

Taller 1 Laravel Academia (20/01/2020)

Índice

- 1.- Requisitos previos.
- 2.- Instalación y creación de un proyecto Laravel.
- 3.- Activación del servidor Laravel.
- 4.- Creación de un Alias.
- 5.- Modelo, migración, controlador y vistas del proyecto.
- 6.- Creación y configuración de la Base de Datos.
- 7.- Modificación de las rutas.
- 8.- Métodos del controlador.
- 9.- Hoja de estilo (CSS).
- 10.- Creación de plantillas.
- 11.- Creación de vistas.
- 12.- Comprobación.

1.- Requisitos previos

Para poder trabajar con Laravel en Windows, es necesario instalar los siguientes programas:

-Servidor XAMPP

Ya que nos serán necesarios dos de los recursos que tiene, el servidor Apache (para la programación en PHP) y MySQL (para la creación y gestión de la base de datos).

-Composer

Es sistema de gestión para programar en PHP, con el podremos manejar dependencias y las librerías de PHP.

2.- Instalación y creación de un proyecto Laravel

Una vez instalados los programas anteriormente mencionados, debemos de abrir la Consola de Comandos (CMD) de nuestro ordenador.

Hay dos formas de instalar Laravel.

La primera es escribiendo el comando: ***composer global require laravel/installer***

La segunda (que es la que yo he utilizado) es la más practica a mi parecer, ya que a la vez que instalas Laravel, creas el proyecto con el que vas a trabajar.

Primero debemos de cambiarnos de directorio si queremos crear el proyecto en otra carpeta que no sea la del usuario:

cd C:\xampp\htdocs\daw\dwese\PracticaLaravel

A continuación, creamos el proyecto:

composer create-project laravel/laravel Academia

Importante, debemos de activar el servidor Apache y el servidor MySQL para poder trabajar.

3.- Activación del servidor Laravel

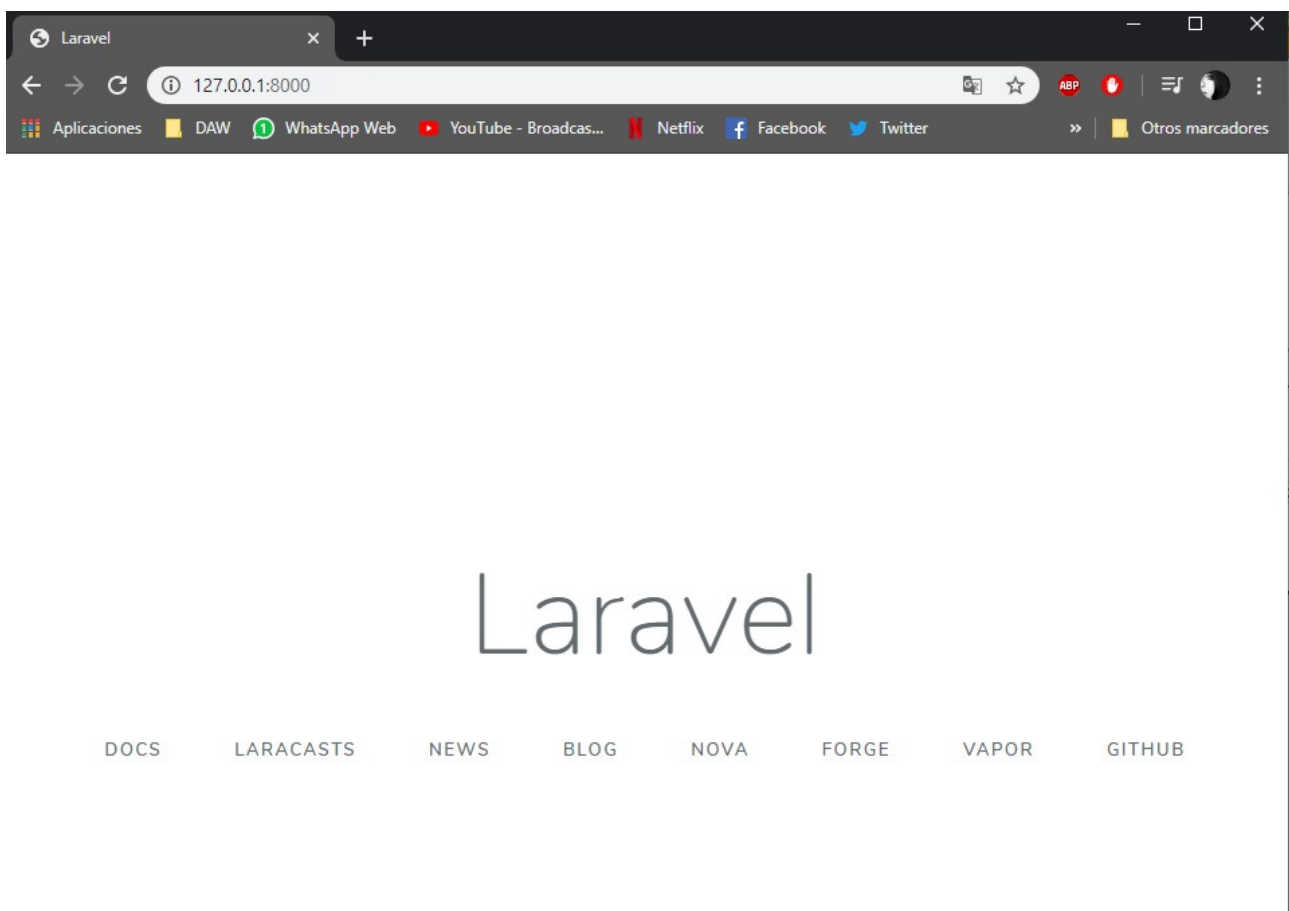
Para activar el servidor Laravel, debemos de escribir el siguiente comando:
php artisan serve

Lo inicia como localhost

```
C:\xampp\htdocs\daw\dwese\PracticalLaravel\Academia>php artisan serve  
Laravel development server started: http://127.0.0.1:8000
```

Para comprobar que se ha iniciado correctamente, debemos de poner en nuestro navegador la dirección que nos indica.

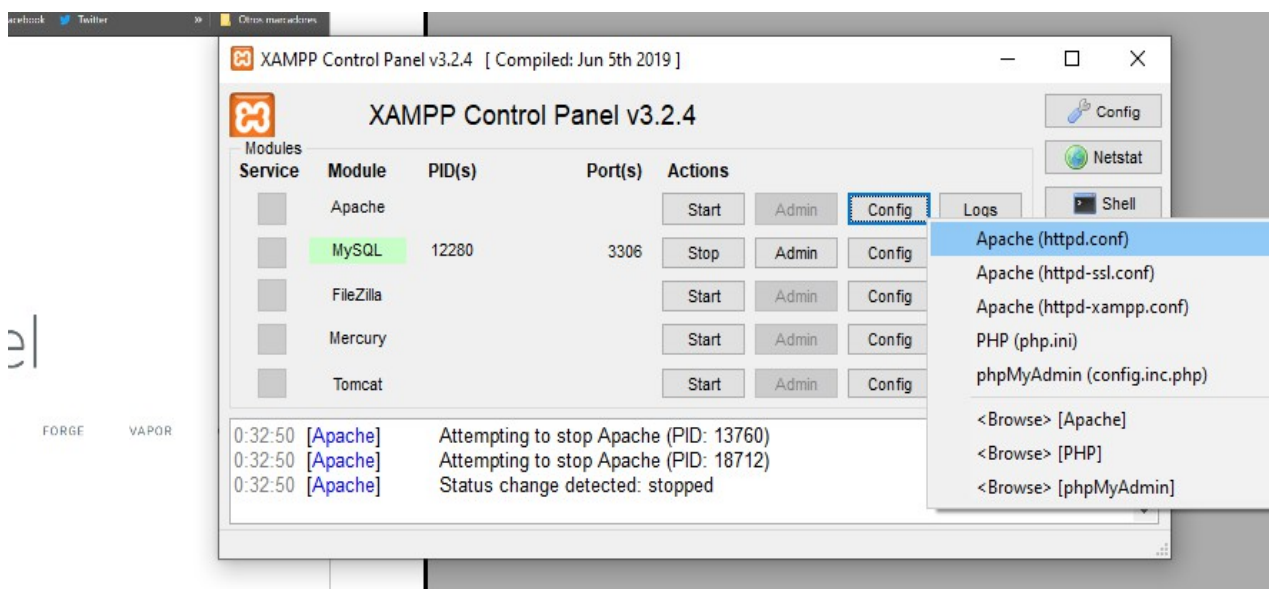
Y debe de aparecer la siguiente página:



4.- Creación de un Alias.

El alias nos sirve para “cambiar” la dirección del servidor para que en vez de que aparezca 127.0.0.1:8000 (como es mi caso), apareciera un nombre customizado.

Para ello, debemos de irnos al panel de control de XAMPP, detener el servidor Apache, abrir su configuración y seleccionar Apache (httpd.conf).

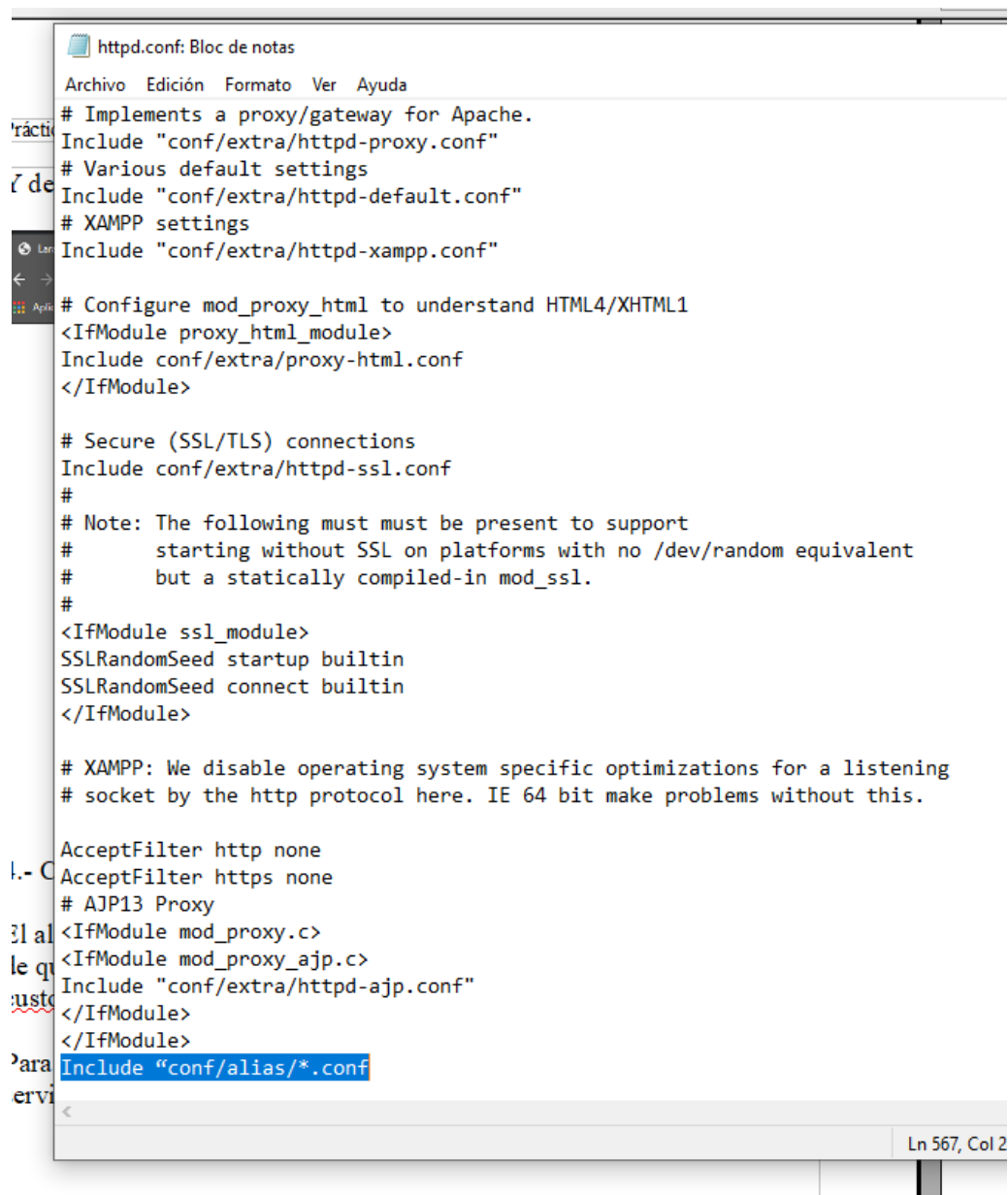


Se nos abrirá el archivo de configuración ***httpd.conf***

En el final de este archivo, debemos de añadir la siguiente línea:

Include "conf/alias/*.conf"

Y guardamos los cambios.



```
httpd.conf: Bloc de notas
Archivo Edición Formato Ver Ayuda
# Implements a proxy/gateway for Apache.
Include "conf/extra/httpd-proxy.conf"
# Various default settings
Include "conf/extra/httpd-default.conf"
# XAMPP settings
Include "conf/extra/httpd-xampp.conf"

# Configure mod_proxy_html to understand HTML4/XHTML1
<IfModule proxy_html_module>
Include conf/extra/proxy-html.conf
</IfModule>

# Secure (SSL/TLS) connections
Include conf/extra/httpd-ssl.conf
#
# 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>

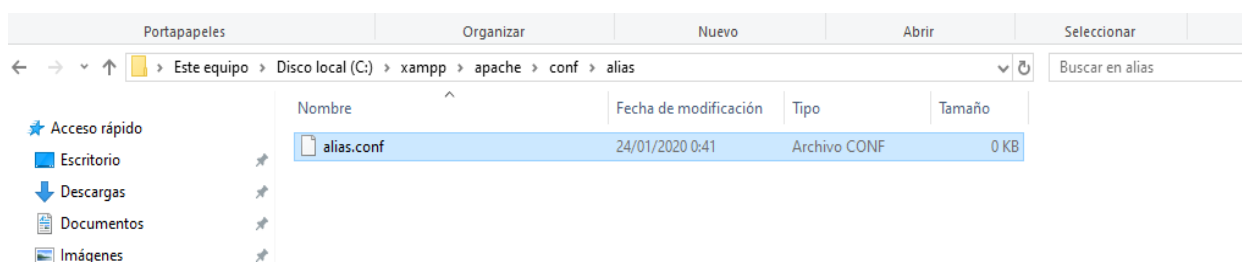
# XAMPP: We disable operating system specific optimizations for a listening
# socket by the http protocol here. IE 64 bit make problems without this.

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"
```

Ln 567, Col 2

Ahora, debemos de ir al directorio *xampp/apache/conf* y crear una carpeta que se llame *alias*.

Por último, dentro de ésta, crearemos los archivos con los alias que sean necesarios. Estos archivos deben de tener la extensión *.conf*



Ahora, abrimos el archivo y pegamos el siguiente código en el:

```
<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"

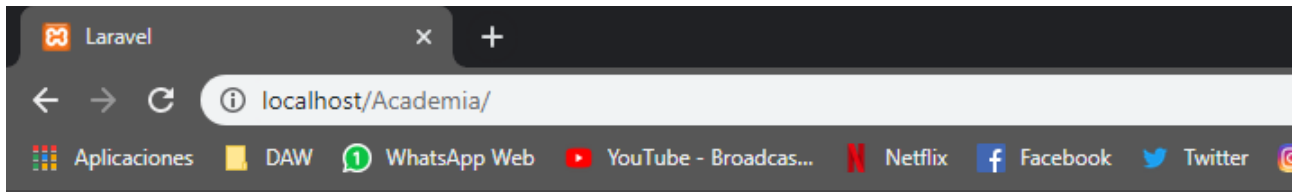
En la línea resaltada en negrita, debemos de poner el directorio de la carpeta **public** de nuestro proyecto.

En mi caso es:

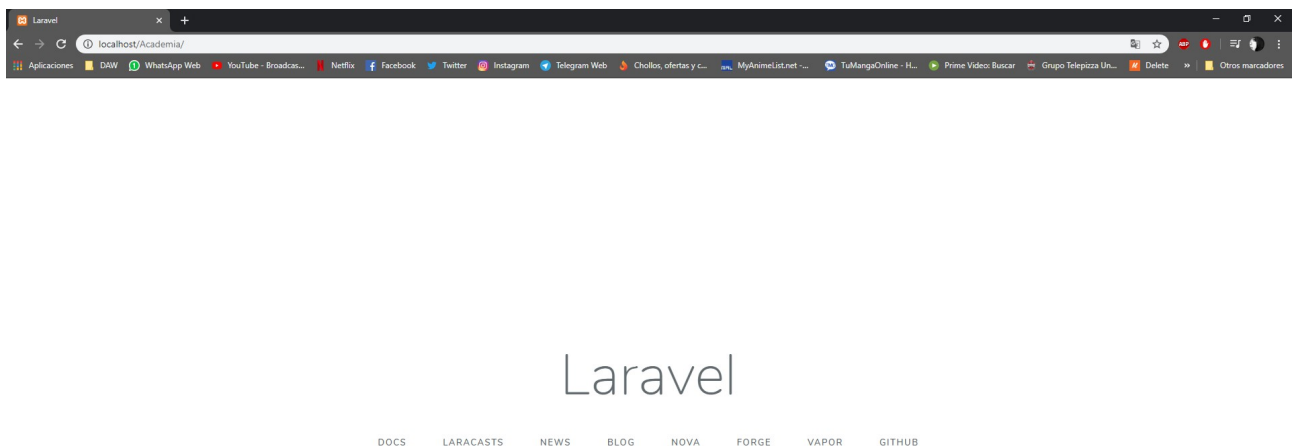
Alias /Academia

"C:\xampp\htdocs\daw\dwese\PracticaLaravel\Academia\public"

Guardamos los cambios y volvemos a nuestro navegador y comprobamos que se haya realizado correctamente.



Y debe de salir la misma pantalla de Laravel de antes.



5.- Modelo, migración, controlador y vistas del proyecto.

En nuestra consola de comandos, donde estábamos anteriormente (dentro de la ruta de nuestro proyecto) escribimos el siguiente comando:

php artisan make:model Alumnos -a

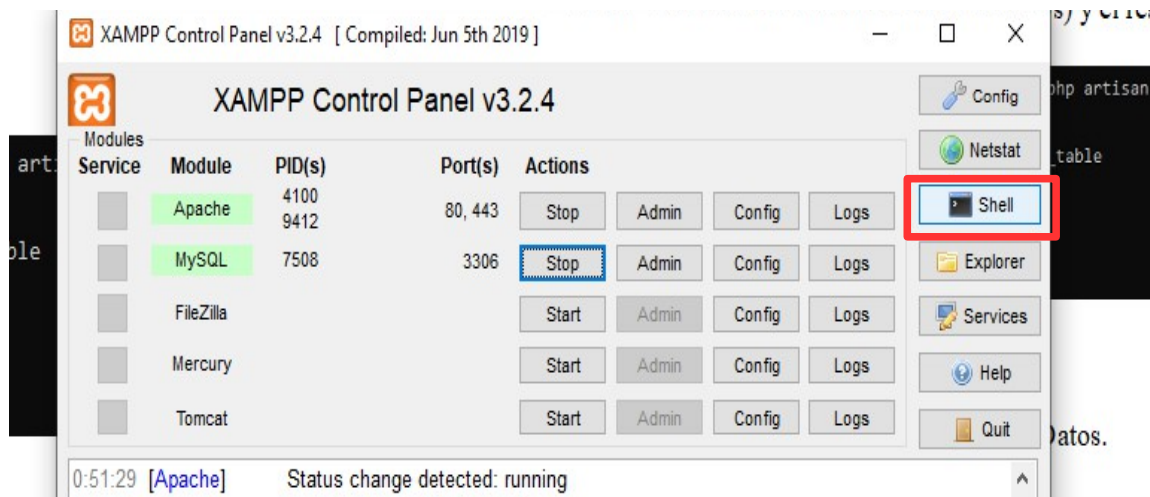
Con él, creamos a la vez el modelo (Alumnos) y el resto de elementos (-a).

```
C:\xampp\htdocs\daw\dwese\PracticaLaravel\Academia>php artisan make:model Alumnos -a
Model created successfully.
Factory created successfully.
Created Migration: 2020_01_24_000534_create_alumnos_table
Seeder created successfully.
Controller created successfully.

C:\xampp\htdocs\daw\dwese\PracticaLaravel\Academia>
```

6.- Creación y configuración de la Base de Datos.

Para la creación de la base de datos, debemos de abrir la consola de comandos de MySQL que viene en XAMPP.



Una vez dentro, iniciamos sesión en MySQL:

```
c:\> XAMPP for Windows - mysql -u root

Setting environment for using XAMPP for Windows.
DAW@DESKTOP-IS3M4KK c:\xampp
# mysql -u root
Welcome to the MariaDB monitor.  Commands end with ; or \g.
Your MariaDB connection id is 8
Server version: 10.4.6-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)]>
```

Una vez dentro, creamos la base de datos y la seleccionamos:

```
MariaDB [(none)]> create database academia;  
Query OK, 1 row affected (0.015 sec)  
  
MariaDB [(none)]> use academia;  
Database changed  
MariaDB [academia]>
```

A continuación, creamos un usuario para nuestra Base de Datos. Para ello, escribimos el siguiente comando:

create user nombreUsuario@'localhost' identified by 'contraseñaUsuario';

Yo ya tengo un usuario creado, que será el que voy a utilizar para continuar con la práctica que es userpdo@'localhost'

Cuando tengamos nuestro usuario creado, debemos de darle permisos para poder acceder y modificar la base de datos a nuestro antojo con el siguiente comando:

grant all privileges on academia.* to userpdo@'localhost';

```
MariaDB [(none)]> use academia;  
Database changed  
MariaDB [academia]> grant all privileges on academia.* to userpdo@'localhost';  
Query OK, 0 rows affected (0.016 sec)  
  
MariaDB [academia]>
```

Debemos de configurar nuestra base de datos en nuestro proyecto para poder tener acceso a ella.

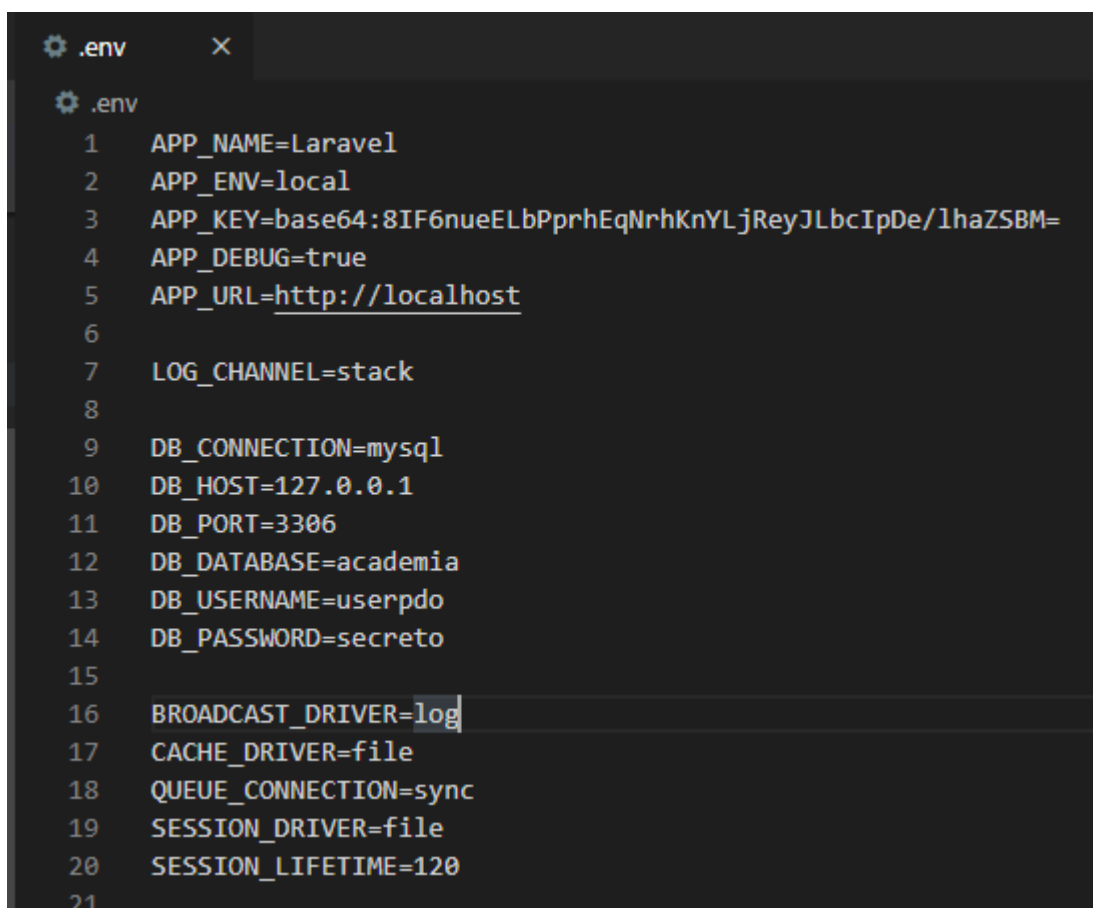
En nuestro proyecto, buscamos el archivo **.env**
Dentro del archivo, buscamos el siguiente texto:

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

DB_PASSWORD=

Y modificamos los valores para que tenga los datos de nuestra base de datos.

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

A screenshot of a code editor showing a .env file. The file contains various configuration variables for a Laravel application. The database configuration section is highlighted, showing DB_CONNECTION=mysql, DB_HOST=127.0.0.1, DB_PORT=3306, DB_DATABASE=academia, DB_USERNAME=userpdo, and DB_PASSWORD=secreto. The variable DB_PASSWORD is currently selected with the mouse.

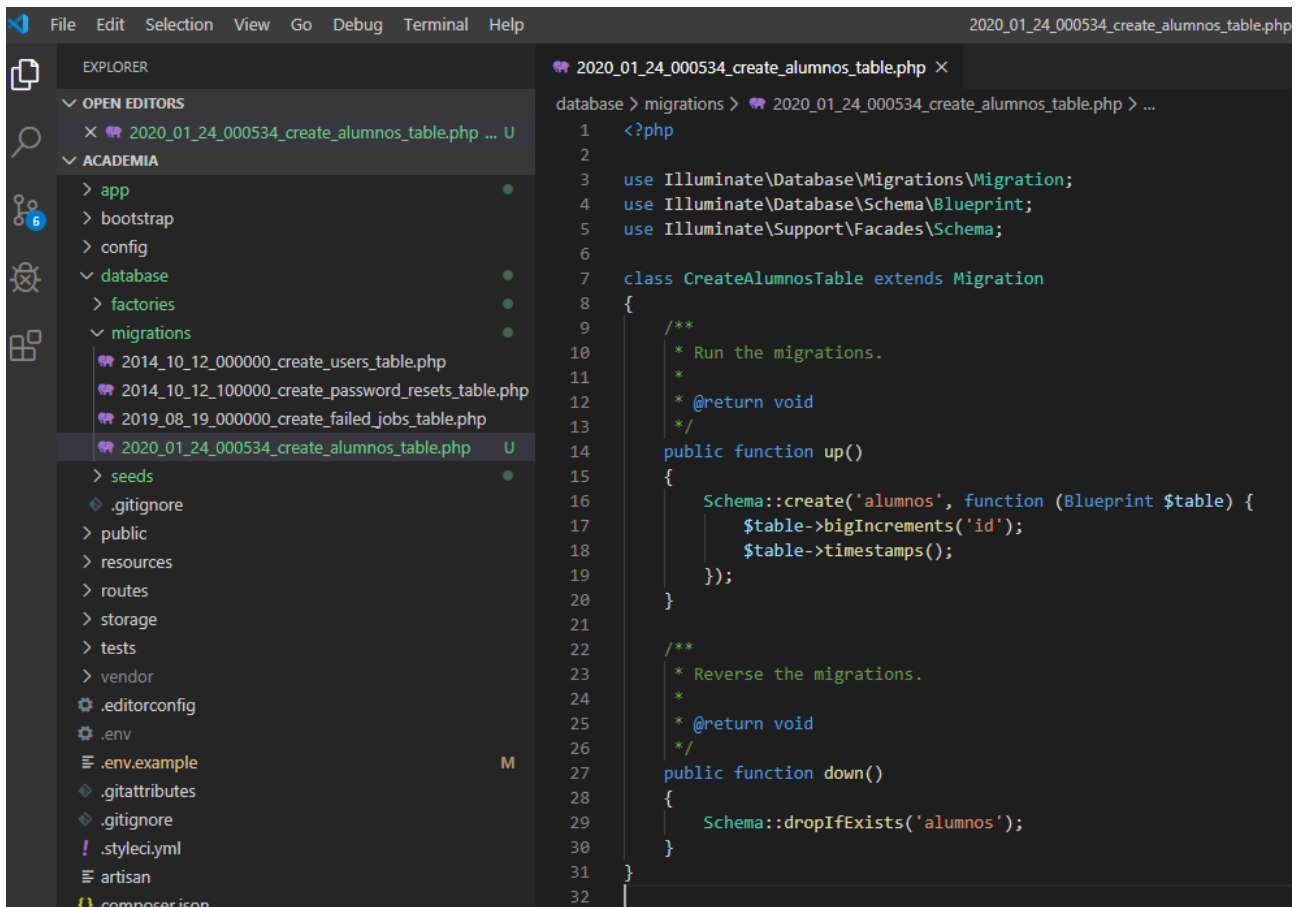
```
.env
1 APP_NAME=Laravel
2 APP_ENV=local
3 APP_KEY=base64:8IF6nueELbPprhEqNrKnyLjReyJLbcIpDe/1haZSBM=
4 APP_DEBUG=true
5 APP_URL=http://localhost
6
7 LOG_CHANNEL=stack
8
9 DB_CONNECTION=mysql
10 DB_HOST=127.0.0.1
11 DB_PORT=3306
12 DB_DATABASE=academia
13 DB_USERNAME=userpdo
14 DB_PASSWORD=secreto
15
16 BROADCAST_DRIVER=log
17 CACHE_DRIVER=file
18 QUEUE_CONNECTION=sync
19 SESSION_DRIVER=file
20 SESSION_LIFETIME=120
21
```

Ahora vamos a modificar los datos de la tabla Alumnos.

Dentro de nuestro proyecto buscamos un archivo dentro de *database/migrations* que acabe en *create_alumnos_table.php*

Contendrá varios dígitos en su nombre, pero no hay que preocuparse por

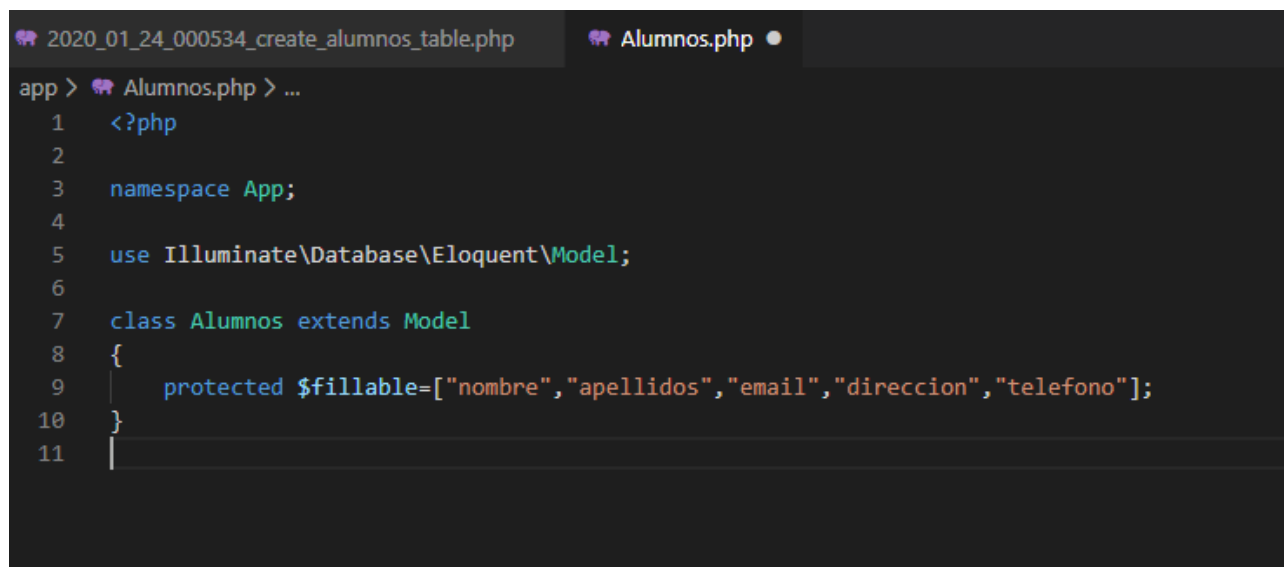
ello ya que es la fecha en la que se creo la migración.



Debemos de modificar la función **up()** para añadir todos los campos necesarios para nuestra tabla de Alumnos:

```
public function up()
{
    Schema::create('alumnos', function (Blueprint $table) {
        $table->bigIncrements('id');
        $table->string('nombre',20);
        $table->string('apellidos',60);
        $table->string('email',200)->unique();
        $table->string('direccion',250);
        $table->string('telefono',60)->nullable();
        $table->timestamps();
    });
}
```

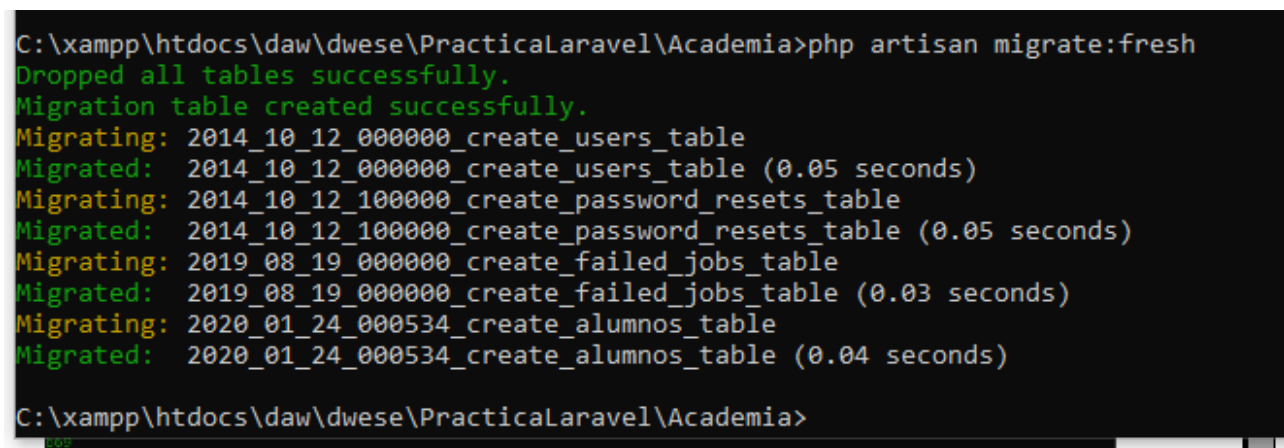
Para que podamos añadir datos a través de los formularios pertinentes, debemos de ir al archivo *app/Alumnos.php* y añadir la siguiente línea dentro de la clase Alumnos:



```
2020_01_24_000534_create_alumnos_table.php  Alumnos.php ●
app > Alumnos.php > ...
1  <?php
2
3  namespace App;
4
5  use Illuminate\Database\Eloquent\Model;
6
7  class Alumnos extends Model
8  {
9      protected $fillable=["nombre","apellidos","email","direccion","telefono"];
10 }
11
```

El siguiente paso es migrar a la base de datos.

En la consola de comandos, (siguiendo dentro de la carpeta de nuestro proyecto) escribimos: *php artisan migrate:fresh*



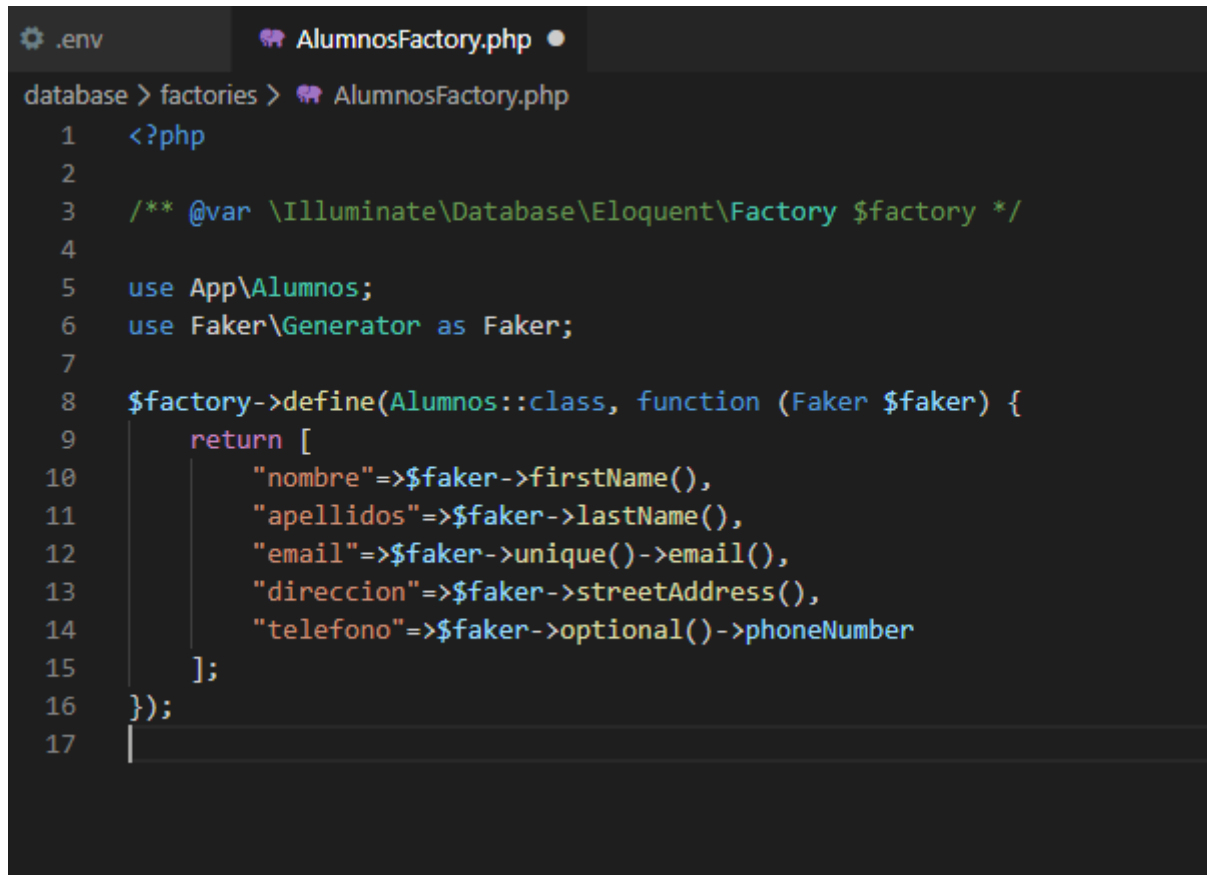
```
C:\xampp\htdocs\daw\dwese\PracticaLaravel\Academia>php artisan migrate:fresh
Dropped all tables successfully.
Migration table created successfully.
Migrating: 2014_10_12_000000_create_users_table
Migrated: 2014_10_12_000000_create_users_table (0.05 seconds)
Migrating: 2014_10_12_100000_create_password_resets_table
Migrated: 2014_10_12_100000_create_password_resets_table (0.05 seconds)
Migrating: 2019_08_19_000000_create_failed_jobs_table
Migrated: 2019_08_19_000000_create_failed_jobs_table (0.03 seconds)
Migrating: 2020_01_24_000534_create_alumnos_table
Migrated: 2020_01_24_000534_create_alumnos_table (0.04 seconds)
C:\xampp\htdocs\daw\dwese\PracticaLaravel\Academia>
```

El próximo paso va a ser crear datos de prueba para nuestra base de datos, configurando el Factory con Faker.

Nos vamos al archivo *database/factories/AlumnosFactory.php* en nuestro proyecto.

Debemos de rellenar el *return* de *\$factory* con los campos de la tabla alumnos y con *faker*.

Faker es un framework que se encarga de generar los datos.

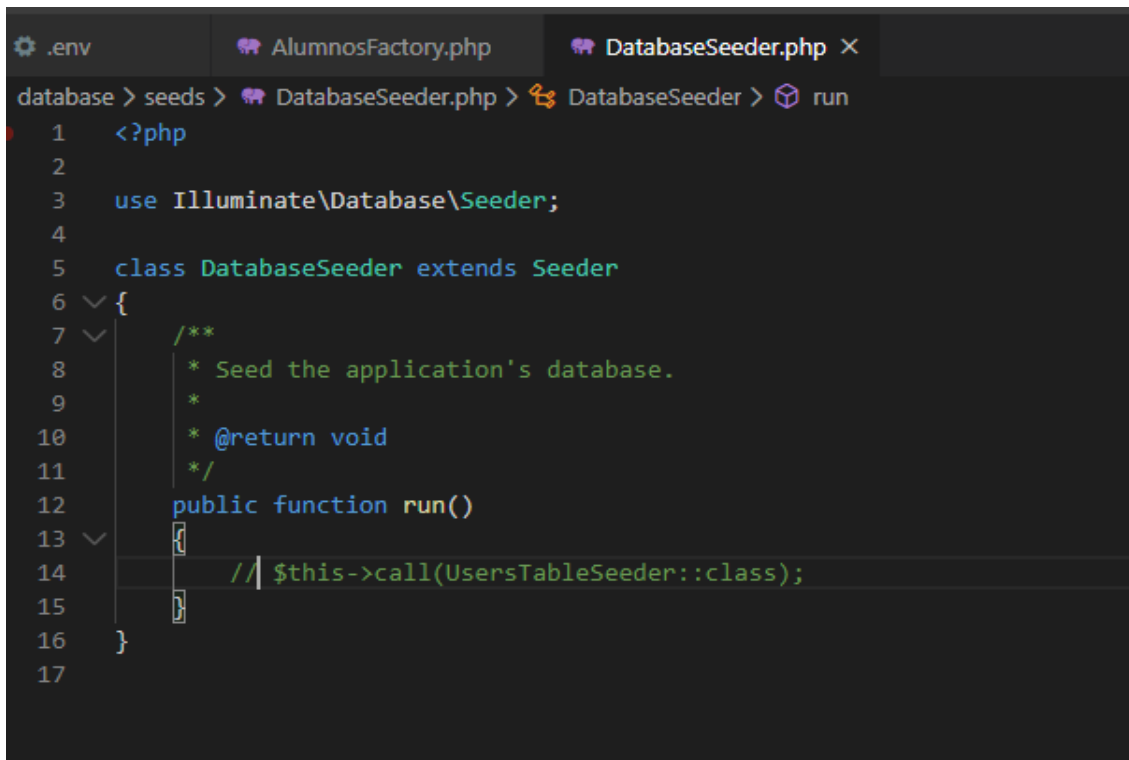


```
.env AlumnosFactory.php
database > factories > AlumnosFactory.php
1  <?php
2
3  /** @var \Illuminate\Database\Eloquent\Factory $factory */
4
5  use App\Alumnos;
6  use Faker\Generator as Faker;
7
8  $factory->define(Alumnos::class, function (Faker $faker) {
9      return [
10         "nombre"=>$faker->firstName(),
11         "apellidos"=>$faker->lastName(),
12         "email"=>$faker->unique()->email(),
13         "direccion"=>$faker->streetAddress(),
14         "telefono"=>$faker->optional()->phoneNumber
15     ];
16 });
17
```

A continuación vamos a configurar el Seeder.

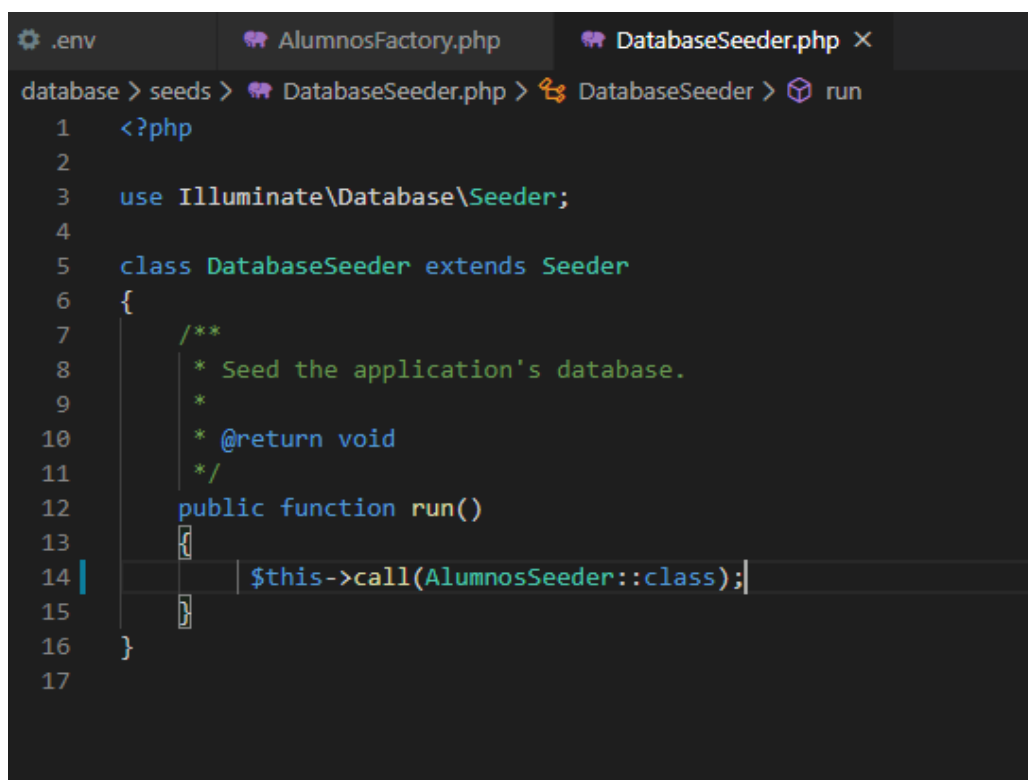
El seeder sirve para inicializar las tablas con datos.

Nos vamos al archivo *database/seeds/DatabaseSeeder.php*



```
.env | AlumnosFactory.php | DatabaseSeeder.php x
database > seeds > DatabaseSeeder.php > DatabaseSeeder > run
1  <?php
2
3  use Illuminate\Database\Seeder;
4
5  class DatabaseSeeder extends Seeder
6  {
7      /**
8       * Seed the application's database.
9       *
10      * @return void
11      */
12     public function run()
13     {
14         // $this->call(UsersTableSeeder::class);
15     }
16 }
17
```

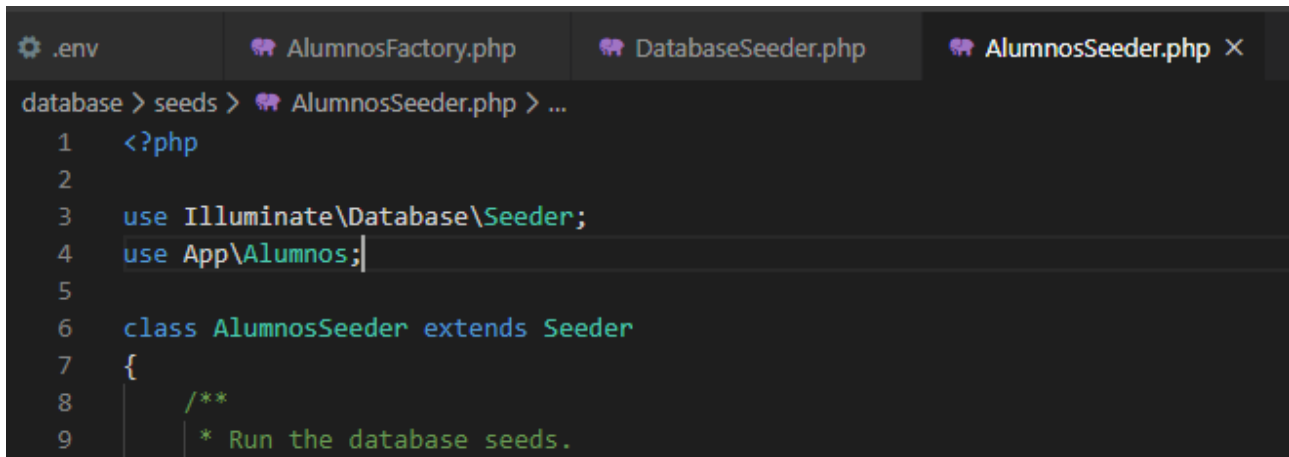
Una vez aquí, descomentamos la línea de dentro de la función *run()* y la cambiamos por la siguiente:



```
.env | AlumnosFactory.php | DatabaseSeeder.php x
database > seeds > DatabaseSeeder.php > DatabaseSeeder > run
1  <?php
2
3  use Illuminate\Database\Seeder;
4
5  class DatabaseSeeder extends Seeder
6  {
7      /**
8       * Seed the application's database.
9       *
10     * @return void
11     */
12     public function run()
13     {
14         $this->call(AlumnosSeeder::class);
15     }
16 }
17
```

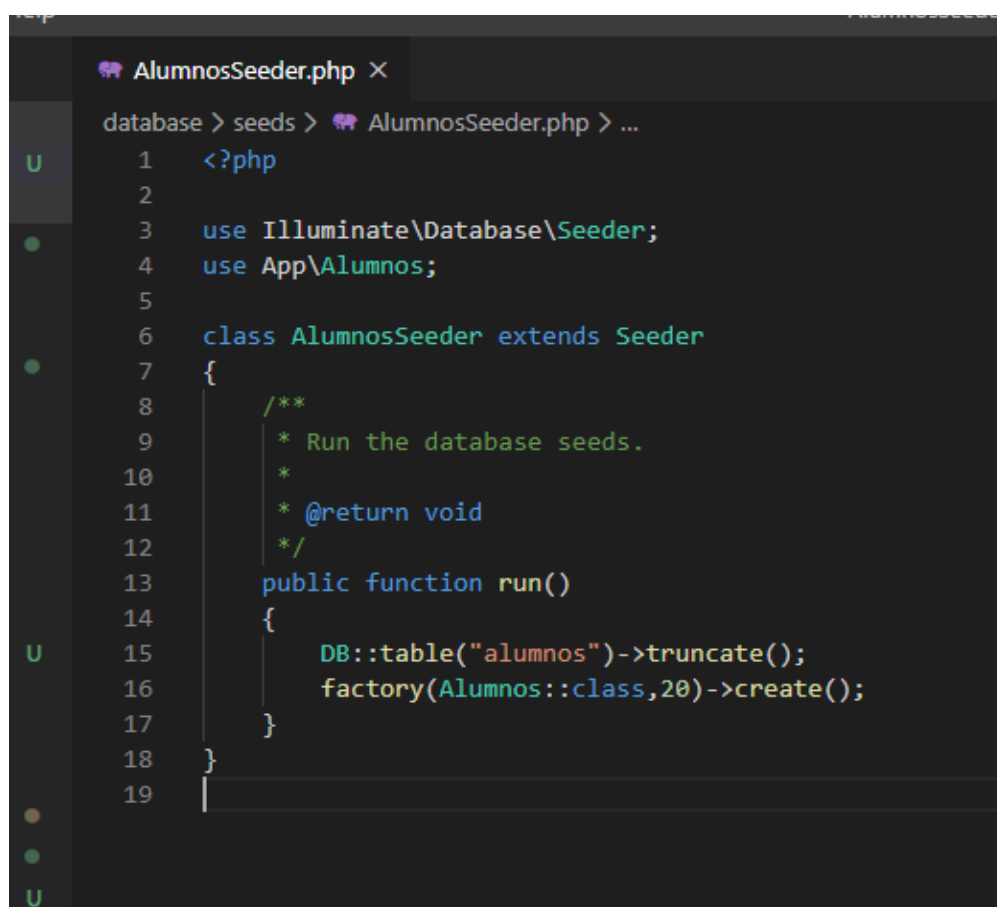
También tendremos que entrar al archivo *database/seeds/AlumnosSeeder.php* y escribir debajo de *Illuminate\Database\Seeder*;

use App\Alumnos para poder tener acceso a la clase:



```
database > seeds > AlumnosSeeder.php > ...
1  <?php
2
3  use Illuminate\Database\Seeder;
4  use App\Alumnos;
5
6  class AlumnosSeeder extends Seeder
7  {
8      /**
9      * Run the database seeds.
```

Por último, en la función *run()* de ese mismo archivo deberemos de añadir lo siguiente:



```
AlumnosSeeder.php
database > seeds > AlumnosSeeder.php > ...
1  <?php
2
3  use Illuminate\Database\Seeder;
4  use App\Alumnos;
5
6  class AlumnosSeeder extends Seeder
7  {
8      /**
9      * Run the database seeds.
10     *
11     * @return void
12     */
13     public function run()
14     {
15         DB::table("alumnos")->truncate();
16         factory(Alumnos::class,20)->create();
17     }
18 }
19
```

La sentencia ***DB::table("alumnos")->truncate();*** vacía la tabla en el caso de que esté llena.

La sentencia ***factory(Alumnos::class,20)->create();*** crea 20 registros en la base de datos.

El siguiente paso es actualizar nuestra base de datos. En la consola de comandos debemos de escribir: ***php artisan db:seed***

(Dependiendo de la cantidad de registros, tardará más o menos en actualizarse).

```
C:\xampp\htdocs\daw\dwese\Practicalaravel\Academia>php artisan db:seed
Seeding: AlumnosSeeder
Seeded: AlumnosSeeder (0.17 seconds)
Database seeding completed successfully.
C:\xampp\htdocs\daw\dwese\Practicalaravel\Academia>
```

Vamos a comprobar que la tabla ha sido rellenada. Nos vamos a la consola de MySQL y hacemos el comando ***select * from alumnos;***

XAMPP for Windows - mysql -u root

```
MariaDB [(none)]> use academia;
Database changed
MariaDB [academia]> grant all privileges on academia.* to userpdo@'localhost';
Query OK, 0 rows affected (0.016 sec)

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

id	nombre	apellidos	email	direccion	telefono	created_at	updated_at
1	Joyce	Stracke	kmccullough@gmail.com	653 Bins Brook Apt. 418	NULL	2020-01-24 02:29:57	2020-01-24 02:29:57
2	Triston	Sponen	zconkery@kshierin.biz	2819 Abigale Club Apt. 065	NULL	2020-01-24 02:29:57	2020-01-24 02:29:57
3	Eliza	Runte	sponen.emery@yahoo.com	8921 Oran Cliffs	NULL	2020-01-24 02:29:57	2020-01-24 02:29:57
4	Christopher	Mueller	hitrhe.karella@nikolaus.com	31201 Edwina Neck Suite 518	NULL	2020-01-24 02:29:57	2020-01-24 02:29:57
5	Julia	Lockman	precious97@yahoo.com	1134 Konopelski Court Suite 753	+1-216-534-8020	2020-01-24 02:29:57	2020-01-24 02:29:57
6	Joeseph	Zemlak	patience.fisher@fahey.com	919 Wintheiser Valleys	748-538-0185	2020-01-24 02:29:57	2020-01-24 02:29:57
7	Adriana	Heller	jakayla83@gmail.com	38664 Edythe Brooks	296-550-3942	2020-01-24 02:29:57	2020-01-24 02:29:57
8	Brayan	Hodkiewicz	bill145@gmail.com	7031 Stacey Prairie	436-397-2469	2020-01-24 02:29:57	2020-01-24 02:29:57
9	Constance	Zieme	adell69@donnelly.com	5195 Elliott Mountain Apt. 032	(460) 389-7262	2020-01-24 02:29:57	2020-01-24 02:29:57
10	Syble	Jacobs	metz.bud@gmail.com	90436 Karlee Inlet	NULL	2020-01-24 02:29:57	2020-01-24 02:29:57
11	Dante	Botsford	koepp.emilia@mante.net	55708 Lucious Shores	+1 (959) 586-0781	2020-01-24 02:29:57	2020-01-24 02:29:57
12	Krista	Weimann	klings.maye@gmail.com	8348 Ratke Course Suite 898	NULL	2020-01-24 02:29:57	2020-01-24 02:29:57
13	Zita	Mertz	frank23@abbott.biz	33340 Ardella Summit	746-691-7638	2020-01-24 02:29:57	2020-01-24 02:29:57
14	Jared	Wisoky	mariah.schmitt@gmail.com	161 Von Groves	1-629-893-1018	2020-01-24 02:29:57	2020-01-24 02:29:57
15	Alfred	Rodriguez	lucie.bradtke@hotmail.com	10557 Sauer Ferry	1-271-619-5335 x35505	2020-01-24 02:29:57	2020-01-24 02:29:57
16	Rowena	Brakus	yvonne.fisher@yahoo.com	98786 Cummings Branch Apt. 473	NULL	2020-01-24 02:29:57	2020-01-24 02:29:57
17	Mya	Champlin	claud.hagenes@gmail.com	35360 Feil Glens	NULL	2020-01-24 02:29:57	2020-01-24 02:29:57
18	Kimberly	Torphy	okeefe.elliott@murray.com	8273 Crawford Parkways	873-492-8130	2020-01-24 02:29:57	2020-01-24 02:29:57
19	Aniyah	Weimann	akertmann@walsh.com	3388 Adalberto Mill	(670) 660-9392	2020-01-24 02:29:57	2020-01-24 02:29:57
20	Tyra	Carter	herdman@hoppe.com	1889 Haag Orchard Suite 191	NULL	2020-01-24 02:29:57	2020-01-24 02:29:57

20 rows in set (0.001 sec)

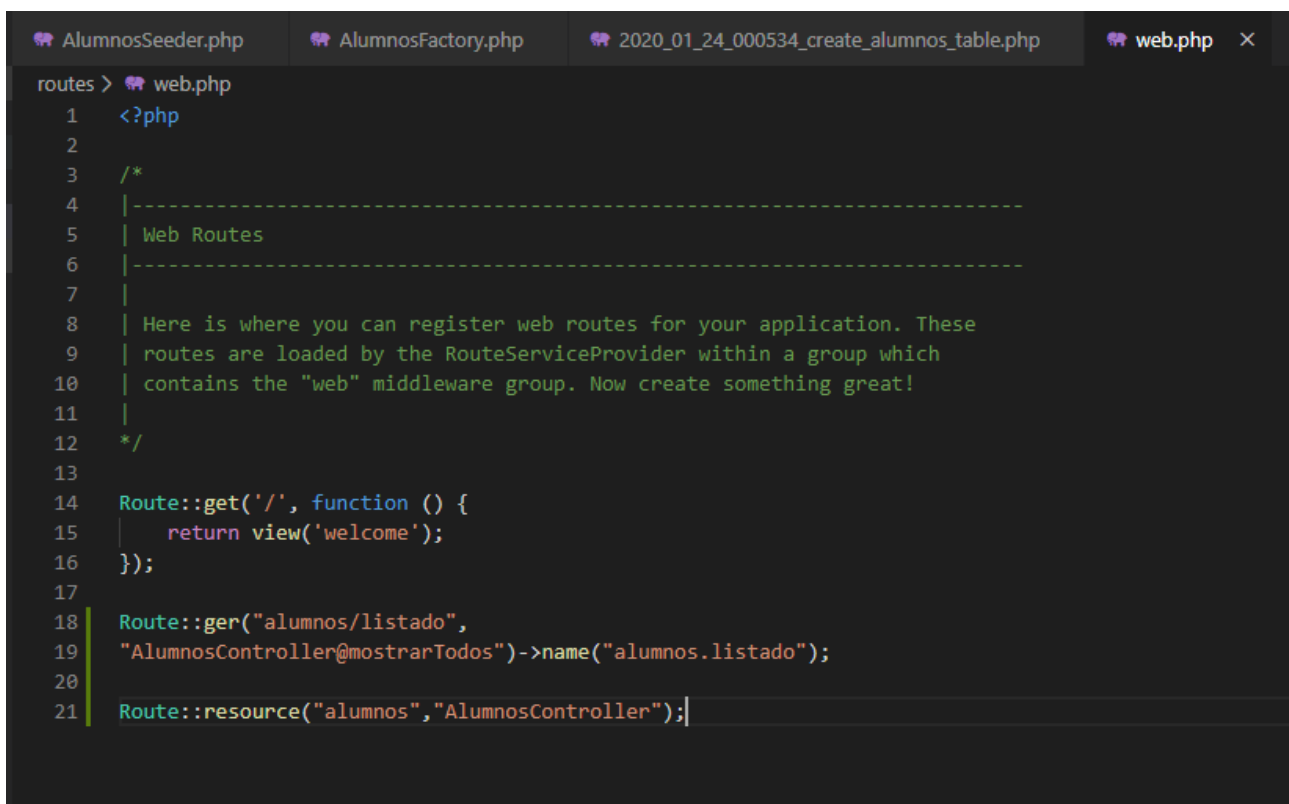
Como podemos observar, se encuentran todos.

7.- Modificación de las rutas.

Para modificar las rutas, debemos de ir al archivo routes/web.php y añadir al final:

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

```
Route::resource("alumnos","AlumnosController");
```



```
AlumnosSeeder.php  AlumnosFactory.php  2020_01_24_000534_create_alumnos_table.php  web.php ×
routes > web.php
1  <?php
2
3  /*
4  |-----
5  | Web Routes
6  |-----
7  |
8  | Here is where you can register web routes for your application. These
9  | routes are loaded by the RouteServiceProvider within a group which
10 | contains the "web" middleware group. Now create something great!
11 |
12 */
13
14 Route::get('/', function () {
15     return view('welcome');
16 });
17
18 Route::ger("alumnos/listado",
19 "AlumnosController@mostrarTodos")->name("alumnos.listado");
20
21 Route::resource("alumnos","AlumnosController");|
```

8.- Métodos del controlador.

Vamos a crear un método para mostrar todos los registros paginados de 5 en 5.

Para ello iremos al archivo *app/Http/Controllers/AlumnosController.php*

Lo primero de todo será poner en la punta arriba del archivo la sentencia *use Session();*

```
<?php

namespace App\Http\Controllers;

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

class AlumnosController extends Controller
{
    /**
     * Display a listing of the resource.
     *
     * @return \Illuminate\Http\Response
     */
    public function index()
    {
        //
    }
}
```

Debajo del método *index()* escribiremos el siguiente método:

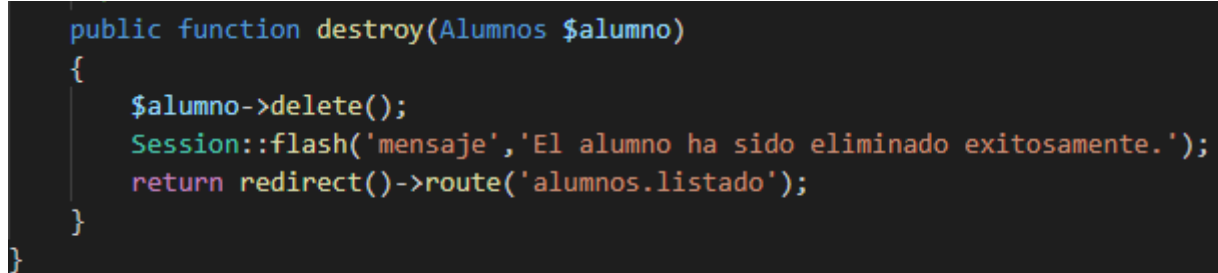
```
public function index()
{
    //
}

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

Este método además de paginar de 5 en 5 los registros, nos dirigirá a la vista ***alumnos.listado***, que crearemos más adelante.

Por último, dentro del mismo archivo, debemos de buscar el método ***destroy()*** en él, escribiremos lo siguiente:

```
public function destroy(Alumnos $alumno)
{
    $alumno->delete();
    Session::flash('mensaje', 'El alumno ha sido eliminado exitosamente.');
```



```
    return redirect()->route('alumnos.listado');
}
```

Con este método podremos eliminar registros de nuestra base de datos.

9.- Hoja de estilo (CSS).

Para crear una hoja de estilo CSS, debemos de crear una carpeta para ello dentro de la carpeta **public** de nuestro proyecto.

La mía se llamará **css** y dentro de ella creamos el archivo **style.css**

```
public > css > # style.css > .extra
1
2  body{
3      background-color: darkgray;
4  }
5  .normal, .grande, .extra{
6      font-family: Impact, Haettenschweiler, 'Arial Narrow Bold', sans-serif;
7  }
8  .normal{
9      font-size: 16px;
10 }
11 .grande{
12     font-size: 18px;
13     font-weight: bolder;
14 }
15 .extra{
16     font-size: 20px;
17     font-weight: bolder;
18 }
```

10.- Creación de plantillas.

Nos vamos a la carpeta *resources/views* y ahí dentro creamos la carpeta *plantillas*. Dentro de esta carpeta, crearemos tantos archivos *.blade.php* como sean necesarios.

Yo crearé uno y se llamará *plantilla.blade.php*

A continuación, rellenaremos nuestra plantilla.

```
# style.css  plantilla.blade.php x
resources > views > plantillas > plantilla.blade.php
1  <!DOCTYPE html>
2  <html lang="es">
3  <head>
4      <meta charset="UTF-8">
5      <meta name="viewport" content="width=device-width, initial-scale=1.0">
6      <meta http-equiv="X-UA-Compatible" content="ie=edge">
7      <title>@yield('titulo')</title>
8      <link rel="stylesheet" href="https://stackpath.bootstrapcdn.com/bootstrap/4.4.1/css/bootstrap.min.css" >
9      <link href="https://fonts.googleapis.com/css?family=Indie+Flower&display=swap" rel="stylesheet">
10     <link href="{{asset('css/style.css')}}" rel="stylesheet">
11 </head>
12 <body>
13     <h1 class='text-center mt-3 grande'>@yield('cabecera')</h1>
14     <div class="container mt-3">
15         @yield('contenido')
16     </div>
17
18 </body>
19 </html>
```


11.- Creación de vistas.

Ahora nos debemos de colocar de nuevo en *resources/views*, pero en esta ocasión crearemos una carpeta para las vistas. La mía se llamará *alumnos*.

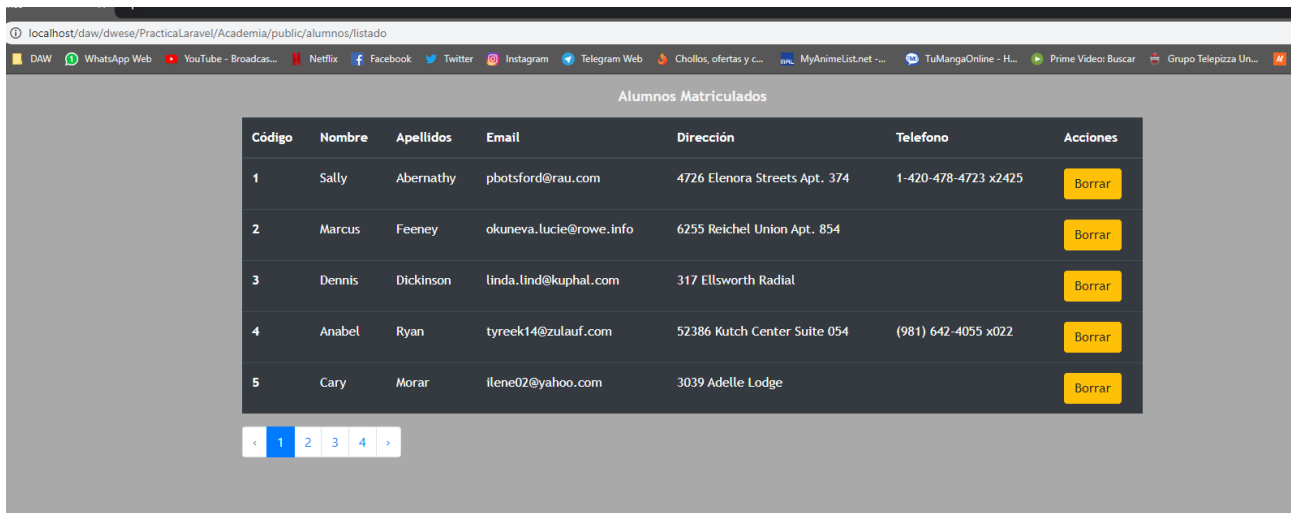
Dentro de mi carpeta alumno, voy a crear un archivo que se llamará *listado.blade.php* escribiendo el siguiente código dentro de el:

```
listado.blade.php X
resources > views > alumnos > listado.blade.php
1  {{--Plantilla--}}
2  @extends('plantillas.plantilla')
3  {{--titulo de la página--}}
4  @section('titulo')
5      Lista de Alumnos
6  @endsection
7  {{-- Cabecera--}}
8  @section('cabecera')
9      Alumnos Matriculados
10 @endsection
11 {{--Contenido de la página--}}
12 @section('contenido')
13 {{--Muestra el mensaje que haya en el Session, si lo tiene--}}
14 @if(Session::has('mensaje'))
15 <div class='container mt-3 mb-3 alert-success'>
16     {{Session::get('mensaje')}}
17 </div>
18 @endif
19 {{--Tabla de la info--}}
20 <table class="table table-dark normal">
21     <thead>
22         <tr>
23             <th scope="col">Código</th>
24             <th scope="col">Nombre</th>
25             <th scope="col">Apellidos</th>
26             <th scope="col">Email</th>
27             <th scope="col">Dirección</th>
28             <th scope="col">Telefono</th>
29             <th scope="col">Acciones</th>
30         </tr>
31     </thead>
32     <tbody>
33         @foreach ($alumnos as $alumno)
34             <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>
    <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-warning normal">
    </form>
</td>
</tr>
@endforeach
</tbody>
</table>
{{ --Paginación-- }}
{{ $alumnos->links() }}
@endsection
```

12.- Comprobación.

Listado de alumnos:



Alumnos Matriculados

Código	Nombre	Apellidos	Email	Dirección	Telefono	Acciones
1	Sally	Abernathy	pbotsford@rau.com	4726 Elenora Streets Apt. 374	1-420-478-4723 x2425	Borrar
2	Marcus	Feeney	okuneva.lucie@rowe.info	6255 Reichel Union Apt. 854		Borrar
3	Dennis	Dickinson	linda.tind@kuphal.com	317 Ellsworth Radial		Borrar
4	Anabel	Ryan	tyreek14@zulauf.com	52386 Kutch Center Suite 054	(981) 642-4055 x022	Borrar
5	Cary	Morar	ilene02@yahoo.com	3039 Adelle Lodge		Borrar

< 1 2 3 4 >

Paginación de 5 en 5:

Código	Nombre	Apellidos	Email	Dirección	Telefono	Acciones
6	Sofia	Marquardt	armstrong.cathryn@yahoo.com	6228 Harrison Isle		Borrar
7	Erna	Daniel	ohessel@gmail.com	674 Leffler Pines	274-359-3719 x350	Borrar
8	Vincenza	Rolfson	gconroy@gmail.com	403 Chadd Locks Apt. 694	230-717-3189 x7018	Borrar
9	Catharine	Stanton	mheathcote@yahoo.com	95854 Domenico Estate		Borrar
10	Unique	Breitenberg	bettie10@nicolas.com	842 Schmidt Motorway	889-938-7030	Borrar

< 1 2 3 4 >

Alumnos Matriculados						
Código	Nombre	Apellidos	Email	Dirección	Telefono	Acciones
11	Merlin	Mann	anita.dibbert@hotmail.com	454 Armstrong Fort	+1 (207) 618-0899	<button>Borrar</button>
12	Carmela	Beahan	ghackett@haley.com	28502 Alanis Harbors Suite 379		<button>Borrar</button>
13	Joy	Kemmer	abbott.brennon@hotmail.com	853 Minerva Throughway		<button>Borrar</button>
14	Eveline	Ryan	leanna.purdy@gmail.com	87302 Genevieve Neck	830-442-6307 x66744	<button>Borrar</button>
15	Jennie	Boyer	camila.willms@hotmail.com	38632 Mortimer Trace	+1-947-309-6189	<button>Borrar</button>

< 1 2 3 4 >

Eliminación:

Alumnos Matriculados						
El alumno ha sido eliminado exitosamente.						
Código	Nombre	Apellidos	Email	Dirección	Telefono	Acciones
1	Sally	Abernathy	pbotsford@rau.com	4726 Elenora Streets Apt. 374	1-420-478-4723 x2425	<button>Borrar</button>
4	Anabel	Ryan	tyreek14@zulauf.com	52386 Kutch Center Suite 054	(981) 642-4055 x022	<button>Borrar</button>
5	Cary	Morar	ilene02@yahoo.com	3039 Adelle Lodge		<button>Borrar</button>
6	Sofia	Marquardt	armstrong.cathryn@yahoo.com	6228 Harrison Isle		<button>Borrar</button>
7	Erna	Daniel	ohessel@gmail.com	674 Leffler Pines	274-359-3719 x350	<button>Borrar</button>

< 1 2 3 4 >