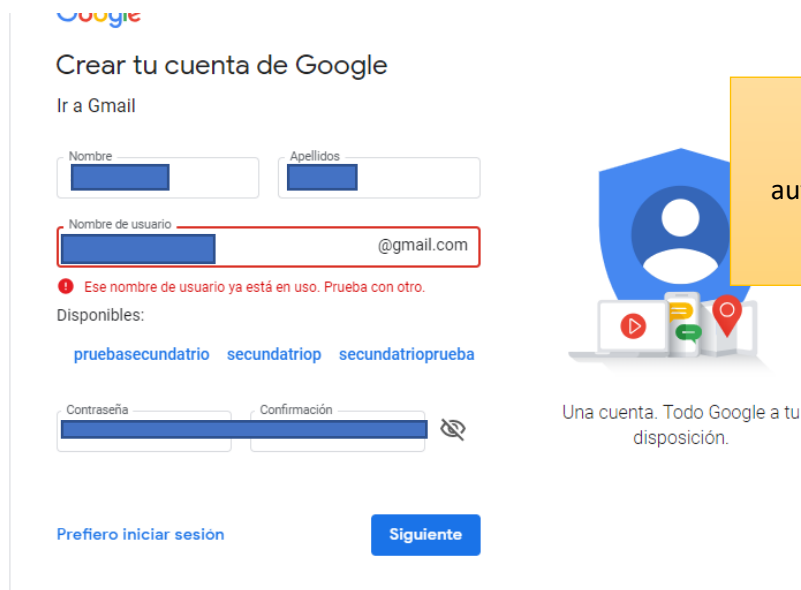


Para finalizar el prototipo vamos a incorporar la funcionalidad de envío de correo automatizado. La idea es crear una tarea programada que se ejecutará una vez al día. Esta tarea llamará a un script PHP que hará una consulta a la BBDD para comparar la fecha actual con el campo fecha_nacimiento de cada cliente. Si coinciden enviará una felicitación por correo con el nombre personalizado.

Ejercicio 1. Envío de correo desde URL

1. Vamos a utilizar [gmail como pasarela de correo](#). Créate una cuenta de prueba, en la cual tienes que habilitar "Permitir el acceso de aplicaciones menos seguras", así como la autenticación en dos pasos.



Google

Crear tu cuenta de Google

Ir a Gmail

Nombre Apellidos

Nombre de usuario @gmail.com

! Ese nombre de usuario ya está en uso. Prueba con otro.

Disponibles:

[pruebasecundatrio](#) [secundatriop](#) [secundatriopueba](#)

Contraseña Confirmación

[Prefiero iniciar sesión](#) [Siguiente](#)

Una cuenta. Todo Google a tu disposición.

Creamos una cuenta en GMAIL, posteriormente habilitamos la autenticación en 2 pasos y generamos una clave para la aplicación

```
REDIS_PORT=6379
MAIL_DRIVER=smtp
MAIL_HOST=smtp.gmail.com
MAIL_PORT=587
MAIL_USERNAME= CORREO Creado
MAIL_PASSWORD= Contraseña de aplicación generada por Google
MAIL_ENCRYPTION=tls
```

En nuestro proyecto de laravel nos iremos hasta el archivo ".env" y configuramos la conexión.

2. Una vez configurado gmail ayúdate de la siguiente [explicación](#) para enviar un correo de prueba desde tu aplicación. Esto nos permitirá probar que somos capaces de enviar correo desde una ruta de nuestra app.

Creamos el MailController

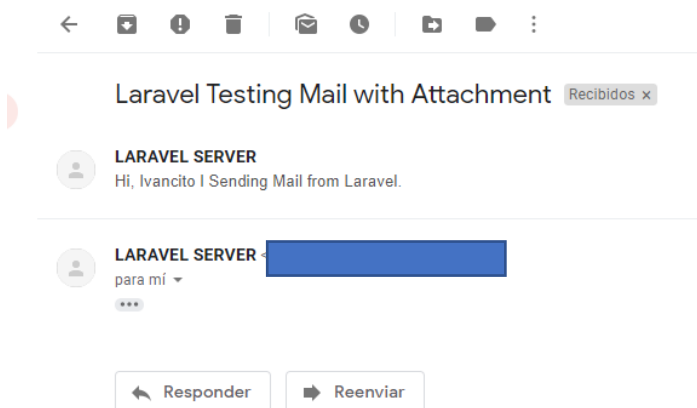
```
vagrant@homestead:~/code/felicitaciones$ artisan make:controller MailController
Controller created successfully.
vagrant@homestead:~/code/felicitaciones$ |
```

Configuramos el "routes/web.php"

```
Route::get('sendattachmentemail', 'MailController@attachment_email');
```

Finalizado el tutorial, accedemos a la URL y vemos como nos ha llegado el mensaje al correo.

```
public function attachment_email($data) {
    echo($data);
    //$datosEmail = array('name'=> $data->nombre);
    $datosEmail = array('name' => $data->nombre, 'fecha' => $data->fecha_nacimiento);
    Mail::send('mail', $datosEmail, function($message) {
        $message->to([REDACTED] 'Tutorials Point')->subject
            ('Feliz Cumpleaños');
        // $message->attach();
        //$message->attach('C:\laravel-master\laravel\public\uploads\test.txt');
        $message->from([REDACTED] 'LARAVEL SERVER');
    });
    echo "Email Sent with attachment. Check your inbox.";
}
```



3. Añade la lógica necesaria para que sólo se envíe el correo el día del cumpleaños y compruébalo.

```
public function enviaCorreo(){  
  
    $cliente = Cliente::All();  
  
    $hoy = date("m-d");  
    //echo $hoy;  
    foreach ($cliente as $c) {  
  
        $fecha = substr($c->fecha_nacimiento,5);  
  
        if($hoy == $fecha){  
            // echo('si cumple');  
            $mail = new MailController();  
            $data = $c;  
            $mail->attachment_email($data);  
        }  
    }  
}
```

Tenemos en nuestra Base de datos un registro que cumple hoy día 6.

| + Opciones | | | | | | | | | |
|---|---|---|---|--------|----------|------------------------------|------------|-------------------|---------------------|
| <div><div><div>←</div><div>→</div><div>↕</div></div></div> | | | | | | | | | |
| | | | id | nombre | imagen | fecha_nacimiento | correo | created_at | updated_at |
| <input type="checkbox"/> |  |  |  | 23 | Trinitys | trinity.jpg | 2019-01-03 | jesus@gmail.com | 2019-01-30 00:25:53 |
| <input type="checkbox"/> |  |  |  | 24 | Morfeo | morfeo.jpg | 2019-01-30 | fer@fer.es | 2019-01-30 00:26:30 |
| <input type="checkbox"/> |  |  |  | 25 | nuevos | Captura de pantalla (39).png | 2019-02-04 | fer@fer.es | 2019-01-30 10:21:25 |
| <input type="checkbox"/> |  |  |  | 26 | sergio | Captura de pantalla (10).png | 1988-02-05 | iherfue@gmail.com | 2019-02-05 23:07:49 |
| <input type="checkbox"/> |  |  |  | 27 | ivan | NULL | 1999-02-06 | fer@fer.es | 2019-02-06 00:01:21 |
| <div><div><div><input type="checkbox"/> Seleccionar todo</div><div>Para los elementos que están marcados:</div><div></div><div></div><div></div><div></div></div></div> | | | | | | | | | |

LARAVEL SERVER para mí 0:02 (hace 0 minutos) ☆ ↶

Hola ivan hoy es un día muy especial para ti, es tu cumpleaños y queremos felicitarte



Responder

Reenviar

Ejercicio 2. Creación del comando y planificador

Lo que nos interesa realmente es que el envío se haga de forma automática. Para ello tendremos que hacer dos cosas:

1. Crear un script como comando.
2. Incluir el comando en el planificador de Laravel (tendrás que utilizar crontab -e para añadir el comando al final del tutorial).

En el siguiente [tutorial](#) se explica cómo hacerlo.

Generamos el comando

```
vagrant@homestead:~/code/pruebas_rutas$ artisan make:command EnviaCorreo
Console command created successfully.
vagrant@homestead:~/code/pruebas_rutas$ |
```

Configuramos el archivo

```
1  <?php
2
3  namespace App\Console\Commands;
4
5  use Illuminate\Console\Command;
6  use App\Cliente;
7  use App\Http\Controllers\MailController;
8
9  class EnviaCorreo extends Command
10 {
11     /**
12      * The name and signature of the console command.
13      *
14      * @var string
15      */
16     protected $signature = 'send:enviaCorreo';
17
18     /**
19      * The console command description.
20      *
21      * @var string
22      */
23     protected $description = 'Envia un correo si es su cumpleaños';
24
25     /**
```

Comando y descripción

```
public function handle()
{
    $cliente = Cliente::All();

    $hoy = date("m-d");
    //echo $hoy;
    foreach ($cliente as $c) {

        $fecha = substr($c->fecha_nacimiento,5);

        if($hoy == $fecha){
            // echo('si cumple');
            $mail = new MailController();
            $data = $c;
            $mail->attachment_email($data);
        }
    }
}
```

Una vez ejecutado el comando, ejecutará lo que tengamos en "handle"

Ejecutamos “artisan” en consola y vemos el comando.

```
Route:list          List all registered routes
schedule
schedule:run        Run the scheduled commands
send
send:enviaCorreo    Envía un correo si es su cumpleaños
session
session:table       Create a migration for the session database
```

Configuramos el archivo “Kernel.php”

```
1  <?php
2
3  namespace App\Console;
4
5  use Illuminate\Console\Scheduling\Schedule;
6  use Illuminate\Foundation\Console\Kernel as ConsoleKernel;
7  use File;
8
9  class Kernel extends ConsoleKernel
10 {
```

Hacemos uso de la clase File

El siguiente método guardará en un log el resultado de la ejecución del comando.

```
protected function schedule(Schedule $schedule)
{
    $cronLog = storage_path('logs/cron.log');
    if (!File::exists($cronLog)) {
        File::put($cronLog, '');
    }
    $schedule->command('send:enviaCorreo')->everyMinute()->withoutOverlapping()->appendOutputTo($cronLog)
}

/**
 * Register the commands for the application.
 */
```

Ahora Iniciamos el crontab mediante crontab -e

Seleccionamos la opción 1 y copiamos la siguiente línea

```
* * * php /home/vagrant/code/pruebas_rutas/artisan schedule:run >> /dev/null 2>&1
```

```
no crontab for vagrant - using an empty one

Select an editor. To change later, run 'select-editor'.
 1. /bin/nano        <---- easiest
 2. /usr/bin/vim.basic
 3. /usr/bin/vim.tiny
 4. /bin/ed

Choose 1-4 [1]: 1
crontab: installing new crontab
vagrant@homestead:~/code/pruebas_rutas$ |
```

Guardamos y vemos el log generado



Por último, comprobamos que el email se envía cada minuto.

