

LAPORAN PEMROGRAMAN WEB FRAMEWORK

“Konsep Router Pada CodeIgniter”



DISUSUN OLEH :

Hamdy Wahid

E31192285

GOLONGAN D

MANAJEMEN INFORMATIKA

TEKNOLOGI INFORMASI

POLITEKNIK NEGERI JEMBER

2021

BAB I

PENDAHULUAN

1.1 Latar Belakang

Perkembangan teknologi informasi telah banyak membantu pekerjaan setiap orang, lembaga atau perusahaan dalam menyelesaikan suatu masalah dan menjalankan tugasnya. Ketatnya persaingan teknologi menyebabkan peningkatan kebutuhan akan penggunaan teknologi informasi. Setiap orang, lembaga atau perusahaan yang ada saat ini selalu mengikuti perkembangan teknologi kebutuhan informasi untuk dapat membantu kegiatan operasionalnya dengan menggunakan sistem yang lebih efektif dan efisien dalam menjalankan bisnisnya. Salah satu kegiatan bisnis yang sedang populer saat ini adalah pembuatan dan penggunaan website untuk berbagai keperluan bagi setiap orang, lembaga atau perusahaan dalam mempromosikan penjualan ataupun kebutuhan bisnis dan manajerial.

Dalam perkembangan web Framework CodeIgniter banyak menawarkan kemudahan-kemudahan dalam membangun aplikasi website, karena Framework CodeIgniter sudah tersedia struktur aplikasi yang baik, coding yang standar, fungsi–fungsi dan library yang telah umum digunakan dalam pengembangan sistem. Dengan menggunakan Framework CodeIgniter pembangunan aplikasi dapat langsung fokus kepada business process yang dihadapi tanpa harus berfikir banyak masalah struktur aplikasi, standard coding dll.

1.2 Rumusan Masalah

- 1.2.1 Bagaimana konsep dasar Router pada CI?
- 1.2.2 Bagaimana membuat beberapa router pada CI?

1.3 Tujuan

- 1.3.1 Mahasiswa mampu memahami konsep dasar Router pada CI.
- 1.3.2 Mahasiswa mampu membuat beberapa router pada CI.

BAB II DASAR TEORI

2.1 Router di Codeigniter

Router pada framework codeigniter, memiliki tugas untuk menentukan controller serta method/fungsi yang akan dijalankan ketika pengguna aplikasi mengakses alamat/url tertentu. Ketika kita mengakses <http://localhost/ci> maka akan muncul tampilan welcome to ci, hal tersebut karena default controller mengarah pada file routers.php, anda bisa mengakses file routers.php didalam direktori application/config/routers.php.

Perhatikan pada line : 52 – 54

```
1 $route['default_controller'] = 'welcome';  
2 $route['404_override'] = '';  
3 $route['translate_uri_dashes'] = FALSE;
```

Keterangan :

- **\$route['default_controller'] = 'welcome'** ini merupakan pengaturan default controller yang otomatis akan dipanggil ketika halaman base_url web diakses, base url disini adalah alamat utama dari web, disitu kita menulis welcome artinya akan mengakses controller welcome, controller welcome adalah controller default yang merupakan bawaan codeigniter, untuk file controller berada di application\controllers, nah pada controller welcome, yang akan dijalankan awal adalah function index, pada function index tersebut menjalankan view welcome_message, dimana file view ini berada pada

direktori application\views, anda bisa mengganti nilai pada nilai default_controller, untuk mengarahkan ke controller tertentu saat base_url diakses

- **\$route['404_override'] = ''** merupakan pengaturan default controller yang akan diakses apabila halaman default controller tidak ditemukan, ataupun sebuah controller lainnya tidak ditemukan.
- **\$route['translate_uri_dashes'] = FALSE**, ini adalah pengaturan yang memperbolehkan anda menggunakan tanda dash (-) pada bagian url, anda bisa menggantinya dengan nilai TRUE, sebagai contoh semisal anda memiliki controller dengan nama produk_makanan maka kita dapat mengakses pada urlnya menjadi produk-makanan

Tampilan dari link : <http://localhost/ci/index.php/welcome/index> akan sama dengan tampilan dari link : <http://localhost/ci>. Hal tersebut dikarenakan saat kita mengakses : <http://localhost/ci/index.php/welcome/index> , kita sedang mengakses function index didalam controller welcome.

Jadi untuk mengakses function / method didalam controller, kita perlu menuliskan :

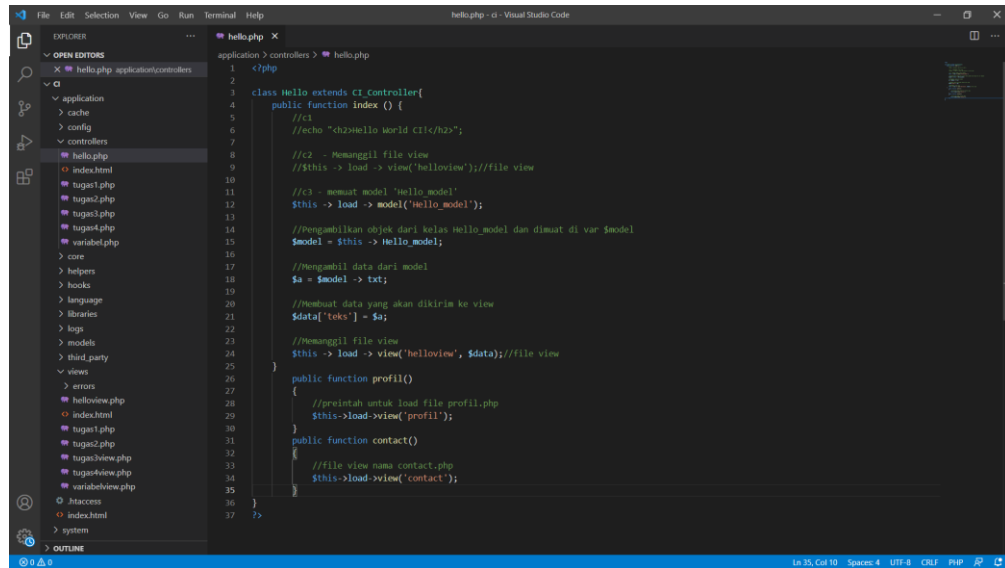
- Base_url : localhost/belajarcodigniter
- Lalu tambahkan index.php
- Lalu tuliskan nama controller
- Lalu tuliskan nama function

BAB III HASIL DAN PEMBAHASAN

3.1 Kegiatan Praktikum

3.1.1 Membuat Beberapa Router

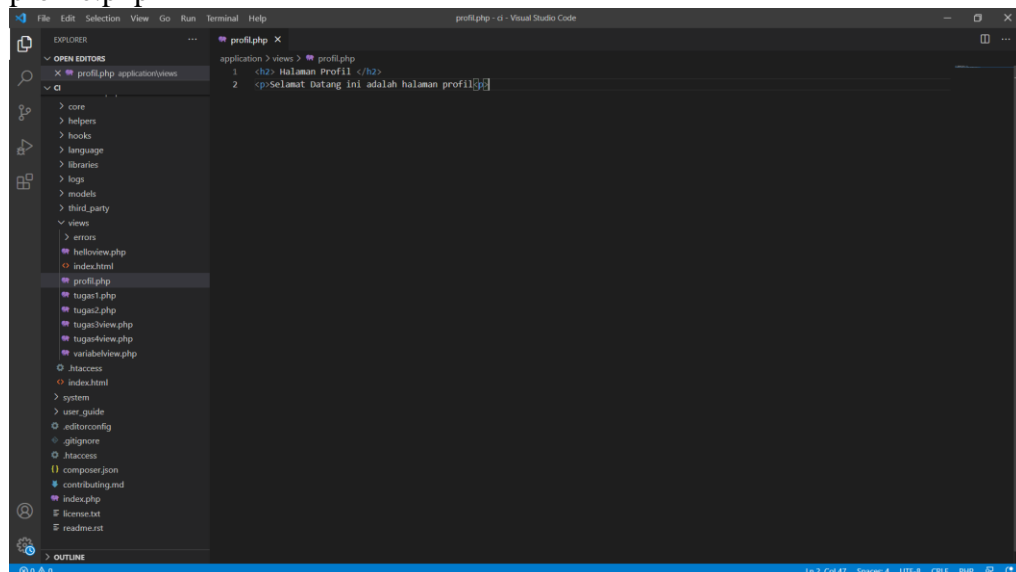
Membuat 2 function didalam controller Hello dibawah function index.



```
1 <?php
2
3 class Hello extends CI_Controller{
4     public function index () {
5         //c1
6         //echo "Hello world CI!</h2>";
7
8         //c2 - Memanggil file view
9         //$this->load->view('helloworld');//file view
10
11         //c3 - memuat model 'Hello_model'
12         //$this->load->model('Hello_model');
13
14         //Panggilan objek dari kelas Hello_model dan dimat di var $model
15         $model = $this->load->model('Hello_model');
16
17         //Panggil data dari model
18         $a = $model->txt;
19
20         //Panggil data yang akan dikirim ke view
21         $data['teks'] = $a;
22
23         //Memanggil file view
24         //$this->load->view('helloworld', $data);//file view
25     }
26
27     public function profil()
28     {
29         //Panggilan untuk load file profil.php
30         //$this->load->view('profil');
31     }
32
33     public function contact()
34     {
35         //file view nama contact.php
36         //$this->load->view('contact');
37     }
38 }
```

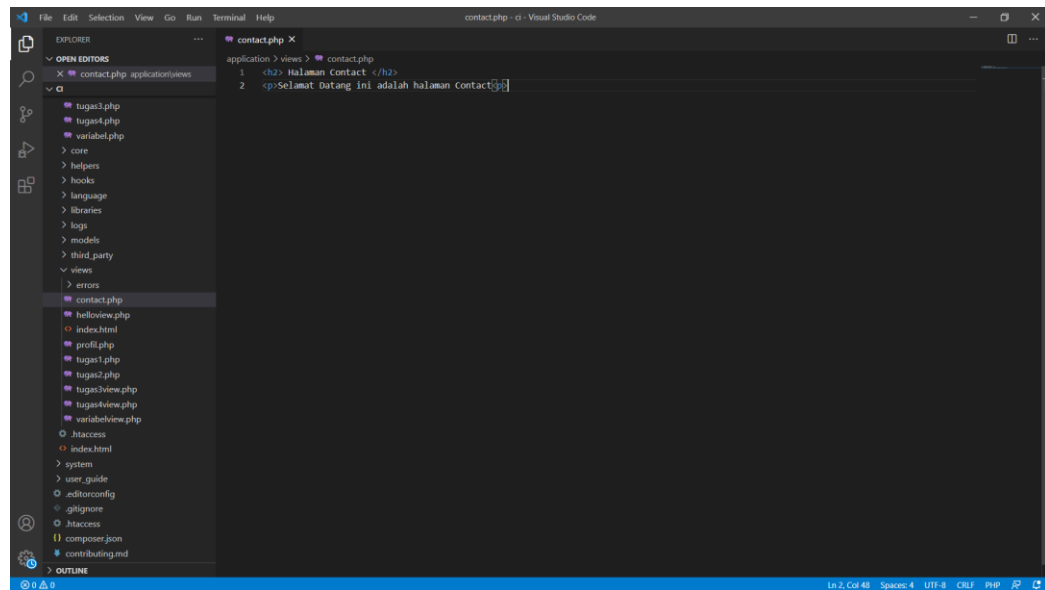
Berikutnya membuat 2 file view, dengan nama profil.php, dan contact.php didalam direktori application/views.

profile.php



```
1 <h2> Halaman Profil </h2>
2 <p>Selamat Datang ini adalah halaman profil!</p>
```

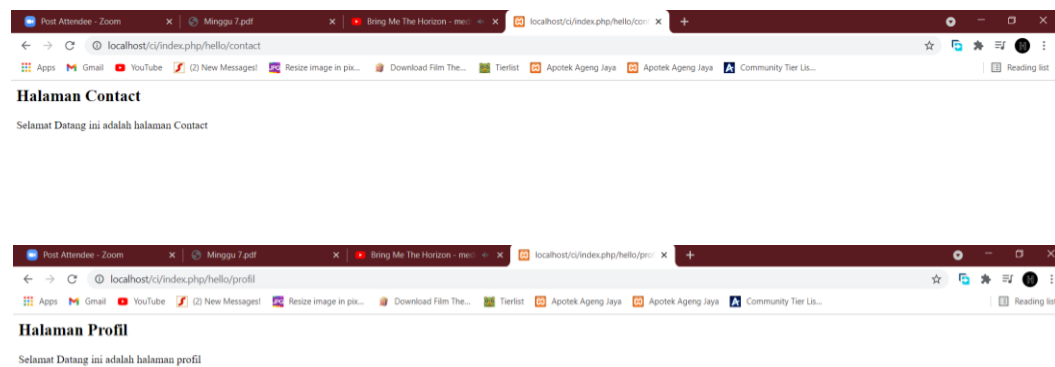
contact.php



```
application > views > contact.php
1  <h2> Halaman Contact </h2>
2  <p>Selamat Datang ini adalah halaman Contact</p>
```

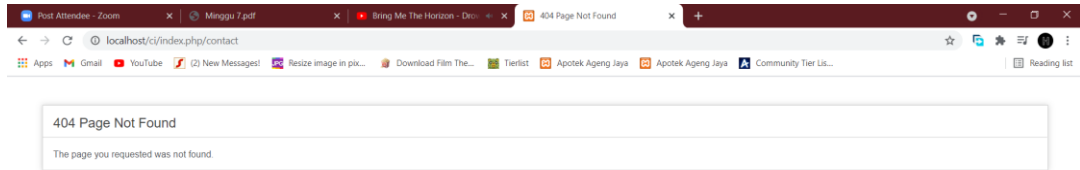
Output saat akses halaman tersebut dengan link :

<localhost/ci/index.php/welcome/contact> dan
<localhost/ci/index.php/welcome/profil>

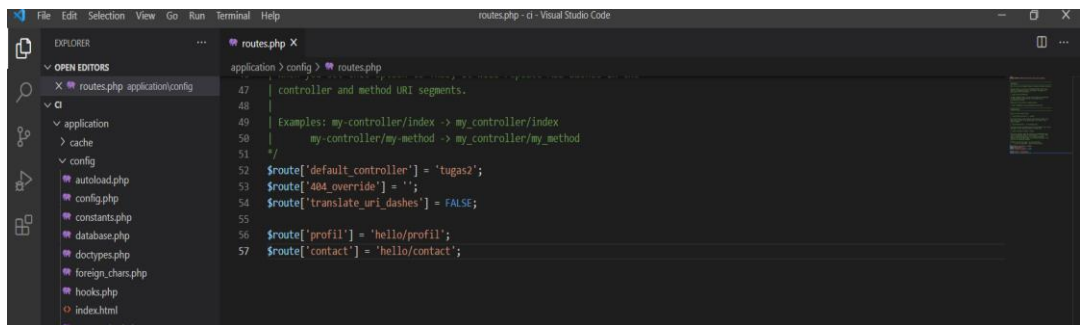


Halaman akan tampil, karena memang kita langsung mengakses function dalam controllerwelcome dibagian URL. Tetapi apakah bisa kita mengaksesnya dengan alamat :

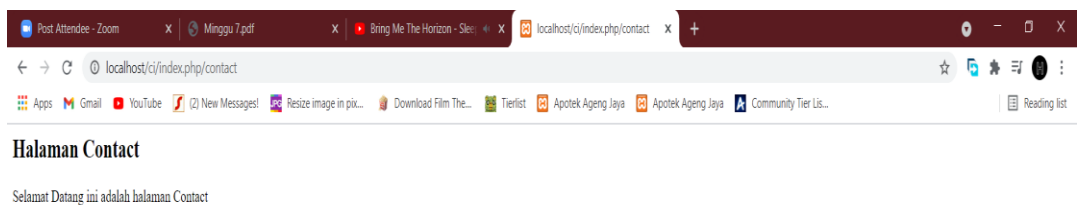
<localhost/ci/index.php/contact> dan <localhost/ci/index.php/profil>



maka akan muncul tampilan 404 not found. Hal ini dikarenakan dibagian router belum diset, sehingga perlu ditambahkan perintah dibagian di router.



Output setelah menambahkan perintah dibagian router :



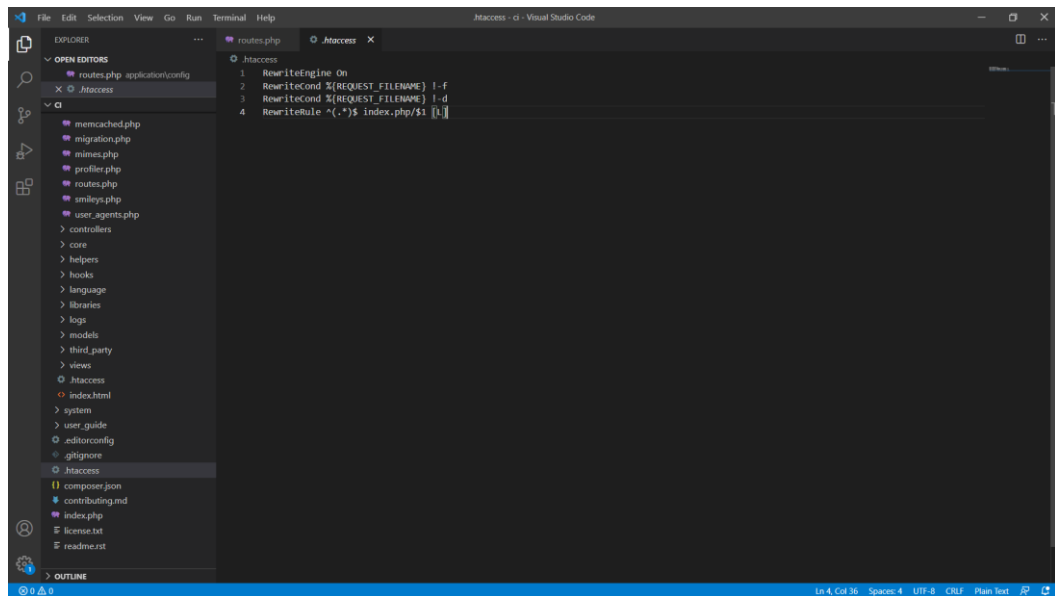
Kita tidak harus menambahkan route di file routers.php, setiap kali anda membuat route baru, karena Codeigniter otomatis mendeteksi route berdasarkan nama controller dan function/method yang dibuat.

3.2 Tugas / Soal

32.1 Untuk mengakses file pada CI normalnya link adalah :

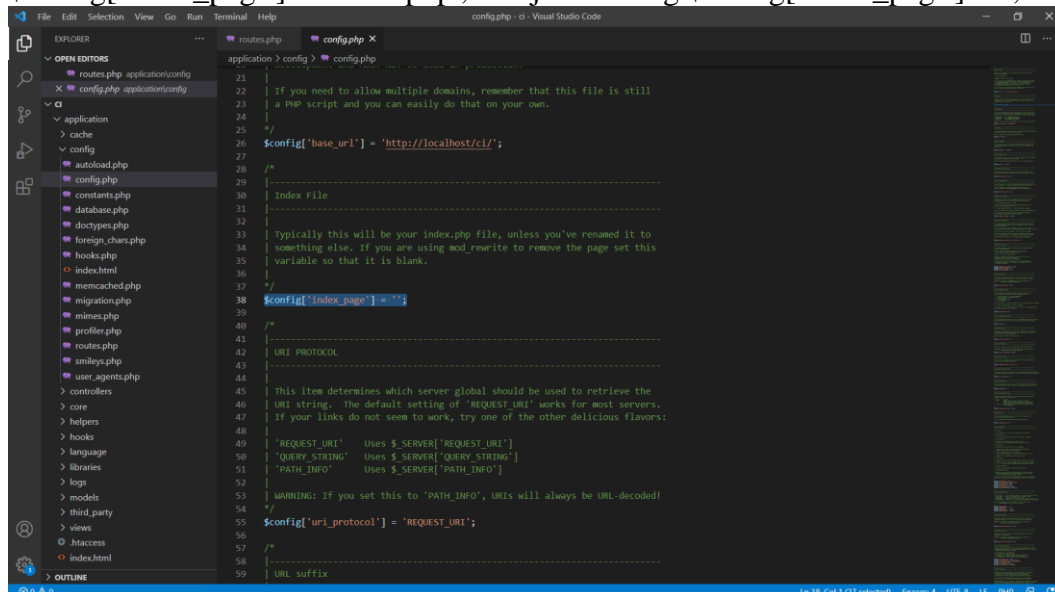
localhost/nama_folder/index.php/controller. Rubahlah agar aksesnya menjadi localhost/ci/controller.

Secara default, file index.php akan dimasukkan ke dalam URL kita akan tetapi kita dapat dengan mudah menghapus file ini menggunakan file .htaccess dengan beberapa aturan sederhana. Berikut adalah contoh file semacam itu, menggunakan metode "negatif" di mana semuanya dialihkan kecuali item yang ditentukan :



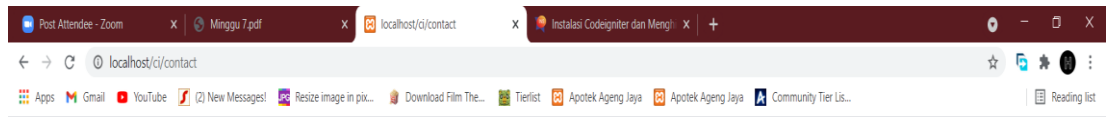
```
1 RewriteEngine On
2 RewriteCond %{REQUEST_FILENAME} !-f
3 RewriteCond %{REQUEST_FILENAME} !-d
4 RewriteRule ^(.*)$ index.php/$1 [L]
```

Ketika kita telah mengganti namanya menjadi sesuatu yang lain atau menggunakan mod_rewrite untuk menghapus halaman, Selanjutnya kita atur `$config['index_page'] = 'index.php';` menjadi kosong `$config['index_page'] = '';`



```
21 |
22 | If you need to allow multiple domains, remember that this file is still
23 | a PHP script and you can easily do that on your own.
24 |
25 |
26 $config['base_url'] = 'http://localhost/ci/';
27
28 /*
29 |-----|
30 | Index File
31 |-----|
32 |
33 | Typically this will be your index.php file, unless you've renamed it to
34 | something else. If you are using mod_rewrite to remove the page set this
35 | variable so that it is blank.
36 |
37 */
38 $config['index_page'] = '';
39
40 /*
41 |-----|
42 | URI PROTOCOL
43 |-----|
44 |
45 | This item determines which server global should be used to retrieve the
46 | URI string. The default setting of 'REQUEST_URI' works for most servers.
47 | If your links do not seem to work, try one of the other delicious flavors:
48 |
49 | 'REQUEST_URI'    Uses $_SERVER['REQUEST_URI']
50 | 'QUERY_STRING'   Uses $_SERVER['QUERY_STRING']
51 | 'PATH_INFO'      Uses $_SERVER['PATH_INFO']
52 |
53 | WARNING: If you set this to 'PATH_INFO', URIs will always be URL-decoded!
54 |
55 */
56 $config['uri_protocol'] = 'REQUEST_URI';
57
58 /*
59 |-----|
60 | URI SUFFIX
61 |-----|
```


Setelah kita menambahkan file .htaccess maka saat kita menuliskan pada URL untuk menjalankan programnya cukup dengan `http://localhost/ci/contact`



Halaman Contact

Selamat Datang ini adalah halaman Contact

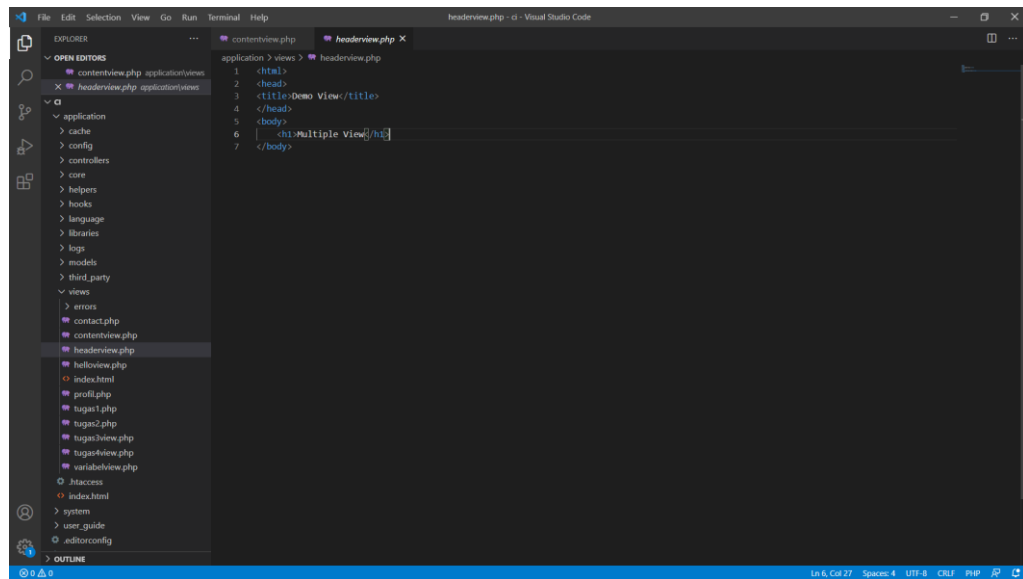
3.2.2 Cobalah kode dibawah ini :

Kode 1 :

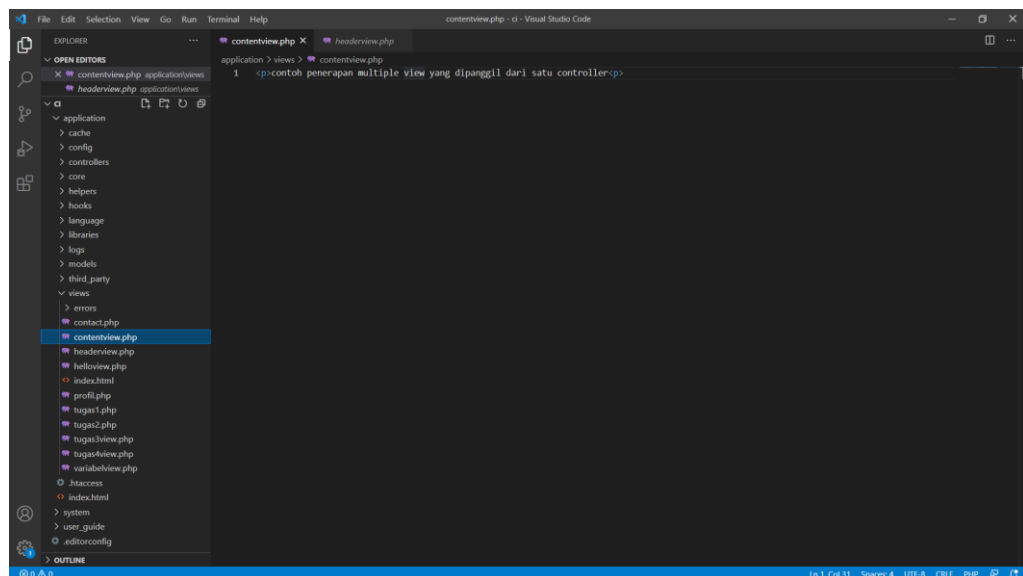
View

Load multiple view

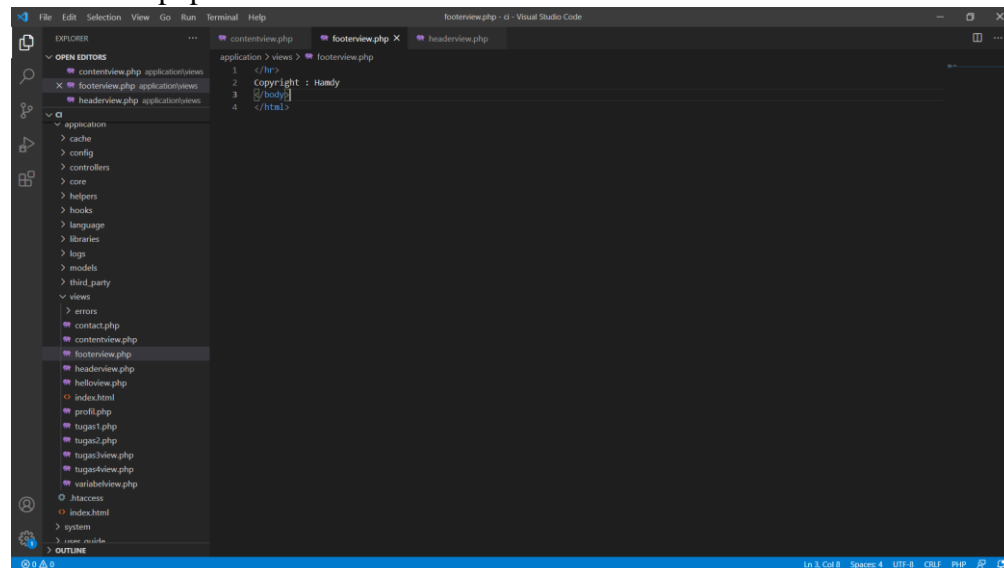
headerview.php



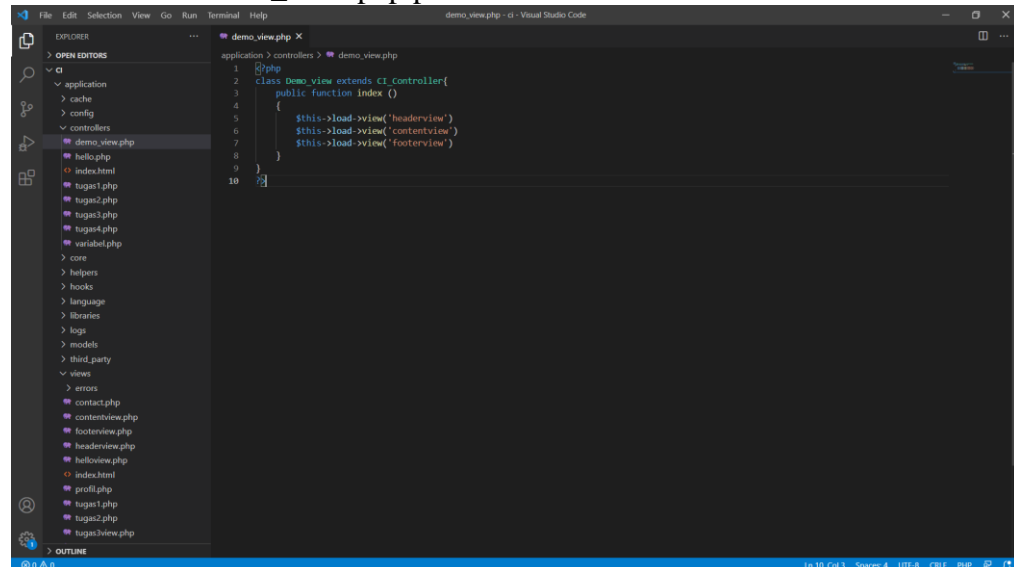
contentview.php



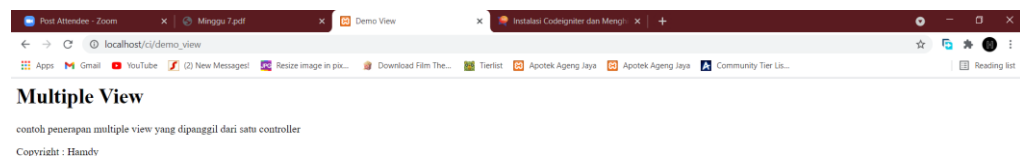
footerview.php



Membuat file Demo_view.php pada controller :



Output :

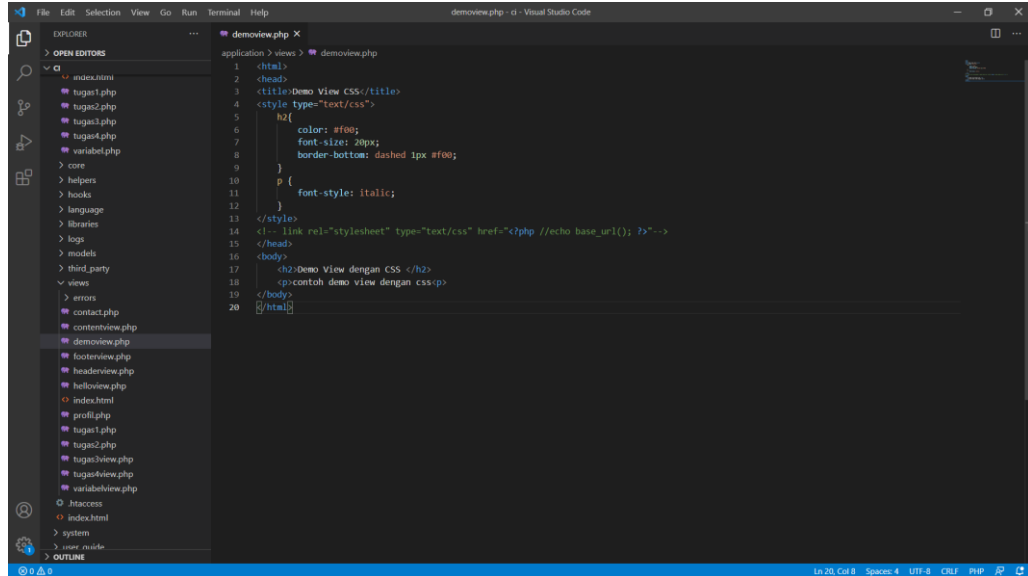


Sama halnya dengan tugas nomer 1 , setelah menambahkan file .htaccess maka pada saat akan menjalankan program kita hanya mengetikkan pada URL <http://localhost/ci/controller> tidak perlu menggunakan index.php lagi karena kita sudah hilangkan pada tugas sebelumnya dan berlaku juga pada tugas berikutnya selama berada pada folder codeigniter yang sama.

Kode 2 :

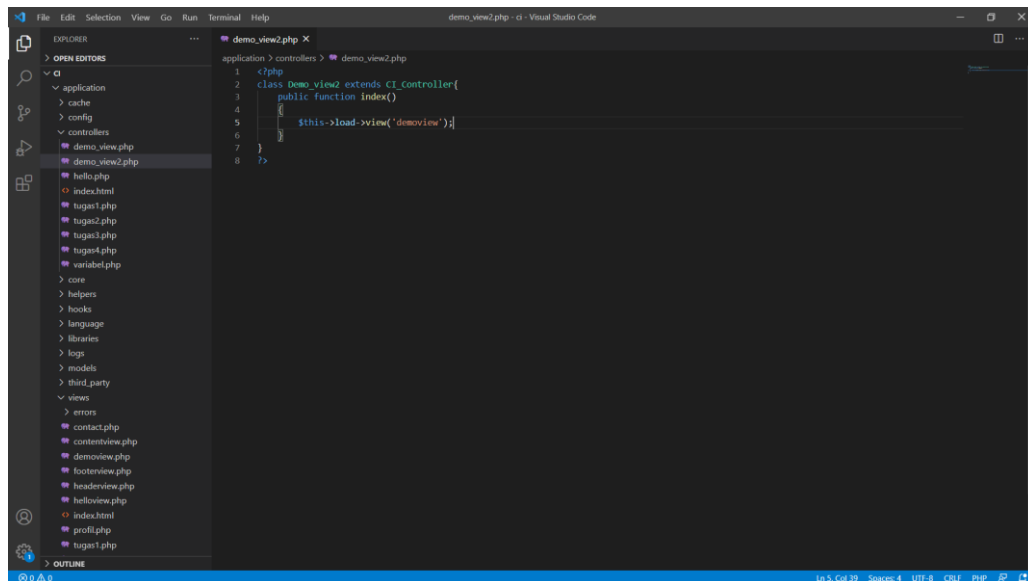
Menyisipkan CSS ke View 1

Demoview.php



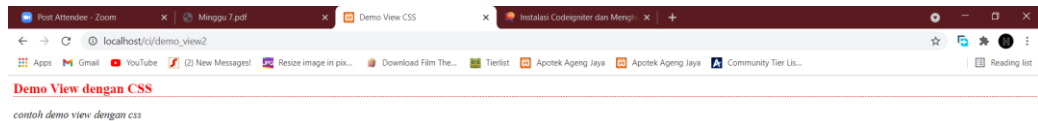
```
1 <html>
2 <head>
3 <title>Demo View CSS</title>
4 <style type="text/css">
5     h2{
6         color: #f00;
7         font-size: 20px;
8         border-bottom: dashed 1px #f00;
9     }
10    p {
11        font-style: italic;
12    }
13 </style>
14 <!-- link rel="stylesheet" type="text/css" href="<?php //echo base_url(); ?>"-->
15 </head>
16 <body>
17     <h2>Demo View dengan CSS </h2>
18     <p>contoh demo view dengan css</p>
19 </body>
20 </html>
```

Demo_view2.php



```
1 <?php
2 class Demo_view2 extends CI_Controller{
3     public function index()
4     {
5         $this->load->view('demoview');
6     }
7 }
8 >
```

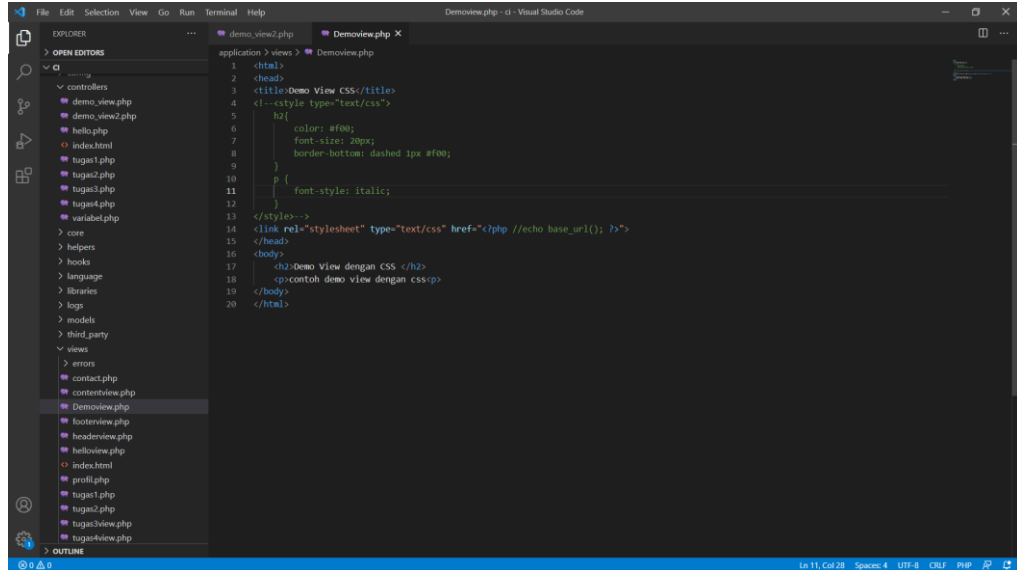
Output :



Kode 3 :

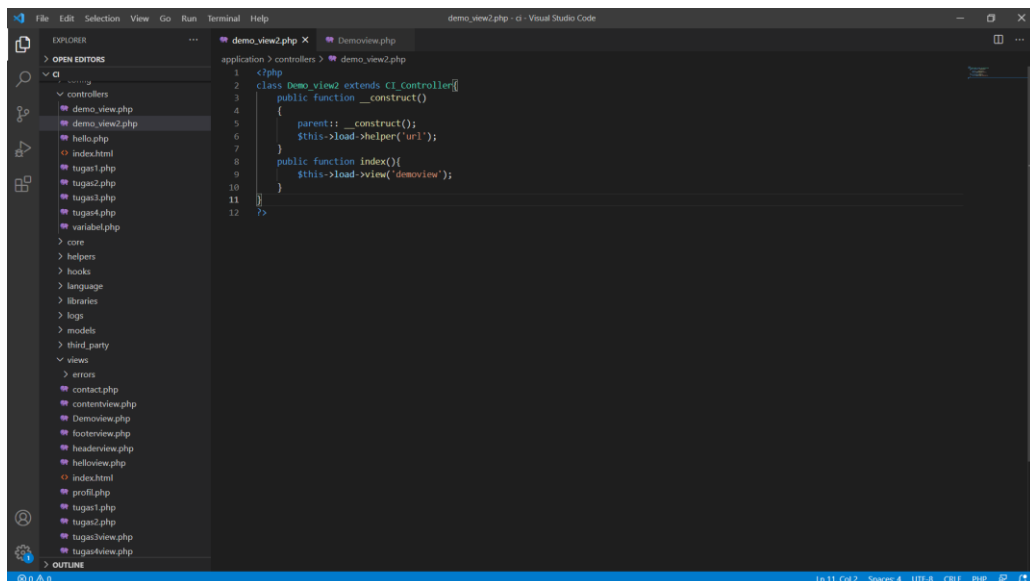
Menyisipkan CSS ke View 2

demoview.php



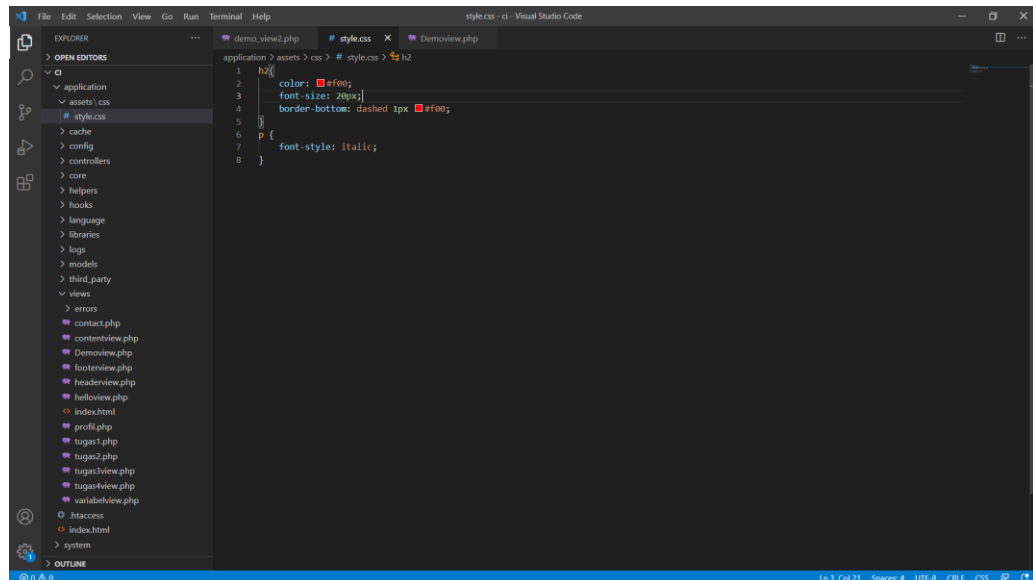
```
1 <html>
2 <head>
3 <title>Demo View CSS</title>
4 <!--style type="text/css"-->
5
6 <h2>
7     color: #f00;
8     font-size: 20px;
9     border-bottom: dashed 1px #f00;
10
11 <p>
12     font-style: italic;
13
14 </style-->
15 <link rel="stylesheet" type="text/css" href="<php //echo base_url(); ?>" />
16 </head>
17 <body>
18 <h2>Demo View dengan CSS </h2>
19 <p>contoh demo view dengan css</p>
20 </body>
21 </html>
```

Menambahkan function_construct di Demo_view2.php untuk menangkap perintah pada Demoview.php memanggil assets/css/style.css



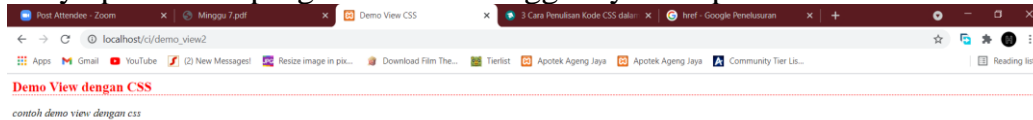
```
1 <?php
2 class Demo_view2 extends CI_Controller{
3     public function __construct()
4     {
5         parent::__construct();
6         $this->load->helper('url');
7     }
8     public function index(){
9         $this->load->view('demoview');
10     }
11
12 }
13 >
```

Sebelumnya saya telah membuat style.css yang diletakkan pada folder assets/css dengan menambahkan beberapa style dengan perintah sebagai berikut :



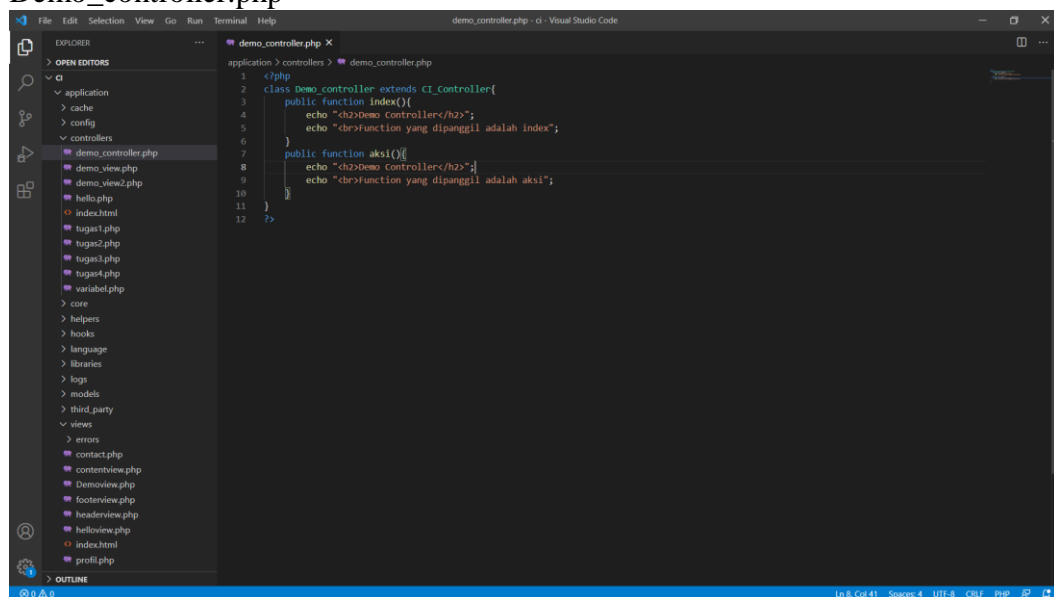
```
1 h2{
2   color: #f00;
3   font-size: 20px;
4   border-bottom: dashed 1px #f00;
5 }
6
7 {
8   font-style: italic;
9 }
```

Untuk output sama dengan style yang sebelumnya dikarenakan saya memasukkan perintah yang sama pada program , namun yang membedakan hanya pada kode program ini memanggil style.css pada folder assets/css



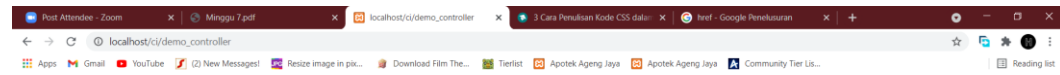
Controller

Demo_controller.php

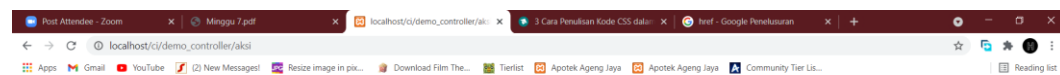


```
1 <?php
2 class Demo_controller extends CI_Controller{
3   public function index(){
4     echo "<?Demo Controller/h2>";
5     echo "<br>function yang dipanggil adalah index";
6   }
7   public function aksi(){
8     echo "<?Demo Controller/h2*>";
9     echo "<br>function yang dipanggil adalah aksi";
10  }
11 }
12 ?>
```

Pada method Demo_controller terdapat 2 function yakni index dan aksi yang dapat kita run dengan mengetikkan URL `http://localhost/ci/demo_controller/index` ataupun juga bisa dengan `http://localhost/ci/demo_controller` untuk menampilkan function index dan `http://localhost/ci/demo_controller/aksi` untuk menampilkan function aksi.



Function yang dipanggil adalah index

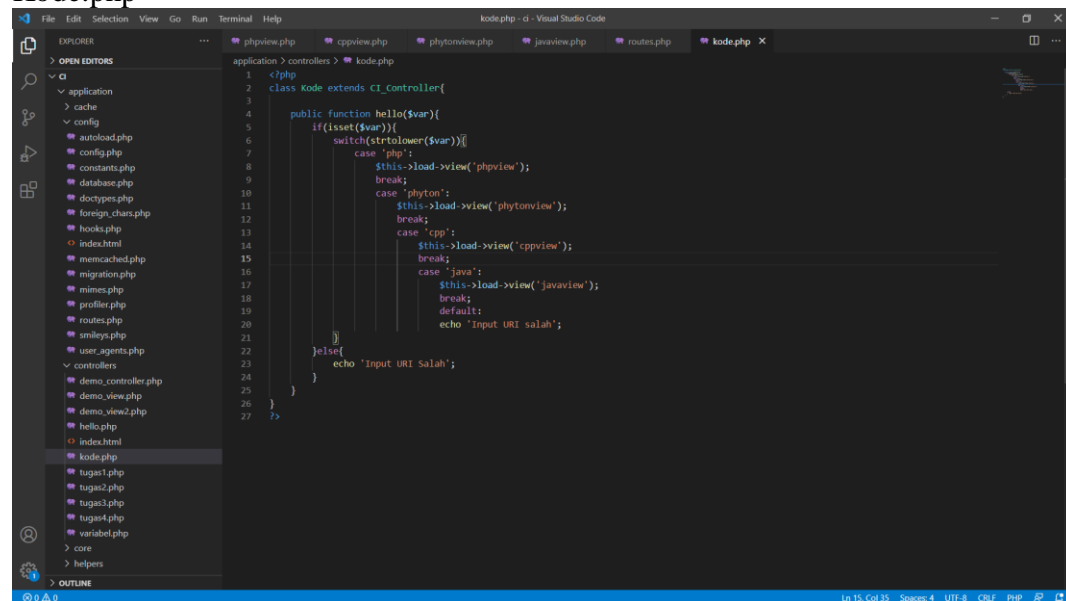


Demo Controller

Function yang dipanggil adalah aksi

Melewatkan segment URI kedalam metode

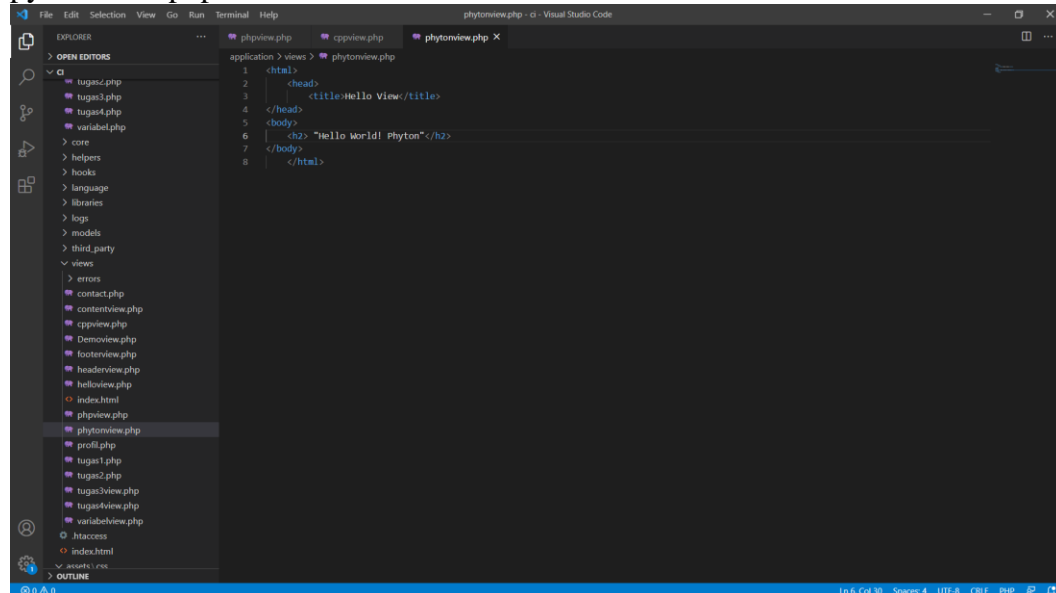
Kode.php



phpview.php

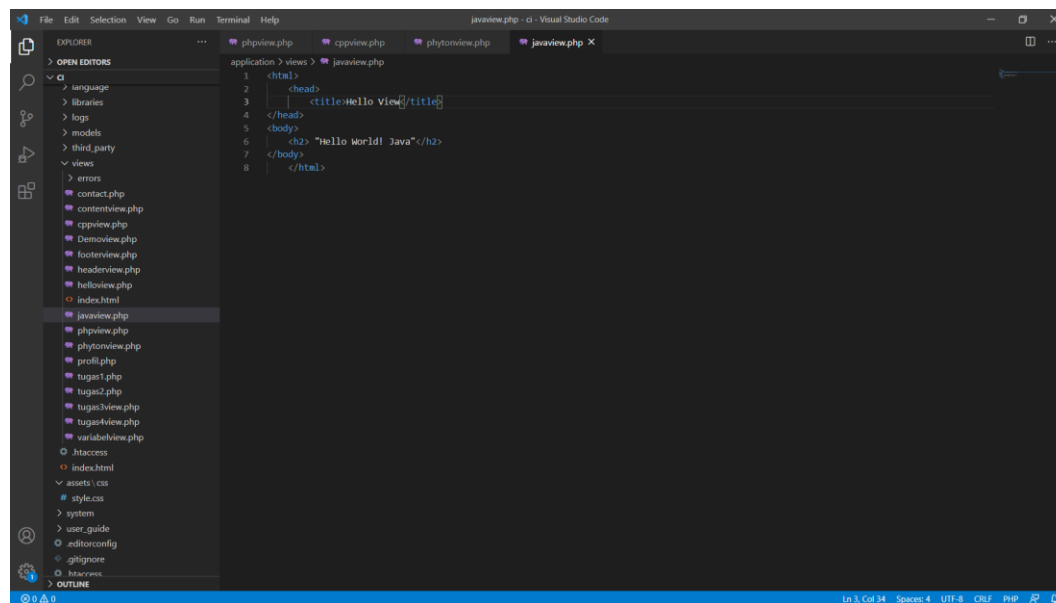
cppview.php

pythonview.php



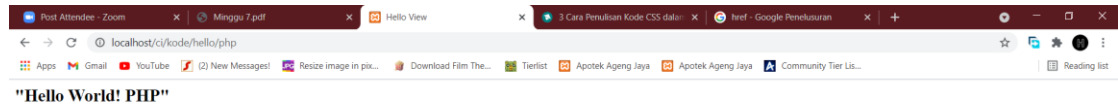
```
1 <html>
2   <head>
3     <title>Hello View</title>
4   </head>
5   <body>
6     <h2> "Hello World! Python"</h2>
7   </body>
8 </html>
```

javaview.php

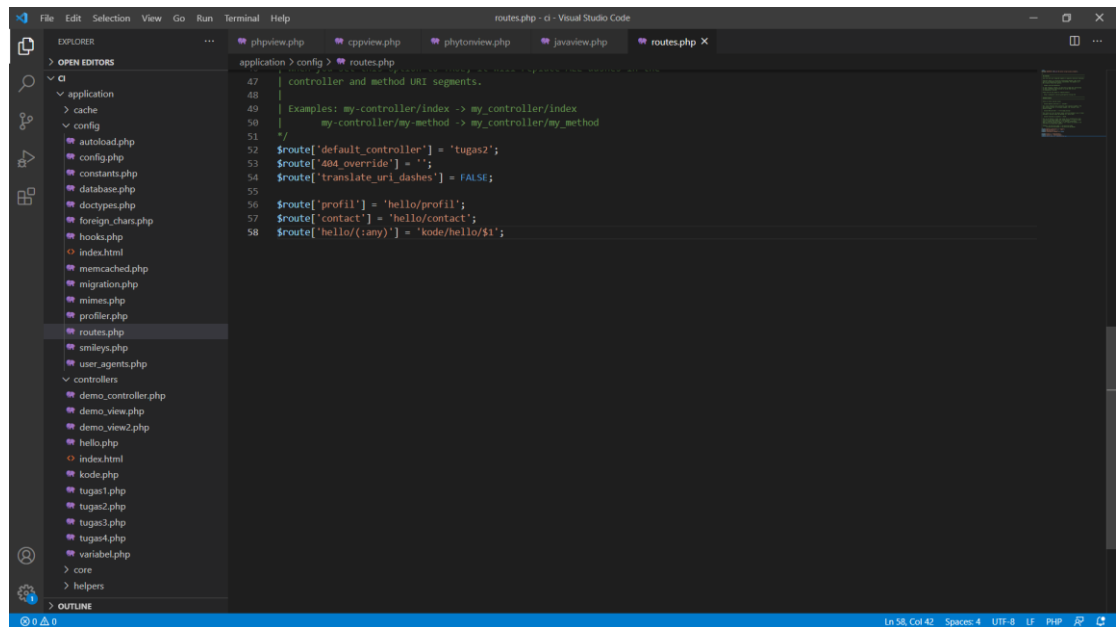


```
1 <html>
2   <head>
3     <title>Hello View</title>
4   </head>
5   <body>
6     <h2> "Hello World! Java"</h2>
7   </body>
8 </html>
```

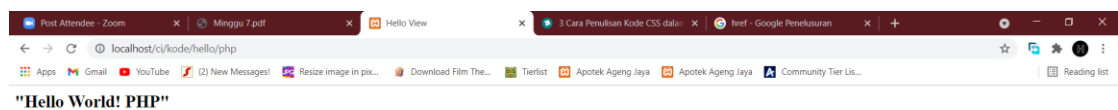
Segmen di URL, mengikuti pendekatan Model-View-Controller, biasanya mewakili: `http://localhost/ci/kode/hello/php` . Segmen pertama mewakili kelas pengontrol yang harus dipanggil. Segmen kedua mewakili fungsi kelas, atau metode, yang harus dipanggil. Segmen ketiga, dan setiap segmen tambahan, mewakili ID dan variabel apa pun yang akan diteruskan ke pengontrol.



Library URI dan URL Helper berisi fungsi yang memudahkan untuk bekerja dengan data URI kita. Selain itu, URL kita dapat dipetakan ulang menggunakan fitur perutean URI agar lebih fleksibel. Sehingga saya menambahkan kode program pada route sebagai berikut :

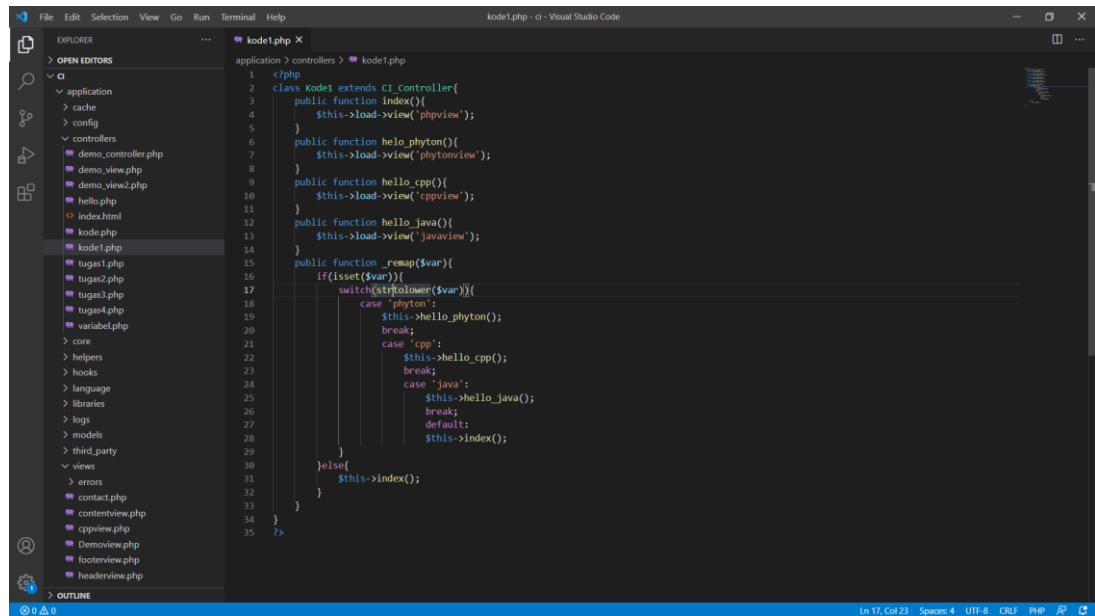


Pada routes.php saya menambahkan `$route['hello/(:any)'] = 'Kode/hello/$1';` yang arti dari program tersebut adalah ketika kita mengetikkan pada URL `http://localhost/ci/hello/(apapun yg user ketikkan)` maka akan diarahkan ke controller Kode dan methodnya hello.



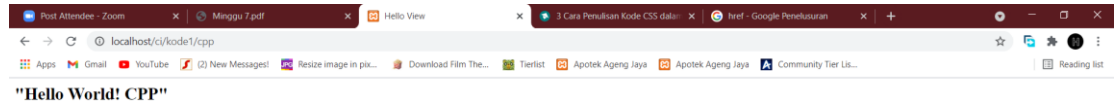
Memetakan nama metode yang akan dipanggil

Kode1

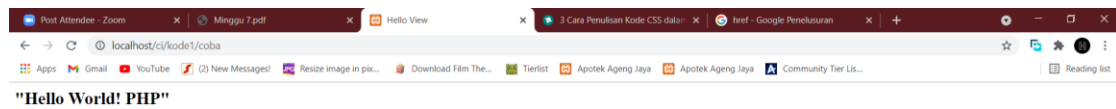


```
1 <?php
2 class Kode1 extends CI_Controller{
3     public function index(){
4         $this->load->view('phpview');
5     }
6     public function hello_python(){
7         $this->load->view('pythonview');
8     }
9     public function hello_cpp(){
10        $this->load->view('cppview');
11    }
12    public function hello_java(){
13        $this->load->view('javaview');
14    }
15    public function _remap($var){
16        if(isset($var)){
17            switch(strtolower($var)){
18                case 'python':
19                    $this->hello_python();
20                    break;
21                case 'cpp':
22                    $this->hello_cpp();
23                    break;
24                case 'java':
25                    $this->hello_java();
26                    break;
27                default:
28                    $this->index();
29            }
30        }else{
31            $this->index();
32        }
33    }
34 }
35 >>
```

Penjelasan pada program diatas adalah apabila kita menuliskan apapun setelah method Kode1 , disini saya mencontohkan <http://localhost/ci/kode1/cpp> maka program akan menangkap function _remap dan kemudian dilanjutkan ke function index untuk mengarahkan view mana yang akan ditampilkan pada output.



Namun apabila variabel yang kita tuliskan tidak ada dalam function `_remap` maka program akan menampilkan halaman index.



BAB IV

PENUTUP

4.1 Kesimpulan

URL routing (route) adalah salah satu metode yang digunakan untuk memetakan URL ke dalam sumber daya tertentu dengan memberikan nama lain dari alamat sumber daya yang dimaksud. URL routing sering digunakan untuk beberapa hal seperti menjadikan URL sumber daya yang sulit dibaca manusia dengan membuat pemetaan baru ke URL alias dari route yang lebih mudah dibaca manusia, membuat URL sumber daya menjadi lebih pendek dengan memberikan penamaan routing yang lebih pendek, dan memantau agar URL sesuai dengan format yang diinginkan dengan memanfaatkan fungsi regular expression (regex).

DAFTAR PUSTAKA

- Basuki, AP. 2010. *Membangun Web Berbasis PHP dengan Framework Codeigniter* : Yogyakarta . Lokomedia.
- Sirenden, Bernadus Herdi dan Ester Laekha Dachi. 2012. *Buat Sendiri Aplikasi Petamu Menggunakan CodeIgniter dan Google Maps API*. Yogyakarta: Andi Offset.