NAIT

Edmonton, Alberta

Building A Web Camera Application With CodeIgniter MVC framework

As a submission to

Mr. Kelly Shepherd, Instructor

English and Communications Department

Submitted by

Xiao Liu, Student

Computer Engineering Technology

Dec 02, 2018

17820 8Ave SW

Edmonton, AB T6W2S6

Dec 02, 2018

Mr. Kelly Shepherd

Instructor, English and Communications Department

NAIT

11762 - 106 St NW

Edmonton, AB T5G 2R1

Dear Mr. Shepherd,

I am submitting the report Building A Web Camera With CodeIgniter MVC Framework, as you requested, for your evaluation. This report partially fulfills my obligations towards the requirements of CMPE2960: Computer Engineering Capstone.

This report details the design and construction of a web camera application using CodeIgniter framework based on MVC (Model-View-Controller) software design pattern. It goes over in details how the framework works and how it is useful for our application software development. It also goes over the design and construction of hardware component of the camera.

I’d like to thank all of my instructors at NAIT for their dedication to work for the success of their students. Their passion for what they teach is a great motivator in the classroom.

Sincerely,

Xiao Liu

CNT Student

Contents

[List of Figures and Tables iv](#_Toc528662580)

[Abstract v](#_Toc528662581)

[1.0 Introduction 1](#_Toc528662582)

[2.0 MVC Overview 2](#_Toc528662583)

[2.1 CodeIgniter 4](#_Toc528662584)

[2.1.1 CodeIgniter Overview 4](#_Toc528662585)

[2.1.2 CodeIgniter File Structure 6](#_Toc528662586)

[3.0 Project Design 6](#_Toc528662587)

[3.1 Features and Requirement 6](#_Toc528662588)

[3.2 Application Flow 6](#_Toc528662589)

[3.3 Code Structure 6](#_Toc528662590)

[3.3.1 Controller 6](#_Toc528662591)

[3.3.2 View 6](#_Toc528662592)

[3.3.3 Model 6](#_Toc528662593)

[4.0 Camera controller 7](#_Toc528662594)

[4.1 Hardware Component 7](#_Toc528662595)

[4.2 Controller Page Flow 7](#_Toc528662596)

[5.0 Conclusion 7](#_Toc528662597)

[Reference 8](#_Toc528662598)

# List of Figures and Tables

# Abstract

This report gives a brief introduction of MVC framework used in Raspberry Pi Secure Camera user authentication module and discusses the implementation of MVC pattern through user authentication in the web camera application using CodeIgniter MVC framework. The report also introduces the hardware design of the web camera using Raspberry Pi.

Building a web camera with CodeIgniter MVC framework

# 1.0 Introduction

With the increasing popularity of web applications, software developers are seeking for the framework which integrates the common web functionalities with built-in supporting libraries, provides scalabilities for enterprise development environment and facilitates team-based coding practice. Therefor the MVC framework has been emerged as one of the popular frameworks to accommodate the software industry’s demand.

The purpose of this report is to work through the process of designing and implementing the MVC architecture pattern using CodeIgniter- a PHP based framework to build a lightweight web camera application where users need to register and login through CodeIgniter-powered authentication process before access to web camera controller page. Compare with other frameworks that are using MVC pattern, such as Java Spring, Asp.net MVC, the CodeIgniter framework is the most suitable one found for this web camera application for its compatibility with Apache server installed on Raspberry Pi, light weight design for small and medium web project.

However there are many features in CodeIgniter will not be covered in this report because CodeIgniter provides various other functionalities such as unit testing, custom libraries support, many of which are not used in this application due to the time constraint of development process. The focus of this report will be the user authorization implementation as a demonstration of how MVC works in web camera application and web camera controller page structure.

An overview of MVC design pattern will be covered in section two, introducing CodeIgniter framework. Overall design of the project and how the application will function will be covered in the third section. Hardware design and web camera controller page code structure will be covered in the fourth section. The last part of the project will discuss the challenges during the development process and possible future enhancement and improvement.

# 2.0 MVC Overview

The Model-View-Controller (MVC) is an architectural pattern that separates an application into three main logical components: the model, the view, and the controller. Each of these components are built to handle specific development aspects of an application. MVC is one of the most frequently used industry-standard web development framework to create scalable and extensible projects.

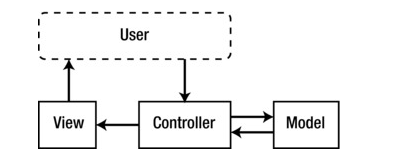


Figure 1 MVC flow(Pitt, 2012)

As the above figure 1 depicts, the controller serves as the “brain” of the application which handles all users’ requests according to the business logic. It acts as “middleman” between view and model.

Model mainly handles all database interactions by setting up connection to database, performing SQL queries and updating data in database or retrieving data for controller.

View is where user interactive elements are kept (HTML, CSS, and JavaScript), any data that are presented to user on webpage is to be processed by Controller first. Sometimes what the user sees are the combination of different views.

An example of work flow of MVC framework is that user clicks a button on the webpage to update his password, the controller receives the update request, calls the model. The Model then retrieve the user’s row in database table and update the password. Upon completion, the model will notify controller and controller will update the view with success message to the user.

As Pitt (2012) points out, in any unstructured application, the development aspects (UI design, business logic, debugging, testing…) tend to melt together in an incoherent mess. When the database needs to be changed to accommodate a new product line, or the company decides to rebrand, it doesn’t only affect the code it should. More developers have to get involved to make sure that changes in one part of the application don’t immediately break other parts of the application. Changes that should only affect a tiny section of code end up spilling into all sorts of strange and problematic areas.

Therefore the MVC has come for rescue. It defines strict containers for all of an application’s code and features so that when there are changes to database, views and controllers will not break. When an application’s UI design changes drastically, its controller and model are not affected by the changes. All of which makes it possible for front-end and back end developers to work independently which is essential to any enterprise application development.

## 2.1 CodeIgniter

### 2.1.1 CodeIgniter Overview

CodeIgniter is a powerful open-source PHP framework with a very small footprint, created by Rick Ellis in 2006. It was born from [ExpressionEngine](https://expressionengine.com/), essentially a collection of refactored classes originally written for EllisLab's flagship CMS. Stripped of the application-specific functionality, CodeIgniter was made to be a simple and elegant toolkit, enabling rapid development of both web sites and web applications, attracting thousands of talented PHP developers (EllisLab, 2018).

As indicated in the table below, compared with other popular PHP framework, CodeIgniter is more suitable for small- medium project and is an ideal framework for web camera application project mainly for the following reasons.

Developer Friendly

CodeIgniter includes libraries covering the common web applications functions such as database query, form validation, which saves the developer’s time and enable them to focus on the main application logic. For example, for querying database without libraries, the following code is needed:

$connection = mysql\_connect ("localhost","username","password");

mysql\_select\_db ("webcamera", $connection);

$result = mysql\_query ("SELECT \* FROM users", $connection);

While ($row = mysql\_fetch\_array ($result, MYSQL\_NUM)) {foreach ($row as $attribute) print "{$attribute [1]}";}   
with built in database library, developer will just need to use the following code:

$this->load->database ('webcamera');

$query = $this->db->get ('users'); foreach ($query->result () as $row) {print $row->url ;}

Code Ignitor handles the database connection and start/end transaction separately from the main code in configuration file which saves the hassle of typing duplicating code whenever database interaction is needed.

Simple Deployment

To deploy framework project from local sever to web host services usually requires complex and extensive configurations and server scripting. CodeIgniter makes the process a lot easier. All that required is to copy project folder and set up database configuration with hosting server. More details about deployment will be covered in the next section.

Table 1. Pros and Cons of PHP framework (Reigns, 2018)

|  |  |  |
| --- | --- | --- |
| Framework | Pro’s | Con’s |
| CodeIgniter | -Very developer friendly Doesn't need any special dependencies or supports  -Ability to use normal web hosting services well, using standard databases such as MySQL  -Outperforms most other frameworks (non MVC)  -Good documentation and LTS (Long Term Support) | -No namespaces, however this can speed up the execution time -Not as friendly towards unit testing as others  -Few libraries that are built inside the framework |
| [Laravel](https://coderseye.com/best-php-frameworks-for-web-developers/#laravel) | -Organized ﬁles and code  -Rapid application development  -MVC architecture (and PHP7)  -Unit testing (FAST on HHVM)  -High level of abstraction  -Overloading capabilities using dynamic methods  -Tons of out of the box functionality  -Payment integration with stripe  -Very strong encryption | -Does NOT work on Shared hosting plans  -Does Many queries on your database |
| [CakePHP](https://coderseye.com/best-php-frameworks-for-web-developers/#cakephp) | · Modern framework · Supports PHP 5.5+ · Scaffolding system and Fast builds · Very good for commercial web applications (MIT License) · Database Access, Caching, Validation, Authentication, are built in  · Extensive safekeeping tools include cross site  · scripting prevention, SQL Injection prevention,  · CSRF, and Form Validation · Good Documentation · Actively developed | -Not as good for constructing Restful APIS as Laravel or others listed |

### 2.1.2 CodeIgniter File Structure

As shown in figure 2 below, being a lightweight framework, the Codeigniter consists of three main folders with total size of only 2MB.

System folder contains the definitions of all system default libraries, stores the system variables, helpers’ functions, fonts, supporting languages and other system-related entities. If required, developers can define their own system variables in system/core folder. In this project, a custom debug variable is inserted into Logs.php under core/compact folder for debugging purpose.

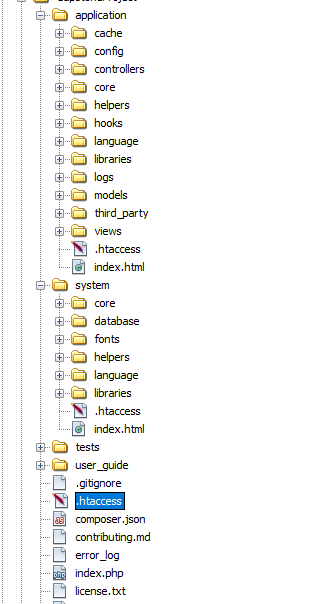
protected $\_levels = array('ERROR' => 1, 'DEBUG' => 2, 'INFO' => 3, 'ALL' => 4, 'PORJECT'=>5);

Figure 2.CodeIgniter File Structure

The main code resides in the application folder, where code for model, view, and controller goes to model, view, and controller folder respectively. Any extra third libraries can be put in libraries folder. Developer can conveniently load all required system-defined libraries in config/autoload.php file. For this project session, form validation and database helping libraries are loaded upon starting the application. Model made for retrieving user information is also auto loaded.

$autoload ['libraries'] = array ('session','Form\_validation','database');

$autoload ['model'] = array ('User\_ model');

The MVC code will be discussed in details in the next chapter.

# 3.0 Project Design

## 3.1 Features and Requirement

The security camera has the following features:

* 640x480 (VGA), 320x240 (QVGA), 160x120 (QQVGA) resolutions
* 307,200 effective pixels
* Framerate 30fps at VGA resolution
* 3.6 mm, F2.0 lens
* Configuration and viewing via standard internet browser
* Automatic infrared night vision function
* External GPIO Servo Control for tiling and rotation
* User registration and authentication

To fulfill the requirement, the security camera is equipped with two SG-90 servo motors connecting to Raspberry Pi’s GPIO (General Purpose Input Output) pins to control the pan and tile of the camera. And the camera part is Pi Noir Camera V2. This Chapter will focus on the code structure of user registration and authentication, the hardware component will be discussed in details in the next chapter.

## 3.2 Application Flow

After entering camera’s IP address in the internet browser, user will be directed to home view page which contains links for sign-in and registration.

Registration view page will send the users’ inputs to controller page for processing. The controller page then requests data from model page for validating again user’s input. Upon successful validating, controller page will push the user data into model page and model page will insert the user data into the database. And then user will be directed to camera control folder index page to access camera features.

Similar to registration, upon receiving user’s inputs from login view page, the controller will validate the inputs against data from model page for authentication. Upon successful sign-in, user will be directed to camera control folder index page to control the camera.

If either login or registration process failed, user will be redirected to home page along with the error messages.

## 3.3 Code Structure

### 3.3.1 Controller

To handle user’s logic, a Users class is created in the controller page with two main functions Register () and Login ()

Class Users extends CI\_Controller{

Public function register () { }

Public function login () { }

}

The Users class will have to inherit from CI\_Controller base class to be identified as a Codeigniter Controller.

In register function, the title information is put into $data array to pass on to registration view page.

$data ['title'] ='Sign Up';

The controller will then use the form\_validation library to set rules for user input controls in the view page. The required parameter of set\_rules function will make sure the form control has value. Developer can specify custom validation rules by defining callback function in the set\_rules function. In this project the custom function callback\_check\_user\_name\_exists is used to check if the username already exists in the database.

$this->form\_validation->set\_rules ('name','Name','required');

$this->form\_validation->set\_rules ('username', 'Username', 'required|callback\_check\_user\_name\_exists');

$this->form\_validation->set\_rules ('email', 'Email', 'required');

$this->form\_validation->set\_rules ('phonenumber', 'Phonenumber', 'required');

$this->form\_validation->set\_rules ('password', 'Password', 'required');

$this->form\_validation->set\_rules ('password2', 'Confirm Password', ‘matches [password]');

Upon setting up the rules for registration view page controller, the form\_validation run () command will be executed to validate user’s input in the view page against the rules. If the form\_validation run fails, view pages in templates/header, footer and users/register folders will be loaded thus user is redirected to the register view page.

if ($this->form\_validation->run()===FALSE) {

$this->load->view ('templates/header');

$this->load->view ('users/register', $data);

$this->load->view ('templates/footer');

} else {

$enc\_password=md5 ($this->input->post ('password'));

$this->User\_model->register ($enc\_password);

$this->session->set\_flashdata ('user\_registered',’you are now registered and can login');

redirect (‘CameraControFolder’);

}

If user passes the validation, the password entry will be encrypted using md5 algorithm and all user’s input will be stored into database by calling the register function in the User Model page.

$this->User\_model->register ($enc\_password);

Meanwhile controller will use the session library to set the registered session message.

$this->session->set\_flashdata ('user\_registered',’you are now registered and can login');

Similar to register () function, in controller login () function the user inputs in login view page are validated against the rules first and then retrieved.

$username=$this->input->post ('username');

$password=md5 ($this->input->post ('password'));

After that controller will call the login function in User model page to check if username and password matches the database record. If $ user\_id returned from model is true, then a session will be created by controller storing user\_id, username, logged\_in status with success login message. Else user will be redirected to login view page.

$user\_id=$this->User\_model->login ($username, $password);

If ($user\_id) {$user\_data=array (

'user\_id'=>$user\_id,

'username'=>$username,

'logged\_in'=>true);

$this->session->set\_userdata ($user\_data);

$this->session->set\_flashdata ('user\_loggedIn','You are now logged in as'. $this->session->username);

redirect (CameraControFolder);

} else {

$this->session->set\_flashdata ('login\_failed','Login is invalid');

redirect ('users/login');

}

### 3.3.2 View

As show in figure 3 below, the view pages contain the html files for front-end user

Figure 3. Registration View Page

interface. To link the view page and controller page properly, view page needs to specify the controller’s class and function using form\_open command so that when user clicks the submit button, all form data will send to users controller for processing.

<? php echo form\_open ('users/register') ;?>

Also view page could also render data in $data array variable sent by controller and display form validation errors by calling validation\_errors () function.

<? php echo validation\_errors () ;?>

### 3.3.3 Model

As Model pages are designed to handle all database transactions, the database configuration needs to be done in the application/config/database.php file first as shown in figure 4, in which the database location, user credential, database engine and other database variables are specified.

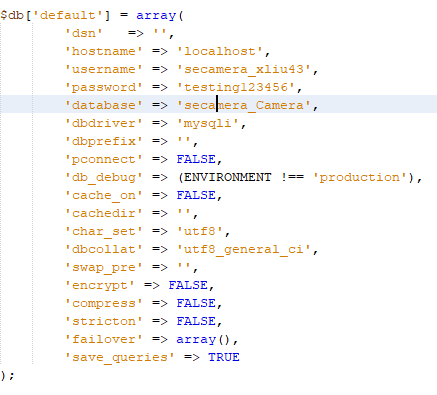


Figure 4. Database Configuration

After database configuration, register and login functions can be defined in User\_model class inheriting from CI\_Model base class:

class User\_model extends CI\_Model {

public function register ($enc\_password){}

public function login ($username, $password){}}

The register function will insert the encrypted password sent by controller along with all user post data in register view page to the users table in the database.

$data=array (

'name'=>$this->input->post('name'),

'email'=>$this->input->post('email'),

'username'=>$this->input->post('username'),

'password'=> $enc\_password,

'zipcode'=>$this->input->post ('zipcode'),

'phonenumber'=>$this->input->post ('phonenumber')

);

return $this->db->insert('users',$data);

The system database library is called when inserting the $data array into users table which saves developer a lot of time writing repetitive database transaction codes.

Similarly, the login function checks $username, $password sent by controller in the users table to see if there is matching entry. If so, return the user id, otherwise return false.

$this->db->where ('username', $username);

$this->db->where ('password', $password);

$result=$this->db->get ('users');

if ($result->num\_rows()==1) {

return $result->row (0)->id;

} else { return false ;}

# 4.0 Camera controller

## 4.1 Hardware Component

## 4.2 Controller Page Flow

# 5.0 Conclusion

# References

Pitt, C. (2012). Pro PHP MVC. Berkeley, CA: Apress. https://doi.org/10.1007/978-1-4302-4165-2

Blanco, J., & Upton, D. (2009). CodeIgniter 1.7 improve your PHP coding productivity with the free compact open-source MVC CodeIgniter framework! Birmingham, U.K.: Packt Pub.

EllisLab. (2018). About CodeIgniter. From Ellislab Web Site: https://ellislab.com/codeigniter

Reigns, S. (2018). 11 Best PHP Frameworks for Modern Web Developers in 2018. From Coders Eye Web Site: https://coderseye.com/best-php-frameworks-for-web-developers/