Guía de LARAVEL

- Buscar la guía de instalación en la página de Laravel

- Instalar composer de https://getcomposer.org/

- Comprobar que funciona poniendo "composer" en el shell. Comprobar en el PATH la ruta de instalación.

- Instalar el instalador de Laravel con el siguiente comando: **composer global require laravel/installer**

- En una carpeta de proyectos laravel (ejecutar en la shell): **laravel new crud**

- Entramos en la carpeta *crud* y abrimos el Visual Studio Code con: **code .**

- Desde el XAMP abrimos el fichero httpd.conf: Apache > Config > Apache (httpd.conf)

- Editamos el documento cambiando la configuración típica por la configuración de laravel:

```
#Configuración típica xamp:
#DocumentRoot "C:/xampp/htdocs"
#<Directory "C:/xampp/htdocs">

#Configuración root laravel:
DocumentRoot "C:/Users/wadmin/Documents/laravel/crud/public"
<Directory "C:/Users/wadmin/Documents/laravel/crud/public">
    #
    # Possible values for the Options directive are "None", "All",
    # ...
```

- Reiniciar Apache

- Comprobar que funciona en el navegador con 'localhost'

- Inicializamos el repositorio Git en VSCode (**git init**)

- Hago el primer commit en el icono (check).

- Lo sincronizo a mi cuenta de github.com:

    git remote add origin https://github.com/Belindavf/crud-laravel

- Le doy a la nube de la barra inferior azul y compruebo que el código se ha subido a *https://github.com/*

- Para probar como funciona MDBootstrap descargamos unas hojas de estilos, pinchando en la versión free (**DIRECT DOWNLOAD**) de https://mdbootstrap.com/docs/jquery/getting-started/download/

- La guardamos en la carpeta *public* con el nombre de *estilos-MDBootstrap_4.11*

- El controlador *router/web.php* nos deriva a la página welcome que está en *resources/view/welcome.blade.php*

- Crear en *crud/resources/views/layout* los archivos *navbar.blade.php*, *main.blade.php* y *footer.blade.php*

- Añadimos a *main.blade.php* el contenido de *crud/public/estilos-MDBootstrap_4.11/index.html*

- Si en el *welcome.blade.php* ponemos una sola línea @extends('layouts.main'), podemos ver lo que se tenga en el cuerpo del *main.blade.php*, que en este caso sería:

```
<!-- Start your project here-->
@include('layouts.navbar')
@yield('content')
@include('layouts.footer')
<!-- End your project here-->
```

- Completamos la página *crud/resources/views/welcome.blade.php* con este código:

```
@extends('layouts.main')
@section('content')
<h1>Página HOME</h1>
@endsection
```

- Con 'localhost' comprobamos en el navegador que se visualice el contenido html. Deberíamos ver el texto: **Página HOME**

- Para que la página principal vaya cogiendo forma, copiamos en el *navbar.blade.php* el código **Basic example** de https://mdbootstrap.com/docs/jquery/navigation/navbar/ y copiamos en el *footer.blade.php* el código **Copyright** de https://mdbootstrap.com/docs/jquery/navigation/footer/

- Crear una base de datos en phpMyAdmin llamada "*laravel_crud*" con el usuario "*laravel_crud_user*" y la contraseña "*abc123.*"

- Modificar en el fichero *.env* el contenido referente a la base de datos, con la que creamos nosotros:

(línea 12)

```
DB_DATABASE=laravel_crud
DB_USERNAME=laravel_crud_user
DB_PASSWORD=abc123.
```

- Modificamos el código de *web.php* añadiendo estas tres rutas:

```
Route::get('/', function () {
   return view('welcome');
});
Route::get('/hola', function(){
   return "¡Hola mundo!";
});
Route::get('/user/{id}', function($id){
   return "Mi código es: " . $id;
});
```

- Visualización en el navegador. Según la ruta de búsqueda veremos diferentes resultados.

localhost/
localhost/hola
localhost/321

- Generar un controlador con el siguiente comando en el terminal de VSCode:

php artisan make:controller StudentController

Lo abro desde *crud/app/Http/Controllers/StudentController.php* y creo el método index:

```
class StudentController extends Controller {
        public function index() {
                return view ('welcome');
        }
}
```

- Comentamos la primera ruta del código *web.php* para añadir esta en su lugar:

```
Route::get('/',  'StudentController@index') -> name('home');
```

- Genero otro controlador

php artisan make:controller TestController --resource

- Creamos un modelo con

php artisan make:model Students -m

que crea un fichero *database/migrations/2020_01_15_100904_create_students_table.php* en dónde definimos los campos de la tabla Students:

```
public function up() {
        Schema::create('students', function (Blueprint $table) {
                $table->bigIncrements('id');
                $table->string('nombre');
                $table->string('apellidos');
                $table->string('email');
                $table->string('telefono');
                $table->timestamps();
        });
}
```

- Migramos todos los esquemas con el comando **php artisan migrate** y comprobamos el resultado en MySql

- Añadimos esta ruta en *web.php*:

```
Route::get('/create',  'StudentController@create') -> name('create');
```

- Creamos la función create() en *StudentController.php* :

```
public function create() {
        return view ('Hola voy a crear algo grande');
}
```

- Editar código de *navbar.blade.php*:

```
<!-- Links -->
  <ul class="navbar-nav mr-auto">
    <li class="nav-item active">
      <a class="nav-link" href="{{url('/')}}">Home
        <span class="sr-only">(current)</span>
      </a>
    </li>
    <li class="nav-item">
      <a class="nav-link" href="{{url('/create')}}">Añadir</a>
    </li>
    <li class="nav-item">
      <a class="nav-link" href="#">Pricing</a>
    </li>
```

- Cambiamos en *app/Http/Controlers/StudentController.php* esta línea:

```
                //return view ('Hola voy a crear algo grande');
```

por esta: `return view('create');`

- Creamos una página: *resources/views/create.blade.php*

Copio el código de *welcome.blade.php* cambiando la líneas del html(<h1>) por el código
Register / Sign up form
Default form register
https://mdbootstrap.com/docs/jquery/forms/basic/

- Lo editamos un poco cambiando los placeholder y textos a español y añadimos atributos name a los campos (nombre, apellidos, email, telefono).

- Añadimos la siguiente ruta en *web.php*:

```
//si se llama por get va a /create
Route::post('/create',  'StudentController@store') -> name('store');  //si se llama por post va a /store
```

- Añadimos/cambiamos estos dos atributos de la etiqueta form en *create.blade.php*:

```
action="{{route('store') }}" method="POST">
```

y añadimos a continuación:

`{{ csrf_field() }}` ----> **sistema de seguridad de laravel que se utiliza para que use este formulario y envíe los datos generados en ese momento**

- Editamos *StudentController.php*:

```
use App\Students;
class StudentController extends Controller{
        public function store(Request $request) {
                $this->validate($request,[
                        'nombre' => 'required',
                        'email' => 'required',
                        'telefono' => 'required'
                        ]
                );
                //base de datos = names de create.blade.php
                $estudiante = new Students;
                $estudiante->nombre = $request->nombre;
                $estudiante->apellidos = $request->apellidos;
                $estudiante->email = $request->email;
                $estudiante->telefono = $request->telefono;
                $estudiante->save();
```

```
                return redirect(route('home'))->with('successMsg', 'Estudiante añadido correctamente');
        }
}
```

- Ahora desde el navegador con http://localhost/create si añadimos en 'sign in' un nuevo alumno debería aparecer en nuestra base de datos (*laravel_crud*) en la tabla students el nuevo alumno creado con su nombre, apellidos, email y teléfono.

- Añado un alert en el *welcome.blade.php* para saber que insertamos correctamente:

```
@if (session('successMsg'))
  <div class="alert alert-success" role="alert">
    {{ session('successMsg') }}
  </div>
@endif
```

- Añado un alert en el *create.blade.php* para ver errores de validación:

```
@if ($errors->any())
        @foreach($errors->all() as $error)
                <div class="alert alert-danger" role="alert">
                        {{ $error }}
                </div>
        @endforeach
@endif
```

- Comprobamos en al navegador que funcionan los alert:

-> mensaje de color verde en caso de datos introducidos correctamente

-> mensaje de color rojo en caso de que exista algún error

- Crear en la página home una tabla copiada de MDBootstrap que se adapte a nuestro contenido, para la consulta de todos los registros y poder añadir botones para realizar acciones (editar, eliminar alumnos) en *resources/views/welcome.blade.php*:

```
@endif
<table class="table">
 <thead class="black white-text">
  <tr>
   <th scope="col">#</th>
   <th scope="col">Nombre</th>
   <th scope="col">Apellidos</th>
   <th scope="col">E-mail</th>
```

```
    <th scope="col">Teléfono</th>
    <th scope="col">Acciones</th>
   </tr>
  </thead>
  <tbody>
  @foreach ($estudiantes as $estudiante)
   <tr>
    <th scope="row">{{$estudiante->id}}</th>
    <td>{{$estudiante->nombre}}</td>
    <td>{{$estudiante->apellidos}}</td>
    <td>{{$estudiante->email}}</td>
    <td>{{$estudiante->telefono}}</td>
    <td>
    <a class="btn btn-raised btn-primary btn-sm" href=""> <i class="fas fa-edit"></i></a>
    <a class="btn btn-raised btn-danger btn-sm" href=""> <i class="fas fa-trash-alt"></i></a>
    </td>
   </tr>
  @endforeach
  </tbody>
</table>
@endsection
```

- Editar el index en el controlador *StudentController.php* para pasar todos los estudiantes:

```
public function index() {
        //return view ('welcome');
        $estudiantes = Students::all();
        return view('welcome', compact('estudiantes'));
}
```

- Creamos dos rutas nuevas para realizar las modificaciones (siempre en *routes/web.php*):

```
Route::get('/edit/{id}',  'StudentController@edit') -> name('edit');
Route::post('/update/{id}',  'StudentController@update') -> name('update');
```

- Creamos el controlador (en *StudentController.php*):

```
public function edit($id){
        //buscar la id en la base de datos:
        $estudiante = Students::find($id);
        //modelo vista controlador:
        return view('edit', compact('estudiante'));
```

}

- Crear la página edit -> *resources/views/edit.blade.php*

Copiamos el código del *create.blade.php* pero

Cambiamos la acción del <form>:

```
<form class="text-center border border-light p-5" action="{{ route('update', $estudiante->id) }}" method="POST">
```

Añadimos el valor a cada input:

```
<input name="nombre" value="{{ $estudiante->nombre }}" ...>
<input name="apellidos" value="{{ $estudiante->apellidos }}" ...>
        ...     value="{{ $estudiante->email }}" >
        ...     value="{{ $estudiante->telefono }}" >
```

Cambiamos el texto del botón 'Sign in' por 'Modificar':

```
<button class="btn btn-info my-4 btn-block" type="submit">Modificar</button>
```

- Añadir la ruta al elemento <a> del *welcome.blade.php*

```
<a class="btn btn-raised btn-primary btn-sm" href="{{route('edit',$estudiante->id)}}"> <i class="fas fa-edit"></i></a>
```

- Copiamos la función **public function store(Request $request)** de *StudentController.php* y creamos una función update editando las siguientes líneas:

```
public function update(Request $request, $id) {            <-------
    $this->validate($request,[
        'nombre' => 'required',
        'email' => 'required',
        'telefono' => 'required'
        ]
    );
    //base de datos = names de create.blade.php
    $estudiante = Students::find($id);                     <-------
    $estudiante->nombre = $request->nombre;
    $estudiante->apellidos = $request->apellidos;
    $estudiante->email = $request->email;
    $estudiante->telefono = $request->telefono;
    $estudiante->save();
    return redirect(route('home'))->with('successMsg', 'Estudiante modificado'); <-------
```

```
}
```

- Para borrar un alumno de forma simple bastaría con:

Crear una nueva ruta (siempre en *web.php*):

```
//                  fichero creado en StudentController -> nombre que se la da
Route::get('/delete/{id}', 'StudentController@delete') -> name('delete');
```

Añadir la ruta al botón de eliminar, en *welcome.blade.php*:

```
<a class="btn btn-raised btn-danger btn-sm" href="{{route('delete',$estudiante->id)}}" > <i class="fas fa-trash-alt"></i></a>
```

Crear el controlador en *StudentController.php*:

```
public function delete($id){
    $estudiante = Students::find($id);
    $estudiante->delete();
    return redirect (route('home'))->with('successMsg','Estudiante borrado');
}
```

- Para un borrado más complejo (con confirmación y método http:delete), se siguen estos pasos:

Cambiamos la ruta del *web.php*

```
Route::delete('/delete/{id}', 'StudentController@delete') -> name('delete');
```

En *welcome.blade.php* introducimos código javascript para el botón de eliminar, cambiamos la etiqueta <a> por un formulario y un botón:

```
<form id="delete_form_{{$estudiante->id}}" action="{{route('delete',$estudiante->id)}}" method="post" style="display: none;" >
        {{csrf_field()}}
        {{method_field('delete')}}
</form>
<button onclick="if(confirm('¿Está seguro de que desea borrar a este alumno?')) {
        document.getElementById('delete_form_{{$estudiante->id}}').submit();
        //hay que poner {{$estudiante->id}} en el form para que borre el alumno en el que se le dio al botón de eliminar, sino borraría el primer alumno de la tabla
} else{
        event.preventDefault(); //Este método cancela el evento si éste es cancelable, sin detener el resto del funcionamiento del evento, es decir, puede ser llamado de nuevo.
}"
```

```
class="btn btn-raised btn-danger btn-sm" href="">
        <i class="fas fa-trash-alt"></i>
</button>
```

- Para paginación introducimos en el controlador (*StudentController.php*) el método **paginate()**, que nos permite ver el número de alumnos registrados en varias páginas, en lugar de todos seguidos en la misma:

```
public function index() {
        //return view ('welcome');
        //$estudiantes = Students::all();
        $estudiantes = Students::paginate(5);  //<-- 5 registros por página
        return view('welcome', compact('estudiantes'));
}
```

- Finalmente añadimos en la vista *welcome.blade.php* dónde queramos tener el control (al final del documento) que nos permite cambiar de una página a otra para poder ver a todos los alumnos:

```
{{$estudiantes->links()}}

@endsection
```