" *Año de la recuperación y consolidación de la economía peruana".*

**INSTITUTO DE EDUCACIÓN SUPERIOR TECNOLÓGICO PÚBLICO**

**“PEDRO P. DIAZ”**

PROGRAMA DE ESTUDIOS

**DESARROLLO DE SISTEMAS DE INFORMACIÓN**

**PROYECTO APLICATIVO INTEGRADOR**

**SISTEMA DE FACTURACIÓN Y CONTROL DE INVENTARIOS**

**UNIDAD DIDÁCTICA**: Desarrollo Web Integrado

**ALUMNO(S):** Leonardo Fabio Choque Sucasaca

Ariana Massiel Garcia Jara

Mirian Gladys Huaynapata Saime

Alexander Edgard Manuel Linares Condori

**DOCENTE:** Ing. Duilio Andre Chavez Cuarite

**TURNO:** Diurno **SEMESTRE: IV**

**Arequipa – Perú**

**2025**

**Proyecto Aplicativo Integrador Sistema de Facturación y control de Inventarios**

1. **INTRODUCCIÓN**

En el entorno empresarial actual, la eficiencia en la gestión de recursos es fundamental para la competitividad. El presente proyecto consiste en el desarrollo de una solución web integral para la "Gestión de Facturación y Control de Inventarios", diseñada para optimizar los procesos de compra, venta y almacenamiento de productos. Utilizando el framework **Laravel**, se ha construido una plataforma robusta que garantiza la integridad de los datos, la automatización de cálculos de impuestos y un control preciso del stock en tiempo real, facilitando la toma de decisiones basada en información actualizada.

1. **OBJETIVOS**

**2.1. Objetivo Principal**

* Diseñar e implementar una solución web empresarial que permita automatizar el proceso de facturación y el control riguroso de inventarios, aplicando el framework Laravel y el motor de base de datos MySQL para garantizar un sistema escalable y eficiente.

**2.2. Objetivos Específicos**

* Desarrollar un módulo de gestión de terceros que permita el registro, búsqueda y actualización de clientes y proveedores.
* Implementar un sistema de control de inventarios que actualice automáticamente el stock de artículos tras cada proceso de facturación o devolución.
* Automatizar el proceso de venta mediante un generador de facturas que calcule valores totales e impuestos de forma precisa.
* Establecer un sistema de seguridad y autenticación para restringir el acceso a las funciones administrativas solo a usuarios autorizados.

1. **ENLACES**

* Repositorio GitHub: <https://github.com/JucuOwl/Sistema_facturacion>
* Youtube:
* Dominio: <https://sisfacturacion002.ct.ws/?i=1>

1. **ALCANCES**

El sistema cubrirá las necesidades operativas de un almacén o empresa comercial, incluyendo:

* **Gestión de Datos Maestros:** Administración de ciudades, tipos de documentos y formas de pago.
* **Operaciones Comerciales:** Registro de compras (proveedores) y ventas (clientes).
* **Documentación Legal:** Emisión de facturas y detalles de facturación conforme a los requerimientos del negocio.
* **Logística Inversa:** Registro de devoluciones de mercancía para el ajuste manual o automático de inventarios.
* **Compatibilidad:** Acceso multiplataforma a través de navegadores web gracias a su diseño responsivo.

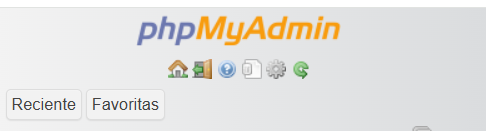
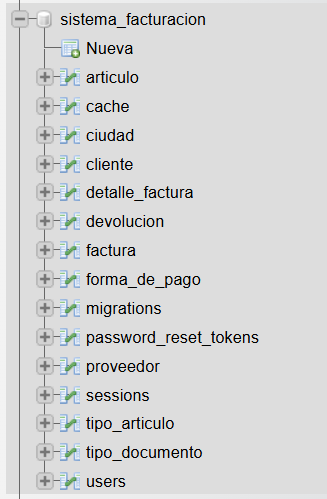
1. **CARACTERÍSTICAS DEL SISTEMA**

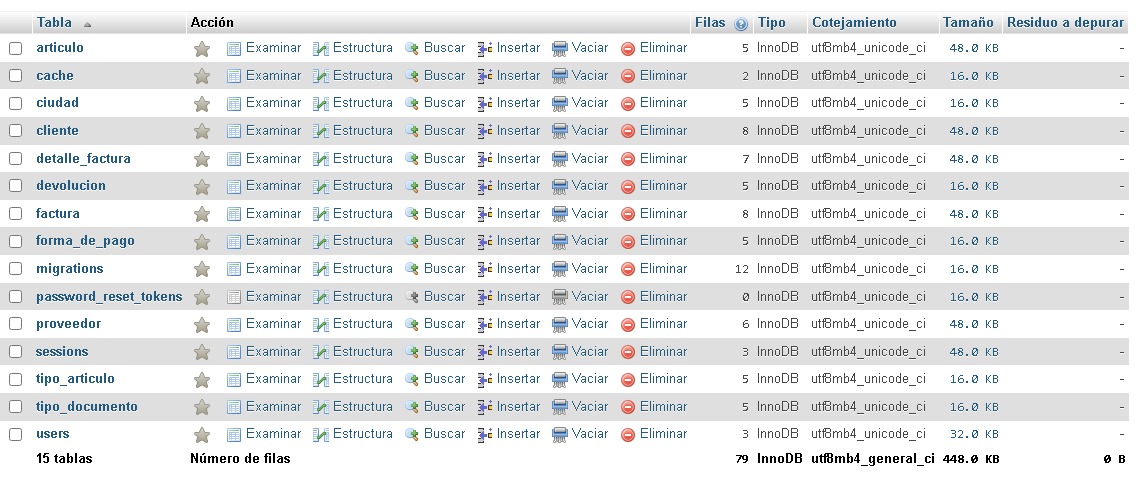
* **Arquitectura MVC:** Desarrollo organizado bajo el patrón Modelo-Vista-Controlador de Laravel.
* **Interfaz Intuitiva:** Diseño limpio y funcional enfocado en la usabilidad del usuario final.
* **Cálculos Automatizados:** Procesamiento automático de subtotales, IGV/IVA y totales en facturas.
* **Seguridad:** Encriptación de contraseñas y validación de sesiones mediante middleware.
* **Persistencia de Datos:** Base de datos relacional optimizada con migraciones de Laravel para asegurar la consistencia del esquema.

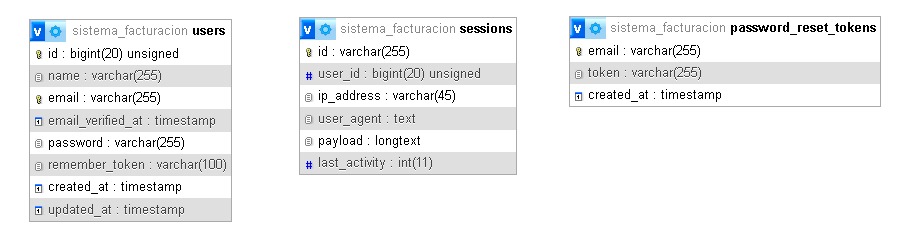
1. **REQUERIMIENTO DE LA APLICACIÓN:**

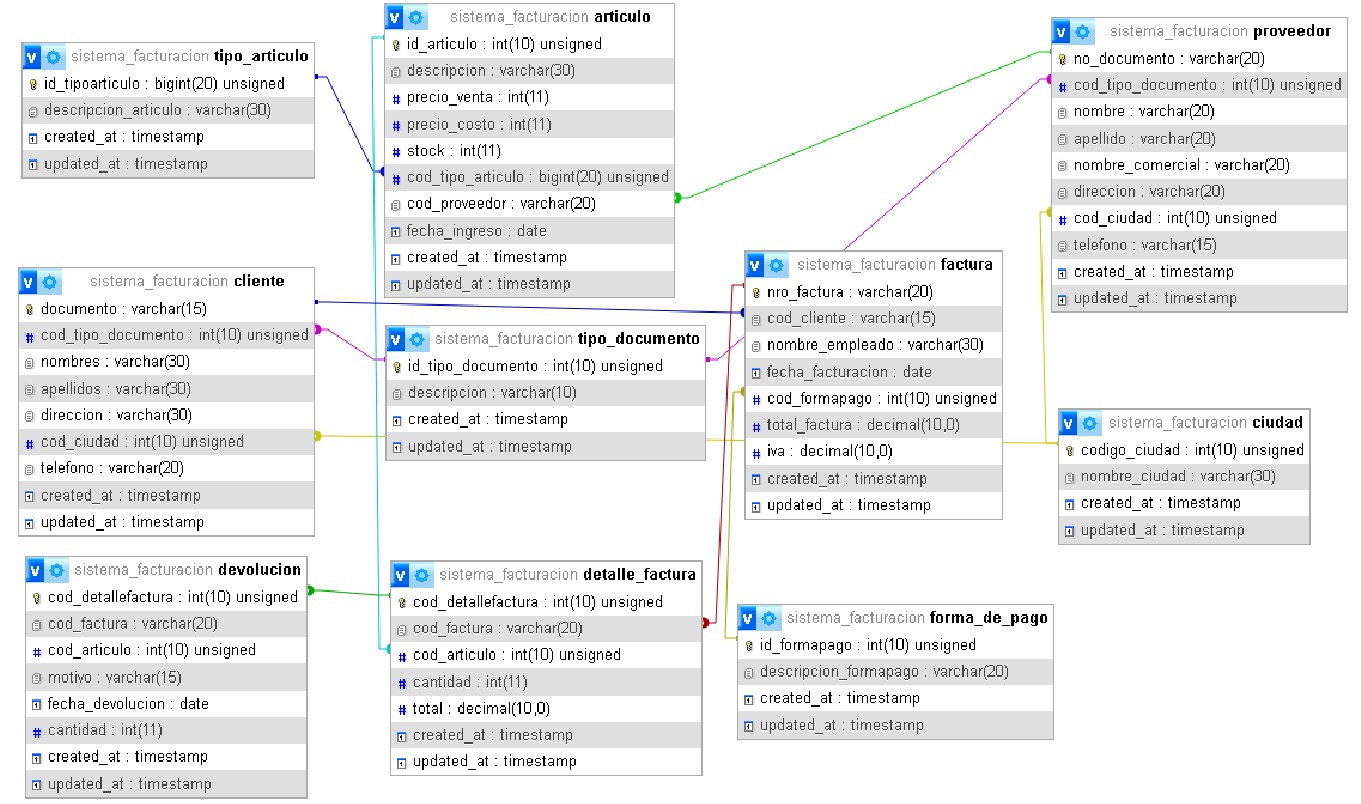
Este sistema de facturación y control de inventarios debe desarrollarse en el framework laravel y la base de datos Mysql y debe tener las siguientes características • Registro de clientes • Búsqueda de clientes • Registro de artículos • lista de artículos • Actualizar Stock artículos • Devoluciones • Realizar venta (facturar) • Consultas • Registro de proveedores • Lista de proveedores

1. **DISEÑO DE LA BASE DE DATOS**

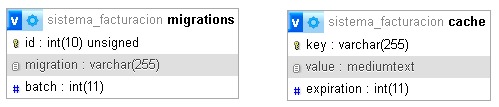
La base de datos debe estar construida en el gestor de base de datos MySQL, el cual es muy estable y robusto, por tal motivo para su uso deben tener instalado el paquete XAMPP, el cual contiene un servidor apache y el gestor de base de datos ya mencionado. La Base de Datos es relacional y consta de 15 tablas que mantienen la integridad referencial y la aplicación de las formas normales mínimas requeridas para un diseño de una base de datos; a continuación,se presenta el diagrama entidad relación:

******BASE DE DATOS NOMBRE: SISTEMA\_FACTURACION**

**MÓDULO DE AUTENTICACIÓN Y SESIÓN**

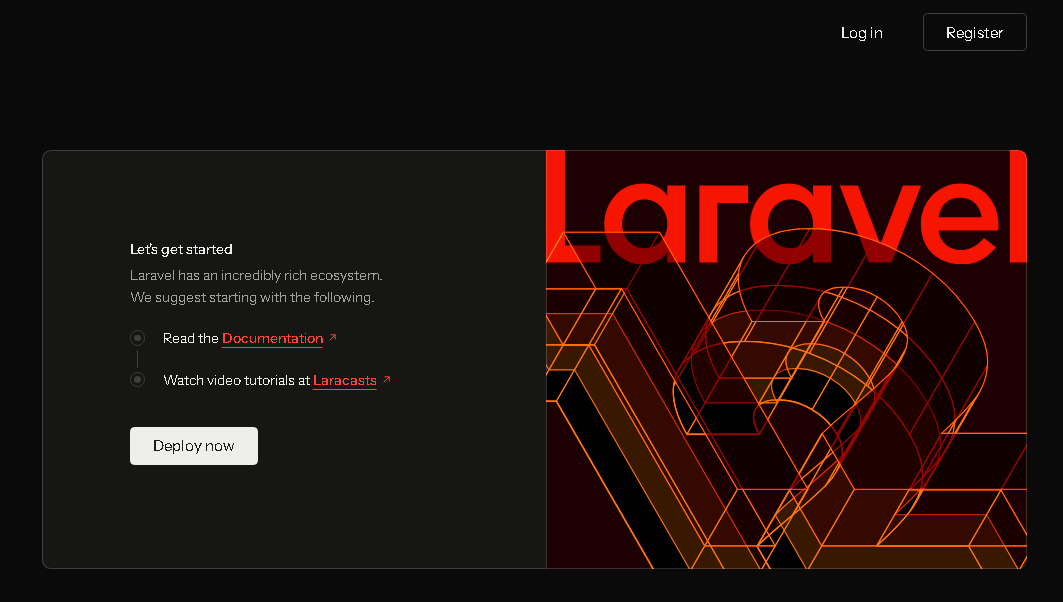
**MÓDULO DE FACTURACIÓN**

**TABLAS INTERNAS DEL SISTEMA**

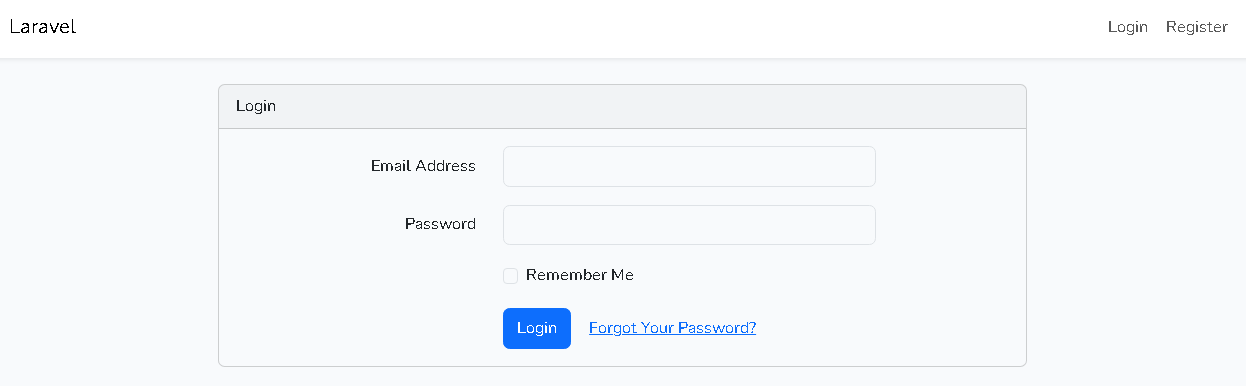


**PANTALLAS EN LARAVEL FRONTEND VISTA A VISTA**

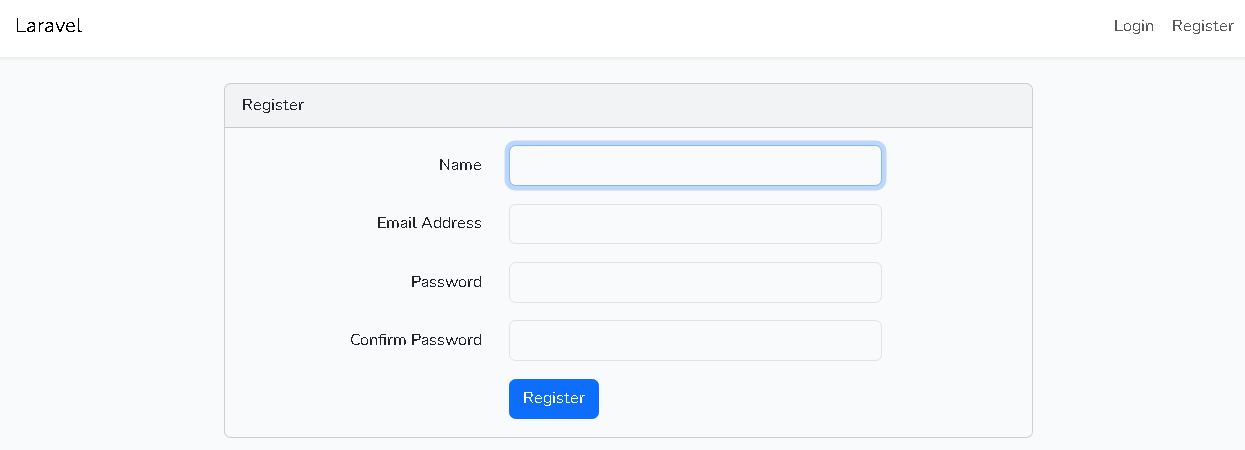
**INICIO**



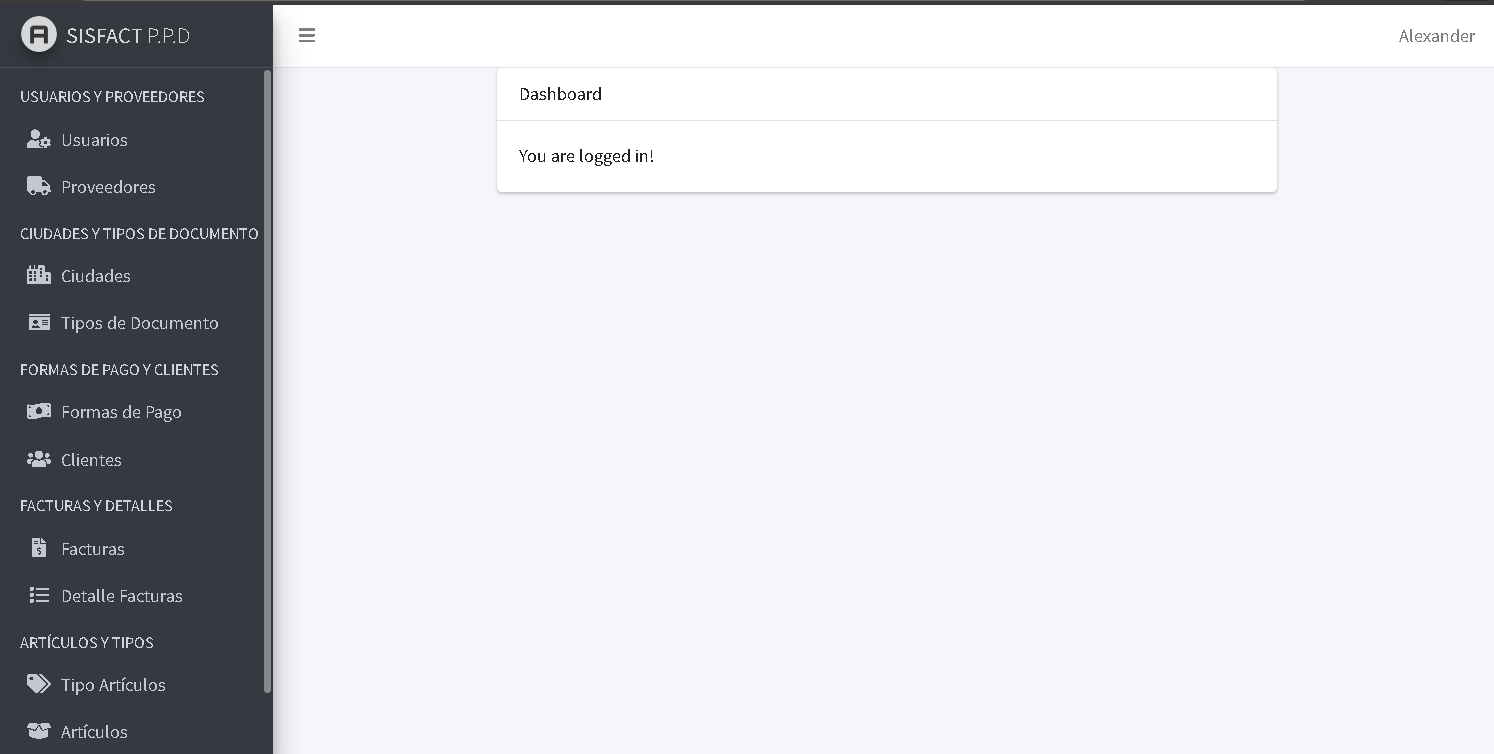
**LOGIN**



**REGISTER**

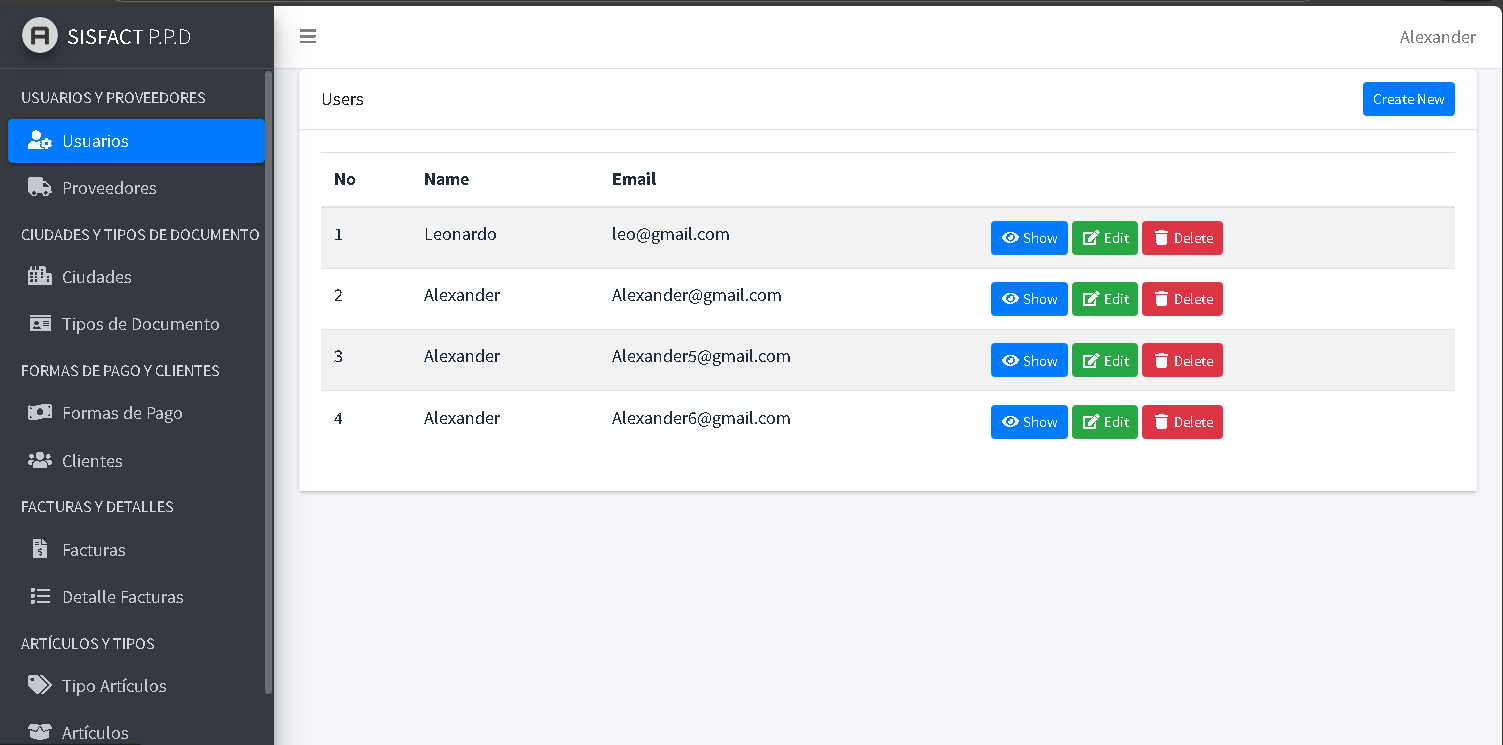


**HOME**

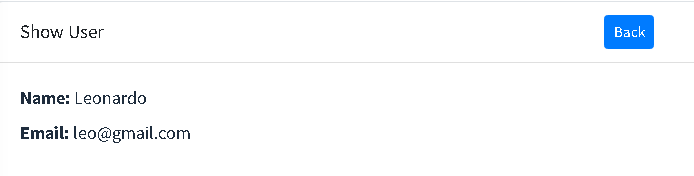


**USUARIOS Y PROVEEDORES**

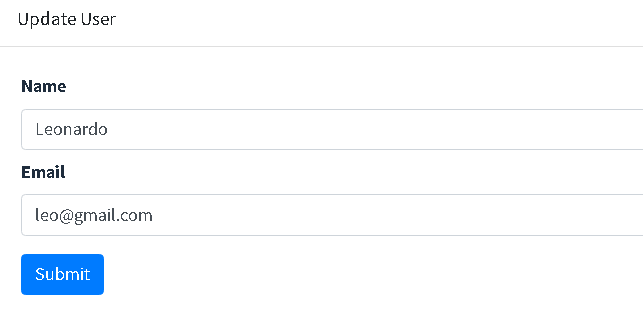
* **Usuarios**



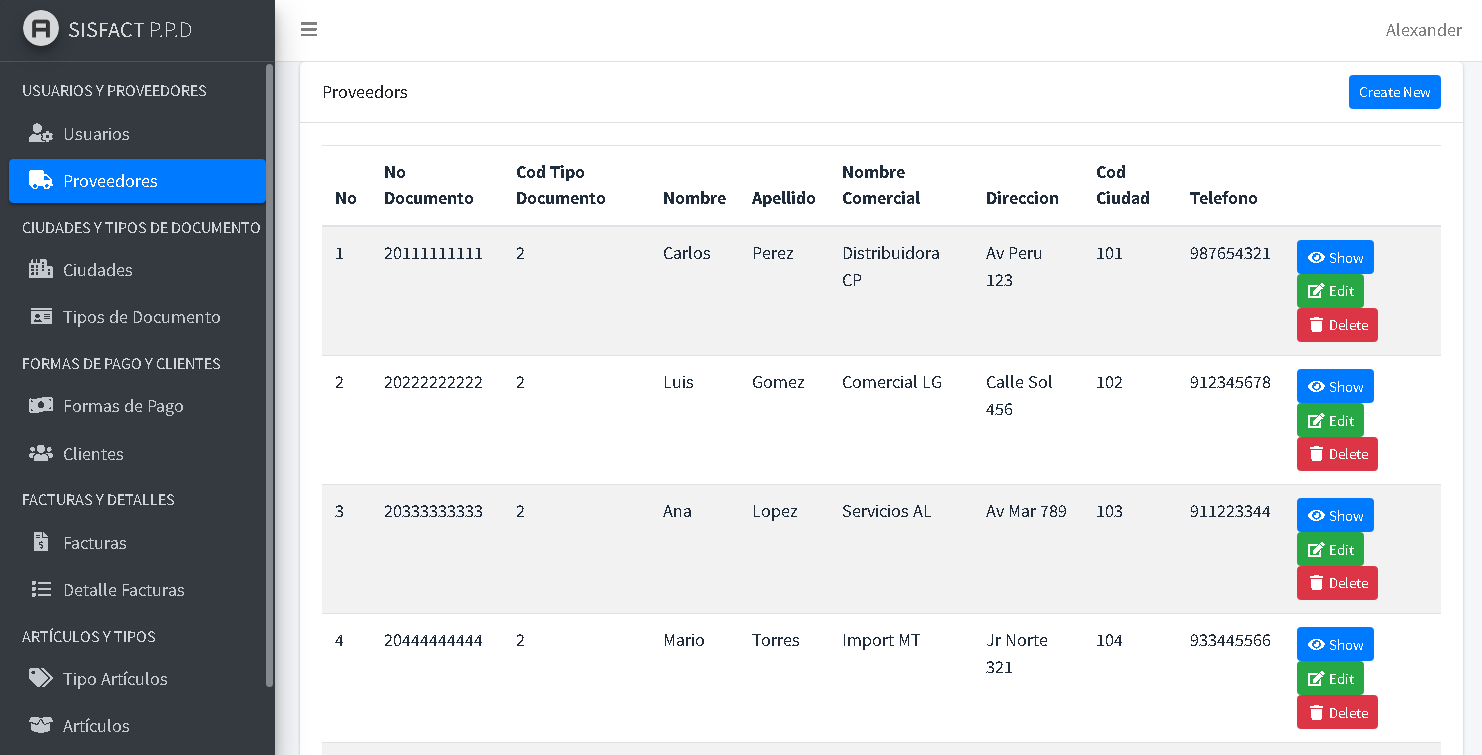
* **Show User**



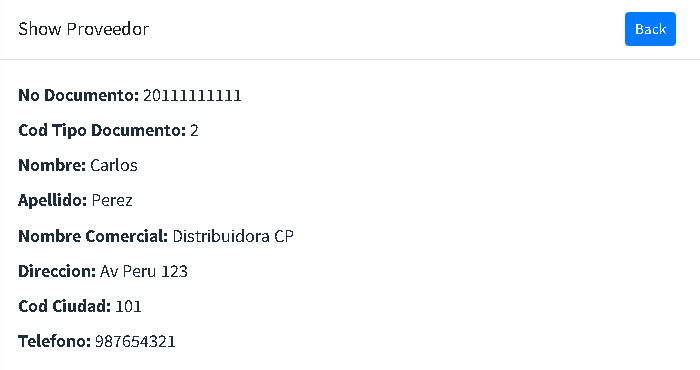
* **Update User**



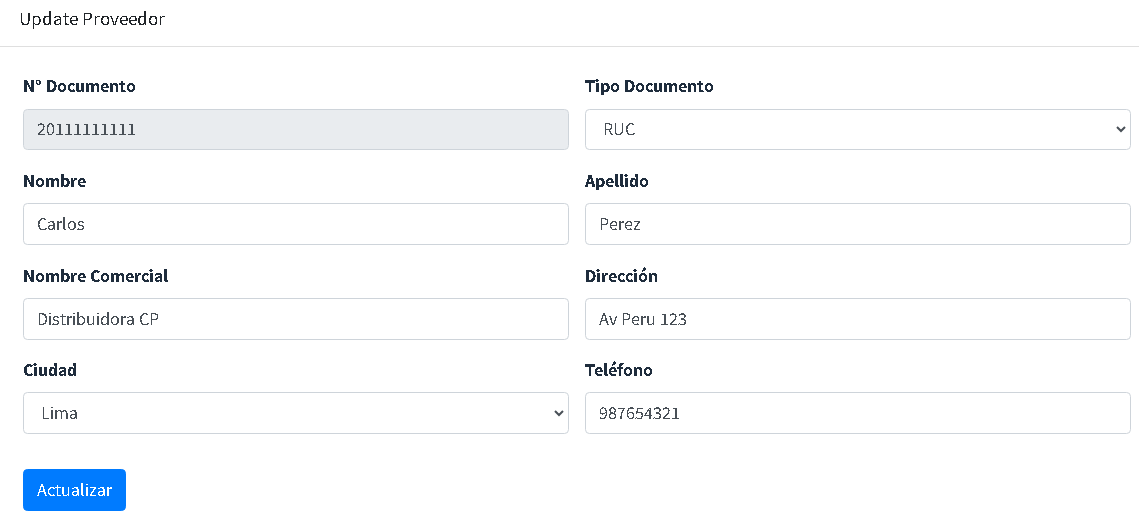
* **Proveedores**



* **Show Proveedor**

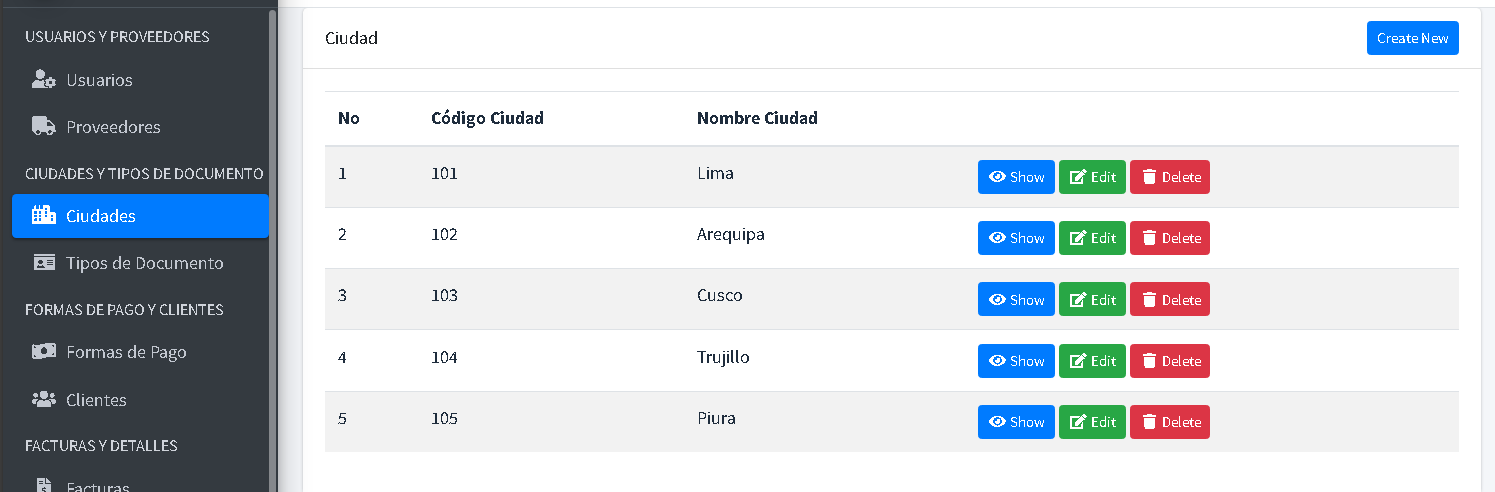


* **Update Proveedor**

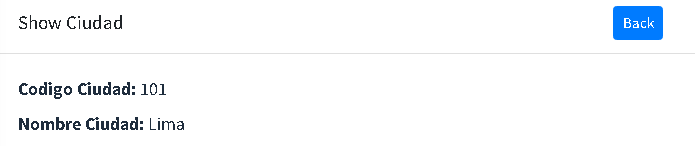


**CIUDADES Y TIPOS DE DOCUMENTO**

* **Ciudades**



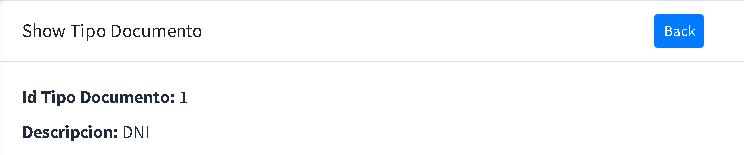
* **Show Ciudad**



* **Tipo de documento**



* **Show Tipo Documento**

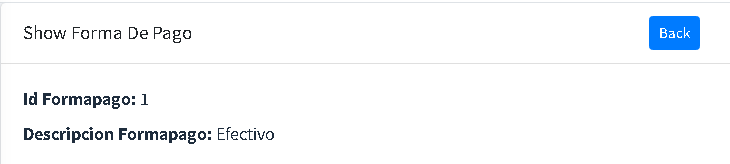


**FORMAS DE PAGO Y CLIENTES**

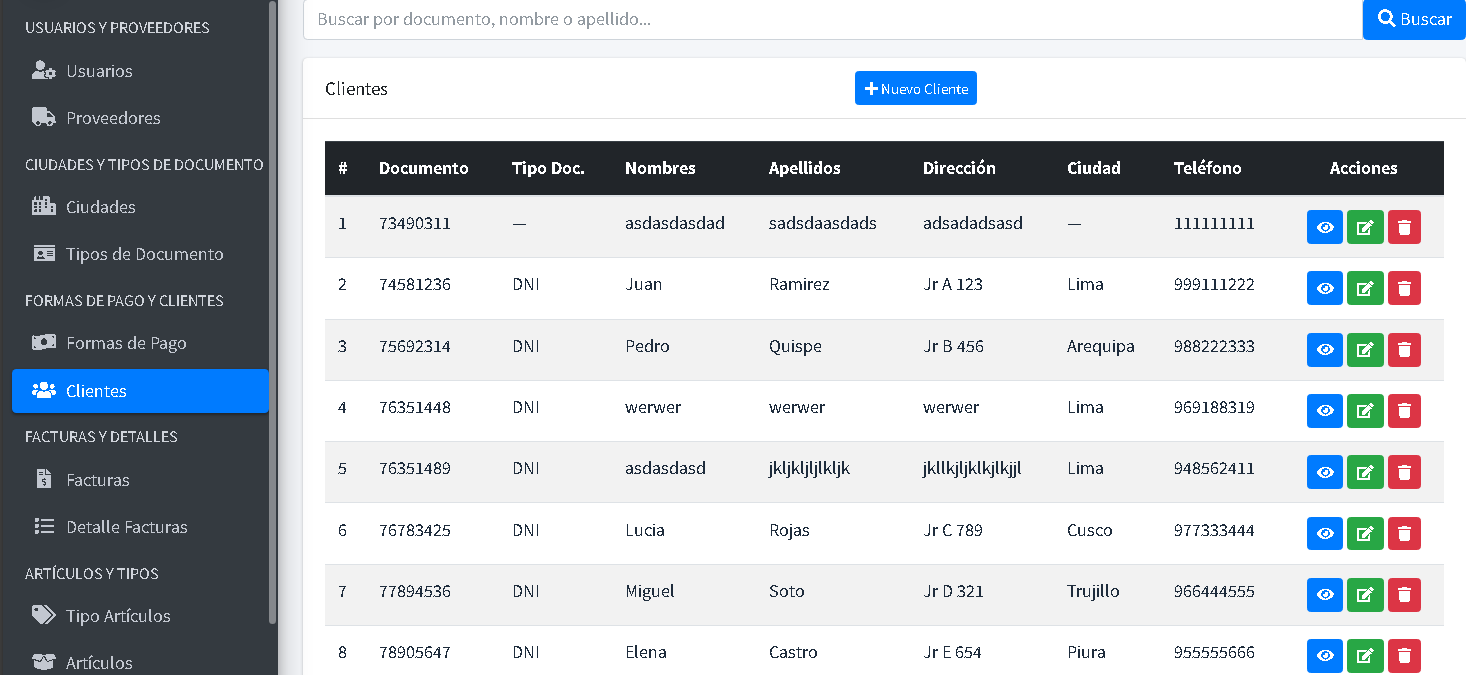
* **Formas de pago**



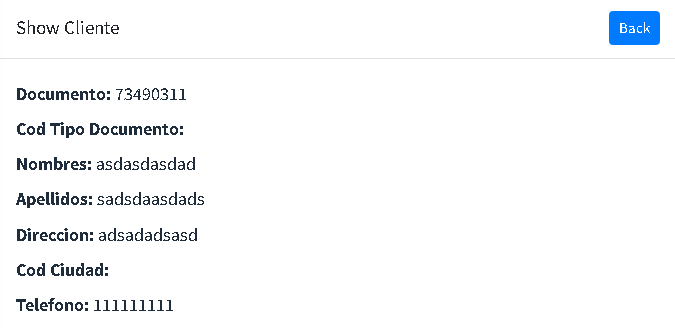
* **Show Forma De Pago**



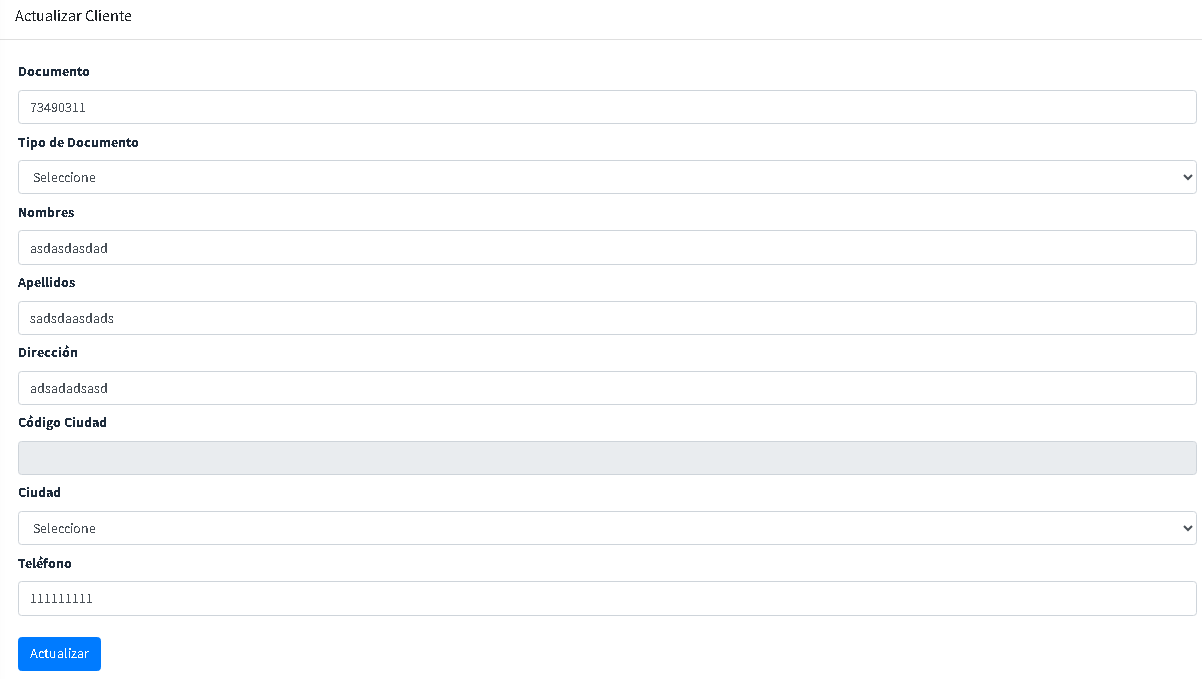
* **Clientes**



* **Show Cliente**



* **Actualizar Cliente**

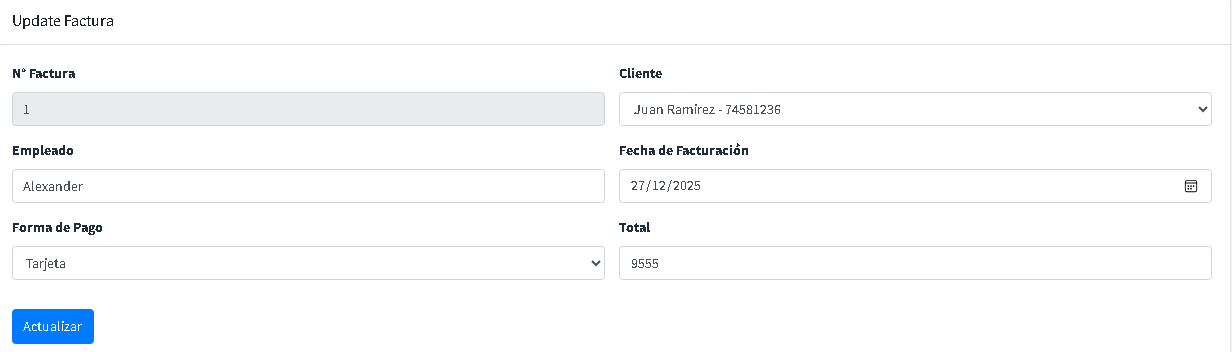


**FACTURAS Y DETALLES**

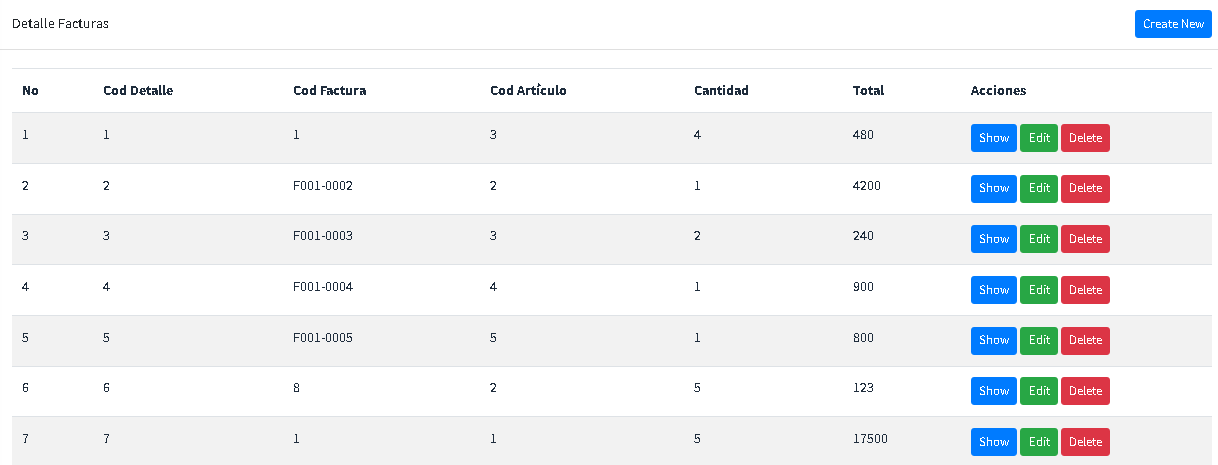
* **Facturas**



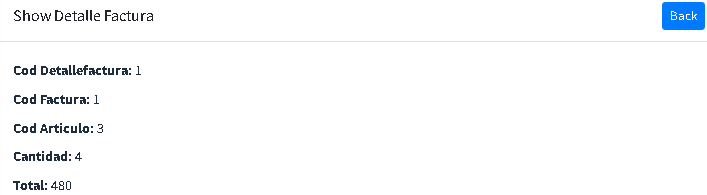
* **Update Factura**



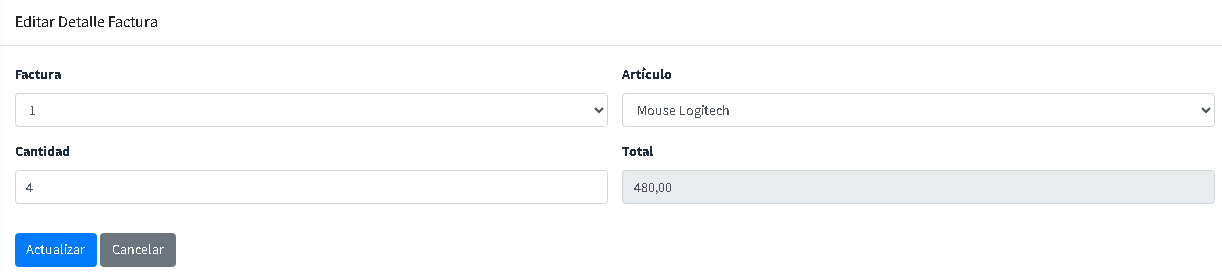
* **Detalle Facturas**

****

* **Show Detalle Factura**



* **Editar Detalle Factura**

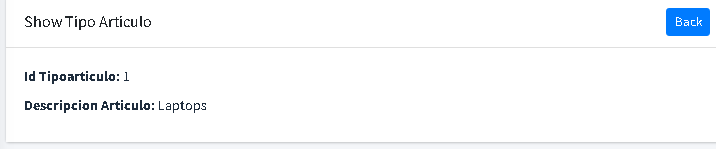


**ARTÍCULOS Y TIPOS**

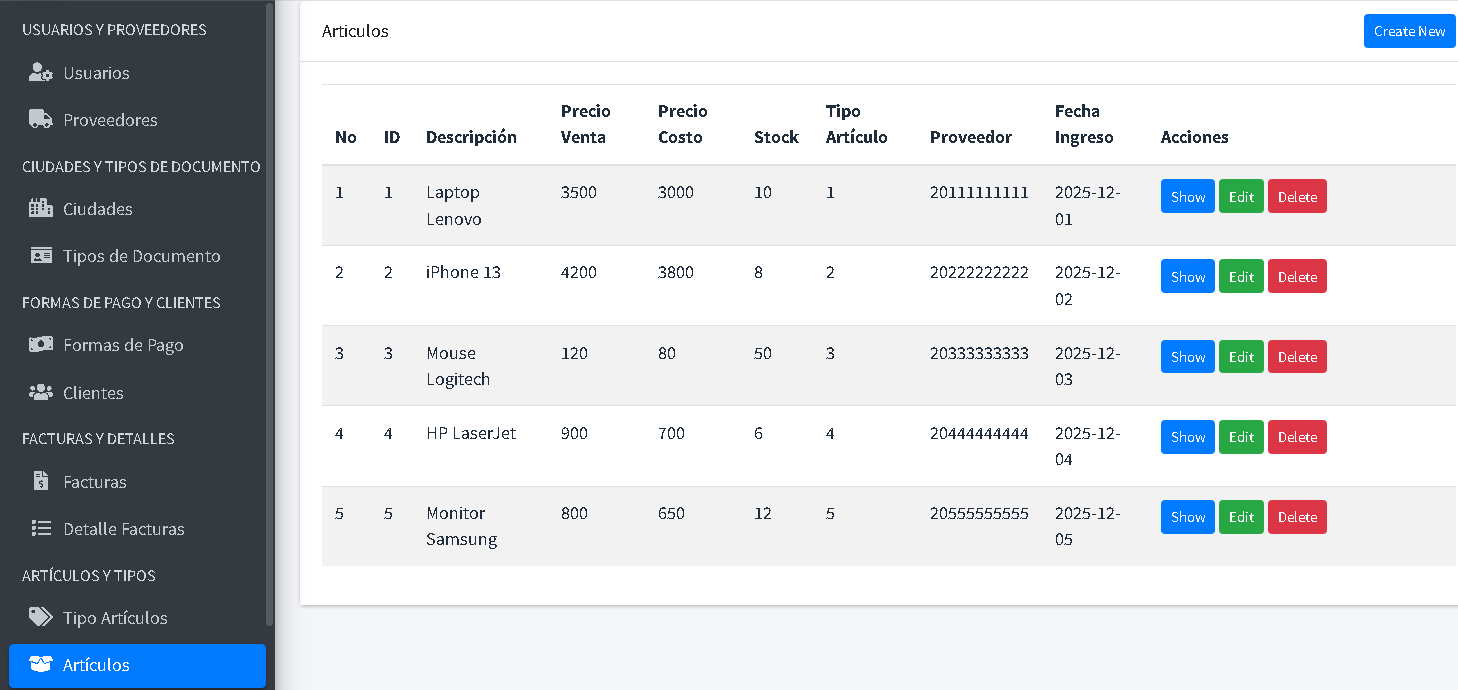
* **Tipo artículos**



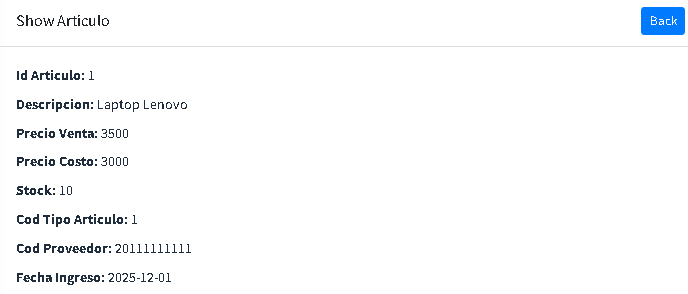
* **Show Tipo Articulo**



* **Artículos**

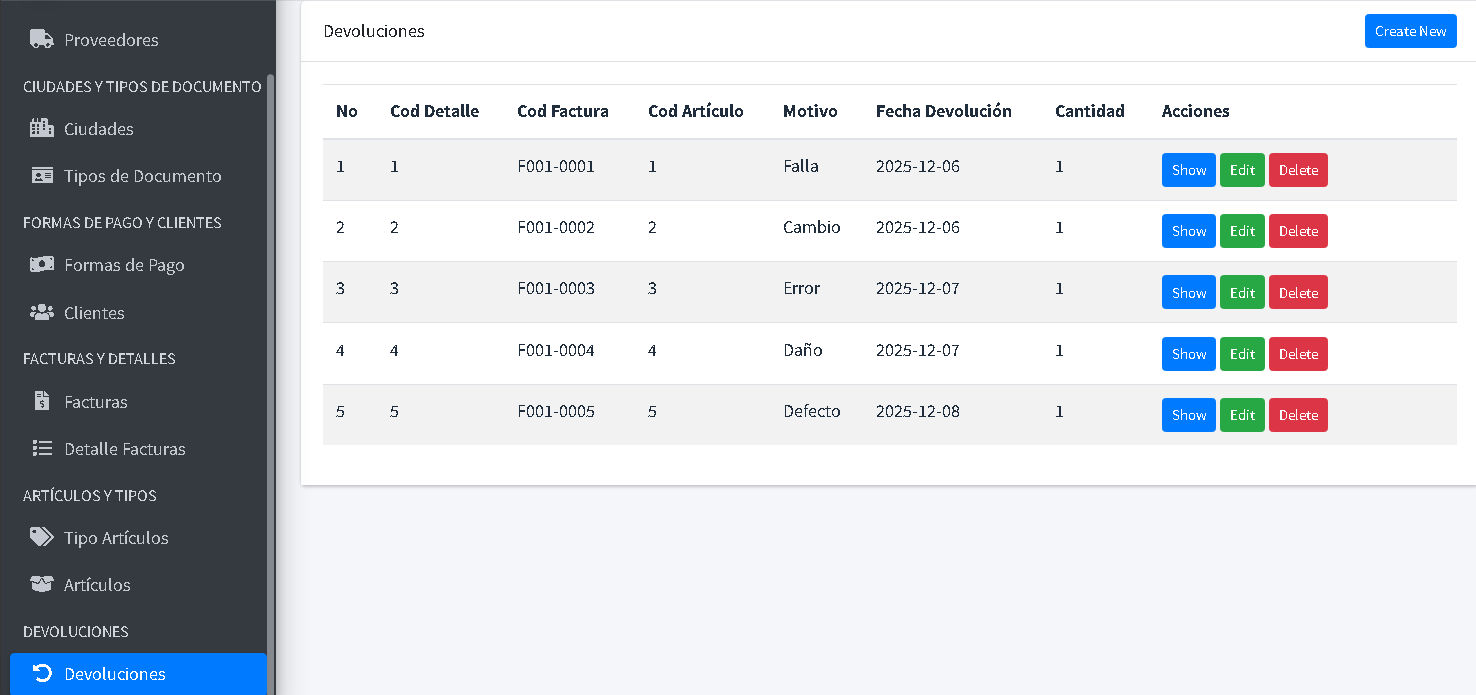


* **Show Articulo**

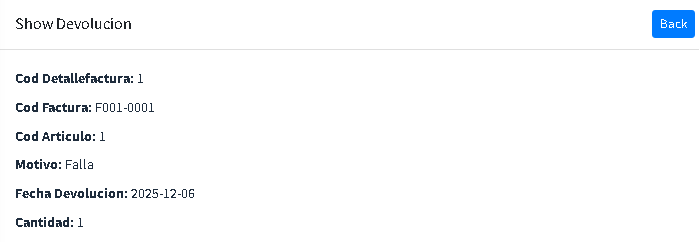


**DEVOLUCIONES**

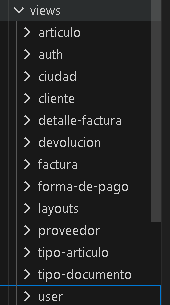
* **Devoluciones**



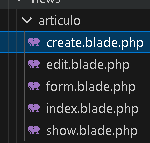
* **Show Devolucion**

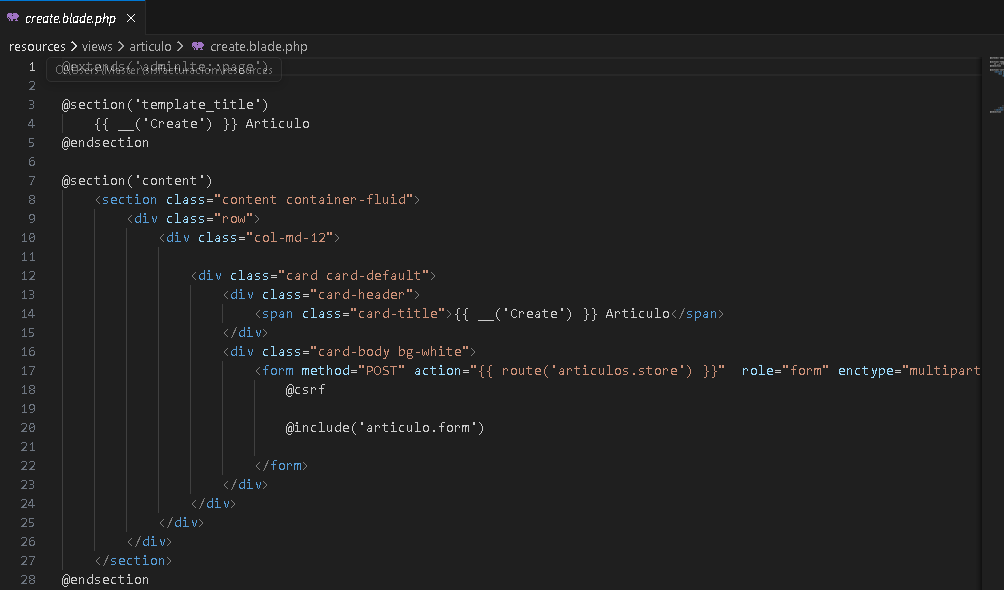


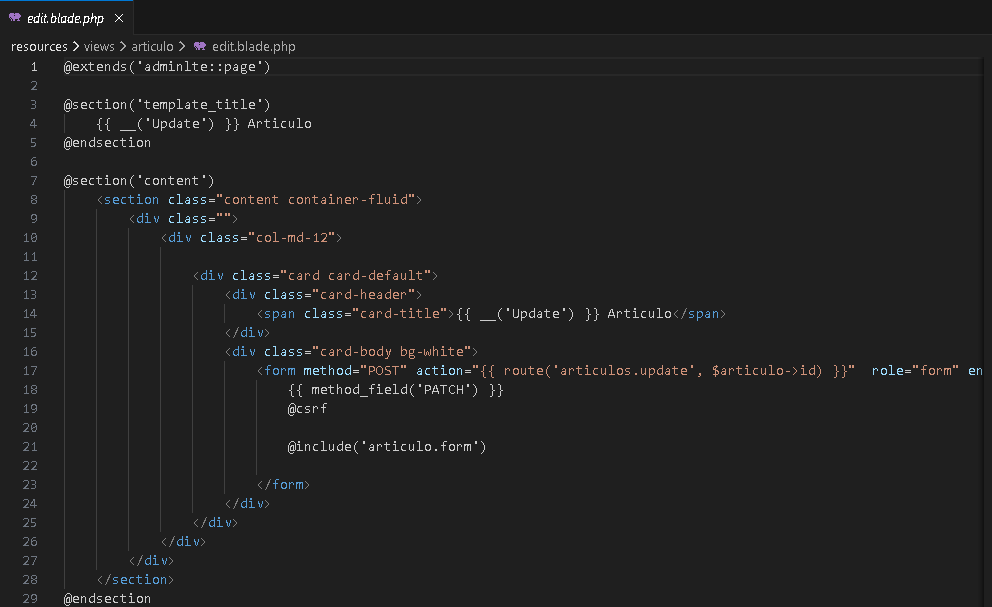
**Código Fuente de la aplicación (Front End-Vistas)**

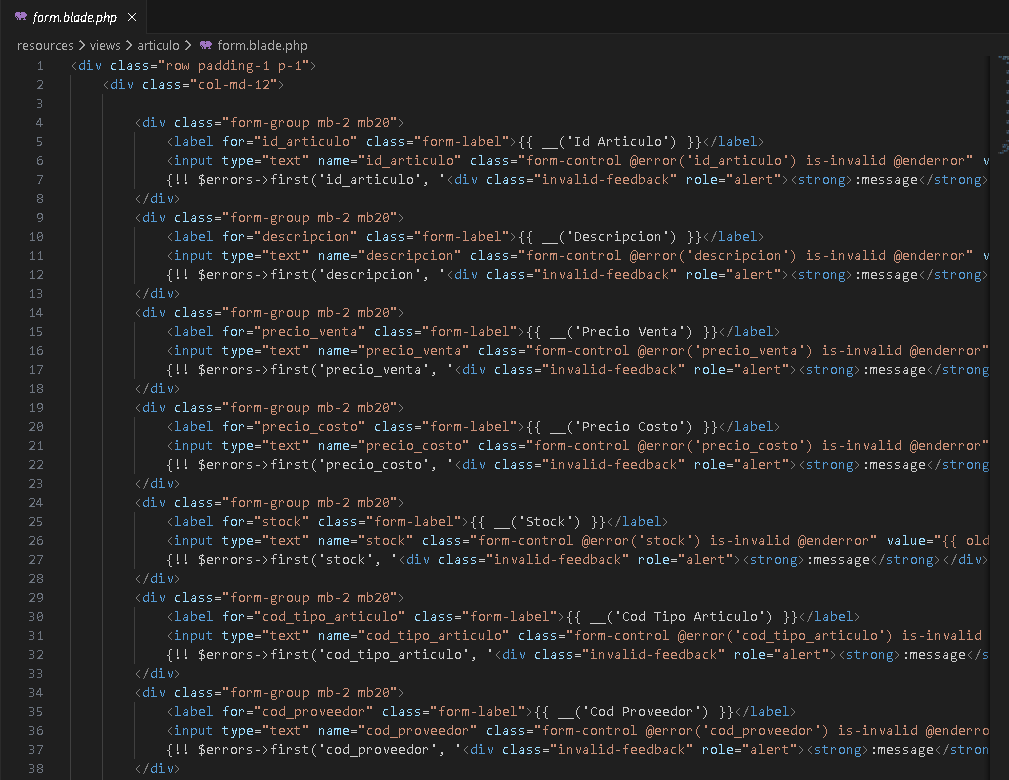
****

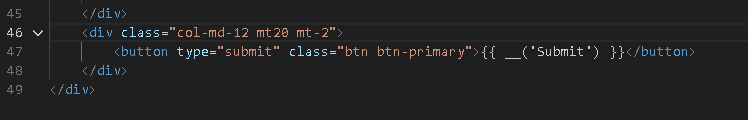
* **Articulo**

****

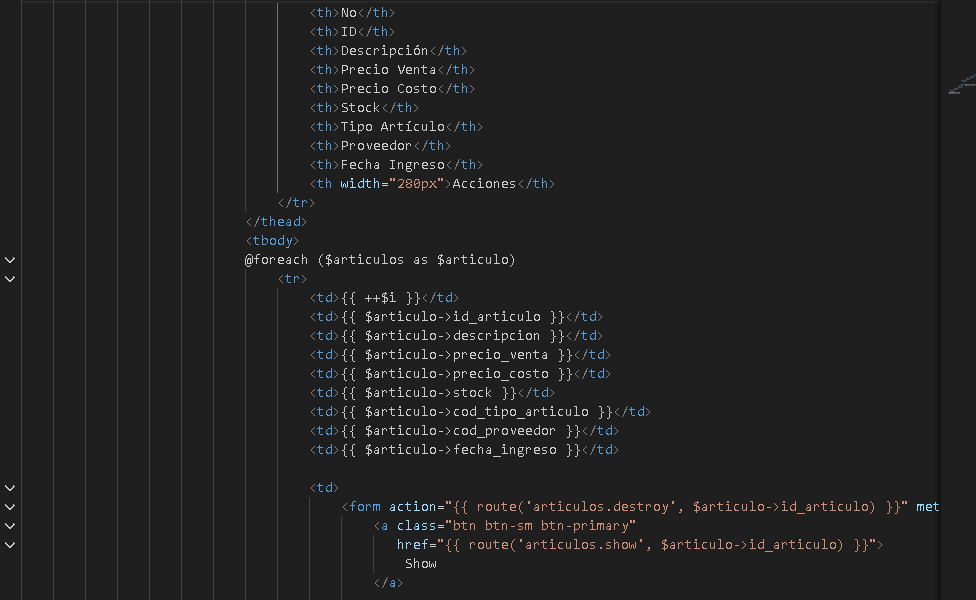
****

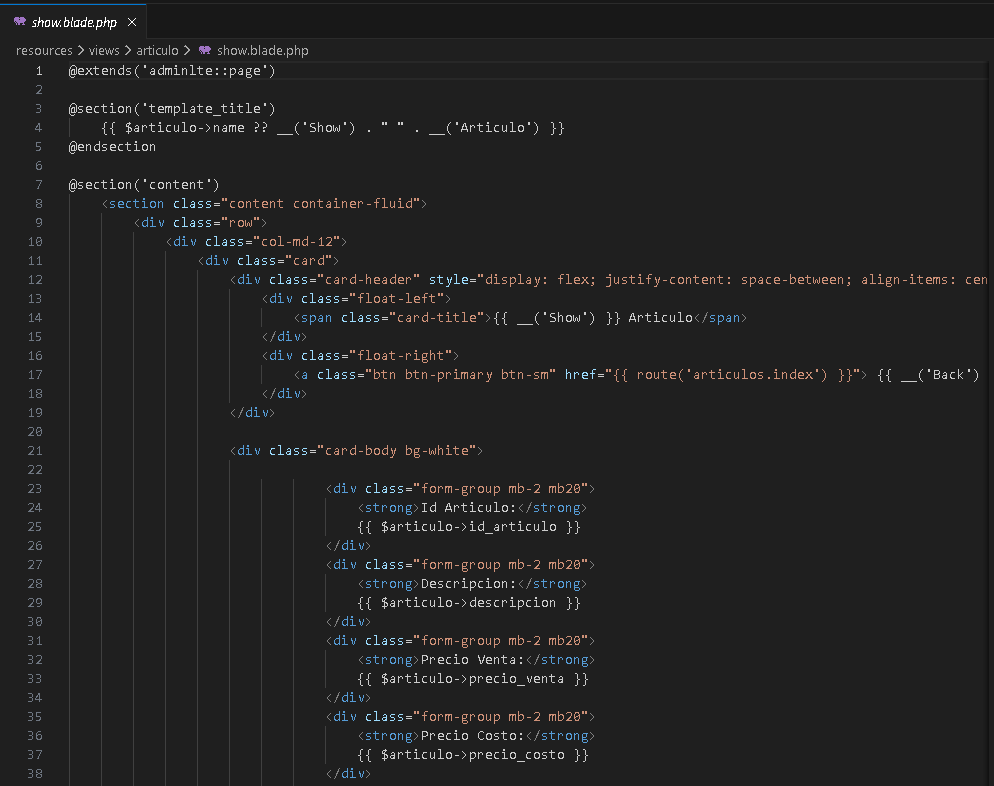
****

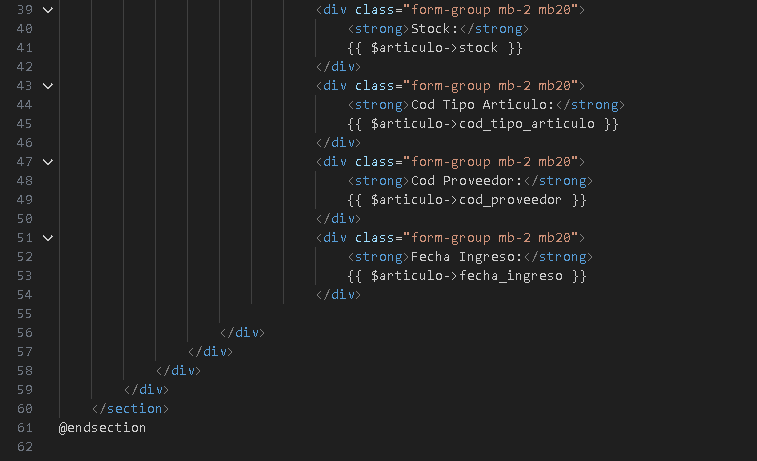
****

****

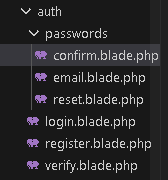
****

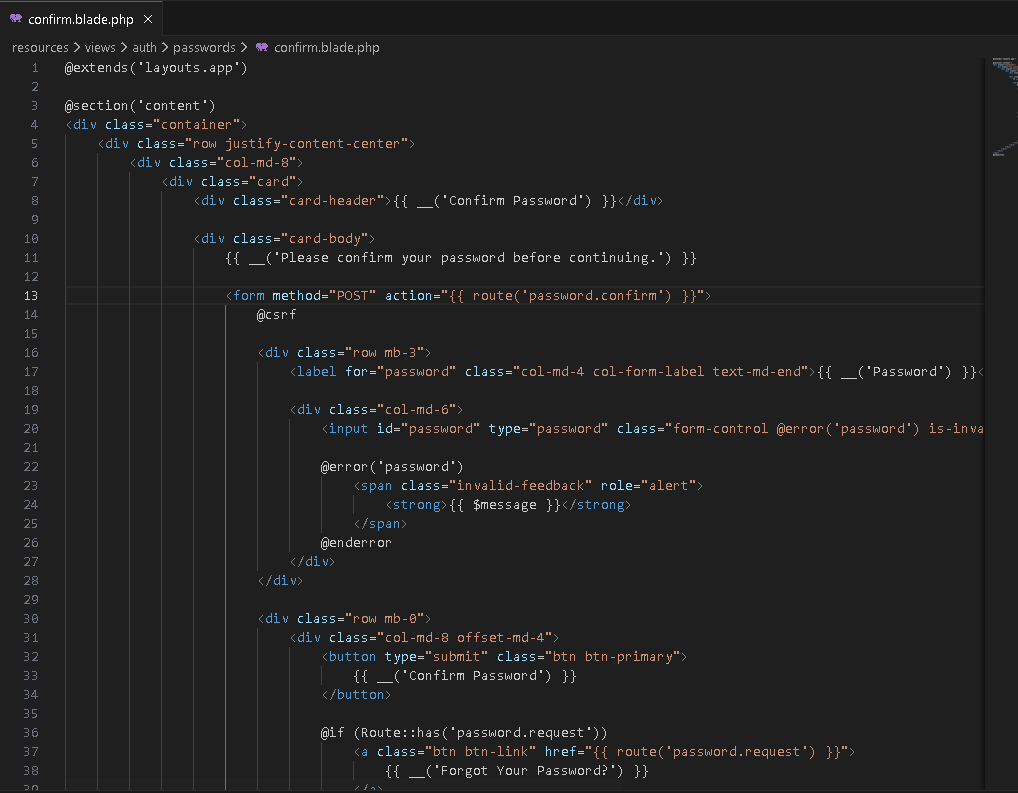
****

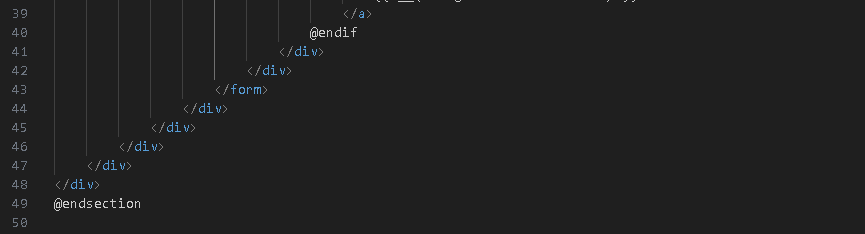
****

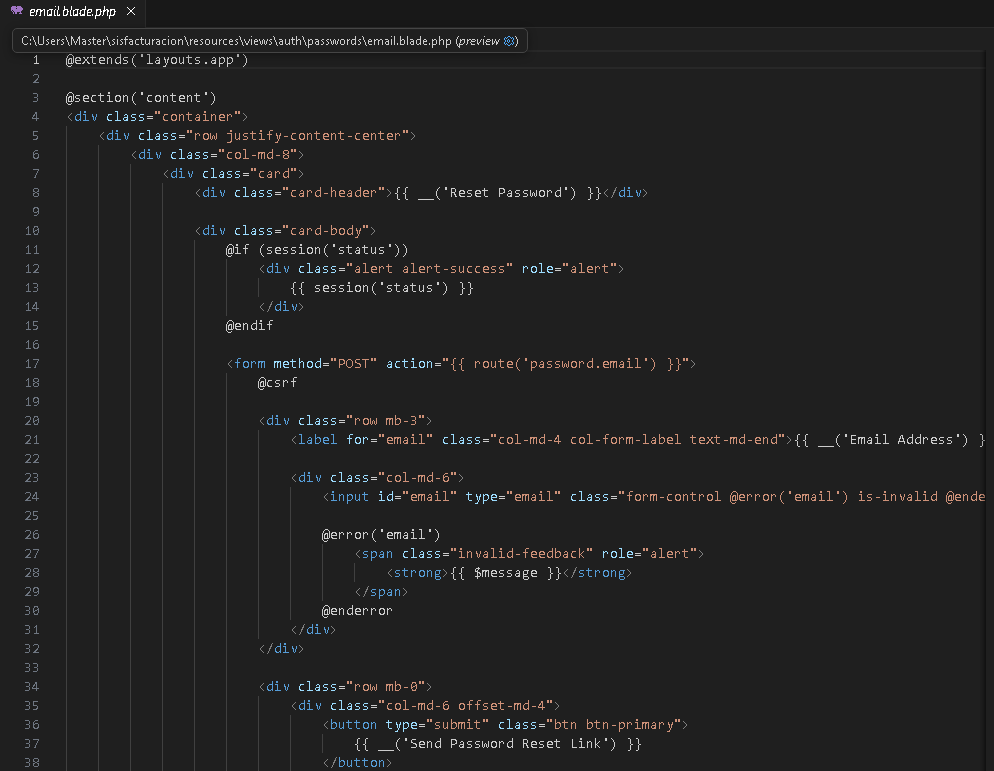
****

* **Auth**

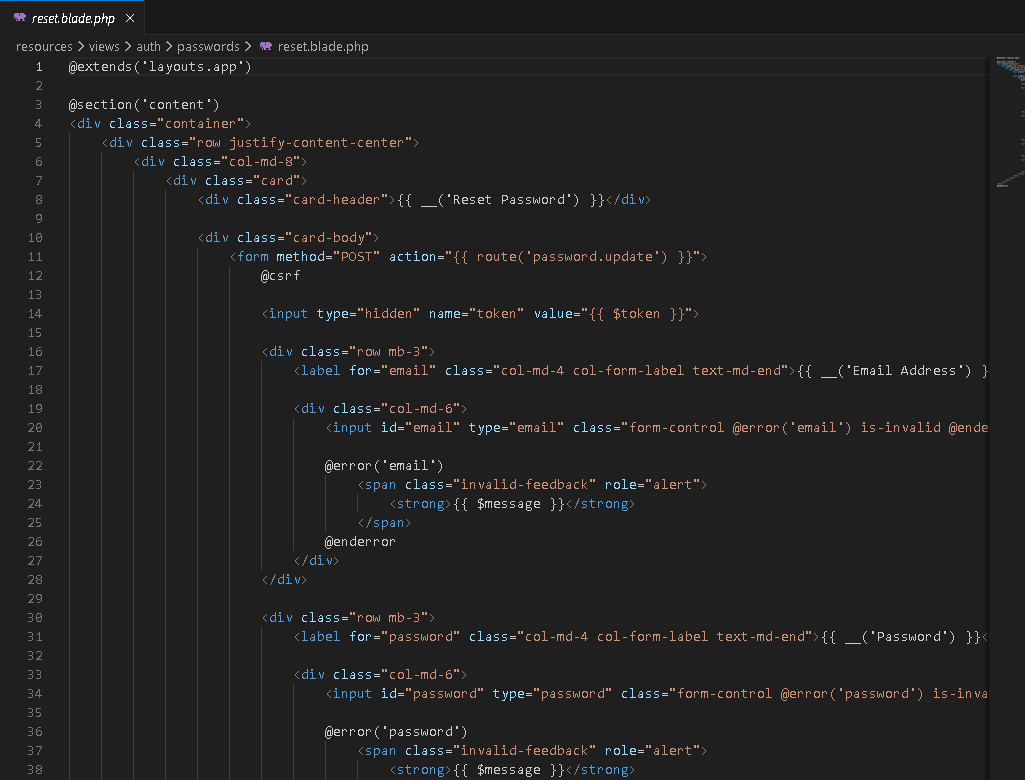
****

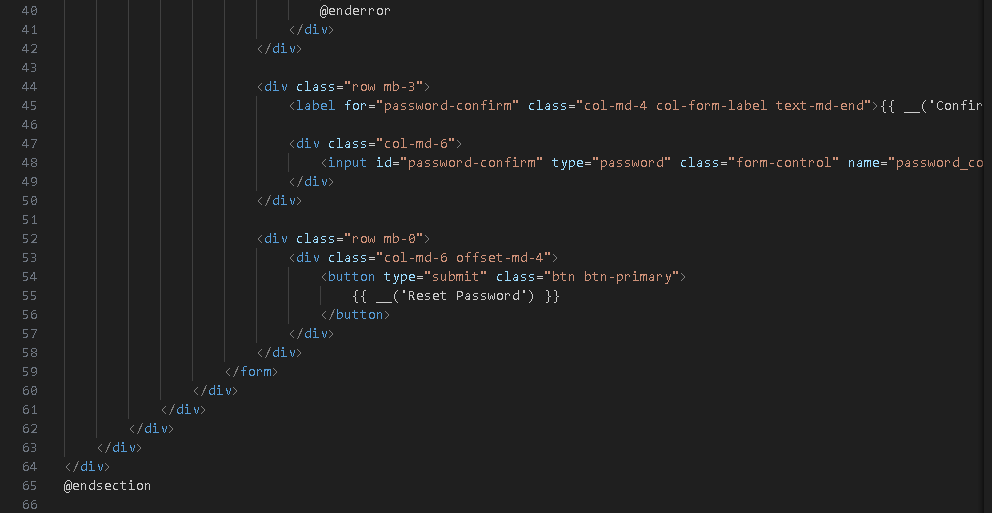
****

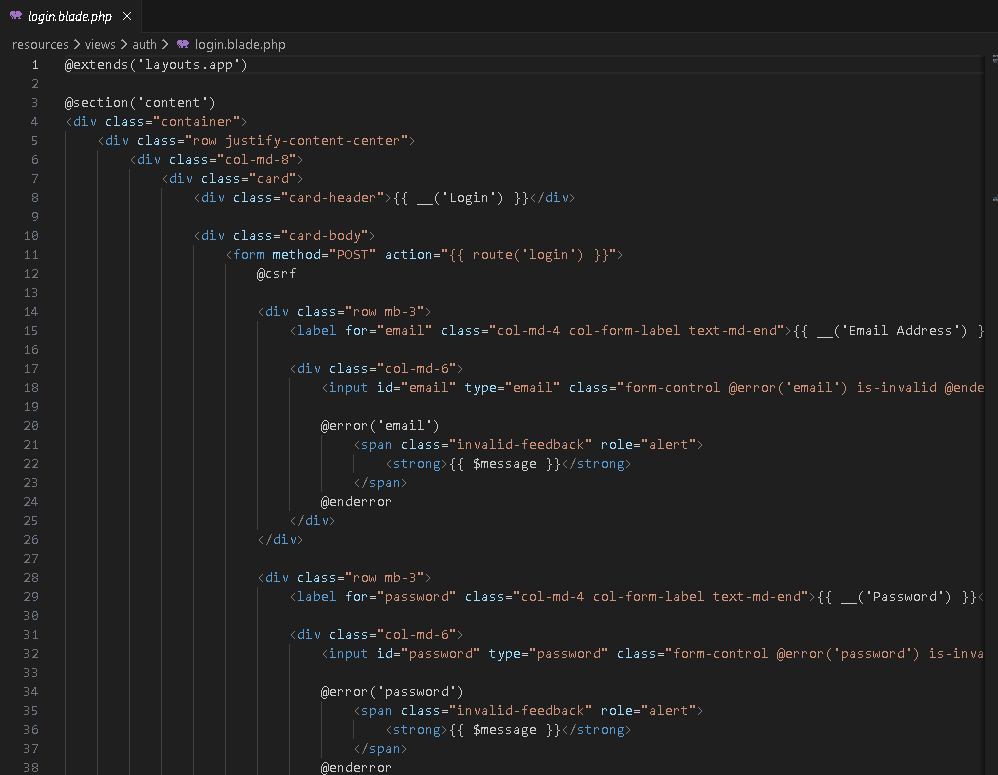
****

****

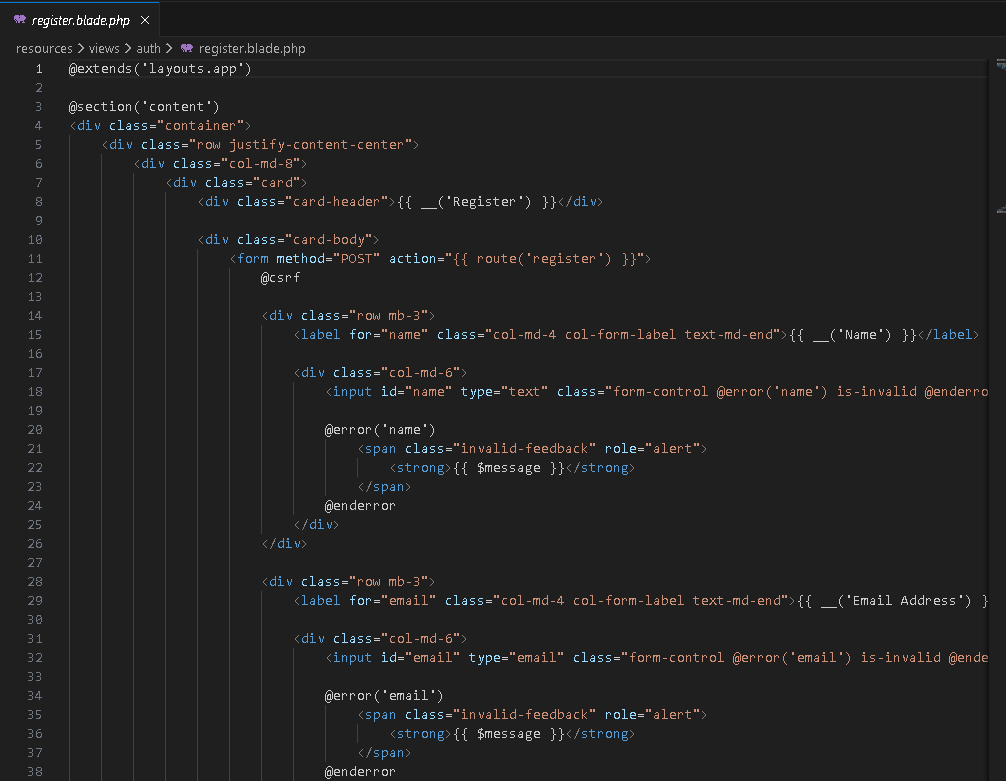
****





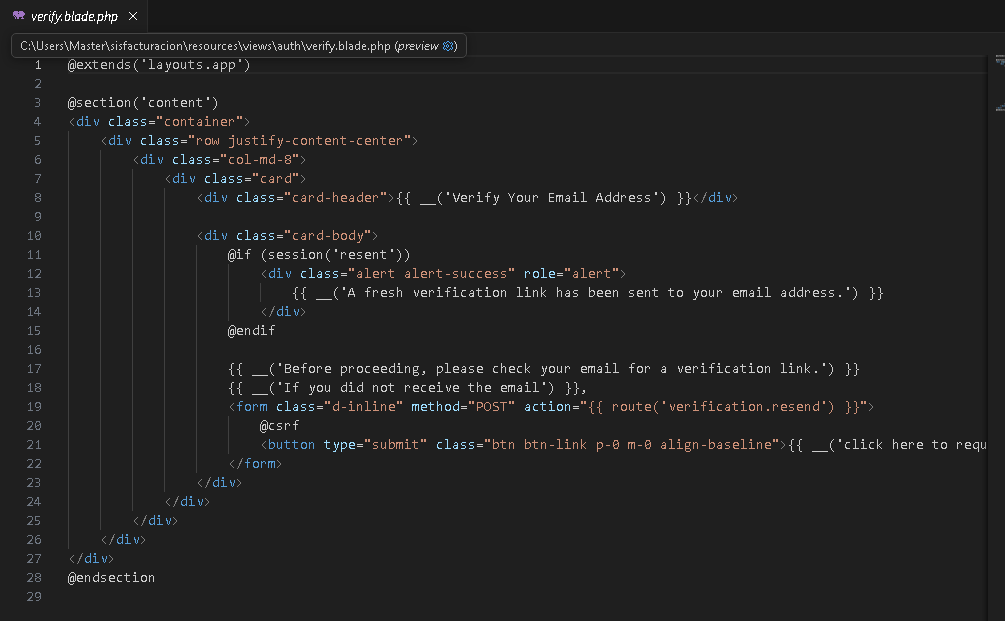




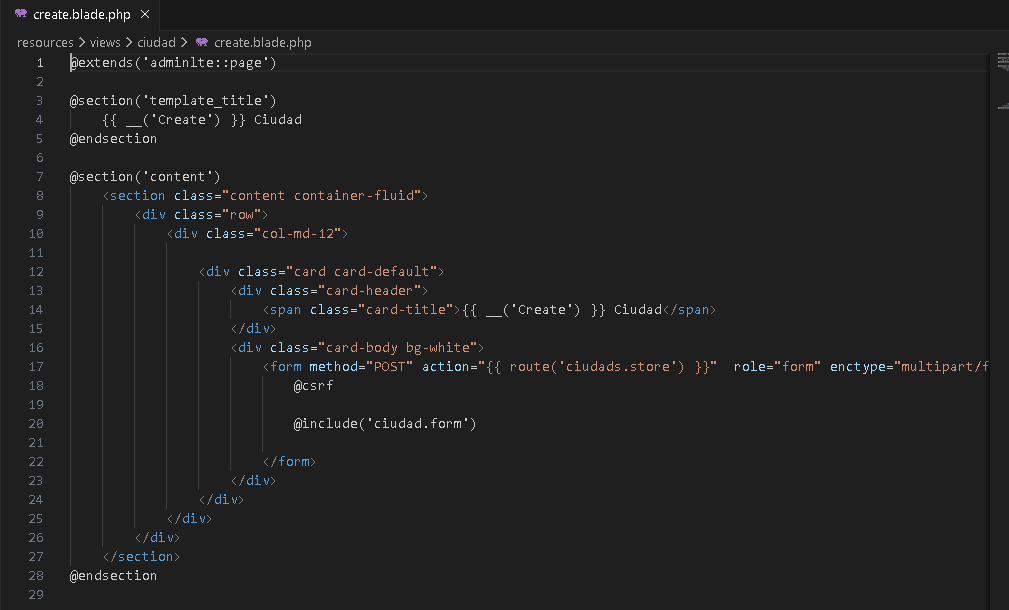


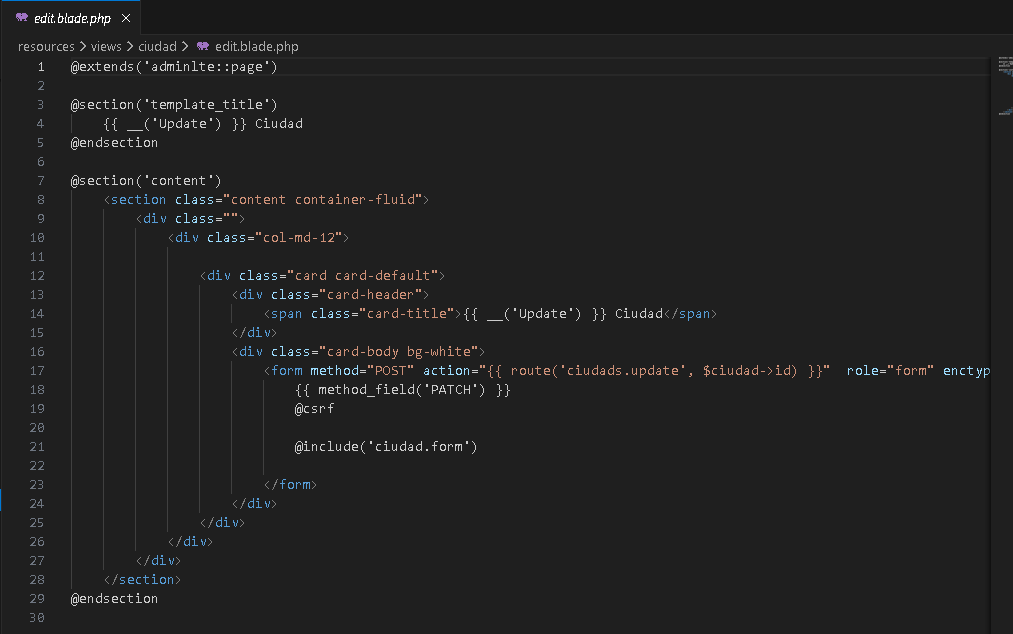




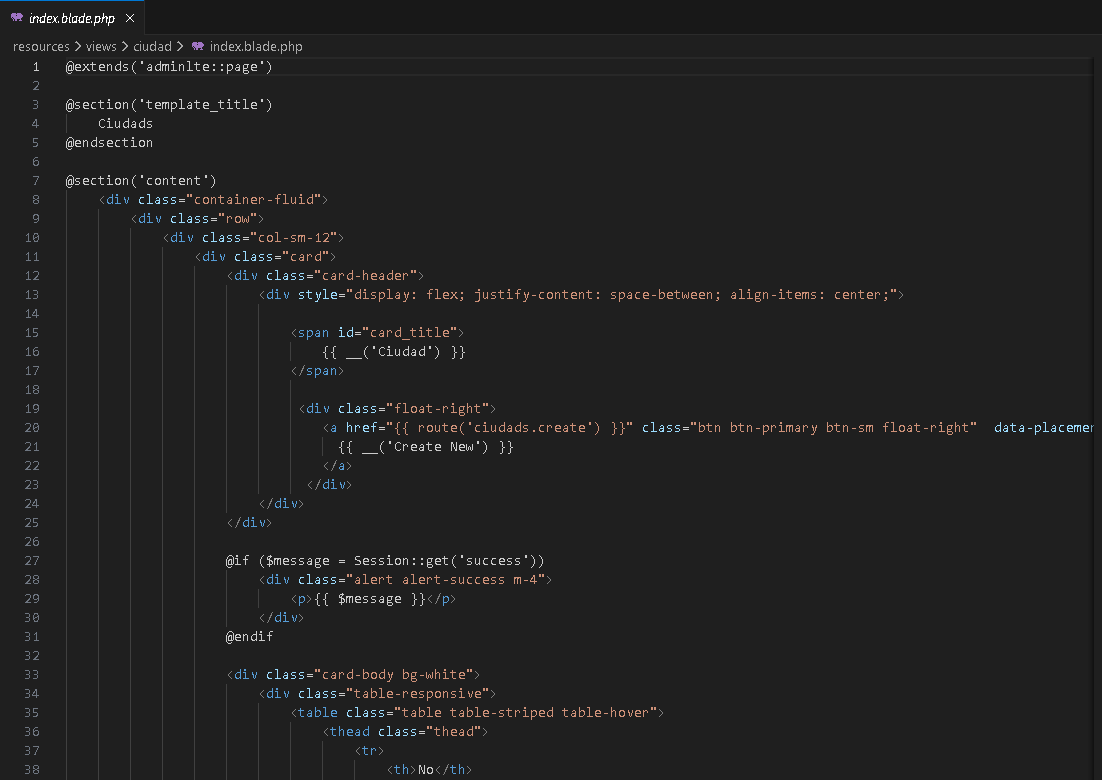


* **Ciudad**

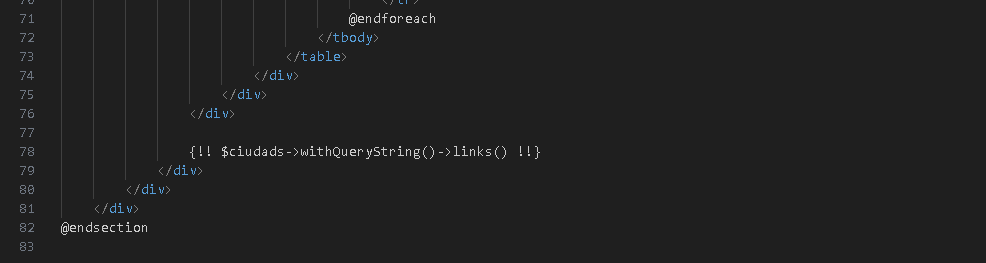


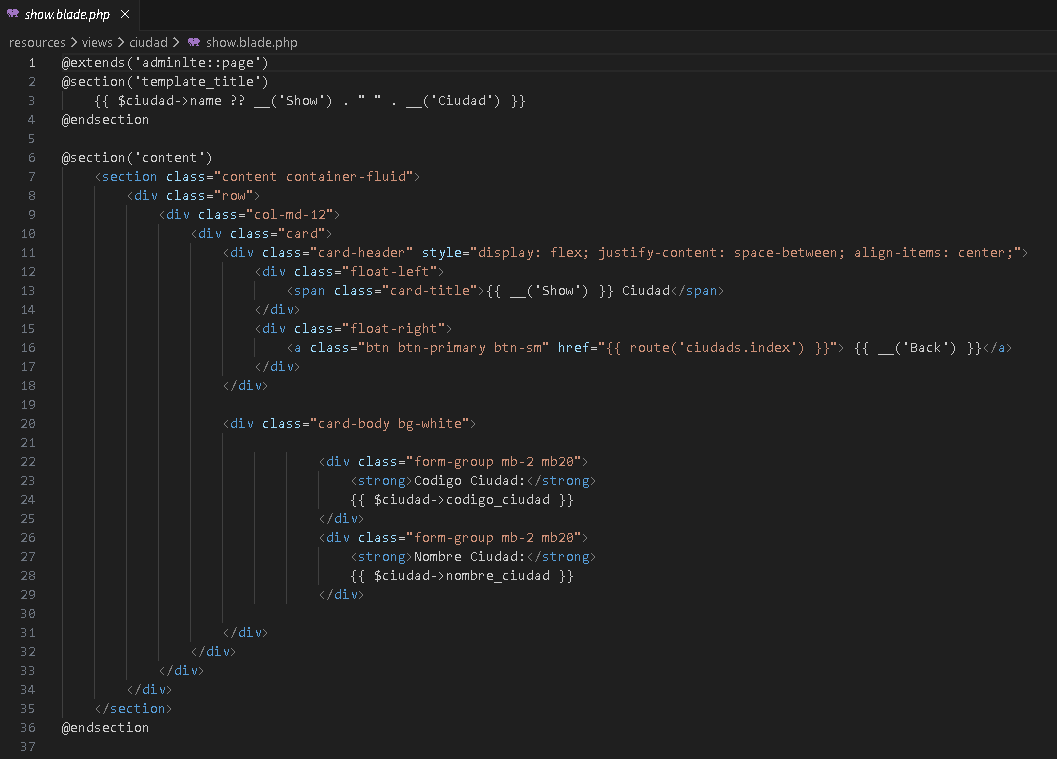




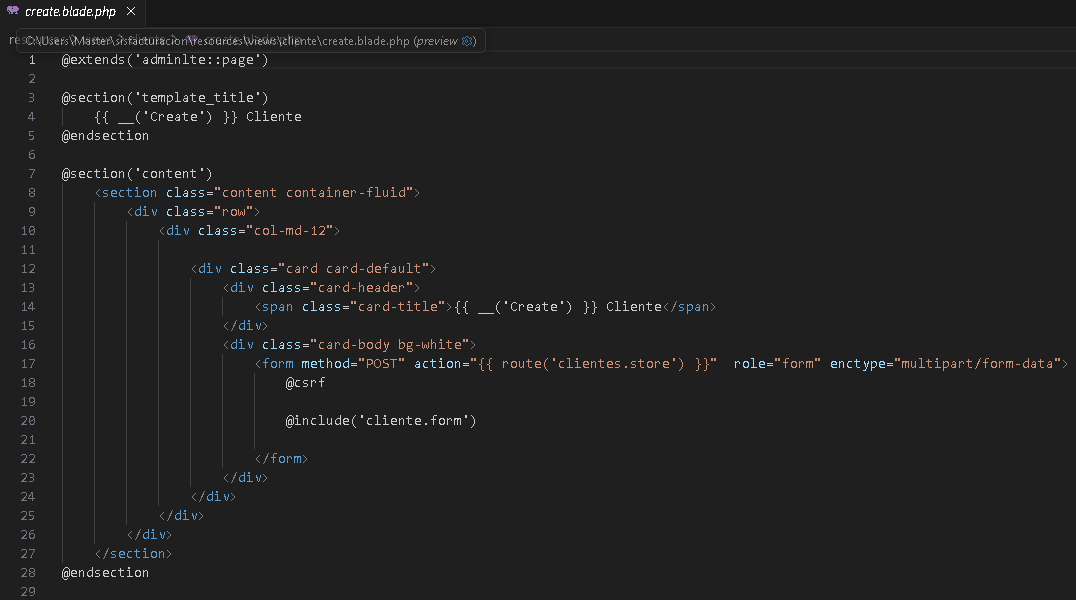


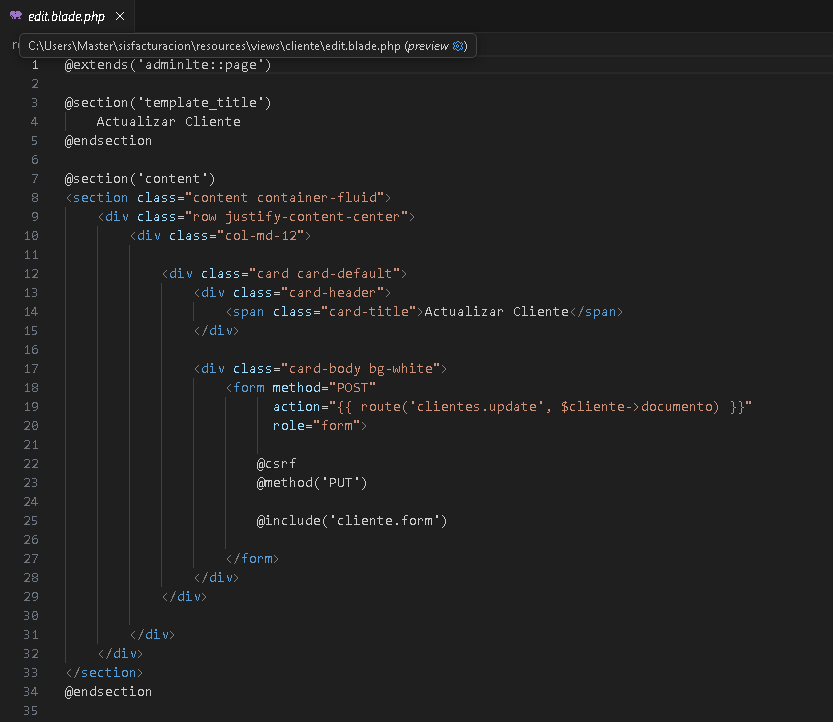


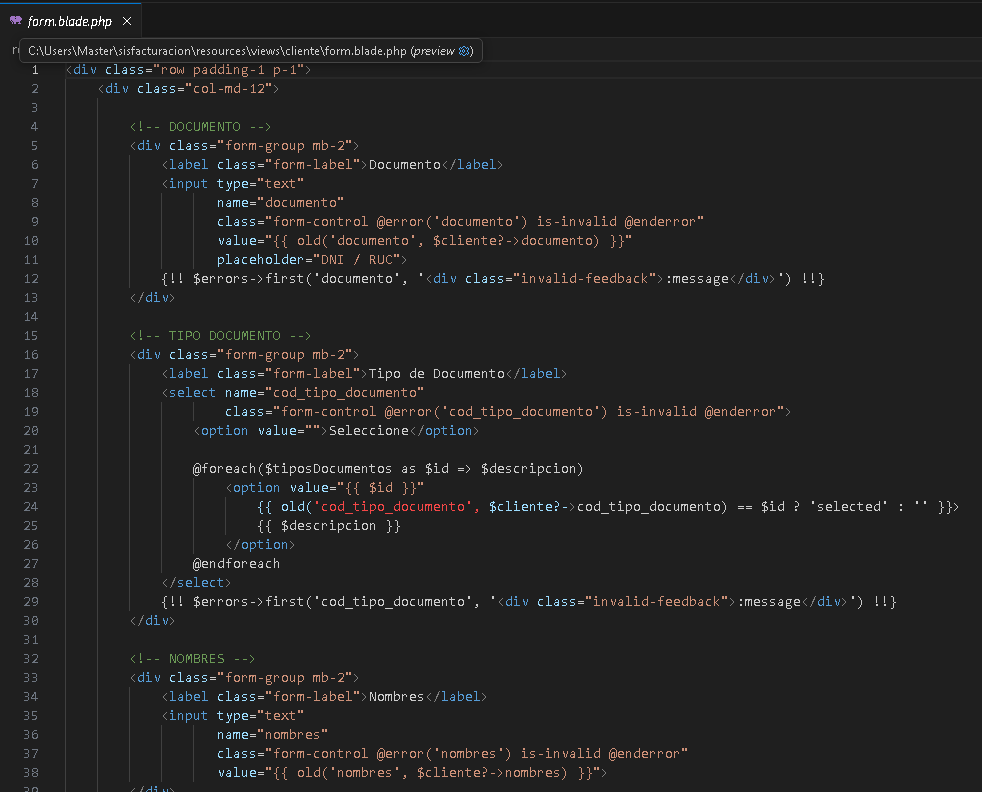




* **Cliente**



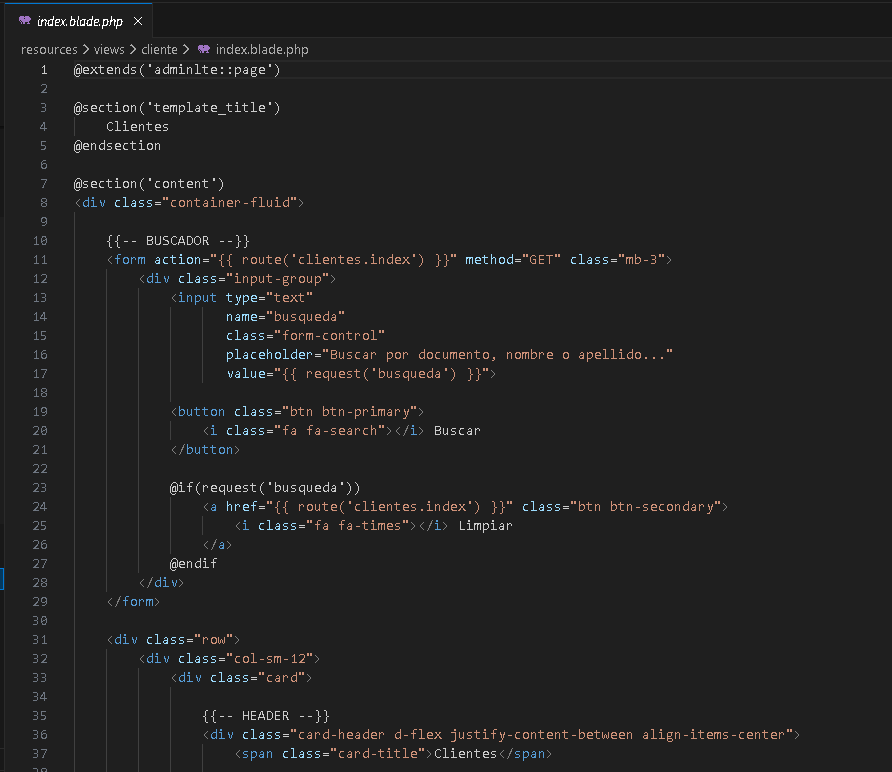


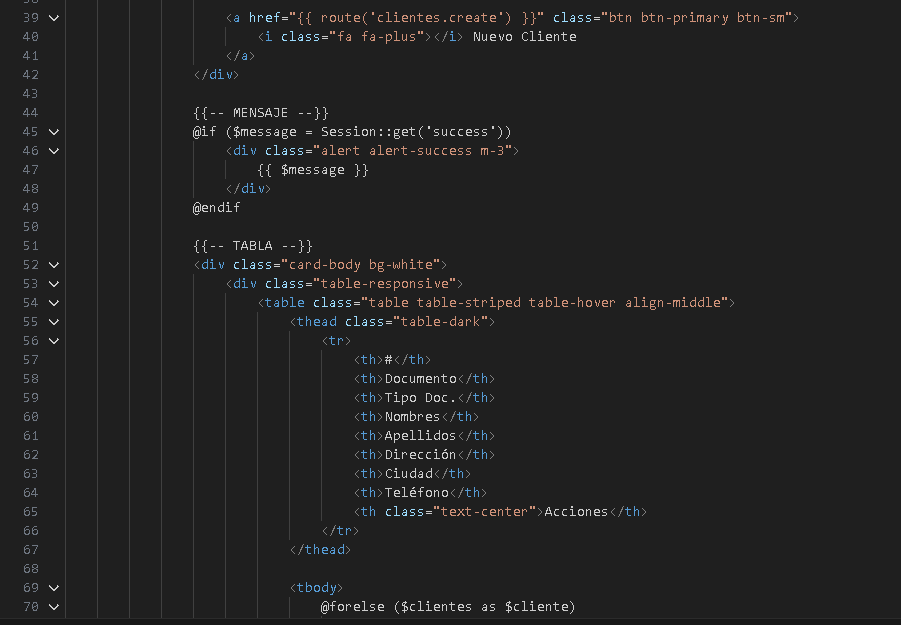


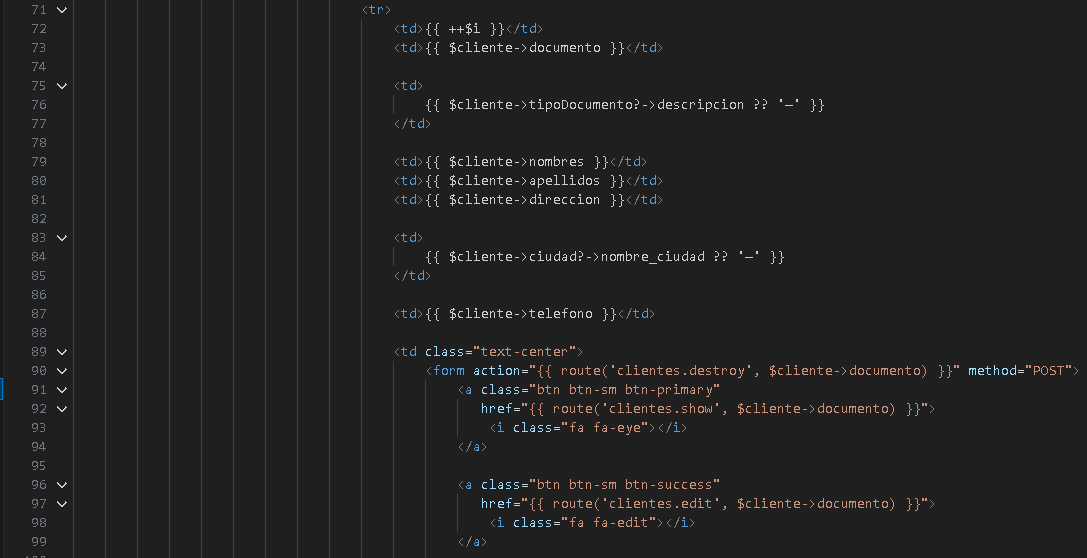






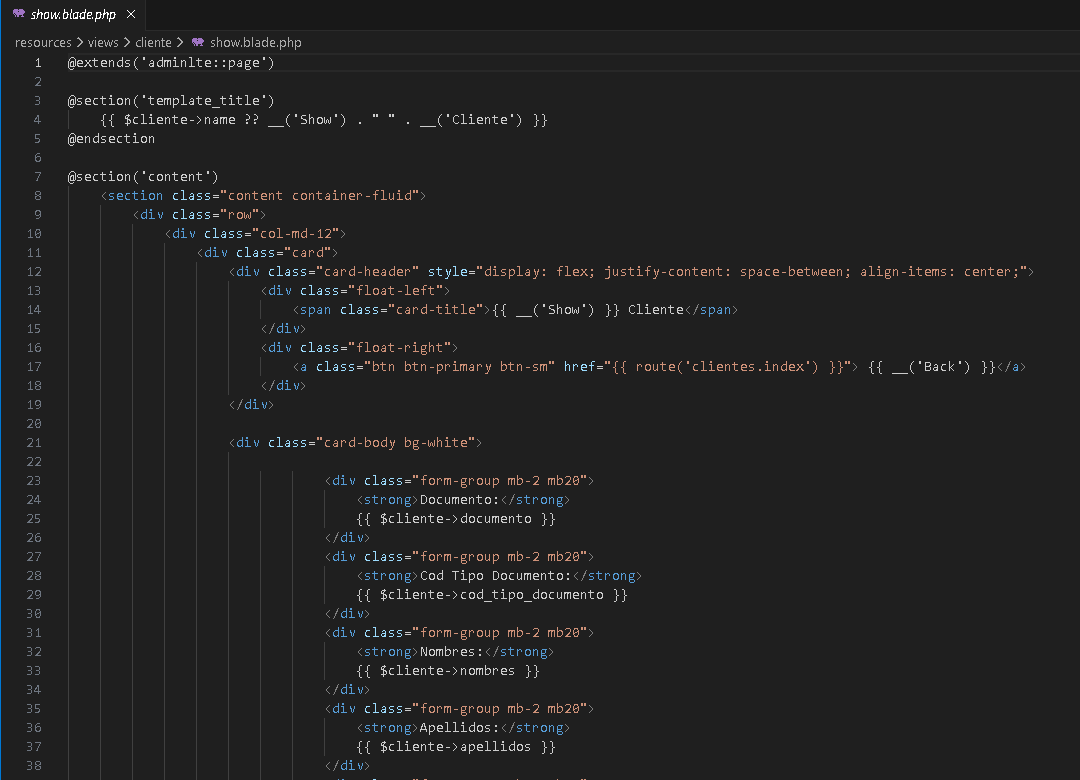


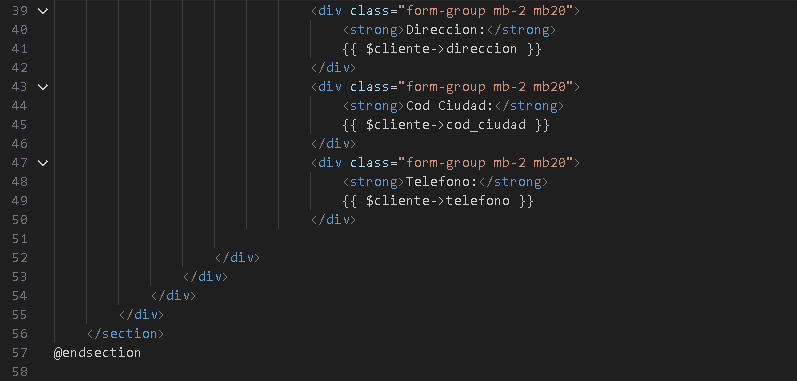




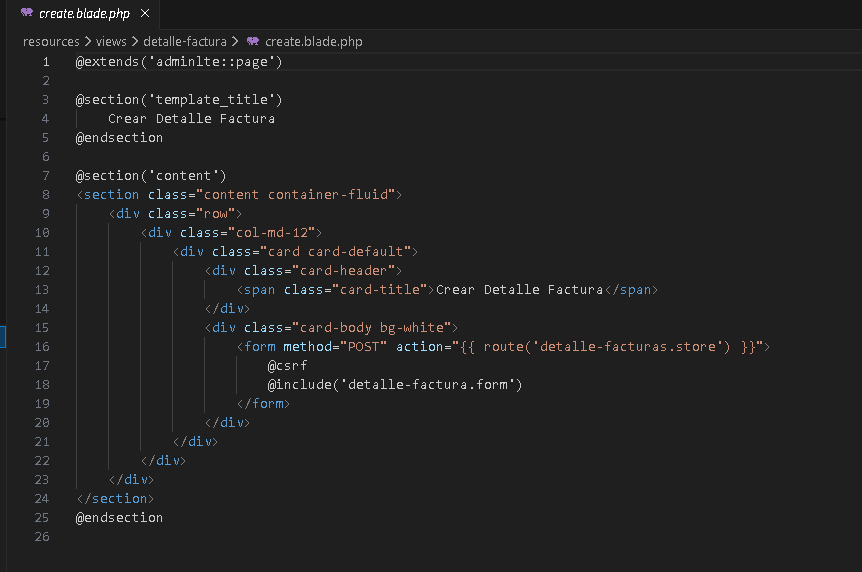


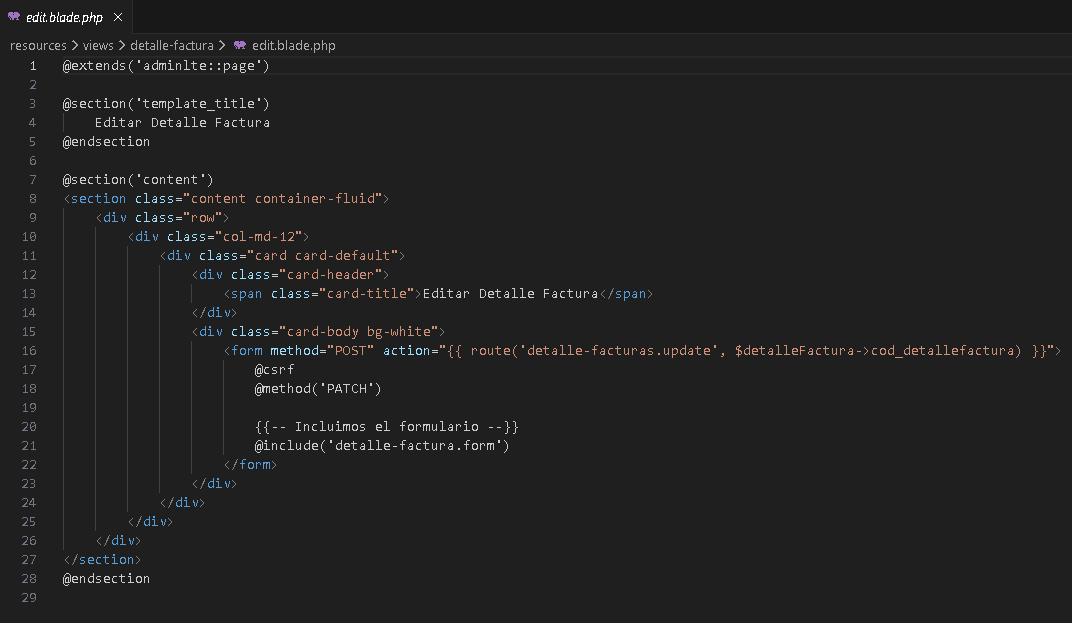


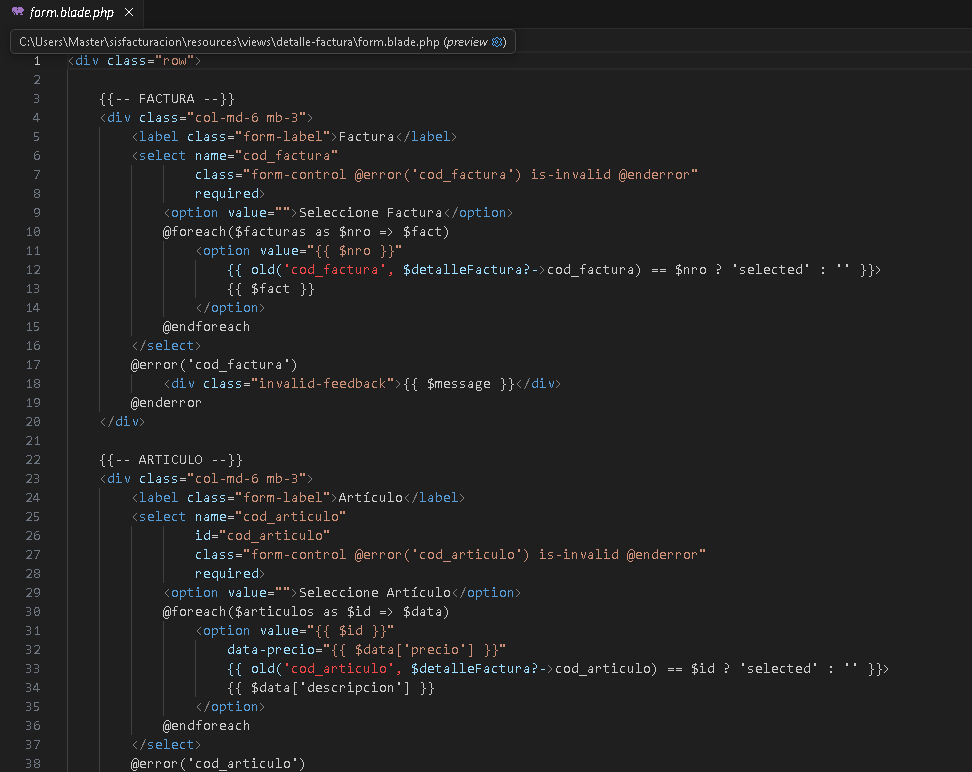


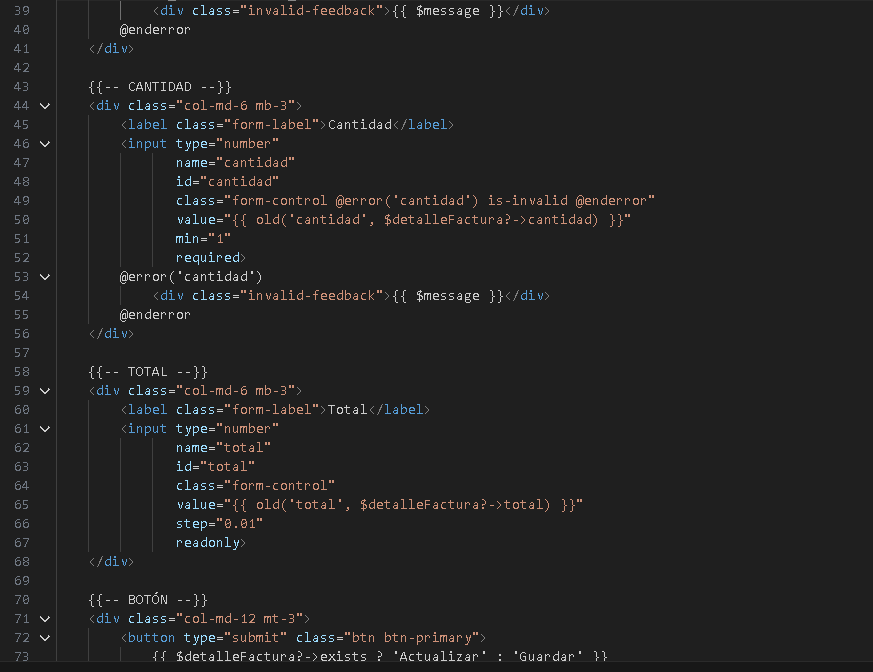


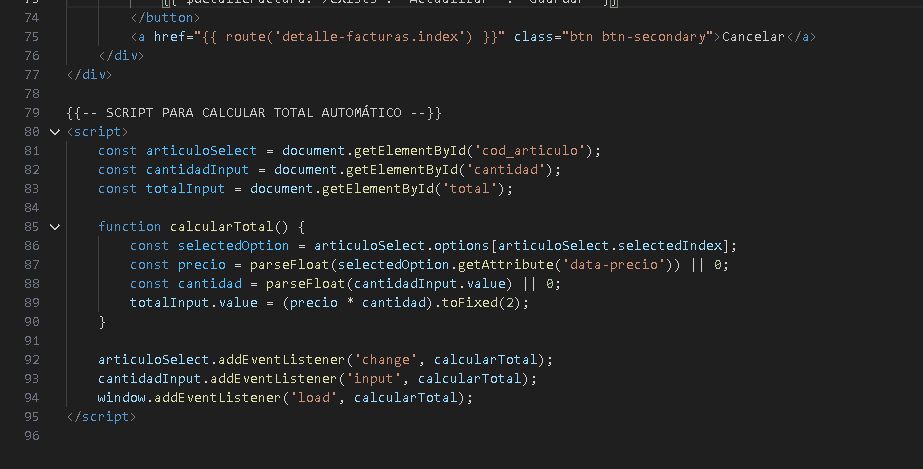
* **Detalle factura**

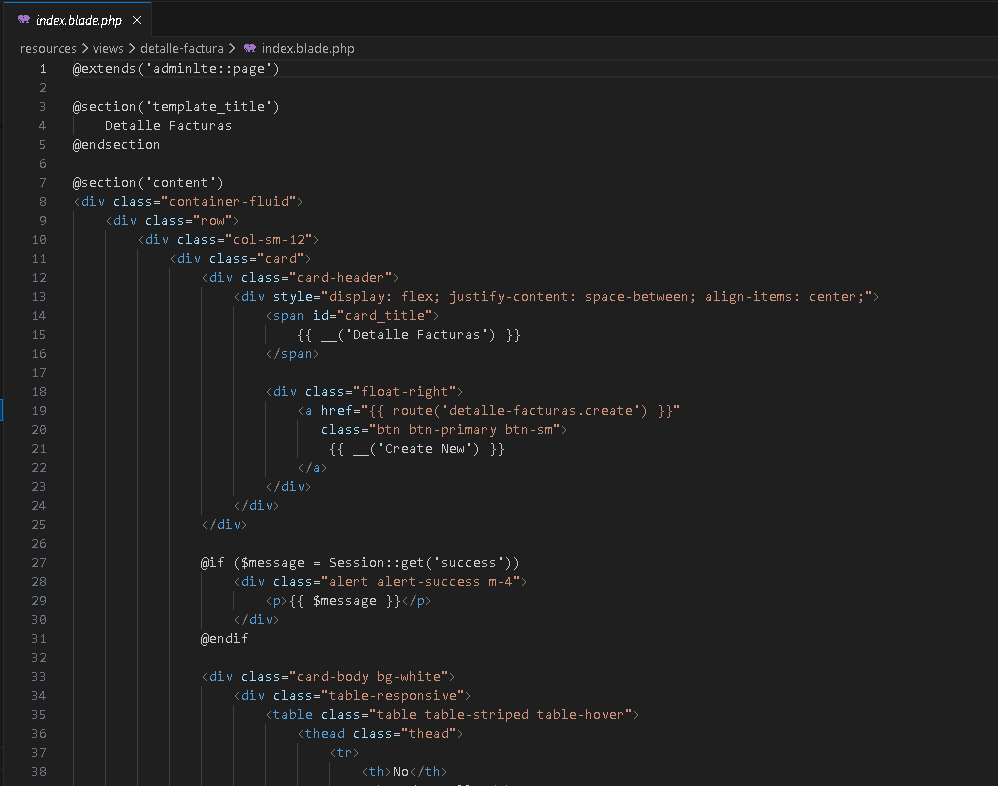


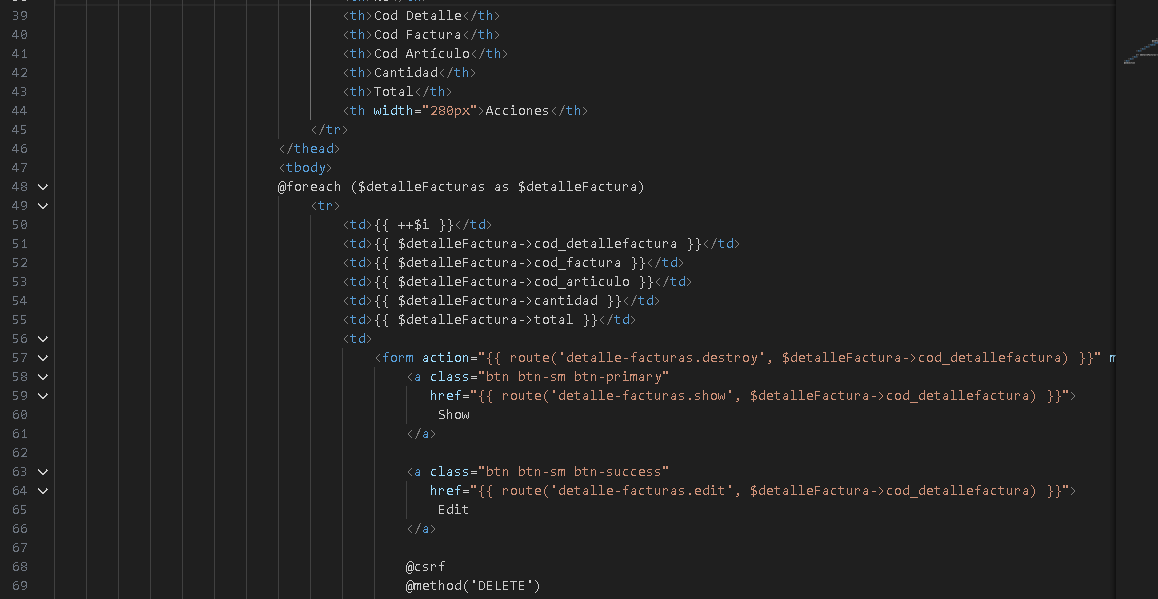




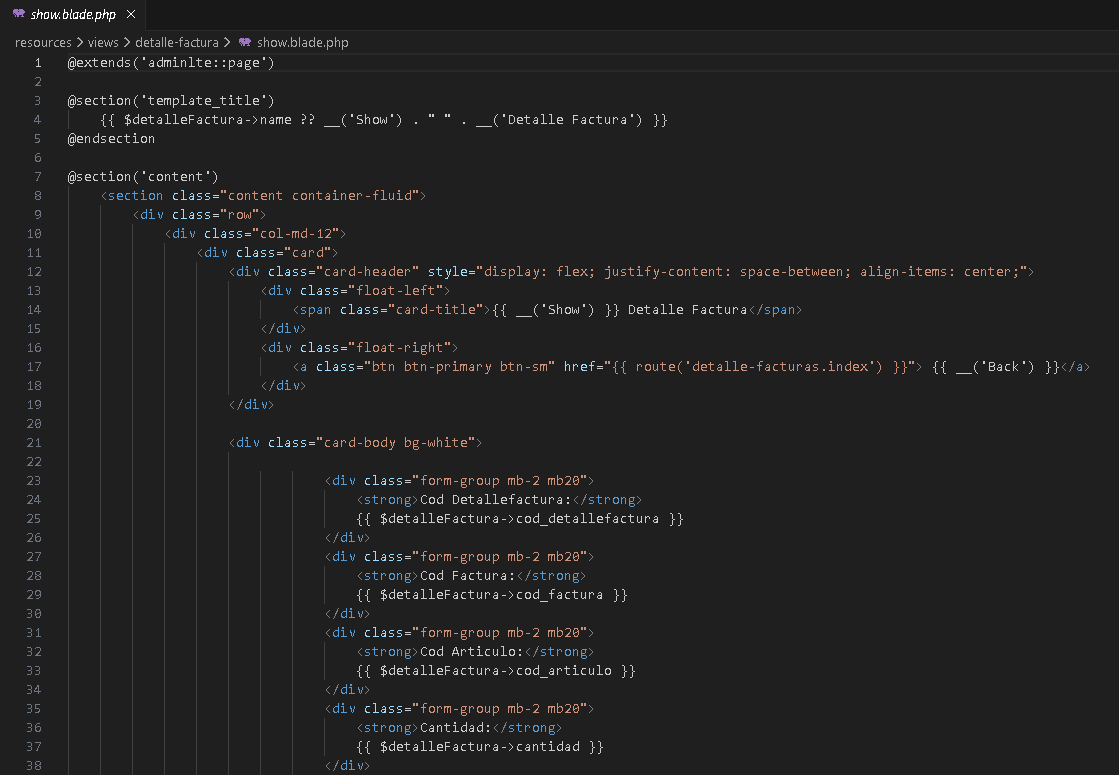


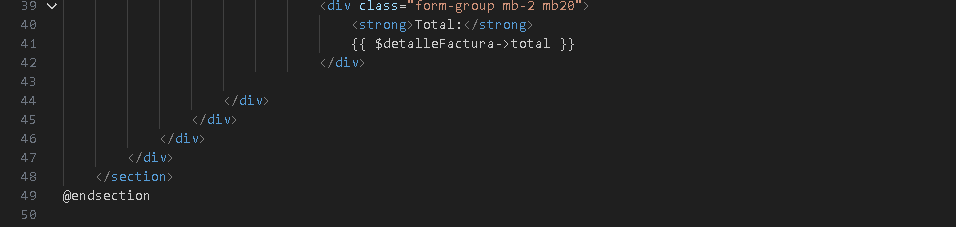




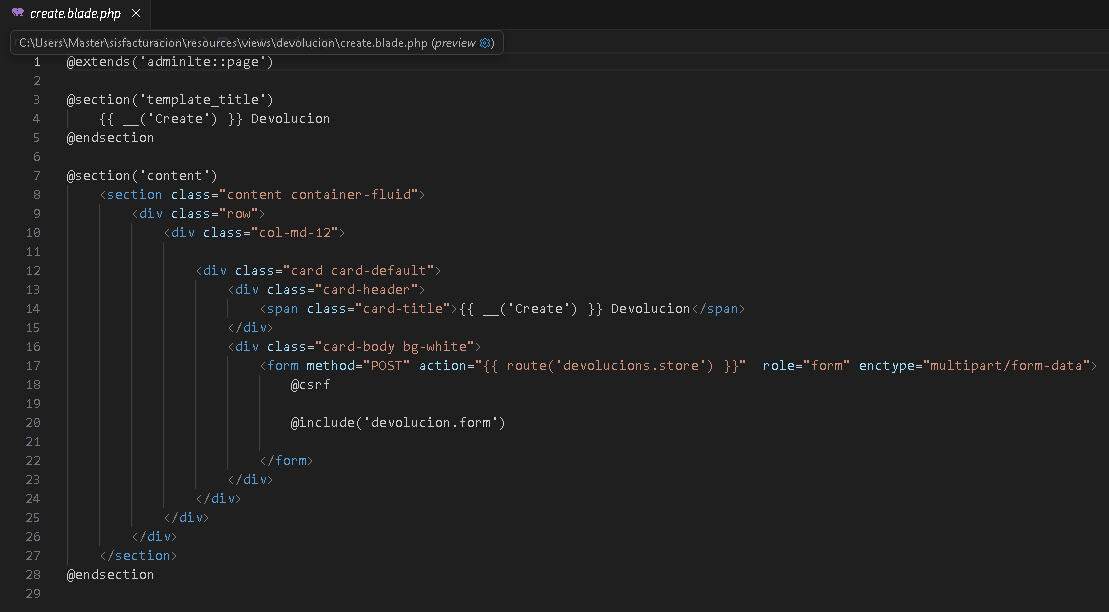


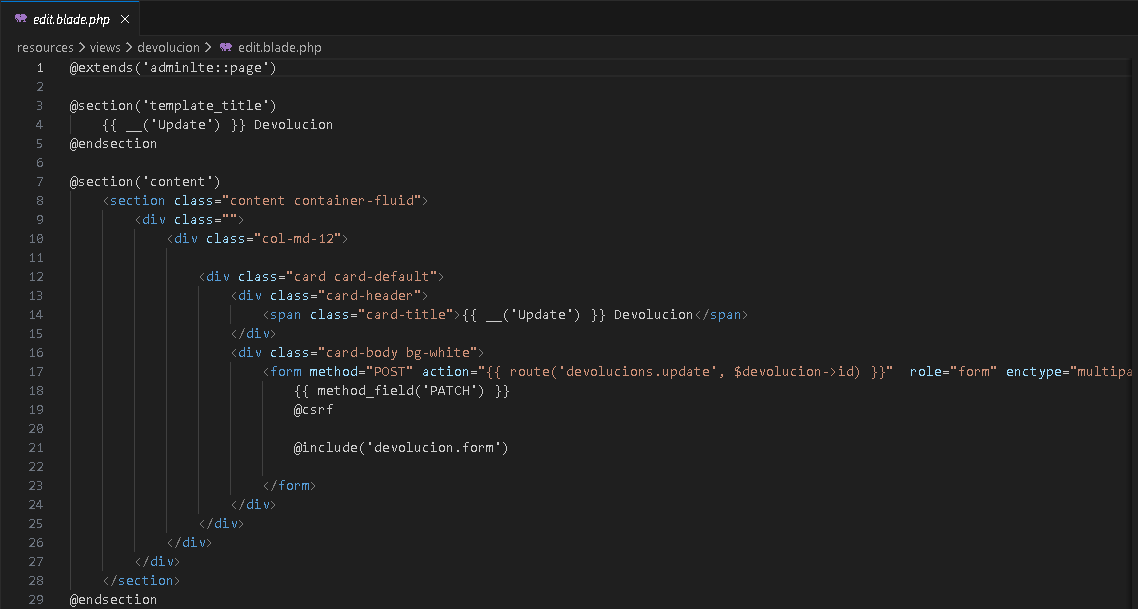


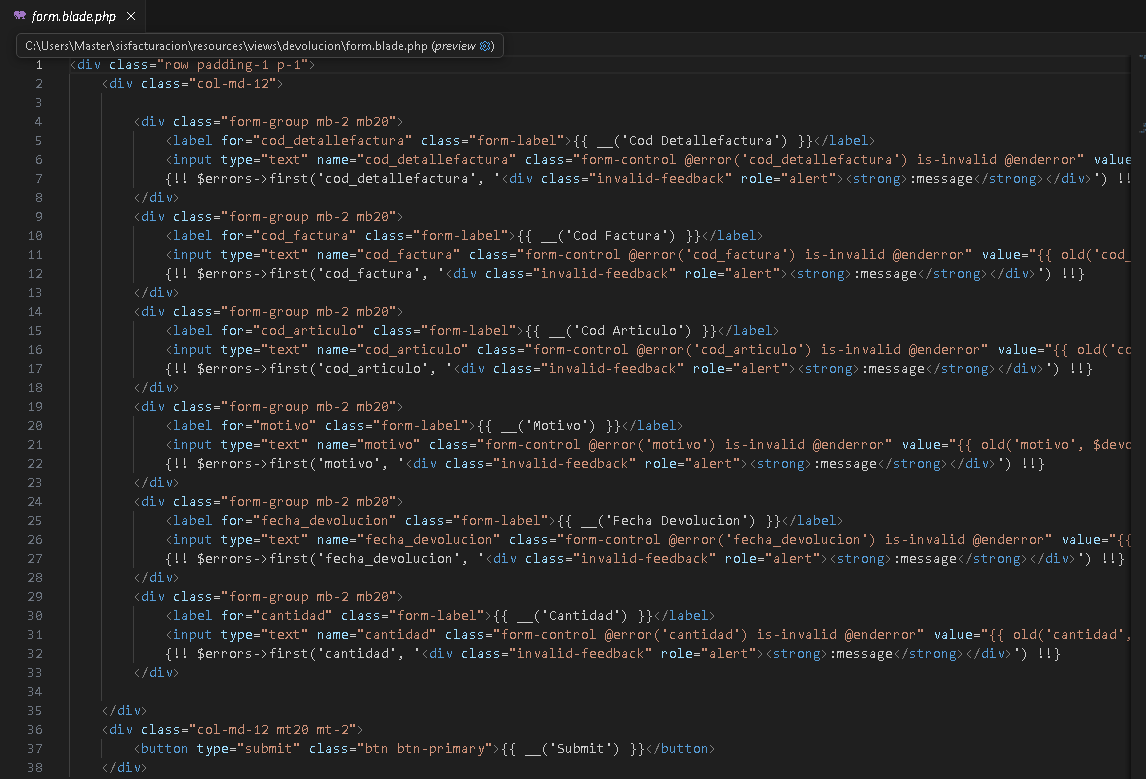




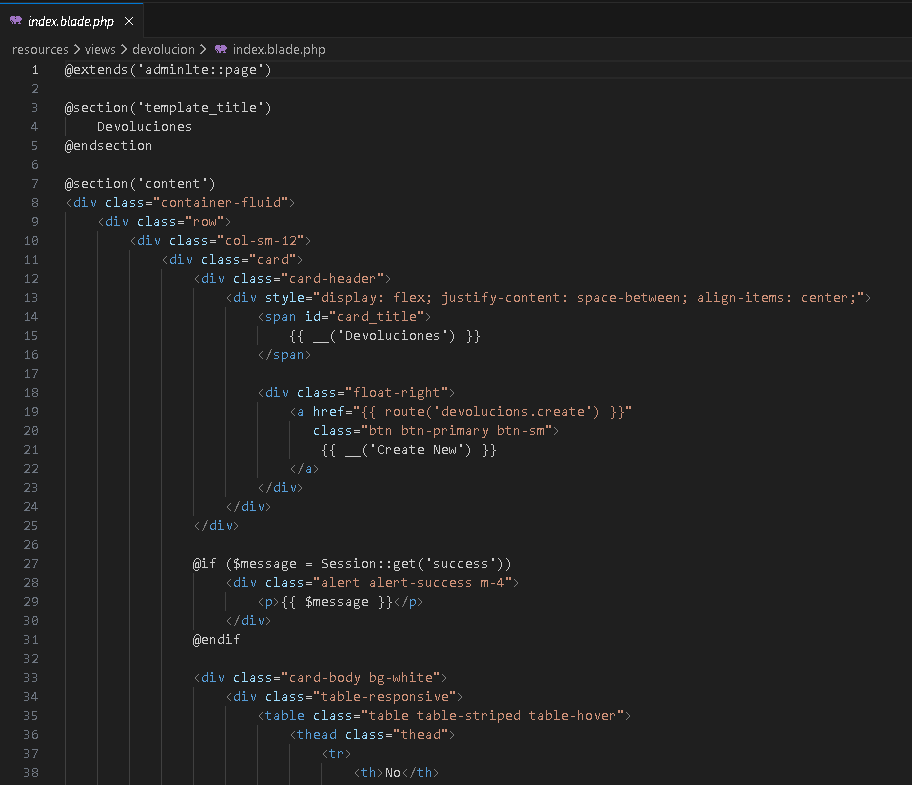
* **Devolución**



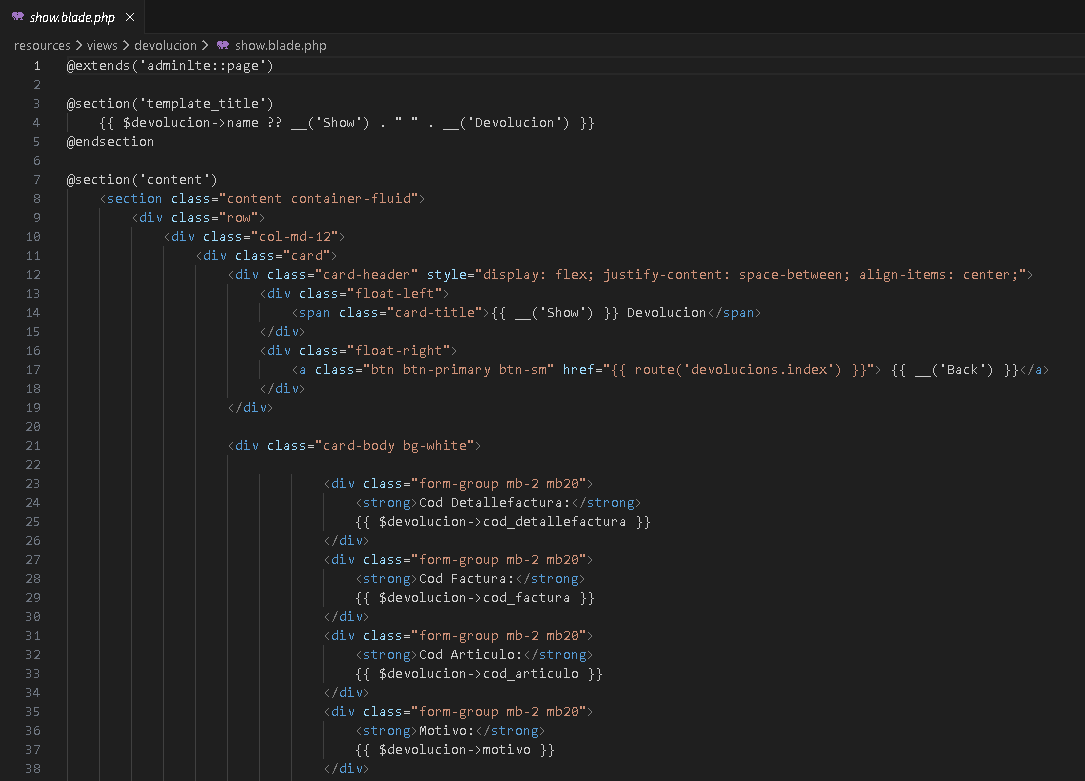


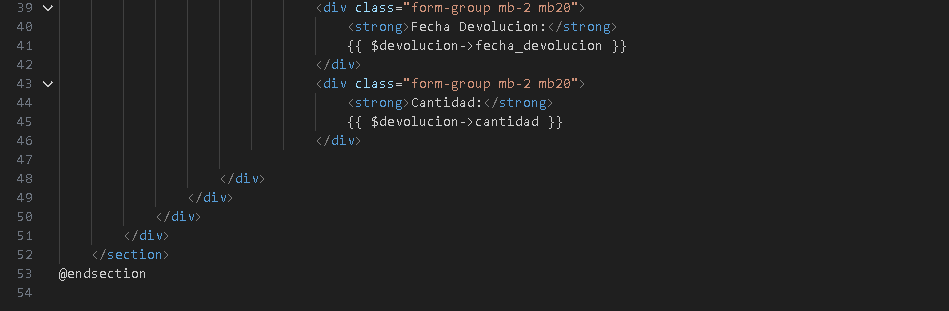




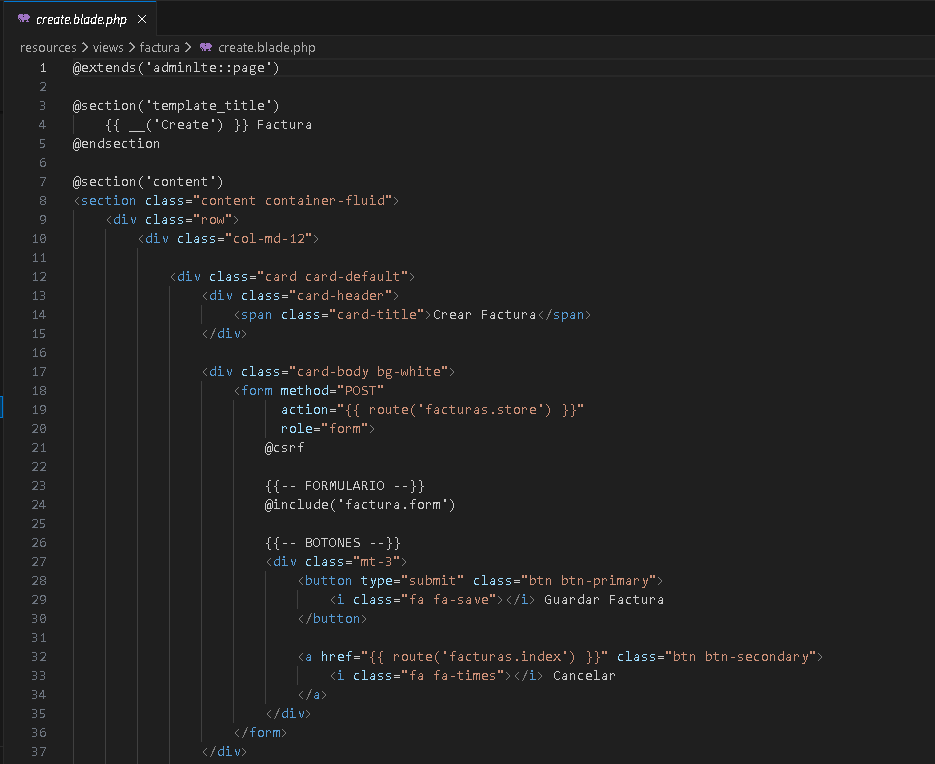




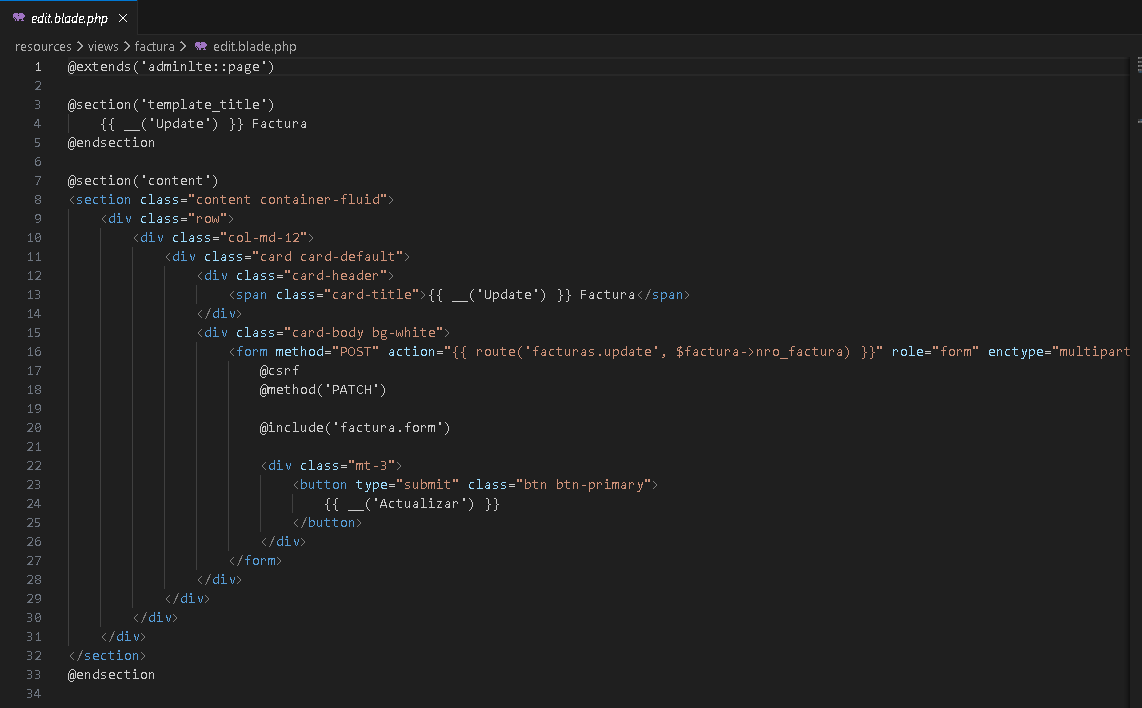


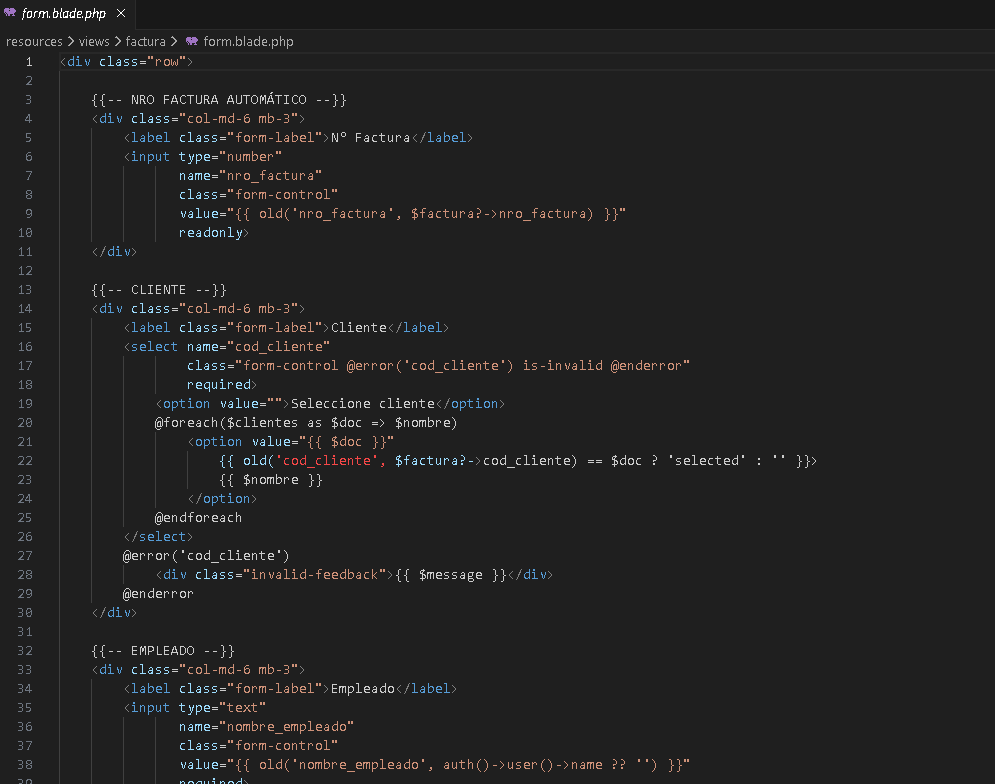


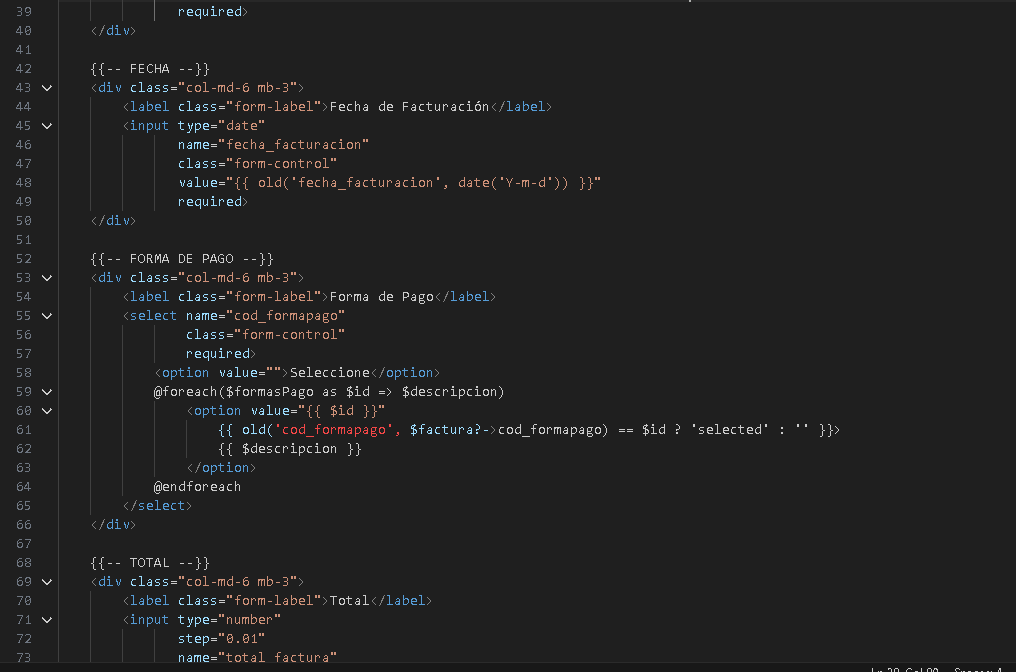
* **Factura**

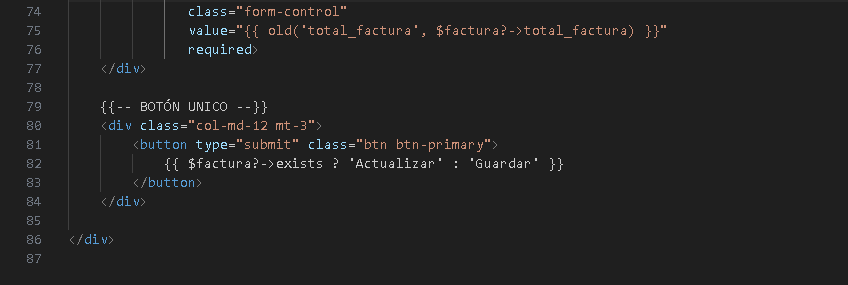




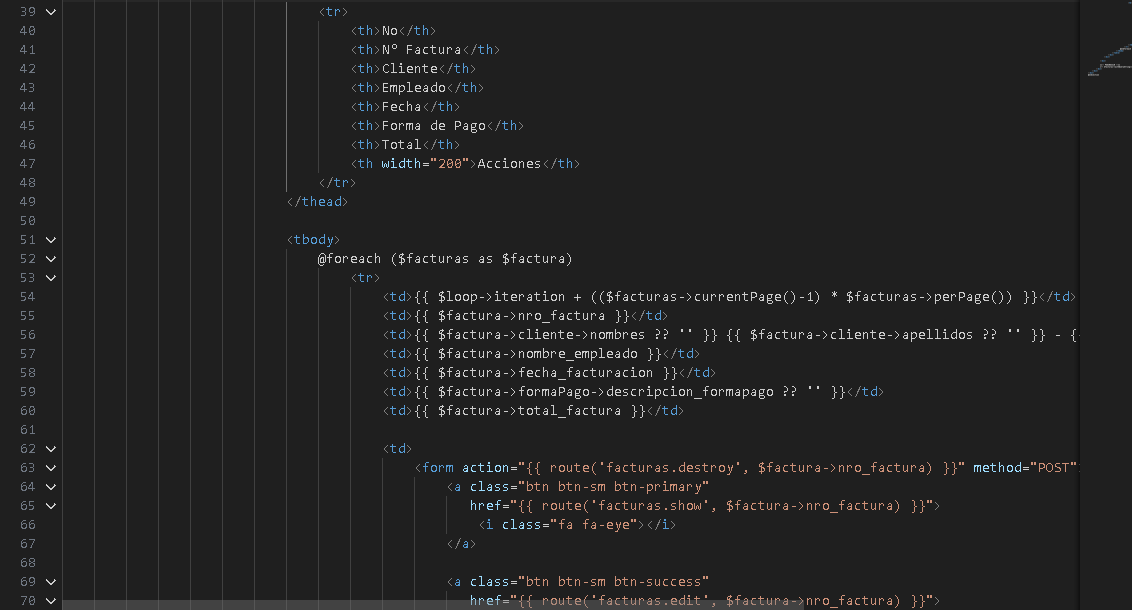




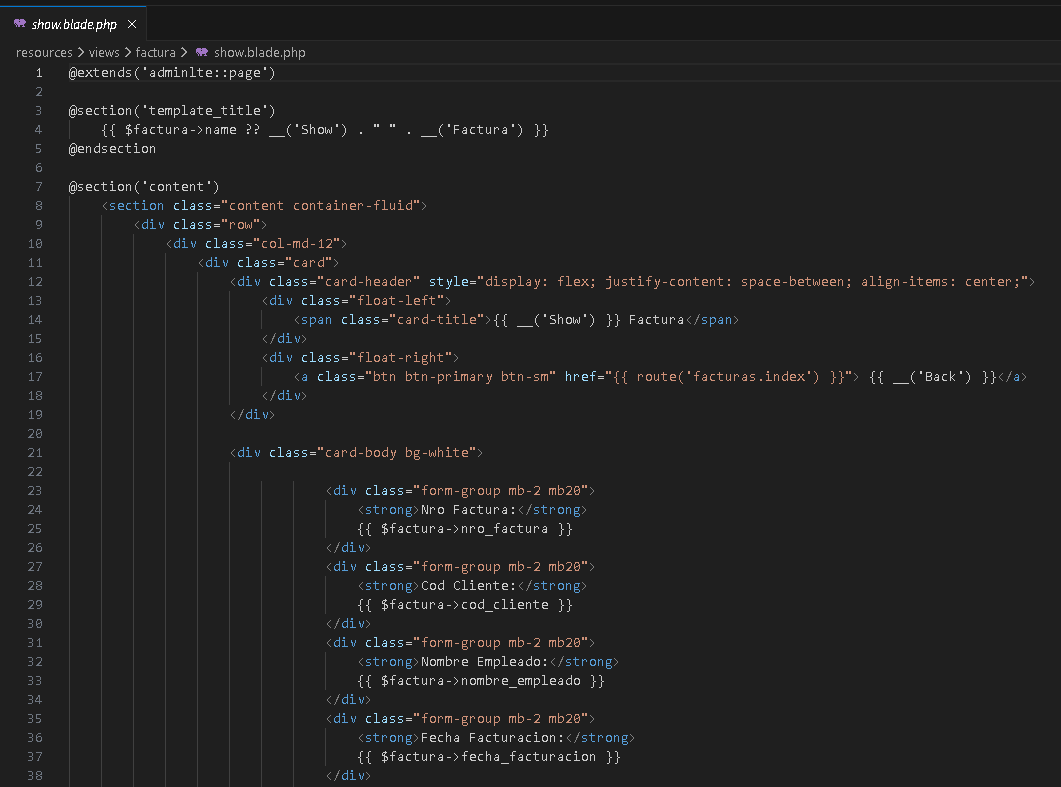






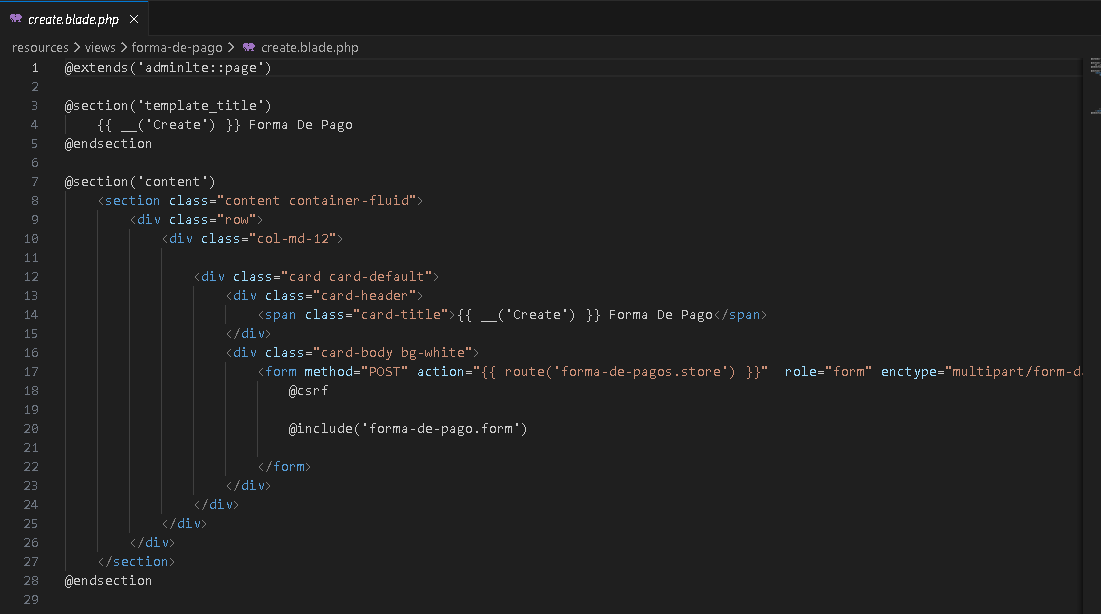


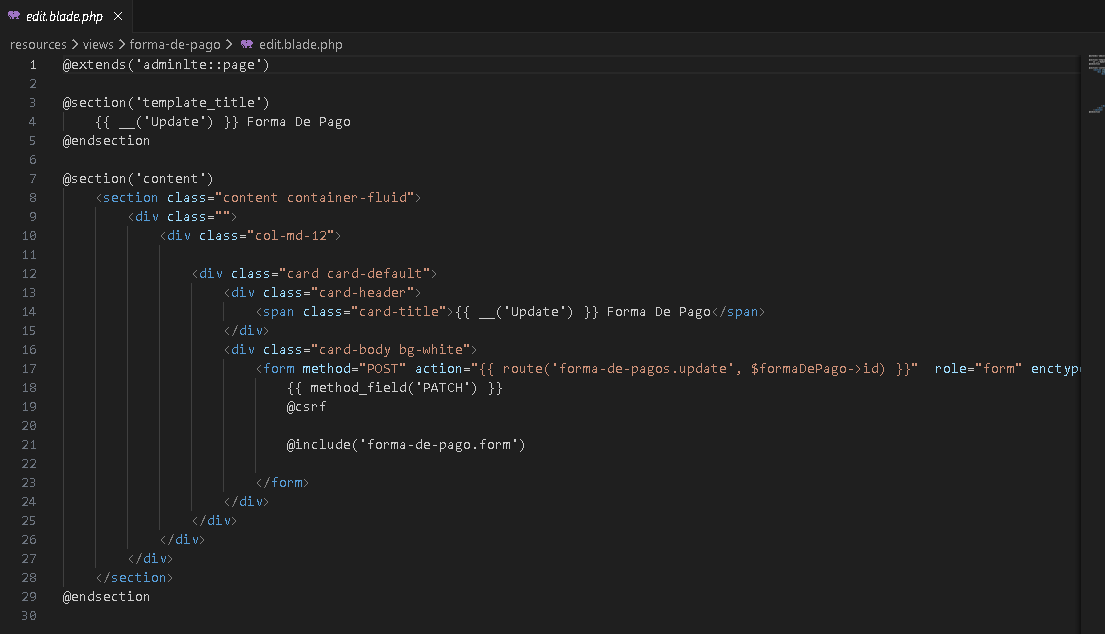


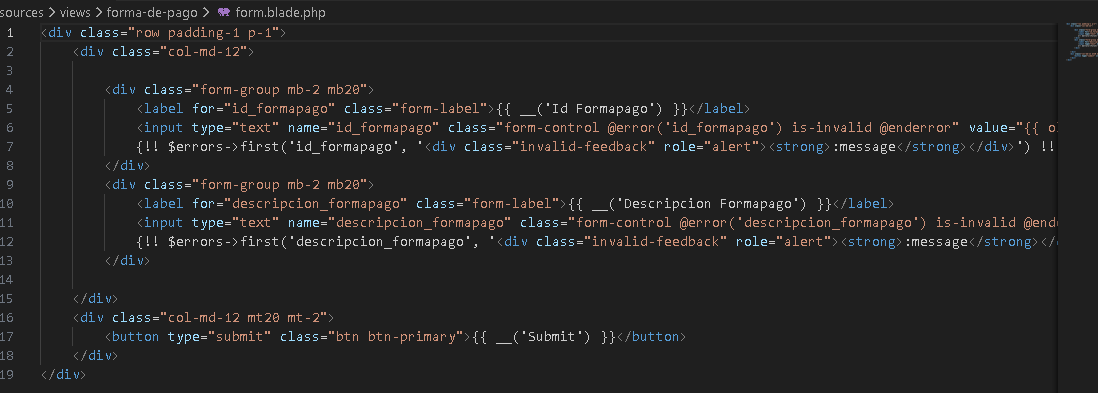


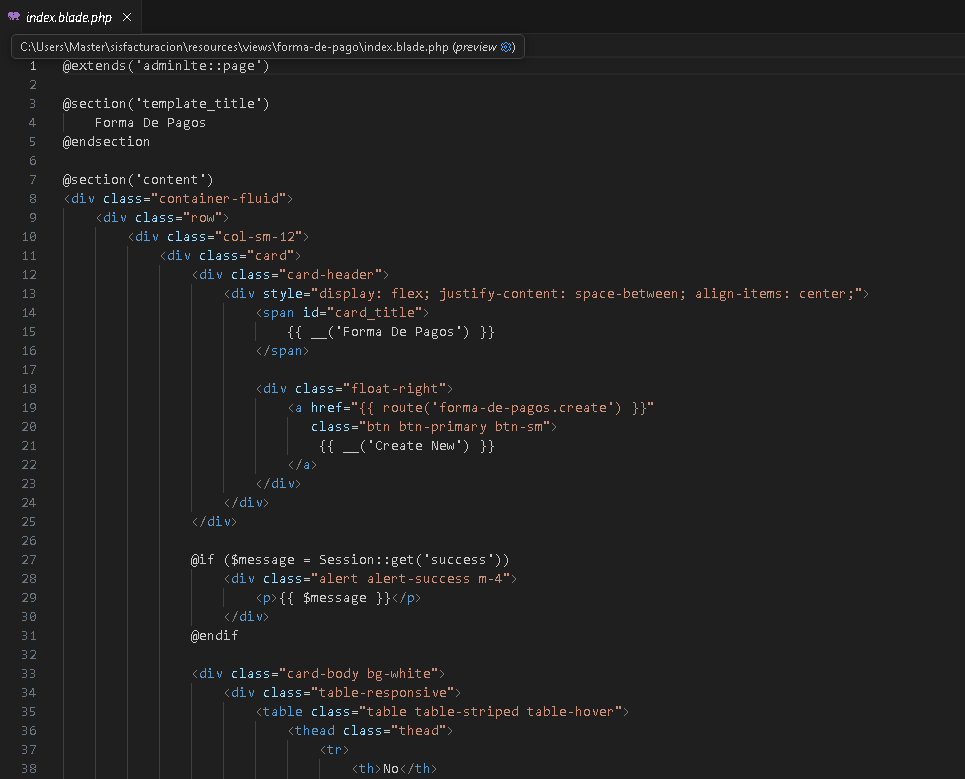


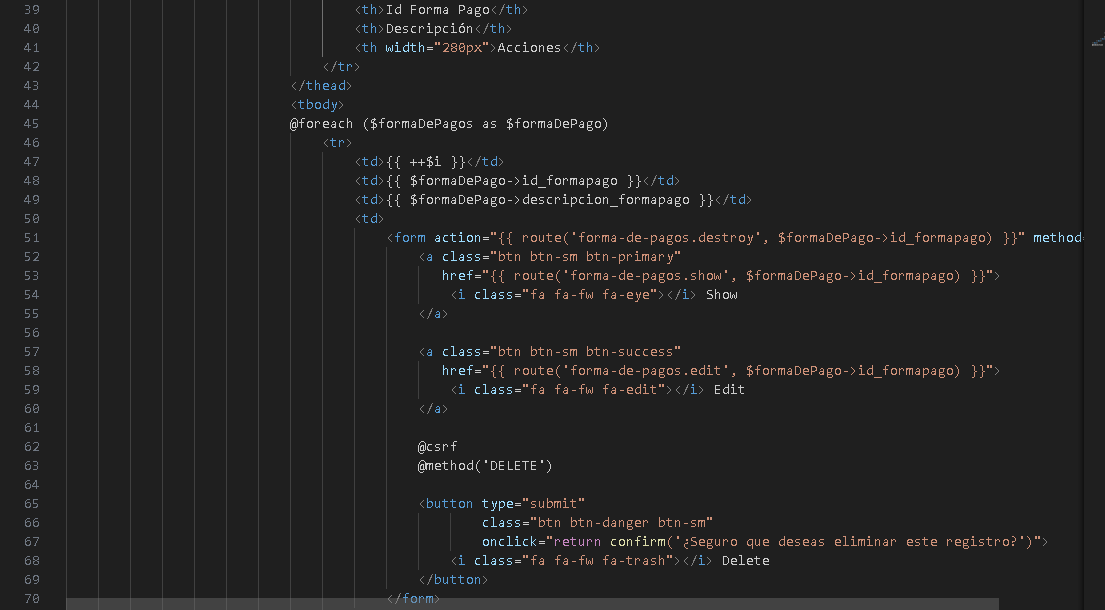
* **Formas de pago**

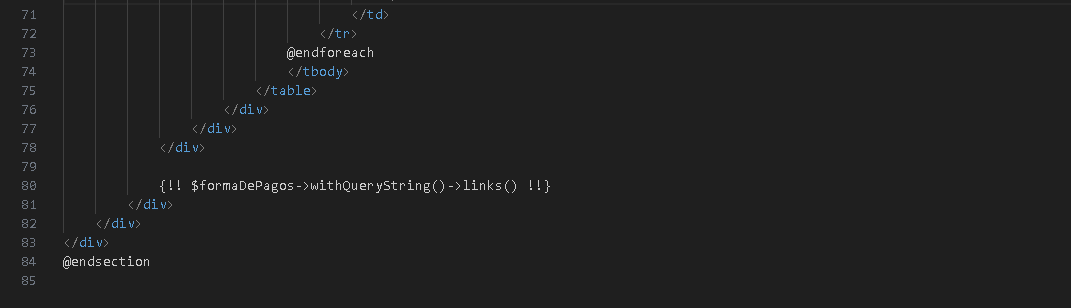


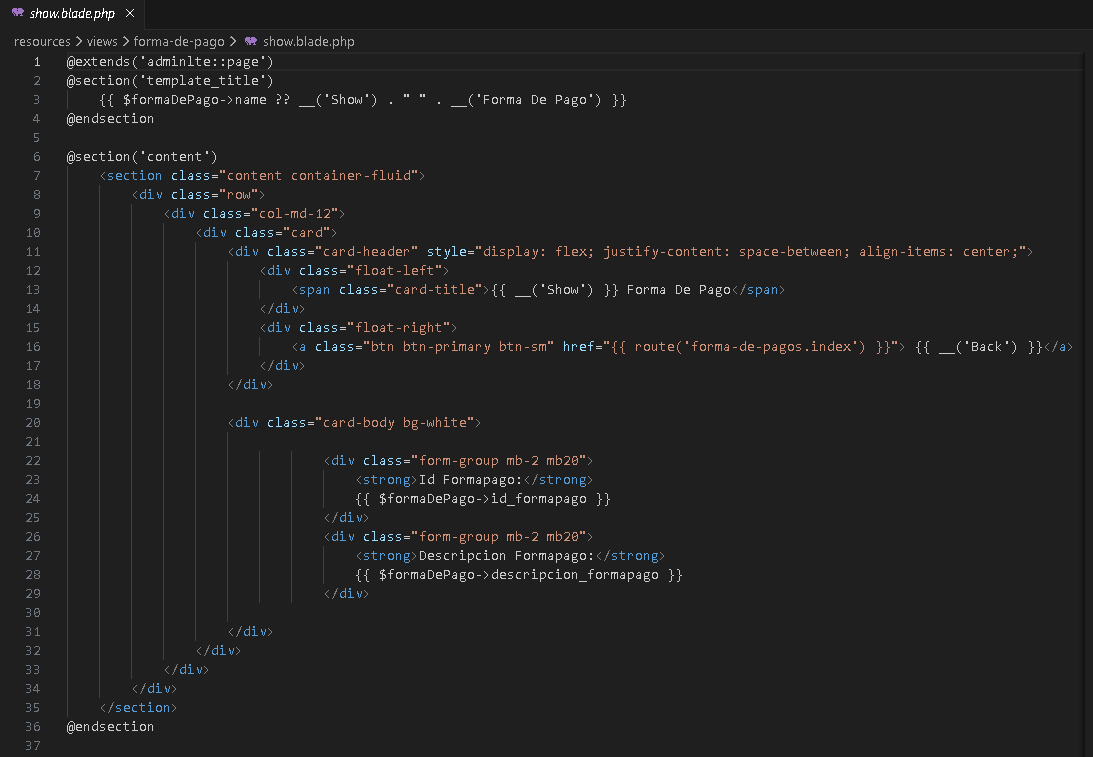




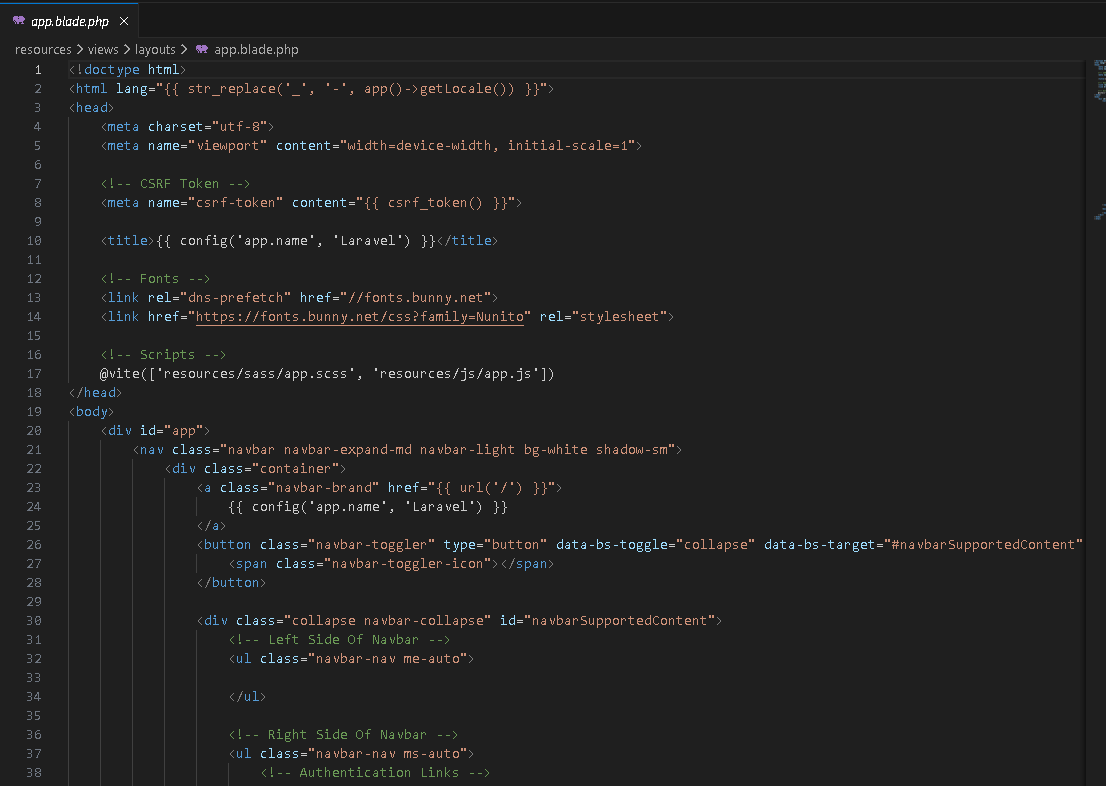




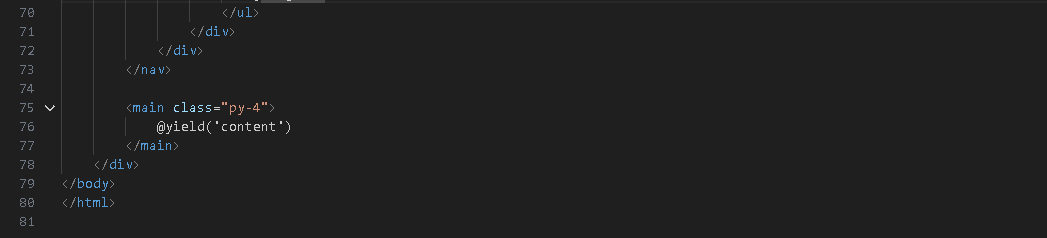




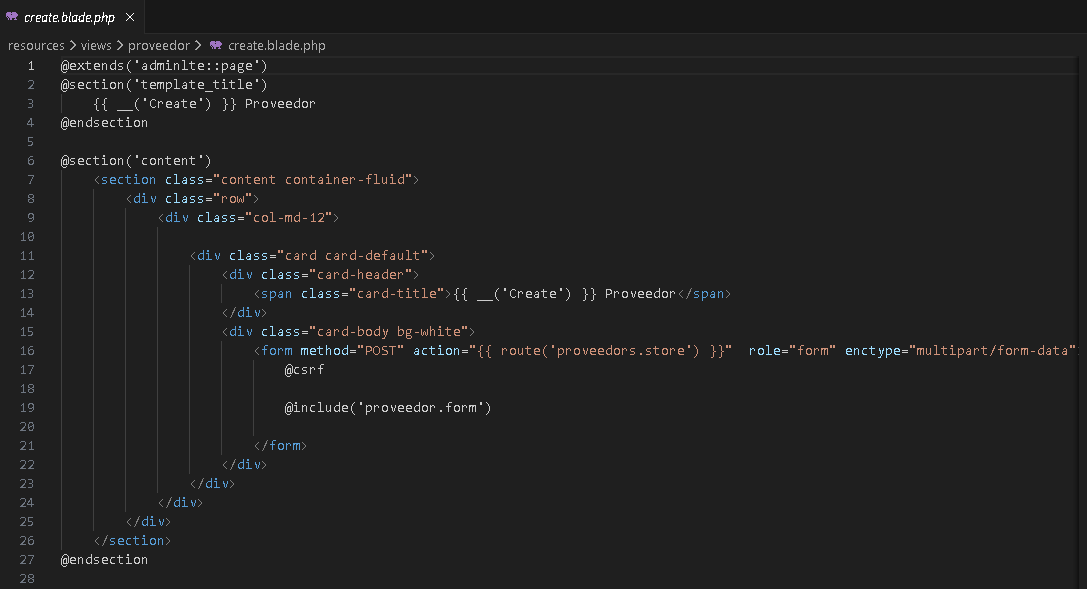
* **Layouts**

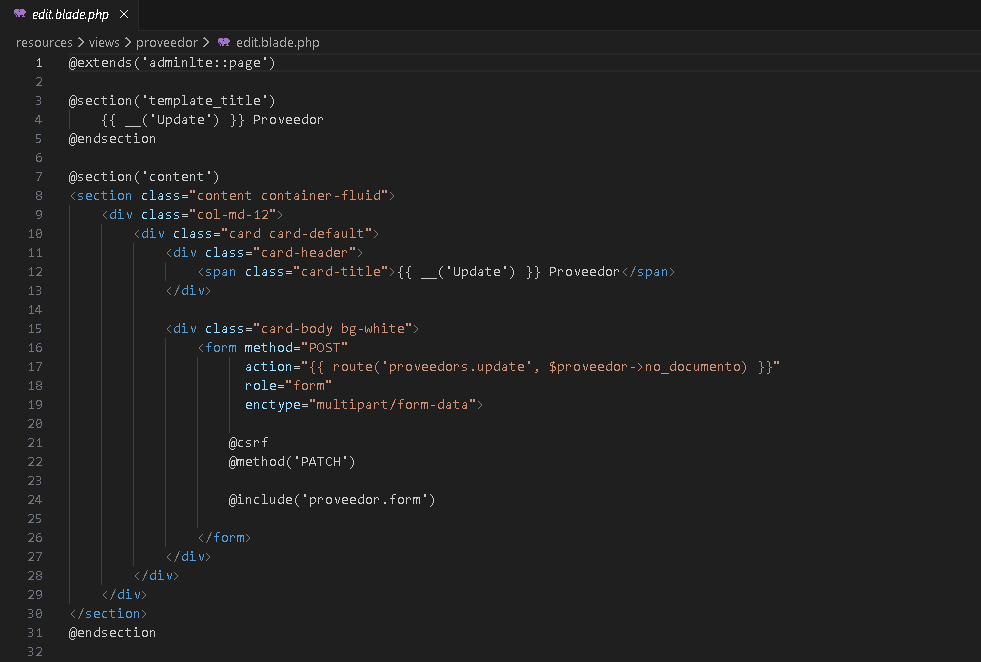


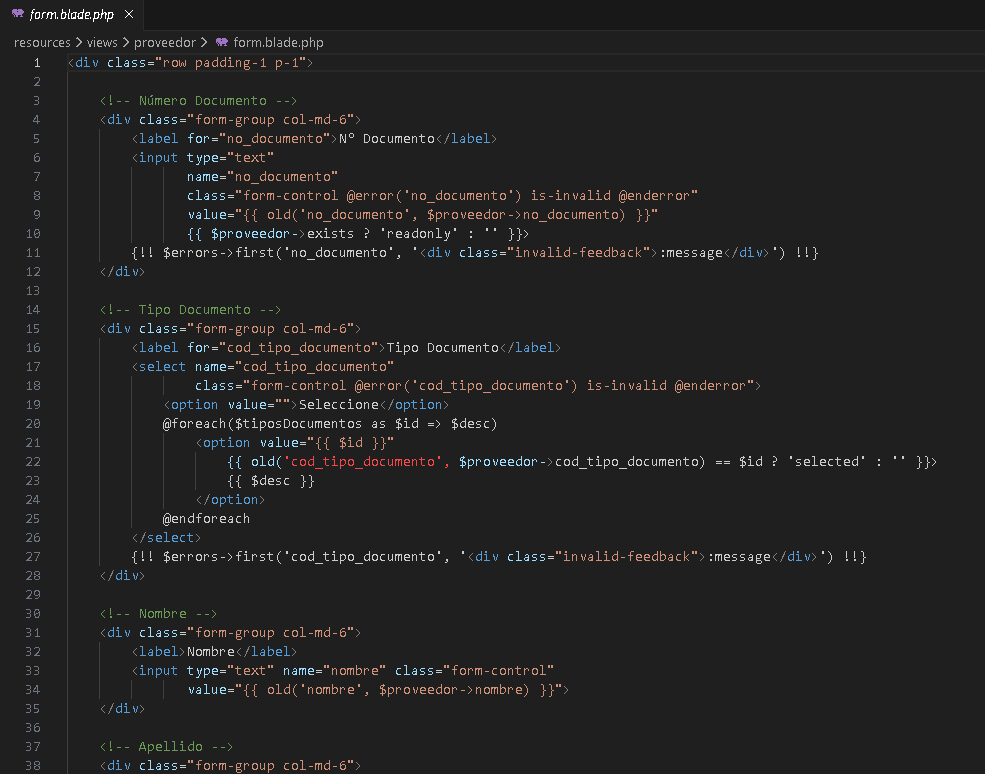


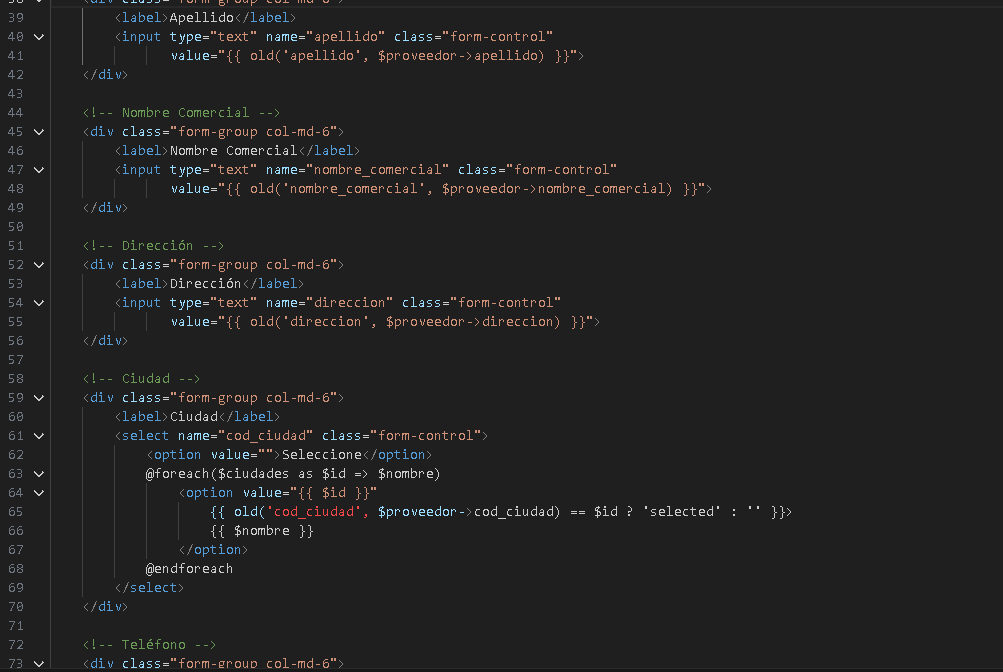


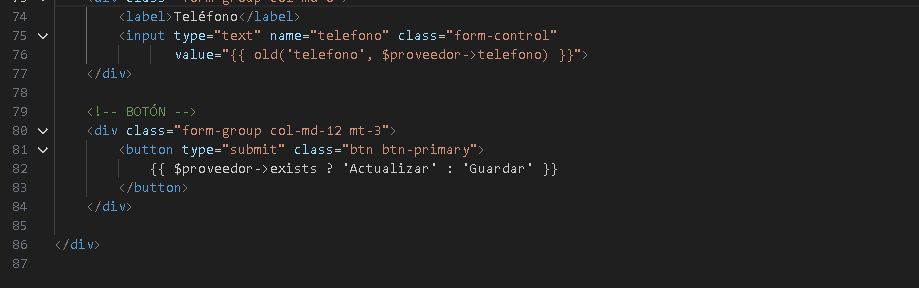
* **Proveedor**

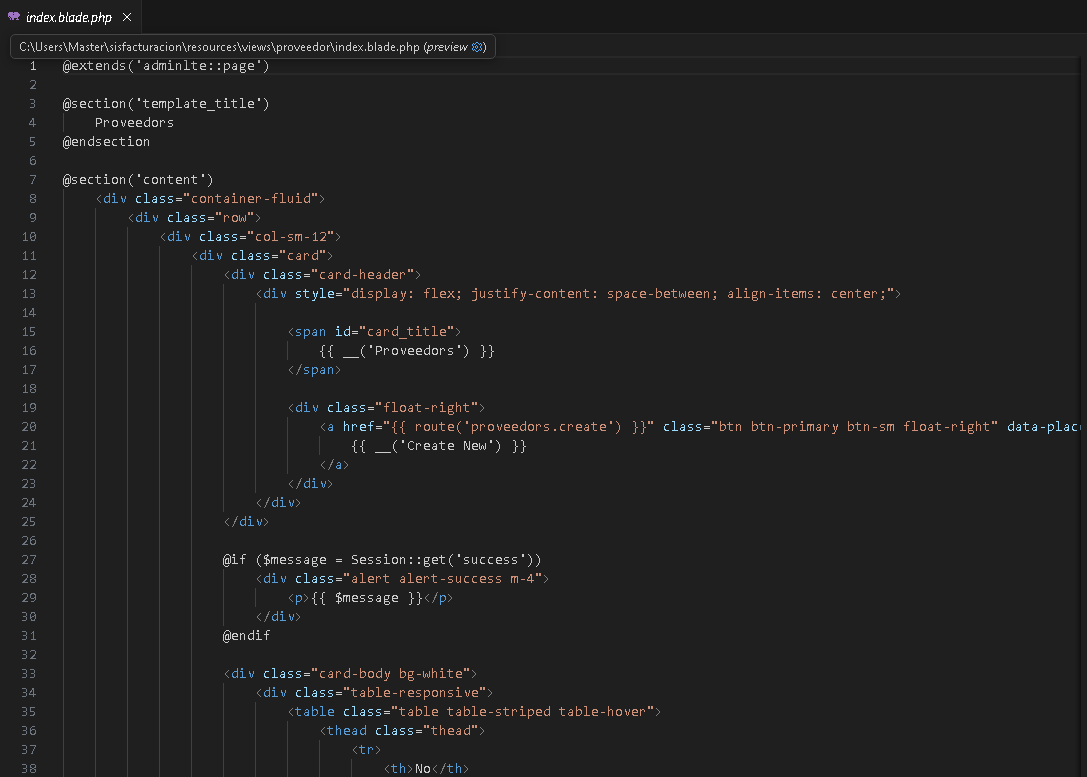






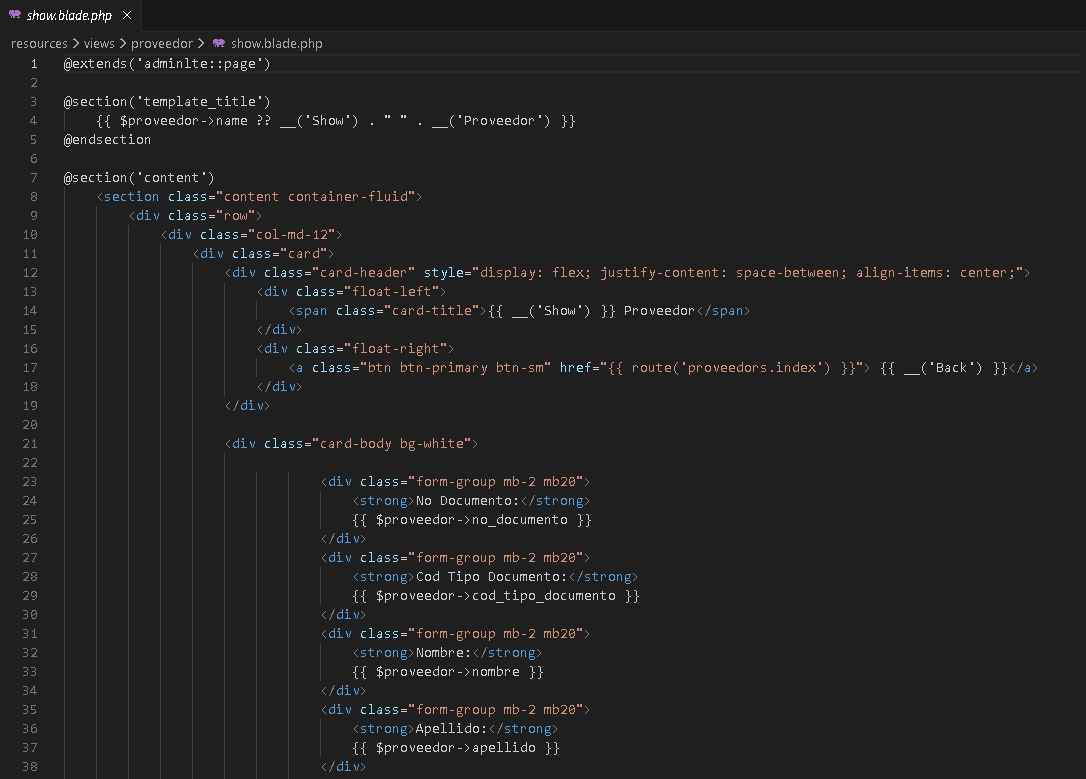


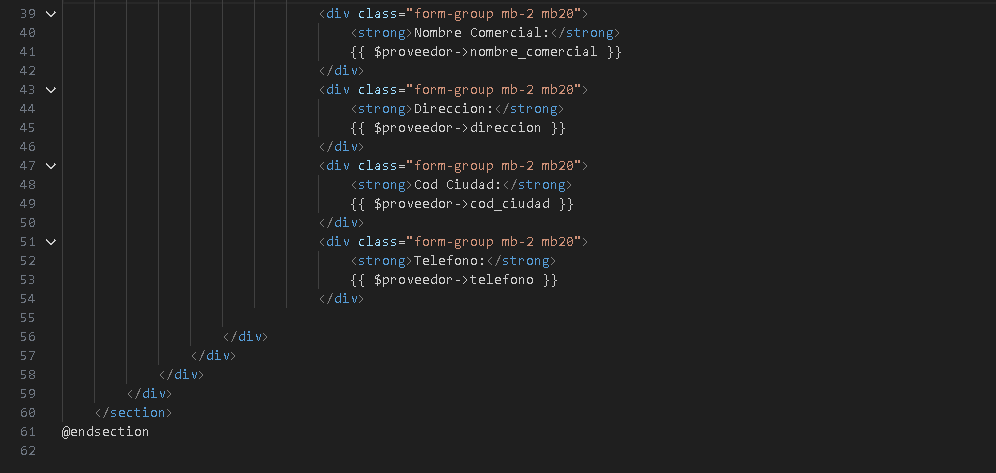




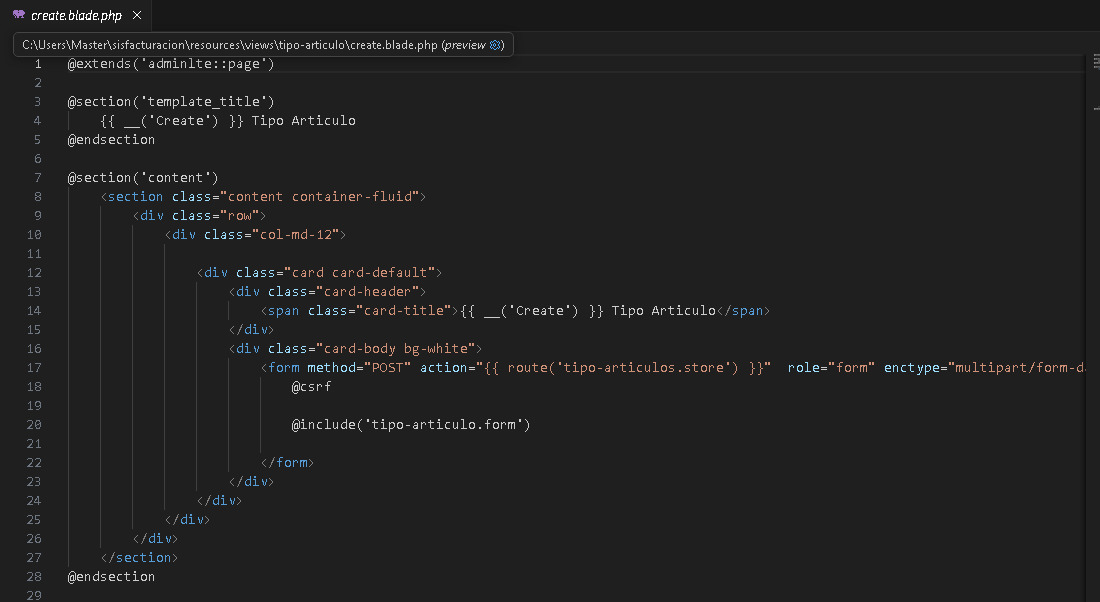


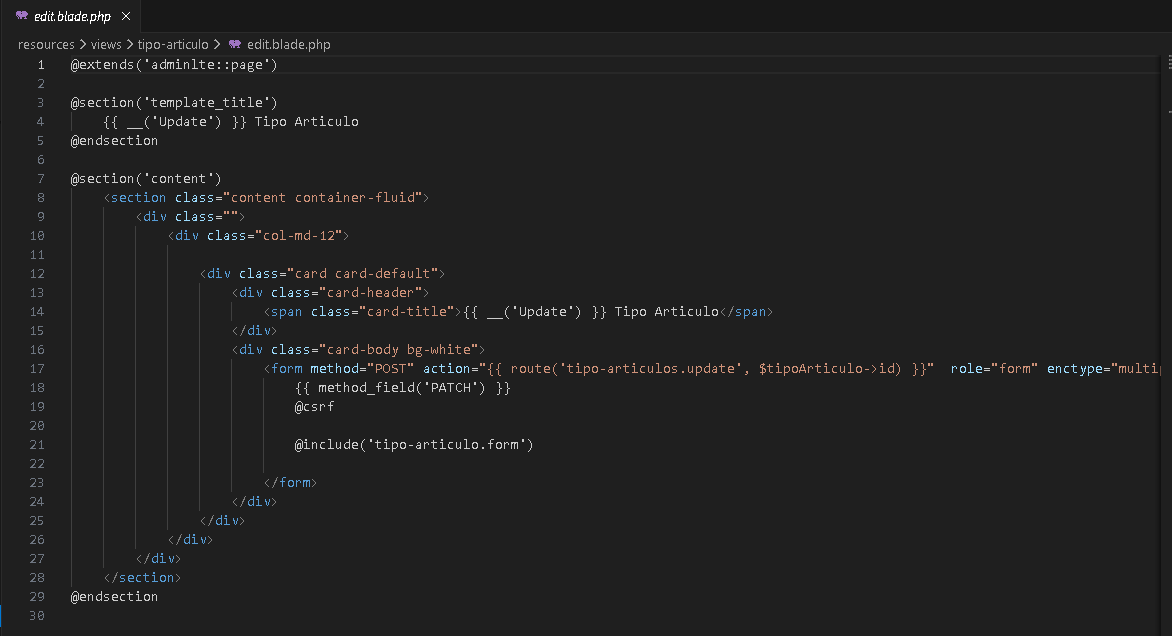


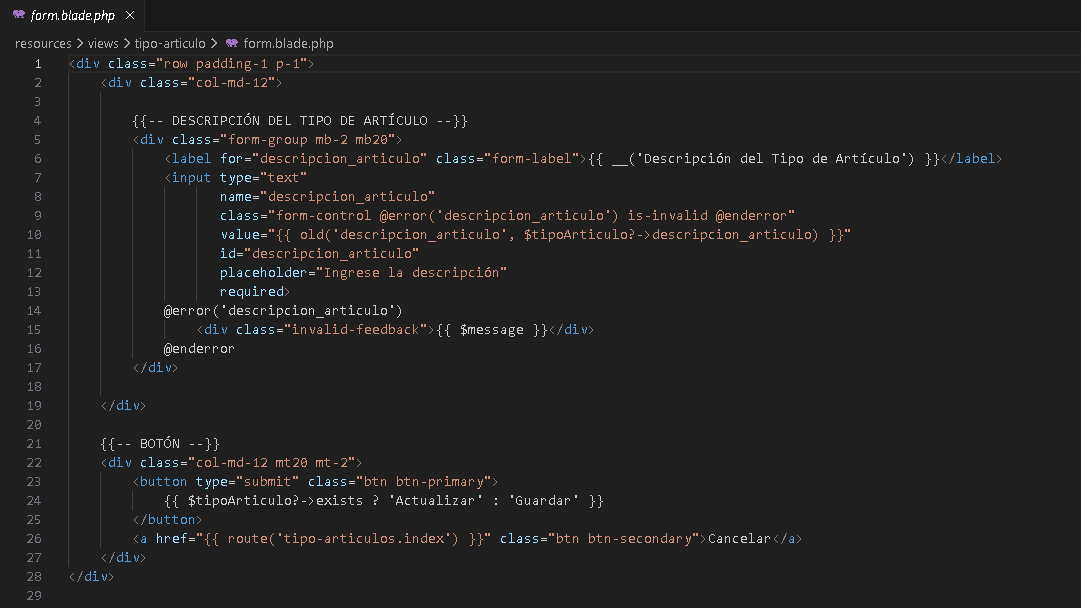


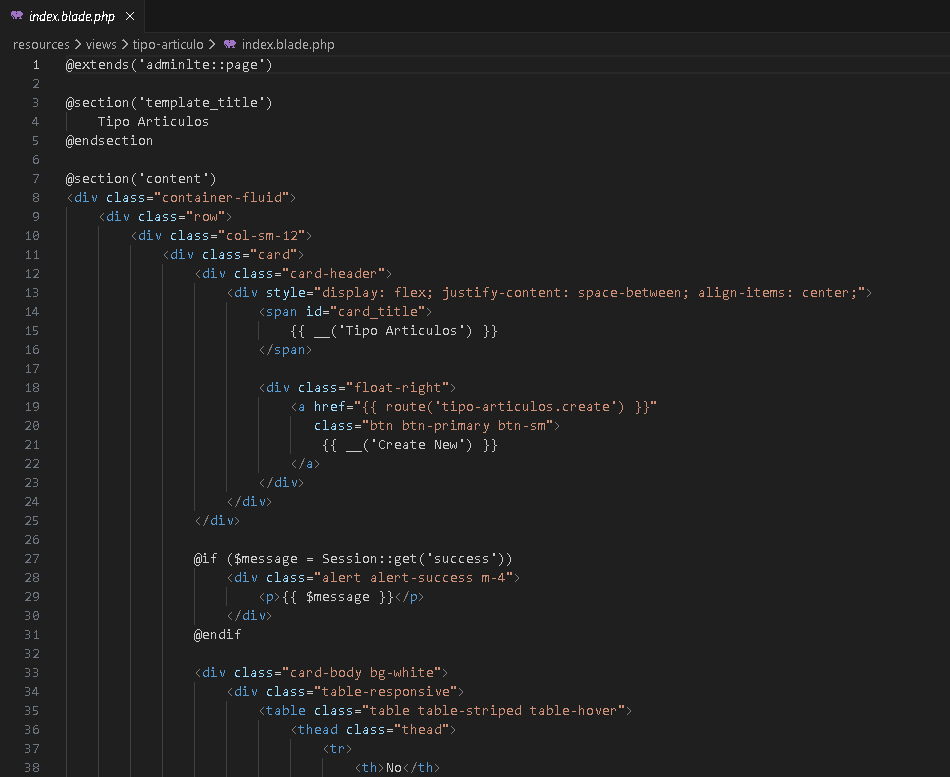


* **Tipo-articulo**

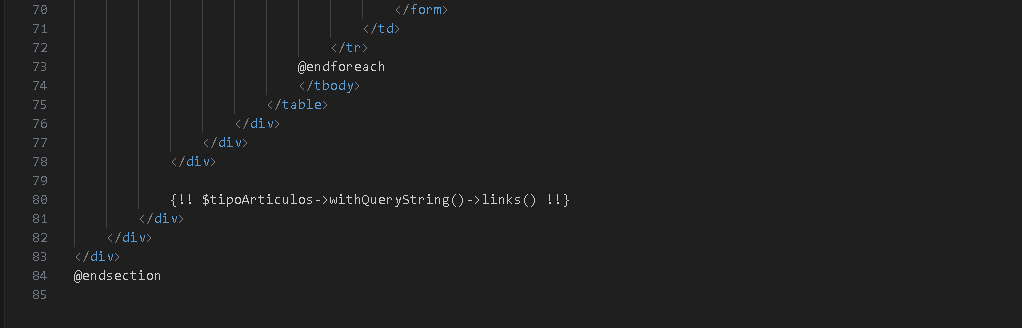


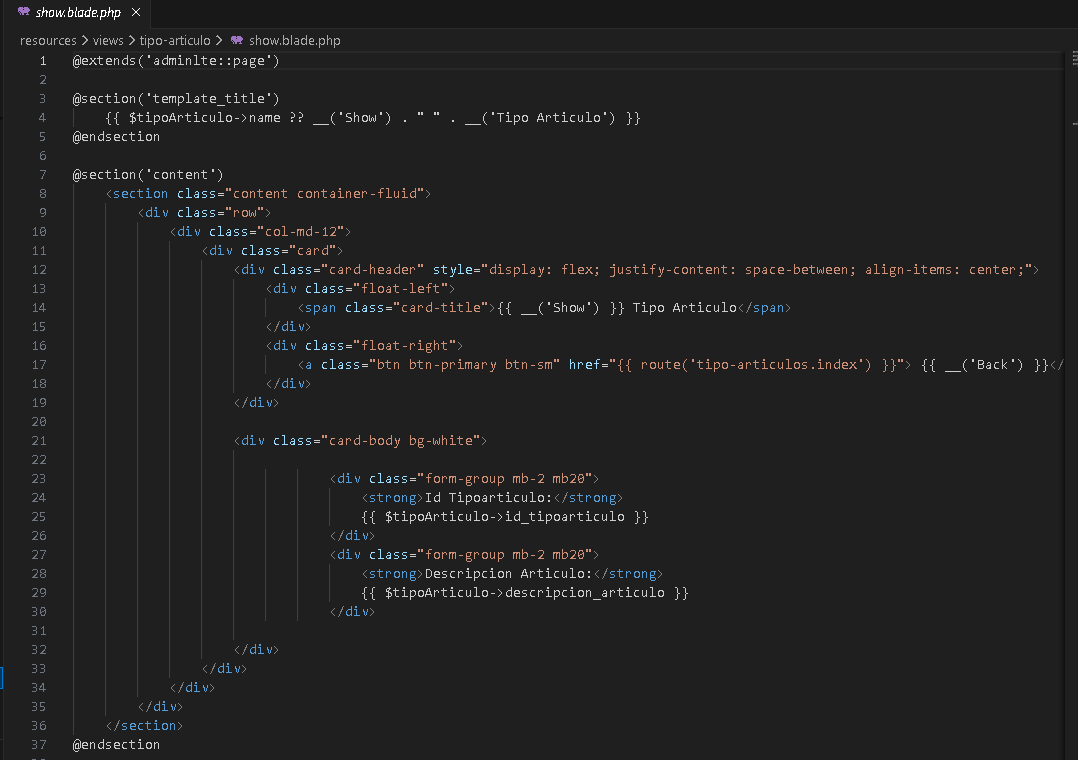




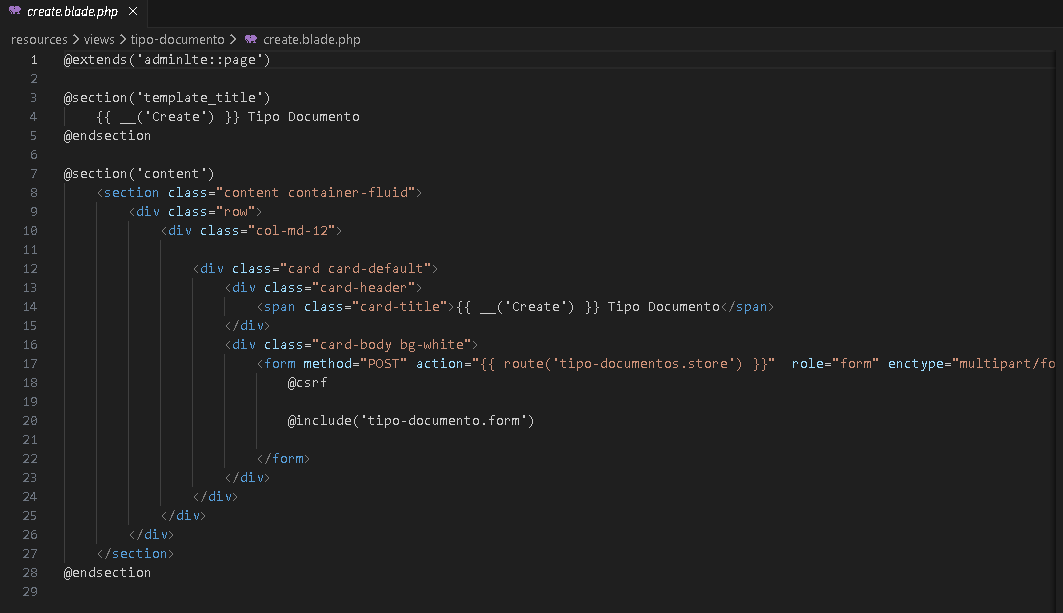


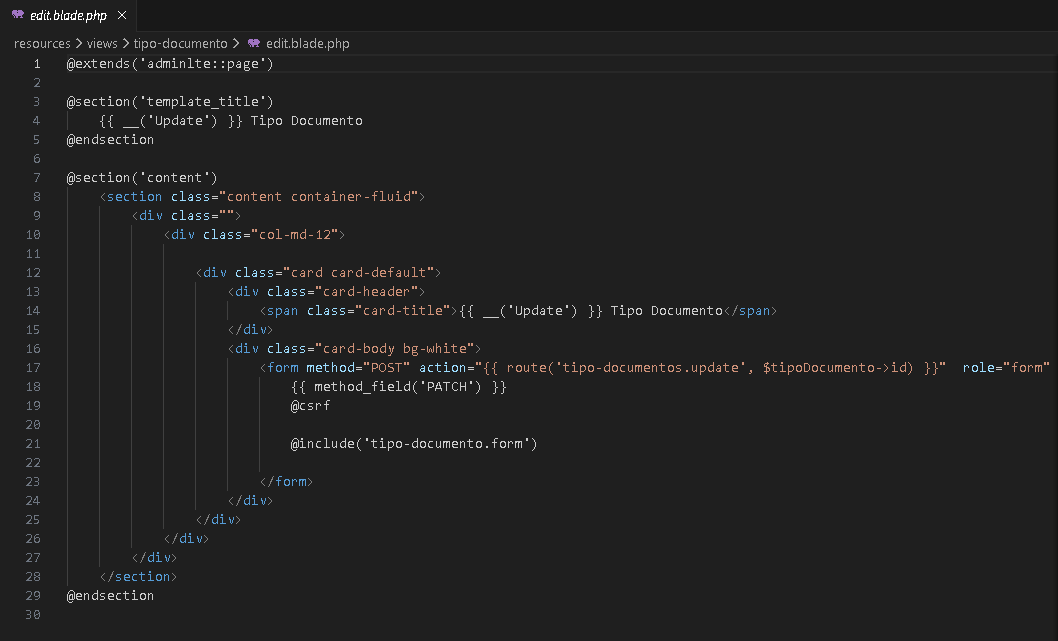


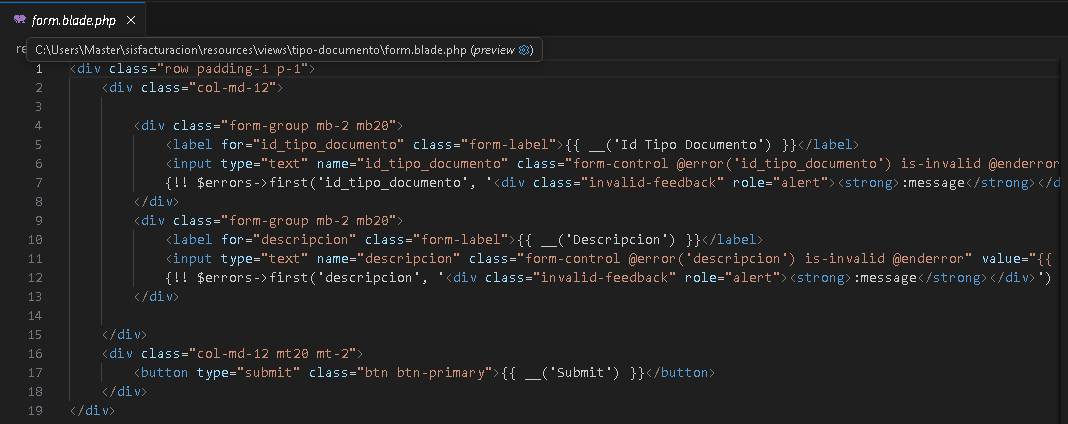


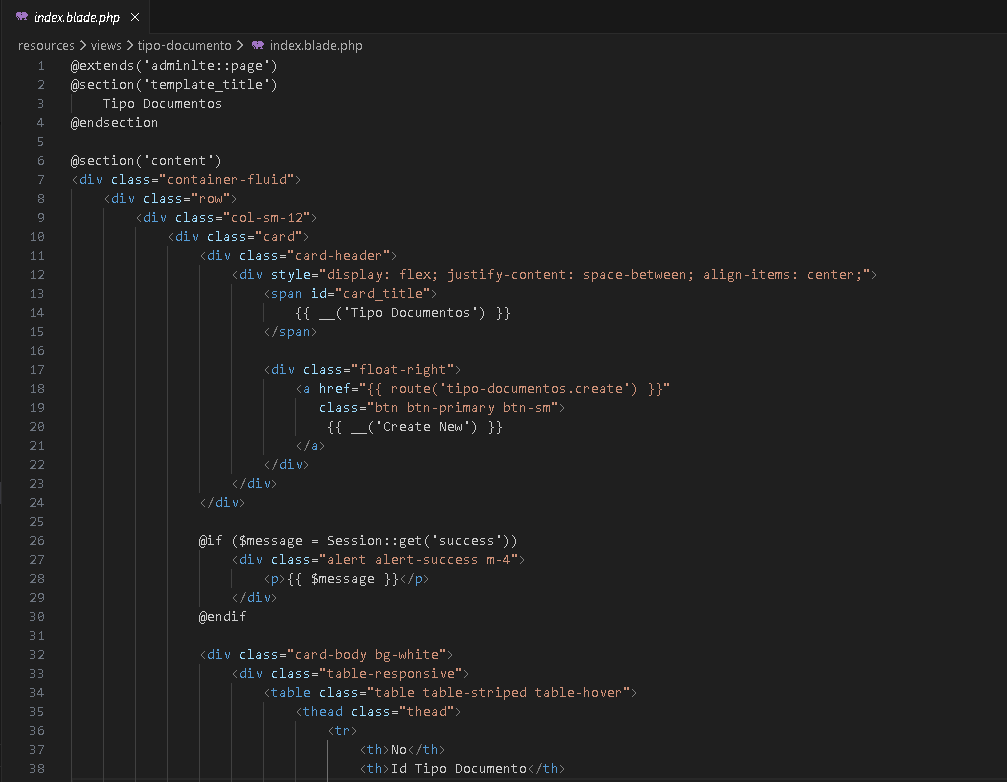


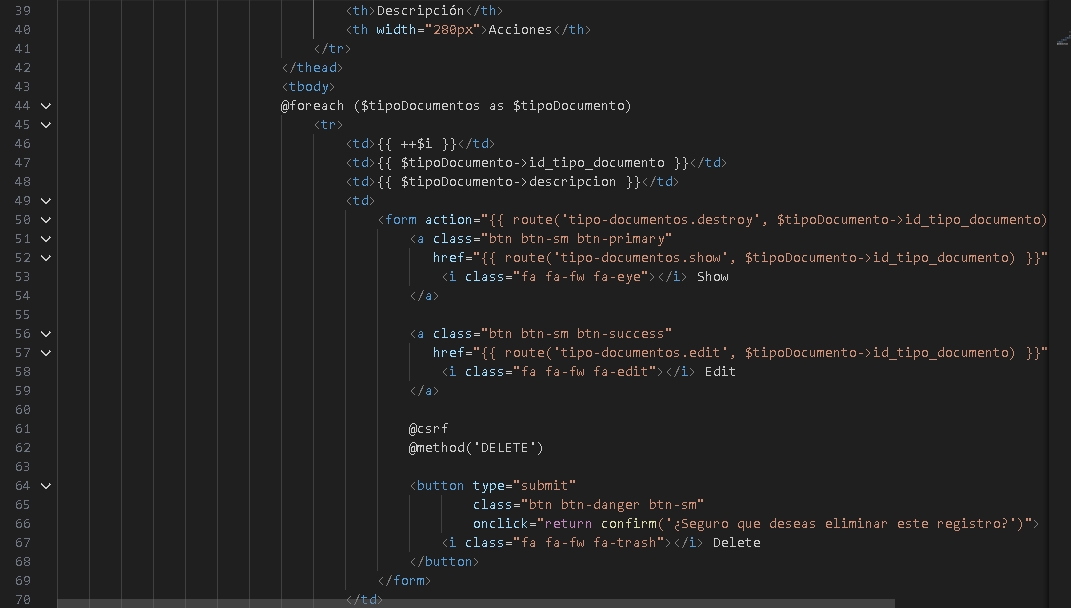
* **Tipo-documento**

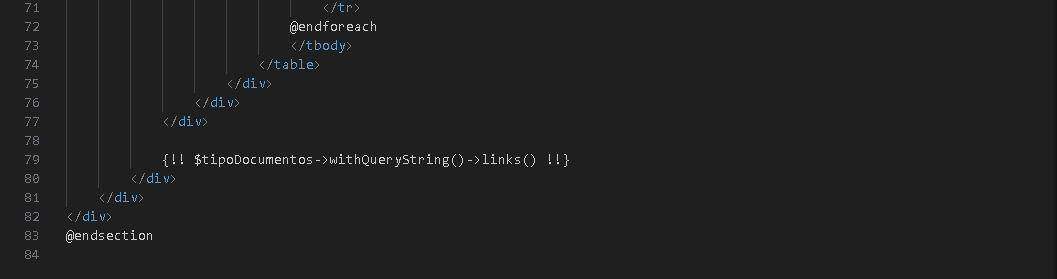






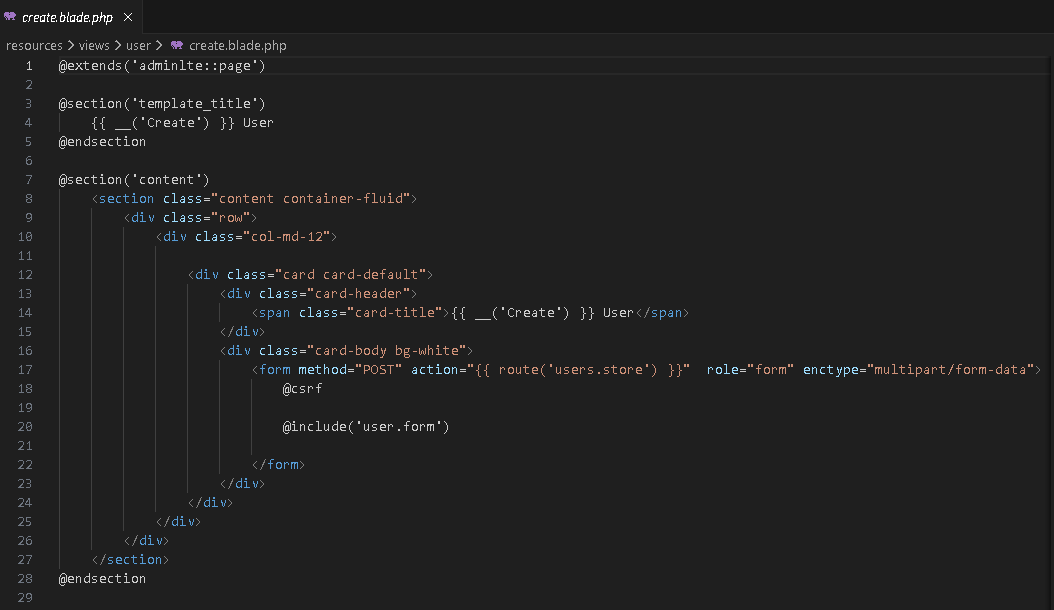


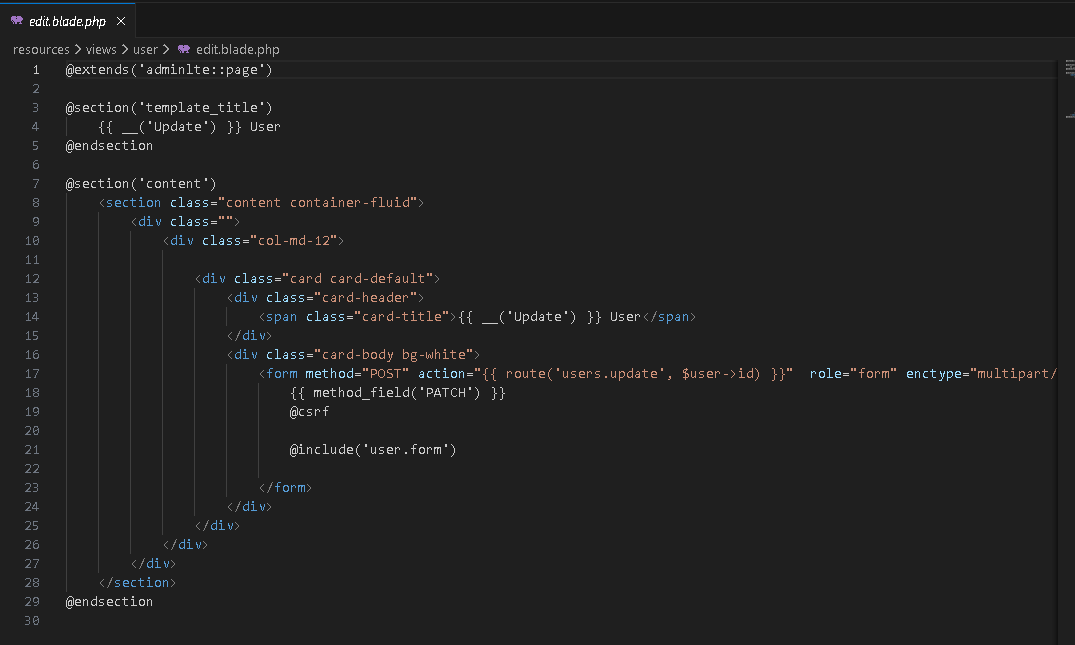


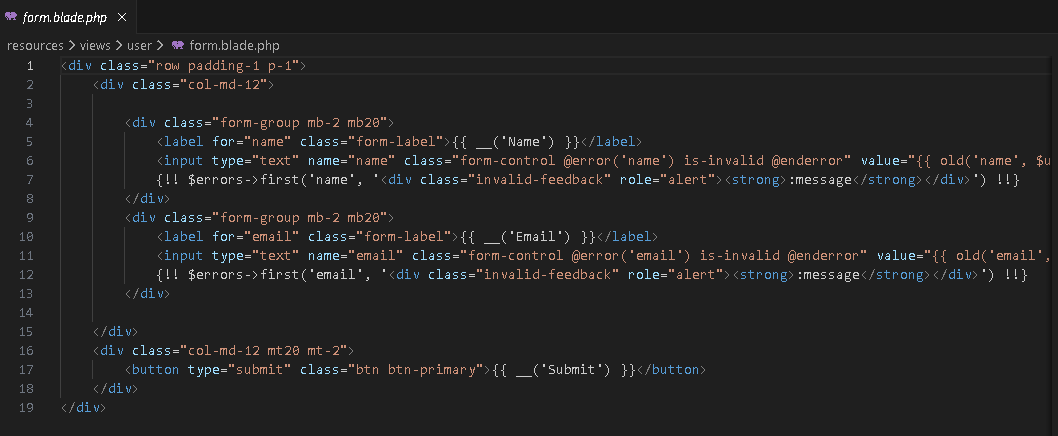


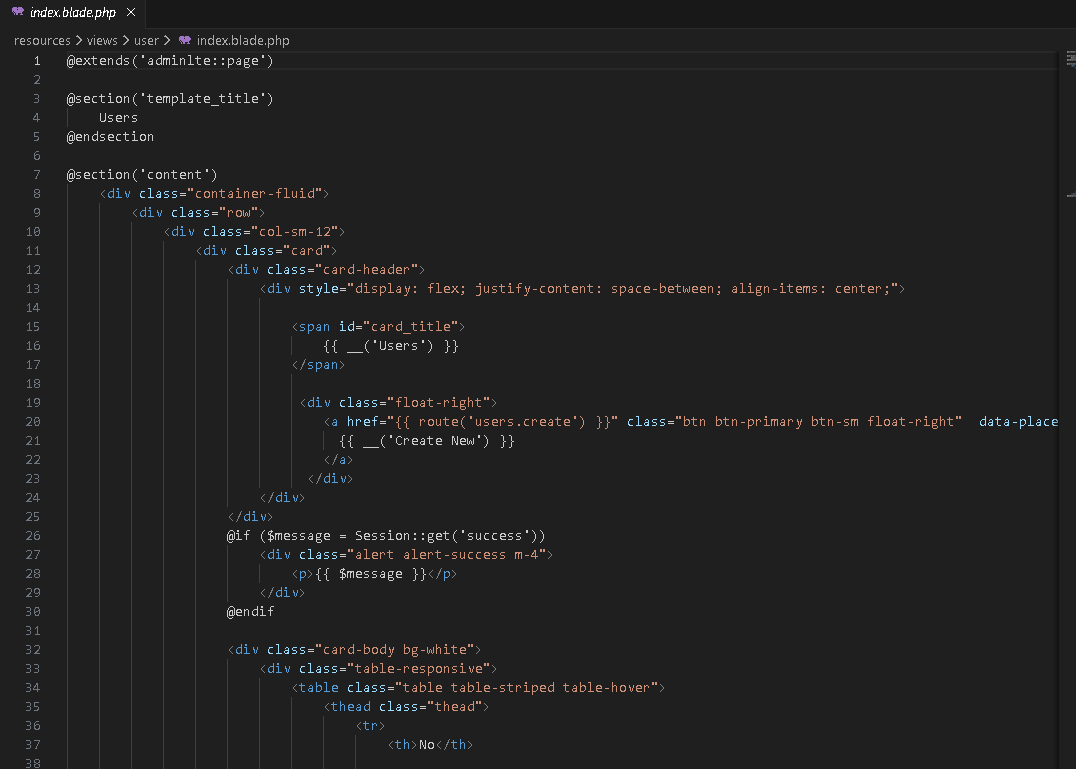


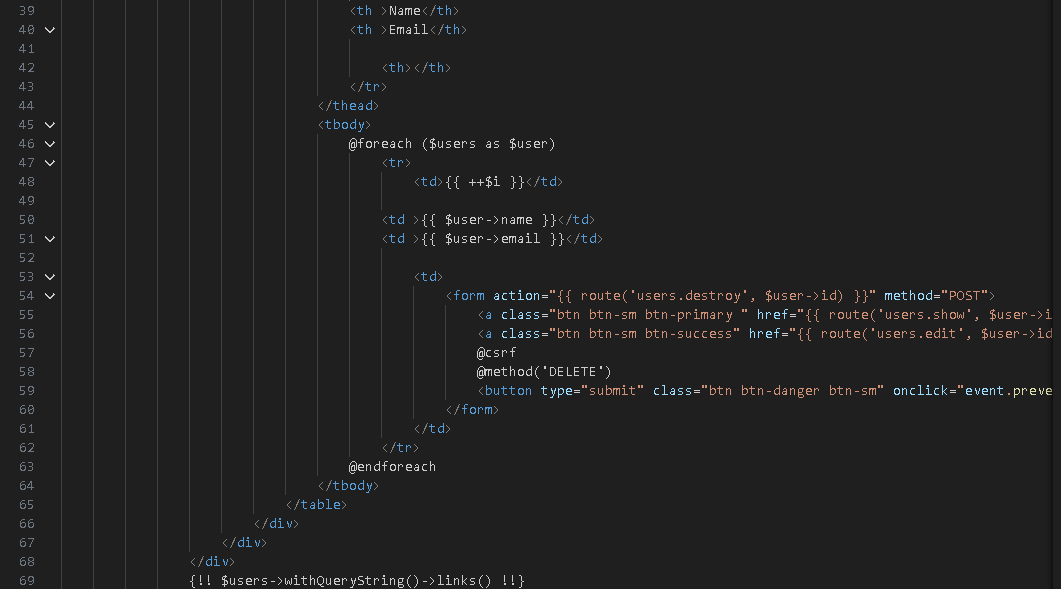
* **User**

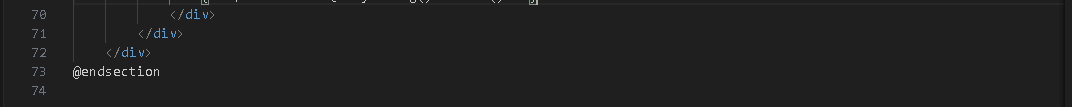


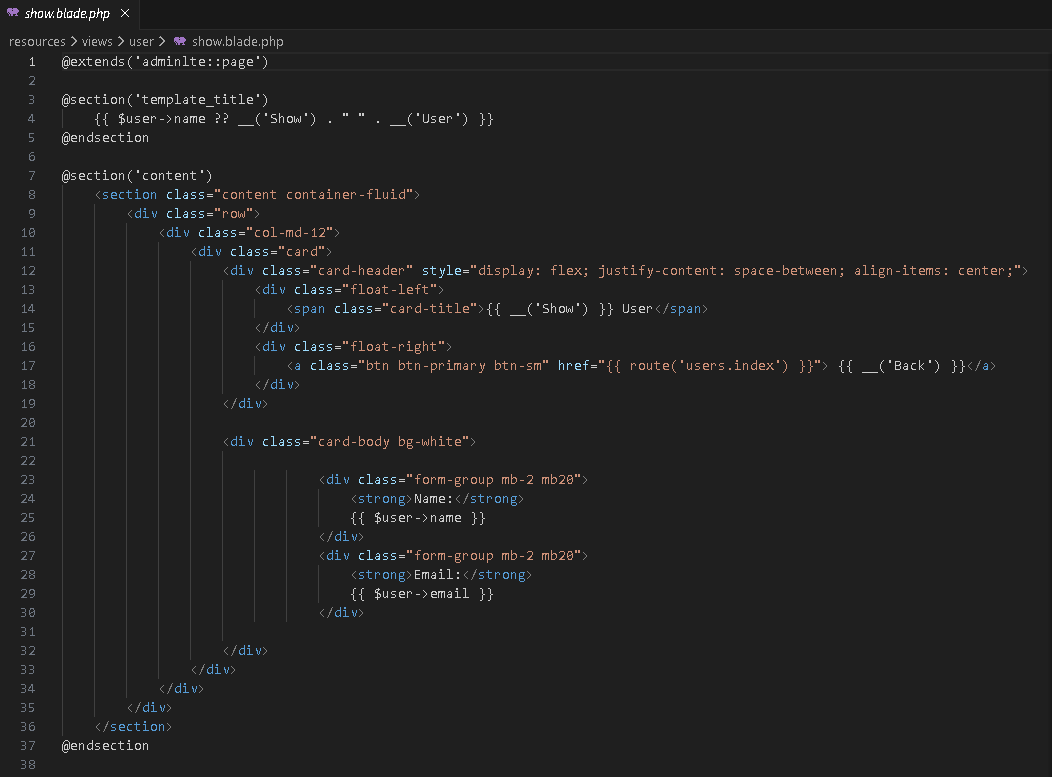




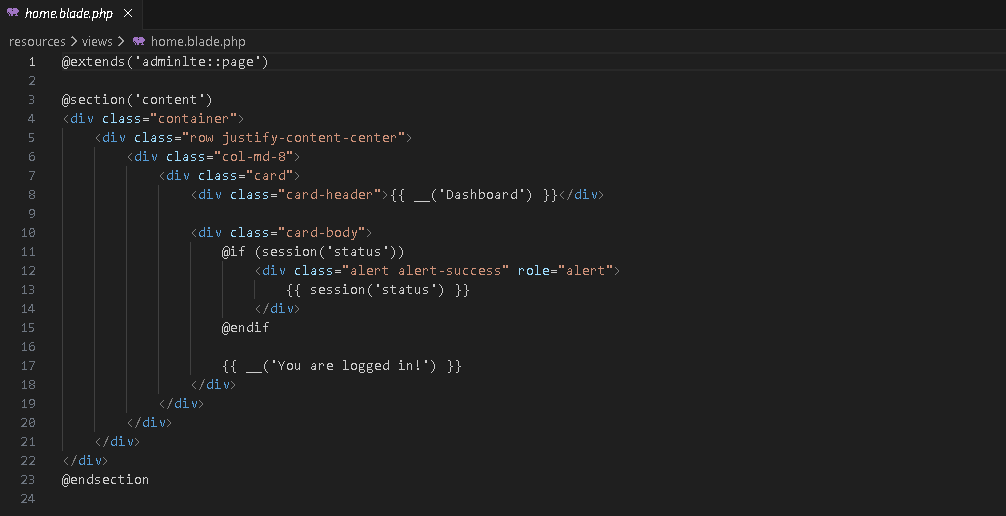




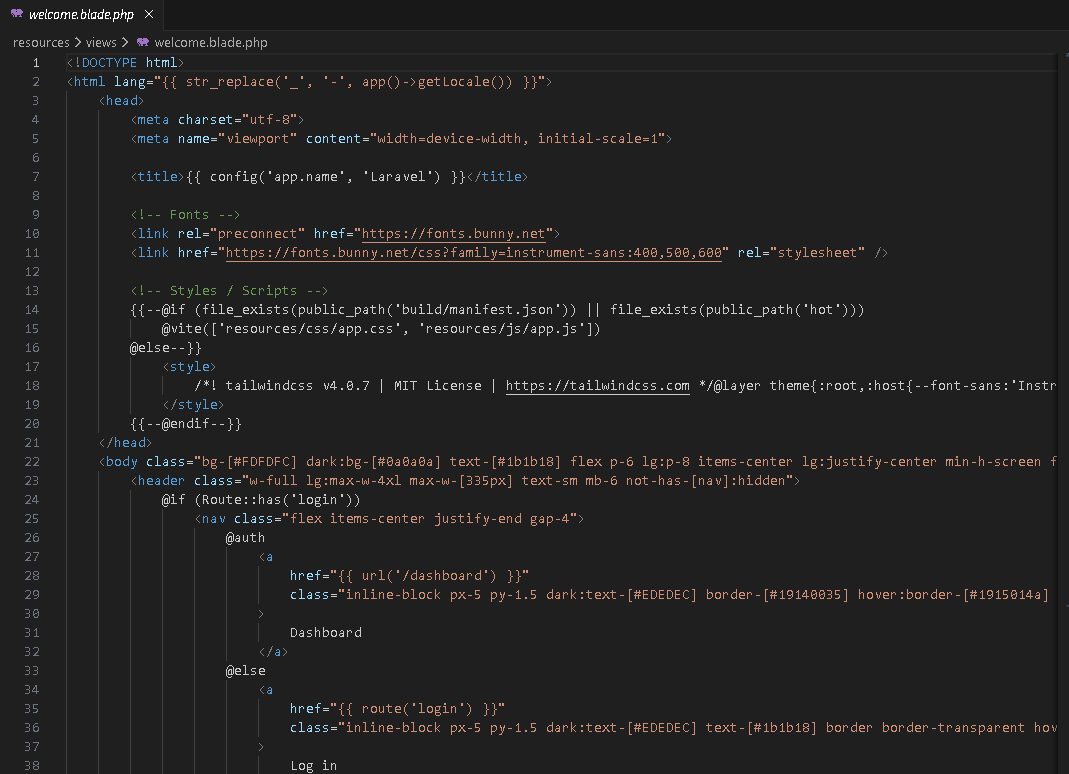


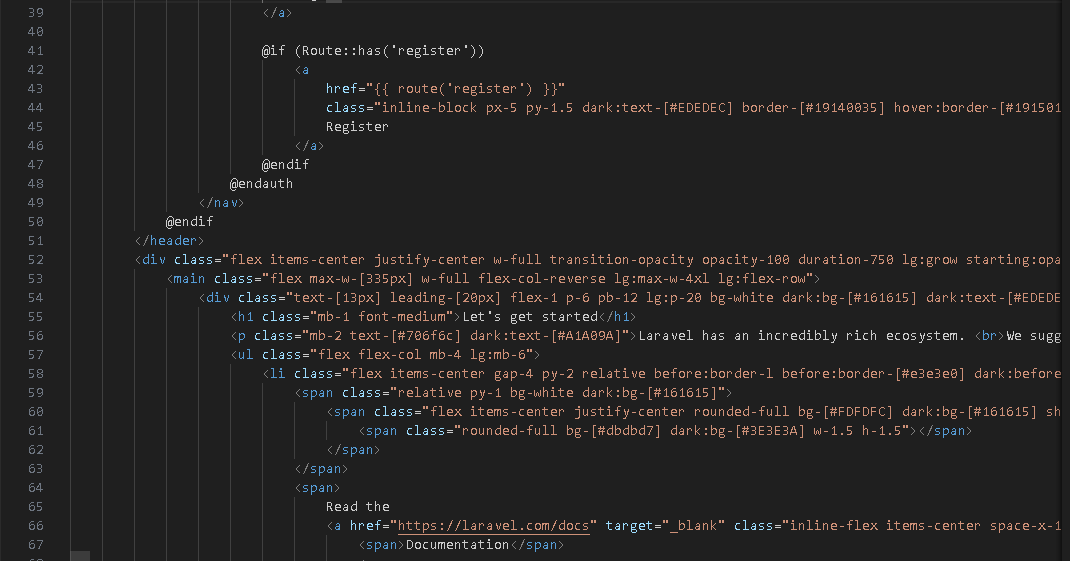


* **Home**

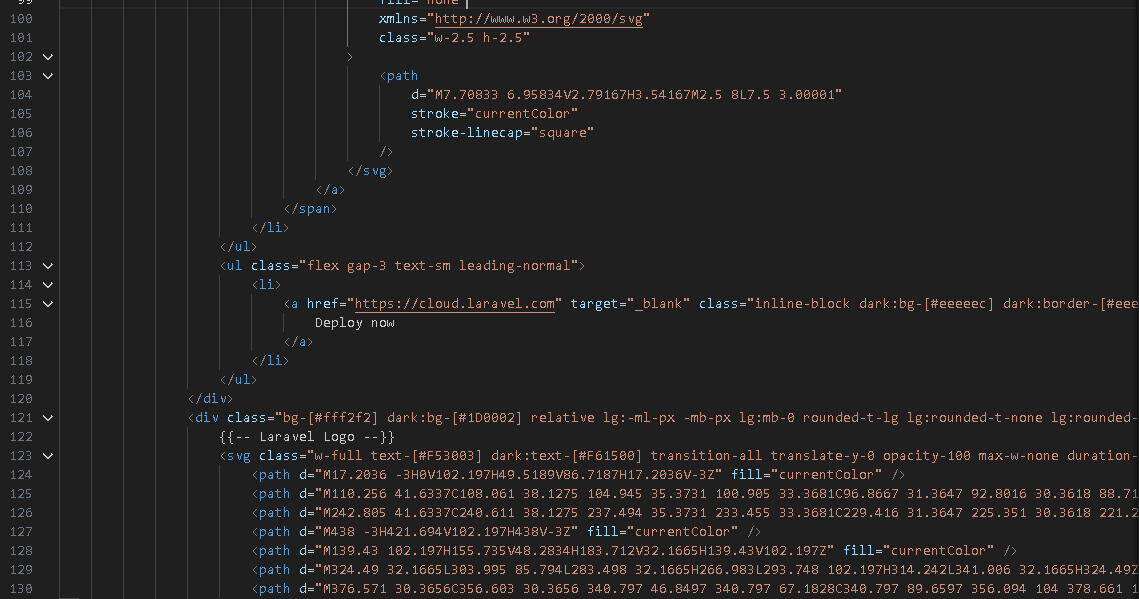


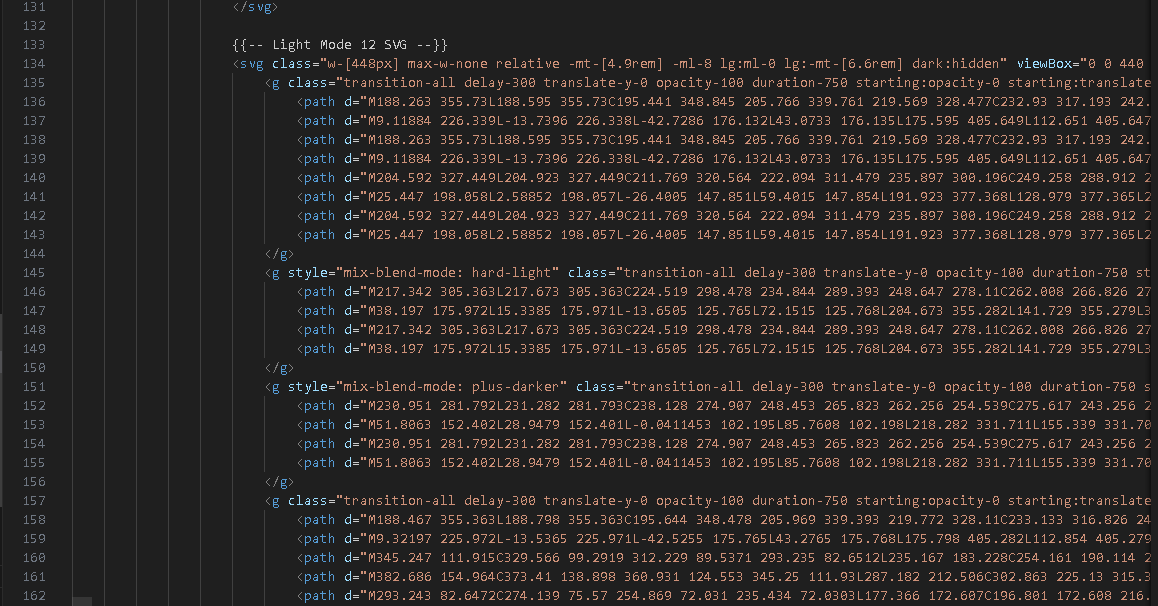
* **Welcome**

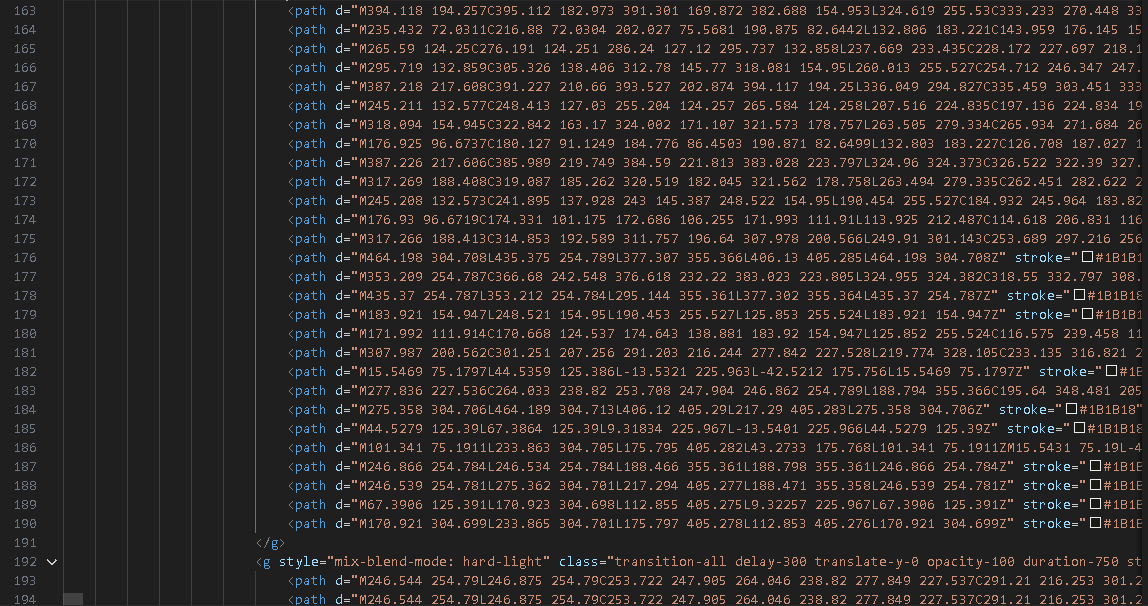


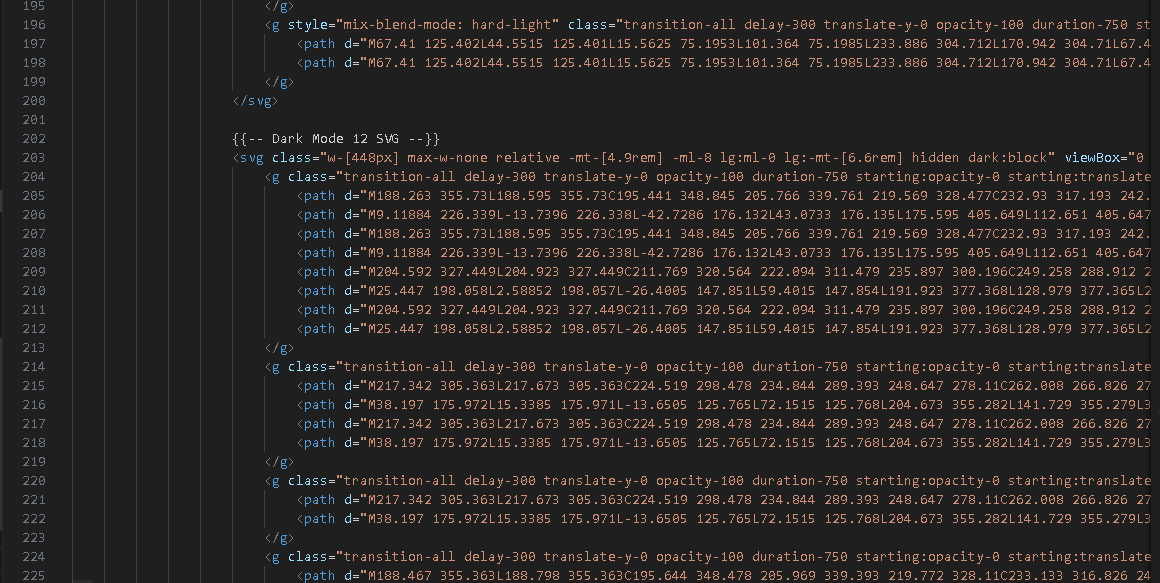


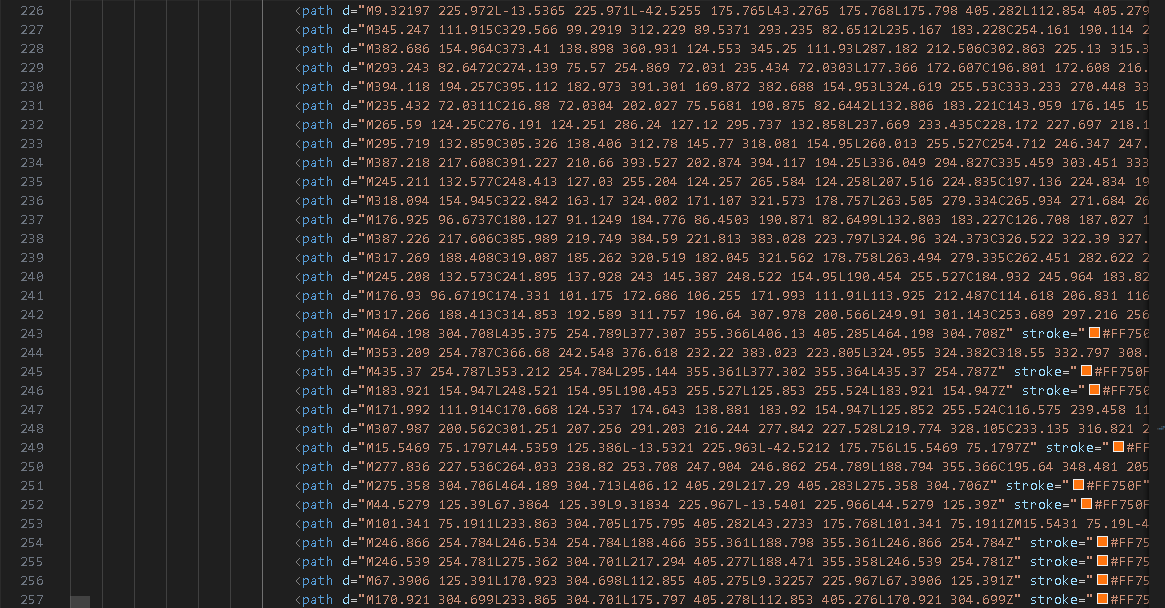


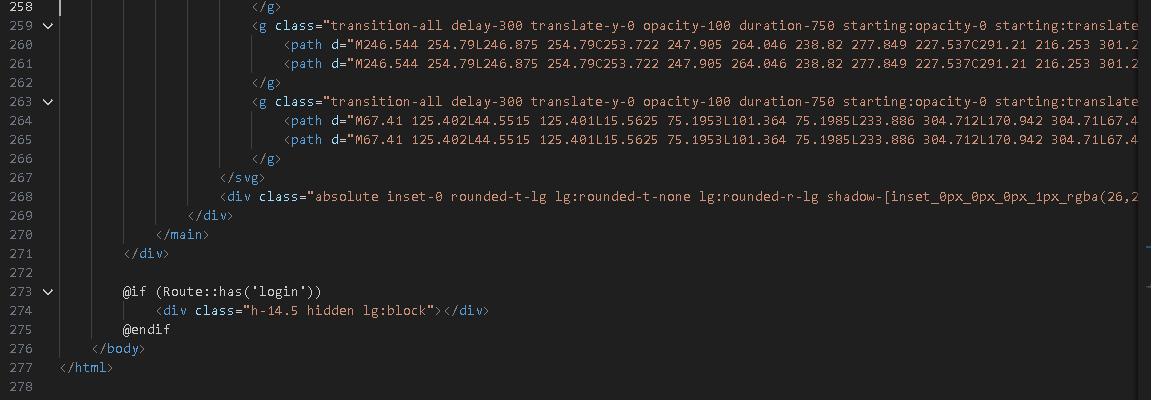




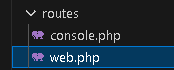


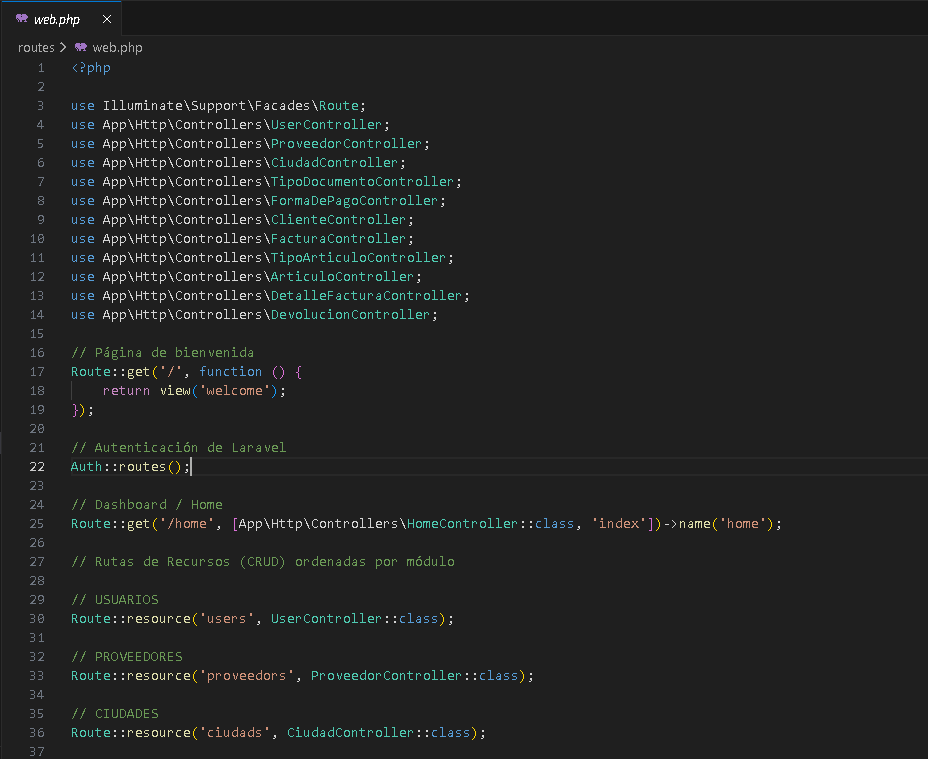






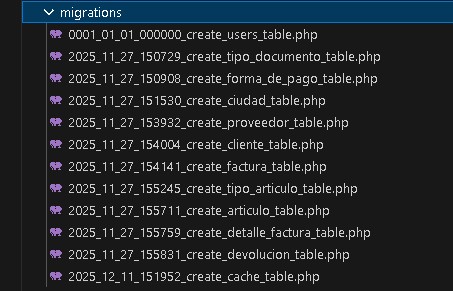
**Rutas utilizadas**

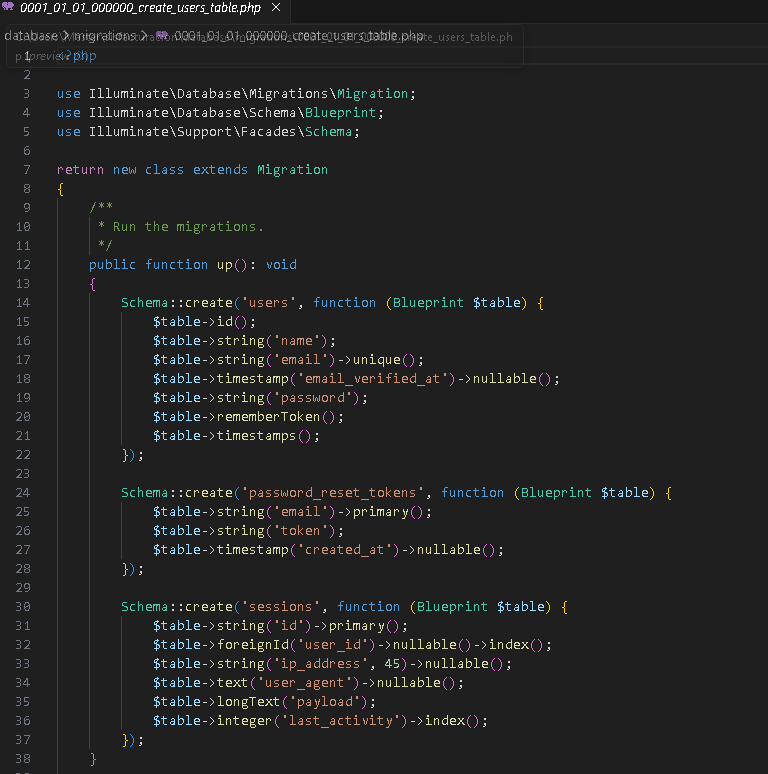


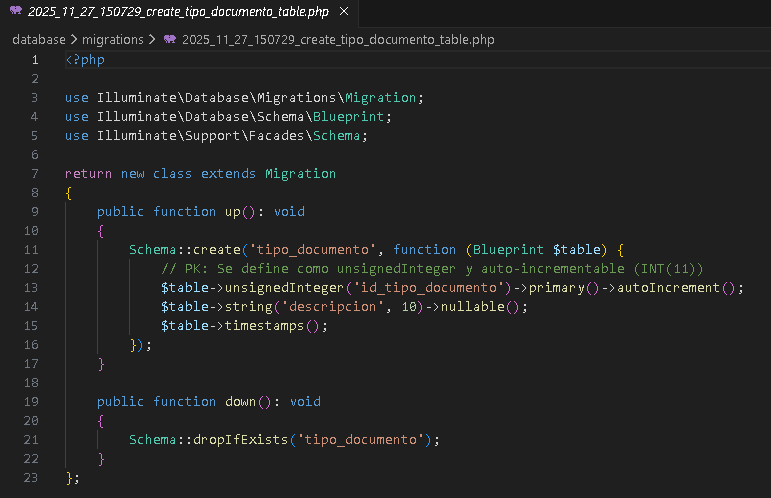


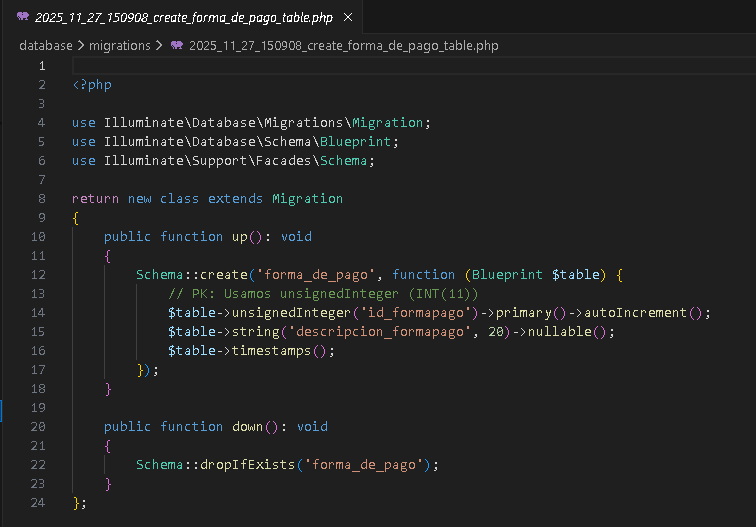


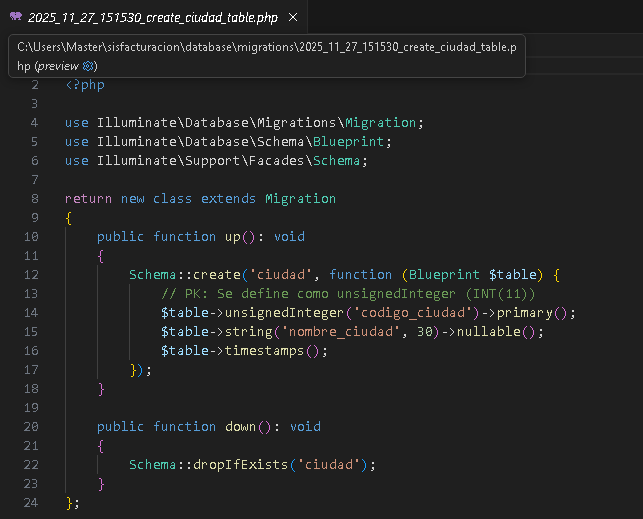
**CAPTURA DE LAS MIGRACIONES**

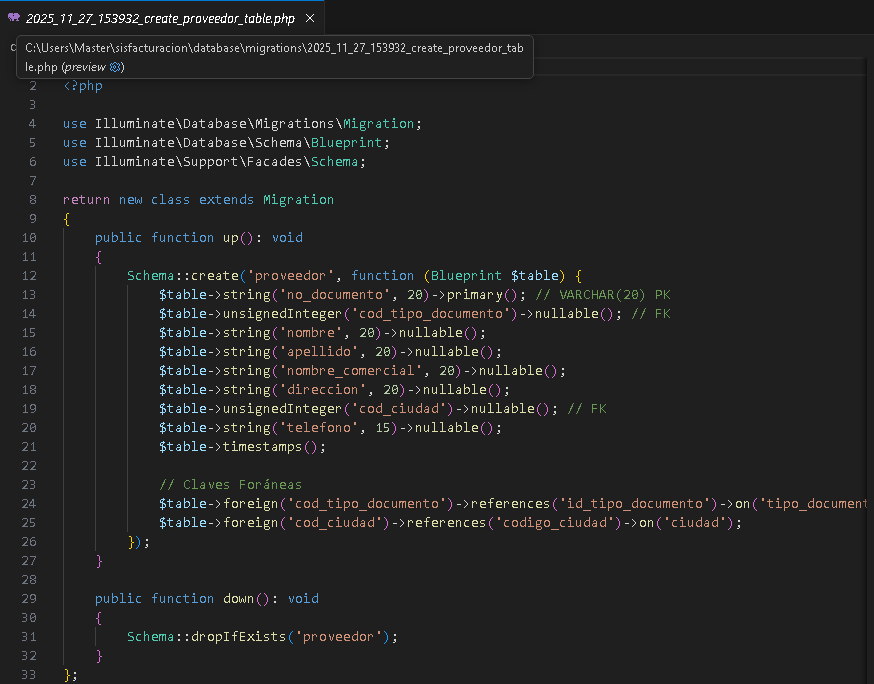


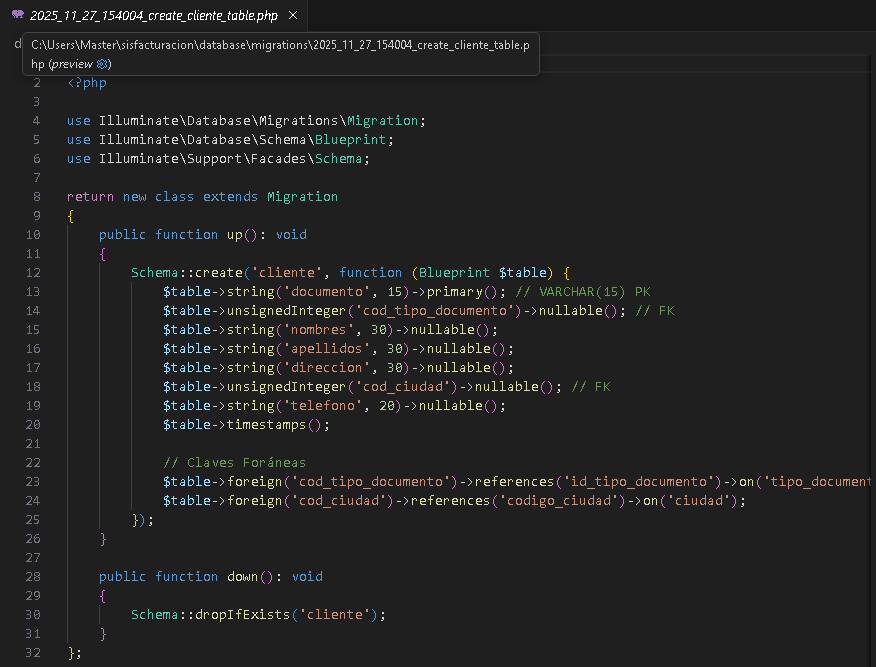


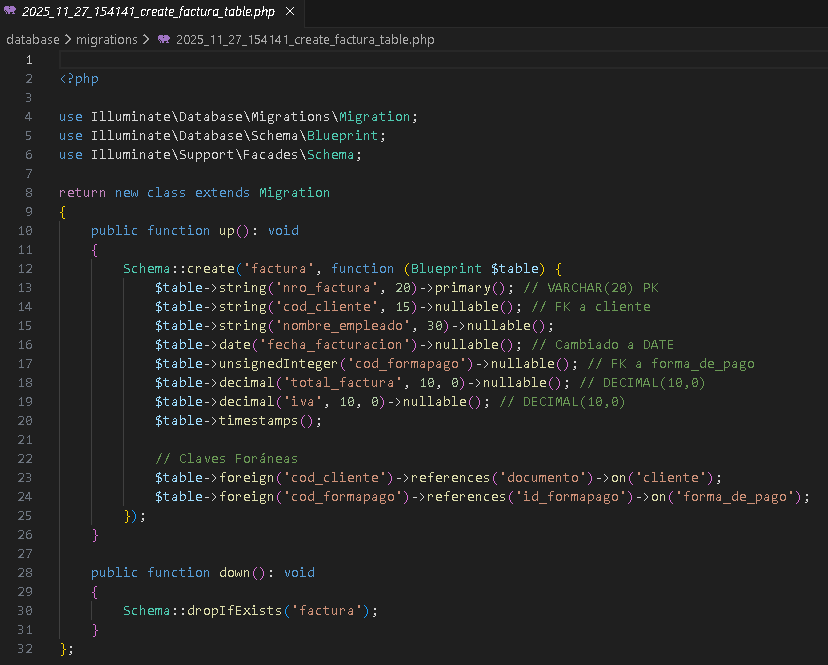


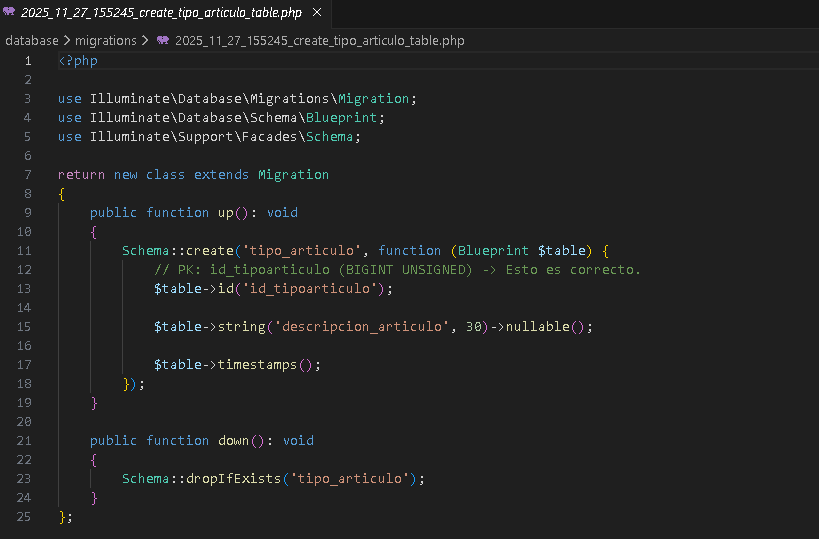


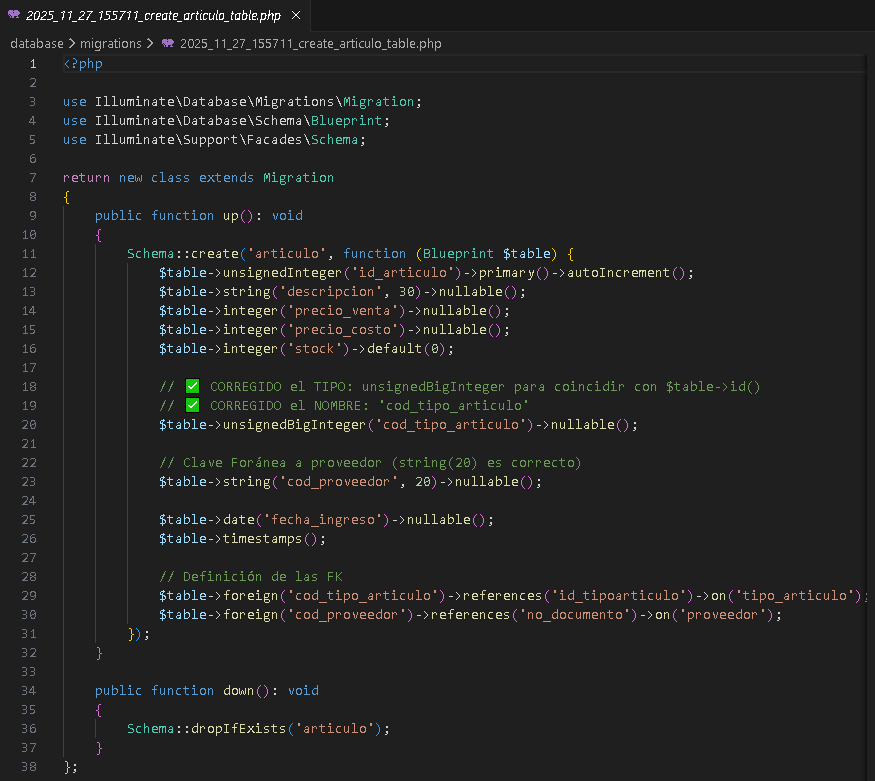


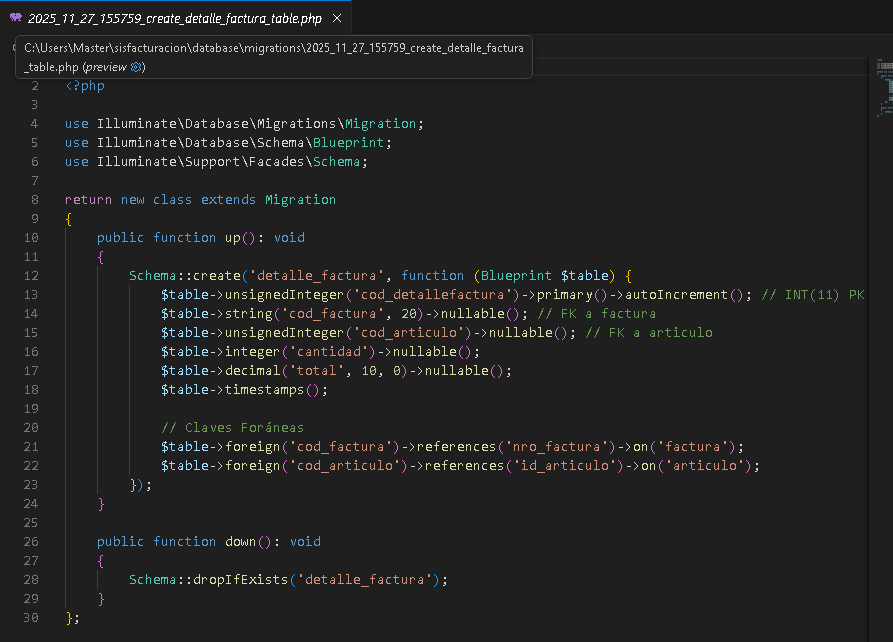


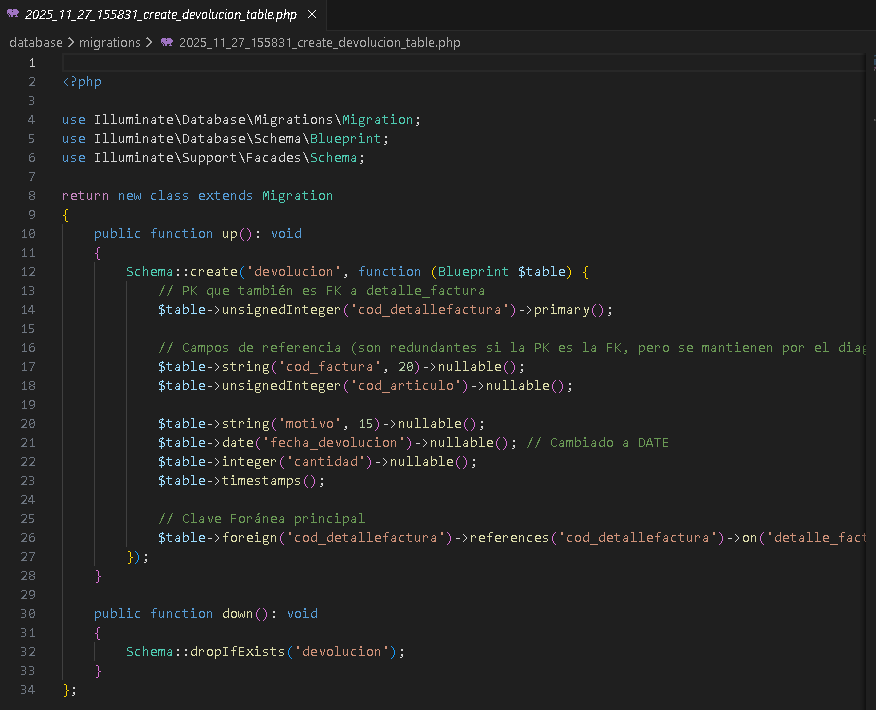


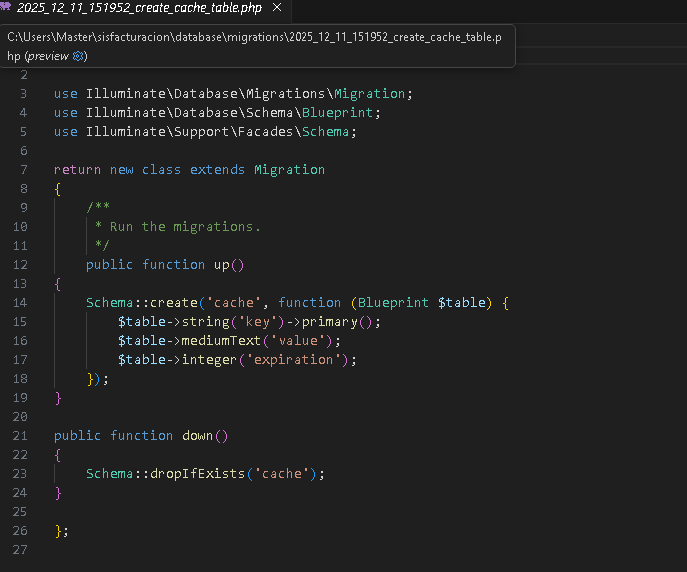




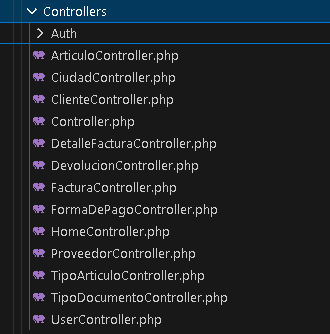








**Controladores**





<?php

namespace App\Http\Controllers;

use App\Models\Articulo;

use Illuminate\Http\RedirectResponse;

use Illuminate\Http\Request;

use App\Http\Requests\ArticuloRequest;

use Illuminate\Support\Facades\Redirect;

use Illuminate\View\View;

class ArticuloController extends Controller

{

/\*\*

\* Display a listing of the resource.

\*/

public function index(Request $request): View

{

$articulos = Articulo::paginate();

return view('articulo.index', compact('articulos'))

->with('i', ($request->input('page', 1) - 1) \* $articulos->perPage());

}

/\*\*

\* Show the form for creating a new resource.

\*/

public function create(): View

{

$articulo = new Articulo();

return view('articulo.create', compact('articulo'));

}

/\*\*

\* Store a newly created resource in storage.

\*/

public function store(ArticuloRequest $request): RedirectResponse

{

Articulo::create($request->validated());

return Redirect::route('articulos.index')

->with('success', 'Articulo created successfully.');

}

/\*\*

\* Display the specified resource.

\*/

public function show($id): View

{

$articulo = Articulo::find($id);

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

}

/\*\*

\* Show the form for editing the specified resource.

\*/

public function edit($id): View

{

$articulo = Articulo::find($id);

return view('articulo.edit', compact('articulo'));

}

/\*\*

\* Update the specified resource in storage.

\*/

public function update(ArticuloRequest $request, Articulo $articulo): RedirectResponse

{

$articulo->update($request->validated());

return Redirect::route('articulos.index')

->with('success', 'Articulo updated successfully');

}

public function destroy($id): RedirectResponse

{

Articulo::find($id)->delete();

return Redirect::route('articulos.index')

->with('success', 'Articulo deleted successfully');

}

}



<?php

namespace App\Http\Controllers;

use App\Models\Ciudad;

use Illuminate\Http\RedirectResponse;

use Illuminate\Http\Request;

use App\Http\Requests\CiudadRequest;

use Illuminate\Support\Facades\Redirect;

use Illuminate\View\View;

class CiudadController extends Controller

{

/\*\*

\* Display a listing of the resource.

\*/

public function index(Request $request): View

{

$ciudads = Ciudad::paginate();

return view('ciudad.index', compact('ciudads'))

->with('i', ($request->input('page', 1) - 1) \* $ciudads->perPage());

}

/\*\*

\* Show the form for creating a new resource.

\*/

public function create(): View

{

$ciudad = new Ciudad();

return view('ciudad.create', compact('ciudad'));

}

/\*\*

\* Store a newly created resource in storage.

\*/

public function store(CiudadRequest $request): RedirectResponse

{

Ciudad::create($request->validated());

return Redirect::route('ciudads.index')

->with('success', 'Ciudad created successfully.');

}

/\*\*

\* Display the specified resource.

\*/

public function show($id): View

{

$ciudad = Ciudad::find($id);

return view('ciudad.show', compact('ciudad'));

}

/\*\*

\* Show the form for editing the specified resource.

\*/

public function edit($id): View

{

$ciudad = Ciudad::find($id);

return view('ciudad.edit', compact('ciudad'));

}

/\*\*

\* Update the specified resource in storage.

\*/

public function update(CiudadRequest $request, Ciudad $ciudad): RedirectResponse

{

$ciudad->update($request->validated());

return Redirect::route('ciudads.index')

->with('success', 'Ciudad updated successfully');

}

public function destroy($id): RedirectResponse

{

Ciudad::find($id)->delete();

return Redirect::route('ciudads.index')

->with('success', 'Ciudad deleted successfully');

}

}



<?php

namespace App\Http\Controllers;

use App\Models\Cliente;

use App\Models\TipoDocumento;

use App\Models\Ciudad;

use Illuminate\Http\RedirectResponse;

use Illuminate\Http\Request;

use App\Http\Requests\ClienteRequest;

use Illuminate\Support\Facades\Redirect;

use Illuminate\View\View;

class ClienteController extends Controller

{

/\*\*

\* Display a listing of the resource.

\*/

public function index(Request $request): View

{

$busqueda = $request->get('busqueda');

$clientes = Cliente::with(['tipoDocumento', 'ciudad'])

->when($busqueda, function ($query, $busqueda) {

$query->where('documento', 'LIKE', "%$busqueda%")

->orWhere('nombres', 'LIKE', "%$busqueda%")

->orWhere('apellidos', 'LIKE', "%$busqueda%");

})

->paginate(10)

->withQueryString();

return view('cliente.index', compact('clientes'))

->with('i', ($request->input('page', 1) - 1) \* $clientes->perPage());

}

/\*\*

\* Show the form for creating a new resource.

\*/

public function create(): View

{

$cliente = new Cliente();

$tiposDocumentos = TipoDocumento::pluck('descripcion', 'id\_tipo\_documento');

$ciudades = Ciudad::pluck('nombre\_ciudad', 'codigo\_ciudad');

return view('cliente.create', compact(

'cliente',

'tiposDocumentos',

'ciudades'

));

}

/\*\*

\* Store a newly created resource in storage.

\*/

public function store(ClienteRequest $request): RedirectResponse

{

Cliente::create($request->validated());

return Redirect::route('clientes.index')

->with('success', 'Cliente creado correctamente.');

}

/\*\*

\* Display the specified resource.

\*/

public function show(Cliente $cliente): View

{

return view('cliente.show', compact('cliente'));

}

/\*\*

\* Show the form for editing the specified resource.

\*/

public function edit(Cliente $cliente): View

{

$tiposDocumentos = TipoDocumento::pluck('descripcion', 'id\_tipo\_documento');

$ciudades = Ciudad::pluck('nombre\_ciudad', 'codigo\_ciudad');

return view('cliente.edit', compact(

'cliente',

'tiposDocumentos',

'ciudades'

));

}

/\*\*

\* Update the specified resource in storage.

\*/

public function update(ClienteRequest $request, Cliente $cliente): RedirectResponse

{

$cliente->update($request->validated());

return Redirect::route('clientes.index')

->with('success', 'Cliente actualizado correctamente.');

}

/\*\*

\* Remove the specified resource from storage.

\*/

public function destroy(Cliente $cliente): RedirectResponse

{

$cliente->delete();

return Redirect::route('clientes.index')

->with('success', 'Cliente eliminado correctamente.');

}

}



<?php

namespace App\Http\Controllers;

use Illuminate\Foundation\Auth\Access\AuthorizesRequests;

use Illuminate\Foundation\Validation\ValidatesRequests;

use Illuminate\Routing\Controller as BaseController;

class Controller extends BaseController

{

use AuthorizesRequests, ValidatesRequests;

}



<?php

namespace App\Http\Controllers;

use App\Models\DetalleFactura;

use App\Models\Factura;

use App\Models\Articulo;

use Illuminate\Http\Request;

use Illuminate\Http\RedirectResponse;

use Illuminate\Support\Facades\Redirect;

use Illuminate\View\View;

class DetalleFacturaController extends Controller

{

public function index(Request $request): View

{

$detalleFacturas = DetalleFactura::with(['factura', 'articulo'])->paginate(10);

return view('detalle-factura.index', compact('detalleFacturas'))

->with('i', ($request->input('page', 1) - 1) \* $detalleFacturas->perPage());

}

public function create(): View

{

$detalleFactura = new DetalleFactura();

$facturas = Factura::pluck('nro\_factura', 'nro\_factura');

// Pasamos descripcion y precio\_venta de artículos

$articulos = Articulo::all()->mapWithKeys(function($art){

return [$art->id\_articulo => ['descripcion' => $art->descripcion, 'precio' => $art->precio\_venta]];

});

return view('detalle-factura.create', compact('detalleFactura', 'facturas', 'articulos'));

}

public function store(Request $request): RedirectResponse

{

$request->validate([

'cod\_factura' => 'required',

'cod\_articulo' => 'required',

'cantidad' => 'required|numeric|min:1',

]);

$articulo = Articulo::find($request->cod\_articulo);

$total = $request->cantidad \* $articulo->precio\_venta;

DetalleFactura::create([

'cod\_factura' => $request->cod\_factura,

'cod\_articulo' => $request->cod\_articulo,

'cantidad' => $request->cantidad,

'total' => $total

]);

return Redirect::route('detalle-facturas.index')

->with('success', 'Detalle de factura registrado correctamente.');

}

public function show(DetalleFactura $detalleFactura): View

{

return view('detalle-factura.show', compact('detalleFactura'));

}

public function edit(DetalleFactura $detalleFactura): View

{

$facturas = Factura::pluck('nro\_factura', 'nro\_factura');

$articulos = Articulo::all()->mapWithKeys(function($art){

return [$art->id\_articulo => ['descripcion' => $art->descripcion, 'precio' => $art->precio\_venta]];

});

return view('detalle-factura.edit', compact('detalleFactura', 'facturas', 'articulos'));

}

public function update(Request $request, DetalleFactura $detalleFactura): RedirectResponse

{

$request->validate([

'cod\_factura' => 'required',

'cod\_articulo' => 'required',

'cantidad' => 'required|numeric|min:1',

]);

$articulo = Articulo::find($request->cod\_articulo);

$total = $request->cantidad \* $articulo->precio\_venta;

$detalleFactura->update([

'cod\_factura' => $request->cod\_factura,

'cod\_articulo' => $request->cod\_articulo,

'cantidad' => $request->cantidad,

'total' => $total

]);

return Redirect::route('detalle-facturas.index')

->with('success', 'Detalle de factura actualizado correctamente.');

}

public function destroy(DetalleFactura $detalleFactura): RedirectResponse

{

$detalleFactura->delete();

return Redirect::route('detalle-facturas.index')

->with('success', 'Detalle de factura eliminado correctamente.');

}

}



<?php

namespace App\Http\Controllers;

use App\Models\Devolucion;

use Illuminate\Http\RedirectResponse;

use Illuminate\Http\Request;

use App\Http\Requests\DevolucionRequest;

use Illuminate\Support\Facades\Redirect;

use Illuminate\View\View;

class DevolucionController extends Controller

{

/\*\*

\* Display a listing of the resource.

\*/

public function index(Request $request): View

{

$devolucions = Devolucion::paginate();

return view('devolucion.index', compact('devolucions'))

->with('i', ($request->input('page', 1) - 1) \* $devolucions->perPage());

}

/\*\*

\* Show the form for creating a new resource.

\*/

public function create(): View

{

$devolucion = new Devolucion();

return view('devolucion.create', compact('devolucion'));

}

/\*\*

\* Store a newly created resource in storage.

\*/

public function store(DevolucionRequest $request): RedirectResponse

{

Devolucion::create($request->validated());

return Redirect::route('devolucions.index')

->with('success', 'Devolucion created successfully.');

}

/\*\*

\* Display the specified resource.

\*/

public function show($id): View

{

$devolucion = Devolucion::find($id);

return view('devolucion.show', compact('devolucion'));

}

/\*\*

\* Show the form for editing the specified resource.

\*/

public function edit($id): View

{

$devolucion = Devolucion::find($id);

return view('devolucion.edit', compact('devolucion'));

}

/\*\*

\* Update the specified resource in storage.

\*/

public function update(DevolucionRequest $request, Devolucion $devolucion): RedirectResponse

{

$devolucion->update($request->validated());

return Redirect::route('devolucions.index')

->with('success', 'Devolucion updated successfully');

}

public function destroy($id): RedirectResponse

{

Devolucion::find($id)->delete();

return Redirect::route('devolucions.index')

->with('success', 'Devolucion deleted successfully');

}

}



<?php

namespace App\Http\Controllers;

use App\Models\Factura;

use App\Models\Cliente;

use App\Models\FormaDePago;

use Illuminate\Http\Request;

use Illuminate\Http\RedirectResponse;

use Illuminate\Support\Facades\Redirect;

use Illuminate\View\View;

class FacturaController extends Controller

{

// LISTADO

public function index(Request $request): View

{

$facturas = Factura::with(['cliente', 'formaPago'])->paginate(10);

return view('factura.index', compact('facturas'))

->with('i', ($request->input('page', 1) - 1) \* $facturas->perPage());

}

// FORM CREAR

public function create(): View

{

$factura = new Factura();

// Generar nro\_factura automáticamente (string incremental)

$lastFactura = Factura::orderBy('nro\_factura', 'desc')->first();

$factura->nro\_factura = $lastFactura

? (string)((int) preg\_replace('/\D/', '', $lastFactura->nro\_factura) + 1)

: '1';

// CLIENTES → documento => "Nombre Apellido - documento"

$clientes = Cliente::selectRaw("

documento,

CONCAT(nombres, ' ', apellidos, ' - ', documento) as nombre

")

->pluck('nombre', 'documento');

// FORMAS DE PAGO

$formasPago = FormaDePago::pluck('descripcion\_formapago', 'id\_formapago');

return view('factura.create', compact('factura', 'clientes', 'formasPago'));

}

// GUARDAR

public function store(Request $request): RedirectResponse

{

$request->validate([

'cod\_cliente' => 'required',

'nombre\_empleado' => 'required|string',

'fecha\_facturacion' => 'required|date',

'cod\_formapago' => 'required',

'total\_factura' => 'required|numeric',

]);

// Generar nro\_factura automáticamente (string incremental)

$lastFactura = Factura::orderBy('nro\_factura', 'desc')->first();

$nuevoNro = $lastFactura

? (string)((int) preg\_replace('/\D/', '', $lastFactura->nro\_factura) + 1)

: '1';

Factura::create([

'nro\_factura' => $nuevoNro,

'cod\_cliente' => $request->cod\_cliente,

'nombre\_empleado' => $request->nombre\_empleado,

'fecha\_facturacion' => $request->fecha\_facturacion,

'cod\_formapago' => $request->cod\_formapago,

'total\_factura' => $request->total\_factura,

]);

return Redirect::route('facturas.index')

->with('success', 'Factura registrada correctamente.');

}

// FORM EDITAR

public function edit(Factura $factura): View

{

$clientes = Cliente::selectRaw("

documento,

CONCAT(nombres, ' ', apellidos, ' - ', documento) as nombre

")

->pluck('nombre', 'documento');

$formasPago = FormaDePago::pluck('descripcion\_formapago', 'id\_formapago');

return view('factura.edit', compact('factura', 'clientes', 'formasPago'));

}

// ACTUALIZAR

public function update(Request $request, Factura $factura): RedirectResponse

{

$request->validate([

'cod\_cliente' => 'required',

'nombre\_empleado' => 'required|string',

'fecha\_facturacion' => 'required|date',

'cod\_formapago' => 'required',

'total\_factura' => 'required|numeric',

]);

$factura->update($request->only([

'cod\_cliente',

'nombre\_empleado',

'fecha\_facturacion',

'cod\_formapago',

'total\_factura',

]));

return Redirect::route('facturas.index')

->with('success', 'Factura actualizada correctamente.');

}

// ELIMINAR

public function destroy(Factura $factura): RedirectResponse

{

$factura->delete();

return Redirect::route('facturas.index')

->with('success', 'Factura eliminada correctamente.');

}

}



<?php

namespace App\Http\Controllers;

use App\Models\FormaDePago;

use Illuminate\Http\RedirectResponse;

use Illuminate\Http\Request;

use App\Http\Requests\FormaDePagoRequest;

use Illuminate\Support\Facades\Redirect;

use Illuminate\View\View;

class FormaDePagoController extends Controller

{

/\*\*

\* Display a listing of the resource.

\*/

public function index(Request $request): View

{

$formaDePagos = FormaDePago::paginate();

return view('forma-de-pago.index', compact('formaDePagos'))

->with('i', ($request->input('page', 1) - 1) \* $formaDePagos->perPage());

}

/\*\*

\* Show the form for creating a new resource.

\*/

public function create(): View

{

$formaDePago = new FormaDePago();

return view('forma-de-pago.create', compact('formaDePago'));

}

/\*\*

\* Store a newly created resource in storage.

\*/

public function store(FormaDePagoRequest $request): RedirectResponse

{

FormaDePago::create($request->validated());

return Redirect::route('forma-de-pagos.index')

->with('success', 'FormaDePago created successfully.');

}

/\*\*

\* Display the specified resource.

\*/

public function show($id): View

{

$formaDePago = FormaDePago::find($id);

return view('forma-de-pago.show', compact('formaDePago'));

}

/\*\*

\* Show the form for editing the specified resource.

\*/

public function edit($id): View

{

$formaDePago = FormaDePago::find($id);

return view('forma-de-pago.edit', compact('formaDePago'));

}

/\*\*

\* Update the specified resource in storage.

\*/

public function update(FormaDePagoRequest $request, FormaDePago $formaDePago): RedirectResponse

{

$formaDePago->update($request->validated());

return Redirect::route('forma-de-pagos.index')

->with('success', 'FormaDePago updated successfully');

}

public function destroy($id): RedirectResponse

{

FormaDePago::find($id)->delete();

return Redirect::route('forma-de-pagos.index')

->with('success', 'FormaDePago deleted successfully');

}

}



<?php

namespace App\Http\Controllers;

use Illuminate\Http\Request;

class HomeController extends Controller

{

/\*\*

\* Create a new controller instance.

\*

\* @return void

\*/

public function \_\_construct()

{

$this->middleware('auth');

}

/\*\*

\* Show the application dashboard.

\*

\* @return \Illuminate\Contracts\Support\Renderable

\*/

public function index()

{

return view('home');

}

}



<?php

namespace App\Http\Controllers;

use App\Models\Proveedor;

use App\Models\TipoDocumento;

use App\Models\Ciudad;

use Illuminate\Http\RedirectResponse;

use Illuminate\Http\Request;

use App\Http\Requests\ProveedorRequest;

use Illuminate\Support\Facades\Redirect;

use Illuminate\View\View;

class ProveedorController extends Controller

{

public function index(Request $request): View

{

$proveedors = Proveedor::paginate();

return view('proveedor.index', compact('proveedors'))

->with('i', ($request->input('page', 1) - 1) \* $proveedors->perPage());

}

// 🔹 CREATE (ARREGLADO)

public function create(): View

{

$proveedor = new Proveedor();

$tiposDocumentos = TipoDocumento::pluck('descripcion', 'id\_tipo\_documento');

$ciudades = Ciudad::pluck('nombre\_ciudad', 'codigo\_ciudad');

return view('proveedor.create', compact(

'proveedor',

'tiposDocumentos',

'ciudades'

));

}

public function store(ProveedorRequest $request): RedirectResponse

{

Proveedor::create($request->validated());

return Redirect::route('proveedors.index')

->with('success', 'Proveedor created successfully.');

}

public function show($id): View

{

$proveedor = Proveedor::findOrFail($id);

return view('proveedor.show', compact('proveedor'));

}

// 🔹 EDIT (YA ESTABA BIEN, SOLO ORDENADO)

public function edit(Proveedor $proveedor): View

{

$tiposDocumentos = TipoDocumento::pluck('descripcion', 'id\_tipo\_documento');

$ciudades = Ciudad::pluck('nombre\_ciudad', 'codigo\_ciudad');

return view('proveedor.edit', compact(

'proveedor',

'tiposDocumentos',

'ciudades'

));

}

public function update(ProveedorRequest $request, Proveedor $proveedor): RedirectResponse

{

$proveedor->update($request->validated());

return Redirect::route('proveedors.index')

->with('success', 'Proveedor updated successfully');

}

public function destroy($id): RedirectResponse

{

Proveedor::findOrFail($id)->delete();

return Redirect::route('proveedors.index')

->with('success', 'Proveedor deleted successfully');

}

}



<?php

namespace App\Http\Controllers;

use App\Models\TipoArticulo;

use Illuminate\Http\RedirectResponse;

use Illuminate\Http\Request;

use App\Http\Requests\TipoArticuloRequest;

use Illuminate\Support\Facades\Redirect;

use Illuminate\View\View;

class TipoArticuloController extends Controller

{

/\*\*

\* Display a listing of the resource.

\*/

public function index(Request $request): View

{

$tipoArticulos = TipoArticulo::paginate();

return view('tipo-articulo.index', compact('tipoArticulos'))

->with('i', ($request->input('page', 1) - 1) \* $tipoArticulos->perPage());

}

/\*\*

\* Show the form for creating a new resource.

\*/

public function create(): View

{

$tipoArticulo = new TipoArticulo();

return view('tipo-articulo.create', compact('tipoArticulo'));

}

/\*\*

\* Store a newly created resource in storage.

\*/

public function store(TipoArticuloRequest $request): RedirectResponse

{

TipoArticulo::create($request->validated());

return Redirect::route('tipo-articulos.index')

->with('success', 'TipoArticulo created successfully.');

}

/\*\*

\* Display the specified resource.

\*/

public function show($id): View

{

$tipoArticulo = TipoArticulo::find($id);

return view('tipo-articulo.show', compact('tipoArticulo'));

}

/\*\*

\* Show the form for editing the specified resource.

\*/

public function edit($id): View

{

$tipoArticulo = TipoArticulo::find($id);

return view('tipo-articulo.edit', compact('tipoArticulo'));

}

/\*\*

\* Update the specified resource in storage.

\*/

public function update(TipoArticuloRequest $request, TipoArticulo $tipoArticulo): RedirectResponse

{

$tipoArticulo->update($request->validated());

return Redirect::route('tipo-articulos.index')

->with('success', 'TipoArticulo updated successfully');

}

public function destroy($id): RedirectResponse

{

TipoArticulo::find($id)->delete();

return Redirect::route('tipo-articulos.index')

->with('success', 'TipoArticulo deleted successfully');

}

}



<?php

namespace App\Http\Controllers;

use App\Models\TipoDocumento;

use Illuminate\Http\RedirectResponse;

use Illuminate\Http\Request;

use App\Http\Requests\TipoDocumentoRequest;

use Illuminate\Support\Facades\Redirect;

use Illuminate\View\View;

class TipoDocumentoController extends Controller

{

/\*\*

\* Display a listing of the resource.

\*/

public function index(Request $request): View

{

$tipoDocumentos = TipoDocumento::paginate();

return view('tipo-documento.index', compact('tipoDocumentos'))

->with('i', ($request->input('page', 1) - 1) \* $tipoDocumentos->perPage());

}

/\*\*

\* Show the form for creating a new resource.

\*/

public function create(): View

{

$tipoDocumento = new TipoDocumento();

return view('tipo-documento.create', compact('tipoDocumento'));

}

/\*\*

\* Store a newly created resource in storage.

\*/

public function store(TipoDocumentoRequest $request): RedirectResponse

{

TipoDocumento::create($request->validated());

return Redirect::route('tipo-documentos.index')

->with('success', 'TipoDocumento created successfully.');

}

/\*\*

\* Display the specified resource.

\*/

public function show($id): View

{

$tipoDocumento = TipoDocumento::find($id);

return view('tipo-documento.show', compact('tipoDocumento'));

}

/\*\*

\* Show the form for editing the specified resource.

\*/

public function edit($id): View

{

$tipoDocumento = TipoDocumento::find($id);

return view('tipo-documento.edit', compact('tipoDocumento'));

}

/\*\*

\* Update the specified resource in storage.

\*/

public function update(TipoDocumentoRequest $request, TipoDocumento $tipoDocumento): RedirectResponse

{

$tipoDocumento->update($request->validated());

return Redirect::route('tipo-documentos.index')

->with('success', 'TipoDocumento updated successfully');

}

public function destroy($id): RedirectResponse

{

TipoDocumento::find($id)->delete();

return Redirect::route('tipo-documentos.index')

->with('success', 'TipoDocumento deleted successfully');

}

}



<?php

namespace App\Http\Controllers;

use App\Models\User;

use Illuminate\Http\RedirectResponse;

use Illuminate\Http\Request;

use App\Http\Requests\UserRequest;

use Illuminate\Support\Facades\Redirect;

use Illuminate\View\View;

class UserController extends Controller

{

/\*\*

\* Display a listing of the resource.

\*/

public function index(Request $request): View

{

$users = User::paginate();

return view('user.index', compact('users'))

->with('i', ($request->input('page', 1) - 1) \* $users->perPage());

}

/\*\*

\* Show the form for creating a new resource.

\*/

public function create(): View

{

$user = new User();

return view('user.create', compact('user'));

}

/\*\*

\* Store a newly created resource in storage.

\*/

public function store(UserRequest $request): RedirectResponse

{

User::create($request->validated());

return Redirect::route('users.index')

->with('success', 'User created successfully.');

}

/\*\*

\* Display the specified resource.

\*/

public function show($id): View

{

$user = User::find($id);

return view('user.show', compact('user'));

}

/\*\*

\* Show the form for editing the specified resource.

\*/

public function edit($id): View

{

$user = User::find($id);

return view('user.edit', compact('user'));

}

/\*\*

\* Update the specified resource in storage.

\*/

public function update(UserRequest $request, User $user): RedirectResponse

{

$user->update($request->validated());

return Redirect::route('users.index')

->with('success', 'User updated successfully');

}

public function destroy($id): RedirectResponse

{

User::find($id)->delete();

return Redirect::route('users.index')

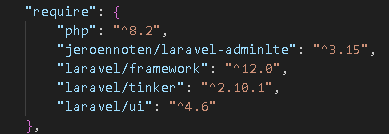
->with('success', 'User deleted successfully');

}

}

**Paquetes o componentes utilizados para la aplicación: (Vendor)**

**Paquetes de producción (require)**

****

**Paquetes de desarrollo (require-dev)**



**DESPLIEGUE DE LA APLICACIÓN**

El despliegue de una aplicación web es una etapa fundamental en el desarrollo de software, ya que permite que un sistema pase del entorno local del desarrollador a un entorno accesible para los usuarios finales a través de internet. Este proceso garantiza que la aplicación funcione de manera estable, segura y eficiente en un entorno real.

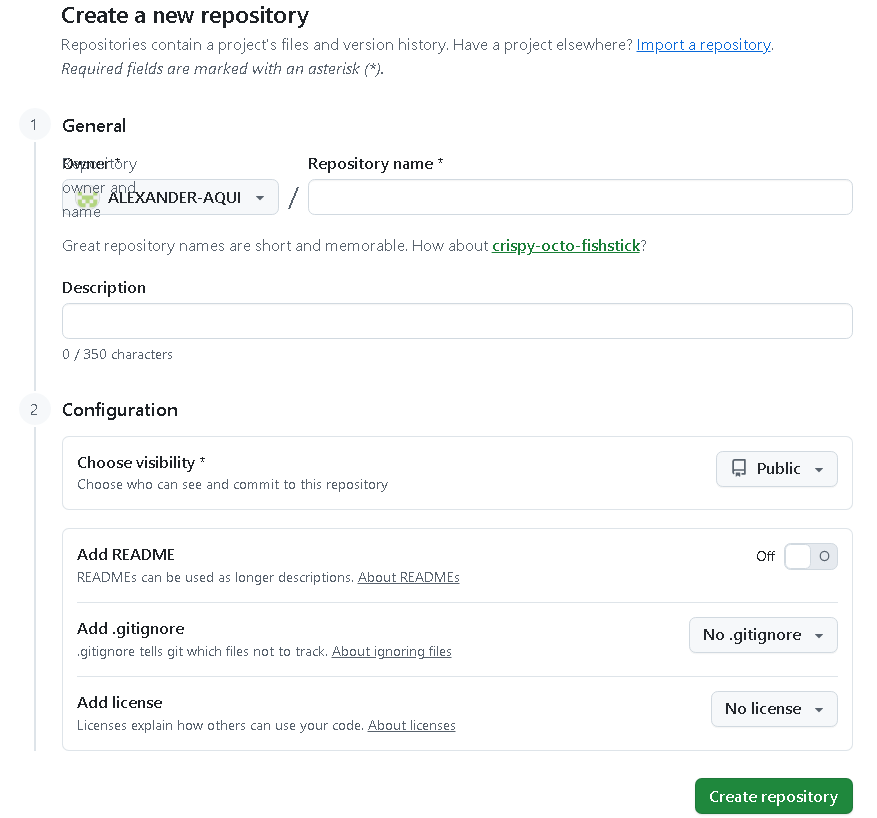
**Desarrollo en entorno local**

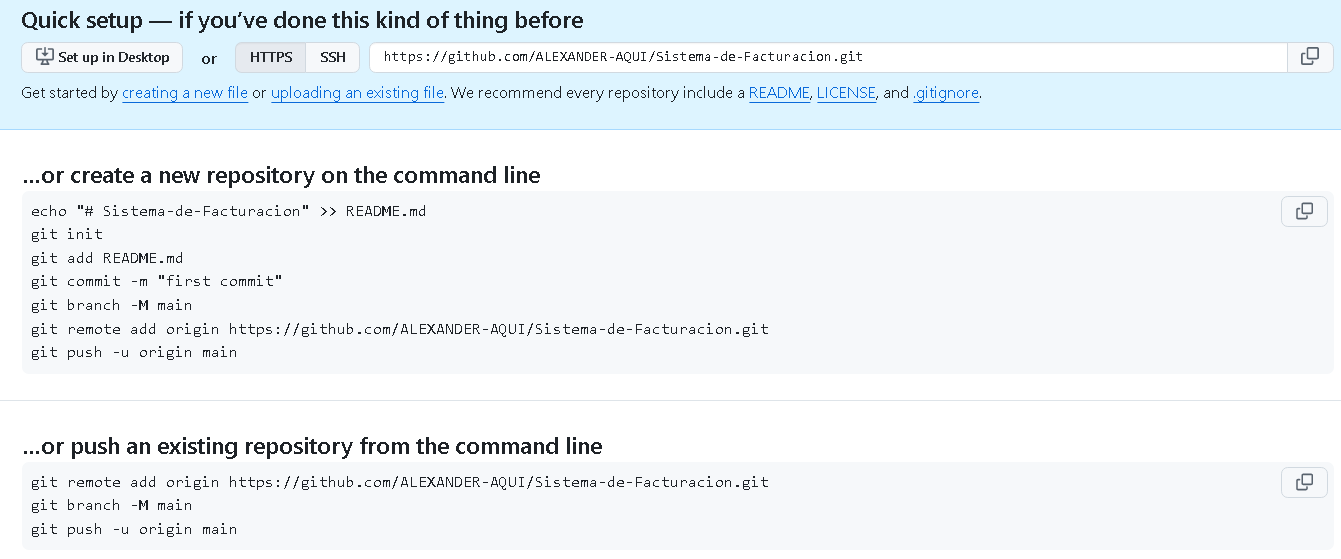
El proyecto fue desarrollado en un entorno local utilizando el framework **Laravel**, con una base de datos **MySQL**, permitiendo la implementación y prueba de las funcionalidades antes de su publicación.

El desarrollo en entorno local permite al programador detectar errores, realizar pruebas y mejorar funcionalidades sin afectar a los usuarios finales. Esta etapa es clave porque reduce riesgos, mejora la calidad del software y permite un aprendizaje progresivo del funcionamiento del sistema.

**Subir Proyecto a GitHub**

GitHub es una plataforma de alojamiento de código que permite almacenar proyectos de software de forma segura, facilitando el control de versiones y el trabajo colaborativo. Su uso es esencial en proyectos modernos, ya que permite registrar cambios, volver a versiones anteriores y compartir el proyecto con otros desarrolladores.



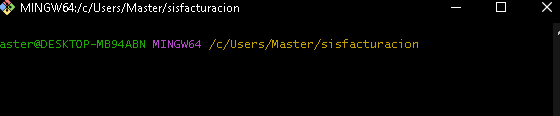


**Inicialización del control de versiones**

Se utilizó **Git** como sistema de control de versiones para registrar los cambios del proyecto y facilitar su gestión.

El uso de control de versiones fomenta la organización, responsabilidad y disciplina en el desarrollo de software, habilidades importantes tanto para estudiantes como para profesionales, ya que permite un mejor seguimiento del progreso del proyecto.







A continuación, se presentan los principales comandos utilizados para gestionar el repositorio y publicar el proyecto en GitHub.

**Comandos de Github**

* **Inicializar repositorio Git**

Se inicializó el repositorio local para el control de versiones del proyecto.

* git init
* **Agregar archivos al control de versiones**

Se añadieron los archivos del proyecto al área de preparación.

* git add .
* **Registrar cambios**

Se creó un commit con los cambios iniciales del proyecto.

* git commit -m "Primer commit"
* **Vincular repositorio local con GitHub**
* git remote add origin https://github.com/ALEXANDER-AQUI/Sistema-de-Facturacion.git
* **Publicar el proyecto en GitHub**

Se enviaron los archivos al repositorio remoto.

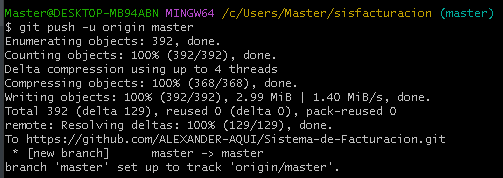
* git push -u origin main
* main - cambiar nombre según consola en este caso fue master



**Publicación del proyecto en GitHub**

El proyecto fue publicado en un repositorio remoto en **GitHub**, permitiendo el almacenamiento del código fuente y el trabajo colaborativo.

Además, GitHub permite que el proyecto pueda ser revisado, evaluado y mejorado, promoviendo el aprendizaje colaborativo y la transparencia en el desarrollo del sistema.



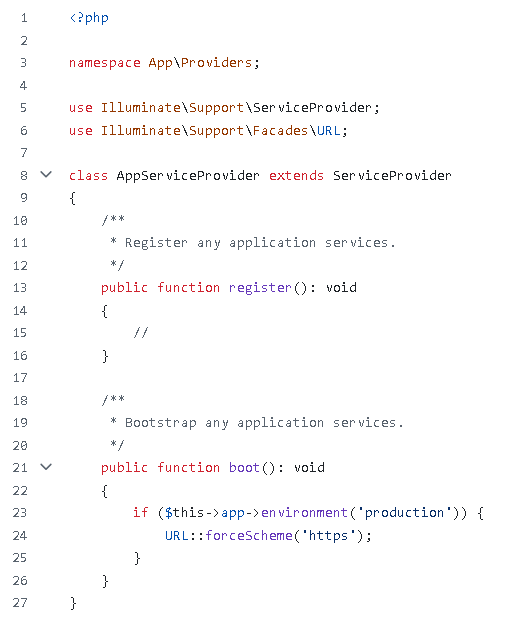
**Preparación del proyecto para producción**

Se realizaron ajustes en la configuración del entorno para su ejecución en producción, incluyendo la configuración de variables de entorno y la optimización de la aplicación.

La preparación para producción es una fase crítica, ya que adapta la aplicación para un entorno real donde será utilizada por usuarios. En esta etapa se prioriza el rendimiento, la seguridad y la correcta configuración del sistema, asegurando una experiencia estable y confiable.

**EN GITHUB**

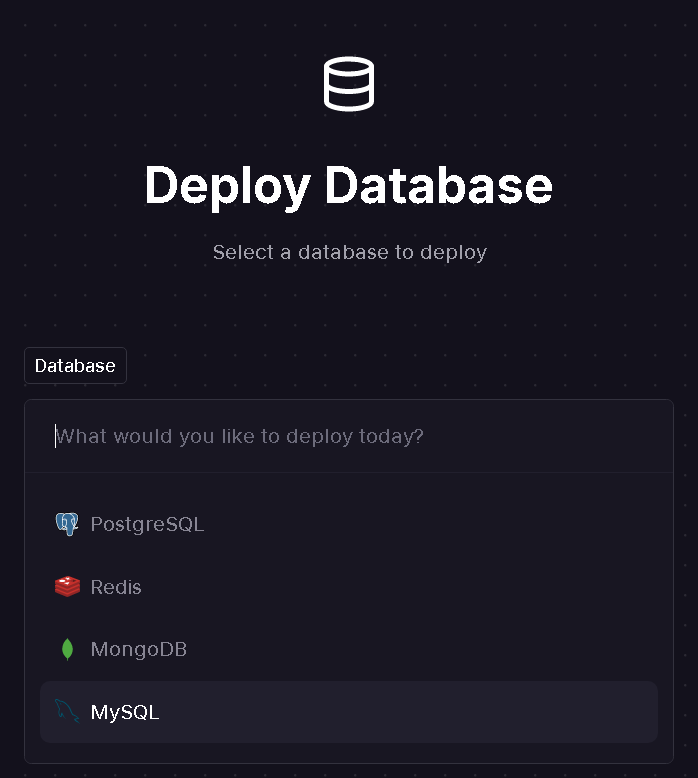




**RAILWAY**

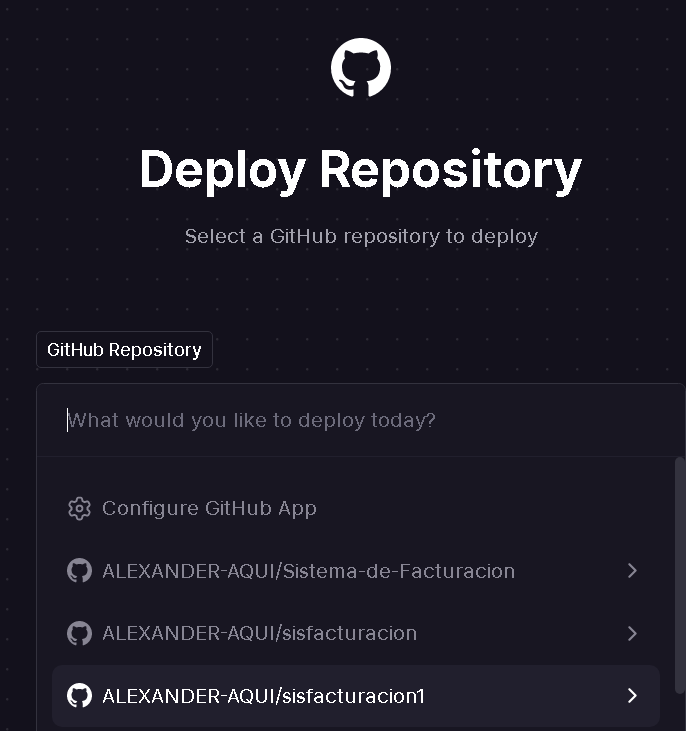
Railway es una plataforma de despliegue en la nube que permite publicar aplicaciones web de forma sencilla y automatizada. Su uso facilita el aprendizaje del despliegue moderno, eliminando la complejidad de configuraciones avanzadas y permitiendo al estudiante centrarse en el funcionamiento de la aplicación.

**Creamos base de datos**

****

**Despliegue del proyecto en Railway**

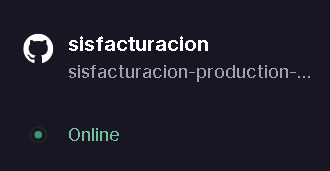
El repositorio de GitHub fue vinculado con Railway, permitiendo el despliegue automático de la aplicación Laravel en un entorno web productivo.

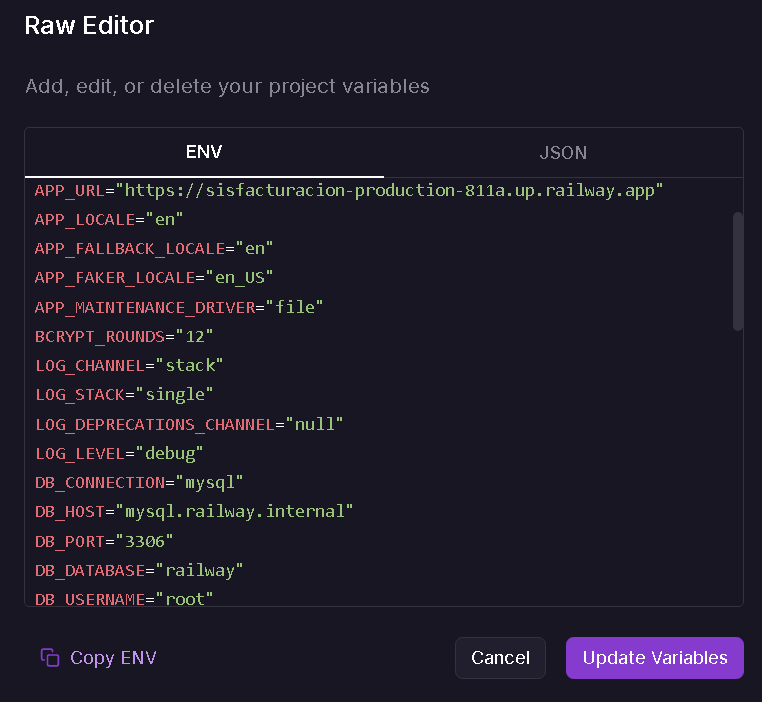


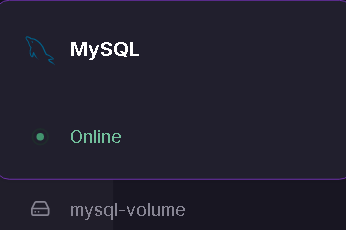
**Configuración de variables de entorno en Railway**

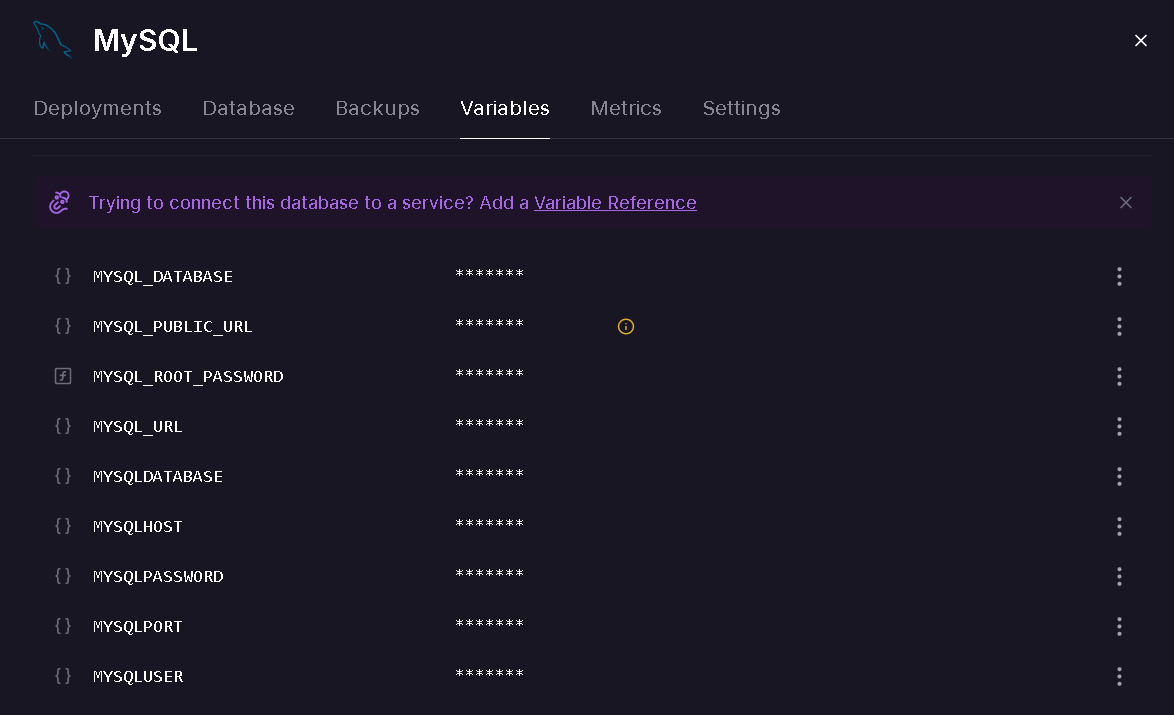
Las variables de entorno permiten separar la configuración del código fuente, lo que mejora la seguridad y flexibilidad del sistema. Gracias a esto, la aplicación puede adaptarse a distintos entornos sin modificar su estructura interna.

* Se configuraron las variables de entorno necesarias en la plataforma Railway, adaptando los valores del entorno local al entorno de producción.





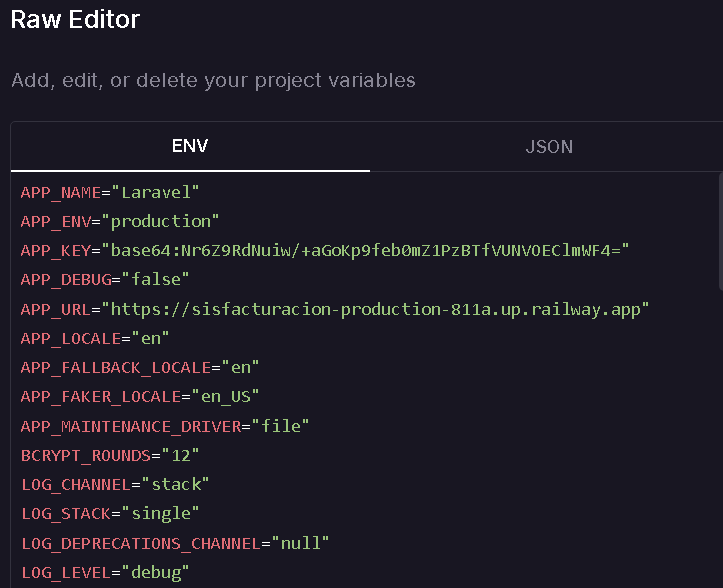


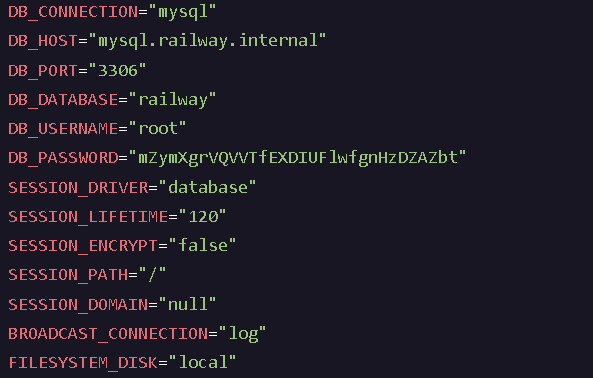


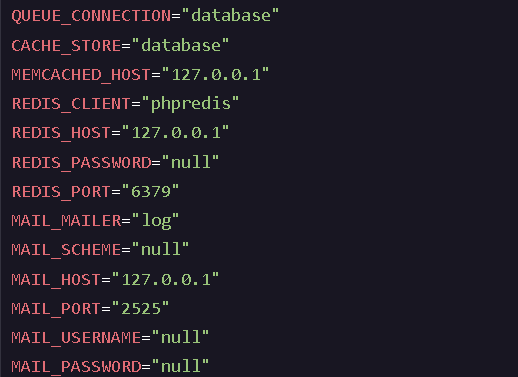
En Railway → Variables

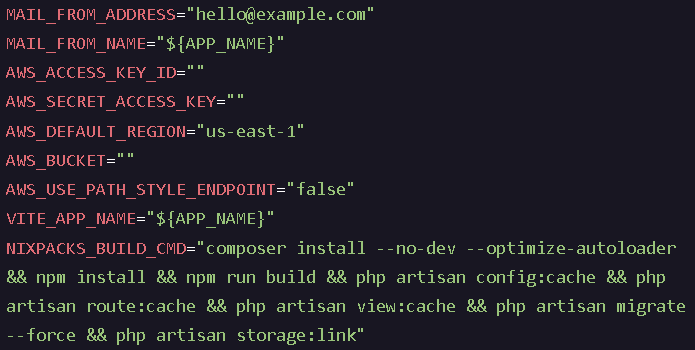
Copiamos nuestro archivo:

* .env del proyecto y reemplazamos las variables



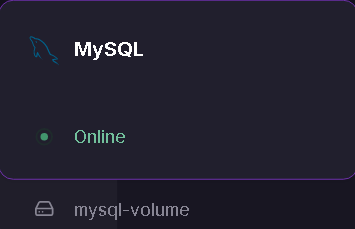


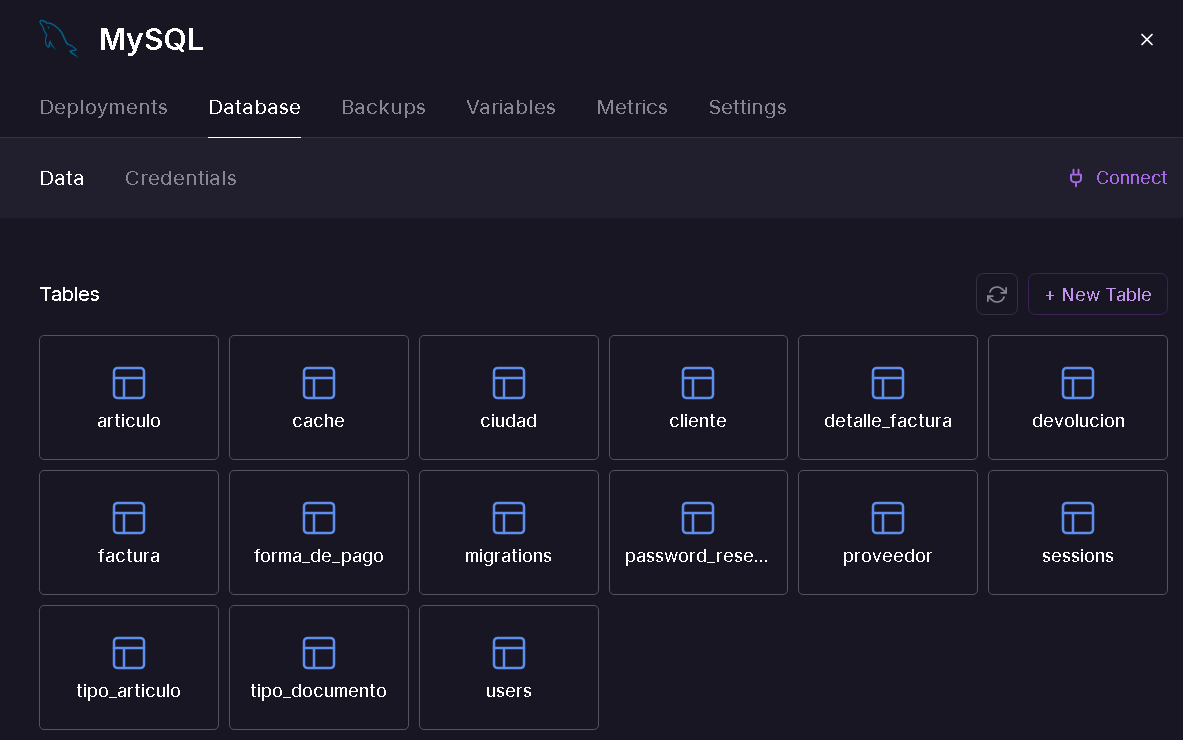




**Ejecución de migraciones y caché**

La ejecución de migraciones asegura que la base de datos esté correctamente estructurada, mientras que el uso de caché mejora el rendimiento del sistema, brindando una experiencia más fluida para el usuario final.

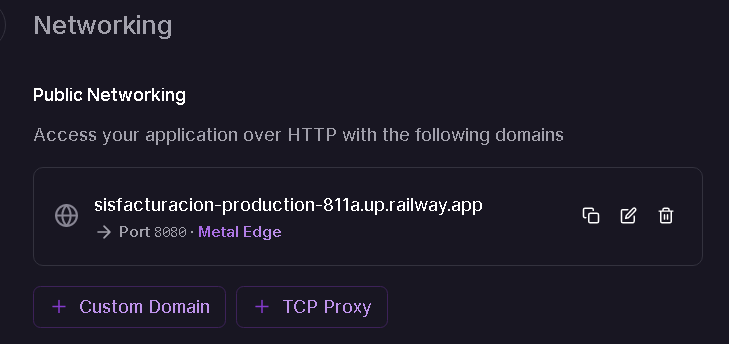




**Verificación del funcionamiento**

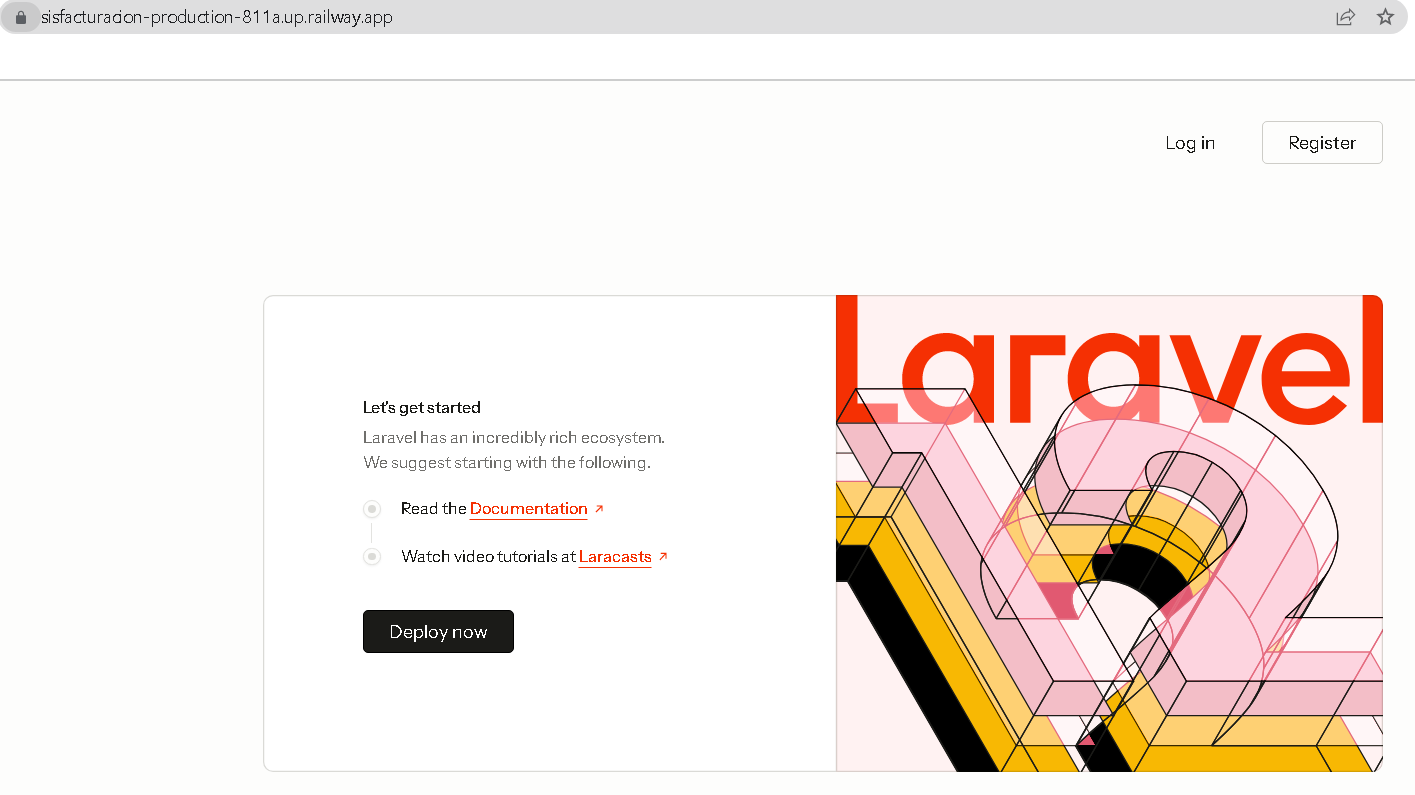
Se verificó el correcto funcionamiento del sistema, confirmando la carga de estilos, conexión a la base de datos y ejecución adecuada de la aplicación.

Esta verificación garantiza que el sistema esté listo para ser utilizado por personas de diferentes edades, ofreciendo una interfaz funcional, estable y accesible.

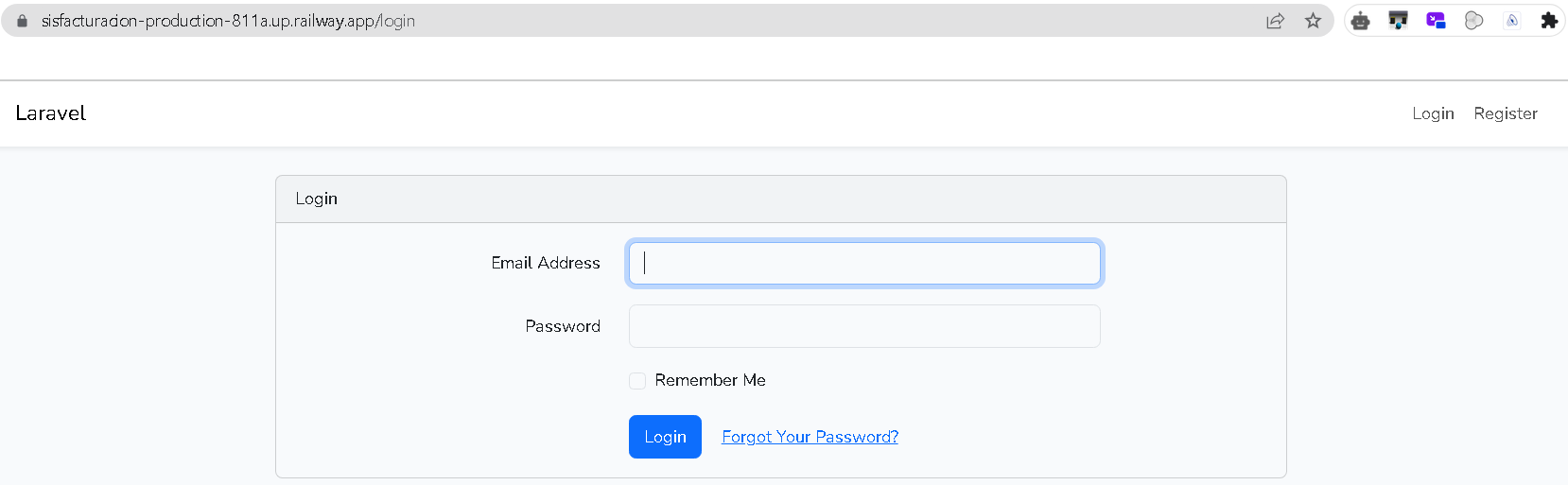


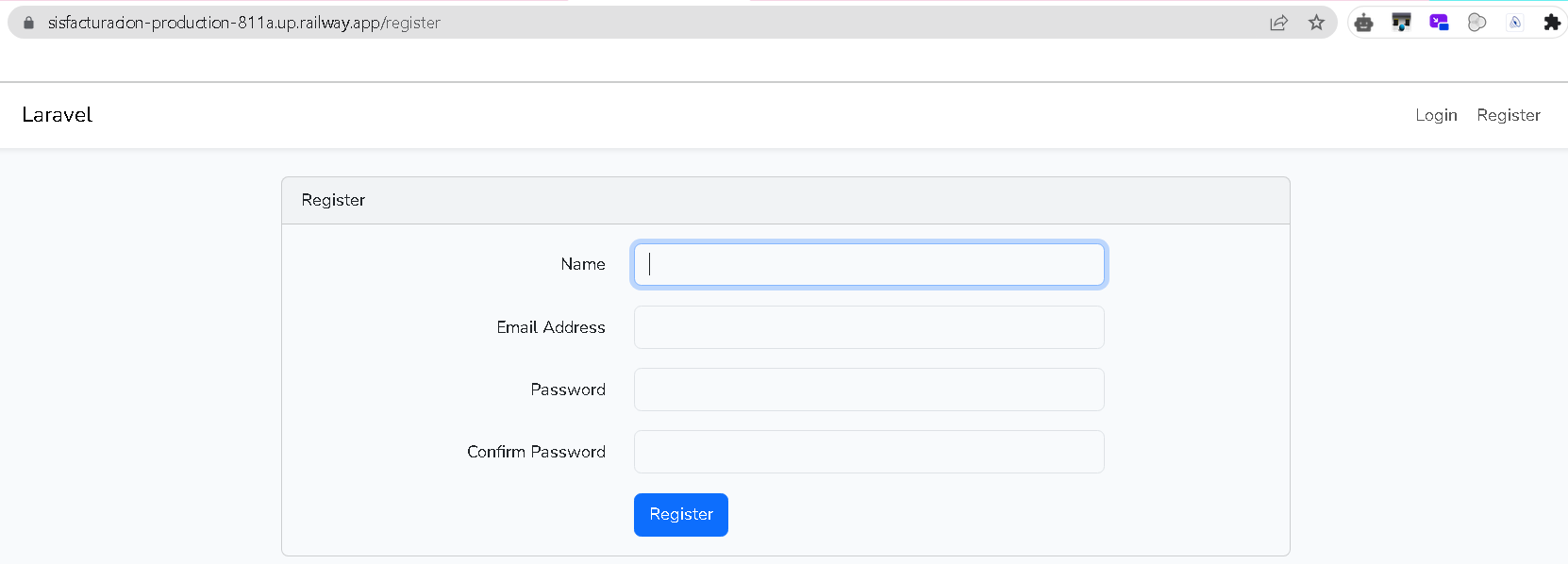


* **Inicio**

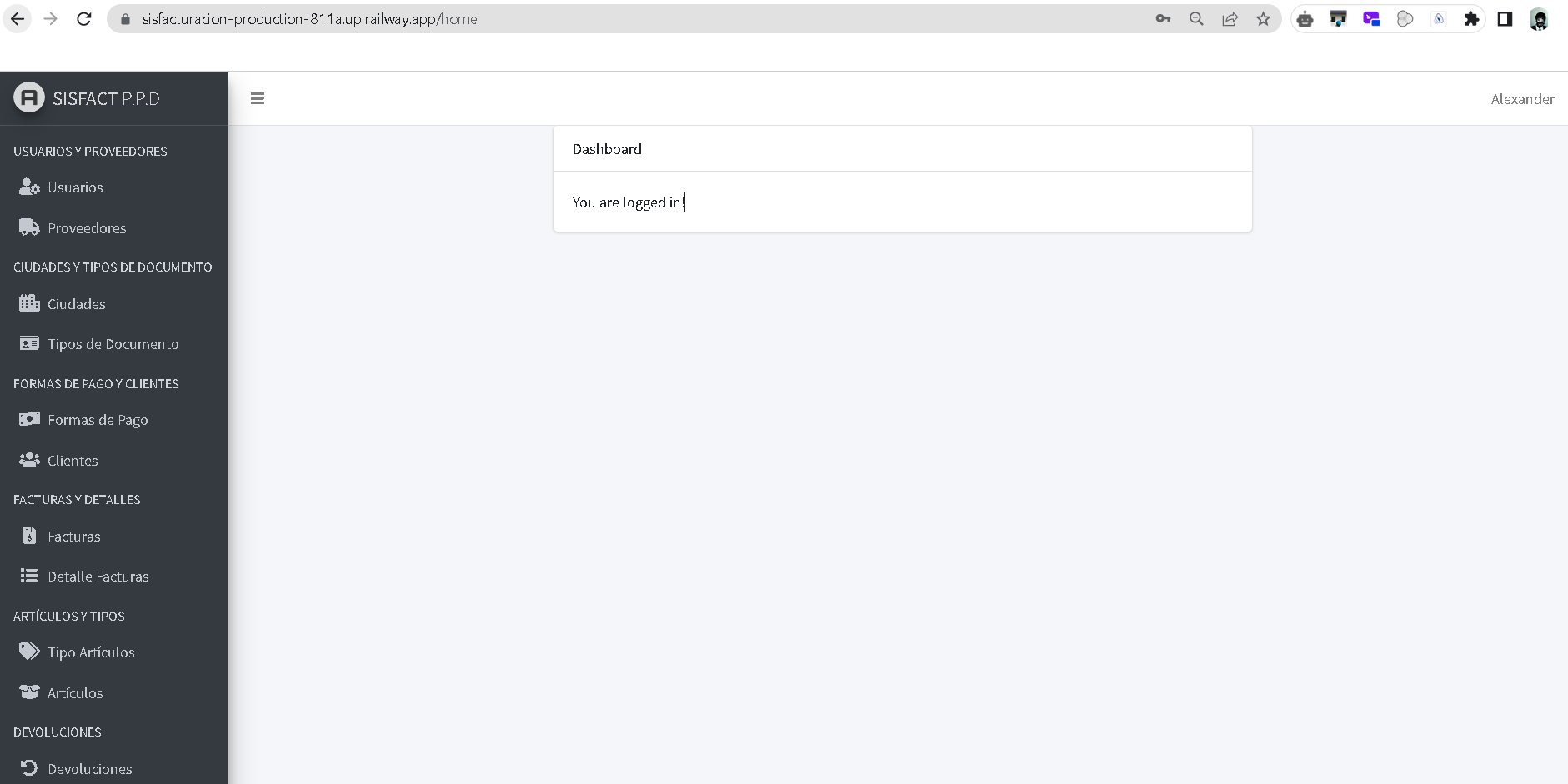


* **Login - Register**





* **Home**



**Conclusión:**

El proceso de despliegue de la aplicación permitió consolidar los conocimientos adquiridos durante el desarrollo del proyecto, desde la programación en entorno local hasta su publicación en la nube. A través del uso de herramientas como GitHub y Railway, se logró comprender la importancia del control de versiones, la seguridad de la información y la correcta configuración de entornos productivos.

Este trabajo no solo fortalece habilidades técnicas, sino que también fomenta la responsabilidad, la organización y la confianza en nosotros como estudiantes, al ver la aplicación funcionando correctamente en un entorno real. Finalmente, el despliegue exitoso del sistema demuestra que el proyecto se encuentra preparado para ser utilizado por usuarios reales, cumpliendo con los objetivos propuestos.