



**TECNOLÓGICO NACIONAL DE MÉXICO**

---

**INSTITUTO TECNOLÓGICO DE TLÁHUAC**

**DEPARTAMENTO DE  
INGENIERÍA EN SISTEMAS COMPUTACIONALES**

**TRABAJO:**

**Codeigniter 4 calcular operaciones básicas**

**P R E S E N T A:**

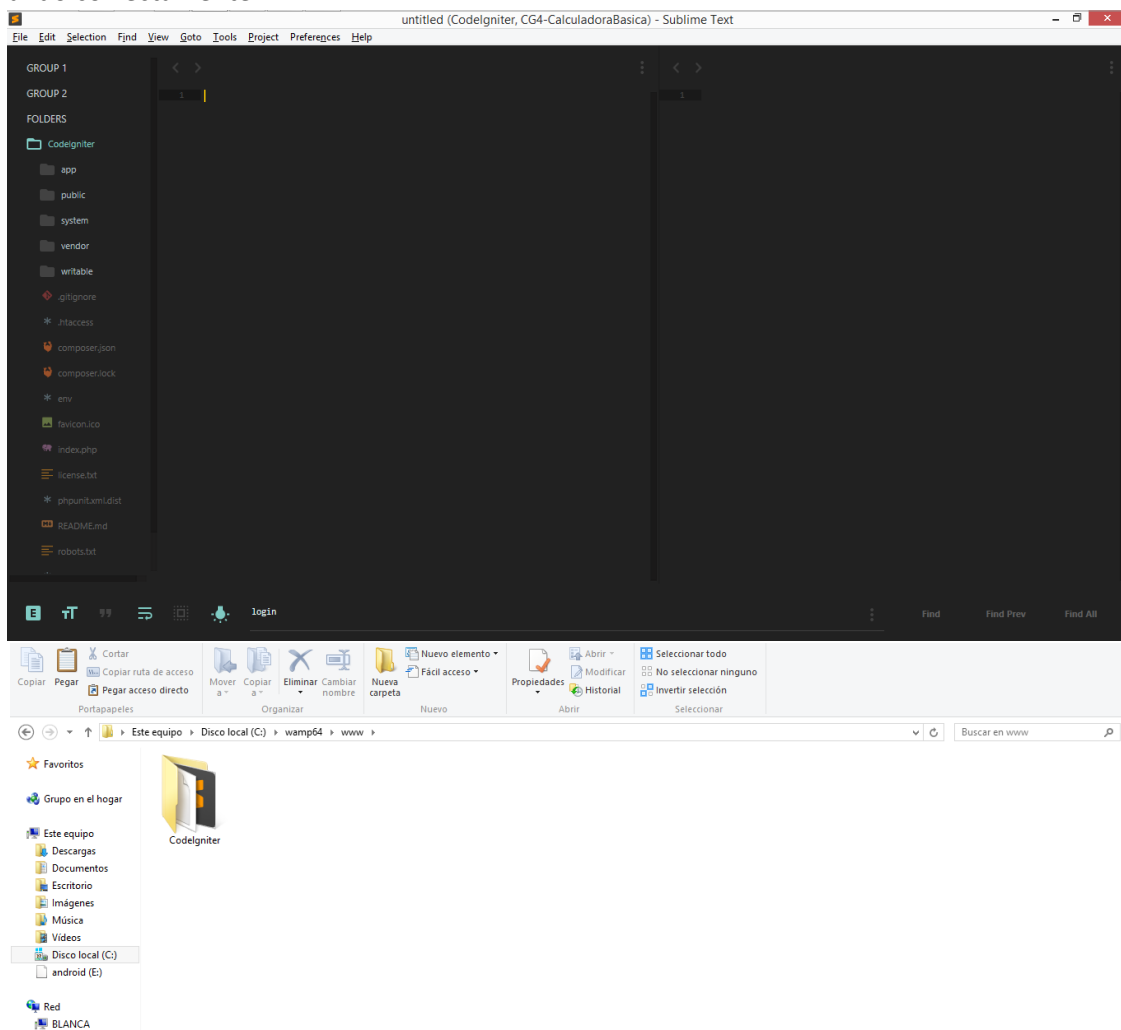
**VENTURA BARRÓN ALEJANDRO EZEQUIEL**

**PROFESOR**

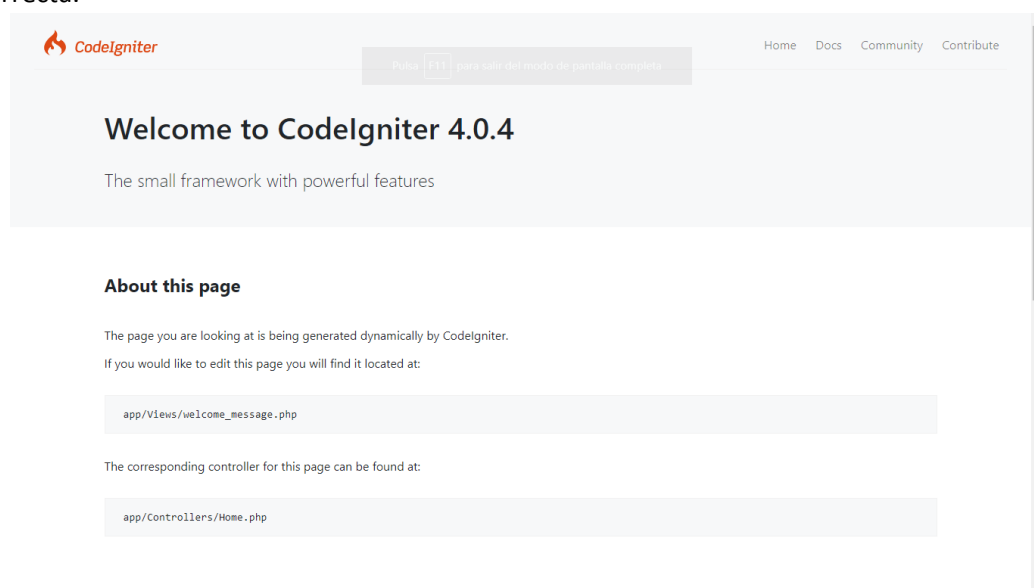
**ROLDAN AQUINO SEGURA**



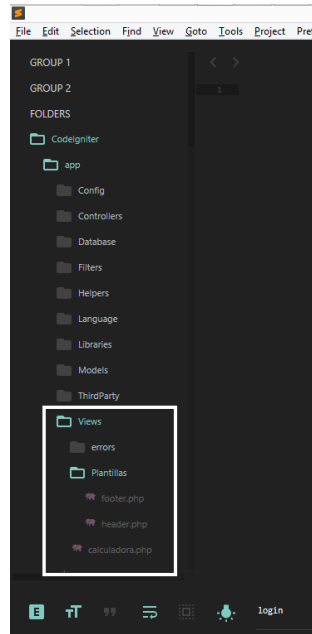
1. Crearemos un nuevo proyecto de CodeIgniter 4 y lo abriremos con un editor de texto en este caso se utilizará SublimeText. El proyecto lo colocaremos donde podamos ejecutarlo con el servidor local en este caso la carpeta de www de wamp server. Abriremos en el navegador con localhost para ver si el archivo está funcionando correctamente.



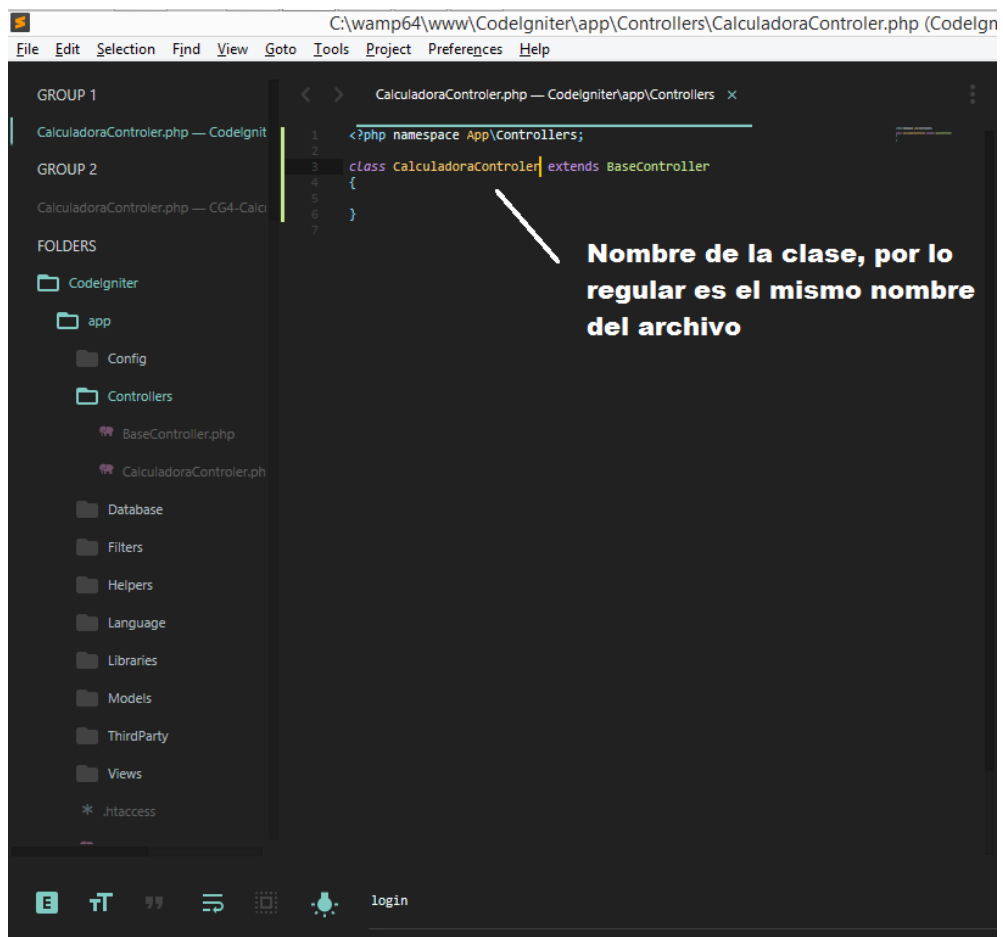
- Cuando aparezca el siguiente mensaje es que CodeIgniter 4 se está ejecutando de una manera correcta.



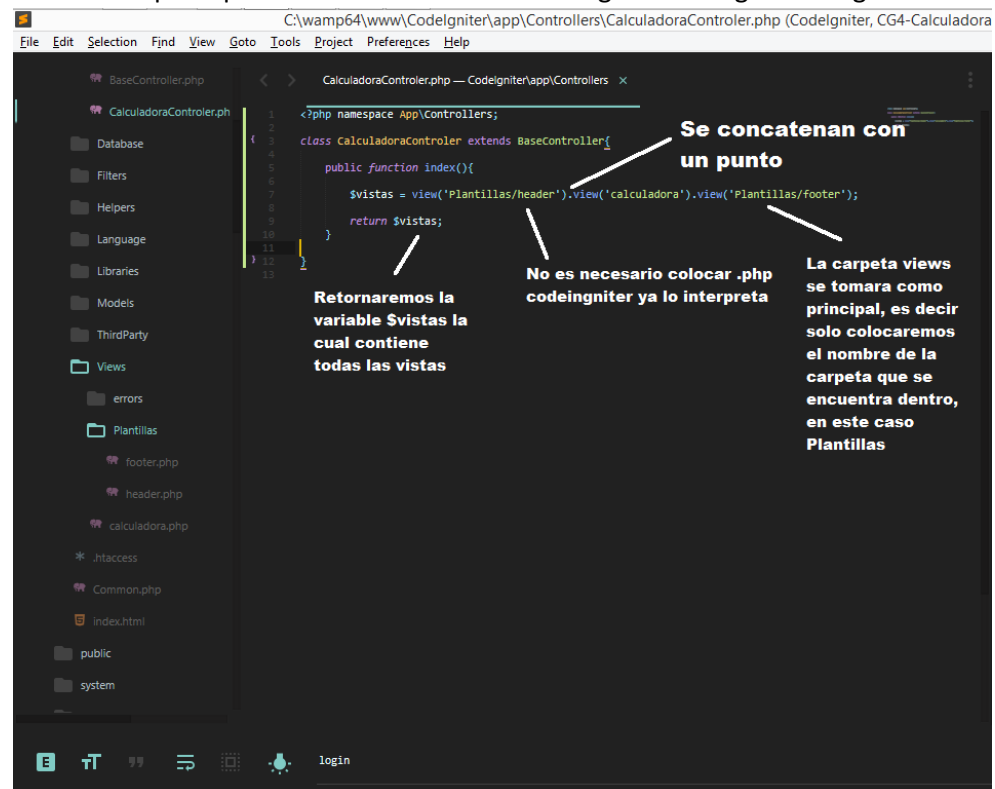
- Una vez ya se allá a agregado la carpeta a un editor de texto vamos a agregar los archivos que ocuparemos para la calculadora comenzaremos con las vistas. Las vistas se encuentran dentro de la carpeta app -> Views, ahí nos aparece un archivo por defecto que se llama welcome\_message.php, lo eliminaremos y crearemos un archivo nuevo llamado calculadora.php. Dentro de la misma carpeta Views crearemos una nueva carpeta llamada Plantillas, dentro de esta carpeta crearemos 2 archivos uno llamado header.php y otro llamado footer.php nos quedaría de la siguiente manera.



- Una vez que tengamos esos archivos agregados en las carpetas iremos a configurar un controlador el cual mandara a llamar a las vistas, este controlador lo agregaremos en la carpeta Controllers que se encuentra en app -> Controllers, por defecto Codeigniter 4 crea un controlador que se llama Home.php lo eliminaremos u crearemos un archivo que le pondremos de nombre CalculadoraControler.php abriremos este archivo dentro del mismo editor de texto y escribiremos el siguiente código.



4. Dentro de esta clase crearemos un método llamado index donde mandaremos a llamar a todas nuestras vistas. Para mandar a todas las vistas que ocuparemos las agregaremos a una variable y a continuación las retornaremos la variable para que sean visualizadas en el navegador. Código en la siguiente imagen.



5. Ahora configuraremos las rutas para que CodeIgniter sepa a donde será enviada y recibida la información, también a que métodos y a que clases. Abriremos el archivo Routes.php que se encuentra dentro de la carpeta app -> Config y modificaremos la siguiente línea de código.

```
<?php namespace Config;

// Create a new instance of our RouteCollection class.
$route = Services::routes();

// Load the system's routing file first, so that the app and ENVIRONMENT
// can override as needed.
if (file_exists(SYSTEMPATH . 'Config/Routes.php')) {
    require SYSTEMPATH . 'Config/Routes.php';
}

/**
 * Router Setup
 */

$route->setDefaultNamespace('App\Controllers');
$route->setDefaultController('CalculadoraController');
$route->setDefaultMethod('index');
$route->setTranslateURIDashes(false);
$route->set404Override();
$route->setAutoRoute(true);

/**
 * Route Definitions
 */

// We get a performance increase by specifying the default
// route since we don't have to scan directories.
$route->get('/', 'Home::index');

/**
 * Additional Routing
 */

// There will often be times that you need additional routing and you
// need it to be able to override any defaults in this file. Environment
// based routes is one such time. require() additional route files here
// to make that happen.

// You will have access to the $route object within that file without
// needing to reload it.
```

## 6. También modificaremos la siguiente línea de código

```
<?php namespace Config;

// Create a new instance of our RouteCollection class.
$route = Services::routes();

// Load the system's routing file first, so that the app and ENVIRONMENT
// can override as needed.
if (file_exists(SYSTEMPATH . 'Config/Routes.php')) {
    require SYSTEMPATH . 'Config/Routes.php';
}

/**
 * Router Setup
 */

$route->setDefaultNamespace('App\Controllers');
$route->setDefaultController('CalculadoraController');
$route->setDefaultMethod('index');
$route->setTranslateURIDashes(false);
$route->set404Override();
$route->setAutoRoute(true);

/**
 * Route Definitions
 */

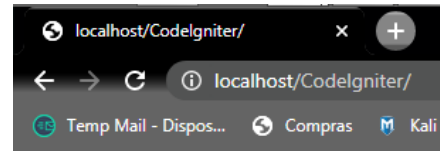
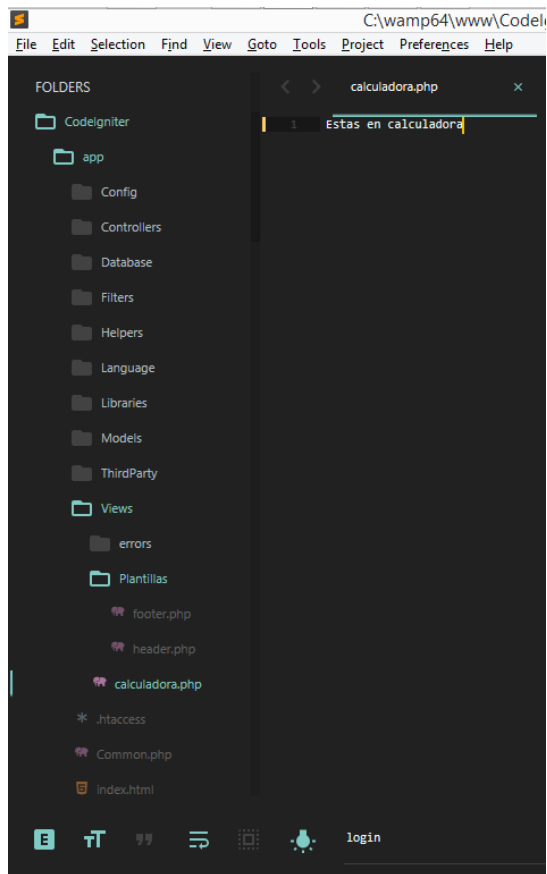
// We get a performance increase by specifying the default
// route since we don't have to scan directories.
$route->get('/', 'CalculadoraController::index');

/**
 * Additional Routing
 */

// There will often be times that you need additional routing and you
// need it to be able to override any defaults in this file. Environment
// based routes is one such time. require() additional route files here
// to make that happen.

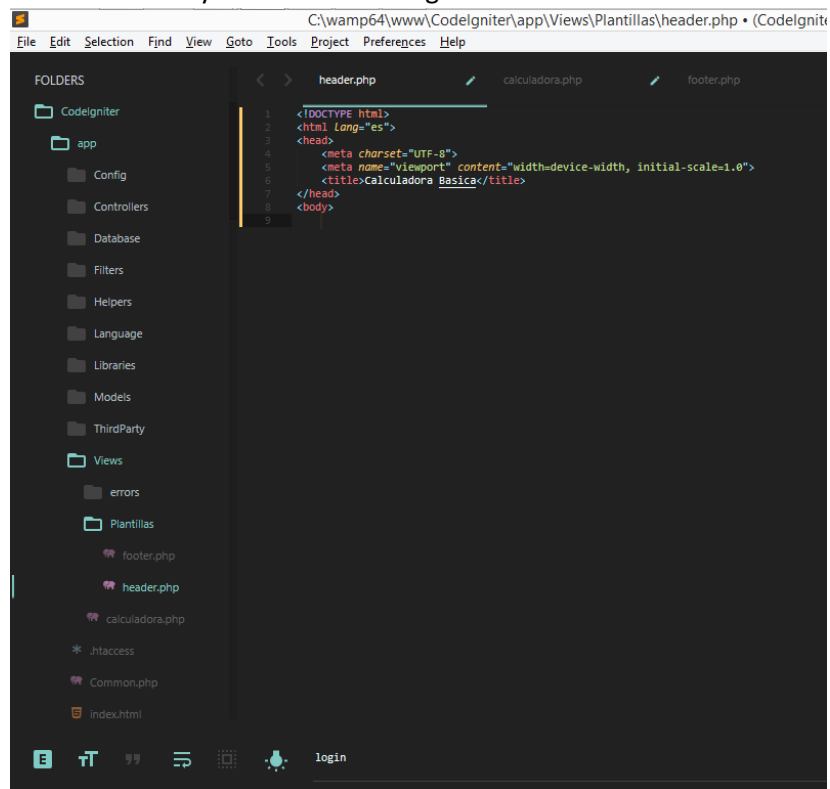
// You will have access to the $route object within that file without
// needing to reload it.
```

- Ahora veremos en el navegador si está funcionando para esto iremos de nuevo a la carpeta Views que se encuentra en app -> view y editaremos el archivo llamado calculadora.php escribiremos cualquier cosa y visualizaremos en el navegador lo que acabamos de escribir. Si se visualiza entonces todo está listo para comenzar a crear nuestra calculadora.

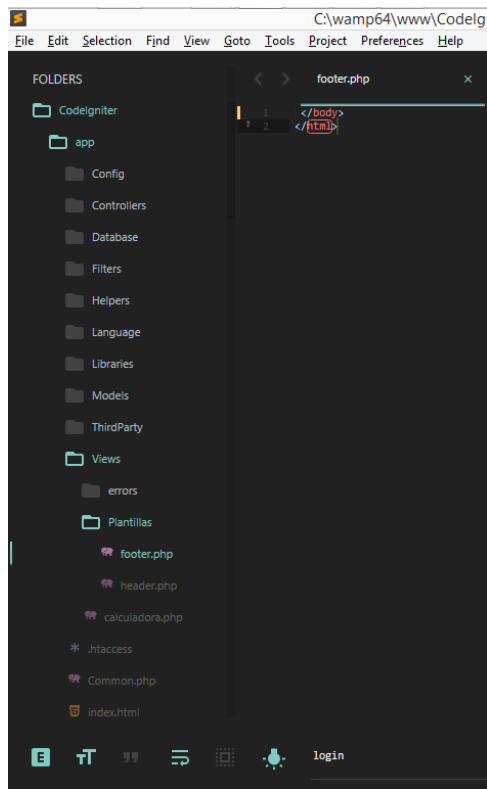


Estas en calculadora

8. Lo siguiente que tenemos que hacer es seccionar la parte de código html5, para esto abriremos el archivo head que se encuentra en Plantillas y escribiremos lo siguiente.



9. En el footer escribiremos lo siguiente.



10. En el archivo llamado calculadora.php agregaremos lo siguiente. Colocaremos un <h1> para el título principal un <h2> para el subtítulo, los <hr> se utilizan para agregar una línea en todo lo ancho de pantalla, aunque también se puede modificar su tamaño, dentro de la imagen se describe, cada parte de código.



A si se visualizaría en el navegador



11. Ahora regresaremos al archivo Routes ya que agregaremos una nueva ruta para que detecte cuando en el url aparezca /números. La agregaremos de la siguiente manera.



```

16  *
17  */
18  $routes->setDefaultNamespace('App\Controllers');
19  $routes->setDefaultController('CalculadoraControler');
20  $routes->setDefaultMethod('index');
21  $routes->setTranslateURIDashes(false);
22  $routes->set404Override();
23  $routes->setAutoRoute(true);
24
25  /**
26   * Route Definitions
27   */
28
29  // We get a performance increase by specifying the default
30  // route since we don't have to scan directories.
31  $routes->get('/', 'CalculadoraControler::index');
32  $routes->post('/numeros', 'CalculadoraControler::operaciones');
33
34  /**
35   * Additional Routing
36   */
37
38  * There will often be times that you need additional routing and you
39  * need it to be able to override any defaults in this file. Environment
40  * based routes is one such time. require() additional route files here
41  * to make that happen.
42  *
43  * You will have access to the $routes object within that file without
44  * needing to reload it.
45  */
46
47  if (file_exists(APPPATH . 'Config/' . ENVIRONMENT . '/Routes.php'))
48  {
49      require APPPATH . 'Config/' . ENVIRONMENT . '/Routes.php';
50  }

```

**Metodo por el cual se envia la informacion**

**Puede ser cualquier nombre**

**Nombre de nuestro controlador**

**Nombre del metodo a donde mandaremos los datos que llegen del petodo post**

12. Ya agregada la ruta iremos a nuestro controlador donde crearemos el método llamado operaciones, ahí recibiremos 3 parámetros lo que desea realizar el usuario (suma, resta, multiplicación, división,) y donde llegaran los 2 números ingresados por el usuario. Agregaremos el siguiente código.

```

1  <?php namespace App\Controllers;
2
3  class CalculadoraControler extends BaseController{
4
5      public function index(){
6
7          $data = [
8              "resultado" => ""
9          ];
10
11          $vistas = view('Plantillas/header')->view('calculadora', $data)->view('Plantillas/footer');
12          return $vistas;
13      }
14
15      public function operaciones(){
16
17          $num1 = $_POST['num1'];
18          $num2 = $_POST['num2'];
19      }
20  }

```

**creamos ste array y le daremos de nombre \$data, este array contiene una variable resultado que esta vacia, y se la enviaremos ala vista calculadora.php**

**Creamos metodo llamado operaciones**

**Creamos una variable llamada \$num1 y va a guardar el valor que viene de la vista nom en nombre num1 el cual los agregamos en los inputs.**

13. Crearemos un if para detectar que es lo que el usuario desea hacer (suma, resta...) de la siguiente manera.

```
<?php namespace App\Controllers;

class CalculadoraController extends BaseController{

    public function index(){

        $data = [
            "resultado" => ""
        ];

        $vistas = view('Plantillas/header').view('calculadora', $data).view('Plantillas/footer');

        return $vistas;
    }

    public function operaciones(){

        $num1 = $_POST['num1'];
        $num2 = $_POST['num2'];

        if ($_POST['opciones'] == "suma") {

        }

    }

}
```

Si mediante el metodo post en opciones llega lo que es suma has todo lo que esta dentro del if

14. Crearemos una variable que se llamara \$resultado y guardaremos la operación que realicen los números que llegaron desde la vista ya previamente guardados en las variables \$num1 y \$num2.

```
<?php namespace App\Controllers;

class CalculadoraController extends BaseController{

    public function index(){

        $data = [
            "resultado" => ""
        ];

        $vistas = view('Plantillas/header').view('calculadora', $data).view('Plantillas/footer');

        return $vistas;
    }

    public function operaciones(){

        $num1 = $_POST['num1'];
        $num2 = $_POST['num2'];

        if ($_POST['opciones'] == "suma") {
            $resultado = $num1 + $num2;
        }

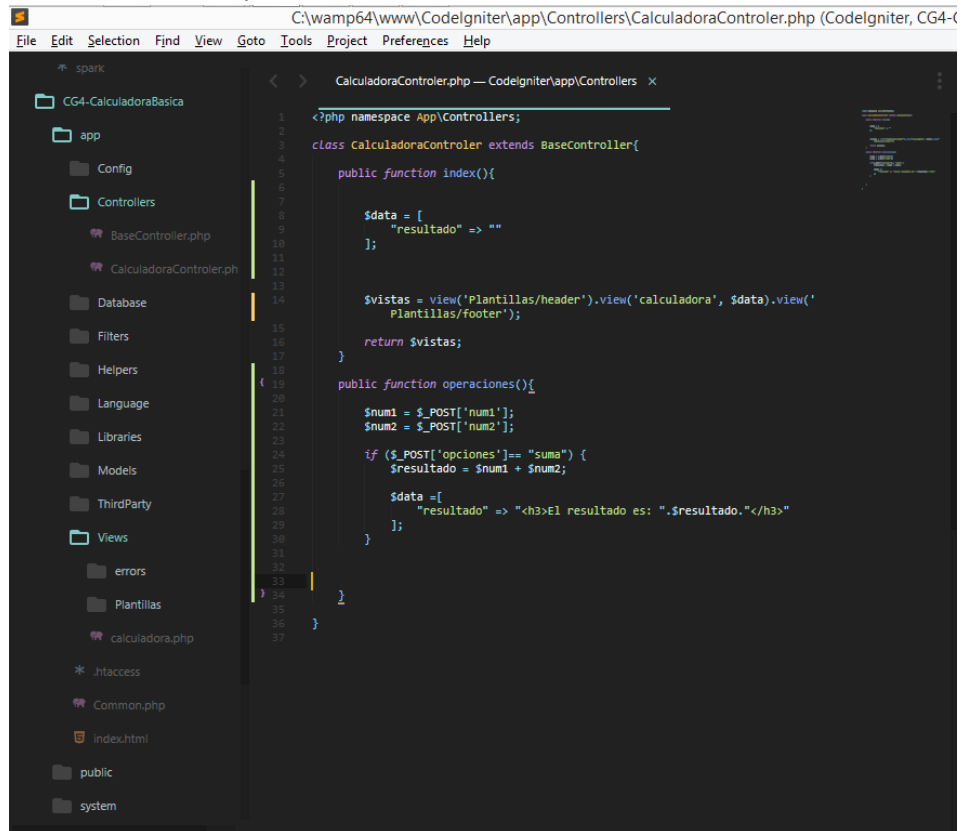
    }

}
```

operacion de los numero que llegaron de la vista calculadora

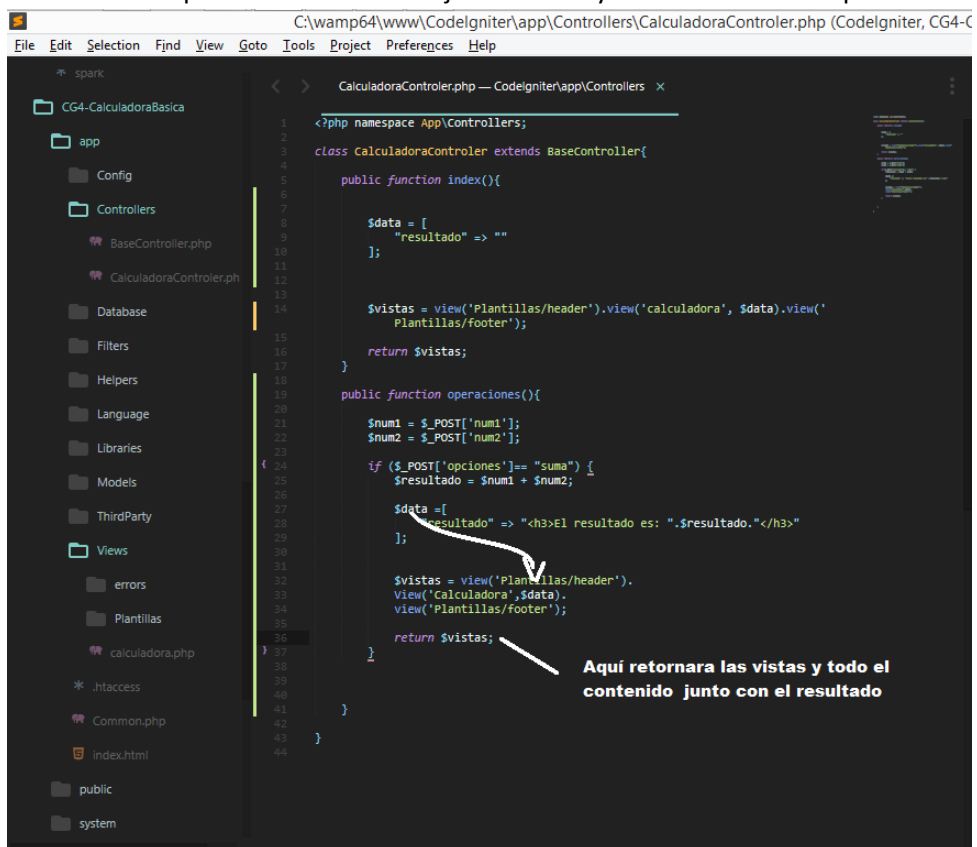
Variable creada

15. Crearemos nuevamente una variable llamada \$data la cual contendrá un array con la etiqueta resultado y esta tendrá un mensaje de “El resultado es: ” y concatenaremos con un punto la variable \$resultado la cual contiene el resultado de nuestra operación.



```
1 <?php namespace App\Controllers;
2
3 class CalculadoraController extends BaseController{
4
5     public function index(){
6
7         $data = [
8             "resultado" => ""
9         ];
10
11         $vistas = view('Plantillas/header').view('calculadora', $data).view('Plantillas/footer');
12
13         return $vistas;
14     }
15
16     public function operaciones(){
17
18         $num1 = $_POST['num1'];
19         $num2 = $_POST['num2'];
20
21         if ($_POST['opciones'] == "suma") {
22             $resultado = $num1 + $num2;
23
24             $data = [
25                 "resultado" => "<h3>El resultado es: ".$resultado."</h3>"
26             ];
27         }
28     }
29 }
```

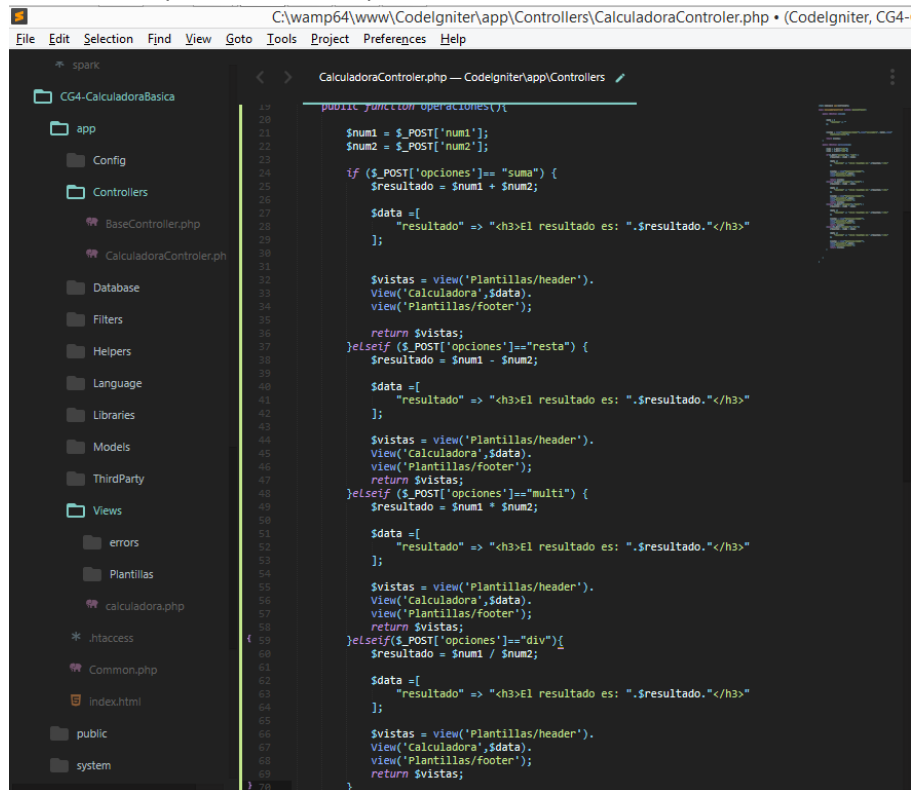
16. Crearemos una variable llamada vistas donde contendrá las vistas (header, calculadora y footer) y a su vez le incluiremos la variable data que contiene el mensaje a mostrar y el resultado de la operación.



```
1 <?php namespace App\Controllers;
2
3 class CalculadoraController extends BaseController{
4
5     public function index(){
6
7         $data = [
8             "resultado" => ""
9         ];
10
11         $vistas = view('Plantillas/header').view('calculadora', $data).view('Plantillas/footer');
12
13         return $vistas;
14     }
15
16     public function operaciones(){
17
18         $num1 = $_POST['num1'];
19         $num2 = $_POST['num2'];
20
21         if ($_POST['opciones'] == "suma") {
22             $resultado = $num1 + $num2;
23
24             $data = [
25                 "resultado" => "<h3>El resultado es: ".$resultado."</h3>"
26             ];
27
28             $vistas = view('Plantillas/header').
29                 view('calculadora', $data).
30                 view('Plantillas/footer');
31
32             return $vistas;
33         }
34     }
35 }
```

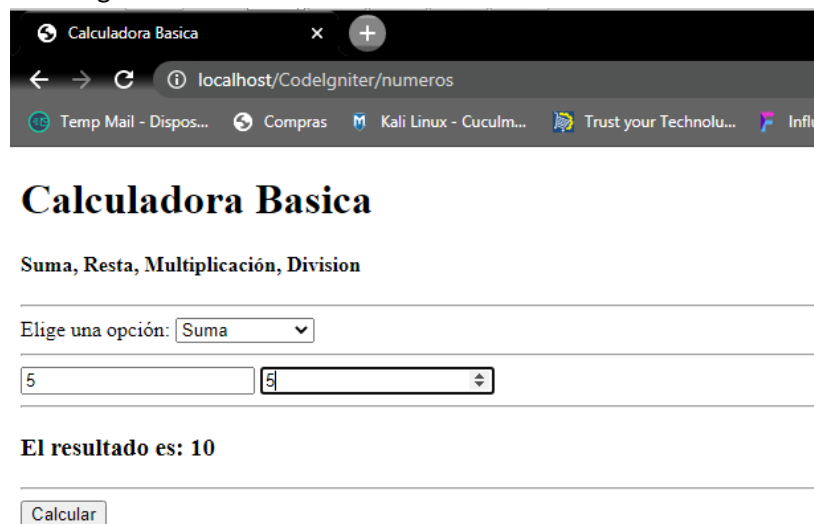
Aquí retornara las vistas y todo el contenido junto con el resultado

17. Así agregaremos los demás if para las demás opciones



```
19 public function operaciones(){
20
21     $num1 = $_POST['num1'];
22     $num2 = $_POST['num2'];
23
24     if ($_POST['opciones']=="suma") {
25         $resultado = $num1 + $num2;
26
27         $data=[
28             "resultado" => "<h3>El resultado es: ".$resultado."</h3>"
29         ];
30
31         $vistas = view('Plantillas/header');
32         view('Calculadora',$data);
33         view('Plantillas/footer');
34
35         return $vistas;
36     }elseif ($_POST['opciones']=="resta") {
37         $resultado = $num1 - $num2;
38
39         $data=[
40             "resultado" => "<h3>El resultado es: ".$resultado."</h3>"
41         ];
42
43         $vistas = view('Plantillas/header');
44         view('Calculadora',$data);
45         view('Plantillas/footer');
46         return $vistas;
47     }elseif ($_POST['opciones']=="multi") {
48         $resultado = $num1 * $num2;
49
50         $data=[
51             "resultado" => "<h3>El resultado es: ".$resultado."</h3>"
52         ];
53
54         $vistas = view('Plantillas/header');
55         view('Calculadora',$data);
56         view('Plantillas/footer');
57         return $vistas;
58     }elseif($_POST['opciones']=="div"){
59         $resultado = $num1 / $num2;
60
61         $data=[
62             "resultado" => "<h3>El resultado es: ".$resultado."</h3>"
63         ];
64
65         $vistas = view('Plantillas/header');
66         view('Calculadora',$data);
67         view('Plantillas/footer');
68         return $vistas;
69     }
70 }
```

18. Así se visualizaría en el navegador



Calculadora Basica

localhost/CodeIgniter/numeros

## Calculadora Basica

Suma, Resta, Multiplicación, Division

Elige una opción:

El resultado es: 10

19. Si le agregamos estilos a lo que es nuestro html se vería de la siguiente manera.

**Calculadora Basica**  
Suma, Resta,  
Multiplicación, Division

---

Elige una opción:

Suma ▼

---

5

5| ▲▼

---

**El resultado es: 10**

---

CALCULAR

EL código de este proyecto se encuentra en guthub.