

CHAPTER III

PROBLEM ANALYSIS

III.1 Definition of Framework Code Igniter

Code Igniter is one of PHP programming language framework. The main aim why developer using framework is because with framework developer can finish their web application faster than using PHP native. Code Igniter using MVC pattern, MVC is one of application design patterns that usually used by developer to make the works faster. Rather than using PHP native, Developer can use Code Igniter to finish their work faster than not use any framework. The MVC design pattern is such a good fit for web application development because they combine several technologies usually split into a set of layers. Also, MVC specific behavior could be to send specific views to different types of user-agents.

III.2 Code Igniter Controller Scope

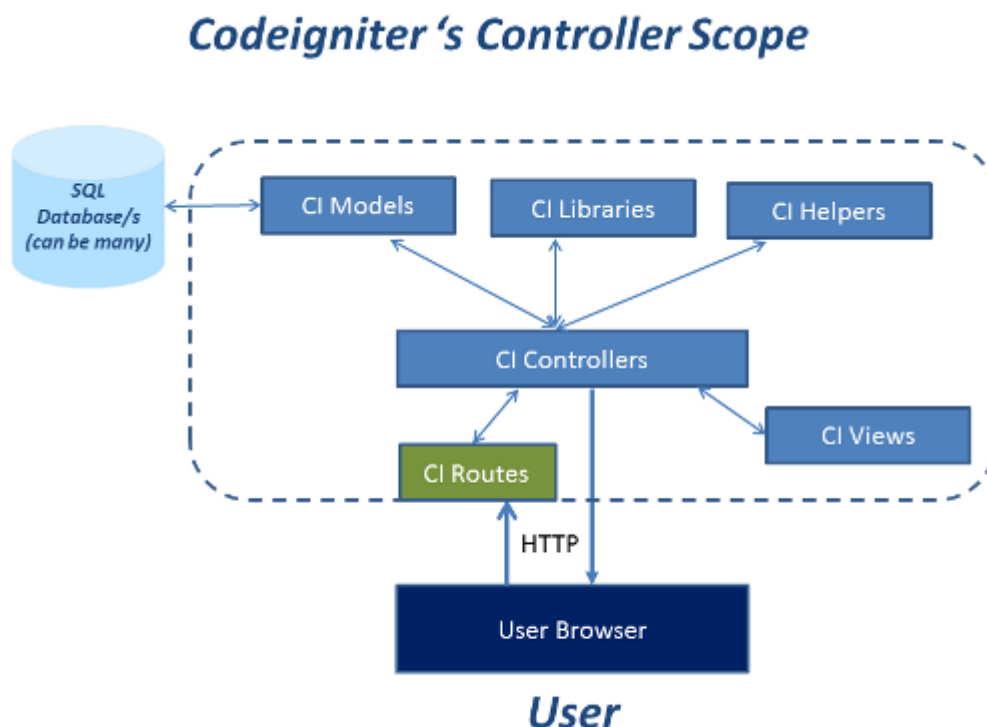


Figure 3.0 Code Igniter's Controller Scope

Source : www.packtub.com

Code Igniter has a complex building. That's the reason why the security level in the Code Igniter is better than using PHP Native. MVC Pattern connected with each other like the picture above. It show the Controller are the main processor of the program because Controller is the main function that display all of logical function. Without Controller, any libraries, model, and views can't be shown to the user. Several relationship will be describe below.

Description of each parts, there are :

1. Model

Model is the part of the Code Igniter system that manages all query related to data. Model usually used to code the query of CRUD (Create Read Update Delete) from database. The Model is responsible of all query database application.

2. View

View is the part of Code Igniter system that display the web. View used to display the template that created by Developer. It also display the code of query. As example we want to display product of today. Result of the code can be seen at View folder.

3. Controller

Controller is the part of Code Igniter system that manage all of logical function related to Model and View. Controller being a bridge between Model and View, because Controller has a function to manage the code which show the result of logical function. Controller also be a place to process the algorithm to show the result at the view folder.

4. Libraries

Libraries is the part of Code Igniter system that manage the libraries of application. The function of libraries in Code Igniter is same like libraries in Java. Libraries will help the application development. The example is when application need a template so the view of Code Igniter using dynamic template and doesn't need a lot of change in one view.

5. Helpers

Helpers is the part of Code Igniter which has the function to help the developer develop the program. The Helper will transform the text into the format of Helper. As example the original format text of Date Time from database is 2016-02-11 24:10:23. The text can be changed into Tuesday, November 23 2016 12:10 PM. Also text of amount from database is 200000 we can change it into \$2000,00.

Helpers is the great feature of Code Igniter that help development of text from database.

6. Routes

Routes is the bridge of Code Igniter codes to help user display the link that requested. The Function as the bridge of application. If the link are not written in routes they can't display the link that requested by user.

7. SQL Database

SQL Database are default database of Code Igniter. We can use MySQL as the default database of our program. SQL Database will stored any data that stored in the database. Most of Developer using MySQL as the default database when develop web application using Code Igniter.

8. User Browser

User Browser are the display of the application. User can see the application through user browser. Every code that written in application will displayed at User Browser. User Browser also is the place of user interaction with the application such as requesting the data and store some information.

Relationship between each part, there are :

1. Model – SQL Database

Model as the query function will call data appropriate the query code. As example, if the code are display the data. The database will bring the query request, that's data.

2. Controller – Libraries

Controller as the logical function need libraries to make the program better. As example if we want use one template to some of template to make the program easy to code, we can using template library to make the template. The function of libraries is same like common libraries, to make developer easier when making the program with the help of library.

3. Controller – Helper

Helper is a logical function that used by developer to make the code to use in some of views file. We can change the format date from 12//12/2016 into Tuesday, December 12 2016 with helper file. The objective is to be a template of code and developer can call it when needed. The perspective are same like Object Oriented Programming.

4. Controller – Routes – User Browser

Routes has a function as a bridge between the logical and user browser. Its mean the logical function that displayed to the user will going through a bridge called Routes. Routes also being a link of HTTP in user browser.

More complex about the architecture is when user want to see the web, they will input the address, after user input it will call the routes to check are the link is available at the controller. Controller will check the libraries and helper which needed in the view. Controller will call the libraries and helper, and also the model to get the data. Model will check data from query code that called by controller and return the data to the controller, after that the data will be displayed to the user via routes.

III.3 How to Use Code Igniter

There are simple steps to using Code Igniter, there are :

1. Turn on The XAMPP

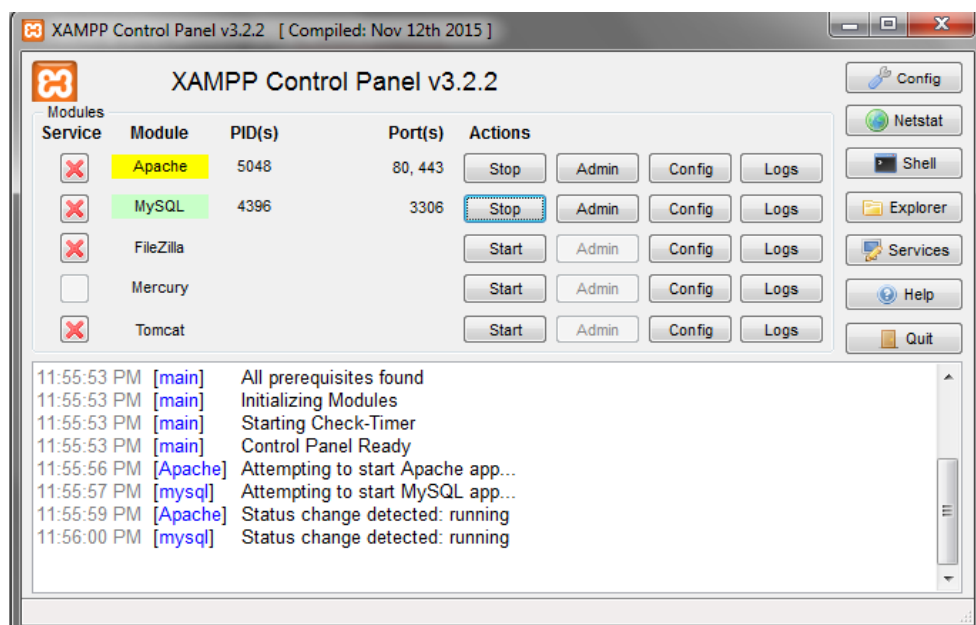


Figure 3.1 XAMPP Server

2. Make a folder of Code Igniter

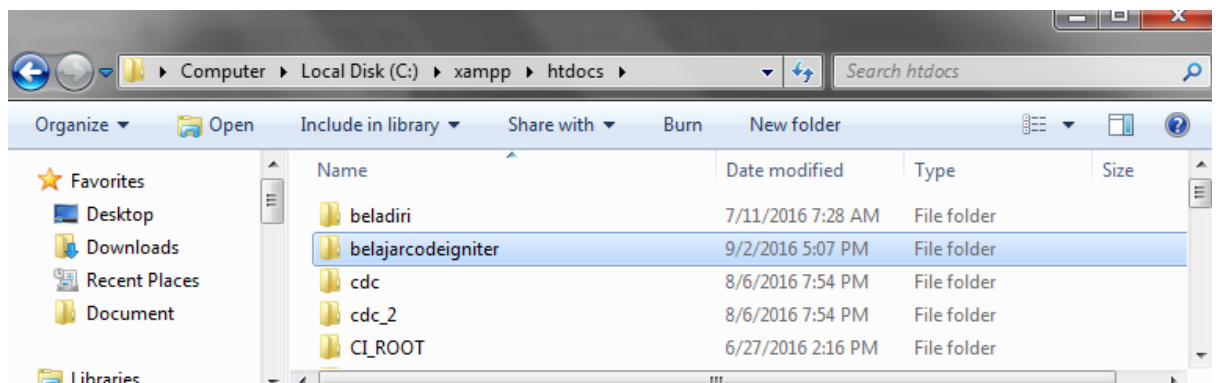


Figure 3.2 Folder of Application

3. Setting the base URL at Config.php

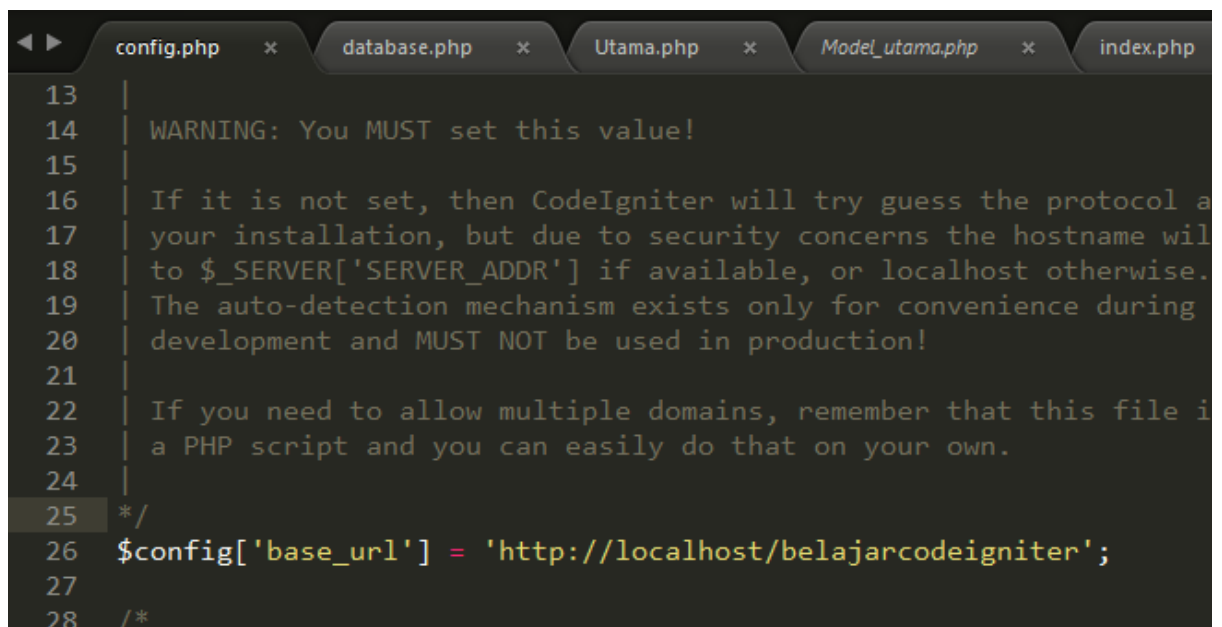
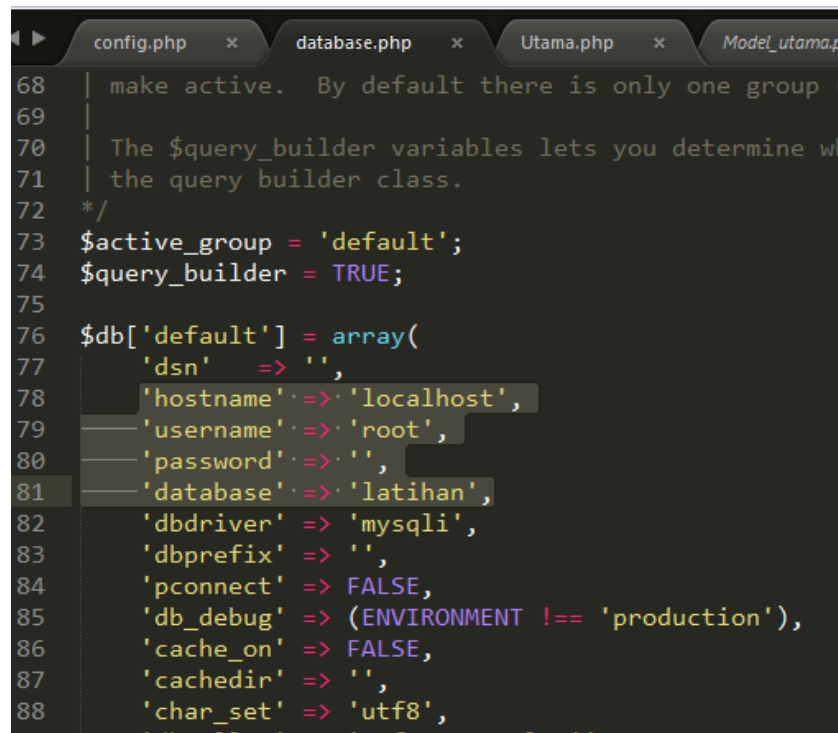


Figure 3.3 Setup of Config.php

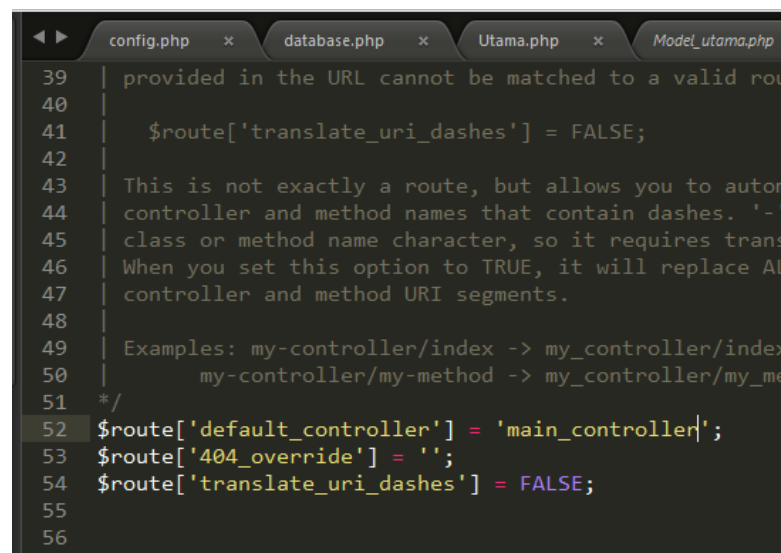
4. Setup the database configuration



```
68 | make active. By default there is only one group (
69 |
70 | The $query_builder variable lets you determine wh
71 | the query builder class.
72 | */
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' => 'latihan',
82 |     'dbdriver' => 'mysqli',
83 |     'dbprefix' => '',
84 |     'pconnect' => FALSE,
85 |     'db_debug' => (ENVIRONMENT !== 'production'),
86 |     'cache_on' => FALSE,
87 |     'cachedir' => '',
88 |     'char_set' => 'utf8',
89 |     'collat' => 'utf8_general_ci',
90 |     'swap_pre' => 'db_',
91 |     'encrypt' => FALSE,
92 |     'compress' => FALSE,
93 |     'strict' => FALSE,
94 |     'foreign_key' => TRUE,
95 |     'auto_increment_on_insert' => TRUE,
96 |     'charset' => 'utf8',
97 |     'collation' => 'utf8_general_ci',
98 |     'table_prefix' => ''
99 | );
```

Figure 3.4 Setup of Database Configuration

5. Setup the routes to set the Main Controller



```
39 | provided in the URL cannot be matched to a valid route
40 |
41 | $route['translate_uri_dashes'] = FALSE;
42 |
43 | This is not exactly a route, but allows you to automatically
44 | controller and method names that contain dashes. '-'
45 | class or method name character, so it requires translating
46 | When you set this option to TRUE, it will replace all dashes
47 | controller and method URI segments.
48 |
49 | Examples: my-controller/index -> my_controller/index
50 |           my-controller/my-method -> my_controller/my_method
51 | */
52 | $route['default_controller'] = 'main_controller';
53 | $route['404_override'] = '';
54 | $route['translate_uri_dashes'] = FALSE;
55 |
56 |
```

Figure 3.5 Routes Setup

6. Setup the model to get the data.

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

class main_model extends CI_Model {

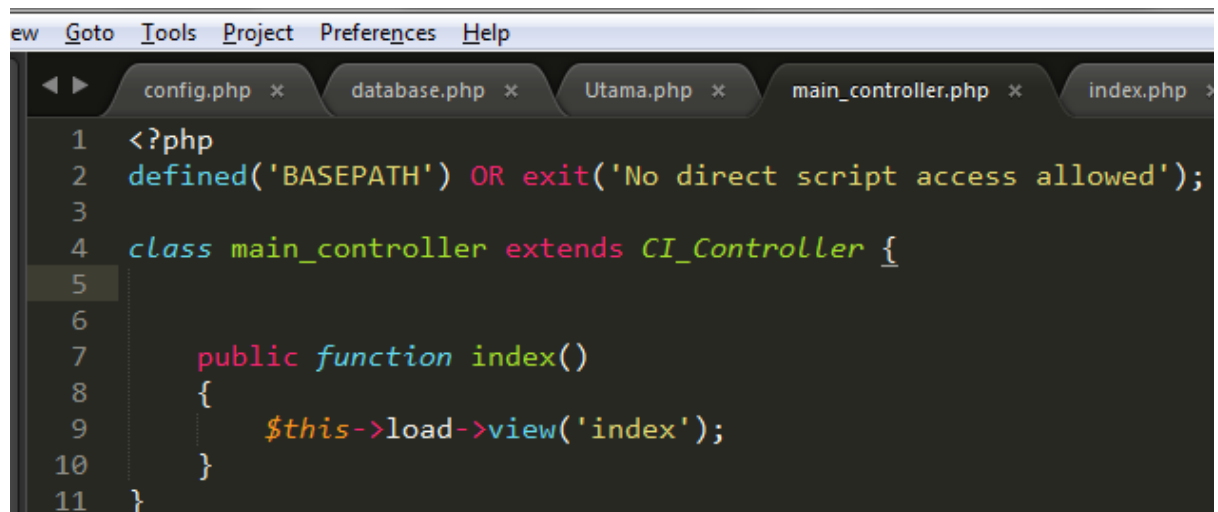
    public function getData(){
        $q = "select * from data";
        return $this->db->query($q);
    }

}

?>
```

Figure 3.6 Model Setup

7. Setup the Controller to make function to Index.



```
ew Goto Tools Project Preferences Help
config.php x database.php x Utama.php x main_controller.php x index.php x
1 <?php
2 defined('BASEPATH') OR exit('No direct script access allowed');
3
4 class main_controller extends CI_Controller {
5
6
7     public function index()
8     {
9         $this->load->view('index');
10    }
11 }
```

Figure 3.7 Controller Setup

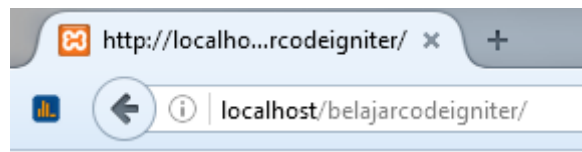
8. Make index.php at view folder.



```
w Goto Tools Project Preferences Help
database.php Utama.php x main_controller.php x main_model.php x Model_utama.php x index.php x
1 <?php
2
3 echo "Welcome to Web Learning Code Igniter";
4
5 ?>
```

Figure 3.8 View of Index Page

9. Go to [http://localhost/\[yourbaseurl\]](http://localhost/[yourbaseurl]) and you can access your web application



Welcome to Web Learning Code Igniter

Figure 3.9 Index Page of Application