

Manual de Usuario - Laravel

24/01/2020

Miguel Ángel López Sánchez

2º DAM

DWESE

Índice

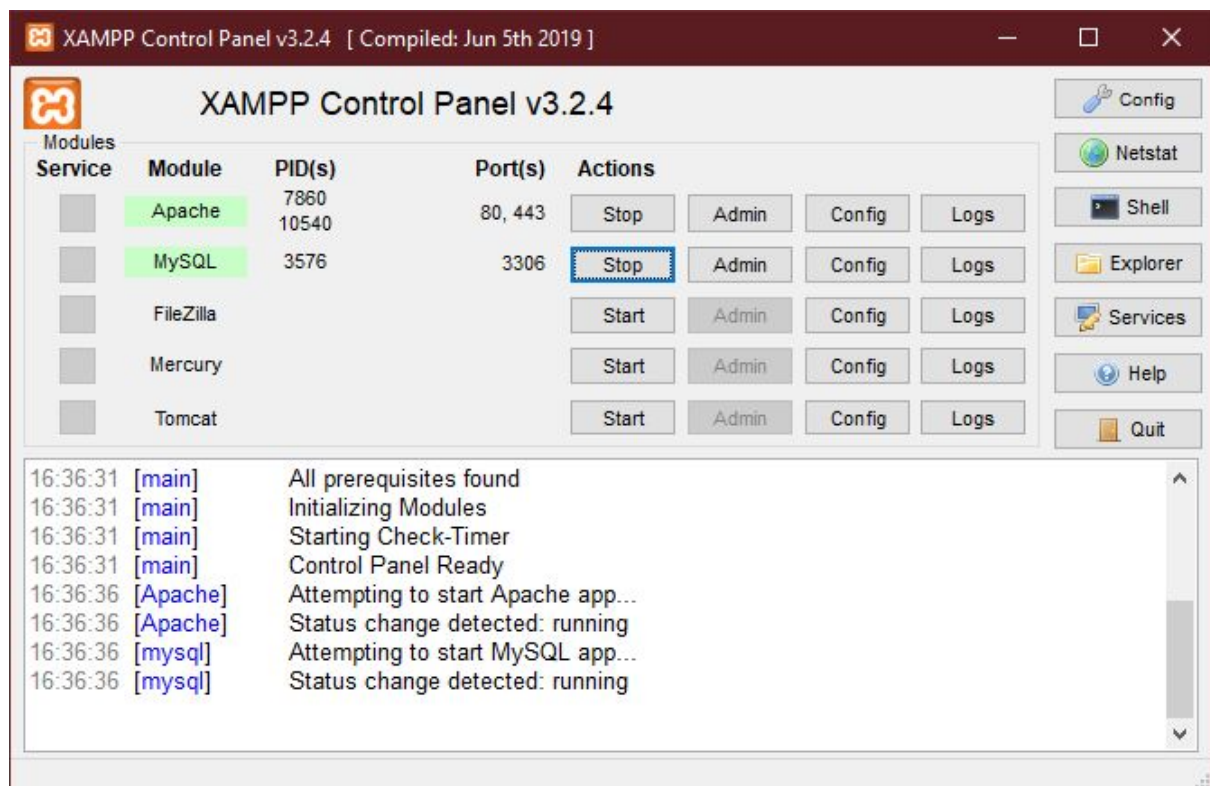
Instalación	2
OPCIONAL	4
Creación del Nuevo Proyecto	5
Activar Servidor Laravel	5
Creación de alias a la carpeta public	6
Creación del modelo, migración, controlador, vistas	9
Configuración de la Base de Datos	10
Crear database	10
Modificar .env	11
Modificar campos de la tabla Alumnos	12
Migrar a la base de datos	13
Generar Datos de Prueba	14
Configuración de Factory con Faker	14
Configuración del Seeder	14
Actualizar la base de datos con los datos de la tabla	14
Comprobar que la tabla ha sido rellena	15
Modificando las rutas (web.php)	15
Métodos del Controlador	15
Mostrar todos (Paginación)	15
Destroy	16
Hoja de estilo propia	16
Creación de plantillas	17
Creación de vistas	18
Comprobamos que todo funcione en la web	20
Listado	20
Paginación	20
Borrado	21

Instalación

Antes de instalar [Laravel](#) en su sistema debe tener en cuenta los siguientes requisitos:

- [PHP](#) >= 7.2.0
- BCMath PHP Extension
- CType PHP Extension
- JSON PHP Extension
- Mbstring PHP Extension
- OpenSSL PHP Extension
- PDO PHP Extension
- Tokenizer PHP Extension
- XML PHP Extension

Para la instalación de [PHP](#) se recomienda usar algún **programa de empaquetados** como puede ser [MAMP](#) o [XAMPP](#), para este tutorial yo estaré usando este último.



También será necesario que previamente tengas instalado [Composer](#), el cual es un **manejador de dependencias** y también es el que te permitirá que instales [laravel](#).



En este tutorial no se mostrará la instalación de [Composer](#) puesto que se centrará en [Laravel](#).

Una vez tengas completados todos los pasos previos, es el turno de abrir la terminal (si estas en Windows como yo, será CMD) donde se realizará el resto del procedimiento.

Si lo deseas, puedes comprobar que [Composer](#) se haya instalado correctamente:

```
C:\Users\loboz>composer

          _
         / \
        /___\
       /_____\
      /_____\
     /_____\
    /_____\
   /_____\
  /_____\
 /_____\
/_____\

Composer version 1.9.2 2020-01-14 16:30:31

Usage:
  command [options] [arguments]
```

OPCIONAL

Finalmente, para completar la instalación de [laravel](#), utiliza el siguiente comando:

```
composer global require laravel/installer
```

```
D:\xampp\htdocs\DWSE\ejercicios>composer global require laravel/installer
Changed current directory to C:/Users/loboz/AppData/Roaming/Composer
Using version ^3.0 for laravel/installer
./composer.json has been created
Loading composer repositories with package information
Updating dependencies (including require-dev)
Package operations: 14 installs, 0 updates, 0 removals
- Installing symfony/process (v5.0.3): Downloading (100%)
- Installing symfony/polyfill-ctype (v1.13.1): Downloading (100%)
- Installing symfony/filesystem (v5.0.3): Downloading (100%)
- Installing psr/container (1.0.0): Downloading (100%)
- Installing symfony/service-contracts (v2.0.1): Downloading (100%)
- Installing symfony/polyfill-php73 (v1.13.1): Downloading (100%)
- Installing symfony/polyfill-mbstring (v1.13.1): Downloading (100%)
- Installing symfony/console (v5.0.3): Downloading (100%)
- Installing ralouphie/getallheaders (3.0.3): Downloading (100%)
- Installing psr/http-message (1.0.1): Downloading (100%)
- Installing guzzlehttp/psr7 (1.6.1): Downloading (100%)
- Installing guzzlehttp/promises (v1.3.1): Downloading (100%)
- Installing guzzlehttp/guzzle (6.5.2): Downloading (100%)
- Installing laravel/installer (v3.0.1): Downloading (100%)
symfony/service-contracts suggests installing symfony/service-implementation
symfony/console suggests installing symfony/event-dispatcher
symfony/console suggests installing symfony/lock
symfony/console suggests installing psr/log (For using the console logger)
guzzlehttp/psr7 suggests installing zendframework/zend-httphandler (Emit PSR-7 responses)
guzzlehttp/guzzle suggests installing psr/log (Required for using the Log middleware)
guzzlehttp/guzzle suggests installing ext-intl (Required for Internationalized Domain Name (IDN) support)
Writing lock file
Generating autoload files
```

Esto le permitirá usar el comando `laravel new DirName`, el cual permite crear una nueva instalación de [Laravel](#) con todas las dependencias ya instaladas, en el directorio que le indique, siendo `DirName` el nombre o ruta del directorio.

Creación del Nuevo Proyecto

Para crear un **nuevo proyecto** con [Laravel](#) debes utilizar el siguiente comando:

```
composer create-project laravel/laravel mi-proyecto-laravel
```

Debes cambiar **mi-proyecto-laravel** por el nombre que le vayas a dar a la carpeta del proyecto, o la ruta en el caso de que no estés ubicado en el directorio donde lo quieras crear.

No te preocupes si tarda en ejecutar el comando.

```
D:\xampp\htdocs\DWSE\ejercicios>composer create-project laravel/laravel academia
Installing laravel/laravel (v6.12.0)
- Installing laravel/laravel (v6.12.0): Downloading (100%)
Created project in academia
> @php -r "file_exists('.env') || copy('.env.example', '.env');"
Loading composer repositories with package information
Updating dependencies (including require-dev)
Package operations: 86 installs, 0 updates, 0 removals
```

Activar Servidor Laravel

Primero dirígete, usando la consola de comandos, al directorio donde tengas el proyecto de [Laravel](#) que estas utilizando y, después, utiliza el siguiente comando:

```
php artisan serve
```

```
D:\xampp\htdocs\DWSE\ejercicios>cd academia
D:\xampp\htdocs\DWSE\ejercicios\academia>php artisan serve
Laravel development server started: http://127.0.0.1:8000
```

Si vas a la dirección que te indica en la terminal, podrás ver que [Laravel](#) está funcionando correctamente.



Laravel

DOCS LARACASTS NEWS BLOG NOVA FORGE VAPOR GITHUB

Si quieres cerrar en algún momento este servicio, basta con que, en la terminal, pulses las teclas: **Ctrl+C**.

Creación de alias a la carpeta public

En el apartado anterior has visto que, encendiendo el servidor de Laravel y accediendo a `127.0.0.1:8000` entrabas a la página inicial de tu proyecto de Laravel, pero también puedes acceder a esta si te diriges a la carpeta **public** de tu proyecto, aunque no tengas el servicio de Laravel activado (aún que seguirás necesitando que el servidor apache si este encendido). La ruta sería algo similar a la siguiente:

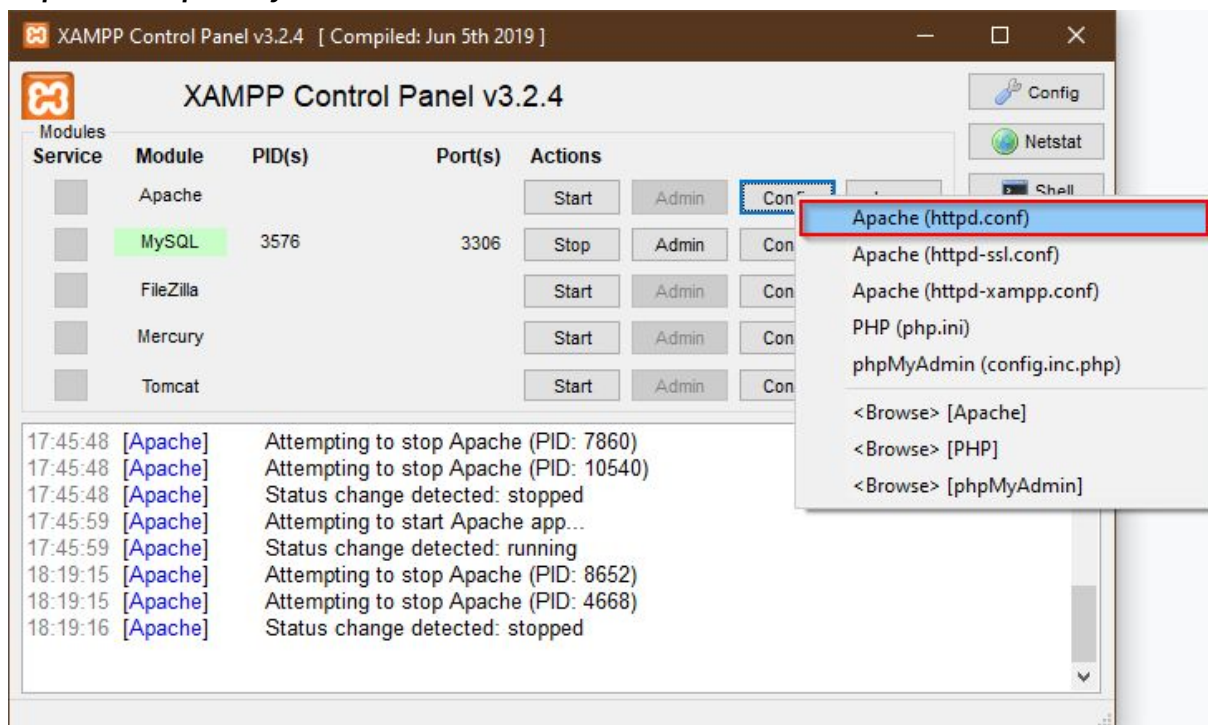
`http://localhost/DWESE/ejercicios/academia/public/`

Sin embargo esta ruta es muy larga, así que lo siguiente que explicaré será como puedes crear un alias para la carpeta **public** de tu proyecto, de esta forma acortarás bastante la ruta.


Como yo estoy utilizando el empaquetado [XAMPP](#) lo explicaré usando este programa de ejemplo.

Detén el servidor Apache mientras haces las siguientes configuraciones para el alias.

A la derecha de cada módulo de [XAMPP](#) podrás ver una serie de 4 botones, pulsa en el botón “**Config**” del módulo Apache y, posteriormente, en la opción “**Apache (httpd.conf)**”.



Al final del archivo de configuración que acabas de abrir, coloca lo siguiente:
Include "conf/alias/*.conf"



```
httpd.conf: Bloc de notas
Archivo Edición Formato Ver Ayuda
#
# Note: The following must must be present to support
#       starting without SSL on platforms with no /dev/random equivalent
#       but a statically compiled-in mod_ssl.
#
<IfModule ssl_module>
SSLRandomSeed startup builtin
SSLRandomSeed connect builtin
</IfModule>

# XAMPP: We disable operating system specific optimizations for a listening
# socket by the http protocol here. IE 64 bit make problems without this.

AcceptFilter http none
AcceptFilter https none
# AJP13 Proxy
<IfModule mod_proxy.c>
<IfModule mod_proxy_ajp.c>
Include "conf/extra/httpd-ajp.conf"
</IfModule>
</IfModule>
Include "conf/alias/*.conf"
```

Dirígete al siguiente directorio **xampp/apache/conf** y crea dentro la carpeta **alias**. Dentro de esta carpeta es donde crearás los archivos con los alias que creas convenientes.

Primero crea un archivo, **no importa como lo llaves**, y terminalo con la extensión **.conf**.

Después **añádele** el siguiente texto:

```
<Directory "c:\users\foo\programming\dev">
#
# Possible values for the Options directive are "None", "All",
# or any combination of:
#   Indexes Includes FollowSymLinks SymLinksifOwnerMatch ExecCGI MultiViews
#
# Note that "MultiViews" must be named *explicitly* --- "Options All"
# doesn't give it to you.
#
# The Options directive is both complicated and important. Please see
# http://httpd.apache.org/docs/2.2/mod/core.html#options
# for more information.
#
Options Indexes FollowSymLinks Includes ExecCGI
```



```
#
# AllowOverride controls what directives may be placed in .htaccess files.
# It can be "All", "None", or any combination of the keywords:
# Options FileInfo AuthConfig Limit
#
AllowOverride All

#
# Controls who can get stuff from this server.
#
Require all granted

</Directory>

Alias /ruta_url "ruta_local"
```

Cambia el texto en **rojo**, **/ruta_url** será el **alias** que le daremos a la **ruta_local** que será la ruta donde se encuentra la **carpeta public** de nuestro proyecto.

Ejemplo de alias:

```
Alias /academia
"D:\xampp\htdocs\DWSESE\ejercicios\academia\public"
```

Ahora vuelve a **encender el servidor Apache** y comprueba que el alias te funcione.



Creación del modelo, migración, controlador, vistas

Para crear un modelo completo, dirígete a la ruta de tu proyecto Laravel y utiliza el siguiente comando:

```
php artisan make:model modelName -mcrfs
```

o el siguiente comando:

```
php artisan make:model modelName -a
```

Nota: Si utilizas este último deberás usar también el siguiente comando:

```
php artisan make:seed modelName
```

modelName → Nombre del **modelo** a crear (deberás cambiarlo por el que corresponda a tu caso)

-m → Crea la **migración**

-c → Crea el **controlador**

-r → Llena el controlador de **recursos**

-f → Crea la fábrica o factoría (**factory**)

-s → Crea las sembradoras (**seeders**)

-a → Crea todo: **migración, controlador, recursos y factoría**

```
D:\xampp\htdocs\DWEE\ejercicios\academia>php artisan make:model Alumnos -mcrfs
Model created successfully.
Factory created successfully.
Created Migration: 2020_01_23_180625_create_alumnos_table
Seeder created successfully.
Controller created successfully.
```

Configuración de la Base de Datos

Crear database

En primer lugar tienes que acceder a mysql.

```
Setting environment for using XAMPP for Windows.
loboz@DESKTOP-48P4AMO d:\xampp
# mysql -u root
Welcome to the MariaDB monitor.  Commands end with ; or \g.
Your MariaDB connection id is 12
Server version: 10.4.10-MariaDB mariadb.org binary distribution

Copyright (c) 2000, 2018, Oracle, MariaDB Corporation Ab and others.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

MariaDB [(none)]>
```

Ahora crea la base de datos, de nombre puedes ponerle academia, por ejemplo.

```
MariaDB [(none)]> create database academia;
Query OK, 1 row affected (0.006 sec)

MariaDB [(none)]> use academia;
Database changed
MariaDB [academia]>
```

Finalmente dale permisos a la base de datos al usuario que vayas a usar.

```
MariaDB [academia]> grant all privileges on academia.* to userpdo@'localhost';
Query OK, 0 rows affected (0.273 sec)

MariaDB [academia]> _
```

Si no tienes ningún usuario puedes crearte uno nuevo con el siguiente comando SQL:

```
CREATE USER userName@'localhost' IDENTIFIED BY 'contraseña';
```

Cambia el **texto** en rojo por el que se adapte en tu caso.

Por ejemplo para el usuario que estoy usando el comando quedaría así:

```
CREATE USER userpdo@'localhost' IDENTIFIED BY 'secreto';
```

Modificar .env

Dentro de tu proyecto, busca el archivo `.env` y localiza este bloque de texto:

```
DB_CONNECTION=mysql
DB_HOST=127.0.0.1
DB_PORT=3306
DB_DATABASE=laravel
DB_USERNAME=root
DB_PASSWORD=
```

Esta casi al principio.

Ahora **cambia los valores de las sentencias en rojo** de forma que se adapte a tu caso, por ejemplo en el mío el resultado sería el siguiente:

```
DB_CONNECTION=mysql
DB_HOST=127.0.0.1
DB_PORT=3306
DB_DATABASE=academia
DB_USERNAME=userpdo
DB_PASSWORD=secreto
```

Modificar campos de la tabla Alumnos

Dirígete a la carpeta de tu proyecto y **modifica el archivo** `database/migrations/XXXX_XX_XX_XXXXXX_create_alumnos_table.php` (las equis representan la **fecha en la que creaste la migración y un código** que puede ser diferente en su caso, busca el archivo que, en cuyo nombre contenga `"create_alumnos_table.php"`, es el que debes modificar).

Dentro de la función `up()` encontrarás las siguientes sentencias:

```
Schema::create('alumnos', function (Blueprint $table) {  
    $table->bigIncrements('id');  
    $table->timestamps();  
});
```

Modifica este bloque añadiendo el contenido que mostraré a continuación para que quede similar a lo siguiente:

```
Schema::create('alumnos', function (Blueprint $table) {  
    $table->bigIncrements('id');  
    $table->string('nombre', 60);  
    $table->string('apellidos', 120);  
    $table->string('email', 180)->unique();  
    $table->string('direccion', 220);  
    $table->string('telefono', 60)->nullable();  
    $table->timestamps();  
});
```

Ahora abre el archivo `app/Alumnos.php` y, dentro de la clase, añade la siguiente línea:

```
protected $fillable=["nombre", "apellidos", "email", "direccion",  
"telefono"];
```

Esto te permitirá añadir datos por formulario.

Migrar a la base de datos

Ahora, en la terminal y posicionado en el directorio de tu proyecto utiliza el siguiente comando:

```
php artisan migrate:fresh
```

Esto **borrará las migraciones anteriores y migrará de nuevo tu proyecto a la base de datos**, de esta forma actualizará las tablas.

```
D:\xampp\htdocs\DWSE\ejercicios\academia>php artisan migrate:fresh
Dropped all tables successfully.
Migration table created successfully.
Migrating: 2014_10_12_000000_create_users_table
Migrated: 2014_10_12_000000_create_users_table (0.79 seconds)
Migrating: 2014_10_12_100000_create_password_resets_table
Migrated: 2014_10_12_100000_create_password_resets_table (0.83 seconds)
Migrating: 2019_08_19_000000_create_failed_jobs_table
Migrated: 2019_08_19_000000_create_failed_jobs_table (0.68 seconds)
Migrating: 2020_01_23_202137_create_alumnos_table
Migrated: 2020_01_23_202137_create_alumnos_table (3.21 seconds)
```

Si te da algún **fallo de sintaxis**, comprueba que no te hayas equivocado al definir la tabla dentro del archivo `create_alumnos_table.php`.

Si te da un fallo indicando **que no encuentra la base de datos** y te indica una base de datos que no es la que tu has especificado anteriormente, deberás borrar la cache con el comando:

```
php artisan config:cache
```

```
D:\xampp\htdocs\DWSE\ejercicios\academia>php artisan config:cache
Configuration cache cleared!
Configuration cached successfully!
```

Generar Datos de Prueba

Configuración de Factory con Faker

Dirígete al archivo `database/factories/AlumnosFactory.php` y modifica el `$factory` utilizando `faker`, que será el framework que se encargue de generar los datos de prueba, debería quedarte algo así:

```
$factory->define(Alumnos::class, function (Faker $faker) {  
    return [  
        "nombre"=>$faker->firstName(),  
        "apellidos"=>$faker->lastName(),  
        "email"=>$faker->unique()->email(),  
        "direccion"=>$faker->streetAddress(),  
        "telefono"=>$faker->optional()->phoneNumber  
    ];  
});
```

Configuración del Seeder

Entra en el archivo `database/seeds/DatabaseSeeder.php` y dentro de la función pública `run()` añade la siguiente sentencia:

```
$this->call(AlumnosSeeder::class);
```

Ahora entra en el archivo `database/seeds/AlumnosSeeder.php` y dentro de la función pública `run()` añade las siguientes sentencias:

```
DB::table("alumnos")->truncate(); //vacía la tabla  
factory(Alumnos::class, 20)->create(); //crea 20 registros
```

En el mismo archivo, deberás indicar en la parte superior (justo debajo de `<?php`, donde se encuentra `use Illuminate\Database\Seeder;`) lo siguiente:

```
use App\Alumnos;
```

Actualizar la base de datos con los datos de la tabla

Para ello en la terminal debes escribir el siguiente comando:

```
php artisan db:seed
```

```
D:\xampp\htdocs\DWSE\ejercicios\academia>php artisan db:seed  
Seeding: AlumnosSeeder  
Seeded: AlumnosSeeder (2.65 seconds)  
Database seeding completed successfully.
```

No te preocupes si tarda al cargar

Comprobar que la tabla ha sido rellena

Comprobamos en mysql que la tabla alumnos tiene datos generados.

```
MariaDB [academia]> select * from alumnos;
```

id	nombre	apellidos	email	direccion	telefono	created_at	updated_at
1	Beatrice	Kunde	ybogan@feil.biz	3414 Streich Drive Suite 257	(225) 449-9884 x62756	2020-01-23 21:56:23	2020-01-23 21:56:23
2	Elmira	Waters	labadie.mary@hotmail.com	7930 Justina Flat	630-468-1065 x95398	2020-01-23 21:56:23	2020-01-23 21:56:23
3	Austen	Ritchie	hane.shea@hand.org	79529 Kiel Light	NULL	2020-01-23 21:56:23	2020-01-23 21:56:23
4	Aletha	Medhurst	xsimonis@gmail.com	63582 Ammand Street Suite 302	NULL	2020-01-23 21:56:23	2020-01-23 21:56:23
5	Edwina	Pouros	nelson.klein@hotmail.com	8770 Tia Branch Suite 050	+13286424772	2020-01-23 21:56:23	2020-01-23 21:56:23
6	Ellen	Hermiston	ncollier@walter.net	702 Bode Vista	NULL	2020-01-23 21:56:24	2020-01-23 21:56:24
7	Odell	Mosciski	jaylin89@collier.com	38563 Carolanne Harbor Suite 378	(519) 377-4475 x51128	2020-01-23 21:56:24	2020-01-23 21:56:24
8	Kristopher	Lind	kuhic.merritt@hotmail.com	4533 Ray Land	NULL	2020-01-23 21:56:24	2020-01-23 21:56:24
9	Melvin	Morissette	clyde.king@gmail.com	7819 Abbey Ridge Suite 162	NULL	2020-01-23 21:56:24	2020-01-23 21:56:24
10	Bridie	Lynch	heaney.keenan@koch.biz	95066 Wintheiser Keys Apt. 948	NULL	2020-01-23 21:56:24	2020-01-23 21:56:24
11	Cristal	Hudson	fhuel@yahoo.com	8463 Dare Landing	NULL	2020-01-23 21:56:24	2020-01-23 21:56:24
12	Yesenia	Lemke	fhalvorson@hotmail.com	234 Luella Hollow Apt. 023	+1-554-461-8963	2020-01-23 21:56:24	2020-01-23 21:56:24
13	Alta	Ernser	vivian.boehm@yahoo.com	136 Nader Plains	NULL	2020-01-23 21:56:24	2020-01-23 21:56:24
14	Jalyn	Harvey	jalen79@robel.org	755 Abernathy Walk	NULL	2020-01-23 21:56:24	2020-01-23 21:56:24
15	Margaret	Winkler	beatty.megan@connelly.com	30704 Kristofer Pine Apt. 805	960-904-0798	2020-01-23 21:56:24	2020-01-23 21:56:24
16	Orion	Dibbert	hintz.hillary@yahoo.com	70025 Langosh Lights Apt. 253	301-591-9553	2020-01-23 21:56:24	2020-01-23 21:56:24
17	Edwina	Hauk	destin.beier@gmail.com	206 Rippin Island	(585) 959-5878 x34604	2020-01-23 21:56:25	2020-01-23 21:56:25
18	Johnpaul	Goyette	abigail06@hotmail.com	863 Michaela Rest	NULL	2020-01-23 21:56:25	2020-01-23 21:56:25
19	Elmira	Mayert	kylie.bernhard@kris.org	176 Gorchany Flats Suite 631	203-323-5461	2020-01-23 21:56:25	2020-01-23 21:56:25
20	Tre	Hammes	turcotte.rico@deckow.biz	96267 Zieme Camp Apt. 535	NULL	2020-01-23 21:56:25	2020-01-23 21:56:25

20 rows in set (0.000 sec)

Modificando las rutas (web.php)

Abre el archivo **routes/web.php** y, añade al final, las siguientes sentencias:

```
Route::get("alumnos/listado",
"AlumnosController@mostrarTodos")->name("alumnos.listado");
Route::resource("alumnos", "AlumnosController");
```

Métodos del Controlador

Mostrar todos (Paginación)

Dirígete a **app/http/controllers/alumnoscontroller.php** y, debajo del método **index()**, crea el siguiente método:

```
public function mostrarTodos() {
    $alumnos=Alumnos::paginate(5);
    return view('alumnos.listado', compact('alumnos'));
}
```

Este método devuelve los alumnos **paginados** de 5 en 5 para que puedas mostrarlos en la vista que explicaré más adelante **'alumnos.listado'**.

Destroy

Busca, dentro del mismo archivo (app/http/controllers/alumnoscontroller.php) el método `destroy` (es el último) y añade dentro de él, las siguientes sentencias:

```
$alumno->delete();  
  
Session::flash('mensaje', 'Alumno Borrado Correctamente.');
```

Importante: Cambia en este método el argumento de `$alumnos` a `$alumno`, esto es para que no hay conflicto debido a que el compact del método `mostrarTodos` también es `alumnos`.

```
public function destroy(Alumnos $alumno)
```

Este método nos permitirá **borrar un alumno de la tabla**.

Importante: En `AlumnosController.php` verás que, al principio del archivo, se encuentran las siguientes sentencias:

```
use App\Alumnos;  
use Illuminate\Http\Request;
```

Añade, a continuación de estas, lo siguiente:

```
use Session;
```

Para que no te de fallos al usar el `Session` cuando borres un alumno.

Hoja de estilo propia

Dentro de la carpeta `public`, crea la carpeta `css` y dentro de ella un archivo llamado `style.css`, rellena el archivo con el siguiente contenido:

```
.normal, .grande, .extra{  
    font-family: 'Indie Flower', cursive;  
}  
  
.normal{  
    font-size: 1.1em;  
}  
  
.grande{  
    font-size: 1.6em;  
    font-weight: bolder;  
}  
  
.extra{  
    font-size: 2em;  
}
```

Estas son clases que utilizaremos para darle formato a la letra de la página.

Creación de plantillas

Dirígete a **resources/views** y crea la carpeta **plantillas**, dentro de esta tendrás los archivos **.blade.php** que vayas a usar de plantillas de forma más organizada.



Crema un archivo llamado **plantilla.blade.php** y añadele el siguiente código:

```
<!DOCTYPE html>
<html lang="es">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width,
initial-scale=1.0">
  <meta http-equiv="X-UA-Compatible" content="ie=edge">
  <title>@yield('titulo')</title>
  <link rel="stylesheet"
href="https://stackpath.bootstrapcdn.com/bootstrap/4.4.1/css/bootstra
p.min.css"
integrity="sha384-Vkoo8x4CGsO3+Hhvxv8T/Q5PaXtkKtu6ug5TOeNV6gBiFeWPGFN9
MuhOf23Q9Ifjh" crossorigin="anonymous">
  <link
href="https://fonts.googleapis.com/css?family=Indie+Flower&display=sw
ap" rel="stylesheet">
  <link href="{{asset('css/style.css')}}" rel="stylesheet">
</head>
<body style="background-color:bisque">
  <h3 class='text-center mt-3 grande'>@yield('cabecera')</h3>
  <div class="container mt-3">
    @yield('contenido')
  </div>
</body>
</html>
```

El contenido que pone '@yield()' será el que se modifique cuando se use la plantilla, es decir, será la **parte personalizada**, el resto será **igual para todas las páginas que utilicen esta plantilla**.

Creación de vistas

Dentro de **resources/views** crea una carpeta llamada **alumnos**, dentro de esta crea el archivo **listado.blade.php** y añádele el siguiente código:

```
{{--Plantilla a utilizar--}}
@extends('plantillas.plantilla')
{{--titulo de la página--}}
@section('titulo')
    Listado Alumnos
@endsection
{{--Texto en la etiqueta h1 (cabecera)--}}
@section('cabecera')
    ALUMNOS MATRICULADOS
@endsection
{{--Contenido de la página--}}
@section('contenido')
    {{--Comprueba si existe algún mensaje en Session y si lo hay lo muestra--}}
    @if(Session::has('mensaje'))
    <div class='container mt-3 mb-3 alert-success'>
        {{Session::get('mensaje')}}
    </div>
    @endif
    {{--Tabla que mostrará la información del alumno--}}
    <table class="table table-dark normal">
        <thead>
            <tr>
                {{--Celdas de la tabla--}}
                <th scope="col">Código</th>
                <th scope="col">Nombre</th>
                <th scope="col">Apellidos</th>
                <th scope="col">Email</th>
                <th scope="col">Dirección</th>
                <th scope="col">Telefono</th>
                <th scope="col">Acciones</th> {{--Contendrá los botones
borrar, y en el futuro el editar--}}
            </tr>
```

```
</thead>
<tbody>
    {{--Datos de los alumnos--}}
    @foreach ($alumnos as $alumno)
        <tr>
            <th scope="row">{{$alumno->id}}</th>
            <td>{{$alumno->nombre}}</td>
            <td>{{$alumno->apellidos}}</td>
            <td>{{$alumno->email}}</td>
            <td>{{$alumno->direccion}}</td>
            <td>{{$alumno->telefono}}</td>
            <td>
                {{--Botón borrar en un formulario--}}
                <form name="borrar"
action="{{$route('alumnos.destroy',$alumno)}}" method='POST'
style='white-space:nowrap;'>
                    @csrf
                    @method('DELETE')
                    <input type="submit" value="Borrar" class='btn
btn-danger normal'>
                </form>
            </td>
        </tr>
    @endforeach
</tbody>
</table>
    {{--Paginación--}}
    {{$alumnos->links()}}
@endsection
```

Comprobamos que todo funcione en la web

Listado

ALUMNOS MATRICULADOS						
Código	Nombre	Apellidos	Email	Dirección	Telefono	Acciones
1	Beatrice	Kunde	ybogan@feil.biz	3414 Streich Drive Suite 257	(225) 449-9884 x62756	Borrar
2	Elmira	Waters	labadie.mary@hotmail.com	7930 Justina Flat	630-468-1065 x95398	Borrar
3	Austen	Ritchie	hane.shea@hand.org	79529 Kiel Light		Borrar
4	Aletha	Medhurst	xsimonis@gmail.com	63582 Armand Street Suite 302		Borrar
5	Edwina	Pouros	nelson.klein@hotmail.com	8770 Tia Branch Suite 050	+13286424772	Borrar
< 1 2 3 4 >						

Paginación

ALUMNOS MATRICULADOS						
Código	Nombre	Apellidos	Email	Dirección	Telefono	Acciones
11	Cristal	Hudson	fhuel@yahoo.com	8463 Dare Landing		Borrar
12	Yesenia	Lemke	fhalvorson@hotmail.com	234 Luella Hollow Apt. 023	+1-554-461-8963	Borrar
13	Alta	Ernser	vivian.boehm@yahoo.com	136 Nader Plains		Borrar
14	Jalyn	Harvey	jalen79@robels.org	755 Abernathy Walk		Borrar
15	Margarett	Windler	beatty.maegan@connelly.com	30704 Kristofer Pine Apt. 805	960.904.0798	Borrar
< 1 2 3 4 >						

Borrado

ALUMNOS MATRICULADOS						
Alumno Borrado Correctamente.						
Código	Nombre	Apellidos	Email	Dirección	Teléfono	Acciones
1	Beatrice	Kunde	ybogan@feilbiz	3444 Streich Drive Suite 257	(225) 44-9884 x62756	Borrar
2	Elmira	waters	labadie.mary@hotmail.com	7930 Justina Flat	630-468-1065 x15398	Borrar
4	Aletha	Medhurst	xsimonis@gmail.com	63582 Armand Street Suite 302		Borrar
6	Elen	Hermiston	ncollier@waternet	702 Bode Vista		Borrar
7	odel	Mosciski	jaylin89@collier.com	38563 Carolanne Harbor Suite 378	(510) 377-4475 x54128	Borrar
< 1 2 3 4 >						