

# Manual de Usuario - Laravel

12/02/2020

---

Miguel Ángel López Sánchez

2º DAM  
DWESE

# Índice

<b>Repositorio GIT</b>	<b>3</b>
<b>Instalación</b>	<b>3</b>
OPCIONAL	5
<b>Creación del Nuevo Proyecto</b>	<b>6</b>
<b>Activar Servidor Laravel</b>	<b>6</b>
<b>Creación de alias a la carpeta public</b>	<b>7</b>
<b>Creación del modelo, migración, controlador, vistas</b>	<b>10</b>
<b>Configuración de la Base de Datos</b>	<b>11</b>
Crear database	11
Modificar .env	12
<b>Modificar campos de la tabla Articulos</b>	<b>13</b>
Migrar a la base de datos	14
<b>Generar Datos de Prueba</b>	<b>15</b>
Configuración del Seeder	15
Imágenes	17
Imágenes locales	17
Carpeta del storage	17
Actualizar la base de datos con los datos de la tabla	17
Comprobar que la tabla ha sido rellena	17
<b>Modificando las rutas (web.php)</b>	<b>18</b>
<b>Métodos del Controlador</b>	<b>18</b>
index	18
Scopes	19
Show	19
Create	20
Edit	21
Destroy	22
<b>Creación de plantillas</b>	<b>23</b>
<b>Creación de vistas</b>	<b>24</b>
index	24
Articulos/index	25
Articulos/show	27
Articulos/create	28
Articulos/edit	30

<b>Comprobamos que todo funcione en la web</b>	<b>32</b>
Listado	32
Paginación	32
Búsqueda	33
Detalles	33
Creación	33
Editado	34
Borrado	35
<b>Google Cloud</b>	<b>36</b>
Instalar PHP	36
Instalar Composer	36
Instalar git y clonar	37
Instalar Mysql Server y Configurar BD	37
Solución Rutas/Permisos de escritura	39
<b>Repositorio GIT</b>	<b>39</b>

# Repositorio GIT

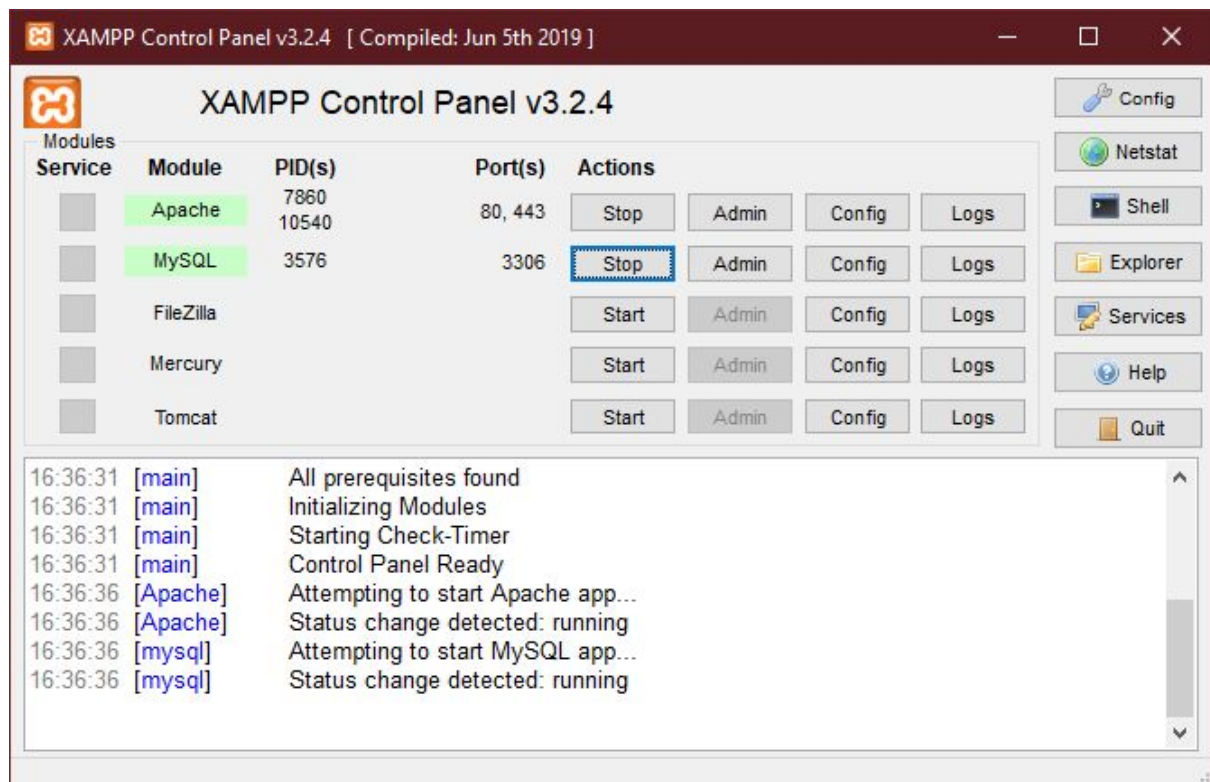
[https://github.com/Lobozel/Laravel\\_Tienda](https://github.com/Lobozel/Laravel_Tienda)

## Instalación

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

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

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



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



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

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

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

```
C:\Users\loboz>composer

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

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

## OPCIONAL

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

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

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

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

# Creación del Nuevo Proyecto

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

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

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

*No te preocupes si tarda en ejecutar el comando.*

```
D:\xampp\htdocs\DWSE\ejercicios\Taller-IV-Laravel\project>composer create-project laravel/laravel tienda
Installing laravel/laravel (v6.12.0)
- Installing laravel/laravel (v6.12.0): Loading from cache
Created project in tienda
> @php -r "file_exists('.env') || copy('.env.example', '.env');"
Loading composer repositories with package information
Updating dependencies (including require-dev)
Package operations: 85 installs, 0 updates, 0 removals
```

# Activar Servidor Laravel

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

```
php artisan serve
```

```
D:\xampp\htdocs\DWSE\ejercicios\Taller-IV-Laravel\project>cd tienda
D:\xampp\htdocs\DWSE\ejercicios\Taller-IV-Laravel\project\tienda>php artisan serve
Laravel development server started: http://127.0.0.1:8000
```

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



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



## Creación de alias a la carpeta public

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

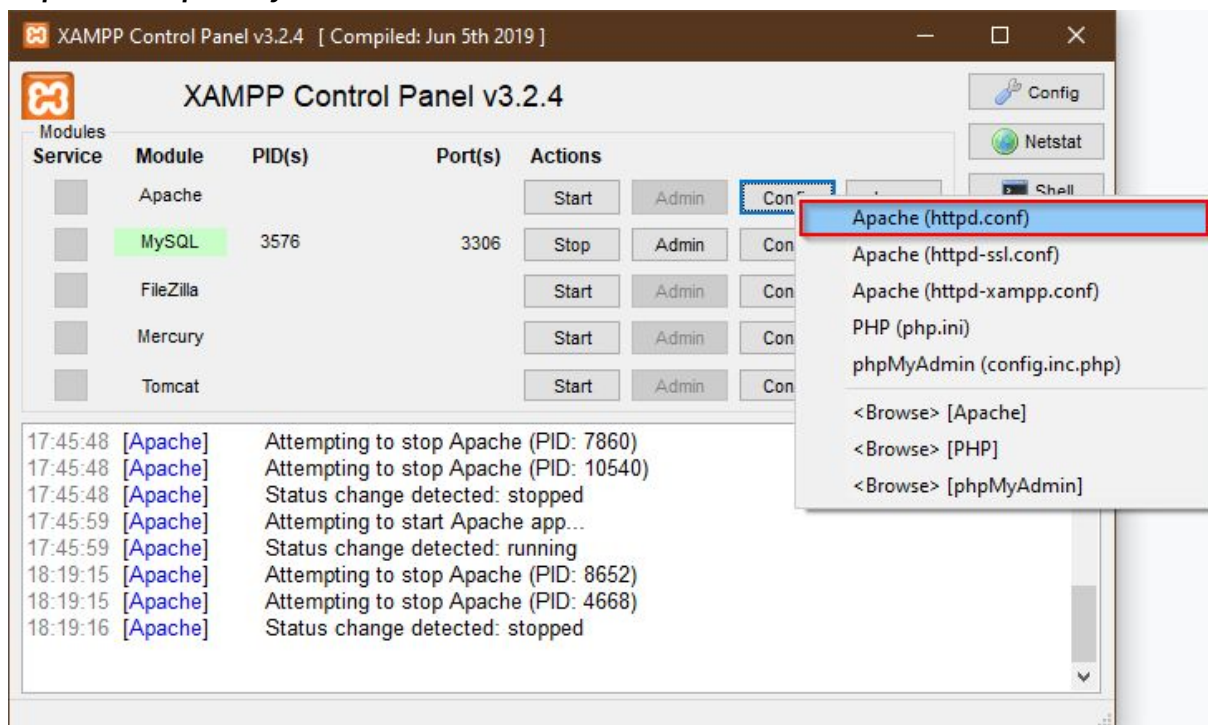
`http://localhost/DWESE/ejercicios/Taller IV Laravel/project/tienda/public/`

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

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

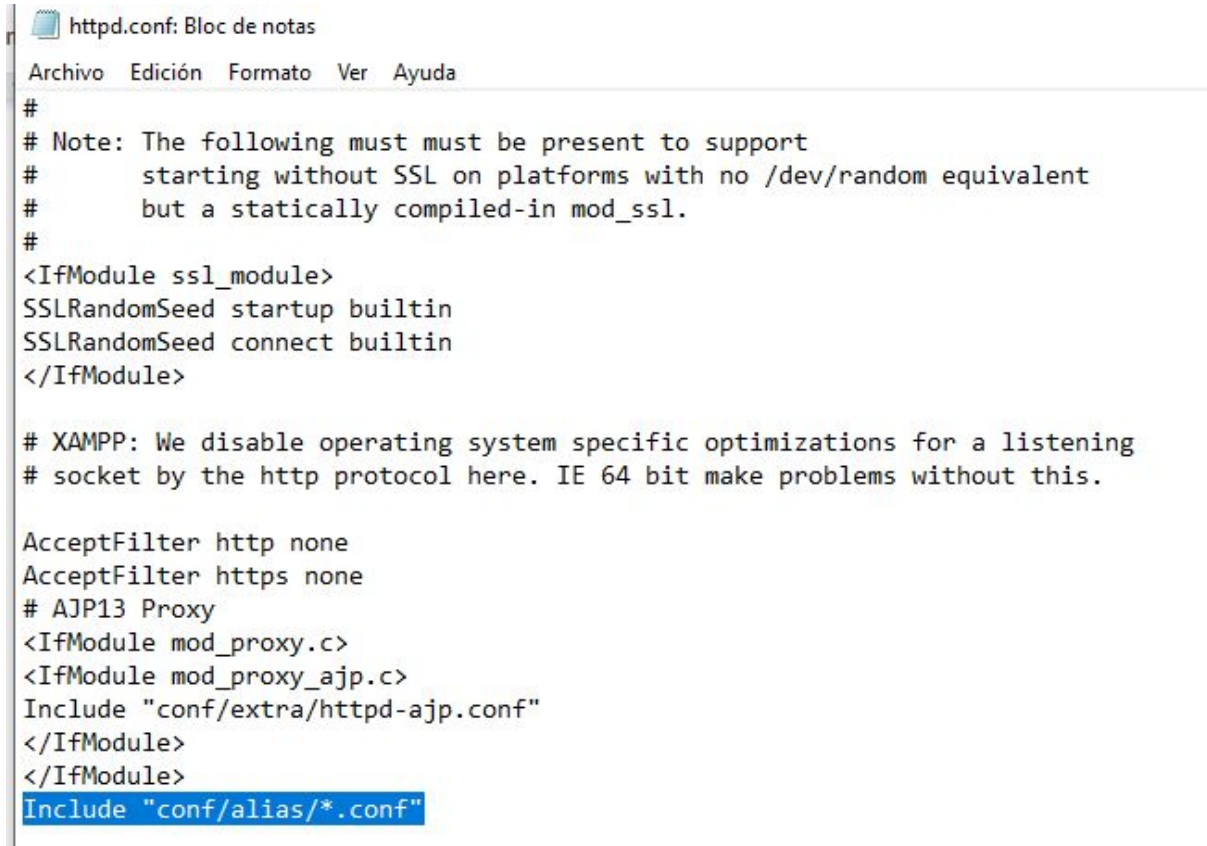
**Detén el servidor Apache** mientras haces las siguientes configuraciones para el alias.

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





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



```

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

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

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

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

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

Después **añádele** el siguiente texto:

```

<Directory "D:\xampp\htdocs\DWES\ejercicios\Taller-IV-Laravel\project\tienda\public">
    DirectoryIndex index.php
    AcceptPathInfo on
    AllowOverride None
    Options None
    Order allow,deny
    Allow from all

    Options +FollowSymLinks
    RewriteEngine on
    RewriteRule . index.php [L]
    RewriteBase /tienda

</Directory>
  
```

```
alias /tienda "D:\xampp\htdocs\DWESE\ejercicios\Taller-IV-Laravel\project\tienda\public"
```

Cambia el texto en rojo, `/ruta_url` (en mi caso `/tienda`) será el **alias** que le daremos a la **ruta\_local** (la parte entre comillas "") que será la ruta donde se encuentra la **carpeta public** de nuestro proyecto.

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



# Creación del modelo, migración, controlador, vistas

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

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

o el siguiente comando:

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

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

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

Además, el `-a` te creará el factory, el cual no te hará falta para esta práctica.

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

**-m** → Crea la **migración**

**-c** → Crea el **controlador**

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

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

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

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

```
D:\xampp\htdocs\DWEE\ejercicios\Taller-IV-Laravel\project\tienda>php artisan make:model Artículo -mcrs
Model created successfully.
Created Migration: 2020_02_13_030056_create_articulos_table
Seeder created successfully.
Controller created successfully.
```

# Configuración de la Base de Datos

## *Crear database*

En primer lugar tienes que acceder a mysql.

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

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

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

MariaDB [(none)]>
```

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

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

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

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

```
MariaDB [tiendadb]> grant all privileges on tiendadb.* to userpdo@'localhost';
Query OK, 0 rows affected (0.029 sec)

MariaDB [tiendadb]> _
```

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

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

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

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

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

## *Modificar .env*

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

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

*Esta casi al principio.*

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

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

# Modificar campos de la tabla Artículos

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

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

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

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

```
Schema::create('articulos', function (Blueprint $table) {  
    $table->bigIncrements('id');  
    $table->string('nombre');  
    $table->string('categoria',12);  
    $table->decimal('precio',10,2);  
    $table->integer('stock')->default(0);  
  
    $table->string('imagen')->default('/img/articulos/default.png');  
    $table->timestamps();  
});
```

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

```
protected $fillable=["nombre", "categoria", "precio", "stock",  
"imagen"];
```

*Esto te permitirá añadir datos por formulario.*



## *Migrar a la base de datos*

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

```
php artisan migrate:fresh
```

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

```
D:\xampp\htdocs\DWSE\ejercicios\Taller-IV-Laravel\project\tienda>php artisan migrate:fresh
Dropped all tables successfully.
Migration table created successfully.
Migrating: 2014_10_12_000000_create_users_table
Migrated: 2014_10_12_000000_create_users_table (0.61 seconds)
Migrating: 2014_10_12_100000_create_password_resets_table
Migrated: 2014_10_12_100000_create_password_resets_table (0.47 seconds)
Migrating: 2019_08_19_000000_create_failed_jobs_table
Migrated: 2019_08_19_000000_create_failed_jobs_table (0.24 seconds)
Migrating: 2020_02_13_030056_create_articulos_table
Migrated: 2020_02_13_030056_create_articulos_table (0.17 seconds)
```

Si te da algún **fallo de sintaxis**, comprueba que no te hayas equivocado al definir la tabla dentro del archivo “*create\_articulos\_table.php*”.

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

```
php artisan config:cache
```

```
D:\xampp\htdocs\DWSE\ejercicios\Taller-IV-Laravel\project\tienda>php artisan config:cache
Configuration cache cleared!
Configuration cached successfully!
```

# Generar Datos de Prueba

## *Configuración del Seeder*

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

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

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

```
DB::table('articulos'); //Vaciamos las tablas
//Datos de prueba
Articulo::create([
    'nombre'=>'Carmona Microphylla',
    'categoria'=>'Bazar',
    'precio'=>'12',
    'stock'=>'2',
    'imagen'=>'/img/articulos/carmona-microphylla.jpg'
]);
Articulo::create([
    'nombre'=>'Absolut Vodka',
    'categoria'=>'Bazar',
    'precio'=>'12.20',
    'stock'=>'50',
    'imagen'=>'/img/articulos/absolut-vodka.jpg'
]);
Articulo::create([
    'nombre'=>'Damel Moras',
    'categoria'=>'Bazar',
    'precio'=>'3.45',
    'stock'=>'34',
    'imagen'=>'/img/articulos/damel-moras.jpg'
]);
Articulo::create([
    'nombre'=>'Pasion Floral',
    'categoria'=>'Hogar',
    'precio'=>'0.95',
    'stock'=>'200',
    'imagen'=>'/img/articulos/pasion-floral.jpg'
]);
```

```
Articulo::create([
    'nombre'=>'Taza Elizabeth',
    'categoria'=>'Hogar',
    'precio'=>'12.47',
    'stock'=>'5',
    'imagen'=>'/img/articulos/taza-elizabeth-gintama.jpg'
]);

Articulo::create([
    'nombre'=>'Horno de Sobremesa',
    'categoria'=>'Hogar',
    'precio'=>'52',
    'stock'=>'17',
    'imagen'=>'/img/articulos/horno-sobremesa.jpg'
]);

Articulo::create([
    'nombre'=>'DeepGaming Havak',
    'categoria'=>'Electrónica',
    'precio'=>'1384',
    'stock'=>'6',
    'imagen'=>'/img/articulos/deepGaming-havak.jpg'
]);

Articulo::create([
    'nombre'=>'AOC 24G2U/BK',
    'categoria'=>'Electrónica',
    'precio'=>'189.99',
    'stock'=>'42',
    'imagen'=>'/img/articulos/AOC-24G2U.jpg'
]);

Articulo::create([
    'nombre'=>'Newskill Kitsune',
    'categoria'=>'Electrónica',
    'precio'=>'139',
    'stock'=>'120',
    'imagen'=>'/img/articulos/newskill-kitsune.jpg'
]);
```

En el mismo archivo, deberás indicar en la parte superior (justo debajo de **use Illuminate\Database\Seeder;**) lo siguiente:

```
use App\Articulo;
use Illuminate\Support\Facades\DB;
```

## Imágenes

### Imágenes locales

Dentro de la carpeta **public**, crea la carpeta **img** y dentro de esta, la carpeta **artículos**. En esta última, añade las imágenes que utilizarás, tanto la de default como las que hayas incluido en el Seeder.

Puedes encontrar las que yo he utilizado en el siguiente enlace de Mega:

[https://mega.nz/#F!jhsH3LYb!KYMWEwvM8k7TGq\\_u9xqhoA](https://mega.nz/#F!jhsH3LYb!KYMWEwvM8k7TGq_u9xqhoA)

### Carpeta del storage

Dirigete al archivo **config/filesystems.php**, cambia la línea 53 y 54:

```
'root' => storage_path('app/public'),
'url' => env('APP_URL').'/storage',
```

Por las líneas siguientes:

```
'root' => public_path('img'),
'url' => env('APP_URL').'/img',
```

## Actualizar la base de datos con los datos de la tabla

Para ello en la terminal debes escribir el siguiente comando:

```
php artisan db:seed
```

```
D:\xampp\htdocs\DWSE\ejercicios\Taller-IV-Laravel\project\tienda>php artisan db:seed
Seeding: ArticleSeeder
Seeded: ArticleSeeder (0.35 seconds)
Database seeding completed successfully.
```

No te preocupes si tarda al cargar

## Comprobar que la tabla ha sido rellena

Comprobamos en mysql que la tabla alumnos tiene datos generados.

```
MariaDB [tiendadb]> select * from articulos;
```

id	nombre	categoria	precio	stock	imagen	created_at	updated_at
1	Carmona Microphylla	Bazar	12.00	2	/img/articulos/carmona-microphylla.jpg	2020-02-13 00:00:21	2020-02-13 00:00:21
2	Absolut Vodka	Bazar	12.20	50	/img/articulos/absolut-vodka.jpg	2020-02-13 00:00:21	2020-02-13 00:00:21
3	Damel Moras	Bazar	3.45	34	/img/articulos/damel-moras.jpg	2020-02-13 00:00:21	2020-02-13 00:00:21
4	Pasion Floral	Hogar	0.95	200	/img/articulos/pasion-floral.jpg	2020-02-13 00:00:21	2020-02-13 00:00:21
5	Taza Elizabeth	Hogar	12.47	5	/img/articulos/taza-elizabeth-gintama.jpg	2020-02-13 00:00:21	2020-02-13 00:00:21
6	Horno de Sobremesa	Hogar	52.00	17	/img/articulos/horno-sobremesa.jpg	2020-02-13 00:00:21	2020-02-13 00:00:21
7	DeepGaming Havak	Electr	1384.00	6	/img/articulos/deepGaming-havak.jpg	2020-02-13 00:00:21	2020-02-13 00:00:21
8	AOC 24G2U/BK	Electr	189.99	42	/img/articulos/AOC-24G2U.jpg	2020-02-13 00:00:21	2020-02-13 00:00:21
9	Newskill Kitsune	Electr	139.00	120	/img/articulos/newskill-kitsune.jpg	2020-02-13 00:00:21	2020-02-13 00:00:21

```
9 rows in set (0.013 sec)
```

# Modificando las rutas (web.php)

Abre el archivo **routes/web.php**, modifica la línea 15:

```
return view('welcome');
```

Cambia **'welcome'** por **'index'**, será el nombre de la nueva view para el inicio de la página.

```
return view('index');
```

Añade, también, al final, la siguiente sentencia:

```
Route::resource('articulos', 'ArticuloController');
```

Será la página inicio de artículos.

## Métodos del Controlador

### *index*

En el controlador de artículos (**app/http/controllers/ArticuloController.php**), dentro de la función **index**, añade el siguiente código:

```
$categorias=['Bazar','Hogar','Electrónica'];  
$precios=['Menos de 10€',  
    'De 10€ a 50€',  
    'De 50€ a 100€',  
    'Más de 100€'];  
  
$categoria=$request->get('categoria');  
$precio=$request->get('precio');  
  
$articulos=Articulo::orderBy('id')  
->categoria($categoria)  
->precio($precio)  
->paginate(3);  
  
return  
view('articulos.index',compact('articulos','categorias','precios','re  
quest'));
```

## Scopes

Añade el siguiente código en `app/Articulo.php`:

```
public function scopeCategoria($query, $v){
    return $query->where('categoria','like','%"$v%"');
}

public function scopePrecio($query, $v){
    switch($v){
        case 0:
            return $query->where('precio','>',0);
            break;
        case 1:
            return $query->where('precio','<',10);
            break;
        case 2:
            return $query->where('precio','>',10)
                ->where('precio','<',50);
            break;
        case 3:
            return $query->where('precio','>',50)
                ->where('precio','<',100);
            break;
        case 4:
            return $query->where('precio','>',100);
            break;
    }
}
```

## Show

En el controlador de artículos (`app/http/controllers/ArticuloController.php`), dentro de la función `show`, añade la siguiente sentencia:

```
return view('articulos.show',compact('articulo'));
```



## Create

En el controlador de artículos (*app/http/controllers/ArticuloController.php*), en la parte superior del fichero, debajo de todos los “**use**” añade la siguiente sentencia:

```
use Illuminate\Support\Facades\Storage;
```

En el método **create**, añade la siguiente sentencia:

```
return view('articulos.create');
```

Finalmente, en el método **store**, añade el siguiente código:

```
$request->validate([
    'nombre'=>['required'],
    'categoria'=>['required'],
    'precio'=>['required']
]);
//compruebo si he subido archiivo
if($request->has('imagen')){
    $request->validate([
        'imagen'=>['image']
    ]);
    //Todo bien hemos subido un archivo y es de imagen
    $file=$request->file('imagen');
    //Creo un nombre

    $nombre='articulos/'.time().'_'.$file->getClientOriginalName();
    //Guardo el archivo de imagen
    Storage::disk('public')->put($nombre, \File::get($file));
    //Guardo el articulo pero la imagen estaria mal
    $articulo=Articulo::create($request->all());
    //actualiza el registro imagen del articulo guardado
    $articulo->update(['imagen'=>"/img/$nombre"]);
}
else{
    Articulo::create($request->all());
}
return
redirect()->route('articulos.index')->with('mensaje','Artículo
guardado correctamente');
```

## Edit

En el controlador de artículos (*app/http/controllers/ArticuloController.php*), dentro de la función **edit**, añade la siguientes sentencias:

```
$categorias=['Bazar','Hogar','Electrónica'];  
return view('articulos.edit',compact('articulo','categorias'));
```

Además, en el método **update**, añade el siguiente código:

```
$request->validate([  
    'nombre'=>['required'],  
    'categoria'=>['required'],  
    'precio'=>['required']  
]);  
//compruebo si he subido archiivo  
if($request->has('imagen')){  
    $request->validate([  
        'imagen'=>['image']  
    ]);  
    //Todo bien hemos subido un archivo y es de imagen  
    $file=$request->file('imagen');  
    //Creo un nombre  
  
    $nombre='articulos/'.time().'_'.$file->getClientOriginalName();  
    //Guardo el archivo de imagen  
    Storage::disk('public')->put($nombre, \File::get($file));  
    //si he subido una imagen nueva borro la vieja, SALVO que  
    sea default.jpg  
    if(basename($articulo->imagen)!='default.png'){  
        unlink(public_path().$articulo->imagen);  
    }  
    //ahora actualizo el articulo  
    $articulo->update($request->all());  
    $articulo->update(['imagen'=>"/img/$nombre"]);  
}  
else{  
    $articulo->update($request->all());  
}  
return  
redirect()->route('articulos.index')->with('mensaje','Artículo  
actualizado correctamente');
```

## *Destroy*

En el controlador de artículos (*app/http/controllers/ArticuloController.php*), dentro de la función **destroy**, añade el siguiente código:

```
//Dos cosas borrar la imagen si no es default.jpg
//y borrar registro
$imagen=$articulo->imagen;
if(basename($imagen)!='default.png'){
    //la borro NO es default.png
    unlink(public_path().$imagen);
}
//en cualquier caso borro el registro
$articulo->delete();
return
redirect()->route('articulos.index')->with('mensaje','Artículo
borrado satisfactoriamente');
```

# Creación de plantillas

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



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

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

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

# Creación de vistas

## *index*

Crea dentro de **resources/views** un archivo llamado **index.blade.php** e inserta en él el siguiente código:

```
<!DOCTYPE html>
<html lang="es">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width,
initial-scale=1.0">
  <meta http-equiv="X-UA-Compatible" content="ie=edge">
  <link rel="stylesheet"
href="https://stackpath.bootstrapcdn.com/bootstrap/4.4.1/css/bootstrap.min.css"
integrity="sha384-Vkoo8x4CGsO3+Hhvxv8T/Q5PaXtkKtu6ug5TOeNV6gBiFeWPGFN9MuhOf23Q9Ifjh" crossorigin="anonymous">
  <title>Tienda online</title>
</head>
<body style="background: bisque">
  <h3 class="text-center mt-3">Mejor que Amazon</h3>
  <div class="container mt-3 text-center">
    <a href="{{route('articulos.index')}}" class="btn btn-info mr-1">Artículos</a>
  </div>
</body>
</html>
```

## Artículos/index

Dentro del directorio **resources/views/articulos** (si no tienes esta carpeta, créala) crea el archivo **index.blade.php** y pega dentro el siguiente código:

```
@extends('plantillas.plantilla')
@section('titulo')
Artículos
@endsection
@section('cabecera')
Artículos Disponibles
@endsection
@section('contenido')
@if($texto=Session::get('mensaje'))
<p class="alert alert-success my-3">{{ $texto }}</p>
@endif
<div class="container">
<a href="{{route('articulos.create')}}" class="btn btn-success
mb-3">Guardar Artículo</a>
<form name="search" method="get"
action="{{route('articulos.index')}}" class="form-inline
float-right">
    <i class="fa fa-search fa-2x ml-2 mr-2" aria-hidden="true"></i>
    <select name='categoria' class='form-control mr-2'>
        <option value='%>Todos</option>
        @foreach($categorias as $categoria)
            @if($categoria==$request->categoria)
                <option selected>{{ $categoria }}</option>
            @else
                <option>{{ $categoria }}</option>
            @endif
        @endforeach
    </select>
    <select name="precio" class="form-control">
        <option value='0'>Todos</option>
        <?php $cont=1; ?>
        @foreach ($precios as $precio)
            @if($cont==$request->precio)
                <option selected="" value="<?php echo $cont
?>">{{ $precio }}</option>
```



```

@else
    <option value="<?php echo $cont ?>">{{ $precio }}</option>
@endif
<?php $cont++; ?>
@endforeach
<input type="submit" value="Buscar" class="btn btn-info ml-2">
</form>
</div>
<table class="table table-striped table-dark mt-3">
    <thead>
        <tr>
            <th scope="col">Detalles</th>
            <th scope="col">Nombre</th>
            <th scope="col">Categoria</th>
            <th scope="col">Imagen</th>
            <th scope="col">Acciones</th>
        </tr>
    </thead>
    <tbody>
        @foreach($articulos as $articulo)
            <tr>
                <th scope="row"><a
href="{{route('articulos.show', $articulo)}}"
style="text-decoration:none"><i class="fa fa-address-card
fa-3x"></i></a></th>
                <td class="align-middle">{{ $articulo->nombre }}</td>
                <td class="align-middle">{{ $articulo->categoria }}</td>
                <td>
                    
                </td>
                <td class="align-middle">
                    <form name="borrar" method='post'
action='{{route('articulos.destroy', $articulo)}}'>
                        @csrf
                        @method('DELETE')
                        <a href='{{route('articulos.edit', $articulo)}}' class='btn
btn-warning'>Editar</a>
                        <button type='submit' class='btn btn-danger' onclick="return

```

```
confirm('¿Borrar artículo?')">Borrar</button>
</form>
</td>
</tr>
@endforeach
</tbody>
</table>
{{ $articulos->appends(Request::except('page'))->links() }}
@endsection
```

## Artículos/show

Dentro del directorio **resources/views/articulos** (si no tienes esta carpeta, créala) crea el archivo **show.blade.php** y pega dentro el siguiente código:

```
@extends('plantillas.plantilla')
@section('titulo')
    Detalle Artículo {{ $articulo->nombre }}
@endsection
@section('cabecera')
    <i>{{ $articulo->categoria."/"/ }}<b>{{ ($articulo->nombre) }}</b></i>
@endsection
@section('contenido')
    <span class="clearfix"></span>
    <div class="card text-white bg-info mt-5 mx-auto"
style="max-width: 48rem;">
        <div class="card-header
text-center"><b>{{ ($articulo->nombre) }}</b></div>
        <div class="card-body" style="font-size: 1.1em">
            <p class="card-text">
                <div class="float-right"></div>
                <p><b>Categoria:</b> {{ $articulo->categoria }}</p>
                <p><b>Precio (€):</b> {{ $articulo->precio }}</p>
                <p><b>Stock:</b> {{ $articulo->stock." unidades" }}</p>
                <a href="{{ route('articulos.index') }}" class="float-left
btn btn-success">Volver</a>
            </div>
```

```
</div>
@endsection
```

## *Articulos/create*

Dentro del directorio **resources/views/articulos** (si no tienes esta carpeta, créala) crea el archivo **create.blade.php** y pega dentro el siguiente código:

```
@extends('plantillas.plantilla')
@section('titulo')
Nuevo Artículo
@endsection
@section('cabecera')
Guardar un Nuevo Artículo
@endsection
@section('contenido')
@if($errors->any())
    <div class="alert alert-danger">
        <ul>
            @foreach($errors->all() as $miError)
                <li>{{ $miError }}</li>
            @endforeach
        </ul>
    </div>
@endif
<form name="c" method='POST' action="{{ route('articulos.store') }}"
enctype="multipart/form-data">
    @csrf
    <div class="form-row">
        <div class="col">
            <input type="text" class="form-control" placeholder="Nombre"
name='nombre' required>
        </div>
    </div>
    <div class="form-row mt-3">
        <div class="col">
            <select name='categoria' class="form-control">
                <option selected>Bazar</option>
                <option>Hogar</option>
                <option>Electrónica</option>
```

```
        </select>
    </div>

    <div class="col">
        <input type="number" class="form-control"
placeholder="Precio (€)" name="precio" required step="0.50" min="0">
    </div>

    <div class="col">
        <input type="number" class="form-control"
placeholder="Stock" name="klms" min="0">
    </div>
</div>

<div class="form-row mt-3">
    <div class="col">
        <b>Imagen</b>&nbsp;<input type='file' name='imagen'
accept="image/*">
    </div>
</div>

<div class="form-row mt-3">
    <div class="col">
        <input type='submit' value='Guardar' class='btn
btn-success mr-3'>
        <input type='reset' value='Limpiar' class='btn
btn-warning mr-3'>
        <a href={{route('articulos.index')}} class='btn
btn-info'>Volver</a>
    </div>
</div>
</form>
@endsection
```

## Articulos/edit

Dentro del directorio **resources/views/articulos** (si no tienes esta carpeta, createla) crea el archivo **edit.blade.php** y pega dentro el siguiente código:

```
@extends('plantillas.plantilla')
@section('titulo')
Actualizar Artículo
@endsection
@section('cabecera')
Actualizar Artículo {!!$articulo->nombre!!}
@endsection
@section('contenido')
@if($errors->any())
    <div class="alert alert-danger">
        <ul>
            @foreach($errors->all() as $miError)
                <li>{!!$miError!!}</li>
            @endforeach
        </ul>
    </div>
@endif
<form name="c" method='POST' action="{{route('articulos.update',
$articulo)}}" enctype="multipart/form-data">
    @csrf
    @method('PUT')
    <div class="form-row">
        <div class="col">
            <input type="text" class="form-control"
value="{!!$articulo->nombre!!}" name='nombre' required>
        </div>
    </div>
    <div class="form-row mt-3">
        <div class="col">
            <select name='categoria' class="form-control">
                @foreach ($categorias as $categoria)
                    @if ($articulo->categoria==$categoria)
                        <option selected>{!!$categoria!!}</option>
                    @else
```

```
        <option>{{$categoria}}</option>
    @endif
    @endforeach
</select>
</div>

<div class="col">
    <input type="number" class="form-control"
value="{{$articulo->precio}}" name="precio" required step="0.50"
min="0">
</div>

<div class="col">
    <input type="number" class="form-control"
value="{{$articulo->stock}}" name="klms" min="0">
</div>
</div>
<div class="form-row mt-3">
    <div class="col">
        
        <b>Imagen</b>&nbsp;<input type='file' name='imagen'
accept="image/*">
    </div>
</div>
<div class="form-row mt-3">
    <div class="col">
        <input type='submit' value='Modificar' class='btn
btn-success mr-3'>
        <a href={{route('articulos.index')}} class='btn
btn-info'>Volver</a>
    </div>
</div>
</form>
@endsection
```



# Comprobamos que todo funcione en la web

## Listado

Artículos Disponibles

[Guardar Artículo](#)    [Buscar](#)

Detalles	Nombre	Categoría	Imagen	Acciones
	Carmona Microphylla	Bazar		<a href="#">Editar</a> <a href="#">Borrar</a>
	Absolut Vodka	Bazar		<a href="#">Editar</a> <a href="#">Borrar</a>
	Damel Moras	Bazar		<a href="#">Editar</a> <a href="#">Borrar</a>

[<](#) [1](#) [2](#) [3](#) [>](#)

## Paginación

Artículos Disponibles

[Guardar Artículo](#)    [Buscar](#)

Detalles	Nombre	Categoría	Imagen	Acciones
	Pasion Floral	Hogar		<a href="#">Editar</a> <a href="#">Borrar</a>
	Taza Elizabeth	Hogar		<a href="#">Editar</a> <a href="#">Borrar</a>
	Horno de Sobremesa	Hogar		<a href="#">Editar</a> <a href="#">Borrar</a>

[<](#) [1](#) [2](#) [3](#) [>](#)

## Búsqueda

Artículos Disponibles

Guardar Artículo  De 10€ a 50€

Detalles	Nombre	Categoría	Imagen	Acciones
	Carmona Microphylla	Bazar		<input type="button" value="Editar"/> <input type="button" value="Borrar"/>
	Absolut Vodka	Bazar		<input type="button" value="Editar"/> <input type="button" value="Borrar"/>

## Detalles

Detalle Artículo Carmona Microphylla

localhost/DWESE/ejercicios/Taller-IV-Laravel/project/tienda/public/articulos/1


Bazar/**Carmona Microphylla**

Carmona Microphylla

Categoría: Bazar

Precio (€): 12.00

Stock: 2 unidades



## Creación

Artículos Disponibles

Guardar Artículo

Detalles	Nombre	Categoría	Imagen	Acciones
	Artículo	Bazar		<input type="button" value="Editar"/> <input type="button" value="Borrar"/>

< 1 2 3 4 >







*Editado*

Artículos Disponibles

[Guardar Artículo](#)

Detalles	Nombre	Categoría	Imagen	Acciones
	Carmona Microphylla	Bazar		<input type="button" value="Editar"/> <input type="button" value="Borrar"/>
	Absolut Vodka	Bazar		<input type="button" value="Editar"/> <input type="button" value="Borrar"/>
	Damel Moras	Bazar		<input type="button" value="Editar"/> <input type="button" value="Borrar"/>

< 1 2 3 >

Detalles	Nombre	Categoría	Imagen	Acciones
	Planta	Bazar		<input type="button" value="Editar"/> <input type="button" value="Borrar"/>
	Absolut Vodka	Bazar		<input type="button" value="Editar"/> <input type="button" value="Borrar"/>
	Damel Moras	Bazar		<input type="button" value="Editar"/> <input type="button" value="Borrar"/>

## Borrado







Artículos Disponibles

Guardar Artículo

Todos

Todos

Buscar

Detalles	Nombre	Categoría	Imagen	Acciones
	Absolut Vodka	Bazar		<div>Editar</div> <div>Borrar</div>
	Damel Moras	Bazar		<div>Editar</div> <div>Borrar</div>
	Pasion Floral	Hogar		<div>Editar</div> <div>Borrar</div>

<

1

2

3

4

5

6

>

# Google Cloud

## *Instalar PHP*

Necesitarás hacer uso de un repositorio externo e instalar dependencias para la instalación de **php**. Ejecuta los siguientes comandos:

```
apt-get install python-software-properties
```

```
sudo add-apt-repository ppa:ondrej/php
```

Si te da el error **add-apt-repository command not found**, utiliza el siguiente comando y prueba de nuevo:

```
sudo apt-get install software-properties-common
```

Ahora realiza un update:

```
apt-get update
```

Finalmente, instala **php** con la siguiente línea de comando:

```
sudo apt-get install php7.2
```

En caso de que no puedas encontrar el paquete **php7.2**, ejecuta el siguiente comando y, después, vuelve a probar a instalar **php**:

```
apt-cache search php7.2
```

Comprueba que está instalado revisando su **versión**:

```
php --version
```

## *Instalar Composer*

Utiliza las siguientes líneas de comandos para instalar **composer**:

```
cd /usr/src
```

```
sudo apt-get install curl php5-cli
```

```
curl -sS https://getcomposer.org/installer | sudo php --  
--install-dir=/usr/local/bin --filename=composer
```

Comprueba que está instalado revisando su **versión**:

```
composer --version
```

## *Instalar git y clonar*

Instala **git** con el siguiente comando:

```
sudo apt install git
```

Comprueba que está instalado revisando su **versión**:

```
git --version
```

**Clona** tu repositorio de la siguiente manera:

```
git clone en /var/www/html
```

## *Instalar Mysql Server y Configurar BD*

Para instalar **mysql server** ejecuta el siguiente comando:

```
apt-get install mysql-server
```

Comprueba que está instalado revisando su **versión**:

```
mysql --version
```

**Crea la base de datos y el usuario** que tenías para el proyecto, y establece, al usuario, **permisos sobre la base de datos**.

Crea el archivo **.env** en tu proyecto, ya que este archivo es ignorado por git. Puedes copiar el archivo **.env.example** y cambiarle el nombre.

**!** Asegurate de que la información sobre tu base de datos es la correcta.

Realiza **uno** de los siguientes comandos sobre el directorio de tu proyecto:

```
sudo composer install
```

```
sudo composer update
```

**Migra las tablas** de la base de datos:

```
sudo php artisan migrate:fresh
```

● Al intentar **migrar**, puede dar una **serie de errores**, he recopilado los que me han sucedido a mí junto con cómo los he logrado solucionar. ●

Si te da el siguiente error:

**Your requirements could not be resolved to an installable set of packages.**

Utiliza los siguientes comandos y vuelve a intentar migrar:

```
sudo apt-get install php7.2-mbstring
```

```
sudo apt-get install php7.2-xml
```

Si te da el error **[RuntimeException]**, prueba a darle permisos a la carpeta **/var/www** con el siguiente comando, y después vuelve a intentar migrar:

```
sudo chmod -R 777 /var/www
```

Si te da el error de que **el usuario no tiene permiso** sobre la base de datos, utiliza los siguientes comandos **sobre la terminal de mysql**, después de **iniciar con root**:

```
drop user userpdo@localhost;
```

```
flush privileges;
```

```
create user userpdo@localhost identified by 'secreto';
```

```
grant all privileges on tiendadb.* to userpdo@localhost;
```

**!** Modifica el texto en rojo por el que se corresponda a tu caso.

Si te da un error acerca de los **drivers**, ejecuta el siguiente comando:

```
sudo apt-get install php7.2-mysql
```

Finalmente, **carga la información** de las tablas contenida en el **Seeder**:

```
sudo php artisan db:seed
```

## Solución Rutas/Permisos de escritura

Es posible que llegados a este punto te funcione la página principal de tu proyecto, pero al intentar entrar en artículos, te encuentres con un error, los siguientes pasos te ayudarán a solucionar dicho inconveniente.

Elimina el `create-users-table` y el `create-password-resets-table` del directorio `database/migrations`.

Estos archivos dan problemas y no los necesitas para esta práctica.

```
miguelangel_lopez_alumnado@taller-iv-laravel:/var/www/html/Laravel_Tienda/tienda/database/migrations$ dir
2014_10_12_000000_create_users_table.php      2019_08_19_000000_create_failed_jobs_table.php
2014_10_12_100000_create_password_resets_table.php  2020_02_13_030056_create_articulos_table.php
miguelangel_lopez_alumnado@taller-iv-laravel:/var/www/html/Laravel_Tienda/tienda/database/migrations$ rm 2014_10_12_000000_create_users_table.php
miguelangel_lopez_alumnado@taller-iv-laravel:/var/www/html/Laravel_Tienda/tienda/database/migrations$ rm 2014_10_12_100000_create_password_resets_table.php
miguelangel_lopez_alumnado@taller-iv-laravel:/var/www/html/Laravel_Tienda/tienda/database/migrations$ dir
2019_08_19_000000_create_failed_jobs_table.php  2020_02_13_030056_create_articulos_table.php
miguelangel_lopez_alumnado@taller-iv-laravel:/var/www/html/Laravel_Tienda/tienda/database/migrations$
```

Modifica el archivo `/etc/apache2/apache2.conf` con el siguiente comando:

```
nano /etc/apache2/apache2.conf
```

Busca la línea que contenga el siguiente texto: “<Directory /var/www/>” y, dentro de esa etiqueta, la que contenga el texto “`AllowOverride`”, cambia el “`None`” de esa línea por “`All`”.

```
miguelangel_lopez_alumnado@taller-iv-laravel:/var/www/html/Laravel_Tienda/tienda/database/migrations$ dir
2014_10_12_000000_create_users_table.php      2019_08_19_000000_create_failed_jobs_table.php
2014_10_12_100000_create_password_resets_table.php  2020_02_13_030056_create_articulos_table.php
miguelangel_lopez_alumnado@taller-iv-laravel:/var/www/html/Laravel_Tienda/tienda/database/migrations$ rm 2014_10_12_000000_create_users_table.php
miguelangel_lopez_alumnado@taller-iv-laravel:/var/www/html/Laravel_Tienda/tienda/database/migrations$ rm 2014_10_12_100000_create_password_resets_table.php
miguelangel_lopez_alumnado@taller-iv-laravel:/var/www/html/Laravel_Tienda/tienda/database/migrations$ dir
2019_08_19_000000_create_failed_jobs_table.php  2020_02_13_030056_create_articulos_table.php
miguelangel_lopez_alumnado@taller-iv-laravel:/var/www/html/Laravel_Tienda/tienda/database/migrations$
```

Finalmente, activa los permisos de escritura en apache con el siguiente comando:

```
sudo a2enmod rewrite
```

También debes reiniciar el servicio de apache, con el siguiente comando:

```
sudo systemctl restart apache2
```

## Repositorio GIT

[https://github.com/LobozeL/Laravel\\_Tienda](https://github.com/LobozeL/Laravel_Tienda)