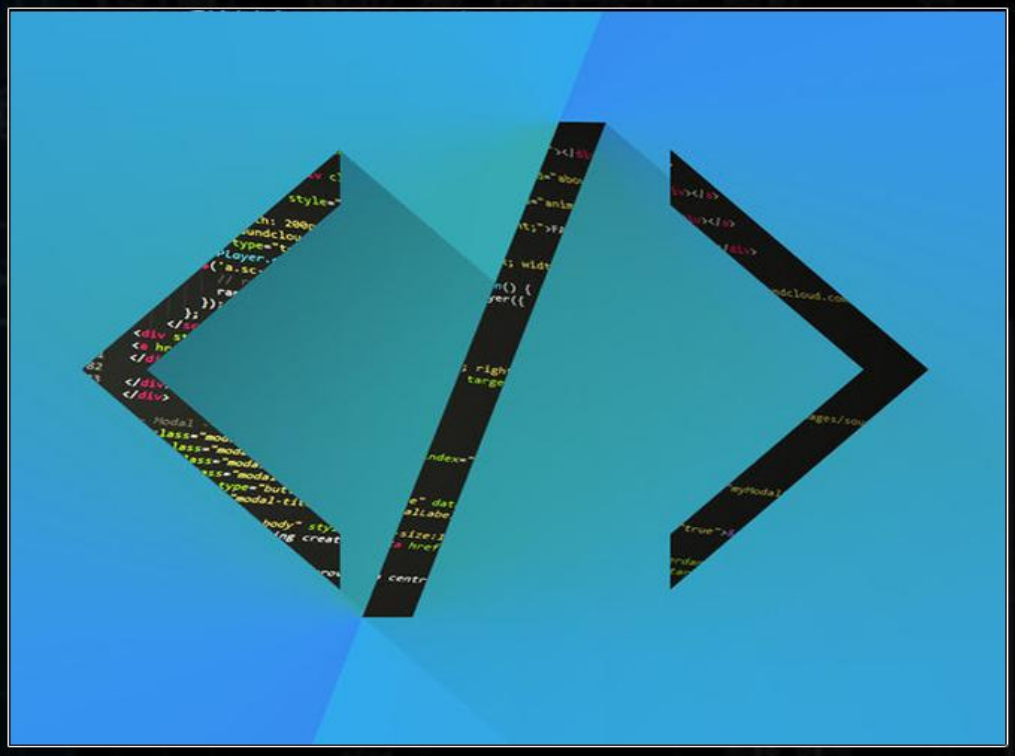
#2

UNDERCODE

TALLER DE PROGRAMACIÓN WEB



TEMAS

VISTA
MODELO
CONTROLADOR
INSTALACIÓN DE CODEIGNITER
ENVÍO DE INFORMACIÓN
Y MAS...!

TUTOR

DEBOBIPRO

Tabla de contenidos

Saludo 3

Instrucción inicial 3

- Vista 3
- Controlador 3
- Modelo 3

Barra de direcciones 4

- Evitar el index.php en la URL 4

Instalando CodeIgniter 5

- Directorio application 5
- Directorio system 5

Primera vista 6

Configuración Inicial 7

- autoload.php 7
- config.php 7
- database.php 7
- routes.php 7

Controladores 8

- Funciones dentro de un controlador 8
- Práctico 1 9

Vistas 10

- Cargando vistas desde un controlador 11
- Enviando información a las vistas 11
- Práctico 2 12

Modelos 13

Controladores, modelos y vistas 14

Despedida 16

Saludo

Muy bienvenido seas a la segunda parte del tutorial de Bootstrap y CodeIgniter, hoy aprenderemos a utilizar las 3 etapas básicas del framework CODEIGNITER, modelo, vista y controlador.

Espero lo disfrutes y te sea de utilidad.

Instrucción inicial

Para entender de mejor manera como utilizaremos este framework les daré una pequeña descripción de lo que haremos en cada "capa":

Vista

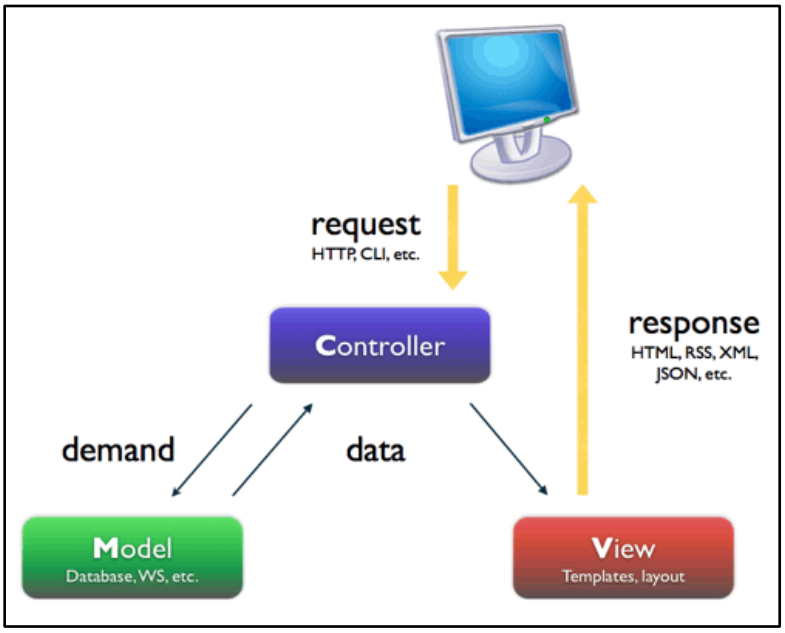
En esta capa tendremos casi exclusivamente códigos html o algunos "echo" de php, esta será la parte visual, la que ve el cliente.

Controlador

En esta capa estará la lógica de nuestra aplicación, 100% código PHP, esta capa envía la información a la "vista".

Modelo

En esta capa tendremos todas las interacciones a la base de datos, no existe lógica de programación.



Barra de direcciones

CodeIgniter trabaja con la siguiente organización en la barra de direcciones:

```
http://www.tuweb.com/CONTROLADOR/FUNCION/PARAMETRO1/PARAMETRO2/.../PARAMETROX
```

Es decir, después de nuestra dirección base podemos especificar el nombre de algún controlador seguido de la función de ese controlador, y finalmente, todos los parámetros de entrada si es que requiere.

Evitar el index.php en la URL

Para evitar agregar la palabra 'index.php' en nuestras direcciones crearemos un archivo llamado '.htaccess' en nuestra raíz con el siguiente contenido:

```
<IfModule mod_rewrite.c>
      RewriteEngine On
      RewriteCond %{REQUEST_FILENAME} !-f
      RewriteCond %{REQUEST_FILENAME} !-d
      RewriteRule ^(.*)$ index.php/$1 [L]
</IfModule>
<IfModule !mod_rewrite.c>
ErrorDocument 404 /index.php
</IfModule>
```

De esta manera podremos utilizar:

```
http://www.tuweb.com/controlador/
```

En vez de:

```
http://www.tuweb.com/index.php/controlador/
```

Así tendremos una URL más limpia.

Instalando CodeIgniter

Los archivos descargados en el la parte I de este taller los dejaremos en nuestra carpeta raíz de nuestro hosting, subdominio o directorio local.

Quedando de la siguiente manera:

```
[+] - Directorio
 | --[+] application
 | --[+] system
 | --    index.php
 | --    .htaccess
```

Que serán los archivos necesarios, la documentación y los archivos .git no son necesarios para este desarrollo.

Directorio application

En este directorio encontraremos todo lo necesario para poder desarrollar, ahí está la carpeta de modelos (models), vistas (views) y controladores (controllers).

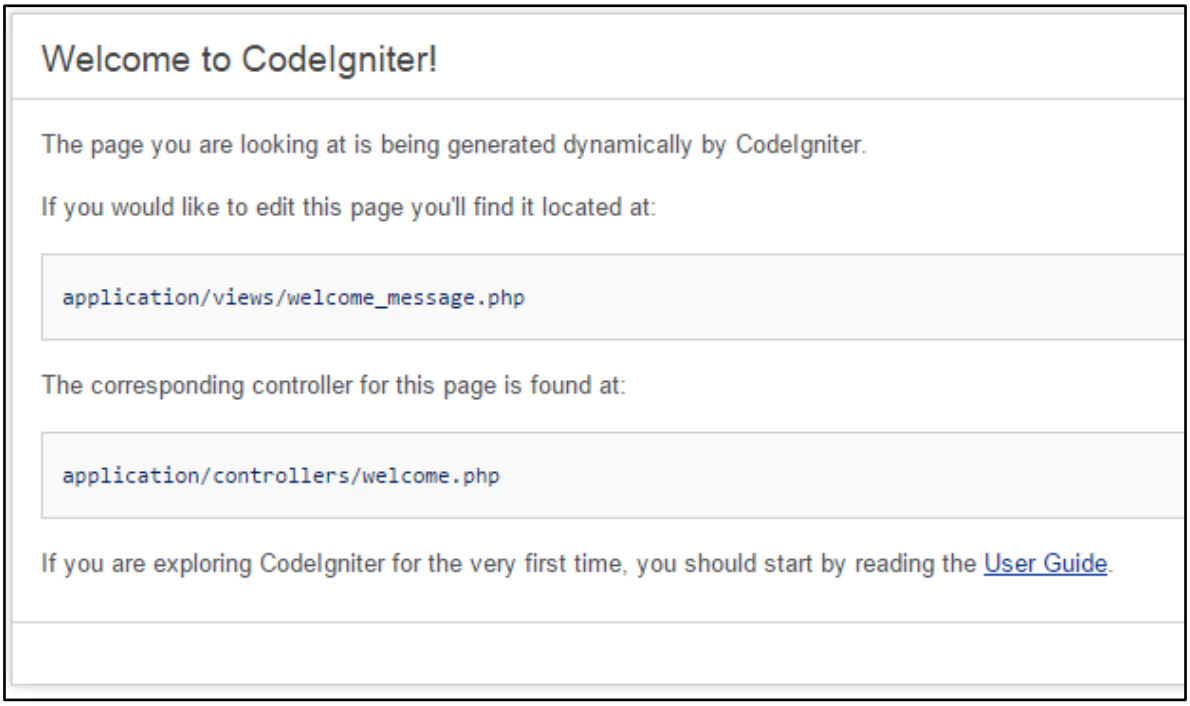
Encontraremos también los archivos de configuración de bases de datos y configuraciones generales.

Directorio system

En este directorio está la configuración propia de CodeIgniter, no nos meteremos aquí al menos en este taller.

Primera vista

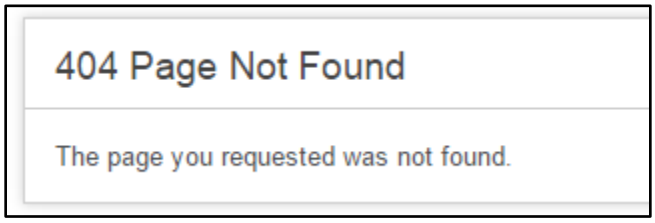
Accedemos por URL a nuestra web y veremos algo como esto:



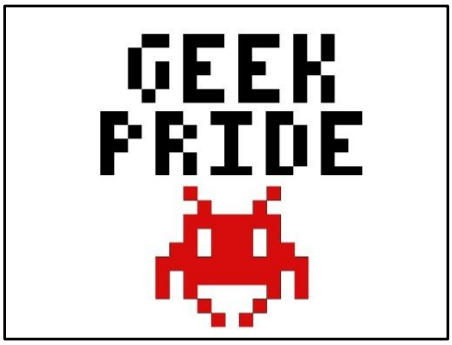
Pero como aprenderemos desde 0 vamos a borrar esos dos archivos que trae por defecto:

Welcome_message.php y welcome.php

Debemos conseguir el error 404:



Ahora si tenemos CodeIgniter VIRGEN, listo para comenzar a trabajar en nuestra aplicación.



Configuración Inicial

Siento mucho hacerlos pasar por esta parte que puede resultar bastante tediosa, pero ya viene la parte divertida y práctica, haremos rápida esta parte.

Nos situaremos en la siguiente dirección: application/config

autoload.php

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

Cargaremos automáticamente las librerías de base de datos, session y el helper de URL, así no tendremos que llamarlos cada vez que los necesitemos.

config.php

```
$config['index_page'] = '';
$config['encryption_key'] = 'underc0de##Taller$$';
```

El encryption_key puede ser cualquier cadena, preferentemente de 32 caracteres de largo, la usaremos para las variables de sesión.

database.php

```
$db['default']['hostname'] = 'localhost';
$db['default']['username'] = 'root';
$db['default']['password'] = '';
$db['default']['database'] = 'underDB';
```

Ustedes completan con la información correcta.

routes.php

```
$route['default_controller'] = "underc0de";
```

Aquí configuraremos el controlador que cargará al inicio, es decir, al cargar la página.

Como hemos eliminado el llamado “welcome.php” nos da error 404 como vimos anteriormente, pero ahora crearemos el controlador con el nombre que ustedes escriban ahí.

Nota: No es necesario escribir la extensión .php

Hecho esto ¡vamos a los códigos!

Controladores

Para trabajar los controladores nos situaremos en el directorio “application/controllers” y crearemos un archivo php con el nombre que escogieron anteriormente, en mi caso ‘underc0de.php’, dentro del archivo escribiremos lo siguiente:

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

class Underc0de extends CI_Controller {

	public function index(){

	}

}
```

Es importante agregar la primera línea, así no pueden tener acceso por otro lado que no sea directamente nuestra aplicación.

Recuerden que el nombre de la clase comienza con mayúsculas aunque el archivo no comience con mayúsculas, mi archivo se llama ‘underc0de.php’.

No cierren la etiqueta php al final del controlador.

Funciones dentro de un controlador

Por defecto podemos incluir la función llamada “index()” que sería la función que carga al llamar el controlador desde un navegador, ahora si cargamos nuestra web veremos una página en blanco, pero podemos hacer unas modificaciones para entenderlo de mejor manera:

```
public function index(){
	echo "Yo inicio inmediatamente";
}

public function saludar($nombre = "Usuario de underc0de.org"){
	echo "Hola: ".$nombre;
}
```

http://www.tuweb.com mostrará lo siguiente:

Yo inicio inmediatamente

http://www.tuweb.com/underc0de/saludar mostrará lo siguiente:

Hola: Usuario de underc0de.org

http://www.tuweb.com/underc0de/saludar/DeBobiPro mostrará lo siguiente:

Hola: DeBobiPro

* Es importante que tengan habilitado el módulo de “Rewrite” en sus servidores.
http://www.anmsaiful.net/blog/php/enable-apache-rewrite-module.html

Práctico 1

Crear un nuevo controlador llamado como tu Nick en el foro de underc0de.org y una función llamada “Jalisco” que reciba un parámetro de entrada numérico.

Al cargar esa función por URL debe imprimir por pantalla el número ingresado, aumentado en una unidad seguido por el mensaje “gané”.

EJEMPLO:

http://www.tuweb.com/DeBobiPro/jalisco/7

Debe decir: Yo digo: 8 gané!

http://www.tuweb.com/DeBobiPro/jalisco/abc

Debe decir: eso no es un número…

Vistas

Las vistas nos ayudarán a darle un toque más llamativo a la parte que el cliente verá, es importante aclarar que no podremos acceder directamente a estas vistas a no ser que pasemos por un controlador que las cargue en nuestra página.

El orden al momento de cargar las vistas debe ser coherente con la programación, lo entenderemos con el siguiente ejemplo:

Trabajaremos en el directorio “views” y creamos 3 archivos: header.php, body.php y footer.php

Header.php

```
<!DOCTYPE html>
<html lang="es-ES">
<head>
	<meta charset="UTF-8">
	<title>Mi primera WEB con CodeIgniter - Underc0de.org</title>
</head>
```

body.php

```
<body>
	<h1>Título</h1>
	Hola, esto cargó por el Body
</body>
```

Footer.php

```
<footer>
	Visitanos en <a href="http://www.underc0de.org">UnderC0de</a>
</footer>
</html>
```

Podemos ver que tenemos una página web partida en 3 partes y en diferentes archivos, esto nos ayudará a mantener encabezados, menús, banners, etc… de manera estática y no tener que escribir el código cada vez que lo necesitemos, simplemente llamando nuestra vista en el lugar correcto.

¿Cómo lo hago…?

Cargando vistas desde un controlador

Volveremos a nuestro controlador previamente creado (en mi caso 'underc0de.php') y en la función index() escribiremos las siguientes líneas:

```
public function index(){
	$this->load->view('header'); // Esto carga primero
	$this->load->view('body');   // Esto carga segundo
	$this->load->view('footer'); // Esto carga tercero
}
```

En esta parte es importante el orden que le damos a las vistas.

Cargamos nuestra web del navegador y veremos lo siguiente:

Enviando información a las vistas

Para enviar información a las vistas creamos un arreglo de esta manera en nuestro controlador:

```
public function index(){
	$datos['miembro'] = "DeBobiPro"; //creamos un arreglo
	$this->load->view('header');
	$this->load->view('body',$datos); //Se lo enviamos a la vista
	$this->load->view('footer');
}
```

Y en la vista tendremos una variable php con el nombre de la asociación de nuestro arreglo con el contenido que le hemos dado, (sí, hablé en Chino, lo entenderás con el ejemplo).

body.php

```
<body>
	<h1>Título</h1>
	Hola<?= "[".$miembro."]"; ?>, esto cargó por el Body
</body>
```

El nombre asociado al String "DeBobiPro" es "miembro" (ver controlador) y es esa la variable que ahora existe en mi body y que puedo utilizar ($miembro).

¿Fácil, no?

Práctico 2

Crear una vista llama “tabla.php” con las cabeceras:

Nombre	Mensaje

Y un controlador que envié el contenido de esa tabla quedando de la siguiente manera:

Nombre	Mensaje
Juan	Saludos!
Catalina	Yo aprendí en underc0de.org!

La estructura para esta tabla es la siguiente:

```
<table>
	<thead>
		<tr>
			<th>Nombre</th>
			<th>Mensaje</th>
		</tr>
	</head>
	<tbody>
		<tr>
			<th>Juan</th>
			<th>Saludos!</th>
		</tr>
		<tr>
			<th>Catalina</th>
			<th>Yo aprendí con underc0de.org</th>
		</tr>
	</tbody>
</table>
```

Modelos

Finalmente y ya llegando al final de este taller, vamos a ver cómo trabajan los modelos, para ello crearemos una tabla en nuestro servidor local de base de datos:

```
CREATE TABLE IF NOT EXISTS `miembros` (
  `id_miembro` int(11) NOT NULL AUTO_INCREMENT,
  `nombre` varchar(200) NOT NULL,
  `mensaje` varchar(200) NOT NULL,
  PRIMARY KEY (`id_miembro`)
);
```

Validar el nombre de la base de datos configurada en el archive database.php

```
INSERT INTO `underc0de`.`miembros` (`id_miembro`, `nombre`, `mensaje`) VALUES
(NULL, 'Juanito', 'Saludos Model!'), (NULL, 'Catalina', 'Aprendí Modelos!');
```

Teniendo esto ¡ya podemos comenzar!

En nuestro directorio models crearemos un archivo llamado “usuarios_model.php” con el siguiente contenido:

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

class Usuarios_model extends CI_Model {

}
```

Y creamos una función llamada obtenerDatos() y atención a las líneas de código:

```
public function obtenerDatos(){
	$consulta = $this->db->get('miembros');
	return $consulta->result();
}
```

$this->db->get(‘miembros’); es equivalente a:

```
SELECT * FROM miembros;
```

Y hacemos un return con los resultados, en este momento nuestro modelo no sirve para nada ya que no se puede acceder mediante URL y no lo estamos llamando en ningún controlador, ahora vamos a ello.

Controladores, modelos y vistas

En el controlador underc0de nos ubicamos en la función index() agregando las siguientes líneas:

```
public function index(){
	$this->load->model('usuarios_model'); //cargamos nuestro modelo
	$datosmiembros = $this->usuarios_model->obtenerDatos(); //cargamos la
variable con los datos del modelo

	$datos['miembro'] = "DeBobiPro";
	$this->load->view('header');
	$this->load->view('body',$datos);
	$this->load->view('footer');
}
```

Y a modo de ejemplo los enviaremos dentro de una tabla (esto les ayudará para entender mejor el practico 2).

Hay diferentes formas de hacer esta parte, yo les enseñaré una:

```
public function index(){
	$this->load->model('usuarios_model'); //cargamos nuestro modelo
	$datosmiembros = $this->usuarios_model->obtenerDatos(); //cargamos la
variable con los datos del modelo

	$tabla = "";

	foreach ($datosmiembros as $fila) {
		$tabla.="<tr>
				<td>".$fila->nombre."</td>
				<td>".$fila->mensaje."</td>
			</tr>";
	}

	$datos['tabla'] = $tabla;
	$datos['miembro'] = "DeBobiPro";
	$this->load->view('header');
	$this->load->view('body',$datos);
	$this->load->view('footer');
}
```

Construimos una tabla en nuestro controlador y la enviamos a la vista:

Body.php

```
<body>
	<h1>Título</h1>
	Hola<?= "[".$miembro."]"; ?>, esto cargó por el Body
	<br><br>
	<table border="1">
		<thead>
			<tr>
				<th>Nombre</th>
				<th>Mensaje</th>
			</tr>
		</thead>
		<tbody>
			<?= $tabla;  ?>
		</tbody>
	</table>
	<br><br>
</body>
```

Y cargamos nuestra variable $tabla donde corresponde, obteniendo el siguiente resultado:

Título

Hola[DeBobiPro], esto cargó por el Body

Nombre	Mensaje
Juanito	Saludos Model!
Catalina	Aprendí Modelos!

Visitanos en UnderC0de

Despedida

Hemos llegado al final de la segunda parte de nuestro taller de CodeIgniter y Bootstrap, espero que les sea de utilidad lo expuesto y a practicar mucho ahora!!

Las respuestas de los prácticos puedes dejarlos en el post oficial del taller