

Como enviar correos con laravel 5.1

1) Lo primero que se debe hacer es modificar el archivo **.env**; las variables concernientes al envío de correos; la cuales son:

MAIL_DRIVER => esta variable define el driver a usar para el envío de los correos (Ej: smtp);

MAIL_HOST => aquí se define el hosting de servicio para el envío del correo (Ej: smtp.zoho.com);

MAIL_PORT => variable que define el numero del puerto, puede variar dependiendo del método de encriptación y el cliente de correos que utilice.

MAIL_USERNAME => Usuario de la cuenta de correos que se utilizara.

MAIL_PASSWORD => Contraseña de la cuneta de correos.

MAIL_ENCRYPTION => método de encriptación.

2) Luego se verifica el archivo **mail.php** que se encuentra ubicado en la carpeta **/config/** debemos ver que se este llamando a la función **env()** en cada una de las variables que nos compete; por ejemplo:

'driver' => `env('MAIL_DRIVER', 'smtp'),` //Esta es la manera correcta

'driver' => `'smtp',` //De esta manera no se están utilizando las variables que definimos en el archivo **.env**

Aquí se deben verificar las variables “**driver, host, port, encryption, username, password, pretend**”.

3) Una vez verificado todas la variables, tenemos varias formas para empezar con la construcción del E-mail en este caso, y como recomendación, crearemos un controlador para el envío de correos.

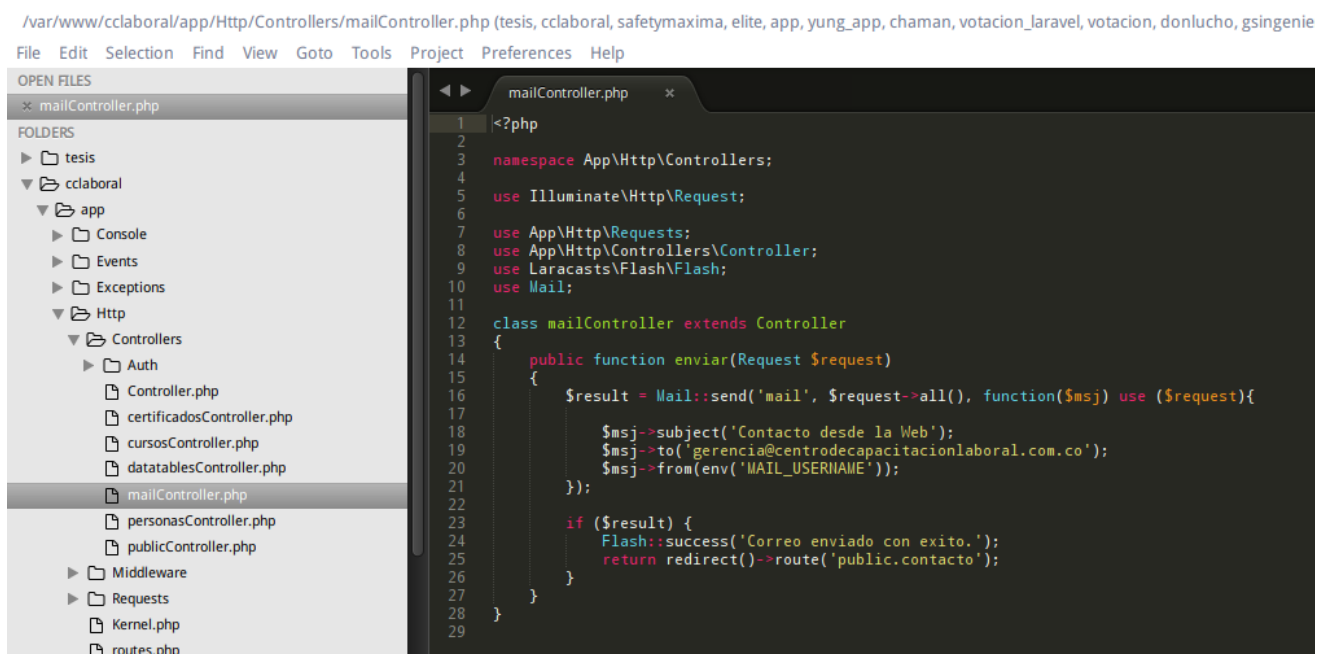


Ilustración 1: Ejemplo Tomado del sistema de Centro de Capacitacion Laboral

Como se muestra en la Ilustración anterior, se hace el llamado a la clase **Mail** y luego dentro del método se declara una variable **\$result** y se hace una instancia de la clase **Mail** trabajando directamente con el método **send** el cual recibe 3 parámetros.

El primero es **la vista** en la cual se estructura el correo electrónico, el segundo es **el request** que contiene todos los campos que se quieren enviar a la vista para la estructuración del mensaje, como tercer parámetro se envía una **función anónima** la cual a demás usa la variable **\$request**.

Dentro de la función se definen elementos básicos de un correo electrónico y que son necesarios para su envío exitoso, **\$msj → to()**, con este método se define a quien se quiere enviar el correo, **\$msj → from()**, con este método se define desde donde se esta enviando el correo recomendando que sea el mismo usuario que se definió anteriormente en el archivo **.env**; por ultimo el método **\$msj → subject()** que es utilizado para agregar el sujeto al correo.

Nota: Para ver más métodos se puede consultar la documentación oficial de laravel 5.1; <https://laravel.com/docs/5.1/mail#sending-mail>

La variable **\$result** se puede tratar como variable booleana la cual podemos validar con un **if()** para saber si el correo fue enviado o no.

4) Ahora pasamos a la vista, en ella construimos la estructura del correo.

Ilustración 2: Ejemplo Tomado del sistema de Centro de Capacitación Laboral

Como se puede apreciar en la ilustración anterior no es necesario llamar a la variable **\$request** podemos usar directamente los atributos de la misma como si fueran variables.

5) Para crear un pdf y enviarlo por correo lo hacemos de la siguiente manera

ller.php (tesis, cclaboral, safetymaxima, elite, app, yung_app, chaman, votacion_laravel, votacion, donlucho, gsingenieria, antv2, domestic, inmobiliaria) - Sublime

```
Project Preferences Help
pdfController.php
28 public function pedidos($id)
29 {
30     $pedido = $this->getDataPedidos($id);
31     $view = \View::make('pdf.pdfpedidos', compact('pedido'))->render();
32     $pdf = \App::make('dompdf.wrapper');
33
34     $pdf->loadHTML($view);
35
36     $pdf = $pdf->output();
37
38     $archivo_nombre = $pedido->cliente->nombre."_" . $pedido->cliente->apellido."_" . date("d-m-Y_s").".pdf";
39
40     $ruta = "pdf/" . $archivo_nombre;
41     $nombre = $pedido->proveedor->nombre;
42     $apellido = $pedido->proveedor->apellido;
43     $email = $pedido->proveedor->email;
44     if ($email==null) {
45         Flash::error('El proveedor ' . $nombre . ' ' . $apellido . ' no tiene un correo asignado para enviar el pedido.'
46     );
47         return redirect()->route('pedido.configurar');
48     }
49     $mensaje = $nombre." " . $apellido."", nuevo pedido de Maedca Cars.";
50     $fecha = $pedido->created_at;
51
52     $contenido = ['nombre' => $nombre, 'apellido' => $apellido, 'ruta' => $ruta, 'email' => $email, 'mensaje' => $
53     mensaje];
54     file_put_contents($ruta, $pdf);
55
56     $result = Mail::send('mail', $contenido, function($msj) use ($contenido)
57     {
58         $msj->subject($contenido['mensaje']);
59         if (\Auth::user()->tipouser->id == 2) {
60             $msj->to($contenido['email'])->cc('ventas@maedcacars.com.ve')->cc(\Auth::user()->cliente->email);
61         } else {
62             $msj->to($contenido['email'])->cc('ventas@maedcacars.com.ve');
63         }
64
65         $msj->from(env('MAIL_USERNAME'));
66         $msj->attach($contenido['ruta'], array(
67             'as' => 'Pedido.pdf',
68             'mime' => 'application/pdf'
69         ));
70     });
71
72     Flash::success('Pedido enviado con éxito.');
```

Ilustración 3: Ejemplo Tomado del sistema de Maedcacars

En este caso el PDF se genera igual a como se se hacen con **Dompdf**, por supuesto debes tener la librería instalada*, solo que en vez de utilizar el método **stream()** utilizaremos el método **output()** el cual genera el PDF pero como un archivo que puede ser guardado en el servidor y eso es justo lo que hacemos; guardamos el **archivo.pdf** en el servidor y conociendo esa ruta lo adjuntamos al mensaje que sera enviado.

Para eso nos vamos a la función anónima que se pasa como tercer parámetro en la instancia de la clase Mail y el método send. Dentro utilizamos el método **\$msj → attach()** este método acepta 2 parámetros, el primero en el cual se le pasa la ruta en la que se encuentra el archivo y el segundo es un array donde se pueden colocar diferentes opciones como un nombre para el archivo y el tipo MIME del mismo, sugiero agregarlos.

Hecho esto podemos validar con un condicional **if()** si nuestro correo se envio o no.

Ademas sugiero borrar el archivo pdf que se creo en el servidor para evitar la saturación del mismo, esto lo podemos hacer utilizando la función de php **unlink()**; a la cual le debemos pasar la ruta del archivo que queremos eliminar en este caso la del pdf. Todo esto antes de hacer **return**.

***Nota:** para ver como instalar y utilizar Dompdf se puede referir a la documentación oficial de Dompdf para laravel. <https://github.com/barryvdh/laravel-dompdf>