

Manual de programador

ÍNDICE:

1. **Inicio**
 - 1.1. Configuración del archivo .env
 - 1.2. Instalación del sistema de registro y login
2. **Migraciones**
 - 2.1. Crear migración para categorías
 - 2.2. Crear migración para productos
 - 2.3. Ejecutar migraciones
3. **Categorías**
 - 3.1. Modelo Categoria.php
 - 3.2. Controlador CategoriaController.php
 - 3.3. Vistas de categorías
 - 3.3.1. Create
 - 3.3.2. Edit
 - 3.3.3. Vista general
 - 3.3.4. Vista detallada
4. **Productos**
 - 4.1. Modelo Producto.php
 - 4.2. Controlador ProductoController.php
 - 4.3. Vistas de productos
 - 4.3.1. Create
 - 4.3.2. Edit
 - 4.3.3. Vista principal
 - 4.3.4. Vista detallada
5. **Rutas**
 - 5.1. Definición de rutas en routes/web.php
 - 5.2. Middleware de autenticación
6. **Vistas Generales**
 - 6.1. Vista home
 - 6.2. Uso de Bootstrap y @csrf
7. **Instalación de Laravel en Español**
 - 7.1. Instalación del paquete laraveles/spanish
 - 7.2. Configuración en config/app.php
8. **Futuras Expansiones**
 - 8.1. Gestión de roles
 - 8.2. Vista de catálogo para usuarios
 - 8.3. Realización de pedidos
 - 8.4. Configuración de pasarelas de pago

Inicio:

En .env cambio app_name de laravel a FitMatch y DB_DATABASE al nombre de mi base de datos.

```
APP_NAME=FitMatch
APP_ENV=local
APP_KEY=base64:fyHVgw3dxgr9r70cZH8Q3sUU1Z6czaa/yIvsEkDZfn4=
APP_DEBUG=true
APP_URL=http://localhost

LOG_CHANNEL=stack
LOG_DEPRECATIONS_CHANNEL=null
LOG_LEVEL=debug

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

Para crear el register/login:

```
composer require laravel/ui
php artisan ui bootstrap --auth
npm install
npm run dev
```

Migraciones:

Con las migraciones podemos gestionar la estructura de la base de datos desde el código. Para crear las tablas vamos a usar estos comandos:

```
php artisan make:migration create_categorias_table
```

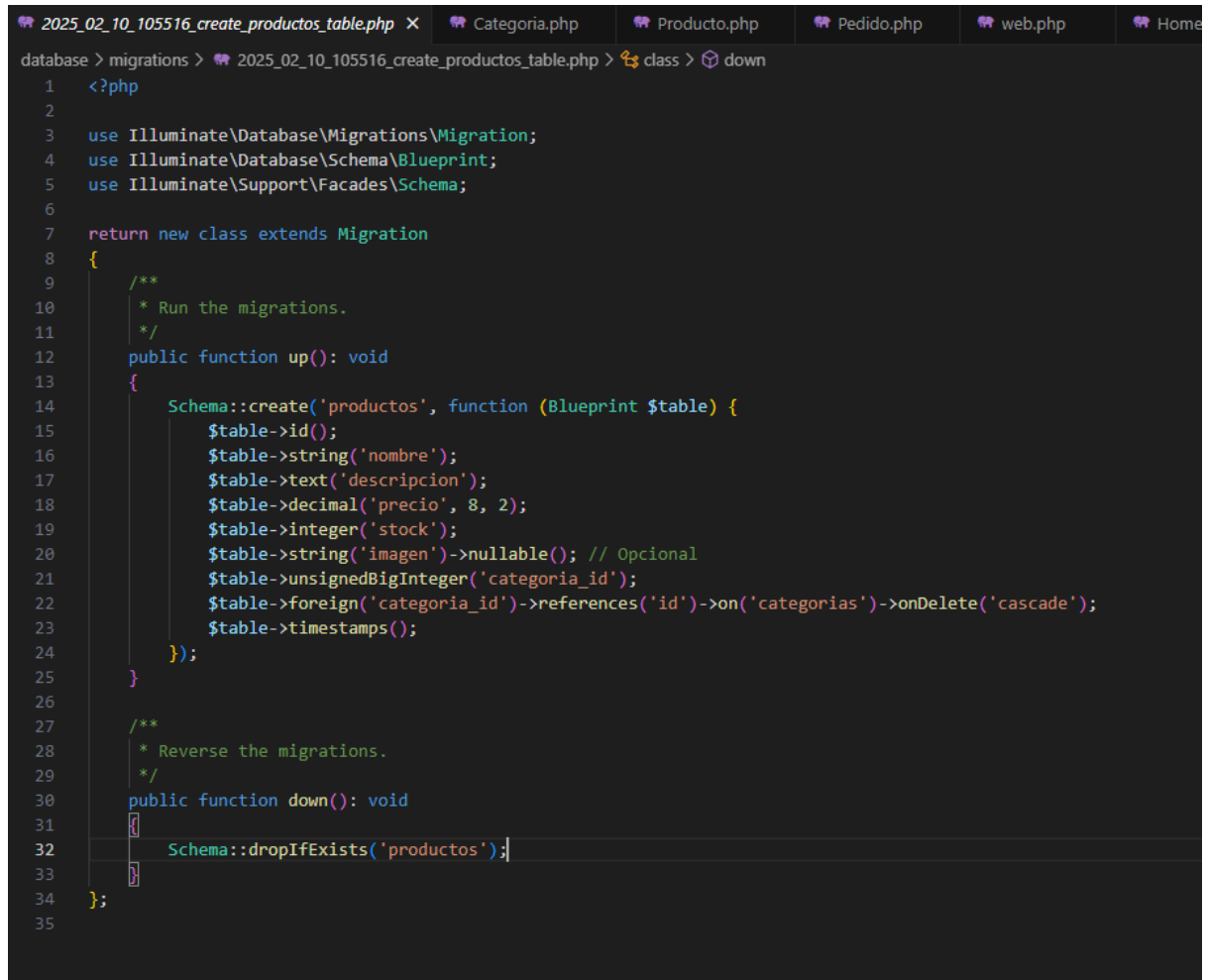
Esto crea un archivo .php donde podremos especificar las características de la tabla a crear.

```
2025_02_10_105425_create_categorias_table.php X Categoria.php Producto.php Pedido.php
database > migrations > 2025_02_10_105425_create_categorias_table.php > ...
1  <?php
2
3  use Illuminate\Database\Migrations\Migration;
4  use Illuminate\Database\Schema\Blueprint;
5  use Illuminate\Support\Facades\Schema;
6
7  return new class extends Migration
8  {
9      /**
10       * Run the migrations.
11       */
12     public function up(): void
13     {
14         Schema::create('categorias', function (Blueprint $table) {
15             $table->id();
16             $table->string('nombre');
17             $table->timestamps();
18         });
19     }
20
21     /**
22      * Reverse the migrations.
23      */
24     public function down(): void
25     {
26         Schema::dropIfExists('categorias');
27     }
28 };
29
```

En este caso, categorías tiene el campo ID, el nombre e información de la hora en la que se ha creado y modificado.

id	nombre	created_at	updated_at
8	Alimentacion	2025-02-20 10:43:14	2025-02-20 10:43:14

php artisan make:migration create_productos_table



```
2025_02_10_105516_create_productos_table.php X Categoria.php Producto.php Pedido.php web.php Home
database > migrations > 2025_02_10_105516_create_productos_table.php > class > down
1  <?php
2
3  use Illuminate\Database\Migrations\Migration;
4  use Illuminate\Database\Schema\Blueprint;
5  use Illuminate\Support\Facades\Schema;
6
7  return new class extends Migration
8  {
9      /**
10       * Run the migrations.
11       */
12     public function up(): void
13     {
14         Schema::create('productos', function (Blueprint $table) {
15             $table->id();
16             $table->string('nombre');
17             $table->text('descripcion');
18             $table->decimal('precio', 8, 2);
19             $table->integer('stock');
20             $table->string('imagen')->nullable(); // Opcional
21             $table->unsignedBigInteger('categoria_id');
22             $table->foreign('categoria_id')->references('id')->on('categorias')->onDelete('cascade');
23             $table->timestamps();
24         });
25     }
26
27     /**
28      * Reverse the migrations.
29      */
30     public function down(): void
31     {
32         Schema::dropIfExists('productos');
33     }
34 };
35
```

En productos tenemos los siguientes campos:

id	nombre	descripcion	precio	stock	imagen	categoria_id	created_at	updated_at
8	Proteína en polvo	Proteína en polvo de la mas alta calidad!	20.00	10	1740048911.png	8	2025-02-20 10:55:11	2025-02-20 10:55:11

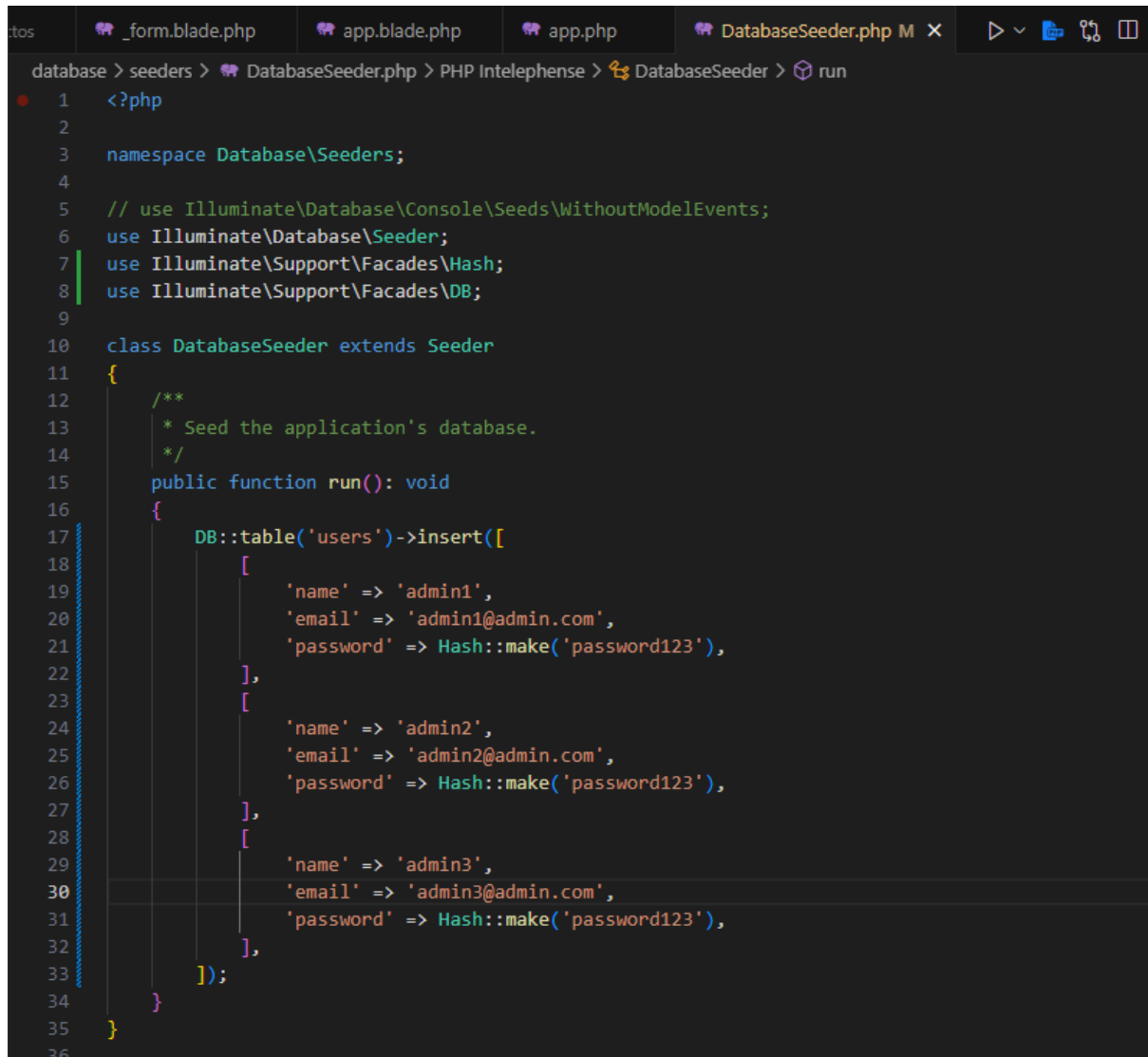
Si nos fijamos en el código, producto está enlazada a categoría. Si se elimina la categoría a la que pertenece un producto, el producto también se elimina.

Una vez tenemos estos dos archivos configurados, hay que ejecutar las migraciones:

php artisan migrate

Creación de usuarios admin:

Configuro DatabaseSeeder.php para que genere los usuarios.



```
database > seeders > DatabaseSeeder.php > PHP Intelephense > DatabaseSeeder > run
1  <?php
2
3  namespace Database\Seeders;
4
5  // use Illuminate\Database\Console\Seeds\WithoutModelEvents;
6  use Illuminate\Database\Seeder;
7  use Illuminate\Support\Facades\Hash;
8  use Illuminate\Support\Facades\DB;
9
10 class DatabaseSeeder extends Seeder
11 {
12     /**
13      * Seed the application's database.
14      */
15     public function run(): void
16     {
17         DB::table('users')->insert([
18             [
19                 'name' => 'admin1',
20                 'email' => 'admin1@admin.com',
21                 'password' => Hash::make('password123'),
22             ],
23             [
24                 'name' => 'admin2',
25                 'email' => 'admin2@admin.com',
26                 'password' => Hash::make('password123'),
27             ],
28             [
29                 'name' => 'admin3',
30                 'email' => 'admin3@admin.com',
31                 'password' => Hash::make('password123'),
32             ],
33         ]);
34     }
35 }
36
```

Después, ejecuto:

php artisan db:seed

Con este comando, se carga en la base de datos los usuarios admins preestablecidos.

Creación de Modelos y Controladores:

Modelos:

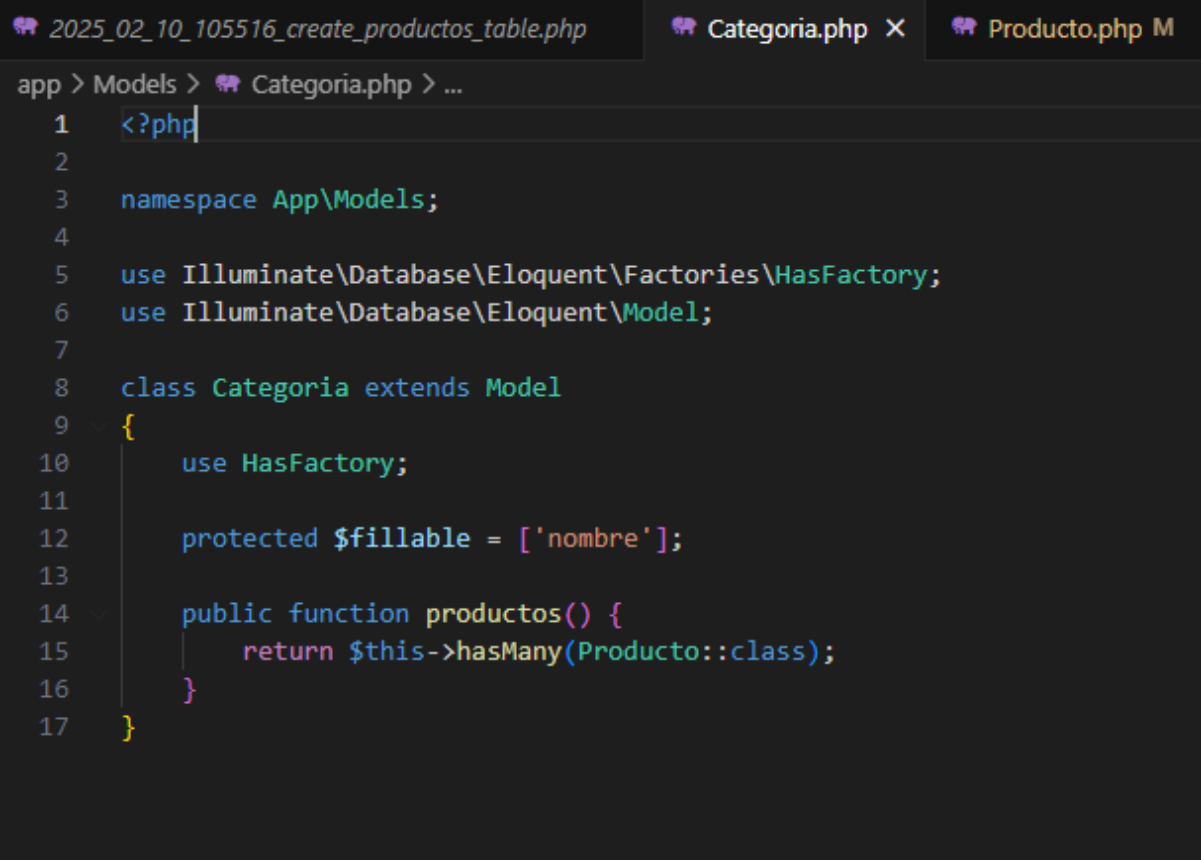
php artisan make:model Categoria

php artisan make:model Producto

Estos comandos sirven para crear clases que conectan tu código con las tablas de la base de datos de forma sencilla.

Crea los archivos Categoria.php y Producto.php

Categoria.php:



```
app > Models > Categoria.php > ...
1  <?php
2
3  namespace App\Models;
4
5  use Illuminate\Database\Eloquent\Factories\HasFactory;
6  use Illuminate\Database\Eloquent\Model;
7
8  class Categoria extends Model
9  {
10     use HasFactory;
11
12     protected $fillable = ['nombre'];
13
14     public function productos() {
15         return $this->hasMany(Producto::class);
16     }
17 }
```

En el modelo Categoria, define la relación con Producto:

Producto.php:

```
2025_02_10_105516_create_productos_table.php  Categoria.php  Producto.php M X
app > Models > Producto.php > PHP Intelephense > Producto > $fillable
1  <?php
2
3  namespace App\Models;
4
5  use Illuminate\Database\Eloquent\Factories\HasFactory;
6  use Illuminate\Database\Eloquent\Model;
7
8  class Producto extends Model
9  {
10     use HasFactory;
11
12     protected $fillable = [
13         'nombre',
14         'descripcion',
15         'precio',
16         'stock',
17         'categoria_id',
18         'imagen'
19     ];
20
21     public function categoria() {
22         return $this->belongsTo(Categoria::class);
23     }
24 }
```

En el modelo Producto, define la relación con Categoria

Controladores:

php artisan make:controller CategoriaController --resource

php artisan make:controller ProductoController --resource

generan controladores listos para gestionar las operaciones básicas de las tablas **categorias** y **productos** sin tener que escribir todo desde cero. Siguen la estructura del CRUD.

Crea los archivos CategoriaController.php y ProductoController.php

CategoriaController.php:

```
CategoriaController.php X
app > Http > Controllers > CategoriaController.php > ...
1  <?php
2
3  namespace App\Http\Controllers;
4
5  use Illuminate\Http\Request;
6  use App\Models\Categoria;
7
8  2 references | 0 implementations
9  class CategoriaController extends Controller
10 {
11     /**
12      * Display a listing of the resource.
13      */
14     0 references | 0 overrides
15     public function index(): Factory|View
16     {
17         $categorias = Categoria::all();
18         return view('categorias.index', data: compact('categorias'));
19     }
20
21     /**
22      * Show the form for creating a new resource.
23      */
24     0 references | 0 overrides
25     public function create(): Factory|View
26     {
27         return view('categorias.create');
28     }
29
30     /**
31      * Store a newly created resource in storage.
32      */
33     0 references | 0 overrides
34     public function store(Request $request): RedirectResponse
35     {
36         $request->validate(rules: [
37             'nombre' => 'required|string|max:255',
38         ]);
39         Categoria::create(attributes: $request->all());
40         return redirect()->route(route: 'categorias.index')->with(key: 'success', value: 'Categoría creada exitosamente.');
```



```

CategoriaController.php X
app > Http > Controllers > CategoriaController.php > PHP > CategoriaController > store()
8 class CategoriaController extends Controller

41 /**
42  * Display the specified resource.
43  */
44 0 references|0 overrides
45 public function show($id): Factory|View
46 {
47     $categoria = Categoria::findOrFail(id: $id);
48     return view(view: 'categorias.show', data: compact(var_name: 'categoria'));
49 }
50
51 /**
52  * Show the form for editing the specified resource.
53  */
54 0 references|0 overrides
55 public function edit($id): Factory|View
56 {
57     $categoria = Categoria::findOrFail(id: $id);
58     return view(view: 'categorias.edit', data: compact(var_name: 'categoria'));
59 }
60
61 /**
62  * Update the specified resource in storage.
63  */
64 0 references|0 overrides
65 public function update(Request $request, $id): RedirectResponse
66 {
67     $request->validate(rules: [
68         'nombre' => 'required|string|max:255',
69     ]);
70
71     $categoria = Categoria::findOrFail(id: $id);
72     $categoria->update(attributes: $request->all());
73
74     return redirect()->route(route: 'categorias.index')->with(key: 'success', value: 'Categoria actualizada exitosamente.');
```

ProductoController.php:

```
ProductoController.php X
app > Http > Controllers > ProductoController.php > PHP > ProductoController > store()
1  <?php
2
3  namespace App\Http\Controllers;
4
5  use Illuminate\Http\Request;
6  use App\Models\Producto;
7  use App\Models\Categoria;
8
9  2 references | 0 implementations
10 class ProductoController extends Controller
11 {
12     /**
13      * Display a listing of the resource.
14      */
15     0 references | 0 overrides
16     public function index(): Factory|View
17     {
18         $productos = Producto::all();
19         return view('productos.index', data: compact('productos'));
20     }
21
22     /**
23      * Show the form for creating a new resource.
24      */
25     0 references | 0 overrides
26     public function create(): Factory|View
27     {
28         $categorias = Categoria::all();
29         return view('productos.create', data: compact('categorias'));
30     }
31
32     /**
33      * Store a newly created resource in storage.
34      */
35     0 references | 0 overrides
36     public function store(Request $request): RedirectResponse
37     {
38         $request->validate(rules: [
39             'nombre' => 'required|string|max:255',
40             'descripcion' => 'required|string',
41             'precio' => 'required|numeric',
42             'stock' => 'required|integer',
43             'categoria_id' => 'required|exists:categorias,id',
44             'imagen' => 'nullable|image|mimes:jpeg,png,jpg,gif,svg|max:2048',
45         ]);
46
47         $data = $request->all();
48
49         if ($request->hasFile(key: 'imagen')) {
50             $fileName = time().'.'.$request->imagen->extension();
51             $request->imagen->move(public_path('images'), $fileName);
52             $data['imagen'] = $fileName;
53         }
54
55         Producto::create(attributes: $data);
56
57         return redirect()->route('productos.index')->with(key: 'success', value: 'Producto creado exitosamente.');
```

```

ProductoController.php X
app > Http > Controllers > ProductoController.php > PHP > ProductoController > update()
9 class ProductoController extends Controller
63 }
64
65 /**
66  * Show the form for editing the specified resource.
67  */
68 0 references | 0 overrides
69 public function edit($id): Factory|View
70 {
71     $producto = Producto::findOrFail(id: $id);
72     $categorias = Categoria::all();
73     return view('productos.edit', data: compact('producto', 'categorias'));
74 }
75
76 /**
77  * Update the specified resource in storage.
78  */
79 0 references | 0 overrides
80 public function update(Request $request, $id): RedirectResponse
81 {
82     $request->validate(rules: [
83         'nombre' => 'required|string|max:255',
84         'descripcion' => 'required|string',
85         'precio' => 'required|numeric',
86         'stock' => 'required|integer',
87         'categoria_id' => 'required|exists:categorias,id',
88         'imagen' => 'nullable|image|mimes:jpeg,png,jpg,gif,svg|max:2048',
89     ]);
90
91     $producto = Producto::findOrFail(id: $id);
92     $data = $request->all();
93
94     if ($request->hasFile(key: 'imagen')) {
95         // Elimina la imagen anterior si existe
96         if ($producto->imagen && file_exists(filename: public_path(path: 'images/' . $producto->imagen))) {
97             unlink(filename: public_path(path: 'images/' . $producto->imagen));
98         }
99
100         $fileName = time().'.'.$request->imagen->extension();
101         $request->imagen->move(public_path(path: 'images'), $fileName);
102         $data['imagen'] = $fileName;
103     }
104
105     $producto->update(attributes: $data);
106
107     return redirect()->route(route: 'productos.index')->with(key: 'success', value: 'Producto actualizado exitosamente.');
```

Rutas:

En el archivo routes/web.php, define las rutas para los CRUDs:

```
web.php x 2025_02_10_105516_create_productos_table.php Categoria.php Producto.php M Pedido.php
routes > web.php > ...
1  <?php
2
3  use Illuminate\Support\Facades\Route;
4  use App\Http\Controllers\CategoriaController;
5  use App\Http\Controllers\ProductoController;
6  use App\Http\Controllers\HomeController;
7  use App\Http\Controllers\Auth\LoginController;
8  use App\Http\Controllers\Auth\RegisterController;
9
10 /*
11 |-----
12 | Web Routes
13 |-----
14 |
15 | Here is where you can register web routes for your application. These
16 | routes are loaded by the RouteServiceProvider and all of them will
17 | be assigned to the "web" middleware group. Make something great!
18 |
19 */
20
21 // Ruta para el login
22 Route::get('login', [LoginController::class, 'showLoginForm'])->name('login');
23 Route::post('login', [LoginController::class, 'login']);
24 Route::post('logout', [LoginController::class, 'logout'])->name('logout');
25
26 // Ruta para el registro
27 Route::get('register', [RegisterController::class, 'showRegistrationForm'])->name('register');
28 Route::post('register', [RegisterController::class, 'register']);
29
30 // Ruta para el home (página principal accesible para todos)
31 Route::get('/home', [HomeController::class, 'index'])->name('home');
32
33 // Redirigir la ruta raíz al home
34 Route::get('/', function () {
35     return redirect()->route('home');
36 });
37
38 // Grupo de rutas protegidas por autenticación
39 Route::middleware('auth')->group(function () {
40     // Rutas para Categorías
41     Route::resource('categorias', CategoriaController::class);
42
43     // Rutas para Productos
44     Route::resource('productos', ProductoController::class);
45 });
```

Se configura las rutas de la web. Define rutas para el login, logout y registro de usuarios usando los controladores LoginController y RegisterController. La ruta raíz /

redirige a /home, gestionada por el HomeController. Además, dentro del middleware auth, se agrupan las rutas CRUD para categorías y productos usando CategoriaController y ProductoController. Estas rutas solo están disponibles si el usuario ha iniciado sesión.

Vistas:

Para el css he usado bootstrap.

En los formularios uso **@csrf** para más seguridad en el envío de los datos.

@foreach para recorrer todo el contenido.

@error si falla algo, que muestre el mensaje.

Create de productos:

```
web.php M create.blade.php X
resources > views > productos > create.blade.php
1 <!-- filepath: /c:/xampp/htdocs/proyecto_laravel/resources/views/productos/create.blade.php -->
2 @extends(view: 'layouts.app')
3
4 @section(section: 'content')
5 <div class="container">
6     <h1>Crear Nuevo Producto</h1>
7     <form action="{{ route(name: 'productos.store') }}" method="POST" enctype="multipart/form-data">
8         @csrf
9         <div class="form-group">
10             <label for="nombre">Nombre:</label>
11             <input type="text" name="nombre" id="nombre" class="form-control" value="{{ old(key: 'nombre') }}">
12             @error('nombre')
13                 <div class="text-danger">{{ $message }}</div>
14             @enderror
15         </div>
16         <div class="form-group">
17             <label for="descripcion">Descripción:</label>
18             <textarea name="descripcion" id="descripcion" class="form-control">{{ old(key: 'descripcion') }}</textarea>
19             @error('descripcion')
20                 <div class="text-danger">{{ $message }}</div>
21             @enderror
22         </div>
23         <div class="form-group">
24             <label for="precio">Precio:</label>
25             <input type="number" step="0.01" name="precio" id="precio" class="form-control" value="{{ old(key: 'precio') }}">
26             @error('precio')
27                 <div class="text-danger">{{ $message }}</div>
28             @enderror
29         </div>
30         <div class="form-group">
31             <label for="stock">Stock:</label>
32             <input type="number" name="stock" id="stock" class="form-control" value="{{ old(key: 'stock') }}">
33             @error('stock')
34                 <div class="text-danger">{{ $message }}</div>
35             @enderror
36         </div>
37         <div class="form-group">
38             <label for="categoria_id">Categoría:</label>
39             <select name="categoria_id" id="categoria_id" class="form-control">
40                 @foreach($categorias as $categoria)
41                     <option value="{{ $categoria->id }}">{{ $categoria->nombre }}</option>
42                 @endforeach
43             </select>
44             @error('categoria_id')
45                 <div class="text-danger">{{ $message }}</div>
46             @enderror
47         </div>
48         <div class="form-group">
49             <label for="imagen">Imagen:</label>
50             <input type="file" name="imagen" id="imagen" class="form-control">
51             @error('imagen')
52                 <div class="text-danger">{{ $message }}</div>
53             @enderror
54         </div>
55         <button type="submit" class="btn btn-primary mt-3">Guardar</button>
56     </form>
57 </div>
58 @endsection
```

Edit de productos:

```
web.php M  edit.blade.php ●
resources > views > productos > edit.blade.php
3  @section(section: 'content')
4  <div class="container">
5      <h1>Editar Producto</h1>
6      <form action="{{ route(name: 'productos.update', parameters: $producto->id) }}" method="POST" enctype="multipart/form-data">
7          @csrf
8          @method('PUT')
9          <div class="form-group">
10             <label for="nombre">Nombre:</label>
11             <input type="text" name="nombre" id="nombre" class="form-control" value="{{ $producto->nombre }}">
12             @error('nombre')
13                 <div class="text-danger">{{ $message }}</div>
14             @enderror
15         </div>
16         <div class="form-group">
17             <label for="descripcion">Descripción:</label>
18             <textarea name="descripcion" id="descripcion" class="form-control">{{ $producto->descripcion }}</textarea>
19             @error('descripcion')
20                 <div class="text-danger">{{ $message }}</div>
21             @enderror
22         </div>
23         <div class="form-group">
24             <label for="precio">Precio:</label>
25             <input type="number" step="0.01" name="precio" id="precio" class="form-control" value="{{ $producto->precio }}">
26             @error('precio')
27                 <div class="text-danger">{{ $message }}</div>
28             @enderror
29         </div>
30         <div class="form-group">
31             <label for="stock">Stock:</label>
32             <input type="number" name="stock" id="stock" class="form-control" value="{{ $producto->stock }}">
33             @error('stock')
34                 <div class="text-danger">{{ $message }}</div>
35             @enderror
36         </div>
37         <div class="form-group">
38             <label for="categoria_id">Categoría:</label>
39             <select name="categoria_id" id="categoria_id" class="form-control">
40                 @foreach($categorias as $categoria)
41                     <option value="{{ $categoria->id }}" {{ $producto->categoria_id == $categoria->id ? 'selected' : '' }}>
42                         {{ $categoria->nombre }}
43                     </option>
44                 @endforeach
45             </select>
46             @error('categoria_id')
47                 <div class="text-danger">{{ $message }}</div>
48             @enderror
49         </div>
50         <div class="form-group">
51             <label for="imagen">Imagen:</label>
52             <input type="file" name="imagen" id="imagen" class="form-control">
53             {{-- @if($producto->imagen)
54                 nombre }}" width="100">
55             @endif --}}
56             @error('imagen')
57                 <div class="text-danger">{{ $message }}</div>
58             @enderror
59         </div>
60         <button type="submit" class="btn btn-primary mt-3">Actualizar</button>
61     </form>
```

Vista principal de productos:

```
index.blade.php X
resources > views > productos > index.blade.php
1 <!-- filepath: /c:/xampp/htdocs/proyecto_laravel/resources/views/productos/index.blade.php -->
2 @extends(view: 'layouts.app')
3
4 @section(section: 'content')
5 <div class="container">
6   <h1>Productos</h1>
7   <a href="{{ route(name: 'productos.create') }}" class="btn btn-success mb-3">Nuevo Producto</a>
8
9   @if(session(key: 'success'))
10     <div class="alert alert-success">
11       {{ session(key: 'success') }}
12     </div>
13   @endif
14
15   <table class="table">
16     <thead>
17       <tr>
18         <th>Nombre</th>
19         <th>Precio</th>
20         <th>Stock</th>
21         <th>Categoría</th>
22         <th>Imagen</th>
23         <th>Acciones</th>
24       </tr>
25     </thead>
26     <tbody>
27       @foreach ($productos as $producto)
28         <tr>
29           <td>{{ $producto->nombre }}</td>
30           <td>{{ $producto->precio }} €</td>
31           <td>{{ $producto->stock }}</td>
32           <td>{{ $producto->categoria->nombre }}</td>
33           <td>
34             @if($producto->imagen)
35               nombre }}" width="50">
36             @else
37               No hay imagen
38             @endif
39           </td>
40           <td>
41             <a href="{{ route(name: 'productos.show', parameters: $producto->id) }}" class="btn btn-info btn-sm">Ver</a>
42             <a href="{{ route(name: 'productos.edit', parameters: $producto->id) }}" class="btn btn-primary btn-sm">Editar</a>
43             <form action="{{ route(name: 'productos.destroy', parameters: $producto->id) }}" method="POST" style="display:inline;">
44               @csrf
45               @method('DELETE')
46               <button type="submit" class="btn btn-danger btn-sm">Eliminar</button>
47             </form>
48           </td>
49         </tr>
50       @endforeach
51     </tbody>
52   </table>
53 </div>
54 @endsection
```


Vista detallada de productos:

```
show.blade.php X
resources > views > productos > show.blade.php
1 <!-- filepath: /c:/xampp/htdocs/proyecto_laravel/resources/views/productos/show.blade.php -->
2 @extends(view: 'layouts.app')
3
4 @section(section: 'content')
5 <div class="container">
6     <h1>Detalles del Producto</h1>
7     <div class="card mb-3">
8         <div class="row no-gutters">
9             <div class="col-md-4">
10                 @if($producto->imagen)
11                     nombre }}">
12                 @else
13                     
14                 @endif
15             </div>
16             <div class="col-md-8">
17                 <div class="card-body">
18                     <h5 class="card-title">{{ $producto->nombre }}</h5>
19                     <p class="card-text"><strong>Descripción:</strong> {{ $producto->descripcion }}</p>
20                     <p class="card-text"><strong>Precio:</strong> {{ $producto->precio }} €</p>
21                     <p class="card-text"><strong>Stock:</strong> {{ $producto->stock }}</p>
22                     <p class="card-text"><strong>Categoría:</strong> {{ $producto->categoria->nombre }}</p>
23                     <a href="{{ route(name: 'productos.edit', parameters: $producto->id) }}" class="btn btn-primary">Editar</a>
24                     <form action="{{ route(name: 'productos.destroy', parameters: $producto->id) }}" method="POST" style="display:inline;">
25                         @csrf
26                         @method('DELETE')
27                         <button type="submit" class="btn btn-danger">Eliminar</button>
28                     </form>
29                 </div>
30             </div>
31         </div>
32     </div>
33     <a href="{{ route(name: 'productos.index') }}" class="btn btn-secondary">Volver a la lista de productos</a>
34 </div>
35 @endsection
```

Vistas de categorías:

Create:

```
create.blade.php X
resources > views > categorias > create.blade.php
1 <!-- filepath: /c:/xampp/htdocs/proyecto_laravel/resources/views/categorias/create.blade.php -->
2 @extends(view: 'layouts.app')
3
4 @section(section: 'content')
5 <div class="container">
6     <h1>Crear Nueva Categoría</h1>
7     <form action="{{ route(name: 'categorias.store') }}" method="POST">
8         @csrf
9         <div class="form-group">
10             <label for="nombre">Nombre:</label>
11             <input type="text" name="nombre" id="nombre" class="form-control" value="{{ old(key: 'nombre') }}">
12             @error('nombre')
13                 <div class="text-danger">{{ $message }}</div>
14             @enderror
15         </div>
16         <button type="submit" class="btn btn-primary mt-3">Guardar</button>
17     </form>
18     <a href="{{ route(name: 'categorias.index') }}" class="btn btn-secondary mt-3">Volver a la lista de categorías</a>
19 </div>
20 @endsection
```

edit:

```
edit.blade.php X
resources > views > categorias > edit.blade.php
1 <!-- filepath: /c:/xampp/htdocs/proyecto_laravel/resources/views/categorias/edit.blade.php -->
2 @extends(view: 'layouts.app')
3
4 @section(section: 'content')
5 <div class="container">
6     <h1>Editar Categoría</h1>
7     <form action="{{ route(name: 'categorias.update', parameters: $categoria->id) }}" method="POST">
8         @csrf
9         @method('PUT')
10        <div class="form-group">
11            <label for="nombre">Nombre:</label>
12            <input type="text" name="nombre" id="nombre" class="form-control" value="{{ $categoria->nombre }}">
13            @error('nombre')
14                <div class="text-danger">{{ $message }}</div>
15            @enderror
16        </div>
17        <button type="submit" class="btn btn-primary mt-3">Actualizar</button>
18    </form>
19    <a href="{{ route(name: 'categorias.index') }}" class="btn btn-secondary mt-3">Volver a la lista de categorías</a>
20 </div>
21 @endsection
```

Vista general:

```
index.blade.php X
resources > views > categorias > index.blade.php
1 <!-- filepath: /c:/xampp/htdocs/proyecto_laravel/resources/views/categorias/index.blade.php -->
2 @extends(view: 'layouts.app')
3
4 @section(section: 'content')
5 <div class="container">
6     <h1>Lista de Categorías</h1>
7     <a href="{{ route(name: 'categorias.create') }}" class="btn btn-success mb-3">Crear Nueva Categoría</a>
8
9     @if(session(key: 'success'))
10        <div class="alert alert-success">
11            {{ session(key: 'success') }}
12        </div>
13    @endif
14
15    <table class="table">
16        <thead>
17            <tr>
18                <th>Nombre</th>
19                <th>Acciones</th>
20            </tr>
21        </thead>
22        <tbody>
23            @foreach ($categorias as $categoria)
24                <tr>
25                    <td>{{ $categoria->nombre }}</td>
26                    <td>
27                        <a href="{{ route(name: 'categorias.show', parameters: $categoria->id) }}" class="btn btn-info btn-sm">Ver</a>
28                        <a href="{{ route(name: 'categorias.edit', parameters: $categoria->id) }}" class="btn btn-primary btn-sm">Editar</a>
29                        <form action="{{ route(name: 'categorias.destroy', parameters: $categoria->id) }}" method="POST" style="display:inline;">
30                            @csrf
31                            @method('DELETE')
32                            <button type="submit" class="btn btn-danger btn-sm" onclick="return confirm('¿Estás seguro de que deseas eliminar esta categoría?')">Eliminar</button>
33                        </form>
34                    </td>
35                </tr>
36            @endforeach
37        </tbody>
38    </table>
39 </div>
40 @endsection
```

Vista en detalle:

```
show.blade.php X
resources > views > categorias > show.blade.php
1 <!-- filepath: /c:/xampp/htdocs/proyecto_laravel/resources/views/categorias/show.blade.php -->
2 @extends(view: 'layouts.app')
3
4 @section(section: 'content')
5 <div class="container">
6   <h1>Detalles de la Categoría</h1>
7   <div class="card">
8     <div class="card-body">
9       <h5 class="card-title">{{ $categoria->nombre }}</h5>
10      <a href="{{ route(name: 'categorias.index') }}" class="btn btn-secondary mt-3">Volver a la lista de categorías</a>
11    </div>
12  </div>
13 </div>
14 @endsection
```

Vista home:

```
home.blade.php X
resources > views > home.blade.php
1 <!-- filepath: /c:/xampp/htdocs/proyecto_laravel/resources/views/home.blade.php -->
2 @extends(view: 'layouts.app')
3
4 @section(section: 'content')
5 <br>
6 <div class="container">
7   <!-- Hero Section -->
8   <div class="jumbotron text-center text-black">
9     <h1 class="display-4">Bienvenido a FitMatch</h1>
10    <p class="lead">Encuentra los mejores suplementos y equipamiento deportivo para alcanzar tus metas.</p>
11  </div>
12 <br>
13 <br>
14 <!-- Products Section -->
15 <div class="row justify-content-center text-center">
16   <div class="col-md-4">
17     <div class="card mb-4 shadow-sm">
18       <div class="card-body">
19         <h5 class="card-title">Suplementos</h5>
20         <p class="card-text">Descubre nuestra amplia gama de suplementos para mejorar tu rendimiento.</p>
21         <a href="http://127.0.0.1:8000/productos" class="btn btn-primary">Ver Productos</a>
22       </div>
23     </div>
24   </div>
25   <div class="col-md-4">
26     <div class="card mb-4 shadow-sm">
27       <div class="card-body">
28         <h5 class="card-title">Ropa Deportiva</h5>
29         <p class="card-text">Explora nuestra colección de ropa deportiva para todos los deportes.</p>
30         <a href="http://127.0.0.1:8000/productos" class="btn btn-primary">Ver Productos</a>
31       </div>
32     </div>
33   </div>
34 </div>
35 <br>
36 <br>
37 <br>
38 <br>
39 <br>
40 <!-- Call to Action Section -->
41 <div class="row text-center mt-5">
42   <div class="col-md-12">
43     <h2>Únete a FitMatch Hoy</h2>
44     <p>Regístrate ahora y comienza tu viaje hacia una vida más saludable y activa.</p>
45     <a href="{{ route(name: 'register') }}" class="btn btn-primary btn-lg">Regístrate</a>
46   </div>
47 </div>
48 </div>
49 @endsection
```

En las vistas uso la función route() para las rutas.

Instalación de laravel en español:

en la carpeta del proyecto:

composer require laravel/lang

php artisan vendor:publish --tag=lang

en config/app.php se cambia: en → es:

```
/*
|-----
| Application Locale Configuration
|-----
|
| The application locale determines the default locale that will be used
| by the translation service provider. You are free to set this value
| to any of the locales which will be supported by the application.
|
*/

'locale' => 'es',
```

Futuras expansiones:

- Gestión de roles.
- Vista de catalogo para usuarios.
- Que los usuarios puedan realizar pedidos.
- Configurar pasarelas de pago.

En la base de datos ya existe la tabla pedidos con sus correspondientes campos.

Hecho por: Marcos González Ferreira