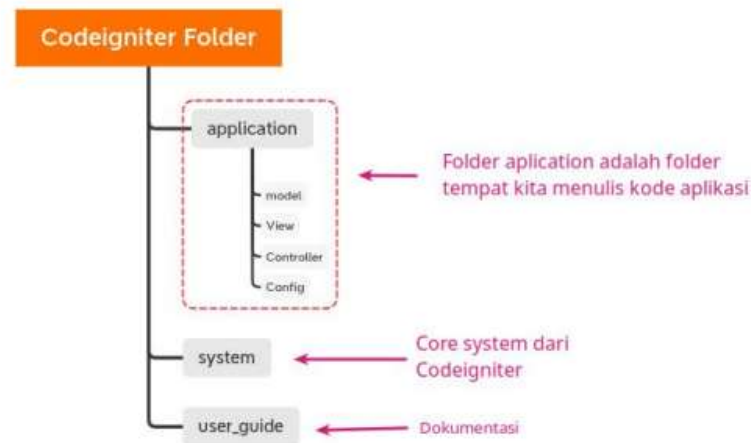
Jika port apache kalian 8080 atau yang lainnya silahkan akses dengan menggunakan http://localhost:8080/nama_proyek/

2. Pengenalan Struktur Folder

- Silahkan buka Vscode
- Open folder dan arahkan ke folder proyek ci yang ada di dalam web server c:/xampp/htdocs



Terdapat dua direktori penting di dalam CI yang harus kamu perhatikan, yakni: `application` dan `system`. Selain itu terdapat juga direktori `user_guide` dan beberapa file lainnya.



Berikut penjelasannya

- application berisi semua kode aplikasi. Di dalam direktori inilah kita akan menulis semua kode aplikasi kita.
 - system berisi kode-kode inti dari Codeigniter. Jangan mengubah apapun di dalam direktori ini. Jika kita ingin upgrade versi, kita cukup me-replace direktori ini dengan yang baru.
 - user_guide berisi dokumentasi codeigniter. Kita bisa menghapus direktori ini saat web sudah jadi.
 - 📄 .editor_config berisi konfigurasi untuk teks editor.
 - 📄 .gitignore berisi daftar file dan folder yang akan diabaikan oleh Git.
 - 📄 composer.json adalah file yang berisi keterangan project dan keterangan library yang digunakan. File ini dibutuhkan oleh composer
 - 📖 contributing.md adalah file yang berisi penjelasan cara berkontribusi di proyek CI. Kita bisa menghapus file ini, apabila web sudah jadi.
 - 📖 license.txt adalah file yang berisi keterangan lisensi dari CI. Kita juga bisa menghapus file saat web sudah jadi.
 - 📖 readme.rst sama seperti file
 - 📖 contributing.md file ini berisi penjelasan dan informasi tentang project CI. Kita juga bisa menghapus file ini saat web sudah selesai.
 - 📄 index.php adalah file utama dari CI. File yang akan dibuka pertamakali saat aplikasi dibuka.
- c. Selanjutnya silahkan buka direktori **application** dan perhatikan direktori yang ada di sana.
- Cache berisi cache dari aplikasi.
 - config berisi konfigurasi aplikasi.
 - 📄 autoload.php tempat kita mendefinisikan autoload;
 - 📄 config.php konfigurasi aplikasi;

- ▣ constants.php berisi konstanta;
- ▣ database.php konfigurasi database aplikasi;
- ▣ doctypes.php berisi definisi untuk doctype HTML;
- ▣ foreign_chars.php berisi karakter dan simbol;
- ▣ hooks.php berisi konfigurasi hooks;
- ▣ index.html untuk mencegah direct access;
- ▣ memcached.php berisi konfigurasi untuk memcached;
- ▣ migration.php konfigurasi untuk migrasi;
- ▣ mimes.php berisi definisi tipe file;
- ▣ profiler.php konfigurasi untuk profiler;
- ▣ routers.php tempat kita menulis route aplikasi;
- ▣ smileys.php berisi kode untuk emoji;
- ▣ user_agents.php berisi definisi untuk user agents.
- ▣ controller berisi kode untuk controller.
- ▣ core berisi kode untuk custom core.
- ▣ helpers berisi fungsi-fungsi helper.
- ▣ hooks berisi kode untuk script hook.
- ▣ language berisi string untuk bahasa, apabila web mendukung multibahasa.
- ▣ libraries berisi library.
- ▣ logs berisi logs dari aplikasi.
- ▣ models berisi kode untuk model.
- ▣ thrid_party berisi library dari pihak ketiga.
- ▣ views berisi kode untuk view.
- ▣ index.html file html untuk mencegah direct access.

3. Pengenalan Dasar

a. Konfigurasi Dasar

- Buka file **autoload.php** yang ada di folder **application/config**
 - Pada line ke 61 autoload['libraries'] tambahkan coding dibawah ini.

```
61 $autoload['libraries'] = array('database', 'session', 'email');
```

Konfigurasi ini bertujuan untuk menentukan sumber daya apa yang akan di load secara otomatis.

- Pada line ke 92 autoload['helper'] tambahkan coding dibawah ini

```
92 $autoload['helper'] = array('url', 'file');
```

Helper digunakan dalam view untuk membantu prosesproses yang berulang, seperti generate html, url, security, dan lain-lain.

- Buka file **config.php** yang ada di folder **application/config**

Pada line 26 tambahkan coding dibawah ini

http://localhost/nama_proyek/

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

Konfigurasi ini berisi alamat url sebuah aplikasi. Jika menggunakan helper url maka konfigurasi ini harus di-set dengan benar. Contoh: aplikasi Anda akan diakses dengan menggunakan domain **www.contoh.com/app_ci** maka pada konfigurasi ini harus diisikan: `$config['base_url']='http://www.contoh.com/app_ci/';`

- Selanjutnya silahkan buka phpmyadmin dan buat database dengan format **nama_ci**



- Buka file **database.php** yang ada di folder **application/config**.
Silahkan tambahkan username dan database seperti pada gambar berikut

```

application > config > database.php > ...
73 $active_group = 'default';
74 $query_builder = TRUE;
75
76 $db['default'] = array(
77     'dsn' => '',
78     'hostname' => 'localhost',
79     'username' => 'root',
80     'password' => '',
81     'database' => 'ardi_ci',
82     'dbdriver' => 'mysqli',
83     'dbprefix' => '',
84     'pconnect' => FALSE,
85     'db_debug' => (ENVIRONMENT !== 'production'),
86     'cache_on' => FALSE,
87     'cachedir' => '',
88     'char_set' => 'utf8',
89     'dbcollat' => 'utf8_general_ci',
90     'swap_pre' => '',
91     'encrypt' => FALSE,
92     'compress' => FALSE,
93     'stricton' => FALSE,
94     'failover' => array(),
95     'save_queries' => TRUE
96 );
97

```

- Buka file routes.php yang ada di folder application/config.

```

51
52 $route['default_controller'] = 'welcome';
53 $route['404_override'] = '';
54 $route['translate_uri_dashes'] = FALSE;
55

```

- \$route['default_controller'] digunakan untuk mengatur controller mana yang akan digunakan sebagai controller default pada website.
- \$route['404_override'] digunakan untuk mengatur tampilan jika halaman tidak ditemukan
- \$route['translate_uri_dashes'] digunakan untuk melakukan otomatis replace koma

- Routing

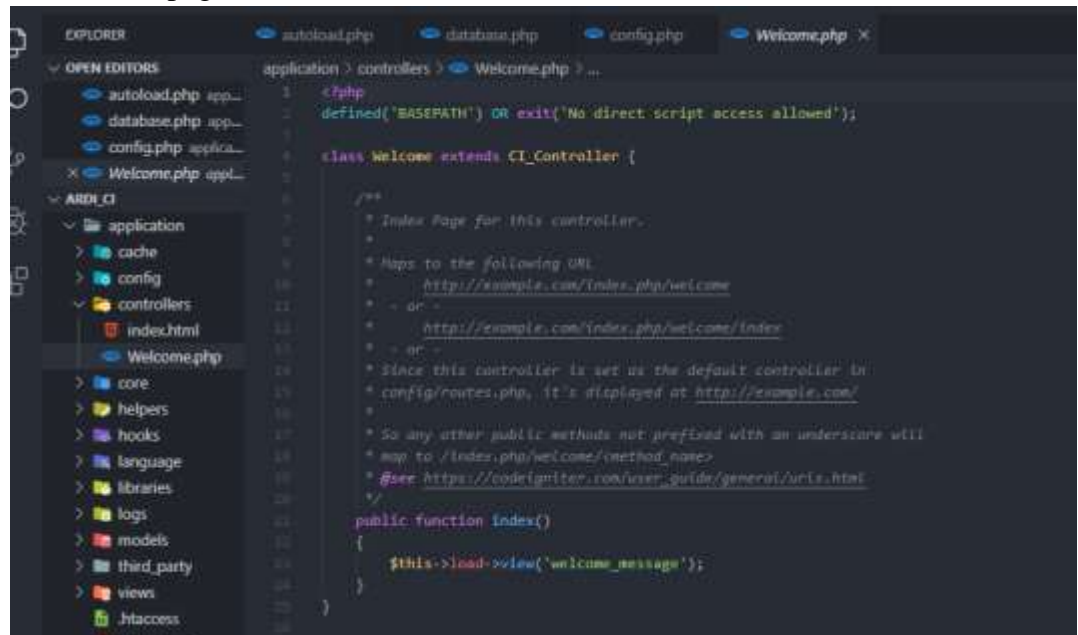
Aturan Routing pada CI

1. Struktur URL adalah sebagai berikut

- localhost/ : host dari web server

- `ardi_ci` : nama folder proyek di web server atau apache (htdocs)
- `Index.php` : file `index.php` yang ada di `ci`. Ini wajib ada tapi bisa juga di hilangkan dengan `.htaccess`
- `Welcome` : nama class dari controller.
- `Index`: nama function yang ada di controller `welcome`
- Buka file `welcome.php` yang terdapat di `application` → `controller`, seperti gambar berikut

Controller berguna sebagai perantara antara Model, View dan berbagai resources yang dibutuhkan untuk memproses HTTP Request dan generate sebuah web page.



- `defined('BASEPATH') OR exit('No direct script access allowed');`, merupakan sintaks PHP yang memastikan tidak ada akses script secara langsung, dan harus diakses melalui www.namaweb.com/controller.
- `class Welcome extends CI_Controller`, merupakan pendeklarasian class yang bernama `Welcome` yang meng-ekstend class `Inti Codeigniter`. Peraturan dalam pembuatan class/controller dalam Codeigniter yaitu, nama class harus sama dengan nama file controller dan berawalan dengan huruf besar. (ex: `welcome.php` maka class nya adalah `Class Welcome extends CI_Controller`).
- `public function index()`, merupakan pendeklarasian function dalam `class`. Sama seperti pembuatan php pada umumnya jika kita membuat `index.php` maka apabila kita membuka parent folder, akan langsung terhubung ke halaman `index.php`, begitu pula dengan public function `index`, jika kita

mengakses suatu Controller yang memiliki function index. maka Controller yang dipilih akan langsung memproses function index.

2. Silahkan tambahkan function baru pada controller **welcome.php**, seperti berikut

```

12  * - or -
13  *   http://example.com/index.php/welcome/index
14  * - or -
15  * Since this controller is set as the default controller in
16  * config/routes.php, it's displayed at http://example.com/
17  *
18  * So any other public methods not prefixed with an underscore will
19  * map to /index.php/welcome/<method_name>
20  * @see https://codeigniter.com/user_guide/general/urls.html
21  */
22  public function index()
23  {
24      $this->load->view('welcome_message');
25  }
26  public function home($nama = "")
27  {
28      echo "hello... ini adalah contoh codeigniter pertama saya ";
29  }
30  public function komentar()
31  {
32      echo "ini adalah function komentar ";
33  }
34  }
35

```

Untuk melihat output home dapat dengan mengakses http://localhost/ardi_ci/index.php/welcome/home dan untuk melihat output komentar dapat dengan mengakses http://localhost/ardi_ci/index.php/welcome/komentar

Tugas :

Silahkan jelaskan dan analisa coding diatas :

- home :
- komentar() :
- \$nama="" :
- Jika \$nama="" dihilangkan, apakah terjadi error?
- Silahkan buat coding dan output seperti pada gambar dibawah ini dari function di atas. Nama sesuaikan dengan nama masing-masing

Silahkan dilengkapi pada laporan

Konsep MVC

MVC merupakan singkatan dari Model-View-Controller. MVC merupakan pola arsitektur yang membagi aplikasi menjadi 3 komponen :

- Model : menangani masalah logika bisnis dan interaksi database
- Controller : mengkoordinasi aktivitas antara Model dan View
- View : bertugas untuk merepresentasikan data kepada user

Menggunakan pola MVC memiliki keuntungan seperti berikut :

- Komponen yang ada berfungsi secara independen antara satu sama lain.
- Fleksibel dalam hal mengubah komponen individual
- Meningkatkan produktivitas karena lebih dari satu orang bisa mengerjakan project yang sama di satu waktu.

Contoh penerapan Controller dan View

- Silahkan buat file controller baru dengan nama **Blog.php** pada **application/controller/** Kemudian tambahkan kode program berikut

```
<?php
defined('BASEPATH') or exit('No direct script access allowed');
class Blog extends CI_Controller
{
    public function __construct()
    {
        parent::__construct();
    }
    function index()
    {
        $this->load->view("hello_codeigniter");
    }
}
```

Didalam function index() terdapat **\$this->load->view**, yang digunakan untuk mengakses halaman view yang kita buat pada folder **application/view**

- Selanjutnya silahkan buat file **hello_codeigniter.php** pada **application/view/** Kemudian tambahkan kode program berikut

```
application > views > hello_codeigniter.php > ...
1 <h1> Hello saya adalah halaman view </h1>
```

Maka kode diatas akan memberikan hasil yang sama dengan contoh kasus pertama (tanpa menggunakan view)

localhost/ardi_ci/index.php/blog

Hello saya adalah halaman view

Fungsi view memiliki 3 parameter

- Nama file view** - Nama file yang hendak di-load yang terletak di dalam folder **application/view** seperti pada contoh sebelumnya

- b. **Data Parameter** - Parameter ini digunakan untuk melewatkan data dari controller ke dalam view.

Contoh kode program, silahkan buat file controller baru dengan nama **Blog2.php** pada **application/controller/**

Kemudian tambahkan kode program berikut

```

application > controllers > Blog2.php > ...
<?php
defined('BASEPATH') or exit('No direct script access allowed');
class Blog2 extends CI_Controller
{
    public function __construct()
    {
        parent::__construct();
    }
    function index()
    {
        $data['judul'] = "Judul blog";
        $data['isi'] = "Isi blog";
        $this->load->view("blog_view", $data);
    }
}

```

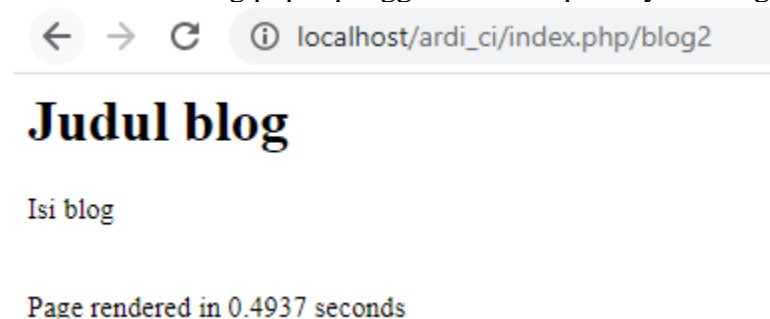
Selanjutnya buatlah file view bernama **blog_view.php** dengan kode program seperti berikut ini:

```

application > views > blog_view.php > ...
1 <h1><?php echo $judul; ?></h1>
2 <p><?php echo $isi; ?></p>
3 <p><br />Page rendered in {elapsed_time} seconds</p>

```

Jika halaman blog.php dipanggil maka tampilannya kurang lebih seperti berikut:



- c. **Output Parameter** - Parameter ini akan di set true jika kita ingin menyimpan hasil view ke dalam sebuah variabel. Jika kita mengambil contoh code controller blog sebelumnya maka kita tinggal mengubah cara pemanggilan view menjadi

```
$out = $this->load->view("blog_view", $data, true);
```

Code diatas berarti kita akan menyimpan hasil view kedalam sebuah variabel.

Contohnya : ganti kode program blog2.php menjadi seperti berikut

```

application > controllers > Blog2.php > ...
1  <?php
2  defined('BASEPATH') or exit('No direct script access allowed');
3  class Blog2 extends CI_Controller
4  {
5      public function __construct()
6      {
7          parent::__construct();
8      }
9      function index()
10     {
11         // $data['judul'] = "Judul blog";
12         // $data['isi'] = "Isi blog";
13         // $this->load->view("blog_view", $data);
14         $data['judul'] = "Judul blog";
15         $data['isi'] = "Isi blog";
16         $out = $this->load->view("blog_view", $data, true);
17         echo $out;
18     }
19 }

```

Library pada Codeigniter

Secara default CodeIgniter telah menyediakan library yang dapat digunakan secara langsung. Adapun library yang telah tersedia antara lain:

- **Benchmarking Class**

Library ini digunakan untuk melakukan pengukuran terhadap aplikasi yang dibuat. Seperti untuk mengetahui berapa lama waktu eksekusi dan berapa jumlah memori yang digunakan. Library ini sudah digunakan dan di-load secara otomatis oleh CodeIgniter.

- **Calendar Class**

Library ini berfungsi untuk menampilkan dan men-generate kalender. • **Cart Class Library** ini berfungsi untuk membuat shopping cart (keranjang belanja). Library ini memiliki ketergantungan terhadap kelas session karena item-item chart tersebut disimpan di dalam session.

- **Config Class**

Library ini berfungsi untuk mengambil data-data di dalam file konfigurasi. Library ini sudah di load secara otomatis oleh CodeIgniter.

- **Database Class**

Library database digunakan untuk memanipulasi serta mendapatkan data dari sebuah sistem database. Secara default database yang sudah didukung oleh CodeIgniter adalah mysql, mssql, oracle, postgres. Sedangkan database yang tidak didukung secara langsung oleh CodeIgniter dapat di-jembatani dengan driver odbc.

- **Email Class**

Library email digunakan untuk mengirimkan email. Pengiriman email tersebut bisa dilakukan dengan menggunakan protokol mail, sendmail dan smtp. • Encryption Class Library Encryption digunakan untuk melakukan penyandian terhadap string tertentu

- **File Uploading Class**

Library Uploading digunakan untuk meng-upload file. Kelas ini sudah dilengkapi dengan pengecekan jenis file, dan ukuran file.

- **Form Validation Class**

Library form Validation digunakan untuk mengecek keabsahan form-form yang sudah di-submit oleh user.

- **FTP Class**

Library FTP digunakan untuk meng-upload atau download file melalui ftp server. • HTML Table Class Library HTML table adalah sebuah kelas yang berfungsi untuk men-generate table dari data array.

- **Image Manipulation Class**

Library image manipulation berfungsi untuk mengolah gambar. Adapun fungsi-fungsi yang telah disediakan adalah Image Resizing, Thumbnail Creation, Image Cropping, Image Rotating dan Image Watermarking.

- **Input and Security Class**

Library Input dan security berfungsi untuk menjamin bahwa inputan dari form telah bersih dari karakter-karakter “aneh”.

- **Loader Class**

Library ini dapat disebut sebagai pengatur sumberdaya CodeIgniter. Semua sumberdaya yang ada akan dikendalikan oleh kelas ini. Library ini sudah di-load secara otomatis oleh CodeIgniter.

- **Language Class**

Library language digunakan untuk mengatur bahasa apa yang akan dipakai oleh CodeIgniter.

- **Output Class**

Library Output bertujuan untuk meng-handle output dari CodeIgniter, mulai dari cache sampai ke profiling bisa dilakukan kelas ini.

- **Pagination Class**

Untuk mem-paginate hasil database untuk performance dan usability, kita bisa mengontrol berapa banyak record untuk ditampilkan disetiap halaman website, berapa banyak record untuk ditarik dari database dan tampilan dari bagian pagination

- **Session Class**

Library Session dapat digunakan untuk memelihara informasi status tentang user (seperti layaknya session di PHP). Tetapi Library ini tidak menggunakan session built-in dari PHP, Library Session men-generate session datanya sendiri yang disimpan di dalam Cookies.

- **Trackback Class**

Library Trackback digunakan untuk mengirim dan menerima data trackback.

- **Template Parser Class**

Library Template Parser digunakan untuk membuat template yang berisi parsable pseudo – templates.

- **Unit Testing Class**

Library Unit Testing digunakan untuk unit test function dalam aplikasi yang sedang dibuat. CodeIgniter menyediakan fungsi evaluasi dan dua fungsi hasil dalam library ini.

- **URI Class**

Library URI digunakan untuk memarsing URL, lalu memecahnya ke dalam beberapa segmen dan kemudian di-passing ke controller atau disimpan sebagai variabel.

- **User Agent Class**

Library User Agent digunakan untuk mengidentifikasi browser, mobile device, atau robot yang mengunjungi website. Kita juga bisa menggunakannya untuk mendeteksi dukungan bahasa, sekumpulan karakter, dan referrer.

- **XML-RPC Class**

Library XML-RPC digunakan untuk men-setup klien XML-RPC dan server.

- **Zip Encoding Class**

Library Zip Encoding digunakan untuk membuat file ZIP baik yang berjenis teks maupun data binary.

Contoh pemanggilan library codeigniter

Pada contoh kali ini kita akan menggunakan library table codeigniter

Ada dua cara untuk menambahkan library pada codeigniter

1. Menambahkan pada file autoload.php pada bagian libraries seperti berikut

```
51 $autoload['libraries'] = array('database', 'session', 'email', 'table');
```

2. Cara kedua yaitu menambahkan pada bagian function __construct yang terdapat pada file controller

Kali ini kita akan menambahkan library pada bagian file controller

Buat sebuah file controller dengan nama **table.php** pada **application/controller**

```

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

class Table extends CI_Controller
{
    function __construct()
    {
        parent::__construct();
        $this->load->library('table');
    }

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

```

Selanjutnya buat halaman view dengan nama **table_vw.php** pada **application/views**

```

application > views > table_vw.php > ...
<?php

$template = array(
    'table_open'      => '<table border="1" cellpadding="4" cellspacing="0">',
    'table_close'     => '</table>'
);

$this->table->set_template($template);
$this->table->set_heading(array('No', 'Nama', 'Alamat'));

$this->table->add_row(array('1', 'Budi', 'Rumbai'));
$this->table->add_row(array('2', 'Badu', 'Panam'));
$this->table->add_row(array('3', 'Bidi', 'Nangka'));

echo $this->table->generate();

```

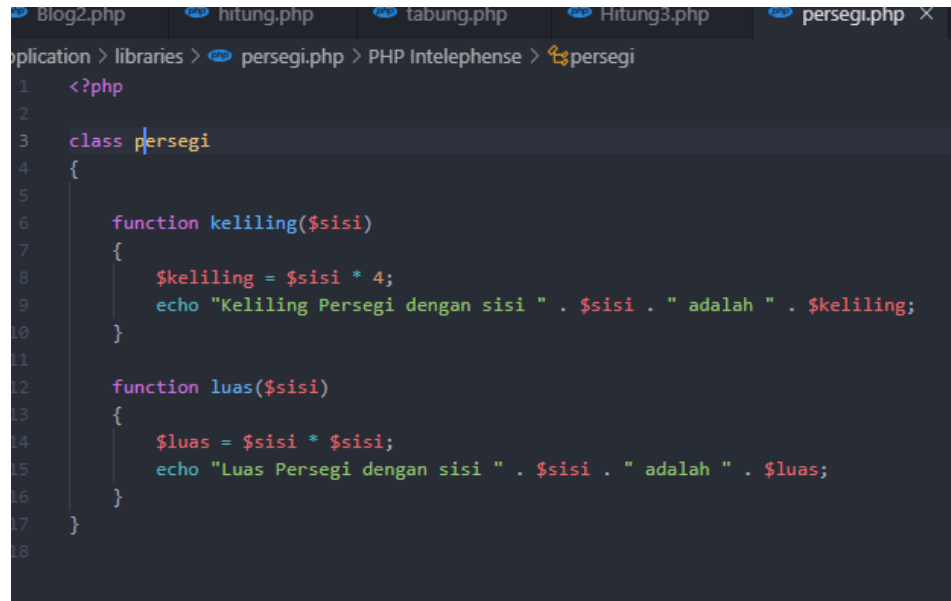
Saat dijalankan akan muncul tampilan seperti berikut

localhost/ardi_ci/table		
No	Nama	Alamat
1	Budi	Rumbai
2	Badu	Panam
3	Bidi	Nangka

Membuat Library Sendiri

Untuk membuat library di codeigniter, kita 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**

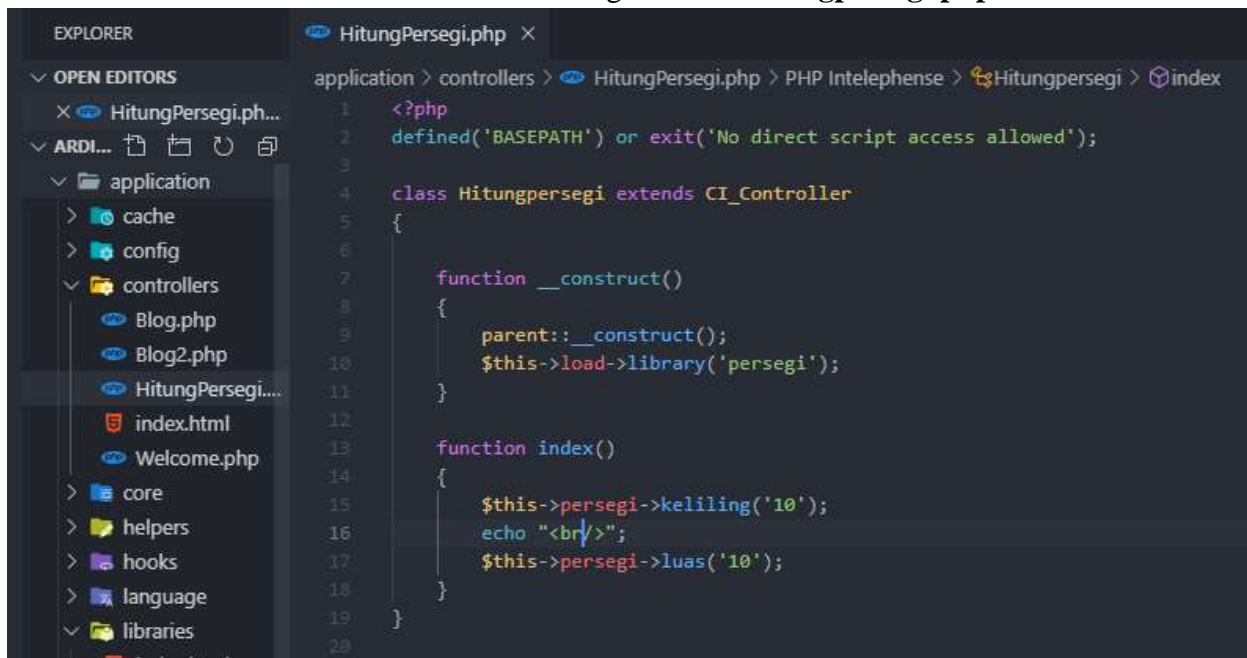


```

1  <?php
2
3  class Persegi
4  {
5
6      function keliling($sisi)
7      {
8          $keliling = $sisi * 4;
9          echo "Keliling Persegi dengan sisi " . $sisi . " adalah " . $keliling;
10     }
11
12     function luas($sisi)
13     {
14         $luas = $sisi * $sisi;
15         echo "Luas Persegi dengan sisi " . $sisi . " adalah " . $luas;
16     }
17 }
18

```

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



```

1  <?php
2  defined('BASEPATH') or exit('No direct script access allowed');
3
4  class Hitungpersegi extends CI_Controller
5  {
6
7      function __construct()
8      {
9          parent::__construct();
10         $this->load->library('persegi');
11     }
12
13     function index()
14     {
15         $this->persegi->keliling('10');
16         echo "<br/>";
17         $this->persegi->luas('10');
18     }
19 }
20

```

Saat di akses maka akan diperoleh hasil sebagai berikut

← → ↻ ⓘ localhost/ardi_ci/index.php/hitungpersegi

Keliling Persegi dengan sisi 10 adalah 40

Luas Persegi dengan sisi 10 adalah 100

Tugas 1

Silahkan buat library sendiri untuk perhitungan berikut

Volume Tabung (V)	$V = \pi \times r^2 \times t$
Luas Permukaan Tabung (L)	$L = 2 \times \pi \times r \times (r + t)$
Luas Selimut (Ls)	$Ls = 2 \times \pi \times r \times t$

Cukup dengan memanggil satu function yang sama, akan menghasilkan luas dan volume

← → ↻ ⓘ localhost/ardi_ci/index.php/hitung3

Phi : 3.14

Jari-jari 5

Tinggi 3

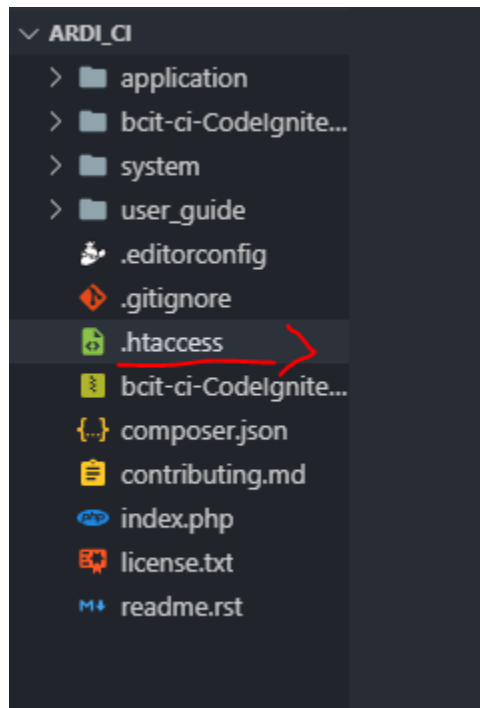
Volume Tabung adalah :235.5

Luas Permukaan Tabung adalah :471

Luas Selimut Tabung adalah :94.2

Mempercantik URL Codeigniter

Untuk mengubah url dari http://localhost/ardi_ci/index.php/hitung2 menjadi **http://localhost/ardi_ci/ /hitung** maka kita perlu menambahkan file baru dengan nama .htaccess, penempatan file sejajar dengan folder application



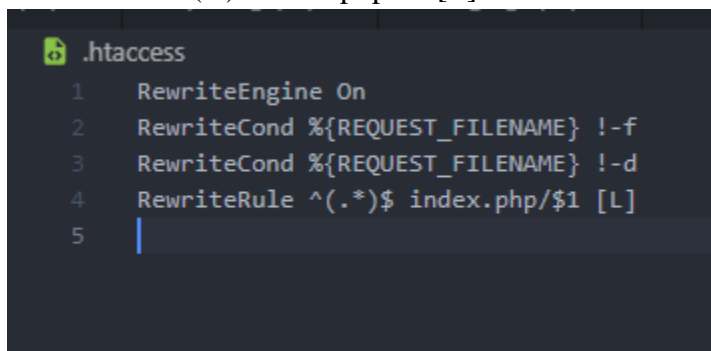
Selanjutnya tambahkan kode program berikut

RewriteEngine On

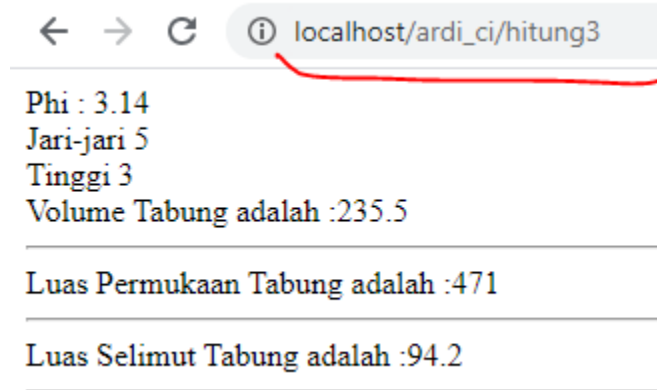
RewriteCond %{REQUEST_FILENAME} !-f

RewriteCond %{REQUEST_FILENAME} !-d

RewriteRule ^(.*)\$ index.php/\$1 [L]



Maka kita sudah dapat mengakses url tanpa menyertakan index.php seperti berikut



Silangkan dilengkapi pada laporan, sertakan analisa

Selamat Mengerjakan 😊