



**Región de
Murcia**

CIFP Carlos III

C/Carlos III, 3
30201 CARTAGENA
TEL: 968 321301
FAX: 968 320111
30019702@murciaeduca.es
www.cifpcarlos3.es



Diseño e implementación de un sistema de gestión de almacenes

**Proyecto Fin de Grado de Desarrollo de
Aplicaciones Web**

Autor: Francisco José Verdú Berdejo

Tutor Académico: José Antoni García Gallego

Centro: CIFP Carlos III

Junio de 2020



**Región de
Murcia**

CIFP Carlos III

C/Carlos III, 3
30201 CARTAGENA
TEL: 968 321301
FAX: 968 320111
30019702@murciaeducas.es
www.cifocarlos3.es



ENSEÑANZAS COFINANCIADAS POR
FONDO SOCIAL EUROPEO (C.F.E.)



Región de
Murcia

CIFP Carlos III

C/Carlos III, 3
30201 CARTAGENA
TEL: 968 321301
FAX: 968 320111
30019702@murciaeduca.es
www.cifp-carlos3.es



ENSEÑANZAS COFINANCIADAS POR
FONDO SOCIAL EUROPEO (C.F.E.)

ÍNDICE

1.- Introducción y Objetivo del proyecto

1.1- Introducción

1.1- Objetivo del proyecto

2.- Herramientas y tecnología

2.1- Lenguajes

2.1.1.- PHP

2.1.2.- JavaScript

2.1.3.- HTML5

2.1.4.- CSS3

2.1.5.- SASS

2.1.6.- Node.js

2.2- Frameworks

2.2.1.- Laravel

2.2.1.1- Artisan

2.2.1.2.- Sistema de rutas

2.2.1.3.- Eloquent ORM. Modelos

2.2.1.3.1.- ORM

2.2.1.3.2.- Eloquent

2.2.1.3.3.- Modelos

2.2.1.4.- Migraciones

2.2.1.5.- Middleware

2.2.1.6.- Requests

2.2.1.7.- Políticas (Policies)

2.2.1.8.- Controladores

2.2.1.9.- Vistas

2.2.1.10.- Sistema de archivos

2.2.1.11.- Sistemas de traducción

2.2.2.- JQuery

2.2.3.- Bootstrap

2.3.- Otras herramientas

2.3.1.- Visual Studio Code

2.3.2.- MySQL Workbench

2.3.3.- Laragon

2.3.4.- MySQL

2.3.5.- Navegadores

2.3.6.- Git

2.3.7.- Composer

3.- Análisis funcional

3.1.- Documento de Requisitos

3.2.- Diseño de base de datos

3.2.1.- Resumen

3.2.2- Procedimientos almacenados (stored procedures)



Región de
Murcia

CIFP Carlos III

C/Carlos III, 3
30201 CARTAGENA
TEL: 968 321301
FAX: 968 320111
30019702@murciaeduca.es
www.cifpccarlos3.es



ENSEÑANZAS COFINANCIADAS POR
FONDO SOCIAL EUROPEO (C.F.E.)

3.2.3- Triggers (disparadores)

3.2.4- Active Record

4.- Arquitectura del sistema

4.1- Patrón MVC

4.2.- API REST

5.- Interfaces de usuario

6.- Despliegue del sistema web

7.- Resultado final

8.- Conclusiones

8.1.- Conclusiones

8.2.- Trabajo futuro

9.- Bibliografía

10.- Anexos

10.1.- Anexo A. Instalación Laragon

10.2.- Anexo C. Instalación PHP

10.3.- Anexo B. Instalación Composer

10.4.- Anexo D. Creación DDNS y configuración router

10.5.- Anexo C. Establecer una cuenta de correo de gmail en Laravel

10.6.- Anexo E. Manual de Usuario (Documento adjunto al proyecto)

10.7.- Anexo F. Modelo de base de datos ER (Imagen y fichero de modelado .mwb adjunto al proyecto)

10.8.- Anexo H. Estructura de Vistas Blade Imagen y modelo .uml adjunto al proyecto)

10.9.- Anexo I. Jerarquía de modelos principales (Imagen y fichero .uml adjunto al proyecto)

10.10.- Anexo J. Jerarquía de controladores creados (Imagen y fichero .uml adjunto al proyecto).

10.11.- Anexo K. Manual de Usuario (Se adjunta en el proyecto un documento PDF que será el manual del usuario)



1. Introducción

1.1 Planteamiento del proyecto

Hoy en día, es indispensable el uso de un software que ayude a gestionar la información. El acceso a la información debe ser rápida, coherente y sin errores.

El objetivo principal del proyecto es el desarrollo de una aplicación web para que facilite la gestión de un almacén genérico.

Para ello, haremos uso de Laravel como framework web, y otras tecnologías como HTML5, JavaScript, CSS, Bootstrap, SASS para el front-end y Node.js y MySQL para el back-end para llevarlo a cabo. La comunicación cliente-servidor seguirá una estructura REST.

La aplicación permitirá la definición de los productos que utilizará el usuario, junto a la jerarquía que seguirá. También permitirá la creación de usuarios y roles. Cada rol permitirá una serie de funcionalidades como son el acceder a un recurso determinado para visualizar información, imprimir o inclusive el crear o modificar entidades (productos, marcas, proveedores, pedidos, etc...).

Adicionalmente, también permitirá la definición de los almacenes o proveedores como entidades principales para llevar a cabo la gestión del almacén.

En base a todas estas definiciones, el sistema dará acceso al usuario a crear documentos como pedidos de compra, recepciones trazadas frente al pedido de compra o inclusive visualizar el stock del almacén o la posibilidad de regularizar su stock por almacén y producto.

Separaremos el proyecto en varias secciones. Primero explicará las distintas tecnologías empleadas para informar al usuario de su uso. Posteriormente se documentará la potencia del framework web utilizado y una breve explicación de la estructura que seguirá el código desarrollado. A continuación, hablará del modelo de datos creado que creará los cimientos de la aplicación y finalmente dará paso a explicar el despliegue de la aplicación en un servidor web.

1.2. Objetivos del proyecto

Los objetivos del proyecto son:

- Aprender sobre las tecnologías web empleadas: HTML5, CSS3, Node.js, MySQL, Bootstrap, JavaScript, Laravel
- Ser capaz de desarrollar una aplicación web desde cero, empezando desde su diseño, desarrollo y posterior implementación.



Región de
Murcia

CIFP Carlos III

C/Carlos III, 3
30201 CARTAGENA
TEL: 968 321301
FAX: 968 320111
30019702@murciaeduca.es
www.cifpcarlos3.es



ENSEÑANZAS COFINANCIADAS POR
FONDO SOCIAL EUROPEO (C.F.E.)

2. Herramientas y tecnologías

2.1. Lenguajes

2.1.1. PHP

PHP (acrónimo de PHP Hypertext Preprocessor) es un lenguaje de programación de código abierto utilizado principalmente para el desarrollo web del lado del servidor y que puede ser incrustado en HTML. PHP fue creado por Rasmus Lerdorf.

Una de las facilidades con la que cuenta PHP es que es un lenguaje que puede ser instalado en la mayoría de servidores web sin ningún coste. Suele ser procesado por un intérprete PHP implementado como un módulo en el servidor web o implementado como un CGI (Common Gateway Interface) ejecutable.

El lenguaje PHP no ha contado con un estándar hasta 2014, con el lanzamiento de la versión 5.6.

Actualmente, PHP se encuentra en la versión 7.3, aunque la versión utilizada es la 7.2.

PHP cuenta con una gran comunidad. Además es un lenguaje maduro donde podemos encontrar fácilmente una gran cantidad de información, tanto en idioma español como en inglés. PHP es uno de los lenguajes más utilizados para desarrollar páginas web que necesiten servicio del lado del servidor. Páginas como Wikipedia, Yahoo! o Facebook han sido desarrolladas con PHP.

2.1.2. JavaScript

Javascript (abreviado comúnmente JS) es un lenguaje de programación interpretado de alto nivel y dinámico. Pertenece al estándar ECMAScript. Es utilizado principalmente en el desarrollo web en el lado del cliente (front-end).

JavaScript se utiliza principalmente para el desarrollo web en el lado del cliente, aunque también se puede utilizar para el desarrollo del lado de servidor (back-end). Es utilizado para manipular el DOM, mejorar la interfaz, ampliar las funcionalidades de la web añadir animaciones, etc.

2.1.3. HTML5

HTML (HyperText Markup Language), es un lenguaje de marcado para el desarrollo de páginas web y aplicaciones web. En combinación con CSS y Javascript, forman los pilares básicos para el desarrollo web.

HTML5 es la última versión del estándar. Se añaden nuevos elementos y atributos para adaptarse a las páginas web actuales, como por ejemplo: <header> para destacar la



Región de
Murcia

CIFP Carlos III

C/Carlos III, 3
30201 CARTAGENA
TEL: 968 321301
FAX: 968 320111
30019702@murciaeduca.es
www.cifpccarlos3.es



ENSEÑANZAS COFINANCIADAS POR
FONDO SOCIAL EUROPEO (C.F.E.)

cabecera de una web, <nav> para definir el menú de navegación de una web, o <video> para definir un elemento multimedia para reproducir un vídeo.

También, añade una serie de cambios en la semántica básica que tenía HTML hasta su versión 5.

Esta es la versión HTML utilizada para este proyecto. Para ejecutarlo se necesitan navegadores actuales y se ha de tener en cuenta que no todos los navegadores soportan todas las etiquetas nuevas.

2.1.4. CSS

CSS (Cascading Style Sheets) es un lenguaje utilizado para desarrollar y definir el estilo de un documento de HTML. Se utiliza para establecer el diseño de una página web o para diseñar interfaces de usuario escritas en HTML. La versión de CSS que se utiliza en el proyecto, durante su redacción, es la versión 3, que es la última publicada.

CSS se diseñó para estructurar mejor el desarrollo de páginas web, pudiendo separar el contenido del documento de los estilos del documento.

CSS cuenta con su propia sintaxis, y se basa en herencia en cascada, es decir, si el elemento padre tiene definidas unas propiedades, los hijos también heredarán estas propiedades.

Con CSS podríamos crear sombras, degradados, modificar colores de fuentes, etc.

Una funcionalidad muy buena de CSS son los media queries, los cuales permiten diferenciar estilos en función del dispositivo que se utilice. A este desarrollo se le conoce como diseño adaptativo (responsive design).

2.1.5. SASS

Sass (Syntactically Awesome Stylesheets) es un metalenguaje de Hojas de Estilo en Cascada (CSS). Es un lenguaje de script que es traducido a CSS.

Fue diseñado inicialmente por Hampton Catlin y desarrollador por Natalie Weizenbaum. Después de sus versiones iniciales, Nathan Weizenbaum y Chris Eppstein han continuado extendiendo Sass con SassScript, un lenguaje de script simple, utilizado en los ficheros Sass.

SASS da la posibilidad de convertir los CSS en algo dinámico. Permite trabajar mucho más rápido en creación de código en CSS. Permite reutilizar código gracias a la incorporación de mixins (variables que nos permiten guardar valores), funciones, o incluso el concepto de herencia, donde puedes crear clases o extenderlas para variarlas ligeramente.

SASS dispone de dos formatos diferentes para la sintaxis, una con extensión .sass y otra .scss.



Región de
Murcia

CIFP Carlos III

C/Carlos III, 3
30201 CARTAGENA
TEL: 968 321301
FAX: 968 320111
30019702@murciaeduca.es
www.cifpcarlos3.es



ENSEÑANZAS COFINANCIADAS POR
FONDO SOCIAL EUROPEO (C.F.E.)

El primero en salir fue .sass, que se caracteriza por no hacer uso de llaves, ni punto y coma final. Los .SCSS salieron en la versión 3 del preprocesador, donde se permite el uso de llaves y el uso de código de CSS clásico.

Tanto la sintaxis de .sass como de .scss no puede ser interpretada por el navegador, por lo tanto hay que compilar para transformar el código a CSS.

2.1.6. Node.js

Node.js Es un entorno en tiempo de ejecución multiplataforma, de código abierto, para la capa del servidor y basando en el lenguaje de programación JavaScript, asíncrono, con E/S de datos en una arquitectura orientada a eventos.

Creado por Ryan Dahl en 2009. Fue creado con el enfoque de ser útil en al creación de programas de red altamente escalables, como por ejemplo, servidores web.

Node.js no se ejecuta en un navegador, sino en el servidor. Implementa algunas especificaciones de CommonJS, y ejecuta V8 JavaScript, el motor subyacente que Google usa con su navegador Chrome. Al igual que Apache, Node también tiene el concepto de módulos que se pueden agregar a su núcleo mismo.

Node, está diseñado específicamente para situaciones en la que se espere una gran cantidad de tráfico y donde la lógica del lado del servidor y el procesamiento requeridos sea sencillo para responder rápidamente al cliente.



2.2. Frameworks y librerías

¿Qué es un framework web?

Podemos definir un framework web como una estructura de software formada por diferentes componentes que facilitan el desarrollo de una aplicación. La mayoría de los frameworks web funcionan de una manera muy similar.

El objetivo de los frameworks web hacer cómodo, elegante y simple el desarrollo de estas aplicaciones o sistemas web, sin necesidad de reescribir código.

2.2.1. Laravel

Laravel es un framework web de código abierto, el cual se utiliza para el desarrollo de aplicaciones y sistemas web.

La filosofía de Laravel es que el desarrollo de las aplicaciones sea de una forma simple y vistosa..

Gran parte de Laravel está formado por dependencias de otros sistemas, como symphony, por lo que su desarrollo depende de estas dependencias.

Características

Laravel incluye muchas funcionalidades que facilitan al usuario su trabajo, como puedan ser:

1. Patrón MVC: Laravel se basa en el patrón MVC. Dicho patrón divide el software en 3 componentes principales (modelo, vista, controlador).
2. Motor de plantillas: Laravel utiliza el motor de plantillas Blade. También aporta funcionalidades como el uso de estructuras de control, o permitir que las plantillas hereden unas de otras.
3. Artisan: Es la interfaz de línea de comandos de Laravel. Viene con una serie de comandos que hacen más fácil el desarrollo de una aplicación web en Laravel, ya sea la creación de los modelos, middlewares, requests...
4. Sistema de rutas: El sistema de generación de rutas de Laravel es sencillo e intuitivo. Se pueden crear rutas diferenciando los métodos HTTP: GET, POST, PUT, PATCH, DELETE ...
5. Object Relational Mapping (ORM): Transformar las tablas de una base de datos en modelos. Esto hace más sencillo las tareas básicas con las bases de datos (CRUD).
6. Sistema de autenticación: Por defecto, Laravel utiliza un módulo de autenticación basado en sesiones, login, comprobación de contraseñas, etc.
7. Migraciones: Las migraciones son archivos que aportan un sistema de control de versiones. Se utiliza para crear tablas, modificaciones o eliminar tablas, columnas, etc. Utiliza programación orientada a objetos para realizar tales funciones.



Región de
Murcia

CIFP Carlos III

C/Carlos III, 3
30201 CARTAGENA
TEL: 968 321301
FAX: 968 320111
30019702@murciaeduca.es
www.cifpccarlos3.es



En la actualidad, Laravel es uno de los frameworks web más utilizados, contando con una gran comunidad creciente. La versión actual de Laravel, utilizada en este proyecto es la 7.0.

2.2.1.1. Artisan

Artisan es la interfaz de línea de comandos de Laravel. Tiene una serie de comandos que facilitan el desarrollo de una aplicación en Laravel.

La sintaxis básica del comando es:

```
php artisan <acción>
```

El comando más básico para ver la lista de comandos completa es:

```
php artisan list
```

Algunos de los comandos de artisan utilizados son:

1. Crear un controlador: `php artisan make:controller <nombre del controlador>`
2. Crear un modelo: `php artisan make:model <nombre del modelo>`
3. Crear una vista: `php artisan make:view <nombre de la vista>`
4. Crear una migración: `php artisan make:migration <nombre de la migración>`
5. Crear un middleware: `php artisan make:middleware <nombre del middleware>`
6. Crear un request: `php artisan make:request <nombre del request>`
7. Crear un recurso: `php artisan make:resource <nombre del recurso>`
8. Crear una política: `php artisan make:policy <nombre de la política>`
9. Crear una semilla: `php artisan make:seeder <nombre del seeder>`
10. Realizar la migración para borrar las tablas y crearlas desde 0, y también para llenarlo de datos con los seeders: `php artisan migrate:fresh --seed`
11. Listar todas las rutas habilitadas en el servidor web: `php artisan route:list`
12. Crear un fichero de caché para aumentar la velocidad de carga: `php artisan config:cache`

2.2.1.2. Sistema de rutas

Laravel cuenta con un sistema de rutas. Es decir, una serie de archivos en los que se definen las diferentes direcciones que va a tener una página web. Por ejemplo:

<http://gestionalo.com> => Ruta raíz

<http://gestionalo.com/articulos> => Una de las posibles direcciones.

La definición de las rutas se hace en los archivos `api.php`, `web.php` o `console.php` respectivamente. Estos ficheros están en el directorio Routes.

La definición de rutas es simple. Para crear una nueva ruta habrá que utilizar la siguiente sintaxis:

```
Route::get('productos', 'ProductosController@index')->name('productos');
```



Región de
Murcia

CIFP Carlos III

C/Carlos III, 3
30201 CARTAGENA
TEL: 968 321301
FAX: 968 320111
30019702@murciaeduca.es
www.cifpcarlos3.es



ENSEÑANZAS COFINANCIADAS POR
FONDO SOCIAL EUROPEO (C.F.E.)

<métodoHTTP>: Debe ser uno de los posibles métodos HTTP: get, post, put, patch, options o delete.

<nombre de la ruta>: Es la URI (Universal Resource Identifier). Por ejemplo: 'articulos'

<función a realizar>: Es la función a la que se llamará cuando se acceda a esa ruta.

Un ejemplo de ello sería:

```
Route::get('ejemplo', function(){  
    return 'Esto es un ejemplo';  
});
```

Esto hará que, cuando un usuario acceda a la ruta /ejemplo, vea en el navegador 'Esto es un ejemplo'.

A una ruta, se le puede pasar una función definida en el mismo método, o una función definida en un controlador.

Por ejemplo:

Siendo ProductosController una clase y @index el método definido dentro de la clase ProductosController.

Al definir la URI de una ruta, también podemos pasarle un parámetro, como puede ser un ID, un código, o el propio objeto. Por ejemplo:

```
Route::get('productos/{producto}', function ($producto){  
    return 'El producto tiene el id: '. $producto->id;  
});
```

```
Route::get('productos/{id}', function ($id){  
    return 'El producto tiene el id: '. $id;  
});
```

Por comodidad, se han definido todas las rutas en el archivo web.php.

2.2.1.3 Eloquent. ORM. Modelos

2.2.1.3.1. ORM

¿Qué es un ORM?

Un ORM (Object Relational Mapping, o Mapeo Objeto-Relacional) es una técnica de programación que transforma los datos de las tablas de una base de datos en objetos,



Región de
Murcia

CIFP Carlos III

C/Carlos III, 3
30201 CARTAGENA
TEL: 968 321301
FAX: 968 320111
30019702@murciaeduca.es
www.cifpcarlos3.es



ENSEÑANZAS COFINANCIADAS POR
FONDO SOCIAL EUROPEO (C.F.E.)

utilizando OOP (Programación orientada a objetos). Posteriormente todas las operaciones se hacen con objetos.

¿Por qué utilizar un ORM?

Un ORM facilita mucho la vida al desarrollador, permitiendo abstraerse de las operaciones de la base de datos. También es sencillo de utilizar si se tiene algo de conocimiento básico en programación orientada a objetos.

2.2.1.3.2. Eloquent

Laravel incluye Eloquent como ORM. Las funciones que Eloquent aporta facilita la realización de consultas y peticiones complejas a nuestra base de datos sin tener que programar en SQL.

Tanto su implementación como su manejo con el patrón “ActiveRecord” es sencillo a la hora de funcionar con la base de datos.

Eloquent transforma cada tabla de la base de datos con el “modelo-objeto” que le corresponde. Dicho modelo-objeto será el que utilice Laravel para interactuar con la tabla.

Los modelos pueden hacer consultas a las tablas de manera transparente al usuario. También da acceso a insertar, modificar o eliminar registros en las tablas.

Las tablas de la base de datos se definen por medio de migraciones y su modelo correspondiente.

2.2.1.3.3. Modelos

Uno de los componentes del patrón MVC es el modelo. Es el componente que trabaja con los datos. Es quien accede a la información, almacenada usualmente en una base de datos.

Modelos en Laravel y relación con Eloquent

Eloquent utiliza los Modelos para recibir o enviar información a la base de datos.

Necesitamos la consola de comandos para generar un comando. Utilizaremos Artisan para lograrlo. El comando para crear un modelo es el siguiente: `php artisan make:model <nombre del modelo>`. Se guardará un archivo con el mismo nombre del modelo en el directorio app.

Por ejemplo, para generar el modelo Producto:

```
php artisan make:model Producto
```

En el propio modelo que creamos, podemos definir el nombre de la tabla que se creará en la base de datos, el cual corresponderá finalmente al Modelo. Podemos especificar los



Región de
Murcia

CIFP Carlos III

C/Carlos III, 3
30201 CARTAGENA
TEL: 968 321301
FAX: 968 320111
30019702@murciaeduca.es
www.cifpcarlos3.es



ENSEÑANZAS COFINANCIADAS POR
FONDO SOCIAL EUROPEO (C.F.E.)

datos que se pueden manipular, datos que serán ocultos, definir algún campo o campos para identificar de manera única un registro concreto (Código, nombre, ID, etc), las relaciones que podrá utilizar el modelo para relacionarse con otros modelos. Por ejemplo un producto con su marca si está en otra tabla, ...etc

Por defecto, si el modelo no tiene especificado un nombre de tabla, buscará como tabla el nombre del modelo y se le añadirá una -s final del nombre del modelo. Es decir, si la tabla se denomina en base de datos 'productos', y el modelo se llama 'Producto', Eloquent deducirá que la tabla del modelo 'Producto' es 'productos'.

También, si se hace uso de objetos para acceder a las URIs, el modelo hará uso del campo {id} del objeto para realizar las operaciones CRUD en el sistema.

Si queremos utilizar otro campo, tendremos que especificarlo en el modelo.

Eloquent también permite convertir tipos de objetos a otros tipos. Para ello se hace uso del \$cast.

Un ejemplo donde personalizamos el modelo es User:

```
<?php

namespace App;

use Illuminate\Contracts\Auth\MustVerifyEmail;
use Illuminate\Foundation\Auth\User as Authenticatable;
use Illuminate\Notifications\Notifiable;

class User extends Authenticatable
{
    use Notifiable;

    protected $guarded = ['id', 'created_at', 'updated_at'];
    protected $hidden = [
        'password', 'remember_token',
    ];
    protected $casts = [
        'email_verified_at' => 'datetime',
    ];

    public function getRouteKeyName(){ return 'codigo'; }
    public function rol(){
        return $this->belongsTo(Rol::class, 'role_id');
    }
}
```



Región de
Murcia

CIFP Carlos III

C/Carlos III, 3
30201 CARTAGENA
TEL: 968 321301
FAX: 968 320111
30019702@murciaeduca.es
www.cifpcarlos3.es



ENSEÑANZAS COFINANCIADAS POR
FONDO SOCIAL EUROPEO (C.F.E.)

El ORM obtiene la información de los modelos. Para ello, dispone de una serie de métodos para poder hacer consultas a la información del modelo. Por ejemplo:

```
<?php

namespace App\Http\Controllers;

use App\User;

class UserController extends Controller
{
    public function index(Request $request)
    {
        return view('usuarios.index',
            ['usuarios' => User::all()]);
    }
}
```

Con el método all(), se recuperan todos los registros existentes en el modelo User. En el ejemplo en cuestión, recupera los registros y se los pasa a la vista ubicada en usuarios.index. Dentro de dicha vista, se utilizan los datos para representarlos y listar todos los usuarios.

Eloquent también permite hacer joins entre tablas de manera cómoda e incluso inyectar SQL en crudo si la consulta es compleja.

Por ejemplo:

```
public function index(Request $request)
{
    $nombre = $request->get('buscarpor');

    return view('articulos.productos.index', ['productos' =>
        Producto::join('marcas', 'marca_id', '=', 'marcas.id')
            ->join('subfamilias', 'subfamilia_id', '=', 'subfamilias.id')
            ->join('familias', 'subfamilias.familia_id', '=', 'familias.id')
            ->join('impuestos', 'impuesto_id', '=', 'impuestos.id')
            ->whereRaw("UPPER(productos.codigo) like UPPER('%".$nombre."%') OR
                UPPER(productos.nombre) like UPPER('%".$nombre."%') ")
            ->select('productos.*')
            ->paginate(5)];
}
```

En el ejemplo en cuestión, se utiliza el Input Buscar de la web a través del Objeto Request. Posteriormente se hace join con distintas tablas a partir del modelo Producto. Se utiliza la cláusula 'whereRaw' para inyectar SQL directo en la consulta que hará Eloquent y se indica



que debe devolver la tabla productos. También le indicamos con la cláusula 'paginate' que la información debe ser devuelta de 5 registros en 5 registros.

La información será devuelta mediante objetos Productos a la vista `articulos.productos.index` de Blade.

Eloquent permite tanto crear nuevos registros, actualizarlos o eliminarlos.

Para crear un nuevo registros, hay que crear previamente una instancia del modelo. Por ejemplo:

```
$rol = new Rol([
    'codigo' => 'Ejemplo1',
    'nombre' => 'Ejemplo2',
    'permisosrole_id' => 0
]);
$rol->save();
```

En este ejemplo, creamos una instancia del modelo Rol, le asignamos los valores que creemos convenientes y luego lo guardamos con `$rol->save()`.

Para modificar un registro, podríamos hacer lo siguiente. En lugar de `save()`, utilizar `update()`.

```
$rol = Rol::find(1);
$rol->codigo = 'Ejemplo';
$rol->update();
```

Para borrar el registro, sería similar. Pero ahora con el método `delete()`.

```
$rol = Rol::find(1);
$rol->delete();
```

Existen más métodos para operar con los modelos, pero estos son los más básicos.

2.2.1.4. Migraciones

Las migraciones son un sistema de control de versiones de la base de datos, nos da acceso a crear tablas o columnas en la base de datos o inclusive a hacer un mantenimiento de estas (editar o borrar) de manera cómoda.

Podemos definir las tablas por medio de programación orientada a objetos (PDO) en vez de usar lenguaje SQL para realizar los procesos. Al abstraerse del lenguaje SQL, nos permite portar el proyecto a distintos sistemas de gestión de bases de datos, como por ejemplo: MySQL, PostgreSQL, SQLite, SQL Server....

¿Cómo se genera una migración?



Región de
Murcia

CIFP Carlos III

C/Carlos III, 3
30201 CARTAGENA
TEL: 968 321301
FAX: 968 320111
30019702@murciaeduca.es
www.cifpcarlos3.es



ENSEÑANZAS COFINANCIADAS POR
FONDO SOCIAL EUROPEO (C.F.E.)

Para crear una migración, debemos usar la consola de comandos y el comando Artisan.
por ejemplo:

```
php artisan make:migration create_table_productos
```

Con este comando le podemos añadir diferentes parámetros, como --table o --create, que añadirán distintas funcionalidades a nuestra migración (como crear una plantilla de trabajo básica).

Estructura de una migración

Una migración se compone por los métodos down y up.

1. **UP** se utiliza para crear tablas, claves foráneas, columnas adicionales, etc en la base de datos.
2. **DOWN** desempeña el efecto contrario que el método **UP**, es decir, borra tablas, columnas, etc.

Con estos métodos, utilizaremos el constructor de esquemas (Schema Builder) de Laravel para crear o modificar tablas.

```
<?php

use Illuminate\Database\Migrations\Migration;
use Illuminate\Database\Schema\Blueprint;
use Illuminate\Support\Facades\Schema;

class CreateAlmacenes extends Migration
{
    public function up()
    {
        Schema::create('geolocalizaciones', function (Blueprint $table)
        {
            $table->id();
            $table->string('latitud');
            $table->string('longitud');
            $table->timestamps();
        });

        Schema::create('almacenes', function (Blueprint $table) {
            $table->id();
            $table->string('codigo');
```




```
$table->unsignedBigInteger('sujeito_id')->unique();
$table->unsignedBigInteger('geolocalizacion_id')->unique();
$table->timestamps();
$table->foreign('sujeito_id')
    ->references('id')->on('sujetos');

$table->foreign('geolocalizacion_id')
    ->references('id')->on('localizaciones');

});
}

public function down()
{
    Schema::dropIfExists('almacenes');
    Schema::dropIfExists('geolocalizaciones');
}
}
```

Como podemos ver en la figura anterior, en la función up creamos las tablas almacenes y geolocalizaciones mediante el método create de la clase Schema, y añadiendo dentro de ella los atributos que tendrán las tablas (columnas).

El Schema Builder

Schema Builder es una clase de Laravel que ayuda a manipular las tablas de manera transparente hacia el usuario.

Ejemplo:

- Schema::create, crea tablas en la base de datos. Necesita que se le defina el nombre de la tabla y una función con los atributos de la tabla.
- Schema::dropIfExists, elimina una tabla de la base de datos en caso de existir la tabla.
- Schema::rename, modificar el nombre de la tabla.

Schema builder cuenta con una amplia variedad de tipos posibles para crear un atributo. En la página oficial de Laravel se especifican todos los atributos disponibles.

Algunos de los tipos son:

Comando	Descripción



Región de
Murcia

CIFP Carlos III

C/Carlos III, 3
30201 CARTAGENA
TEL: 968 321301
FAX: 968 320111
30019702@murciaeduca.es
www.cifpcarlos3.es



<code>\$table->id();</code>	Alias de <code>\$table->bigIncrements('id');</code> .
<code>\$table->foreignId('user_id');</code>	Alias of <code>\$table->unsignedBigInteger('user_id');</code> .
<code>\$table->bigIncrements('id');</code>	Columna equivalente a UNSIGNED BIGINT (key primaria) autoincremental.
<code>\$table->bigInteger('votes');</code>	Columna equivalente a BIGINT
<code>\$table->binary('data');</code>	Columna equivalente a un objeto BLOB
<code>\$table->boolean('confirmed');</code>	Columna equivalente a BOOLEAN
<code>\$table->char('name', 100);</code>	Columna equivalente a CHAR con longitud
<code>\$table->date('created_at');</code>	Columna equivalente a DATE
<code>\$table->dateTime('created_at', 0);</code>	Columna equivalente a DATETIME con precisión

Estos son algunos de los tipos más típicos, se puede consultar todos los tipos posibles en la Documentación de Laravel, en el enlace siguiente:

<https://laravel.com/docs/7.x/migrations>

Otro componente a tener en cuenta son los Seeders, permiten insertar datos por defecto de manera sencilla.



Seeders, Factories y Faker

Los Seeders, componentes utilizados para poblar las tablas de datos.

Su uso principal es para:

1. Crear datos de prueba para trabajar con la aplicación web durante su desarrollo.
2. Configurar el estado de las tablas a las que accederá nuestra aplicación web.

Los Seeders se generan a través de la consola de comandos, mediante el comando maestro artisan:

```
php artisan make:seed <nombre-seeder>
```

Faker es un componente que permite crear datos de prueba de manera sencilla. Puede crear datos aleatorios a partir de funciones predefinidas.

Por ejemplo, para crear un usuario de manera aleatoria, podríamos hacerlo de este modo

```
use Faker\Generator as Faker;
use Illuminate\Support\Str;

$factory->define(App\User::class, function (Faker $faker) {
    return [
        'name' => $faker->name,
        'email' => $faker->unique()->safeEmail,
        'email_verified_at' => now(),
        'password' =>
'$2y$10$92IXUNpkjO0rOQ5byMi.Ye4oKoEa3Ro9llC/.og/at2.uheWG/igi', //
password
        'remember_token' => Str::random(10),
    ];
});
```

2.2.1.5. Middleware

Un middleware es proporciona un mecanismo para filtrar peticiones HTTP realizadas por los usuarios. Reduce la carga de los controladores, y aplica las restricciones definidas en el sistema.

Se utiliza usualmente para autenticar al usuario. Gracias a este componente podemos restringir el acceso a zonas restringidas a usuarios que carecen del permiso para utilizar dicho recurso.



Región de
Murcia

CIFP Carlos III

C/Carlos III, 3
30201 CARTAGENA
TEL: 968 321301
FAX: 968 320111
30019702@murciaeduca.es
www.cifpcarlos3.es



ENSEÑANZAS COFINANCIADAS POR
FONDO SOCIAL EUROPEO (C.F.E.)

¿Cómo se crea un Middleware?

Para crear un Middleware, es necesario utilizar Artisan. Por ejemplo para crear el Middleware CheckUsuario, utilizamos el comando:

```
php artisan make:middleware CheckUsuario
```

Al crear un Middleware, se genera un fichero con el mismo nombre dentro de la carpeta app\http\Middleware.

```
<?php

namespace App\Http\Middleware;

use Closure;

class CheckUsuario
{
    public function handle($request, Closure $next)
    {
        if (!(bool)$request->user()->rol->permisosRol->verPanelUsuarios)
            abort('403', 'No tiene
                autorización para acceder a esta sección');

        return $next($request);
    }
}
```

En este ejemplo, sólo permitiremos dar acceso a nuestra ruta a los usuarios que tengan autorización para ver el panel de usuarios.

Si por el contrario, no tienen acceso, verían una página vacía con el mensaje: “No tiene autorización para acceder a esta sección”.

Para utilizar un Middleware, sería conveniente incluirlo en el archivo app\http\Kernel.php, en el listado de Middlewares. Una vez haya sido definido, podrá ser utilizado.

```
public function __construct()
{
    $this->middleware('auth');
    $this->middleware(CheckUsuario::class);
}
```



Región de
Murcia

CIFP Carlos III

C/Carlos III, 3
30201 CARTAGENA
TEL: 968 321301
FAX: 968 320111
30019702@murciaeduca.es
www.cifpcarlos3.es



ENSEÑANZAS COFINANCIADAS POR
FONDO SOCIAL EUROPEO (C.F.E.)

En este ejemplo en cuestión, cuando se crea el objeto Usuario, se verifica que tiene permisos de autenticación y posteriormente si tiene autorización para acceder al panel de usuarios.

El middleware de autenticación (auth)

Laravel incluye por defecto una serie de Middlewares, uno de ellos es 'auth', el cual permite autenticar al usuario.

Este middleware está incluido por defecto, pero hay que activarlo para poder utilizarlo.

Para ello tendremos que utilizar el comando:

```
php artisan make:auth
```

Al ejecutar este comando, se generarán una serie de archivos, como un controlador y vistas de login y registro de usuarios, necesarios para llevar a cabo la autenticación.

2.2.1.6. Requests

Un Request es un objeto que nos facilita Laravel para consultar información sobre el cliente que realiza la solicitud y los datos que está enviando. Dicho objeto hereda de la clase padre Request.

Request utiliza el patrón de diseño: Inyección de dependencias. Este patrón busca separar la responsabilidad de creación de los objetos de su uso, simplificando y abstrayéndonos de toda la complejidad que puede suponer crear todas y cada una de las dependencias que tenga un código. Cuando una clase necesite de un objeto para completar sus operaciones no lo construye él, sino que lo recibe en el constructor o en los métodos que lo necesiten.

El hecho de recibir los objetos de los que depende una clase por parámetro es lo que se conoce como inyección. La dependencia es aquello que necesita, que no es más que un objeto de una clase.

¿Cómo se crea un request?

Para crear un request es necesario utilizar Artisan. Por ejemplo, para crear el Request SaveUserRequest, utilizamos el siguiente comando:

```
php artisan make:request SaveUserRequest
```

Al crear un Request, se crea un fichero con el mismo nombre dentro de la carpeta app\Requests.

```
<?php  
  
namespace App\Http\Requests;  
  
use Illuminate\Foundation\Http\FormRequest;
```



Región de
Murcia

CIFP Carlos III

C/Carlos III, 3
30201 CARTAGENA
TEL: 968 321301
FAX: 968 320111
30019702@murciaeducas.es
www.cifpcarlos3.es



ENSEÑANZAS COFINANCIADAS POR
FONDO SOCIAL EUROPEO (C.F.E.)

```
use Illuminate\Support\Facades\Hash;

class SaveUserRequest extends FormRequest
{
    public function authorize()
    {
        return true;
    }

    public function rules()
    {
        return [
            'codigo' => 'required|max:20|unique:users',
            ($this->usuario? ',codigo','.$this->usuario->id : '),
            'nombre' => 'required|max:50',
            'password' => $this->getMethod()=='POST'? 'required': '',
            'email' => 'required|unique:users',
            ($this->usuario? ',email','.$this->usuario->id : '),
            'telefono' => 'required|regex:/[0-9]{9}/',
            'role_id' => 'required'
        ];
    }

    public function messages() {
        return [
            'codigo' => 'El usuario necesita un codigo',
            'nombre' => 'El usuario necesita un nombre',
            'password' => 'Debe establecer una password',
            'email' => 'Debe introducir un email valido',
            'telefono' => 'El teléfono debe estar
                        compuesto de 9 digitos',
            'role_id' => 'El usuario requiere un rol'
        ];
    }
}
```

En este ejemplo en cuestión, cuando se hace uso del Request SaveUserRequest podemos ver 3 funciones distintas:

1. authorize: Indica si el usuario está autorizado o no. Puede vincularse con los permisos del usuario.



Región de
Murcia

CIFP Carlos III

C/Carlos III, 3
30201 CARTAGENA
TEL: 968 321301
FAX: 968 320111
30019702@murciaeduca.es
www.cifpcarlos3.es



2. rules(): valida todos los campos definidos en el return, si alguno de esos campos no es válido indicará al sistema que no son válidos los datos.
3. messages(): buscará los campos que no son válidos en la función rules() y devolverá un mensaje al usuario de los campos inválidos.

Por ejemplo, supongamos que el usuario envía el formulario con el campo nombre vacío. En el momento de actualizar o crear el usuario la función rules() comprobará si todas las reglas impuestas son correctas. Si lo son, el request dejará continuar el proceso. En caso contrario, devolverá un mensaje de error al usuario, que en este caso es “El usuario necesita un nombre”.

```
public function store(SaveUserRequest $request)
{
    $user = new User($request->validated());
    $user->password = Hash::make($request->password);
    $user->save();

    return redirect()->route('usuarios.index')->with('status', 'El
usuario fue creado con éxito');
}
```

En el ejemplo anterior se hace uso del Request SaveUserRequest en el momento de validar los datos que introduce el usuario para crear un usuario nuevo en el sistema. Si son válidos los datos, se redirigirá al usuario a la vista usuarios.index.

2.2.1.7. Políticas (Policies)

Una política (policy) es un objeto que nos facilita Laravel para agrupar autorizaciones a nivel de cualquier modelo o recurso en particular.

¿Cómo se crea una política?

Para crear una política hay que hacer uso de Artisan. Por ejemplo para crear la política UsuarioPolicy, utilizaremos el comando siguiente:

```
php artisan make:policy UsuarioPolicy
```

Al crear una política se genera un fichero con el mismo nombre dentro de la carpeta app\Policies.

```
<?php

namespace App\Policies;
```



Región de
Murcia

CIFP Carlos III

C/Carlos III, 3
30201 CARTAGENA
TEL: 968 321301
FAX: 968 320111
30019702@murciaeduca.es
www.cifpcarlos3.es



ENSEÑANZAS COFINANCIADAS POR
FONDO SOCIAL EUROPEO (C.F.E.)

```
use App\User;
use Illuminate\Auth\Access\HandlesAuthorization;

class UsuarioPolicy
{
    use HandlesAuthorization;

    public function verPanelUsuarios(User $user) {
        return (bool) $user->rol->permisosRol->verPanelUsuarios;
    }

    public function modificarPanelUsuarios(User $user) {
        return (bool) $user->rol->permisosRol->modificarPanelUsuarios;
    }
}
```

En este ejemplo en cuestión, podemos ver que hemos creado dos funciones:

1. VerPanelUsuarios. Se hace uso del permiso que tiene el rol para identificar si el usuario tiene permiso para ver el panel de usuarios.
2. ModificarPanelUsuarios. Se hace uso del permiso que tiene el rol para identificar si el usuario tiene permiso para modificar el panel de usuarios.



Región de
Murcia

CIFP Carlos III

C/Carlos III, 3
30201 CARTAGENA
TEL: 968 321301
FAX: 968 320111
30019702@murciaeducas.es
www.cifpcarlos3.es



ENSEÑANZAS COFINANCIADAS POR
FONDO SOCIAL EUROPEO (C.F.E.)

¿Cómo se configura una política?

Estas políticas deben ser registradas dentro del fichero
app\Providers\AuthServiceProvider.php, donde tenemos que añadir la política y el objeto
que la utilizará para identificar si el usuario puede utilizar un recurso o no.

```
protected $policies = [  
    Proveedor::class => ProveedorPolicy::class,  
    Almacen::class => AlmacenPolicy::class,  
  
    PedidoCompra::class => PedidoPolicy::class,  
    Recepcion::class => RecepcionPolicy::class,  
    Stock::class => StockPolicy::class,  
    RegularizacionManual::class => StockPolicy::class,  
  
    // ClaseDesconocida::class => RecursoPolicy::class,  
    User::class => UsuarioPolicy::class,  
    Rol::class => UsuarioPolicy::class,  
  
    Producto::class => ProductoPolicy::class,  
    Marca::class => ProductoPolicy::class,  
    Impuesto::class => ProductoPolicy::class,  
    Familia::class => ProductoPolicy::class,  
    Subfamilia::class => ProductoPolicy::class,  
  
    Ubicacion::class => UbicacionPolicy::class,  
    Proceso::class => ProcesoPolicy::class,  
  
];
```

En esta imagen se pueden ver todas las políticas que se han incluido en el fichero
AuthServiceProvider.php.

¿Cómo se utiliza una política?

Una vez se ha añadido la política en AuthServiceProvider.php, podemos utilizarla en
cualquier sección del código como puerta para permitir o denegar el acceso a un recurso al
usuario.

```
public function destroy(User $usuario)  
{  
    $this->authorize('modificarPanelUsuarios', $usuario);  
}
```



Región de
Murcia

CIFP Carlos III

C/Carlos III, 3
30201 CARTAGENA
TEL: 968 321301
FAX: 968 320111
30019702@murciaeduca.es
www.cifpcarlos3.es



ENSEÑANZAS COFINANCIADAS POR
FONDO SOCIAL EUROPEO (C.F.E.)

```
//Si el usuario es el administrador, no se podrá modificar
if($usuario->codigo = "admin")
    return redirect()->route('usuarios.index')->with('status',
'El usuario no se puede eliminar. Es administrador');

$usuario->delete();
return redirect()->route('usuarios.index')->with('status', 'El
usuario fue eliminado con éxito');
}
```

Para utilizarla, necesitamos hacer uso de la función “authorize(<método-de-política>), <Modelo>”).

Cuando inyectamos el modelo en el método authorize, Laravel buscará la política utilizada en ese usuario y posteriormente buscará el método especificado.

Si la lógica aplicada dentro del método ‘modificarPanelUsuario’ devuelve que el usuario no tiene permisos, se lanzará un mensaje devolviendo que el usuario no tiene autorización para utilizar el recurso.

2.2.1.8. Controladores

Es uno de los tres componentes principales que utiliza Laravel, el cual utiliza e

¿Qué son los controladores?

Un controlador es un componente que hace de intermediario entre un modelo y una vista, tomando los datos del modelo, procesándolos y enviándolos a la vista para que los muestre. Dentro del controlador, se definirán las funciones que se deben llevar a cabo en la aplicación.

Por ejemplo, si se quiere mostrar todos los datos de un producto en el sistema, en el controlador Producto@controller se definirá el código que dará esta funcionalidad, tomando los datos necesarios del modelo y se los devuelve a la vista correspondiente, para que esta última los muestre.

¿Cómo funciona un controlador?

Cuando un usuario realiza una petición HTTP (HTTP request), el controlador recibe la petición. Posteriormente recoge los datos que tiene guardada la request. Más tarde, realiza un proceso que utiliza estos datos y ejecuta la tarea que ha solicitado el usuario. Recupera los datos del modelo y los procesa para cumplir su función. Finalmente, el controlador envía una respuesta HTTP (HTTP response) con los datos que ha pedido el usuario hacia la vista, para que puedan ser mostrados apropiadamente.



Región de
Murcia

CIFP Carlos III

C/Carlos III, 3
30201 CARTAGENA
TEL: 968 321301
FAX: 968 320111
30019702@murciaeduca.es
www.cifpcarlos3.es



ENSEÑANZAS COFINANCIADAS POR
FONDO SOCIAL EUROPEO (C.F.E.)

¿Cómo se generan los controladores?

Para crear un controlador, es necesario utilizar la consola de comandos. Para ello, se utilizará el siguiente comando:

```
php artisan make:controller ProductoController
```

Con este comando, generaremos el controlador `ProductoController`, que se alojará en la carpeta `app/http/controllers`.

Dentro de un controlador, se definirán las diferentes funciones que se van a realizar en las vistas asociadas a este controlador. Por ejemplo, para mostrar el perfil de un producto, se podría utilizar este controlador:

```
<?php

namespace App\Http\Controllers;
use App\Producto;
use App\Subfamilia;
class ProductoController extends Controller
{
    public function show(Producto $producto)
    {
        return view('articulos.productos.show', [
            'producto' => $producto
        ]);
    }
}
```

Como podemos comprobar, el método `show` recibe como parámetro el objeto `Producto` (mediante una petición HTTP de tipo GET). Lo que hará este método es recuperar los datos del modelo `Producto` y enviárselos a la vista `articulos.productos.show`, que mostrará los datos del producto en cuestión.

2.1.1.9. Vistas

Las vistas son otro de los componentes del patrón MVC. Representa la información a partir del modelo que envía el controlador. Se muestra a través de la interfaz del usuario.

Blade

Laravel incluye el motor de plantillas Blade. Este motor de plantillas facilita mucho el desarrollo de las vistas. La composición de las mismas están formadas a partir de código HTML, y se complementa con código JavaScript y CSS, con el que se facilita mucho la tarea de las vistas. Blade permite utilizar PHP y funciones como condicionales, bucles, etc.



Región de
Murcia

CIFP Carlos III

C/Carlos III, 3
30201 CARTAGENA
TEL: 968 321301
FAX: 968 320111
30019702@murciaeduca.es
www.cifpcarlos3.es



ENSEÑANZAS COFINANCIADAS POR
FONDO SOCIAL EUROPEO (C.F.E.)

Las plantillas de Blade utilizan la extensión “blade.php”, para poder diferenciarlo de un archivo PHP común.

Ejemplo:

```
@extends('layouts.principal')
@section('title', 'Home')
@section('content')
    <div class="container">
        <div class="row py-4">
            <div class="col-12 col-lg-6 shadow-sm py-4">
                <h1 class="display-6 text-primary ">Gestionalo </h1>
                <p class="lead text-secondary">
                    Gestiona tu almacén cómodamente</p>
                <a class="btn btn-lg
                    btn-block btn-outline-dark shadow-sm"
                    href="{{ route('articulos') }}"
                    >@lang('Articulos') </a>
                <a class="btn btn-lg btn-block
                    btn-outline-dark shadow-sm"
                    href="{{ route('logout') }}"
                    onclick="event.preventDefault();
document.getElementById('logout-form').submit();"
                    @lang('Logout') </a>
                <form id="logout-form"
                    action="{{ route('logout') }}"
                    method="POST" style="display: none;"
                    @csrf </form></div>
            <div class="col-12 col-lg-6 p-4">
                
            </div></div>
        </div>
    @endsection
```

Las vistas pueden ser invocadas a través de los controladores o las rutas. Para ello, se debe utilizar el método view(). Para el ejemplo de la figura anterior sería:

```
public function index(Request $request)
```



Región de
Murcia

CIFP Carlos III

C/Carlos III, 3
30201 CARTAGENA
TEL: 968 321301
FAX: 968 320111
30019702@murciaeduca.es
www.cifpcarlos3.es



ENSEÑANZAS COFINANCIADAS POR
FONDO SOCIAL EUROPEO (C.F.E.)

```
{  
  
    return view('home');  
  
}
```

Una de las ventajas que aporta Blade es la herencia entre plantillas. Con ello podemos generar una plantilla base y extender la misma plantilla con plantillas hijas.

Al llamar en una ruta a la plantilla hija que extiende de la plantilla padre, llamaremos a la combinación de ambas plantillas.

2.2.1.10 Sistema de archivos

Otro de los puntos fuertes de Laravel es poder gestionar su sistema de archivos. Laravel permite gestionar de manera simple los archivos.

El archivo de configuración del sistema de archivos está alojado en config/filesystem.php. En este archivo se configurarán los discos utilizados, la carpeta donde se almacenarán los archivos.

El disco público

Es el disco destinado a los archivos que son accesibles públicamente.

El disco público usa por defecto el directorio storage/app/public para guardar los datos.

Laravel nos permite hacerlo accesible desde la web al crear un enlace simbólico desde la carpeta storage/app/public hacia public/storage.

Para ello, es necesario utilizar el siguiente comando en la consola de comandos:

```
php artisan storage:link
```

Gestión de archivos

Las funciones del sistema de archivos se realizarán con la función Storage. A continuación, se explicarán las funcionalidades básicas para recuperar un archivo, subir un archivo y borrar un archivo:

1. Para subir un archivo, se utilizará el método put(). Ejemplo:
Storage::put('archivo.jpg', \$contenido);
 2. Para recuperar un archivo, el método get(). Ejemplo: \$archivo =
Storage::get('archivo.jpg');
 3. Para borrar un archivo, el método delete(). Ejemplo: Storage::delete('archivo.jpg');
- Storage cuenta con muchas más funcionalidades.

2.2.1.11. Sistema de traducción

Laravel cuenta con un sistema de traducción que permite crear una aplicación web en varios idiomas de manera sencilla y rápida. El idioma del sistema está en inglés por defecto.



Región de
Murcia

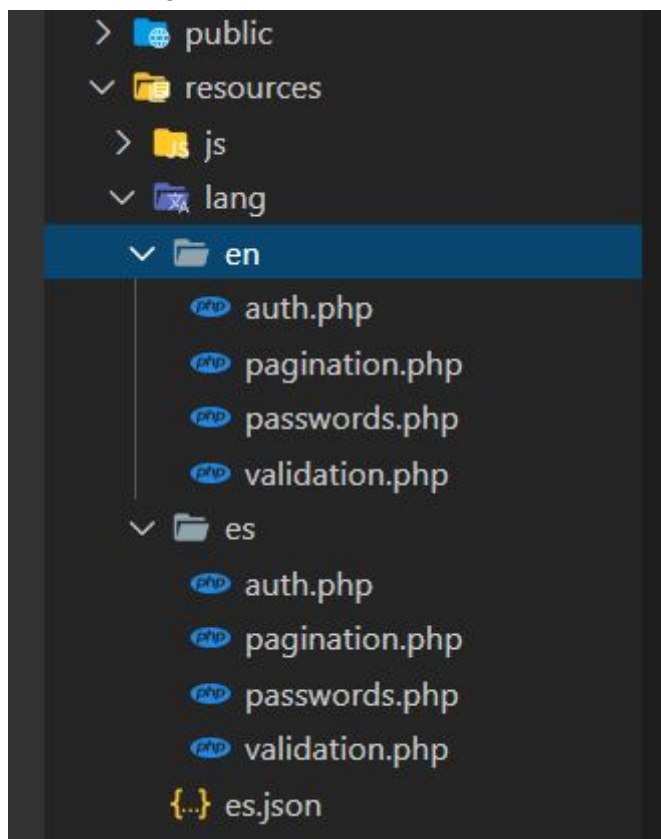
CIFP Carlos III

C/Carlos III, 3
30201 CARTAGENA
TEL: 968 321301
FAX: 968 320111
30019702@murciaeduca.es
www.cifpcarlos3.es



ENSEÑANZAS COFINANCIADAS POR
FONDO SOCIAL EUROPEO (C.F.E.)

Las traducciones de Laravel se guardan en el directorio resources/lang. Dentro de este directorio se debe definir un directorio por cada idioma. Si se va a definir como idioma español e inglés, debe quedar el directorio de este modo:



En cada directorio deben crearse los archivos donde se definirán las diferentes traducciones. Por ejemplo:



```
'accepted'      => ':attribute debe ser aceptado.',
'active_url'    => ':attribute no es una URL válida.',
'after'         => ':attribute debe ser una fecha posterior a :date.',
'after_or_equal' => ':attribute debe ser una fecha posterior o igual a :date.',
'alpha'         => ':attribute sólo debe contener letras.',
'alpha_dash'    => ':attribute sólo debe contener letras, números y guiones.',
'alpha_num'     => ':attribute sólo debe contener letras y números.',
'array'         => ':attribute debe ser un conjunto.',
'before'        => ':attribute debe ser una fecha anterior a :date.',
'before_or_equal' => ':attribute debe ser una fecha anterior o igual a :date.',
'between'       => [
    'numeric' => ':attribute tiene que estar entre :min - :max.',
    'file'    => ':attribute debe pesar entre :min - :max kilobytes.',
    'string'  => ':attribute tiene que tener entre :min - :max caracteres.',
    'array'   => ':attribute tiene que tener entre :min - :max ítems.',
],
'boolean'       => 'El campo :attribute debe tener un valor verdadero o falso.',
'confirmed'     => 'La confirmación de :attribute no coincide.',
'date'          => ':attribute no es una fecha válida.',
'date_equals'   => ':attribute debe ser una fecha igual a :date.',
'date_format'   => ':attribute no corresponde al formato :format.',
'different'     => ':attribute y :other deben ser diferentes.',
'digits'        => ':attribute debe tener :digits dígitos.',
'digits_between' => ':attribute debe tener entre :min y :max dígitos.',
'dimensions'    => 'Las dimensiones de la imagen :attribute no son válidas.',
'distinct'      => 'El campo :attribute contiene un valor duplicado.',
'email'         => ':attribute no es un correo válido',
```

```
"A fresh verification link has been sent to your email"
"All rights reserved.": "Todos los derechos reservados."
"Before proceeding, please check your email for a verification link."
"click here to request another": "haga clic aquí para solicitar otro",
"Confirm Password": "Confirmar contraseña",
"E-Mail Address": "Correo electrónico",
"Forbidden": "Prohibido",
"Forgot Your Password?": "¿Olvidó su contraseña?",
"Go Home": "Ir a inicio",
"Hello!": "¡Hola!",
```

Aplicar las traducciones en las vistas

Para utilizar las diferentes traducciones, utilizaremos @lang(). Por ejemplo:



Región de
Murcia

CIFP Carlos III

C/Carlos III, 3
30201 CARTAGENA
TEL: 968 321301
FAX: 968 320111
30019702@murciaeduca.es
www.cifpcarlos3.es



ENSEÑANZAS COFINANCIADAS POR
FONDO SOCIAL EUROPEO (C.F.E.)

```
<a class="btn btn-lg btn-block btn-outline-dark shadow-sm"
  href="{{ route('articulos') }}"
  >@lang('Articulos')
</a>

<a class="btn btn-lg btn-block btn-outline-dark shadow-sm" href="{{ route('logout') }}"
  onclick="event.preventDefault();
  document.getElementById('logout-form').submit();"
  >@lang('Logout')
</a>

form id="logout-form" action="{{ route('logout') }}" method="POST" style="display: none;"
  @csrf
form>
```




Región de
Murcia

CIFP Carlos III

C/Carlos III, 3
30201 CARTAGENA
TEL: 968 321301
FAX: 968 320111
30019702@murciaeduca.es
www.cifpcarlos3.es



ENSEÑANZAS COFINANCIADAS POR
FONDO SOCIAL EUROPEO (C.F.E.)

2.2.2. JQuery

Es una librería de JavaScript de software libre y código abierto. Fue creado originalmente por Josh Resig. JQuery simplifica la manera en la que se puede interactuar con documentos HTML y manipular el DOM.

Adicionalmente, simplifica otras funcionalidades de Javascript como manejo de eventos, o la implementación de técnicas AJAX en una página web.

2.2.3 Bootstrap

Es un conjunto de herramientas para el diseño de sitios y aplicaciones web. Contiene plantillas de diseño con distintos elementos de una web como pueden ser botones, menús textos, tipografías, etc, desarrollado en HTML y CSS. Adicionalmente, incluye algunas funciones de Javascript.

Bootstrap es utilizado únicamente para el desarrollo front-end. Bootstrap se denominaba inicialmente Twitter Blueprint. Fue creado por Mark Otto y Jacob Thornton, dos desarrolladores de Twitter.

2.3. Otras herramientas

2.3.1. Visual Studio Code (IDE)

Es un editor de código desarrollado por Microsoft para Windows, Linux y macOS. Incluye soporte para la depuración, control integrado de Git, resaltado de sintaxis, finalización inteligente de código, fragmentos y refactorización de código.

También es personalizable, gracias a ello, los usuarios pueden cambiar el tema del editor, atajos de teclado, y preferencias. Es gratuito y de código abierto.

Visual Studio Code se basa en Electron, un framework que se utiliza para implementar Chromium y Node.js como aplicaciones para escritorio.

Es compatible con varios lenguajes de programación. Muchas de las características de Visual Studio Code no están expuestas a través de los menús o la interfaz del usuario. Se puede extender a través de plugins y funcionalidades para facilitar el trabajo al desarrollador.

2.2.2. MySQL Workbench

Workbench es una herramienta gráfica para trabajar con MySQL.

Proporciona una interfaz muy sencilla de utilizar que integra desarrollo de programación SQL, administración de bases de datos y creación y diseño de bases de datos en un único entorno de desarrollo.

Fue escrito en 2002/2003 por el programador Michael G. Zinner



Región de
Murcia

CIFP Carlos III

C/Carlos III, 3
30201 CARTAGENA
TEL: 968 321301
FAX: 968 320111
30019702@murciaeducas.es
www.cifpcarlos3.es



ENSEÑANZAS COFINANCIADAS POR
FONDO SOCIAL EUROPEO (C.F.E.)

Es muy similar al SQL Server Management de Microsoft, con el que se utiliza para administrar SQL Server.

Contiene herramientas visuales con la que se puede crear muchas tareas, entre ellas:

- Ver y crear bases de datos.
- Ver el estado del servidor.
- Administrar usuarios.
- Configurar servidores.
- Crear y ejecutar sentencias SQL.
- Ver y crear otros objetos como vistas, triggers, procedimientos almacenados, etc.

2.3.3. Laragon

Laragon es una suite de desarrollo para PHP que funciona sobre Windows. Está diseñado especialmente para trabajar con Laravel.

Es similar a otras herramientas como Xampp o Wampp.

Laragon nos permite crear un entorno de desarrollo con estas características:

1. Cmder (Consola para Windows).
2. Git
3. Node.js
4. npm
5. SSH
6. Putty
7. PHP 7 / 5.6
8. Python
9. Extensiones de PHP
10. xDebug
11. Composer
12. Apache
13. MariaDB/MySQL/MongoDB/PostgreSQL
14. phpMyAdmin
15. Gestión automática de Virtualhosts.

Entre las funcionalidades de Laragon, hay una muy interesante que es crear automáticamente virtual hosts para cada proyecto, así por ejemplo, en lugar de ingresar desde el navegador a un proyecto con localhost/mi-proyecto, se puede utilizar una url más legible como miproyecto.test.

La versión utilizada de Laragon en este proyecto es la versión 4.0.



Región de
Murcia

CIFP Carlos III

C/Carlos III, 3
30201 CARTAGENA
TEL: 968 321301
FAX: 968 320111
30019702@murciaeduca.es
www.cifcarlos3.es



ENSEÑANZAS COFINANCIADAS POR
FONDO SOCIAL EUROPEO (C.F.E.)

2.3.4. MySQL

MySQL es un sistema de Gestión de Bases de Datos (SGBD), de código abierto. Es considerada la base de datos open-source más popular del mundo, especialmente para el desarrollo web.

2.3.5. Navegadores

Un navegador es un programa o aplicación que permite acceder a la web. También puede interpretar la información recibida desde distintas fuentes y visualizarlo correctamente.

Los recursos son identificados por medio de una URL (Uniform Resource Locator, en español “Localizador Uniforme de Recursos”) o bien, mediante una URI (Uniform Resource Identifier, en español “Identificador Uniforme de Recursos”).

Los recursos pueden representar distintos tipos de archivos: sonido, video, texto, imágenes, etc.

Los navegadores más populares del mercado son Mozilla Firefox, Opera, Safari, Internet Explorer/Microsoft Edge y Google Chrome. De entre estos navegadores, se han utilizado Mozilla Firefox, Microsoft Edge y Google Chrome para hacer pruebas.

2.3.6. Git

Git es un sistema de control de versiones para el seguimiento de cambios en los archivos de un proyecto y pensado para el mantenimiento de versiones en aplicaciones cuando estas tienen un gran volumen de archivos, además de utilizarse para coordinar el trabajo de un proyecto realizado por varias personas.

Fue diseñado por Linus Torvald.

2.3.7. Composer

Composer es un sistema de gestión de paquetes para programar PHP. Para utilizar Composer requiere tener instalado PHP.

Composer proporciona un estándar que permite administrar, descargar e instalar librerías y dependencias. También detecta declarar las librerías y paquetes de las que un proyecto depende y las gestiona (instala o actualiza) de forma automática.

Por defecto, trabaja e instala las librerías en un directorio local del proyecto denominado “vendor”, pero también se le puede indicar que instale una librería en formato global. De este modo se podría acceder a ella desde otros proyectos.

Composer utiliza Packagist para la búsqueda de las dependencias. Durante el desarrollo de la aplicación se han hecho uso de varios de ellos, lo que nos ha permitido dotar a nuestra aplicación de nuevas funcionalidades con muy poco esfuerzo.



3. Análisis Funcional

3.1. Documento de Requisitos

Identificador	Requisito
R1	El sistema debe contar con un diseño responsive que se adapte al dispositivo.
R1.1	Cuando la dimensión del dispositivo sea superior a 800px se visualizará una interfaz de escritorio web.
R1.1.	La barra de navegación aparecerá con todas las secciones, una tras otra.
R1.2	Cuando la dimensión del dispositivo sea inferior a 800px, se visualizará una interfaz de dispositivo móvil.
R1.2.1	La barra de navegación se ocultará y aparecerá un botón que permitirá desplegarla o volverla a ocultar.
R2	El sistema contará con un administrador que cuente con todos los privilegios.
R2.1	El administrador puede crear un usuario.
R2.2	El administrador puede editar un usuario, exceptuando el usuario de administrador.
R2.3	El administrador puede eliminar un usuario, exceptuando el usuario de administrador.
R2.4	El administrador puede crear un rol.
R2.5	El administrador puede editar un rol, exceptuando el rol de administrador.
R2.6	El administrador puede eliminar un rol, exceptuando el rol de administrador.
R2.7	El administrador puede crear usuarios administradores.
R2.8	El administrador puede asignar roles a los usuarios
R3.	El sistema contará con una página de login / identificación.
R3.1	Todos los usuarios, incluido el administrador, se identificarán con un código de usuario y una contraseña.



R3.2	Si el usuario ha olvidado la contraseña, verá una opción en el login para recuperar la contraseña a partir del correo electrónico. Recibirá un correo para reestablecer la contraseña.
R4	Todos los usuarios funcionarán por un sistema de roles.
R4.1	Cada rol tiene una serie de permisos.
R4.1.1	Cada sección tendrá un permiso para ver el recurso y otro para modificarlo.
R4.1.2	Si el usuario no tiene ningún permiso para ver el recurso, el sistema le devolverá un mensaje de que ha accedido a una sección no autorizada.
R4.1.2	Si el usuario sólo puede ver el recurso, no podrá modificarlo ni eliminarlo.
R4.1.3.	Para modificar un recurso, el usuario debe tener permiso para ver el recurso y además para modificarlo.
R.5	El sistema tendrá la sección Home. Se visualizará una breve descripción de la app y el logo de la app.
R.6	El sistema tendrá la sección “Artículos”.
R6.1	La sección artículos se dividirá en las subsecciones “Productos”, “Impuestos”, “Marcas”, “Familias” y “Subfamilias” y “Artículos”.
R7	La subsección “Productos” podrá ser visible por el usuario si tiene permisos para visualizar el recurso.
R7.1.	Si el usuario tiene permisos para visualizar productos, podrá ver un listado de productos.
R7.2.	Si el usuario tiene permisos para visualizar productos, podrá ver los datos completos de un producto.
R7.3.	Si el usuario tiene permisos de ver y modificar productos, podrá crear un producto.
R7.4.	Si el usuario tiene permisos de ver y modificar productos, podrá modificar los datos de un producto.
R7.5.	Si el usuario tiene permisos de ver y modificar productos, podrá eliminar un producto.
R7.6.	Si el usuario tiene permisos de visualizar productos, podrá filtrar los productos a partir de la marca, familia, subfamilia, impuesto o datos del propio producto.



R7.7.	Si el producto está siendo utilizado en algún pedido, recepción, documento de regularización o en resumen de stock, no podrá ser eliminado. Se avisará del motivo.
R8	La subsección “Impuestos” podrá ser visible por el usuario si tiene permisos para visualizar el recurso.
R8.1.	Si el usuario tiene permisos para visualizar productos, podrá ver un listado de impuestos.
R8.2.	Si el usuario tiene permisos para visualizar productos, podrá ver los datos completos de un impuesto.
R8.3.	Si el usuario tiene permisos de ver y modificar productos, podrá crear un impuesto.
R8.4.	Si el usuario tiene permisos de ver y modificar productos, podrá modificar los datos de un impuesto.
R8.5.	Si el usuario tiene permisos de ver y modificar productos, podrá eliminar un impuesto.
R8.6.	Si el usuario tiene permisos de visualizar productos, podrá filtrar los impuestos.
R8.7.	Si el impuesto está siendo utilizado en algún producto no podrá ser eliminado. Se avisará del motivo.
R9	La subsección “Marcas” podrá ser visible por el usuario si tiene permisos para visualizar el recurso.
R9.1.	Si el usuario tiene permisos para visualizar productos, podrá ver un listado de marcas.
R9.2.	Si el usuario tiene permisos para visualizar productos, podrá ver los datos completos de una marca.
R9.3.	Si el usuario tiene permisos de ver y modificar productos, podrá crear una marca.
R9.4.	Si el usuario tiene permisos de ver y modificar productos, podrá modificar los datos de una marca.
R9.5.	Si el usuario tiene permisos de ver y modificar productos, podrá eliminar una marca.
R9.6.	Si el usuario tiene permisos de visualizar productos, podrá filtrar las marcas.
R9.7.	Si el la marca está siendo utilizada en algún producto no podrá ser eliminada. Se avisará del motivo.



R10	La subsección “Familias” podrá ser visible por el usuario si tiene permisos para visualizar el recurso.
R10.1.	Si el usuario tiene permisos para visualizar productos, podrá ver un listado de familias de productos.
R10.2.	Si el usuario tiene permisos para visualizar productos, podrá ver los datos completos de una familia de producto.
R10.3.	Si el usuario tiene permisos de ver y modificar productos, podrá crear una familia de producto.
R10.4.	Si el usuario tiene permisos de ver y modificar productos, podrá modificar los datos de una familia de producto.
R10.5.	Si el usuario tiene permisos de ver y modificar productos, podrá eliminar una familia de producto.
R10.6.	Si el usuario tiene permisos de visualizar productos, podrá filtrar las familias.
R10.7.	Si la familia está siendo utilizada en algún producto no podrá ser eliminada. Se avisará del motivo.
R11	La subsección “Subfamilias” podrá ser visible por el usuario si tiene permisos para visualizar el recurso.
R11.1.	Si el usuario tiene permisos para visualizar productos, podrá ver un listado de subfamilias de productos.
R11.2.	Si el usuario tiene permisos para visualizar productos, podrá ver los datos completos de una subfamilia de producto.
R11.3.	Si el usuario tiene permisos de ver y modificar productos, podrá crear una subfamilia de producto.
R11.4.	Si el usuario tiene permisos de ver y modificar productos, podrá modificar los datos de una subfamilia de producto.
R11.5.	Si el usuario tiene permisos de ver y modificar productos, podrá eliminar una subfamilia de producto.
R11.6.	Si el usuario tiene permisos de visualizar productos, podrá filtrar las subfamilias.
R11.7.	Si la subfamilia está siendo utilizada en algún producto no podrá ser eliminada. Se avisará del motivo.
R12	La subsección “Articulos” podrá ser visible por el usuario si tiene permisos para visualizar el recurso.



R12.1	Mostrará a modo de lista todas las subsecciones que contiene la sección Artículos.
R13.	El sistema tendrá la sección “Ubicaciones”
R13.1.	Si el usuario tiene permisos para visualizar proveedores o almacenes, verá la sección “Ubicaciones”. También se visualizar una sección u otra en función del permiso que tenga el usuario. Si tiene ambos permisos, visualizará ambas subsecciones.
R14.2	La sección ubicaciones se dividirá en las subsecciones “Almacenes” y “Proveedores”.
R15	La subsección “Almacenes” podrá ser visible por el usuario si tiene permisos para visualizar el recurso.
R15.1.	Si el usuario tiene permisos para visualizar almacenes, podrá ver un listado de almacenes.
R15.2.	Si el usuario tiene permisos para visualizar almacenes, podrá ver los datos completos de un almacén.
R15.3.	Si el usuario tiene permisos de ver y modificar almacenes, podrá crear un almacén.
R15.4.	Si el usuario tiene permisos de ver y modificar almacenes, podrá modificar los datos de un almacén.
R15.5.	Si el usuario tiene permisos de ver y modificar almacenes, podrá eliminar un almacén.
R15.6.	Si el usuario tiene permisos de visualizar almacenes, podrá filtrar los almacenes.
R15.7.	Si el almacén está siendo utilizada en algún pedido, recepción, documento de regularización o resumen de stock, no podrá ser eliminado. .Se avisará del motivo.
R16	La subsección “Proveedores” podrá ser visible por el usuario si tiene permisos para visualizar el recurso.
R16.1.	Si el usuario tiene permisos para visualizar proveedores, podrá ver un listado de proveedores.
R16.2.	Si el usuario tiene permisos para visualizar proveedores, podrá ver los datos completos de un proveedor.
R16.3.	Si el usuario tiene permisos de ver y modificar proveedores, podrá crear un proveedor.
R16.4.	Si el usuario tiene permisos de ver y modificar proveedores, podrá



	modificar los datos de un proveedor.
R16.5.	Si el usuario tiene permisos de ver y modificar proveedores, podrá eliminar un proveedor.
R16.6.	Si el usuario tiene permisos de visualizar proveedores, podrá filtrar los proveedores.
R16.7.	Si el proveedor está siendo utilizada en algún pedido o recepción, no podrá ser eliminado. Se avisará del motivo.
R17.	El sistema tendrá la sección "Procesos".
R17.1.	Si el usuario tiene permisos para visualizar pedidos, recepciones o resumen de stock, verá la sección "Procesos". También se visualizará una sección u otra en función del permiso que tenga el usuario. Si tiene los tres permisos, visualizará todas las subsecciones.
R17.2	La sección procesos se dividirá en las subsecciones "Pedidos", "Recepciones", "Stock" y "Regularizaciones Manuales".
R18	La subsección "Pedidos" podrá ser visible por el usuario si tiene permisos para visualizar el recurso.
R18.1.	Si el usuario tiene permisos para visualizar pedidos, podrá ver un listado de pedidos.
R18.2.	Si el usuario tiene permisos para visualizar pedidos, podrá ver los datos completos de un pedido.
R18.3.	Si el usuario tiene permisos de ver y modificar pedidos, podrá crear un pedido.
R18.4.	Si el usuario tiene permisos de ver y modificar pedidos, podrá modificar los datos de un pedido.
R18.5.	Si el usuario tiene permisos de ver y modificar pedidos, podrá eliminar un pedido.
R18.6.	Si el usuario tiene permisos de visualizar pedidos, podrá filtrar los pedidos.
R18.7.	Un pedido estará formado por una cabecera y una o más líneas.
R18.8.	Cuando el usuario añada una línea en el pedido, podrá seleccionar un producto, la cantidad y un precio.
R18.9.	El importe se autocalculará en todo momento en función de la cantidad y el precio.



R18.10.	Cada vez que se seleccione un producto, se buscará el precio que tenga asociado el producto. Posteriormente volverá a autocalcularse el importe.
R18.11.	Si el pedido tiene alguna línea en estado servido o parcialmente servido, el pedido no podrá ser eliminada. Se avisará al usuario del motivo.
R18.12.	Si la cabecera del pedido está en estado servido, no podrá ser eliminado ni modificado. Se avisará al usuario del motivo.
R18.13.	Si el usuario está modificando el pedido y una de las líneas está en estado parcialmente servido o servido, la línea no podrá ser modificada ni eliminada.
R18.14.	Si el usuario está modificando el pedido y alguna de las líneas está en estado servido o parcialmente servido, se permitirá añadir nuevas líneas en estado Pendiente o modificar las existentes en estado Pendiente.
R18.15.	Si el usuario tiene permisos para visualizar pedidos, podrá visualizar la información del pedido en PDF.
R18.16.	Si el usuario está modificando un pedido, podrá modificar el estado del pedido desde Pendiente a Servido.
R18.17.	Si el usuario está modificando el pedido, se permitirá modificar el proveedor indicado en el pedido.
R18.18.	Si el usuario está modificando el pedido, se permitirá modificar la fecha indicada en el pedido.
R18.19.	Si el usuario está modificando el pedido, se permitirá modificar el almacén indicado en el pedido.
R18.20	Si el usuario está modificando el pedido, se permitirá modificar el campo de observaciones.
R19	La subsección "Recepciones" podrá ser visible por el usuario si tiene permisos para visualizar el recurso.
R19.1.	Si el usuario tiene permisos para visualizar recepciones, podrá ver un listado de recepciones.
R19.2.	Si el usuario tiene permisos para visualizar recepciones, podrá ver los datos completos de una recepción.
R19.3.	Si el usuario tiene permisos de ver y modificar recepciones, podrá crear una recepción.
R19.4.	Si el usuario tiene permisos de ver y modificar recepciones, podrá



	modificar los datos de una recepción.
R19.5.	Si el usuario tiene permisos de ver y modificar recepciones, podrá eliminar una recepción.
R19.6.	Si el usuario tiene permisos de visualizar recepciones, podrá filtrar las recepciones.
R19.7.	Una recepción estará formada por una cabecera y una o más líneas.
R19.8.	Cuando el usuario añada una línea en la recepción, podrá seleccionar un producto y la cantidad .
R19.9.	El usuario tendrá una opción donde podrá ver un listado de pedidos para capturar los productos pedidos. Esto vinculará la recepción con el pedido. Es decir, visualizará un listado de líneas de pedido en estado pendiente y podrá seleccionar las que considere válidas. Posteriormente se crearán las líneas que haya seleccionado y podrá modificar la cantidad por si realmente no se ha recibido la cantidad pedida. Que puede ser inferior, igual o superior a lo pedido. Posteriormente se vinculará la línea de recepción con la línea del pedido y se registrará como cantidad recibida en el pedido.
R19.10.	Si la recepción ha sido creada, y desea editarse, no podrá modificar el proveedor seleccionado, puesto que puede tener algún pedido vinculado.
R19.11.	Si se vincula una recepción con el pedido, el sistema modificará el estado de la línea del pedido a parcialmente servido si ha recibido algo o servido si la cantidad servida es igual o superior a la pedida. Posteriormente, el sistema verificará si el pedido vinculado tiene más líneas en estado Pendiente, de no ser así se modificará a estado Servido.
R19.12.	Si se borra alguna línea de recepción vinculado con el pedido, el sistema volverá a modificar el estado de las líneas del pedido que guarden relación con la línea de recepción a estado Pendiente. Nuevamente, el estado de la cabecera del pedido se modificará a estado Pendiente.
R19.13.	Si el usuario está modificando el pedido y alguna de las líneas está en estado servido o parcialmente servido, se permitirá añadir nuevas líneas en estado Pendiente o modificar las existentes en estado Pendiente.
R19.14.	Si el usuario tiene permisos para visualizar recepciones, podrá visualizar la información de la recepción en PDF.
R20	La subsección “Stock” podrá ser visible por el usuario si tiene permisos para visualizar el recurso.



Región de
Murcia

CIFP Carlos III

C/Carlos III, 3
30201 CARTAGENA
TEL: 968 321301
FAX: 968 320111
30019702@murciaeduca.es
www.cifpcarlos3.es



ENSEÑANZAS COFINANCIADAS POR
FONDO SOCIAL EUROPEO (C.F.E.)

R20.1.	Si el usuario tiene permisos para visualizar stock, podrá ver un listado de resumen de stock.
R20.2.	Si el usuario tiene permisos para visualizar stock, podrá ver los movimientos que trazados que tiene un almacén y un producto concreto.
R20.3.	Si el usuario tiene permisos para visualizar stock, podrá visualizar el informe de stock en PDF.
R21	La subsección “Regularizaciones manuales” podrá ser visible por el usuario si tiene permisos para visualizar el recurso.
R21.1.	Si el usuario tiene permisos para visualizar stock, podrá ver un listado de documentos de regularizaciones manuales realizadas.
R21.2.	Si el usuario tiene permisos para visualizar stock, podrá ver los datos completos de un documento de regularización.
R21.3.	Si el usuario tiene permisos de ver y modificar stock, podrá crear un documento de regularización.
R21.4.	Si el usuario tiene permisos de ver y modificar stock, podrá modificar los datos de un documento de regularización.
R21.5.	Si el usuario tiene permisos de ver y modificar stock, podrá eliminar un documento de regularización.
R21.6.	Si el usuario tiene permisos de visualizar stock, podrá filtrar los documentos de regularizaciones.
R21.7.	Un documento de regularización estará formada por una cabecera y una o más líneas.
R21.8.	Si el usuario ha creado un documento de regularización y desea eliminar una línea, el sistema no le permitirá hacerlo, deberá borrar el documento entero.
R21.9	Cuando se crea o modifica un documento de regularización, y se ha seleccionado un almacén y un producto, el sistema buscará qué cantidad hay de ese producto en stock y le sugerirá esa cantidad por defecto.
R21.10	Cuando se crea una línea de un documento de regularización manual, el usuario deberá poner la cantidad del producto que hay real en stock.
R21.11	Si el usuario tiene permisos para visualizar stock, podrá visualizar un listado de documentos de regularización.
R22	El stock del sistema funcionará por medio de movimientos de



	almacén.
R22.1	Cuando se cree una recepción, cada línea de la recepción será un movimiento de almacén de entrada.
R22.2	Cuando se cree un documento de regularización, el sistema detectará si el valor puesto por el usuario es inferior o negativo al que figura en stock en el sistema. En base a esto, creará un movimiento de entrada o salida para que la cantidad resultante sea la que el usuario ha introducido.
R22.3	Cada vez que se crea o borra un movimiento de almacén se recalcula la cantidad de producto que hay en stock en cada almacén.
R23	El sistema representará el recurso actual en el que se encuentra el usuario utilizando la técnica breadcrumb (migas de pan).
R24	El menú debe de estar en la parte superior de la página.
R25	Cada página del sistema debe tener un footer (pie de página) donde se mostrará el usuario registrado y el nombre de la web.
R26	Una vez el usuario esté logeado, el sistema contará con un recurso para deslogearse del sistema.
R27	El sistema tendrá una barra donde buscar en cada sección
R28	El color de fondo de la web será blanco grisáceo.
R29	El color del texto será negro.
R30	El color de los botones de eliminación será en fondo rojo y letras blancas.
R31	El color de fondo de los botones de impresión o de ver recurso será en azul.
R32	Los botones para regresar al paso anterior será en fondo negro y color de texto blanco.



Región de
Murcia

CIFP Carlos III

C/Carlos III, 3
30201 CARTAGENA
TEL: 968 321301
FAX: 968 320111
30019702@murciaeduca.es
www.cifpcarlos3.es



ENSEÑANZAS COFINANCIADAS POR
FONDO SOCIAL EUROPEO (C.F.E.)

3.2. Diseño de base de datos

3.2.1. Resumen

La aplicación Gestionalo utiliza una base de datos relacional para almacenar la información de todos los usuarios del sistema, productos y sus jerarquías, pedidos, recepciones, stock actual en el almacén, documentos de regularización, proveedores, almacenes... etc.

Cada usuario tendrá su propio rol con sus propios permisos para acceder a unos recursos u otros del sistema. Exceptuando el usuario administrador, que tendrá permiso para acceder a cualquier recurso, sin ninguna restricción.

Los productos se jerarquizarán por medio de las entidades familia y subfamilia.

Una familia estará formada por varias subfamilias, que a su vez estarán formados por varios productos.

Un producto estará formado a su vez por una marca, un impuesto y una subfamilia.

Un proveedor estará formado por un sujeto, que a su vez estará formado por una dirección.

Un almacén estará formado por una entidad de geolocalización y un sujeto, que a su vez estará formado por una dirección.

Un pedido de compra (cabecera) estará formado por un almacén, un proveedor, un estado (Servido, Anulado, Pendiente) y una o más líneas de pedido de compra.

Una línea de pedido de compra estará formado por un producto y un estado (Pendiente, Servido, Anulado, Parcialmente Servido).

Una recepción (cabecera) estará formada, similar al pedido de compra, por un almacén, un proveedor y una o más líneas de recepción.

La línea de recepción, a su vez, estará formada por un producto y una tabla intermedia que vinculará con la línea del pedido de compra. Es opcional, y crea una trazabilidad entre la recepción y el pedido de compra.

Un documento de regularización (cabecera) estará formado por un almacén y una o más líneas de regularización.

Cada Línea de regularización estará formada por un producto.

Cada movimiento de almacén estará formado por un producto, un almacén, un tipo de movimiento de almacén y un documento de origen.

Cada tipo de movimiento de almacén simbolizará la entrada o salida de un tipo de documento y estará formado por una clave de almacén, que a su vez puede ser entrada o salida (0,1).

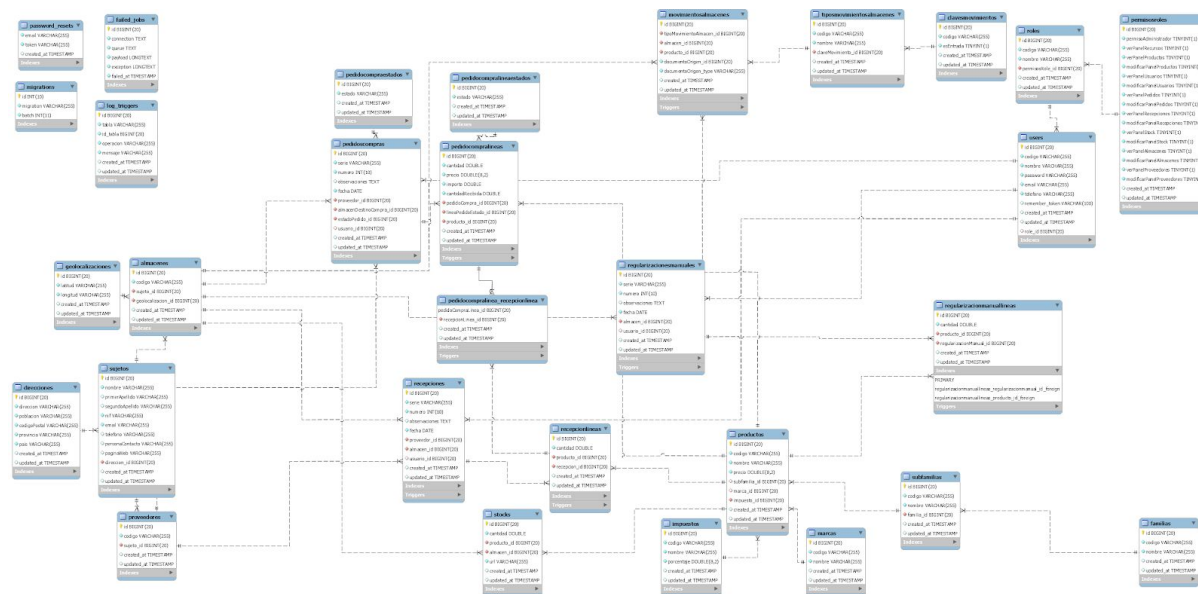
Cada documento origen podrá representar un documento u otro en función del tipo de documento origen. Como por ejemplo la recepción o el documento de regularización.

En base a los movimientos de almacén se compondrá el stock, que estará formado por un almacén y un producto.

Además de estas tablas habrán algunas que se utilizarán para funciones más específicas del sistema como son:

- password_resets. Cuando un usuario necesite restablecer la password porque no la recuerda, se mantendrá en esta tabla un registro con el email y un token de usuario.

- `log_triggers`. Se guardará un registro cuando un trigger sea disparado. Auditoría.
- `migrations`. Laravel necesita registrar las tablas que han sido migradas y en el paso en el que lo han hecho. Esto puede utilizarse para hacer rollback.
- `failed_jobs`. Laravel utiliza esta tabla para registrar ejecuciones fallidas.



Dada la extensión del modelo relación ER se adjuntará el modelo de la base de datos en el proyecto en formato .png, .svg y .mwb, donde se podrán visualizar fácilmente las relaciones entre las tablas, sus atributos y sus claves foráneas.

Para el desarrollo del proyecto hemos tenido que hacer uso de dos procedimientos “pedidocompralineas_recepcionlineas_ReestablecerEstados” y “pedidocompralineas_recepcionlineas_CambiarEstados”.

El usuario puede crear un documento de recepción y buscar los pedidos asociados al proveedor. Una vez visualiza dicho listado, puede determinar qué se ha recepcionado



Región de
Murcia

CIFP Carlos III

C/Carlos III, 3
30201 CARTAGENA
TEL: 968 321301
FAX: 968 320111
30019702@murciaeduca.es
www.cifpcarlos3.es



ENSEÑANZAS COFINANCIADAS POR
FONDO SOCIAL EUROPEO (C.F.E.)

realmente y en qué cantidad. El sistema creará una vinculación entre ambas entidades utilizando como tabla pivote la tabla “pedidocompralineas_recepcionlineas”.

Además, detectará si el pedido ha sido recibido completamente y cambiará el estado de las líneas del pedido y de la cabecera del pedido.

Si el usuario comete un error y borra la recepción, se borrará la vinculación con el pedido servido y se modificará su estado a “Pendiente”.

¿Qué es un procedimiento almacenado?

Un procedimiento almacenado se puede definir como un bloque de código guardado en base de datos que puede ser reutilizado de manera recursiva.

Una sentencia SQL, debe ser escrita en cualquier parte del código donde sea necesaria. En cambio, un procedimiento almacenado sólo necesita escribirse una sola vez y puede ser ejecutado desde cualquier parte del código. Es común utilizarlo junto a triggers.

Tiene varias ventajas:

- Se ejecuta en el motor de la base de datos, por lo que disminuye la sobrecarga de de datos entrantes y salientes y aumenta su velocidad de procesamiento.
- Simplifica la creación y mantenimiento de la lógica de negocio de la aplicación.
- Disminuye la probabilidad de que los datos se corrompan por el uso de programas cliente defectuosos.

3.2.3. Triggers (disparadores)

En adición a lo mencionado en el punto anterior, hemos hecho uso de triggers para mejorar la integridad de la base de datos.

Se han creado triggers en las siguientes tablas:

- movimientosalmacenes.
- pedidocompralineas_recepcionlineas.
- pedidocompralineas..
- recepciones.
- recepcionlineas.
- regularizacionmanual.
- regularizacionmanuallineas.

El principal motivo de su uso es para mejorar la integridad de los datos. Durante la fase de pruebas me di cuenta que si una tabla está configurada para que se elimine en cascada si la tabla con la que se referencia es borrada, los triggers no se disparan.

Por lo tanto, se tuvo que crear triggers que respetasen la lógica de negocio de la aplicación.

¿Qué es un trigger?



Región de
Murcia

CIFP Carlos III

C/Carlos III, 3
30201 CARTAGENA
TEL: 968 321301
FAX: 968 320111
30019702@murciaeducas.es
www.cifpcarlos3.es



ENSEÑANZAS COFINANCIADAS POR
FONDO SOCIAL EUROPEO (C.F.E.)

Un trigger se puede definir como una especie de procedimiento que se ejecuta automáticamente cuando se produce un evento en alguna de las tablas de la base de datos.

Estos triggers se desencadenan cuando se realiza alguna de las operaciones de INSERT, UPDATE o DELETE. El trigger se produce tanto antes como después de ejecutarse el evento.

Su principal utilidad es mejorar la gestión de la base de datos, implementando reglas de negocio para mejorar la integridad de la base de datos. Los trigger permiten sincronizar datos entre tables, modificar valores o incluso prevenir errores.

3.2.4. Active Record

Laravel utiliza el patrón de Active Record para la persistencia de datos.

¿Qué es Active Record?

Active Record es un enfoque para acceso de datos en una base de datos. Una tabla de la base de datos o vista (view) está envuelta en una clase.

Cuando se crea un objeto y se graba posteriormente, se crea un nuevo registro en la tabla. Cuando se carga el objeto, la información obtenida se carga a través de la base de datos. Igualmente, cuando es actualizado o eliminado, se actualiza o elimina la información en base de datos.

De este modo, Laravel cuenta con un ORM (Object-Relational mapping) llamado Eloquent para mapear las tablas y convertirlas en objetos.

Laravel también cuenta con un constructor de consultas SQL basado en programación orientada a objetos (PDO) llamado Fluent y además, permite utilizar SQL directo para realizar consultas más complicadas.



4. Arquitectura del sistema

4.1. Patrón MVC

La arquitectura Modelo-Vista-Controlador (MVC) es un patrón de arquitectura que está pensado para separar dentro de una aplicación los tres componentes principales, que son los datos (modelos), interfaces de usuario (vistas) y la lógica de negocio de sistema (controladores).

Componentes MVC

Modelo. El componente modelo corresponde a los datos con los que el usuario trabaja. Estos datos son transferidos entre los componentes Vista y Controladores.

Supongamos como ejemplo un objeto Producto el cual recuperará la información del producto desde la base de datos, posteriormente lo manipulará y actualizará y finalmente devolverá el dato a la base de datos o bien será utilizado para renderizarlo en una vista..

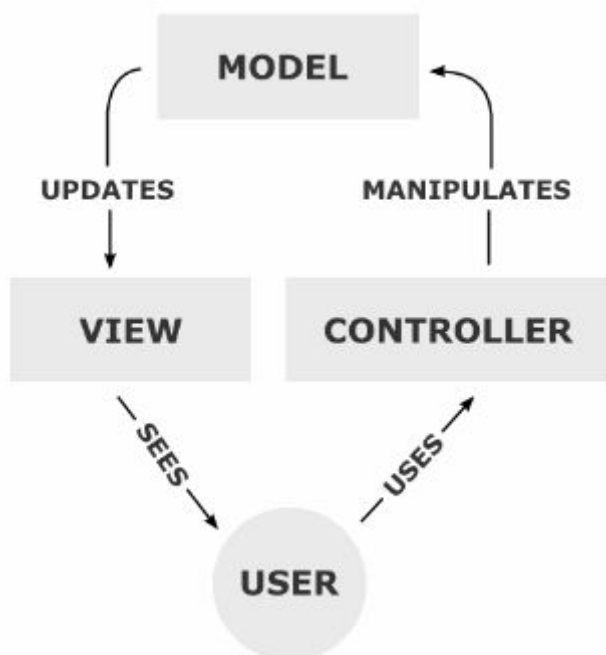
Vista. El componente Vista es utilizado por todas las interfaces de usuario de la aplicación. Por ejemplo, la vista de un producto incluirá todos los elementos con los que interactuará el usuario final (selectores, checkboxes, textbox.... etc).

Controlador. El componente Controlador funciona como una capa intermedia entre los componentes Vista y Modelo, el cual procesa toda la lógica de negocio y las solicitudes entrantes, manipula los datos haciendo uso del componente Modelo e interactúa con la vista para que renderice los datos finales que retorne.

Un ejemplo de ello sería un controlador de un producto el cual controlará todas las interacciones y entradas de datos desde la vista Producto y actualizará la base de datos utilizando el componente Producto.

Por lo tanto, la arquitectura MVC aporta múltiples ventajas, entre ellas:

- Aislar de manera clara la lógica de negocio entre componentes, facilitando el mantenimiento y la escalabilidad de la aplicación
- Permite representar los mismos datos de distintas maneras. Por ello podríamos utilizar distintas vistas.
- Código más legible.
- Pruebas aisladas.



Esquema MVC. Wikipedia

4.2. API REST

REST (Representational State Transfer), que significa “Transferencia de Representación de Estado”, también conocido como RESTful. Es un tipo de arquitectura de software que define un conjunto de pautas a la hora de crear servicios web.

Principios de la arquitectura REST

- **Cliente-servidor sin estado.** Cada mensaje por sí solo contiene la información necesaria para comprender la petición. El estado de la sesión debe mantenerse en la parte del cliente. Pese a ello, se pueden definir algunas respuestas a peticiones HTTP como cacheables, dando la posibilidad de ejecutar la misma respuesta ante peticiones iguales.
- **Un sistema REST debe** poder hacer uso de las operaciones más importantes y cuya especificación HTTP es: POST (crear), PUT (editar), DELETE (eliminar) y GET (leer, consultar). Dichas operaciones coinciden con las operaciones básicas de un sistema CRUD en una base de datos.
- **Los objetos deben ser manipulados siempre a partir de la URI** (Universal Resource Identifier). Cada URI identificará un recurso en concreto, dándonos la posibilidad de leer, modificar, crear o borrar ese recurso en concreto.
- **Interfaz uniforme.** La interfaz se basa en recursos. El servidor enviará al usuario los datos y con la representación del recurso que le llega al cliente, se podrá modificar/borrar el recurso. Esto se aplicará con acciones concretas como son POST, GET, PUT o PATCH.



- **Sistema compuesto por capas.** Cada una de las capas desempeña una funcionalidad dentro del sistema REST.
- **Uso de hipermedios.** La interfaz deberá proporcionar los medios necesarios para que el usuario pueda navegar entre los objetos, haciendo uso de enlaces.

Cualquier aplicación construida bajo API REST está obligada a hacer uso del principio HATEOAS (Hypermedia As The Engine Of Application State), traducido como “hipermedia como motor del estado de la aplicación”. Este principio significa que el usuario debe ser capaz de descubrir sus recursos basándose únicamente en las respuestas del servidor. Es decir, parte de la información enviada desde el servidor al cliente serán los hipervínculos (enlaces) de navegación que estarán asociados a otros recursos del cliente.

Un ejemplo de API REST sería el siguiente, para la URI

“(<http://gestionalo.ddns.net/stocks/1-2>)” se mostraría el siguiente recurso

```
"id" => 2
"cantidad" => 200.0
"producto_id" => 2
"almacen_id" => 1
"created_at" => "2020-05-01 15:05:45"
"updated_at" => "2020-05-01 18:35:45"
"url" => "1-2"
"codigo" => "CAJ0001001"
"sujeto_id" => 1
"geolocalizacion_id" => 1
"nombre" => "Caja de carton 49x56 Servo"
"primerApellido" => ""
"segundoApellido" => ""
"nif" => "1234123A"
"email" => "almacen@ejemplo.com"
"telefono" => "333666999"
"personaContacto" => "Roberto"
"paginaWeb" => ""
"direccion_id" => 1
"precio" => 0.3
"subfamilia_id" => 4
"marca_id" => 2
"impuesto_id" => 2
]
```



Región de
Murcia

CIFP Carlos III

C/Carlos III, 3
30201 CARTAGENA
TEL: 968 321301
FAX: 968 320111
30019702@murciaeduca.es
www.cifpcarlos3.es



ENSEÑANZAS COFINANCIADAS POR
FONDO SOCIAL EUROPEO (C.F.E.)

5. Interfaces de usuario

La web cuenta con un menú de 8 items, donde:

- 4 son secciones.
- 1 cuadro de búsqueda por recurso.
- Un enlace al menú inicio.
- Un menú de logín si el usuario está activo.
- La sección de Ver usuarios sólo será accesible por el administrador del sistema, y el resto de secciones serán operativas por los usuarios si tienen los permisos necesarios para poder visualizar o modificar el recurso.

El diseño de la web sigue el concepto “Responsive Design”, es decir, diseño adaptativo. Este diseño busca que la web se adapte al tamaño del dispositivo.

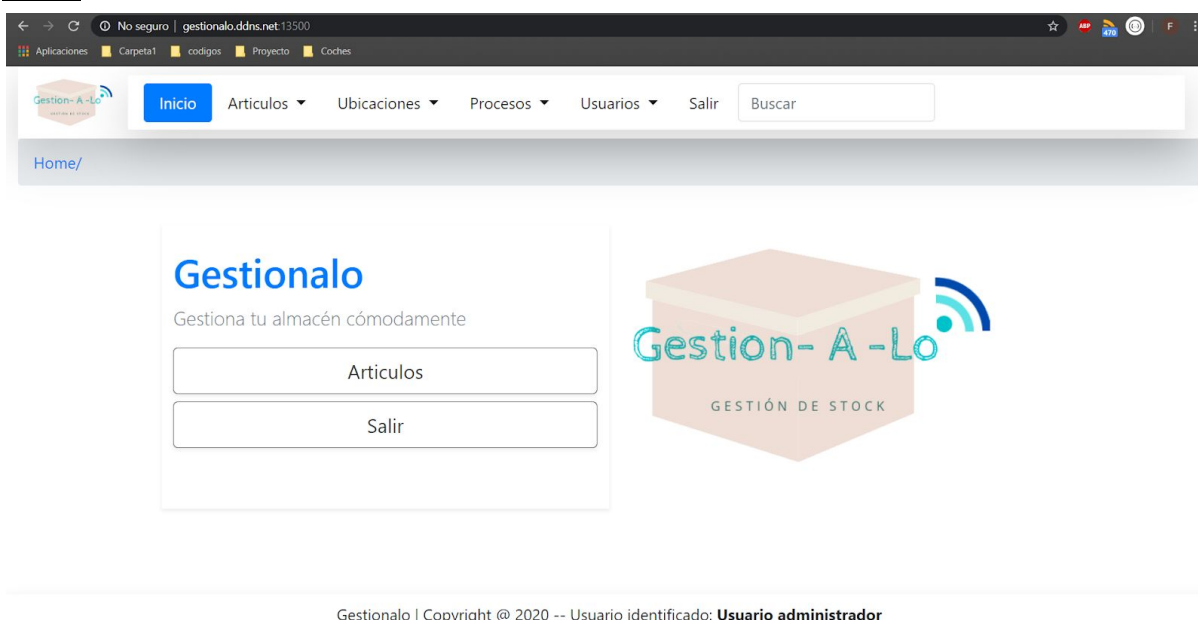
Todas las plantillas cuentan con una plantilla base que contiene una cabecera, donde podemos ver el logo de la página, el menú de navegación y el pie de página, donde puede visualizarse el usuario registrado. Todas las secciones tienen un fondo blanco, exceptuando la ventana de logín que contiene un fondo azul.

Como estamos utilizando recursos, todas las secciones siguen el patrón <nombre recurso>/<recurso>? y /edit o /create si se está modificando el recurso. Por ejemplo:

- /productos/create
- /productos/10
- /productos/10/edit

Se muestran algunas imágenes aleatorias de cómo se visualizan las interfaces de usuario.

Home





Región de
Murcia


CIFP Carlos III

C/Carlos III, 3
30201 CARTAGENA
TEL: 968 321301
FAX: 968 320111
30019702@murciaeducas.es
www.cifpcarlos3.es



ENSEÑANZAS COFINANCIADAS POR
FONDO SOCIAL EUROPEO (C.F.E.)

Artículos



[Inicio](#) [Artículos](#) [Ubicaciones](#) [Procesos](#) [Usuarios](#) [Salir](#)

Home/articulos

Artículos

Impuestos
Familias
Marcas
Productos
Subfamilias

Gestionalo | Copyright @ 2020 -- Usuario identificado: **Usuario administrador**


Productos

Home/productos

[Crear Producto](#)

#	Codigo	Nombre	Subfamilia	Marca	Operación
1	CAJ0001001	Caja de carton 49x56 Servo	Cajas de carton	Marca de Reposol	Ver Producto
2	BOL00001	Bolsa Elsa Pataki	Herramientas electricas	Marca Tania	Ver Producto
3	BOL00002	Bolsita de carton morado	Cajas de carton	Marca de Romero	Ver Producto

Almacenes



[Inicio](#) [Artículos](#) [Ubicaciones](#) [Procesos](#) [Usuarios](#) [Salir](#)

Home/almacenes / AD752472C

Codigo	Nombre	NIF	Email	Persona de Contacto	Página web	Creado
AD752472C	Super Mario S.L.	A75250667	taniananana@gmail.com	Tania	http://supermariobros.es	hace 1 mes

Direccion	Codigo Postal	Poblacion	Provincia	País
C\ Palomita, Cine 08	04700	Republica Independiente de la Loma	Almeria	Coronavirus

[Regresar](#) [Editar](#) [Eliminar](#)



Región de Murcia

CIFP Carlos III

C/Carlos III, 3
30201 CARTAGENA
TEL: 968 321301
FAX: 968 320111
30019702@murciaeduca.es
www.cifpcarlos3.es



ENSEÑANZAS COFINANCIADAS POR
FONDO SOCIAL EUROPEO (C.F.E.)

Pedidos

Gestion-A-Lo

Inicio Artículos Ubicaciones **Procesos** Usuarios Salir

Buscar

[Home/pedidos_compra](#)

Crear Pedido

#	Numero	Estado	Fecha	Proveedor	Lineas	Operación
1	PE/23	Pendiente	2020-07-09	Hermanos Electricistas SL	1	Imprimir Ver
2	PE/22	Servido	2020-06-01	Hermanos Electricistas SL	4	Imprimir Ver
3	PE/21	Servido	2020-05-31	Hermanos Electricistas SL	1	Imprimir Ver
4	PE/20	Servido	2020-06-05	Hermanos Electricistas SL	1	Imprimir Ver
5	SE/19	Servido	2020-05-29	Hermanos Electricistas SL	1	Imprimir Ver
6	SE/18	Servido	2020-05-29	Hermanos Electricistas SL	2	Imprimir Ver
7	SE/16	Servido	2020-05-15	Hermanos Electricistas SL	1	Imprimir Ver

Visualización de un pedido de compra

Home/pedidos_compra / 39

Serie	Numero	Fecha	Proveedor	Almacen	Estado	Usuario	Creado
PE	22	2020-06-01	Hermanos Electricistas SL	Almacen Cartagena	Servido	Usuario Pruebas Tania <3	hace 1 semana

Observaciones

#	Producto	Cantidad	Cant. Recibida	Precio	Importe	Estado
1	BOL00002 -- Bolsita de carton morado	100	50	0.59	59	ParcialmenteServido
2	CAJ0001001 -- Caja de carton 49x56 Servo	89	89	0.3	26.7	Servido
3	CAJ0001001 -- Caja de carton 49x56 Servo	100	100	0.3	30	Servido
4	CAJ0001001 -- Caja de carton 49x56 Servo	200	200	0.3	60	Servido

[Regresar](#) [Editar](#) [Eliminar](#)

Gestionalo | Copyright @ 2020 -- Usuario identificado: **Usuario administrador**



Región de
Murcia

CIFP Carlos III

C/Carlos III, 3
30201 CARTAGENA
TEL: 968 321301
FAX: 968 320111
30019702@murciaeduca.es
www.cifocarlos3.es



ENSEÑANZAS COFINANCIADAS POR
FONDO SOCIAL EUROPEO (C.F.E.)

Impresión de un pedido de compra en PDF

Pedido de Compra

Almacen

Código: 20000000
Nombre: Almacen Cartagena
NIF: 1234123A
Email: almacen@ejemplo.com
Teléfono: 333666999
Persona Contacto: Roberto
Dirección: C' Ejemplo Almacen, nº 1
CP: 30900
Población: Cartagena
Provincia: Murcia
País: España

Proveedor

Código: 40000000
Nombre: Hermanos Electricistas SL
NIF: 9887654A
Email: hermanos@prueba.com
Teléfono: 222999666
Persona Contacto: Sergio Egea Martinez
Dirección: C\ Ejemplo Proveedor, nº 1
CP: 30800
Población: Mazarron
Provincia: Murcia
País: España

Observaciones

#	Producto	Cantidad	Precio	Importe
1	BOL00002 -- Bolsita de carton morado	100	0.59	59
2	CAJ0001001 -- Caja de carton 49x56 Servo	89	0.3	26.7
3	CAJ0001001 -- Caja de carton 49x56 Servo	100	0.3	30
4	CAJ0001001 -- Caja de carton 49x56 Servo	200	0.3	60



Región de
Murcia

CIFP Carlos III

C/Carlos III, 3
30201 CARTAGENA
TEL: 968 321301
FAX: 968 320111
30019702@murciaeducas.es
www.cifpccarlos3.es



ENSEÑANZAS COFINANCIADAS POR
FONDO SOCIAL EUROPEO (C.F.E.)

Recepciones

Gestion- A-Lo

Inicio Artículos Ubicaciones Procesos Usuarios Salir Buscar

Home/recepciones / 9

Serie	Numero	Fecha	Proveedor	Almacen	Usuario	Creado
RE	7	2020-05-29	Hermanos Electricistas SL	Almacen Cartagena	Usuario administrador	hace 2 semanas

#	Producto	Cantidad
1	CAJ0001001 -- Caja de carton 49x56 Servo	3
2	CAJ0001001 -- Caja de carton 49x56 Servo	2
3	BOL00001 -- Bolsa Elsa Pataki	1
4	CAJ0001001 -- Caja de carton 49x56 Servo	3

Observaciones

Edición de una recepción

No seguro | gestionalo.ddns.net:13500/recepciones/9/edit

Aplicaciones Carpeta1 codigos Proyecto Coches

29/07/2020

Proveedor 4000000

Observaciones

Capturar Pedido

Pedido: PE/23. Fecha: 2020-07-09. Estado: Pendiente

☐ Producto: **Caja de carton 49x56 Servo**. Cantidad pend. recibir: **99 ud.**

Cerrar Guardar

Product

CAJ0001001 -- Caja de carton 49x56 Servo	3	Eliminar
CAJ0001001 -- Caja de carton 49x56 Servo	2	Eliminar
BOL00001 -- Bolsa Elsa Pataki	1	Eliminar
CAJ0001001 -- Caja de carton 49x56 Servo	3	Eliminar
CAJ0001001 -- Caia de carton 49x56 Servo	0	Eliminar

Gestionalo | Copyright © 2020 -- Usuario identificado: **Usuario administrador**



Región de
Murcia


CIFP Carlos III

C/Carlos III, 3
30201 CARTAGENA
TEL: 968 321301
FAX: 968 320111
30019702@murciaeduca.es
www.cifpcarlos3.es



ENSEÑANZAS COFINANCIADAS POR
FONDO SOCIAL EUROPEO (C.F.E.)

Stock

Inicio Artículos Ubicaciones **Procesos** Usuarios Salir

Home/stocks


Regularizar stock

Informe Inventario

#	Cantidad	Almacen	Producto	Operación
1	200	Almacen Cartagena	CAJ0001001 -- Caja de carton 49x56 Servo	Ver Movimientos
2	4601	Almacen Cartagena	BOL00001 -- Bolsa Elsa Pataki	Ver Movimientos
3	300	Almacen Cartagena	BOL00002 -- Bolsita de carton morado	Ver Movimientos
4	900	Hermanos Muñoz SL	CAJ0001001 -- Caja de carton 49x56 Servo	Ver Movimientos
5	9	Hermanos Muñoz SL	BOL00001 -- Bolsa Elsa Pataki	Ver Movimientos
6	100	Hermanos Muñoz SL	BOL00002 -- Bolsita de carton morado	Ver Movimientos
7	50	Super Mario S.L.	BOL00002 -- Bolsita de carton morado	Ver Movimientos

Gestionalo | Copyright @ 2020 -- Usuario identificado: **Usuario administrador**

Ver movimientos de stock

Inicio Artículos Ubicaciones **Procesos** Usuarios Salir

Home/stocks / 1-2

Almacen

Producto

20000000 -- Almacen Cartagena

CAJ0001001 -- Caja de carton 49x56 Servo

#	Codigo	Movimiento	Cantidad	Documento	Fecha	Proveedor
1	ENPED	Entrada Pedido	4	SE/18	2020-05-29	Hermanos Electricistas SL
2	ENREC	Entrada Recepcion	3	RE/7	2020-05-29	Hermanos Electricistas SL
3	SAREG	Salida Regularizacion	142	RG/1	2020-05-30	
4	ENREG	Entrada Regularizacion	201	RG/2	2020-05-30	
5	ENREG	Entrada Regularizacion	834	RG/2	2020-05-30	
6	ENREC	Entrada Recepcion	100	RE/9	2020-05-29	Hermanos Electricistas SL
7	ENREC	Entrada Recepcion	4	RE/10	2020-05-14	Hermanos Electricistas SL
8	ENREC	Entrada Recepcion	2	RE/10	2020-05-14	Hermanos Electricistas SL

Gestionalo | Copyright @ 2020 -- Usuario identificado: **Usuario administrador**



Región de
Murcia

CIFP Carlos III

C/Carlos III, 3
30201 CARTAGENA
TEL: 968 321301
FAX: 968 320111
30019702@murciaeduca.es
www.cifpccarlos3.es



ENSEÑANZAS COFINANCIADAS POR
FONDO SOCIAL EUROPEO (C.F.E.)

Crear documento de regularización

 Inicio Artículos ▾ Ubicaciones ▾ Procesos ▾ Usuarios ▾ Salir

[Home/regularizaciones_manual](#)

Stock		Crear Documento de Regularización				
#	Numero	Fecha	Almacen	Lineas	Operación	
1	RG/7	2020-06-11	Almacen Cartagena	2	Imprimir	Ver
2	RG/6	2020-06-12	Almacen Cartagena	7	Imprimir	Ver
3	RG/5	2020-06-24	Almacen Cartagena	2	Imprimir	Ver
4	RG/4	2020-05-31	Almacen Cartagena	1	Imprimir	Ver
5	RG/3	2020-05-15	Hermanos Muñoz SL	2	Imprimir	Ver
6	RG/2	2020-05-30	Almacen Cartagena	2	Imprimir	Ver
7	RG/1	2020-05-30	Almacen Cartagena	1	Imprimir	Ver



Región de
Murcia


CIFP Carlos III

C/Carlos III, 3
30201 CARTAGENA
TEL: 968 321301
FAX: 968 320111
30019702@murciaeducas.es
www.cifpccarlos3.es



ENSEÑANZAS COFINANCIADAS POR
FONDO SOCIAL EUROPEO (C.F.E.)

Creación de usuarios


Inicio Artículos ▾ Ubicaciones ▾ Procesos ▾ Usuarios ▾ Salir

Home/usuarios

[Roles disponibles](#) [Crear usuario](#)

#	Codigo	Nombre	Rol	Operación
1	ejemplo2	pruebas ejemplo2	Rol para probar el comando can	Ver Usuario
2	04600	Tany	administrador	Ver Usuario
3	pepe	pepe	administrador	Ver Usuario
4	ejemplo	Usuario ejemplo	Rol para probar el comando can	Ver Usuario
5	150120	Tany	Responsable del Dpto. de pedidos	Ver Usuario
6	prueba5	Usuario para productos2	Rol Productos	Ver Usuario

Visualización de un rol aleatorio

Inicio Artículos ▾ Ubicaciones ▾ Procesos ▾ Usuarios ▾ Salir

Home/roles / R. Pedidos

Codigo	Nombre	permisoAdministrador	VerP.Recursos	Creado
R. Pedidos	Responsable del Dpto. de pedidos	NO	NO	hace 1 semana

VerP.Productos	M.P.Productos	Ver.Usuarios	M.P.Usuarios	Ver.Pedidos	M.P.Pedidos
SI	SI	NO	NO	SI	SI

VerP.Recepciones	M.P.Recepciones	VerP.Stock	M.P.Stock
NO	NO	NO	NO

VerP.Almacenes	M.P.Almacenes	VerP.Proveedores	M.P.Proveedores
NO	NO	NO	NO

[Regresar](#) [Editar](#) [Eliminar](#)



Región de
Murcia


CIFP Carlos III

C/Carlos III, 3
30201 CARTAGENA
TEL: 968 321301
FAX: 968 320111
30019702@murciaeducas.es
www.cifpcarlos3.es



ENSEÑANZAS COFINANCIADAS POR
FONDO SOCIAL EUROPEO (C.F.E.)

Edición de un rol aleatorio

Inicio Artículos ▾ Ubicaciones ▾ Procesos ▾ **Usuarios ▾** Salir

Home/roles / R. Pedidos / edit

Editar rol

Código del rol

Nombre del rol

Permisos Administrador

☒

Ver Panel Recursos

☐

Ver Panel Productos

Gestionalo | Copyright © 2020 -- Usuario identificado: **Usuario administrador**



6. Despliegue del sistema web

Consiste en implementar en un servidor real el proyecto desarrollado en Laravel. Para ello he estado investigando hostings gratuitos que trabajen apropiadamente con Laravel. La mayoría de hostings que utilizan Laravel se implementan por medio de máquinas virtuales, pues son totalmente configurables.

Desafortunadamente no he encontrado ningún hostings que utilice composer, git y sea compatible con Laravel de manera gratuita.

Así pues, por falta de recursos he utilizado un pc local que actuará como servidor local, accesible desde Internet y vinculado a un dominio gratuito.

Para ello hemos de seguir una serie de pasos:

- Instalar una base de datos MySQL. Para ello, he tenido que instalar Laragon, una alternativa a XAMPP, el cual nos ofrece una serie de herramientas para desarrollar. En mi situación me ha bastado con poder utilizar su base de datos.
- Composer. Un sistema de gestión de paquetes para programar en PHP.
- PHP. Ha sido suficiente con descargarlo y añadirlo en la variable PHP.
- Aunque Laragon nos permite crear servidores virtuales de manera sencilla e inclusive crear proyectos base de Laravel, he preferido utilizar composer para crear un proyecto base de Laravel.
- Registrarse en my.noip.com y crear un dominio que apunte a la IP que nos haya asignado nuestro proveedor de servicios.
- Configurar el router para hacer un NAT para que a redirija una petición externa a un puerto determinado al servidor web alojado en el pc local.
- Creamos las tablas y le añadimos algún valor de ejemplo utilizando las migraciones de Laravel. Para ello, utilizamos el comando “php artisan migrate:fresh --seed”
- Posteriormente, desde php artisan, y habiendo abierto el proyecto de laravel, utilizar el comando “php artisan serve --host=192.168.0.4 --port=80”. Esto quiere decir lo siguiente:
 1. host: la ip del equipo del servidor web local
 2. puerto: el puerto que que abriremos en el servidor web para gestionar las peticiones HTTP.

En los anexos finales, se explicará paso a paso cómo instalar todo el software necesario para implementarlo.



7. Resultado final

Una vez desarrollado, configurado y desplegado, se puede ver la aplicación web en la URL:
<http://www.gestionaloya.ddns.net>.

Como también se ha trabajado con Github, tanto el proyecto de la aplicación web “Gestionalo” como la propia documentación se encuentra en:
https://github.com/FranciscoJoseVerduB/gestion_almacen/

8. Conclusiones

En este capítulo, se analizan las conclusiones finales del proyecto y se reflexiona sobre las posibles tareas de mantenimiento y extensión de la aplicación.

8.1. Conclusiones

Este trabajo ha consistido en el desarrollo de Gestionalo, una aplicación web para gestionar de manera sencilla almacenes. Las principales funcionalidades son las siguientes:

- Registro de usuarios gracias al módulo de usuarios.
- Registro y definición de productos, pudiendo agruparlos en jerarquías, o asignarles un precio o marca por defecto.
- Registro de almacenes y proveedores, que supondrán el pilar sobre el que registrar los documentos.
- Registro de pedidos de compra, gracias al módulo de compras.
- Registro de recepciones, donde se puede trazar lo que se ha pedido frente a lo que realmente ha llegado.
- Regularización del stock del almacén para establecer la cantidad de real que hay en cada almacén.
- Visualización del stock del almacén en tiempo real, en función de las operaciones con los distintos documentos registrados.
- Impresión de documentos y stock, gracias al módulo de informes.
- Sistema de login, gracias al cual se podrá trazar lo realizado por cada usuario. A modo de auditoría.

8.2. Trabajo futuro

Este sistema web conlleva unas expectativas bastante interesantes, ya que se ha diseñado como una base para incorporar nuevas funcionalidades, como el envío automático de email al proveedor cada vez que se realice un pedido donde se adjunte el documento en PDF, o bien gestionar de manera automática las tarifas de los precios por semana/mes por cada producto, proveedor y almacén.

También se ha preparado el modelo de datos para geoposicionar almacenes, lo cual facilitaría la gestión del almacén a la hora de realizar compras, ventas o movimientos entre almacenes.



Región de
Murcia

CIFP Carlos III

C/Carlos III, 3
30201 CARTAGENA
TEL: 968 321301
FAX: 968 320111
30019702@murciaeduca.es
www.cifocarlos3.es



Adicionalmente, también se ha replanteado implementar otros módulos como son el de ventas, que incluiría la venta del producto y su posterior facturación o el de cobros, que facilite la gestión de los cobros a los clientes a quien se les venda dicho producto o servicio.

La aplicación espera posibles futuras actualizaciones que mejorarán la seguridad del sistema web, creando un sitio más seguro y la utilización de tecnologías como Vue.js que faciliten la sincronía con el servidor realizando procesos en segundo plano, evitando abusar de las vistas para recargar la página.

Laravel cuenta por defecto con mecanismos que aportan seguridad al sistema, como son la protección frente a inyecciones SQL y protección frente a CSRF (Cross Site Request Forgery).

CSRF es un tipo de ataque en el cual los delincuentes se apoderan de una sesión autorizada por el usuario para realizar actos dañinos. Dicho proceso se lleva a cabo mediante peticiones HTTP.



Región de
Murcia

CIFP Carlos III

C/Carlos III, 3
30201 CARTAGENA
TEL: 968 321301
FAX: 968 320111
30019702@murciaeduca.es
www.cifocarlos3.es



ENSEÑANZAS COFINANCIADAS POR
FONDO SOCIAL EUROPEO (C.F.E.)

9. Bibliografía

<https://www.digitalocean.com/community/tutorials/como-instalar-en-ubuntu-18-04-la-pila-lamp-linux-apache-mysql-y-php-es>
<https://omarbarbosa.com/posts/los-mejores-paquetes-para-laravel>
<https://programacionymas.com/blog/como-enviar-mails-correos-desde-laravel>
<https://medium.com/@se.israel.mc/trabajar-en-proyecto-web-laravel-con-git-y-vagrant-2abb9c447e63>
<https://adminlte.io/>
<https://www.it-swarm.dev/es/php/como-instalar-font-awesome-en-laravel-mix/830654073/>
<https://fontawesome.com/icons>
<https://www.nigmacode.com/laravel/modificar-p%C3%A1ginas-de-error-Laravel>
https://www.youtube.com/watch?v=P4ehI6btNQ&list=PLpKWS6gp0jd_uZiWmjuqLY7LAMA_D8UJhc&index=8
<https://laragon.org/download/>
<https://dev.mysql.com/downloads/mysql/>
<https://www.youtube.com/playlist?list=PLhCiuvlix-rSgQNLII7Qg2KbQni3fz-ea>
<https://es.vuejs.org/v2/guide/>
<https://www.nicesnippets.com/blog/laravel-7-generate-pdf-from-view-example-tutorial>
<https://laravel.com/>
<https://www.it-swarm.dev/es/laravel/laravel-que-es-remember-token-en-la-tabla-de-base-de-datos-de-usuarios/1045869983/>
<https://fernando-gaitan.com.ar/laravel-parte-8-crud/>
<https://getbootstrap.com/>
<https://stackoverflow.com/questions/42342411/laravel-validation-check-array-size-min-and-max>
https://en.wikipedia.org/wiki/Cross-site_request_forgery
<https://github.com/barryvdh/laravel-dompdf>
https://es.wikipedia.org/wiki/Inyecci%C3%B3n_SQL
<https://desarrolloweb.com/manuales/manual-laravel-5.html>
<https://es.wikipedia.org/wiki/Laravel>
<https://www.arsys.es/blog/programacion/que-es-laravel/>
<https://getcomposer.org/>
<https://www.php.net/manual/es/intro-what-is.php>
<https://www.php.net/downloads>
<https://code.visualstudio.com/>
<https://stackoverflow.com/questions/46268211/how-to-format-laravel-blade-codes-in-visual-studio-code>
<https://styde.net/laragon-un-entorno-de-desarrollo-para-laravel-en-windows/>
https://en.wikipedia.org/wiki/Responsive_web_design
<https://bbvaopen4u.com/es/actualidad/api-rest-que-es-y-cuales-son-sus-ventajas-en-el-desarrollo-de-proyectos>
<https://styde.net/aprende-a-usar-eloquent-el-orm-de-laravel/>



**Región de
Murcia**

CIFP Carlos III

C/Carlos III, 3
30201 CARTAGENA
TEL: 968 321301
FAX: 968 320111
30019702@murciaeducas.es
www.cifocarlos3.es



ENSEÑANZAS COFINANCIADAS POR
FONDO SOCIAL EUROPEO (C.F.E.)

<https://www.noip.com/>

<https://sass-lang.com/>

<https://es.wikipedia.org/wiki/Node.js>

https://es.wikipedia.org/wiki/Inyecci%C3%B3n_de_dependencias

<https://styde.net/instalar-y-actualizar-paquetes-con-composer/>

<https://desarrolloweb.com/articulos/composer-gestor-dependencias-para-php.html>

<https://es.wikipedia.org/wiki/JQuery>



Región de
Murcia

CIFP Carlos III

C/Carlos III, 3
30201 CARTAGENA
TEL: 968 321301
FAX: 968 320111
30019702@murciaeduca.es
www.cifpcarlos3.es



ENSEÑANZAS COFINANCIADAS POR
FONDO SOCIAL EUROPEO (C.F.E.)

10.- Anexos

10.1.- Anexo A. Instalación Laragon

Para instalar Laragon, accedemos por medio del siguiente enlace:

<https://laragon.org/download/index.html>

Download

Laragon is a universal development environment. It has many features to make you more productive:

Benefits of Laragon

After downloading, You can add **git**, **phpmyadmin**, **Node.js/MongoDB**, **Python/Django/Flask/Postgres**, **Ruby**, **Java**, **Go** using “Tools > Quick add”

Note: [You can also download from GitHub](#)

Edition

- **Laragon Full:** Apache 2.4, Nginx, MySQL 5.7, PHP 7.2, Redis, Memcached, Node.js 11, npm, yarn, git, ...
[Download Laragon - Full \(130 MB\)](#)
- **Laragon Lite:** Don't include Node.js 11, npm, yarn, git but you can add them easily using “Tools > Quick add”
[Download Laragon - Lite \(85 MB\)](#)
- **Laragon Portable:** PHP 5.4, MySQL 5.1 - Good for getting started with PHP, then you can add newer versions of PHP/MySQL easily later
[Download Laragon - Portable \(18 MB\)](#)

Why Laragon?

Elegimos la opción Laragon Lite y lo descargamos. una vez descargado, lo instalamos e iniciamos el programa.



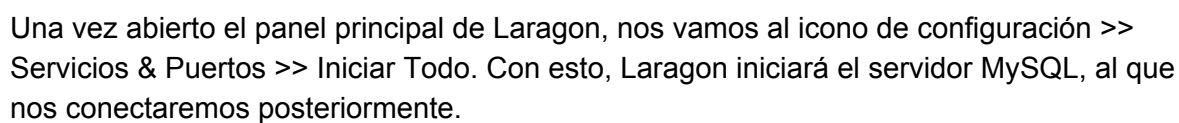
[Contents](#)

[Edition](#)

[Why Laragon?](#)

[Tutorials](#)

[Back to Top](#)





Región de
Murcia

CIFP Carlos III

C/Carlos III, 3
30201 CARTAGENA
TEL: 968 321301
FAX: 968 320111
30019702@murciaeduca.es
www.cifpcarlos3.es



Laragon Full 4.0.15 190704 php-7.2.19-Win32-VC15-x64 [TS] 169.254.120.92

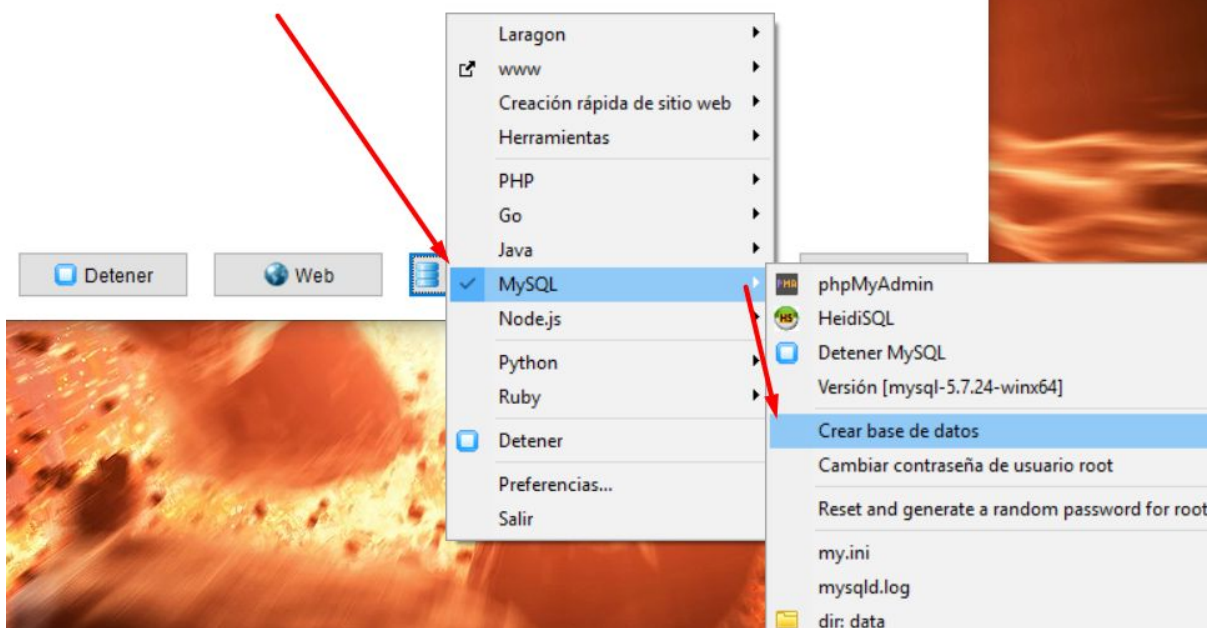


Menú

© Leo K

MySQL mysql-5.7.24-winx64 started

3306



Hacemos click derecho sobre la ventana principal de Laragon, nos desplegará la ventana secundaria, seleccionamos MySQL>> Crear base de datos.

Nos aparecerá una ventana para crear una base de datos. Le pondremos como ejemplo sistema_gestion. Por defecto, el usuario es root y la password nula.

Nos movemos al proyecto de Laravel en caso de que lo hayamos creado y nos movemos al fichero .env.

```
9
10 DB_CONNECTION=mysql
11 DB_HOST=127.0.0.1
12 DB_PORT=3306
13 DB_DATABASE=gestion_almacen
14 DB_USERNAME=root
15 DB_PASSWORD=
16
```

Aquí podemos configurar la conexión al servidor MySQL.



Región de
Murcia

CIFP Carlos III

C/Carlos III, 3
30201 CARTAGENA
TEL: 968 321301
FAX: 968 320111
30019702@murciaeduca.es
www.cifpcarlos3.es



ENSEÑANZAS COFINANCIADAS POR
FONDO SOCIAL EUROPEO (C.F.E.)

10.2.- Anexo C. Instalación PHP

Para instalar PHP, lo haremos desde la siguiente url:

<https://windows.php.net/download#php-7.4>

VC15 x64 Thread Safe (2020-Jun-09 17:07:54)

- [Zip](#) [24.93MB]

sha256: cb68159e37f93e9381c0b7821756e292218c27e5a149deef8b83e5898d1f0342

- [Debug Pack](#) [21.95MB]

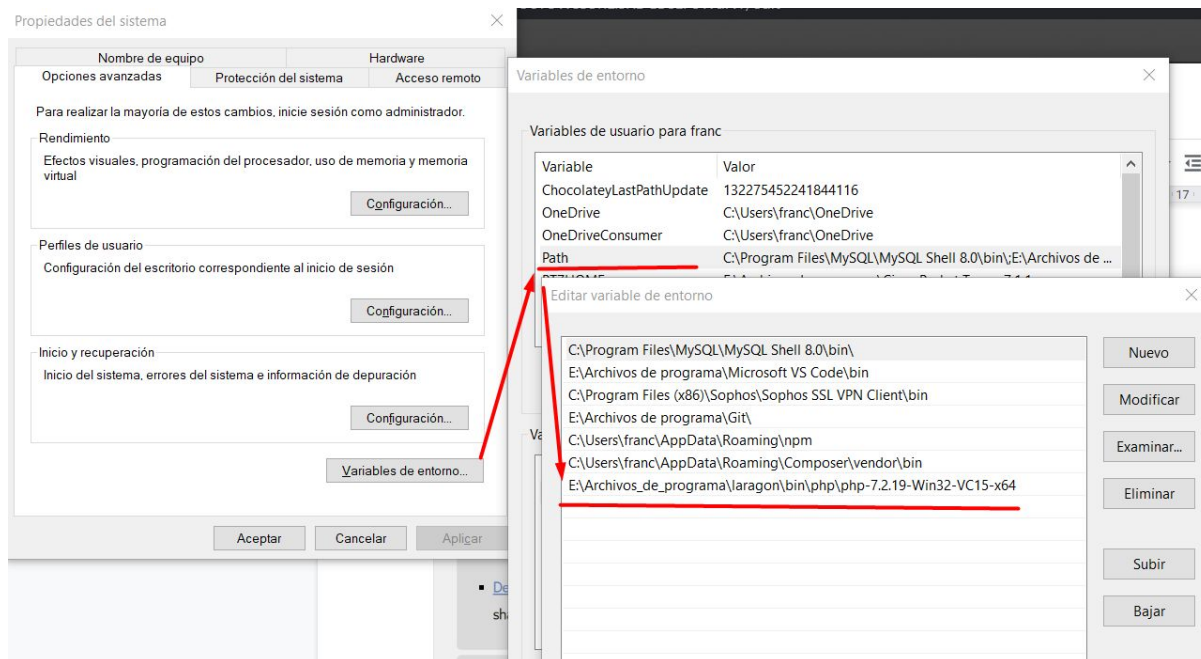
sha256: 3e0a3a977a30e8bdd0d0e8f6283b5e3181f91ef4407233e3e740b85dd1a58840

- [Development package \(SDK to develop PHP extensions\)](#) [1.08MB]

sha256: fe47b81426278539b305c9a67f8c9ea03b89b642ae55adcda46906b7753b1b2d

VC15 x86 Non Thread Safe (2020-Jun-09 17:07:43)

Nos descargamos la opción que ponga ZIP, posteriormente lo descomprimos en la ruta que queramos.



Finalmente añadiremos la ruta de la carpeta de PHP en la configuración de variables de entorno para poder utilizar PHP de manera cómoda.



Región de
Murcia

CIFP Carlos III

C/Carlos III, 3
30201 CARTAGENA
TEL: 968 321301
FAX: 968 320111
30019702@murciaeducas.es
www.cifpcarlos3.es



ENSEÑANZAS COFINANCIADAS POR
FONDO SOCIAL EUROPEO (C.F.E.)

10.3.- Anexo B. Instalación Composer

Para instalar Composer, lo haremos desde la siguiente url:

<https://getcomposer.org/download/>.

[Home](#) | [Getting Started](#) | [Download](#) | [Documentation](#) | [Browse Packages](#)

Download Composer Latest: v1.10.7

Windows Installer

The installer will download composer for you and set up your PATH environment variable so you can simply call `composer` from any directory.

Download and run [Composer-Setup.exe](#) - it will install the latest composer version whenever it is executed.

Command-line installation

Lo descargamos y en uno de los pasos nos pedirá seleccionar la ruta de php.exe.

Le asignamos la ruta donde se instaló PHP

Una vez instalado, utilizamos el comando `composer update`. Esto instalará la carpeta vendor en el directorio local al usuario o actualizará los paquetes en caso de tenerlo instalado previamente.

Posteriormente, creamos un proyecto de Laravel utilizando el comando siguiente comando:

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

Esto nos creará un proyecto de Laravel por defecto.

Utilizamos Visual Studio Code para abrir el proyecto desde la ruta creada y trabajaremos sobre ella.



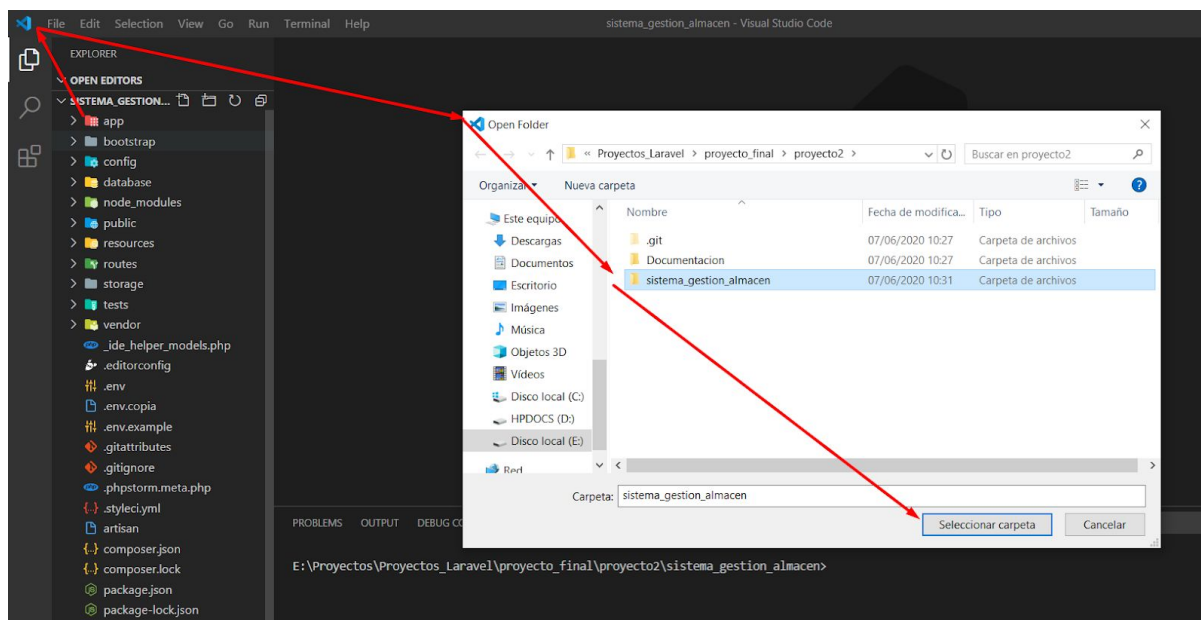
Región de
Murcia

CIFP Carlos III

C/Carlos III, 3
30201 CARTAGENA
TEL: 968 321301
FAX: 968 320111
30019702@murciaeduca.es
www.cifpcarlos3.es



ENSEÑANZAS COFINANCIADAS POR
FONDO SOCIAL EUROPEO (C.F.E.)

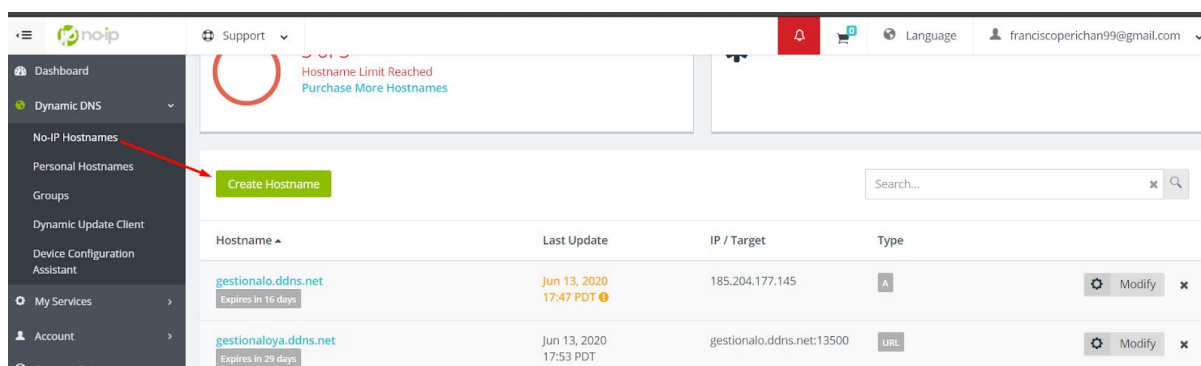


10.4.- Anexo D. Creación DDNS y configuración router

Un DDNS es un servidor de nombres de dominio dinámico. En términos simples es el nombre de las “páginas web” que buscamos en el navegador, como por ejemplo www.elpais.com. Esos nombres apuntan a la IP del servidor web donde se localiza el recurso.

Pasos a seguir

En primer lugar debemos registrarnos en la siguiente url: <https://my.noip.com/>



Posteriormente, creamos un hostname (dominio).



Región de
Murcia

CIFP Carlos III

C/Carlos III, 3
30201 CARTAGENA
TEL: 968 321301
FAX: 968 320111
30019702@murciaeducas.es
www.cifpcarlos3.es



Modify Hostname: gestionalo.ddns.net

IPv4 Address ⓘ

185.204.177.145

Last Update ⓘ

Jun 13, 2020
17:47 PDT

☐ Offline ⓘ **Upgrade to Enhanced** to enable offline settings.

MX Records

[+ Add MX Records](#)

Cancel

Update Hostname

El hostname deberá apuntar a la IP que nos habrá otorgado el proveedor de servicios que hayamos contratado para tener Internet. Por defecto, No-IP pone a modo de sugerencia la IP del equipo que se ha logeado (que será la que tengamos en nuestra WAN).

NAT – Servidores Virtuales

Host DMZ

Parámetros ALG

DDNS

Parámetros UPNP

Enrutamiento Estático

Tabla de Enrutamiento

NAT – Servidores Virtuales

La apertura de un rango de puertos es útil para servidores web, servidores FTP, servidores de correo, juegos, así como otras aplicaciones específicas de Internet. Cuando habilita la apertura de puertos, las peticiones de comunicación desde Internet hacia los puertos de la WAN del router serán reenviados a las direcciones IP especificadas de la LAN.

NO.	Puerto externo	Puerto interno	Dirección IP de la LAN	Protocolo	Habilitada	Borrar
5.	13500 – 14000	80	192.168.0.101	Ambc	<input checked="" type="checkbox"/>	<input type="checkbox"/>

El siguiente paso es logearnos en el router.

Accedemos al apartado que permita configurar el protocolo NAT (Network Address Translation), traducido al español “Traducción de Direcciones de Red”

¿Qué es NAT?

NAT es el protocolo que utilizan los routers para comunicarse paquetes que se transmiten entre distintas redes.

Ahora bien, el siguiente paso es indicarle en la configuración que si una petición viene desde el puerto que queramos (en mi caso es desde el 13500-14000), nos lo redirija al



Región de
Murcia

CIFP Carlos III

C/Carlos III, 3
30201 CARTAGENA
TEL: 968 321301
FAX: 968 320111
30019702@murciaeduca.es
www.cifpcarlos3.es



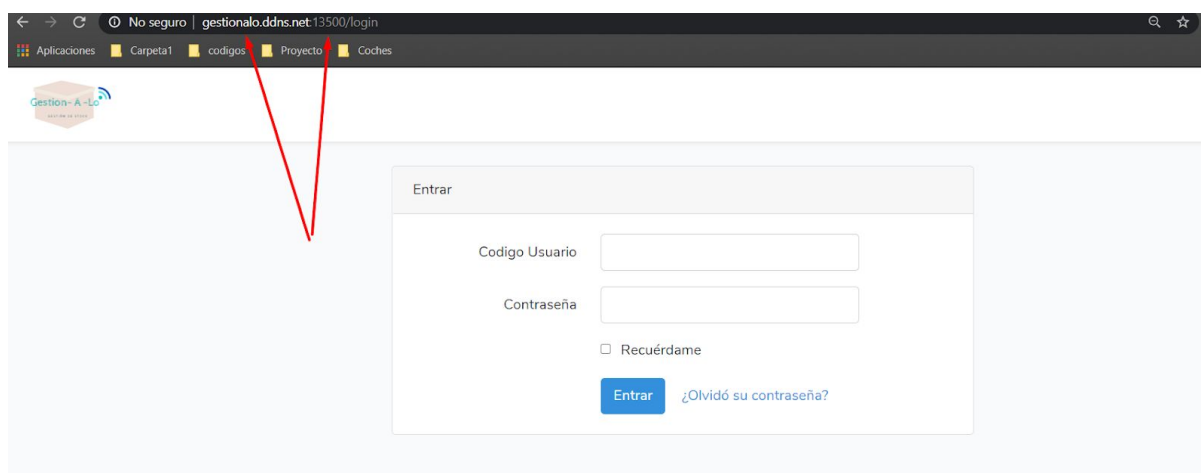
equipo que tenga la IP 192.168.0.101 y en el puerto 80 (puede ser otro también) dentro de la red local y que permita ambos protocolos.

El siguiente paso sería arrancar el servidor web de Laravel con esa IP y en el puerto 80

```
Laravel development server started: http://192.168.0.101:80
php artisan serve --host=192.168.0.101 --port=80
```

Utilizaremos el comando `php artisan serve --host=192.168.0.101 --port=80`

Donde el host es la IP del equipo donde está el servidor web y el puerto, el que queramos.



Una vez iniciado el servidor para que sea visible públicamente, configurado el router para que redirija la petición al equipo interno y configurado el DDNS, comprobamos que funciona correctamente.

```
Laravel development server started: http://192.168.0.101:80
[Sun Jun 14 06:05:23 2020] 192.168.0.1:54550 [200]: /js/app.js
[Sun Jun 14 06:05:23 2020] 192.168.0.1:54551 [200]: /css/app.css
[Sun Jun 14 06:05:23 2020] 192.168.0.1:54552 [200]: /img/icono-gestionalo.png
[Sun Jun 14 06:05:23 2020] 192.168.0.1:54553 [200]: /favicon.ico
php artisan serve --host=192.168.0.101 --port=80
```

Ahora mismo, el servidor web es accesible desde Internet. Cualquier usuario podría operar sobre él sin ningún problema.

10.5.- Anexo C. Establecer una cuenta de correo de gmail en Laravel

La gestión de emails para gestionar las notificaciones del servidor es algo necesario en todo servidor web. Para ello, vamos a utilizar una cuenta personal de gmail para hacer las pruebas.



Región de
Murcia

CIFP Carlos III

C/Carlos III, 3
30201 CARTAGENA
TEL: 968 321301
FAX: 968 320111
300197002@murciaeduca.es
www.cifocarlos3.es



Crear tu cuenta de Google

Ir a Gmail

Puedes usar letras, números y signos de puntuación



Usa 8 o más caracteres con una combinación de letras,
números y símbolos

[Acceder a tu cuenta en su lugar](#)

Siguiente



Una cuenta. Todos los servicios de
Google a tu disposición.

Una vez creada la cuenta, nos logeamos en ella



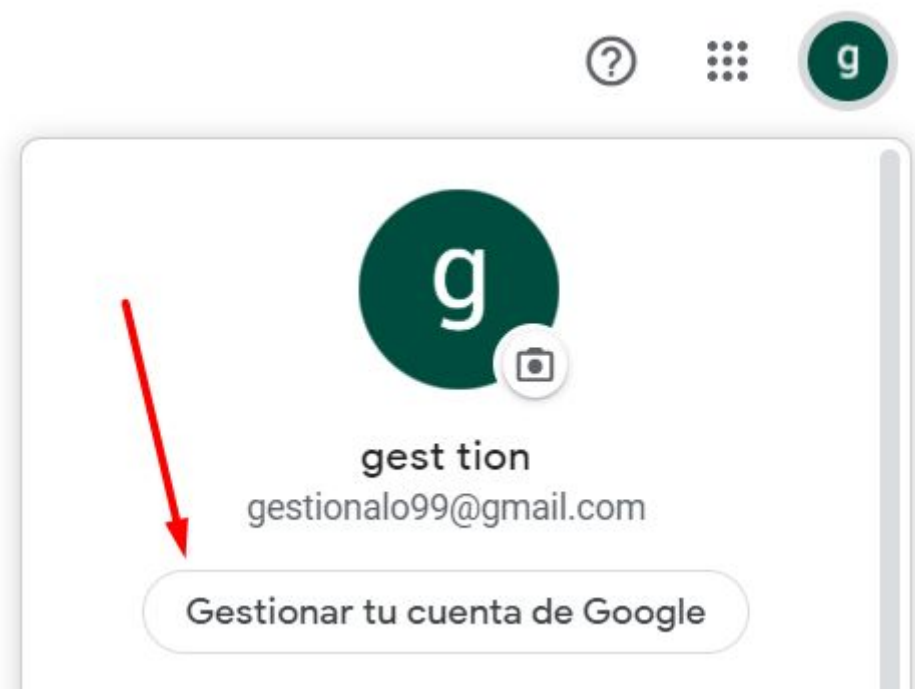
Región de
Murcia

CIFP Carlos III

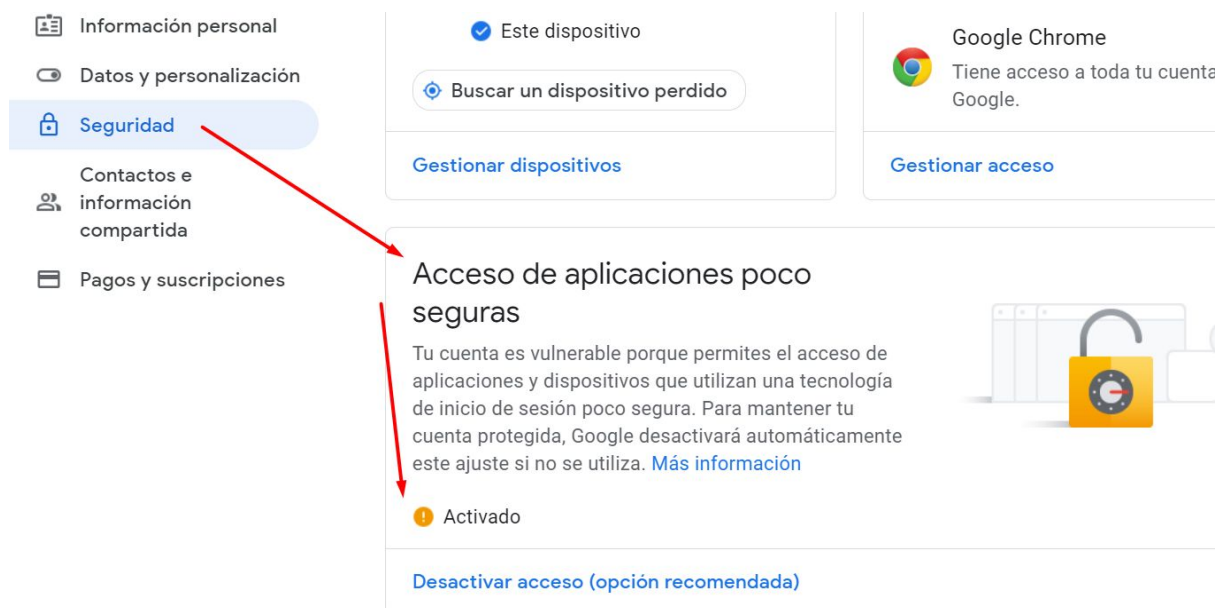
C/Carlos III, 3
30201 CARTAGENA
TEL: 968 321301
FAX: 968 320111
30019702@murciaeduca.es
www.cifpcarlos3.es



ENSEÑANZAS COFINANCIADAS POR
FONDO SOCIAL EUROPEO (C.F.E.)



Accedemos a la opción “Gestionar tu cuenta de Google”



Accedemos a Seguridad >> Acceso de aplicaciones poco seguras y activamos la opción.



Región de
Murcia

CIFP Carlos III

C/Carlos III, 3
30201 CARTAGENA
TEL: 968 321301
FAX: 968 320111
30019702@murciaeduca.es
www.cifpcarlos3.es



ENSEÑANZAS COFINANCIADAS POR
FONDO SOCIAL EUROPEO (C.F.E.)

```
MAIL_MAILER=smtp
MAIL_HOST=smtp.gmail.com
MAIL_PORT=465
MAIL_USERNAME=gestionalo99@gmail.com
MAIL_PASSWORD=password del gmail
MAIL_ENCRYPTION=ssl
MAIL_FROM_ADDRESS=gestionalo99@gmail.com
MAIL_FROM_NAME="${APP_NAME}"
```

Dentro del proyecto de Laravel, editamos el fichero .env y nos movemos a la opción donde se configura el MAIL, y establecemos una configuración similar.

En MAIL_USERNAME, ponemos el nombre del correo recién creado.

MAIL_PASSWORD, establecemos la password del correo recién creado.

MAIL_FROM_ADDRESS, el nombre del correo que enviará el email.

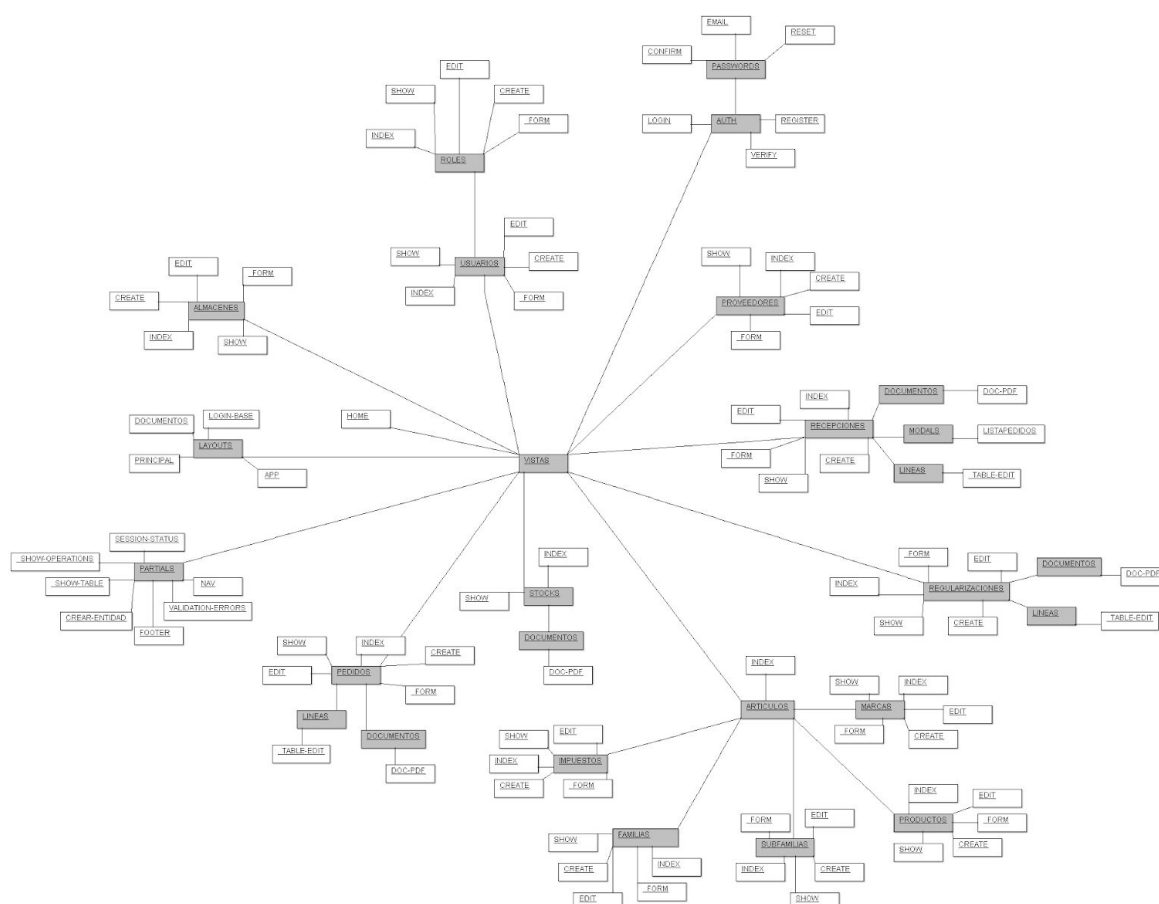
Desde ahora, todo correo que salga desde la app creada con Laravel, se enviará desde gestionalo99@gmail.com (el correo gmail que se creó desde 0, a modo de prueba).

10.6.- Anexo E. Manual de Usuario (Documento adjunto al proyecto)

Dada la extensión del manual, se adjunta un documento en PDF que contendrá el manual de usuario.

10.7.- Anexo F. Modelo de base de datos ER (Imagen y fichero de modelado .mwb adjunto al proyecto)

Dada la dimensión del modelo ER creado, se adjuntará el esquema relacional editable creado en MySQL Workbench y también en los formatos .PNG y .SVG para poder visualizarlo con buena calidad.





Región de
Murcia

CIFP Carlos III

C/Carlos III, 3
30201 CARTAGENA
TEL: 968 321301
FAX: 968 320111
30019702@murciaeduca.es
www.cifocarlos3.es

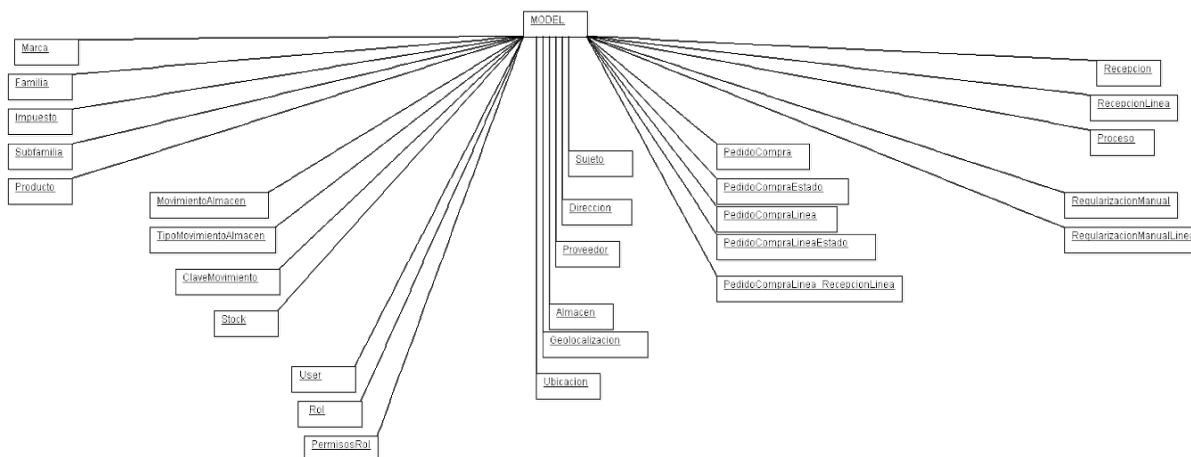


ENSEÑANZAS COFINANCIADAS POR
FONDO SOCIAL EUROPEO (C.F.E.)

10.9.- Anexo I. Jerarquía de modelos principales(Imagen y fichero .uml adjunto al proyecto)

Uno de los aspectos más importantes son los Modelos. Están situados en la carpeta principal app/. La clase base de la que heredan todos los modelos es la clase Model.

Se adjunta junto al proyecto un fichero .uml y una imagen en png más visible.

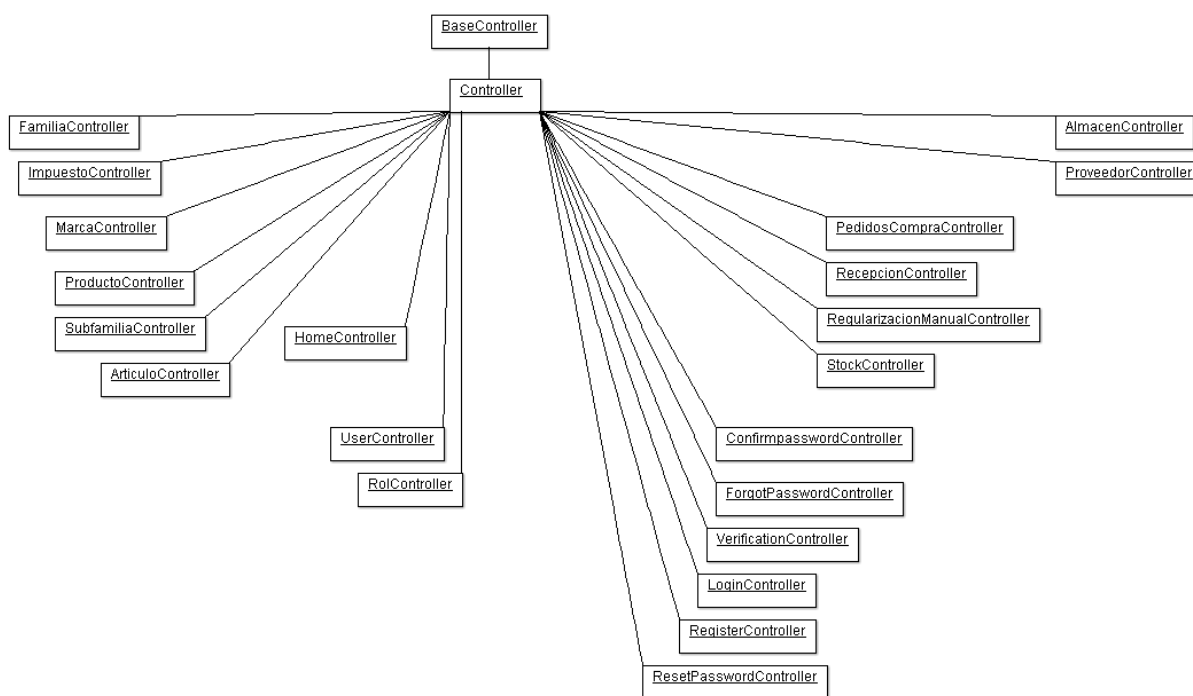




10.10.- Anexo J. Jerarquía de controladores creados (Imagen y fichero .uml adjunto al proyecto).

Otro de los aspectos vitales de un proyecto diseñado en Laravel son los controladores. Con ellos representamos toda la lógica de negocio de la aplicación. Todos los controladores heredan de la clase base Controller, que a su vez extiende de la clase BaseController.

Se adjunta junto al proyecto un fichero .uml y una imagen en png más visible.



10.11.- Anexo K. Manual de Usuario (Se adjunta en el proyecto un documento PDF que será el manual del usuario)

Un requisito indispensable a la hora de realizar un proyecto es crear un manual de usuario, que explique al usuario paso a paso cómo debe operar para utilizar correctamente la aplicación web.

Se adjunta un manual en .PDF junto al proyecto.