

Desarrollo de Aplicaciones Web

Manual de Laravel

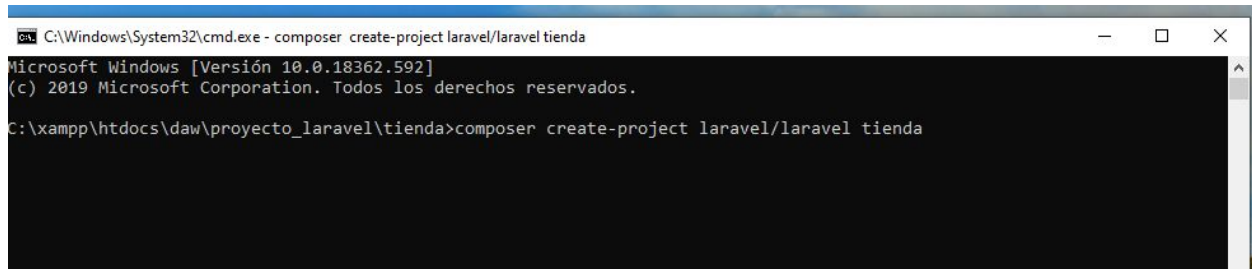


Índice

Creación de un proyecto	3
Activación del Servidor Laravel	3
Migración, Controlador y Vistas	4
Creación de la base de datos	4
Configurar nuestro proyecto Laravel	6
Creación de campos	7
Migración a la base de datos	8
Inserción de datos en la base de datos	9
Rutas	13
Controlador	13
Método Index	14
Método create	14
Método store	15
Método show	16
Método edit	16
Método update	16
Método destroy	17
Scopes	17
Creación de plantillas	18
Creación de Views	19
Index General	19
Index Artículos	20
Mostrar Artículos	22
Crear Artículos	23
Modificar Artículos	24
URL Git	25

Creación de un proyecto

Nos situamos en el directorio donde queremos crear el proyecto y usamos el siguiente comando: **composer create-project laravel/laravel nombreDelProyecto**



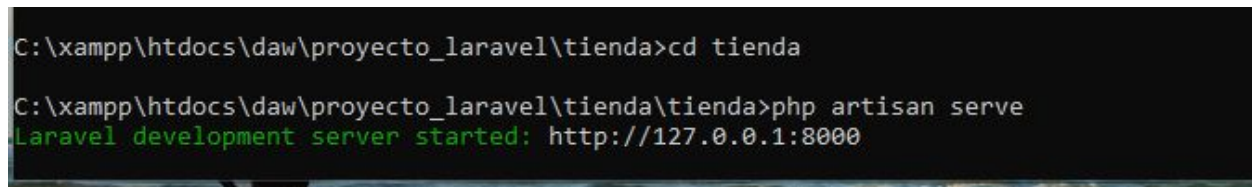
```
C:\Windows\System32\cmd.exe - composer create-project laravel/laravel tienda
Microsoft Windows [Versión 10.0.18362.592]
(c) 2019 Microsoft Corporation. Todos los derechos reservados.

C:\xampp\htdocs\daw\proyecto_laravel\tienda>composer create-project laravel/laravel tienda
```

Activación del Servidor Laravel

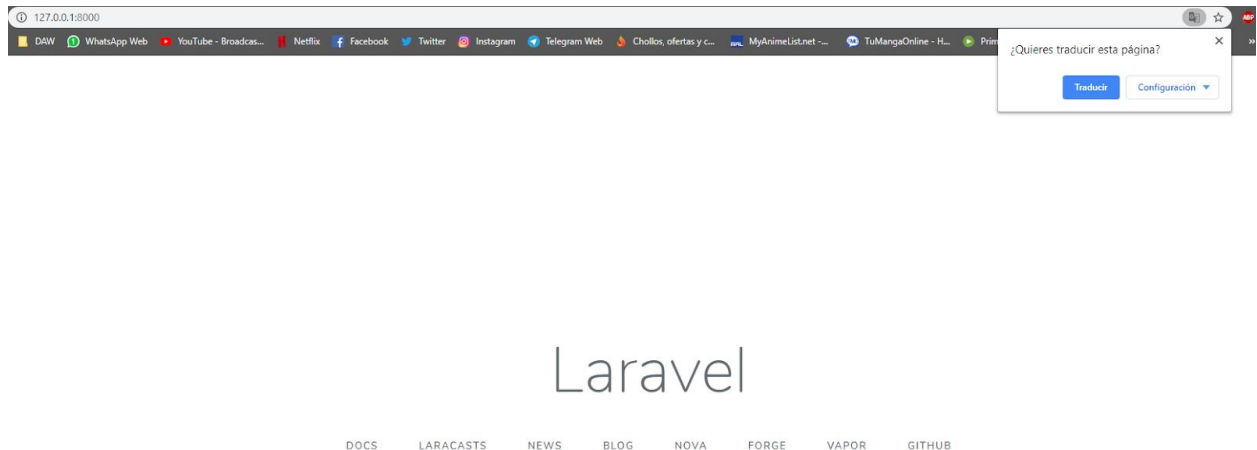
A continuación activaremos el servidor.

Para ello, nos vamos a la carpeta del proyecto que hemos creado y hacemos uso del comando: **php artisan serve**



```
C:\xampp\htdocs\daw\proyecto_laravel\tienda>cd tienda
C:\xampp\htdocs\daw\proyecto_laravel\tienda\tienda>php artisan serve
Laravel development server started: http://127.0.0.1:8000
```

Ponemos en el navegador la dirección que nos ha dado para comprobar que todo ha ido bien. Nos debe de salir la siguiente página:



Migración, Controlador y Vistas

En la consola de comandos, dentro de nuestro proyecto, escribimos: **php artisan make:model Artículo -mcrs**

```
C:\Windows\System32\cmd.exe
Microsoft Windows [Versión 10.0.18362.592]
(c) 2019 Microsoft Corporation. Todos los derechos reservados.

C:\xampp\htdocs\daw\proyecto_laravel\tienda\tienda>php artisan make:model Artículo -mcrs
Model created successfully.
Created Migration: 2020_02_13_184130_create_articulos_table
Seeder created successfully.
Controller created successfully.

C:\xampp\htdocs\daw\proyecto_laravel\tienda\tienda>
```

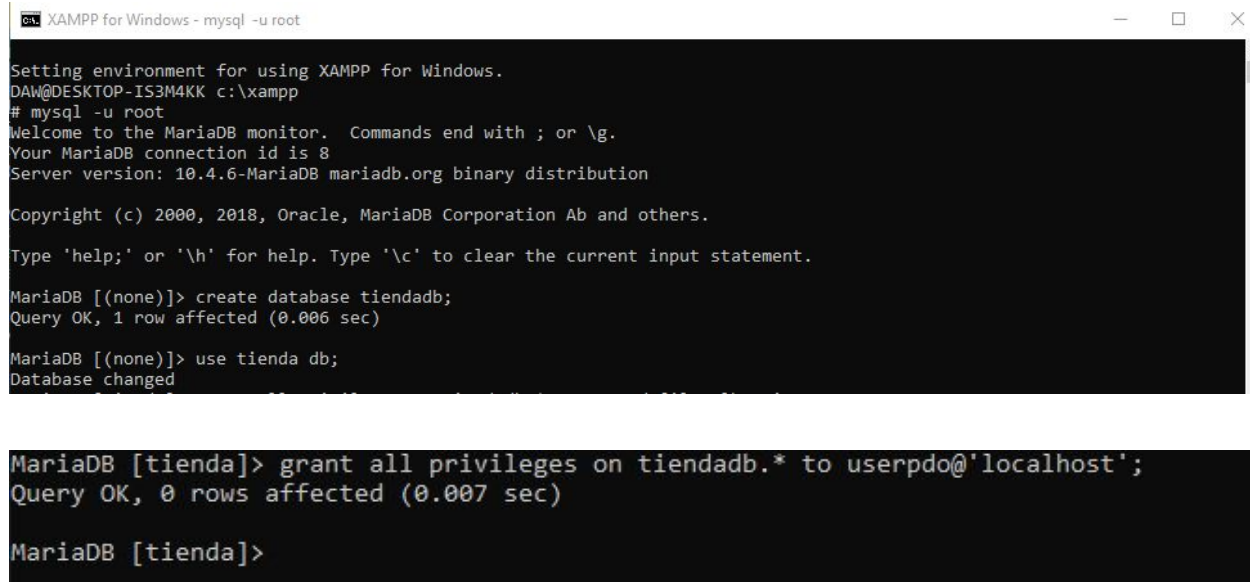
Creación de la base de datos

Abrimos la consola de comandos del Xampp (Shell), e iniciamos sesión en MySQL.

Creamos la base de datos con: **create database tiendadb;**

Hacemos uso de la base de datos con: **use tiendadb;**

Le damos permisos a un usuario que tengamos en esa base de datos: **grant all privileges on tiendadb.* to userpdo@'localhost';**

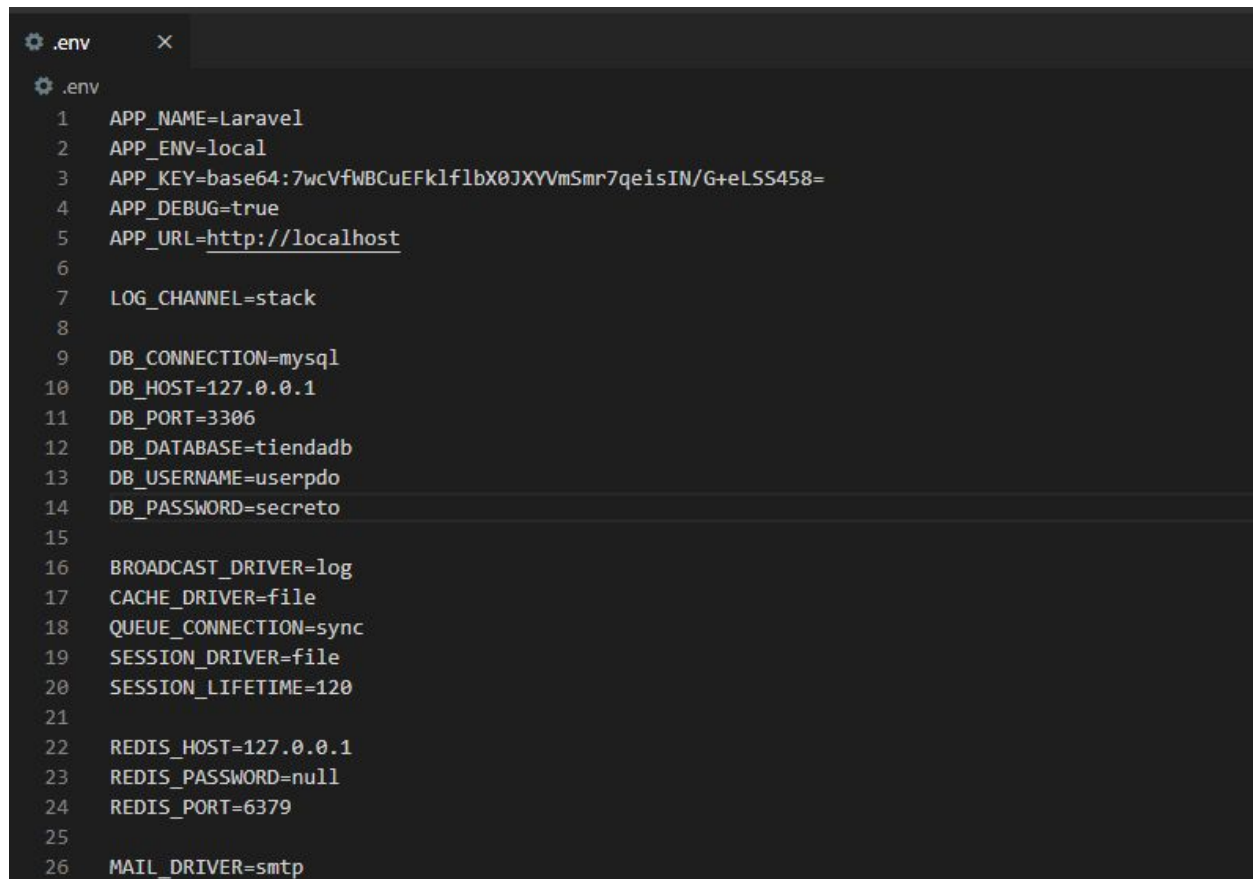
A screenshot of a terminal window titled "XAMPP for Windows - mysql -u root". The terminal shows the following text:

```
Setting environment for using XAMPP for Windows.  
DAW@DESKTOP-IS3M4KK c:\xampp  
# mysql -u root  
Welcome to the MariaDB monitor.  Commands end with ; or \g.  
Your MariaDB connection id is 8  
Server version: 10.4.6-MariaDB mariadb.org binary distribution  
Copyright (c) 2000, 2018, Oracle, MariaDB Corporation Ab and others.  
Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.  
  
MariaDB [(none)]> create database tiendadb;  
Query OK, 1 row affected (0.006 sec)  
  
MariaDB [(none)]> use tienda db;  
Database changed  
  
MariaDB [tienda]> grant all privileges on tiendadb.* to userpdo@'localhost';  
Query OK, 0 rows affected (0.007 sec)  
  
MariaDB [tienda]>
```

Configurar nuestro proyecto Laravel

Abrimos nuestro proyecto en Visual Studio y seleccionamos el archivo **.env**

A continuación, escribimos los datos de nuestra base de datos:



```
.env
1 APP_NAME=Laravel
2 APP_ENV=local
3 APP_KEY=base64:7wcVfWBCuEFk1f1bX0JXVvmSmr7qeisIN/G+eLSS458=
4 APP_DEBUG=true
5 APP_URL=http://localhost
6
7 LOG_CHANNEL=stack
8
9 DB_CONNECTION=mysql
10 DB_HOST=127.0.0.1
11 DB_PORT=3306
12 DB_DATABASE=tiendadb
13 DB_USERNAME=userpdo
14 DB_PASSWORD=secreto
15
16 BROADCAST_DRIVER=log
17 CACHE_DRIVER=file
18 QUEUE_CONNECTION=sync
19 SESSION_DRIVER=file
20 SESSION_LIFETIME=120
21
22 REDIS_HOST=127.0.0.1
23 REDIS_PASSWORD=null
24 REDIS_PORT=6379
25
26 MAIL_DRIVER=smtp
```

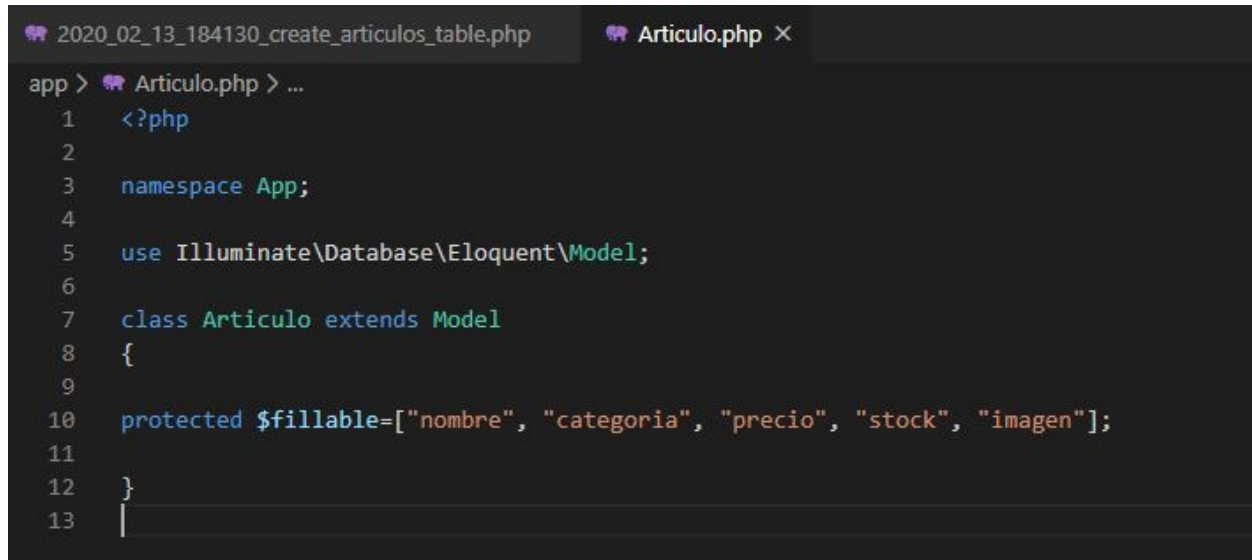
Creación de campos

En nuestro proyecto, nos deberemos de ir a **database>migrations** y ahí dentro busquemos el archivo que contenga la palabra **create_articulos_table.php**.

Una vez abierto, escribiremos los campos que tendrá nuestra tabla Artículos.

```
2020_02_13_184130_create_articulos_table.php X
database > migrations > 2020_02_13_184130_create_articulos_table.php > CreateArticulosTable > up
1  <?php
2
3  use Illuminate\Database\Migrations\Migration;
4  use Illuminate\Database\Schema\Blueprint;
5  use Illuminate\Support\Facades\Schema;
6
7  class CreateArticulosTable extends Migration
8  {
9      /**
10       * Run the migrations.
11       *
12       * @return void
13       */
14     public function up()
15     {
16         Schema::create('articulos', function (Blueprint $table) {
17             $table->bigIncrements('id');
18             $table->string('nombre');
19             $table->string('categoria',20);
20             $table->decimal('precio',10,2);
21             $table->integer('stock')->default(0);
22             $table->string('imagen')->default('/img/default.png');
23             $table->timestamps();
24         });
25     }
26
27     /**
28      * Reverse the migrations.
29      *
30      * @return void
31      */
32     public function down()
33     {
34         Schema::dropIfExists('articulos');
35     }
36 }
37
```

Ahora, debemos de ir a `app>Providers>Articulo.php` y añadir lo siguiente:



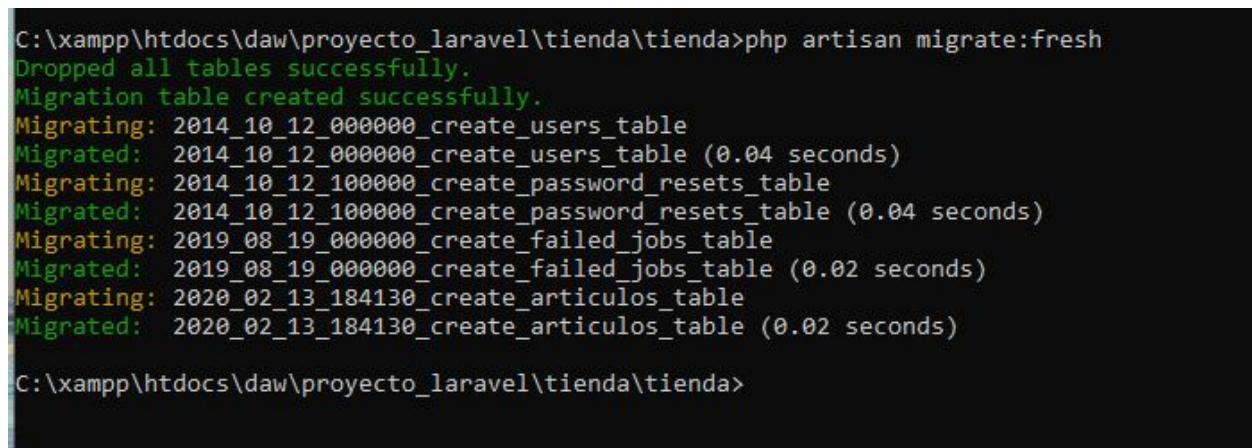
The screenshot shows a code editor with two tabs: `2020_02_13_184130_create_articulos_table.php` and `Articulo.php`. The `Articulo.php` tab is active, showing the following PHP code:

```
app > Article.php > ...
1  <?php
2
3  namespace App;
4
5  use Illuminate\Database\Eloquent\Model;
6
7  class Articulo extends Model
8  {
9
10     protected $fillable=["nombre", "categoria", "precio", "stock", "imagen"];
11
12 }
13 |
```

Para así poder añadir datos por formulario.

Migración a la base de datos

En la consola de comandos, dentro del directorio de nuestro proyecto, escribimos: **php artisan migrate:fresh**



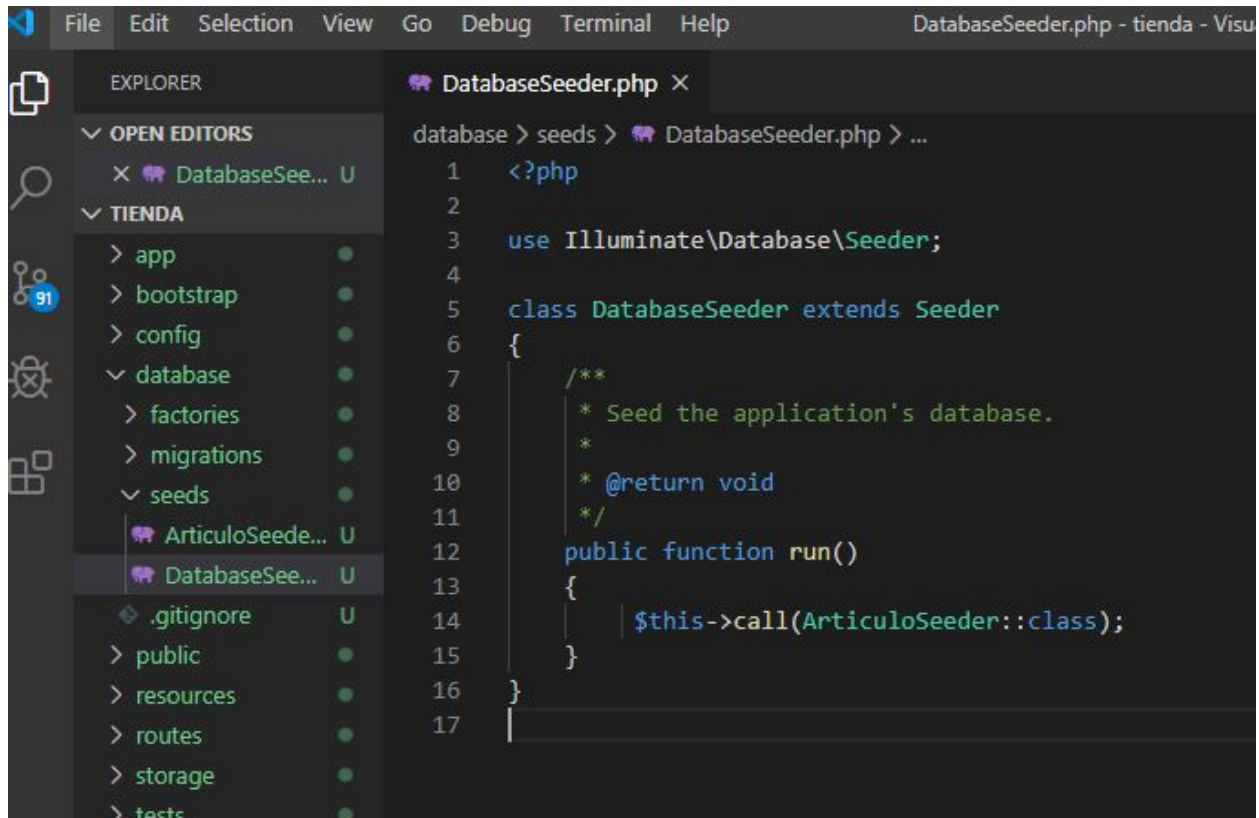
The screenshot shows a command prompt window with the following output:

```
C:\xampp\htdocs\daw\proyecto_laravel\tienda\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 (0.04 seconds)
Migrating: 2014_10_12_100000_create_password_resets_table
Migrated: 2014_10_12_100000_create_password_resets_table (0.04 seconds)
Migrating: 2019_08_19_000000_create_failed_jobs_table
Migrated: 2019_08_19_000000_create_failed_jobs_table (0.02 seconds)
Migrating: 2020_02_13_184130_create_articulos_table
Migrated: 2020_02_13_184130_create_articulos_table (0.02 seconds)

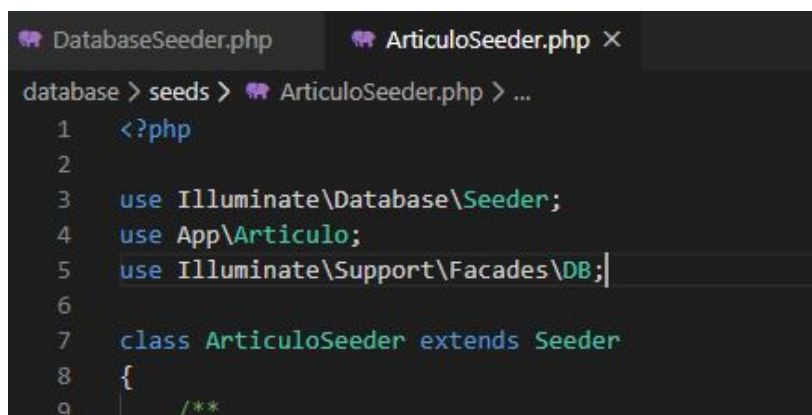
C:\xampp\htdocs\daw\proyecto_laravel\tienda\tienda>
```


Inserción de datos en la base de datos

Nos vamos a **database>seeds>DatabaseSeeder.php** y dentro de la función **run()**, escribimos: **\$this->call(ArticuloSeeder::class);**

A screenshot of a code editor window titled 'DatabaseSeeder.php - tienda - Visual Studio'. The Explorer sidebar on the left shows a project structure with folders like 'app', 'bootstrap', 'config', 'database', 'factories', 'migrations', 'seeds', 'public', 'resources', 'routes', 'storage', and 'tests'. The 'seeds' folder is expanded, showing 'ArticuloSeeder.php' and 'DatabaseSeeder.php'. The main editor area shows the code for 'DatabaseSeeder.php' with line numbers 1 through 17. The code includes a PHP opening tag, a use statement for 'Illuminate\Database\Seeder', a class definition 'class DatabaseSeeder extends Seeder', a docblock for the 'run()' method, and the method body which calls '\$this->call(ArticuloSeeder::class);'.

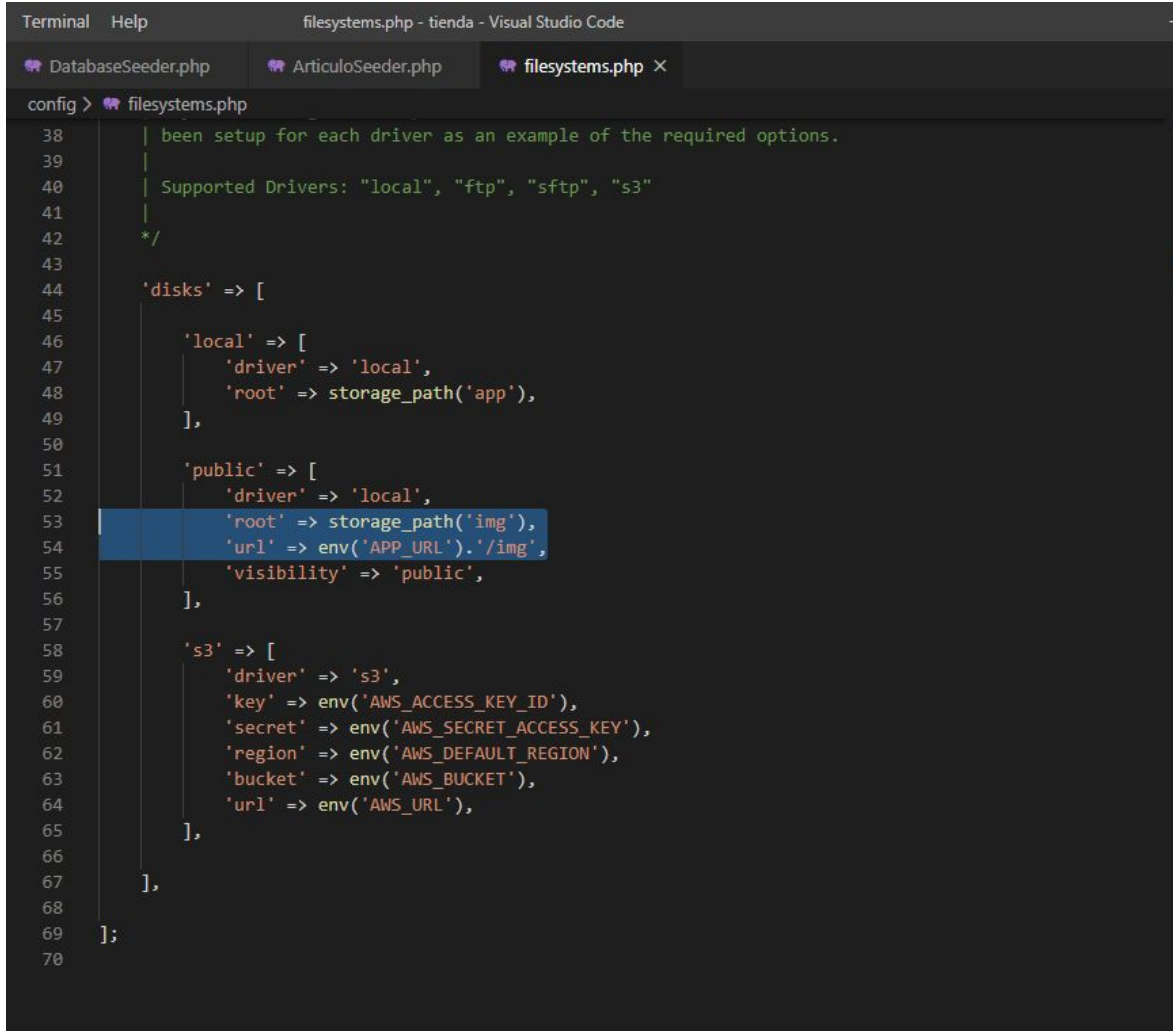
A continuación, en **ArticuloSeeder.php**, añadimos lo siguiente en la parte de arriba del archivo:

A screenshot of a code editor window showing two tabs: 'DatabaseSeeder.php' and 'ArticuloSeeder.php'. The 'ArticuloSeeder.php' tab is active, showing the beginning of the file with line numbers 1 through 9. The code includes a PHP opening tag, use statements for 'Illuminate\Database\Seeder', 'App\Articulo', and 'Illuminate\Support\Facades\DB', and the start of a class definition 'class ArticuloSeeder extends Seeder'.

Y creamos algunos artículos dentro de la función **run()**:

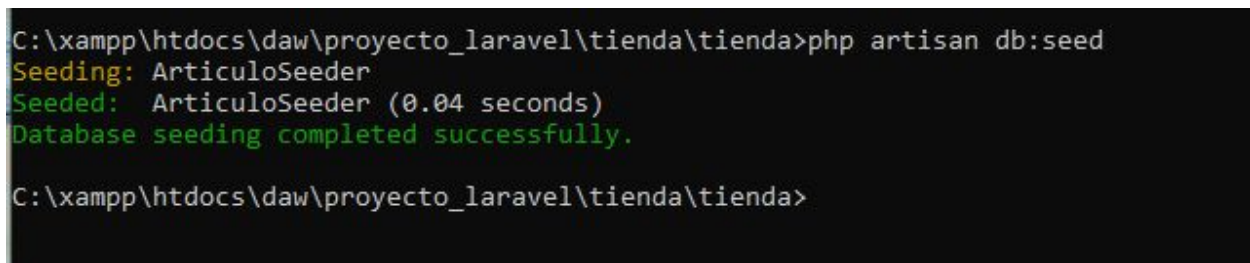
```
11  */
12  public function run()
13  {
14      DB::table('articulos');
15
16      Artículo::create([
17          'nombre'=>'Ventilador',
18          'categoria'=>'Hogar',
19          'precio'=>'30',
20          'stock'=>'30',
21          'imagen'=>'/img/ventilador.png'
22      ]);
23      Artículo::create([
24          'nombre'=>'Cachimba',
25          'categoria'=>'Bazar',
26          'precio'=>'25',
27          'stock'=>'5',
28          'imagen'=>'/img/cachimba.png'
29      ]);
30      Artículo::create([
31          'nombre'=>'Ordenador',
32          'categoria'=>'Electronica',
33          'precio'=>'600',
34          'stock'=>'10',
35          'imagen'=>'/img/ordenador.png'
36      ]);
37      Artículo::create([
38          'nombre'=>'Vajilla',
39          'categoria'=>'Hogar',
40          'precio'=>'15',
41          'stock'=>'90',
42          'imagen'=>'/img/vajilla.png'
43      ]);
44      Artículo::create([
45          'nombre'=>'Lavadora',
46          'categoria'=>'Hogar',
47          'precio'=>'300',
48          'stock'=>'50',
49          'imagen'=>'/img/lavadora.png'
50      ]);
51  }
52  }
```

Importante: Las imágenes las situamos dentro de una carpeta llamada `img` que debemos de crear en la carpeta `public` de nuestro proyecto. Por último, nos vamos a **config>filesystems.php** y cambiamos las líneas 53 y 54 por lo siguiente:



```
Terminal  Help  filesystems.php - tienda - Visual Studio Code
DatabaseSeeder.php  ArtículoSeeder.php  filesystems.php X
config > filesystems.php
38  | been setup for each driver as an example of the required options.
39  |
40  | Supported Drivers: "local", "ftp", "sftp", "s3"
41  |
42  */
43
44  'disks' => [
45
46      'local' => [
47          'driver' => 'local',
48          'root' => storage_path('app'),
49      ],
50
51      'public' => [
52          'driver' => 'local',
53          'root' => storage_path('img'),
54          'url' => env('APP_URL').'/img',
55          'visibility' => 'public',
56      ],
57
58      's3' => [
59          'driver' => 's3',
60          'key' => env('AWS_ACCESS_KEY_ID'),
61          'secret' => env('AWS_SECRET_ACCESS_KEY'),
62          'region' => env('AWS_DEFAULT_REGION'),
63          'bucket' => env('AWS_BUCKET'),
64          'url' => env('AWS_URL'),
65      ],
66  ],
67
68
69 ];
70
```

Y actualizamos la base de datos con el comando: **php artisan db:seed**



```
C:\xampp\htdocs\daw\proyecto_laravel\tienda\tienda>php artisan db:seed
Seeding: ArtículoSeeder
Seeded: ArtículoSeeder (0.04 seconds)
Database seeding completed successfully.

C:\xampp\htdocs\daw\proyecto_laravel\tienda\tienda>
```

Comprobamos que los datos están dentro:

```
MariaDB [(none)]> use tiendadb;
Database changed
MariaDB [tiendadb]> show tables;
+-----+
| Tables_in_tiendadb |
+-----+
| articulos           |
| failed_jobs         |
| migrations          |
| password_resets     |
| users               |
+-----+
5 rows in set (0.001 sec)

MariaDB [tiendadb]> select * from articulos;
+-----+-----+-----+-----+-----+-----+-----+-----+
| id | nombre | categoria | precio | stock | imagen | created_at | updated_at |
+-----+-----+-----+-----+-----+-----+-----+-----+
| 1 | Ventilador | Hogar | 30.00 | 30 | /img/ventilador.png | 2020-02-13 19:36:56 | 2020-02-13 19:36:56 |
| 2 | Cachimba | Bazar | 25.00 | 5 | /img/cachimba.png | 2020-02-13 19:36:56 | 2020-02-13 19:36:56 |
| 3 | Ordenador | Electronica | 600.00 | 10 | /img/ordenador.png | 2020-02-13 19:36:56 | 2020-02-13 19:36:56 |
| 4 | Vajilla | Hogar | 15.00 | 90 | /img/vajilla.png | 2020-02-13 19:36:56 | 2020-02-13 19:36:56 |
| 5 | Lavadora | Hogar | 300.00 | 50 | /img/lavadora.png | 2020-02-13 19:36:56 | 2020-02-13 19:36:56 |
| 6 | Ventilador | Hogar | 30.00 | 30 | /img/ventilador.png | 2020-02-13 19:38:43 | 2020-02-13 19:38:43 |
| 7 | Cachimba | Bazar | 25.00 | 5 | /img/cachimba.png | 2020-02-13 19:38:43 | 2020-02-13 19:38:43 |
| 8 | Ordenador | Electronica | 600.00 | 10 | /img/ordenador.png | 2020-02-13 19:38:43 | 2020-02-13 19:38:43 |
| 9 | Vajilla | Hogar | 15.00 | 90 | /img/vajilla.png | 2020-02-13 19:38:43 | 2020-02-13 19:38:43 |
| 10 | Lavadora | Hogar | 300.00 | 50 | /img/lavadora.png | 2020-02-13 19:38:43 | 2020-02-13 19:38:43 |
+-----+-----+-----+-----+-----+-----+-----+-----+
10 rows in set (0.000 sec)

MariaDB [tiendadb]>
```

Rutas

Para modificar la ruta y añadir nuevas, debemos de ir a **routes>web.php** y a continuación escribimos lo siguiente:



```
web.php x
routes > web.php
1  <?php
2
3  /*
4  |-----
5  | Web Routes
6  |-----
7  |
8  | Here is where you can register web routes for your application. These
9  | routes are loaded by the RouteServiceProvider within a group which
10 | contains the "web" middleware group. Now create something great!
11 |
12 */
13
14 Route::get('/', function () {
15     return view('index');
16 });
17
18 Route::resource('articulos', 'ArticuloController');
```

Controlador

Para modificar el controlador debemos de ir a **app>Http>Controllers>ArticuloController.php**. Una vez ahí, debemos de añadir al principio del archivo:

```
<?php

namespace App\Http\Controllers;

use App\Articulo;
use Illuminate\Http\Request;
use Illuminate\Support\Facades\Storage;
|
class ArticuloController extends Controller
{
    /**
```

Y modificaremos los siguientes métodos.

Método Index

```
public function index()
{
    $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(5);

    return view('articulos.index',compact('articulos','categorias','precios','request'));
}

/**
 * Show the form for creating a new resource.
 */
```

Método create

```
*/
public function create()
{
    return view('articulos.create');
}

/**
 * Store a newly created resource in storage.
 */
```

Método store

```
public function store(Request $request)
{
    $request->validate([
        'nombre'=>['required'],
        'categoria'=>['required'],
        'precio'=>['required']
    ]);

    if($request->has('imagen')){
        $request->validate([
            'imagen'=>['image']
        ]);

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

Método show

```
public function show(Articulo $articulo)
{
    return view('articulos.show',compact('articulo'));
}
```

Método edit

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

Método update

```
public function update(Request $request, Articulo $articulo)
{
    $request->validate([
        'nombre'=>['required'],
        'categoria'=>['required'],
        'precio'=>['required']
    ]);

    if($request->has('imagen')){
        $request->validate([
            'imagen'=>['image']
        ]);

        $file=$request->file('imagen');
        $nombre=time().'_'.$file->getClientOriginalName();
        Storage::disk('public')->put($nombre, \File::get($file));
        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','Articulo Actualizado');
```

Método destroy

```
public function destroy(Articulo $articulo)
{
    $imagen=$articulo->imagen;

    if(basename($imagen)!='default.png'){
        unlink(public_path().$imagen);
    }
    $articulo->delete();
    return redirect()->route('articulos.index')->with('mensaje','Artículo Eliminado');
}
```

Scopes

Por último, en **app>Articulo.php**, añadimos el siguiente código para el rango de precio:

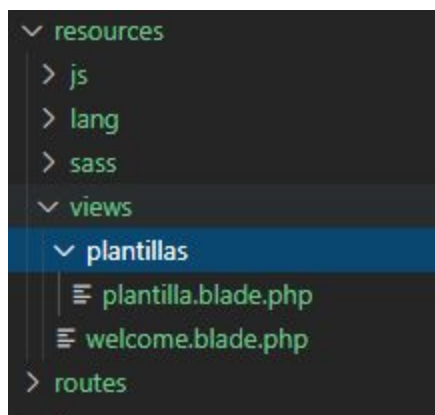
```
class Articulo extends Model
{
    protected $fillable=["nombre", "categoria", "precio", "stock", "imagen"];

    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;
        }
    }
}
```

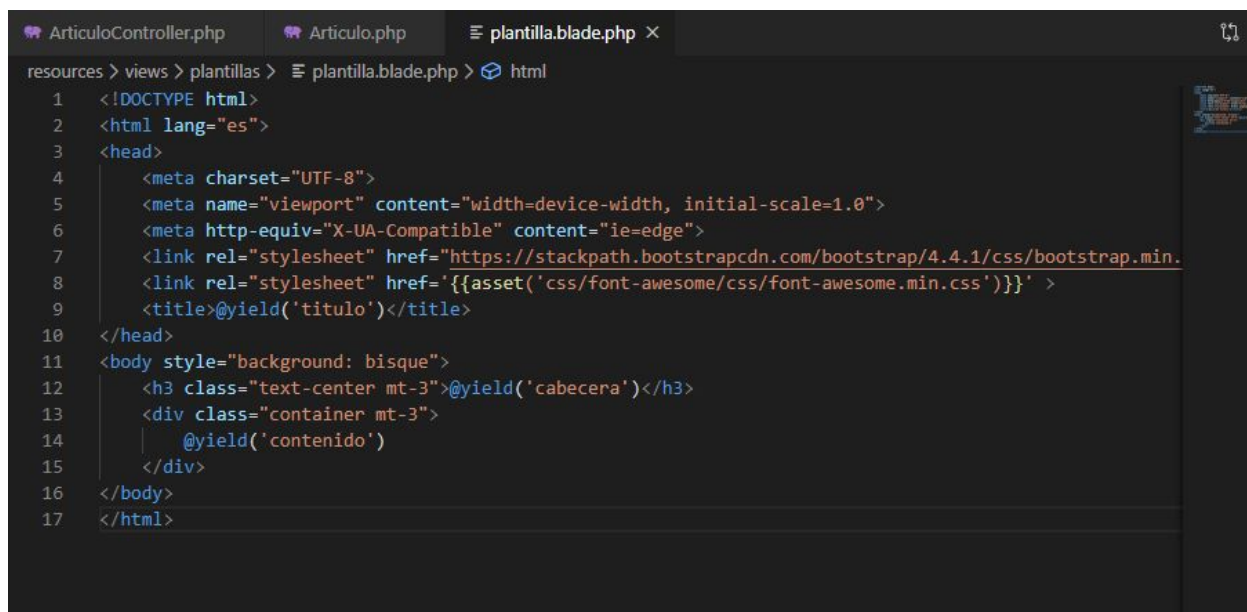
Creación de plantillas

Para crear las plantillas, debemos de situarnos en **resources>views**. Una vez ahí, crearemos la carpeta **plantillas** y dentro de ésta, será donde crearemos la plantilla de nuestra tienda. El archivo debe de tener la extensión **blade.php**.



Ahora, crearemos nuestra plantilla, la cual usarán algunas de las páginas de nuestro crud.

Las partes titulo, cabecera y contenido llevan @yield por delante. Esto indica que serán modificadas en el resto de páginas.

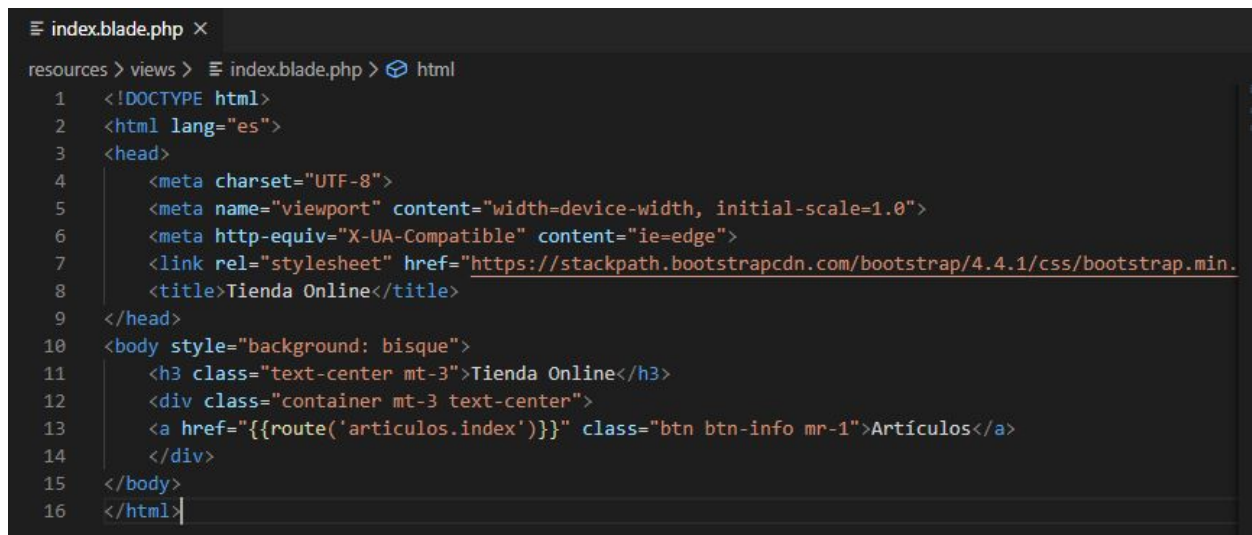


Creación de Views

Para crear las vistas, debemos de dirigirnos a **resources>views**. Dentro de esta carpeta, crearemos un archivo llamado `index.blade.php` y una carpeta llamada `artículos`, que contendrá el `index` de artículos y las páginas de mostrar, crear, modificar y eliminar artículos.

Comenzamos:

Index General

A screenshot of a code editor with a dark theme. The file name 'index.blade.php' is visible in the top left. The breadcrumb navigation shows 'resources > views > index.blade.php'. The code is written in HTML and Blade templating syntax. It includes a DOCTYPE declaration, HTML lang attribute, a head section with meta tags for charset, viewport, and http-equiv, a link to Bootstrap CSS, and a title 'Tienda Online'. The body has a background color of 'bisque' and contains an h3 'Tienda Online', a container div, and a link to 'Articulos' using a Laravel route. The code is numbered from 1 to 16.

```
1 <!DOCTYPE html>
2 <html lang="es">
3 <head>
4     <meta charset="UTF-8">
5     <meta name="viewport" content="width=device-width, initial-scale=1.0">
6     <meta http-equiv="X-UA-Compatible" content="ie=edge">
7     <link rel="stylesheet" href="https://stackpath.bootstrapcdn.com/bootstrap/4.4.1/css/bootstrap.min.
8     <title>Tienda Online</title>
9 </head>
10 <body style="background: bisque">
11     <h3 class="text-center mt-3">Tienda Online</h3>
12     <div class="container mt-3 text-center">
13         <a href="{{route('articulos.index')}}" class="btn btn-info mr-1">Articulos</a>
14     </div>
15 </body>
16 </html>
```

Index Artículos

```
@extends('plantillas.plantilla')
@section('titulo')
Artículos
@endsection
@section('cabecera')
Artículos
@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-warning 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
        </select>
        <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', $articulo)}}" class="btn btn-info">Editar</a>
                        <button type="submit" class="btn btn-danger">Borrar</button>
                    </form>
                </td>
            </tr>
        @endforeach
    </tbody>
</table>
{{$articulos->appends(Request::except('page'))->links()}}
@endsection

```

Mostrar Artículos

Nos creamos el archivo: **show.blade.php**

```
index.blade.php ...views  index.blade.php ...articulos  show.blade.php X
resources > views > articulos > show.blade.php > ...
1  @extends('plantillas.plantilla')
2  @section('titulo')
3      Detalles {{$articulo->nombre}}
4  @endsection
5  @section('cabecera')
6      <i>{{$articulo->categoria."/"}<b>{{$articulo->nombre}}</b></i>
7  @endsection
8  @section('contenido')
9      <span class="clearfix"></span>
10     <div class="card text-white bg-info mt-5 mx-auto" style="max-width: 48rem;">
11         <div class="card-header text-center"><b>{{$articulo->nombre}}</b></div>
12         <div class="card-body" style="font-size: 1.1em">
13             <p class="card-text">
14                 <div class="float-right"></div>
15                 <p><b>Categoria:</b> {{$articulo->categoria}}</p>
16                 <p><b>Precio:</b> {{$articulo->precio}}</p>
17                 <p><b>Stock:</b>{{$articulo->stock}}</p>
18             </p>
19             <a href="{{route('articulos.index')}}" class="float-left btn btn-warning">Volver</a>
20         </div>
21     </div>
22 @endsection
23
```

Crear Artículos

```
@extends('plantillas.plantilla')
@section('titulo')
    Nuevo Artículo
@endsection
@section('cabecera')
    Nuevo Artículo
@endsection
@section('contenido')
    @if($errors->any())
        <div class="alert alert-danger">
            <ul>
                @foreach($errors->all() as $error)
                    <li>{{$error}}</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>
```

```
        <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><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-danger mr-3'>
            <a href="{{route('articulos.index')}}" class='btn btn-info'>Volver</a>
        </div>
    </div>
</form>
@endsection
```

Modificar Artículos

```
@extends('plantillas.plantilla')
@section('titulo')
Modificar Artículo |
@endsection
@section('cabecera')
Modificar Artículo {{$articulo->nombre}}
@endsection
@section('contenido')
@if($errors->any())
    <div class="alert alert-danger">
        <ul>
            @foreach($errors->all() as $error)
                <li>{{$error}}</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='Guardar' class='btn btn-success mr-3'>
    <a href="{{route('articulos.index')}}" class='btn btn-warning'>Volver</a>
  </div>
</div>
</form>
@endsection
```

URL Git

<https://github.com/Santzu-159/tienda.git>