

**Proyecto Final**

**Sistema CRUD de Compras con AdminLTE**

**Estudiante:** Jorge Rodrigo Urizar Rivera  
**Docente:** Ing. José Ovando Moscoso  
**Materia:** Taller de Sistemas III  
**Fecha:** 24 de junio de 2025

**1. Introducción**

**1.1. Objetivo General**

Desarrollar un sistema web robusto para la gestión comercial que implemente operaciones CRUD (Crear, Leer, Actualizar, Eliminar) sobre cinco tablas relacionales, utilizando Laravel como backend y AdminLTE como interfaz administrativa.

**1.2. Justificación**

El sistema resuelve la necesidad de las pequeñas y medianas empresas de gestionar eficientemente su inventario, clientes y transacciones comerciales. La implementación de relaciones complejas (como muchos-a-muchos con atributos adicionales) y el control de stock son esenciales para evitar errores operativos y proveer información confiable para la toma de decisiones.

**2. Análisis de Tablas**

El sistema gestiona las siguientes entidades:

| **Tabla** | **Descripción** | **Campos Relevantes** |
| --- | --- | --- |
| tbl\_categorias | Clasificación de productos | id, nombre, descripción |
| tbl\_productos | Artículos comercializados | id, nombre, precio, stock, categoria\_id |
| tbl\_clientes | Información de compradores | id, nombre, email, teléfono |
| tbl\_compras | Registro de transacciones | id, fecha, total, cliente\_id |
| tbl\_compra\_producto | Relación compra-producto (muchos-a-muchos) | id, compra\_id, producto\_id, cantidad, total |

**Relaciones:**

* Categoría → Productos (1:N)
* Cliente → Compras (1:N)
* Compra → Productos (N:M a través de tbl\_compra\_producto)

**3. Diseño de Base de Datos**

Modelo Relacional

text

Copy

Download

tbl\_categorias tbl\_productos tbl\_clientes

+----+------------+ +----+---------+----+ +----+---------+

| id | nombre | | id | nombre |...| | id | nombre |

+----+------------+ +----+---------+----+ +----+---------+

| | |

| 1:N | N:M | 1:N

v v v

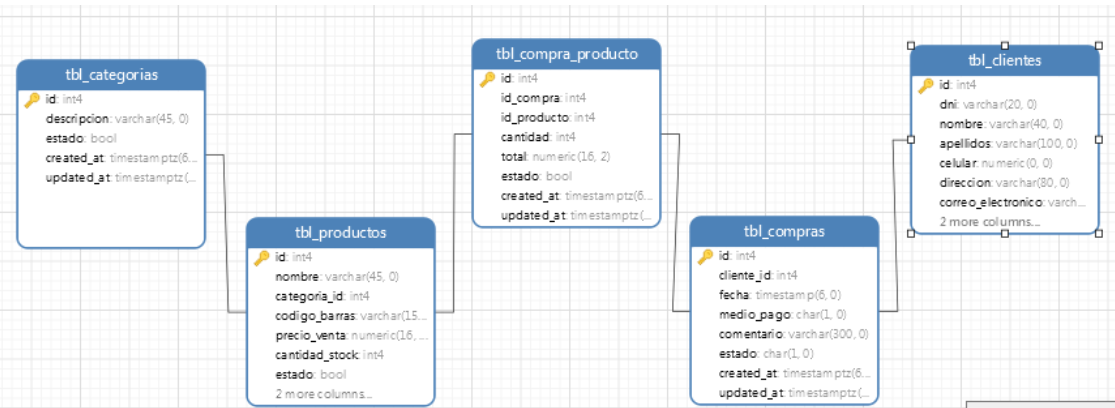
tbl\_compra\_producto tbl\_compras

+----+----------+-------+ +----+------------+

| id | cantidad | total | | id | fecha |

+----+----------+-------+ +----+------------+

Diagrama E-R



**4. Implementación Técnica**

**4.1. Entorno de Desarrollo**

* **Backend:** Laravel 10.x
* **Frontend:** AdminLTE 3.2
* **Base de Datos:** MySQL 8.0
* **Gestión de Paquetes:** Composer

**4.2. Estructura de Rutas**

php

Copy

Download

*// routes/web.php*

Route::resource('categorias', CategoriaController::class);

Route::resource('productos', ProductoController::class);

Route::resource('clientes', ClienteController::class);

Route::resource('compras', CompraController::class);

Route::post('compras/agregar-producto', [CompraController::class, 'agregarProducto']);

**4.3. CRUD para tbl\_compra\_producto (Desafío Técnico)**

**Implementación en**CompraController.php**:**

php

Copy

Download

public function agregarProducto(Request $request, $compra\_id) {

$validated = $request->validate([

'producto\_id' => 'required|exists:productos,id',

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

]);

$producto = Producto::find($request->producto\_id);

*// Validar stock*

if ($producto->stock < $request->cantidad) {

return back()->withErrors(['stock' => 'Stock insuficiente']);

}

*// Calcular total*

$total = $producto->precio \* $request->cantidad;

*// Registrar en tabla pivote*

CompraProducto::create([

'compra\_id' => $compra\_id,

'producto\_id' => $request->producto\_id,

'cantidad' => $request->cantidad,

'total' => $total

]);

*// Actualizar stock*

$producto->decrement('stock', $request->cantidad);

*// Actualizar total de compra*

$compra = Compra::find($compra\_id);

$compra->increment('total', $total);

return back()->with('success', 'Producto agregado');

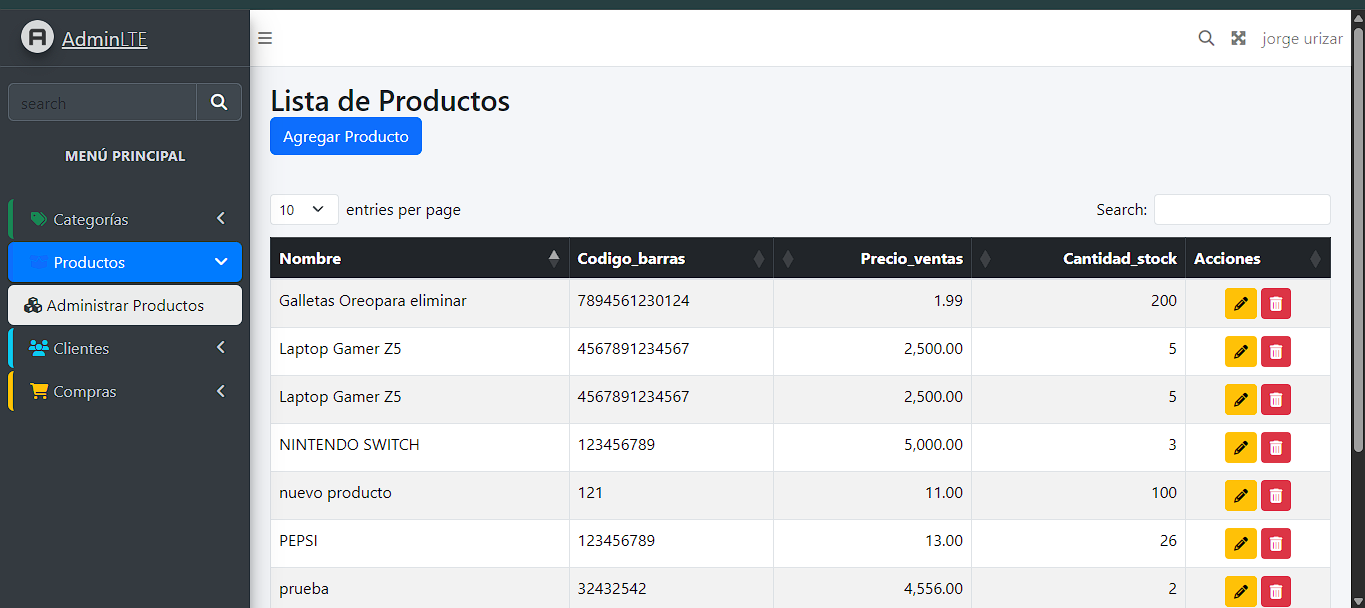
}

**Características clave:**

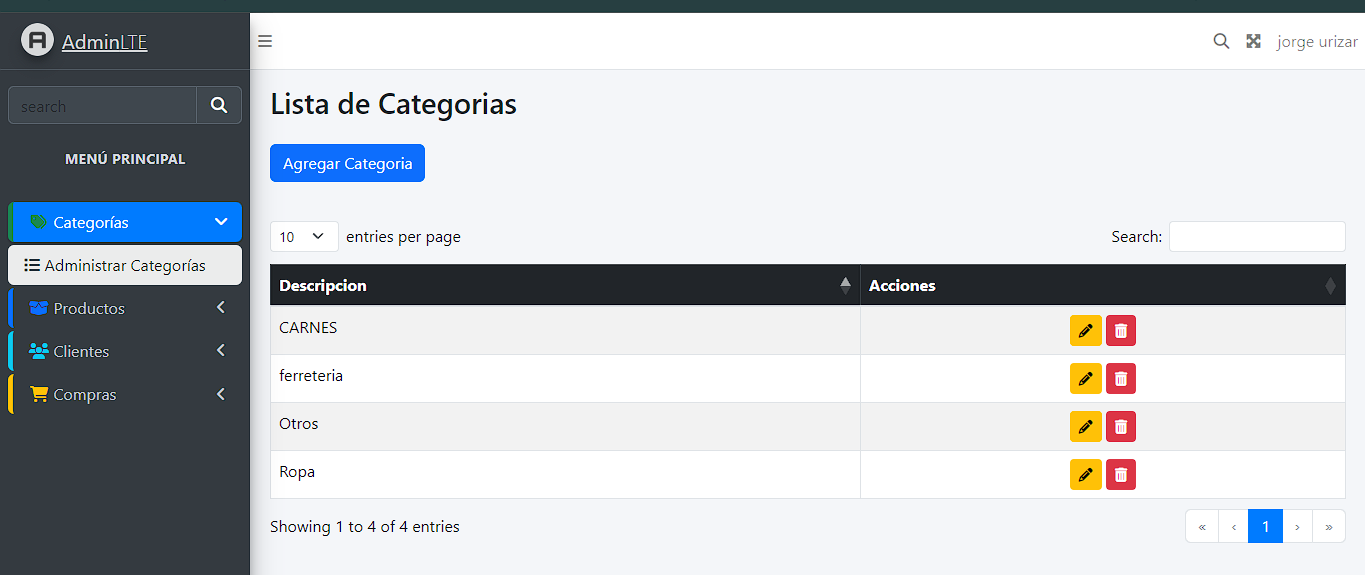
1. Validación de stock en tiempo real
2. Cálculo automático de totales
3. Actualización sincronizada de inventario
4. Manejo de errores con mensajes contextuales

**5. Capturas de Pantalla**

**5.1. Gestión de Productos**



**5.2. Gestión de Categorías**



**6. Problemas y Soluciones**

| **Problema** | **Solución Implementada** |
| --- | --- |
| Sincronización de stock | Transacciones de base de datos |
| Cálculo de totales en tiempo real | Eventos de Livewire |
| Validación compleja N:M | Form Request personalizados |
|  |  |

**Lección aprendida:**

"La gestión de relaciones muchos-a-muchos con atributos adicionales requiere un diseño cuidadoso de la capa de negocio para mantener la integridad de los datos, especialmente cuando involucra operaciones concurrentes."

**7. Conclusiones**

El proyecto demostró la efectividad de Laravel para implementar sistemas complejos con relaciones avanzadas entre entidades. La integración con AdminLTE permitió desarrollar una interfaz profesional en tiempo reducido. Los principales logros incluyen:

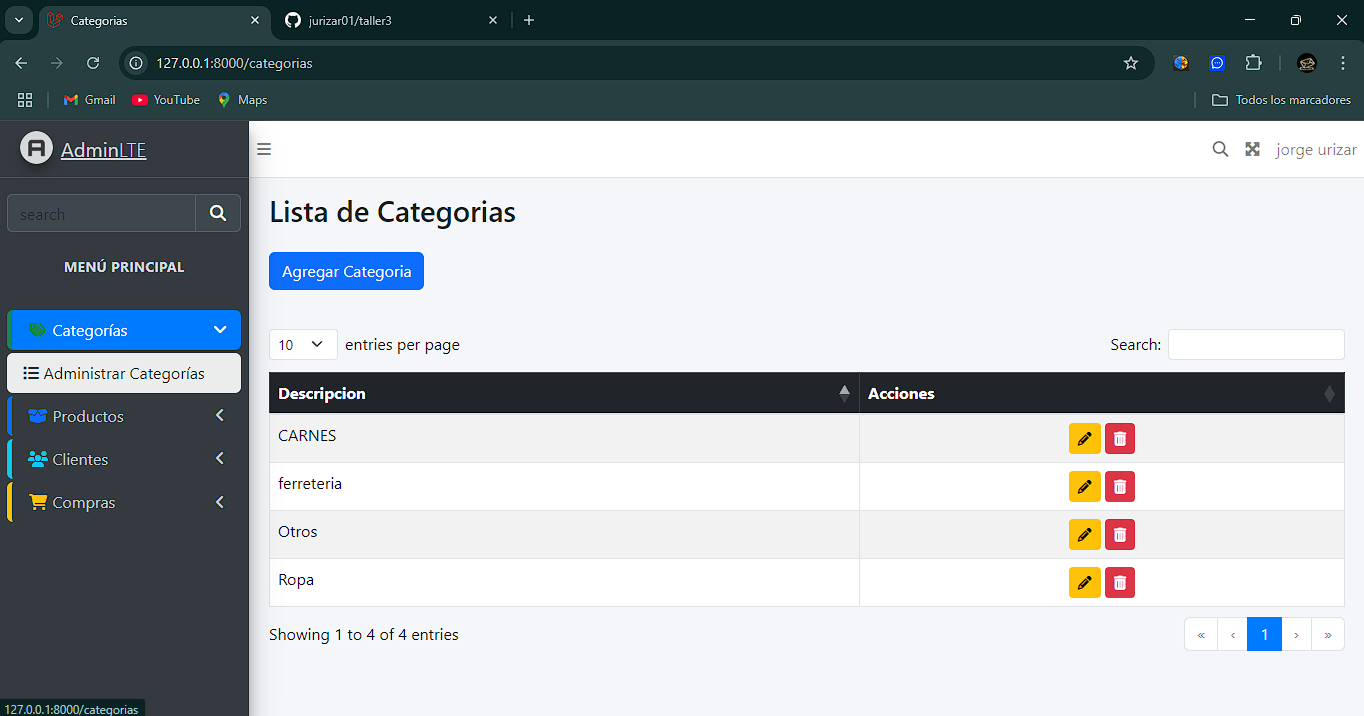
1. Implementación exitosa de operaciones CRUD para 5 entidades
2. Solución eficiente al desafío de la relación N:M con atributos
3. Mecanismos de validación que garantizan integridad de datos
4. Sistema escalable con arquitectura MVC

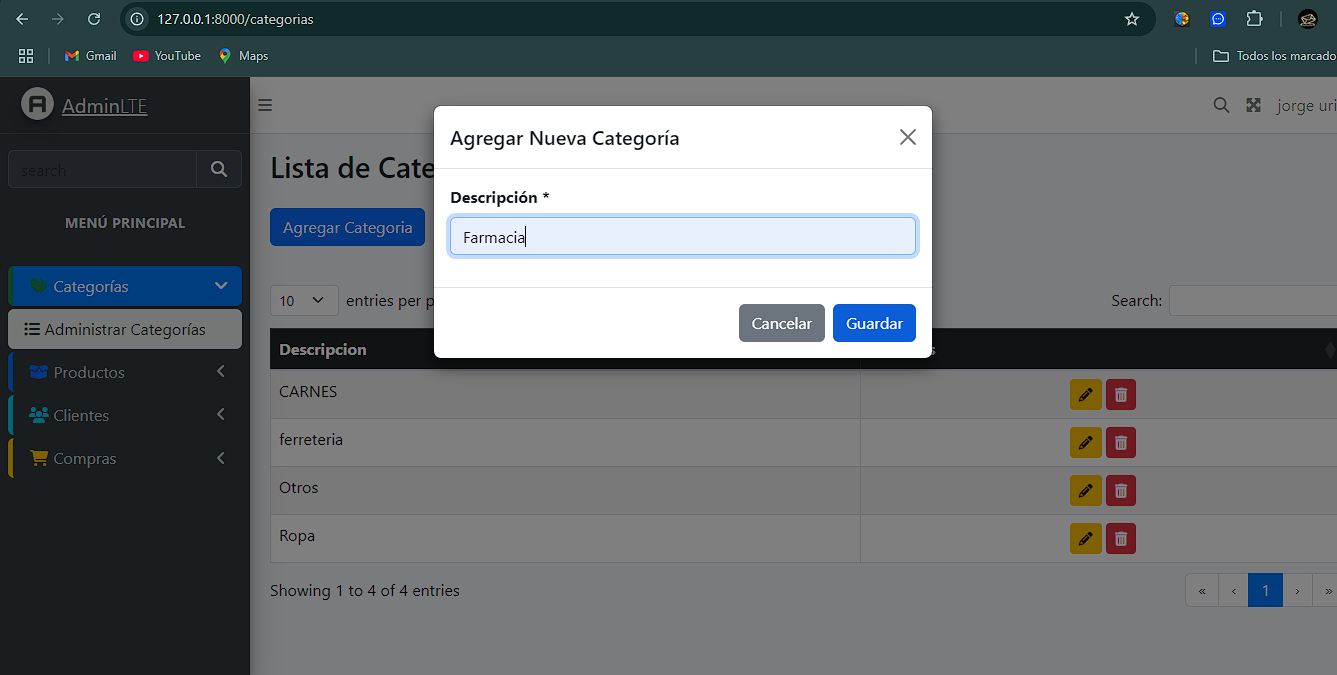
**8. Recomendaciones**

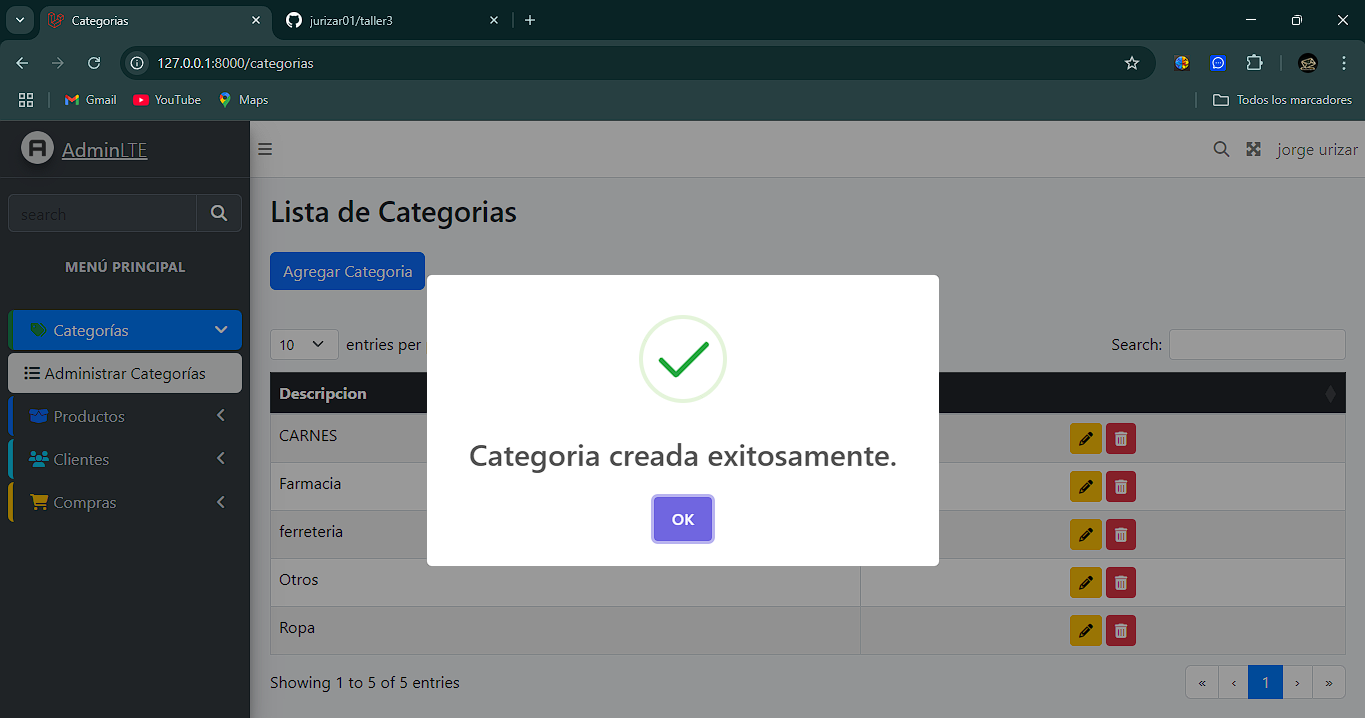
1. Implementar módulo de proveedores
2. Agregar gráficos analíticos con Chart.js
3. Desarrollar API REST para integración móvil
4. Implementar sistema de roles y permisos
5. Agregar exportación a Excel de reportes

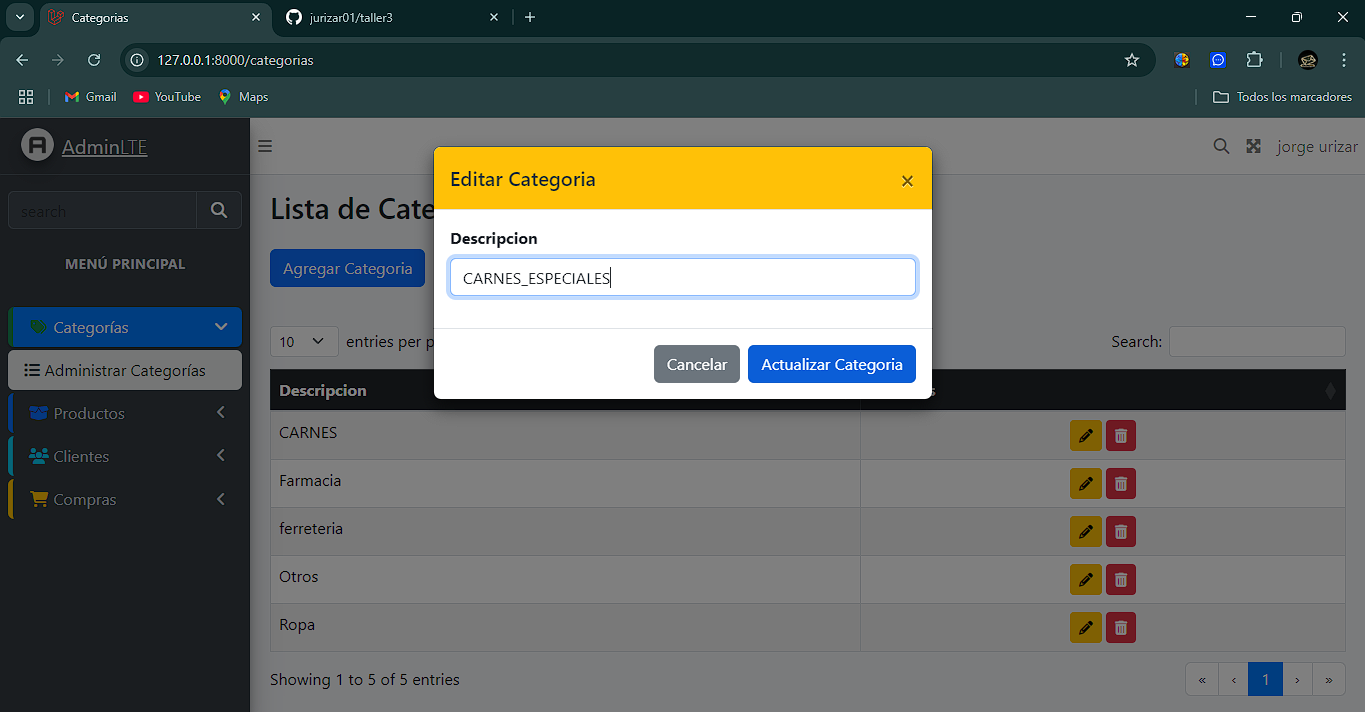
**9. Anexos**

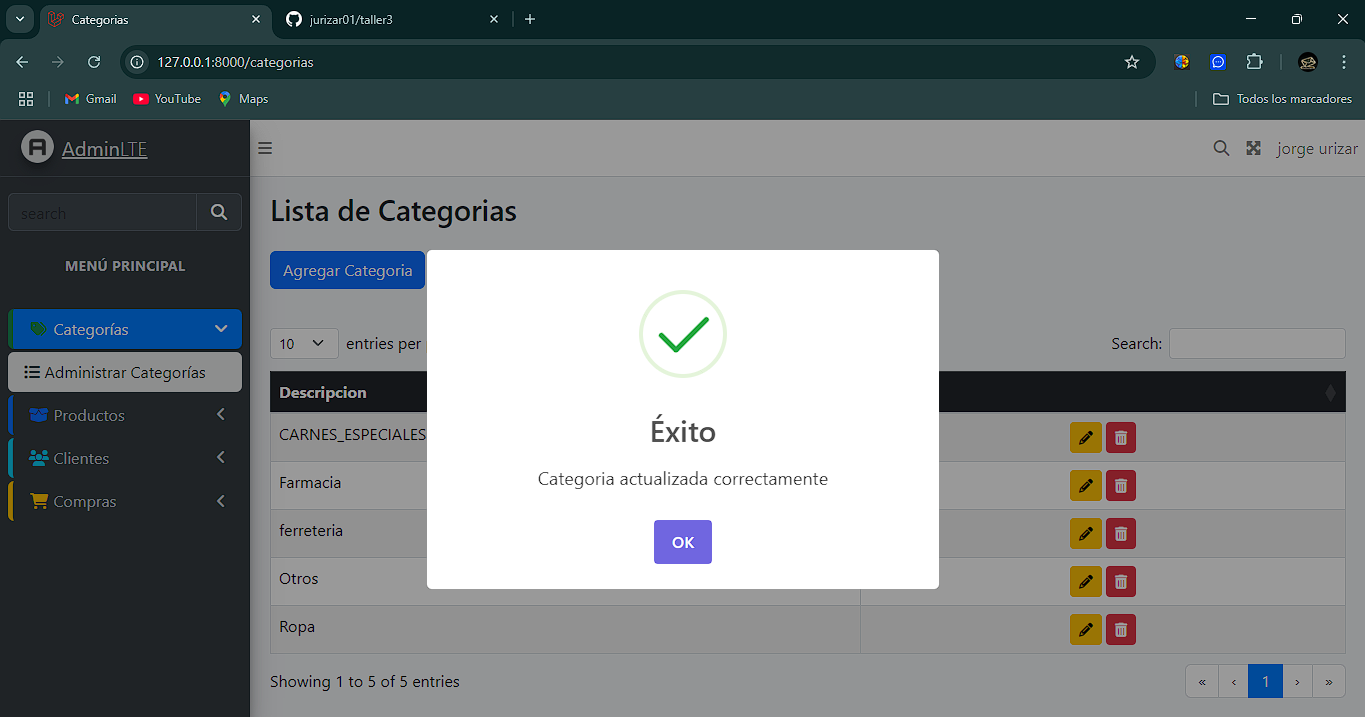
****

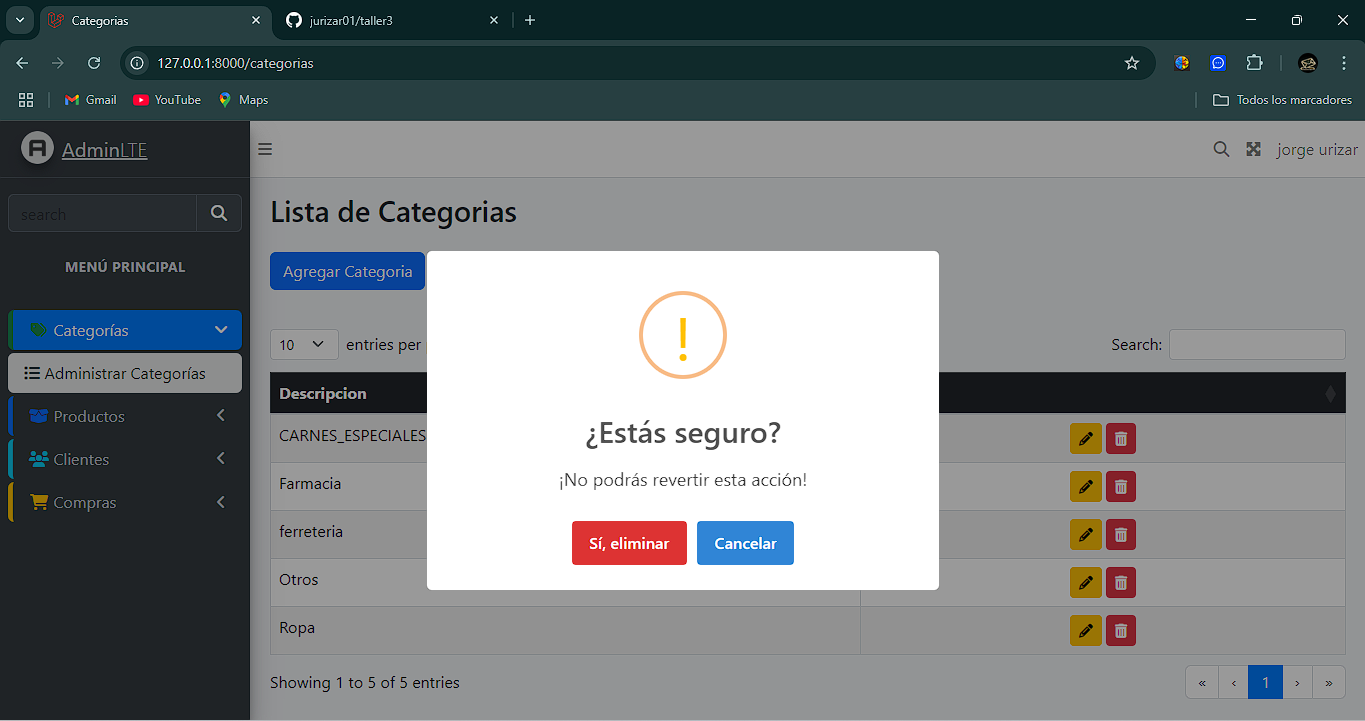
****

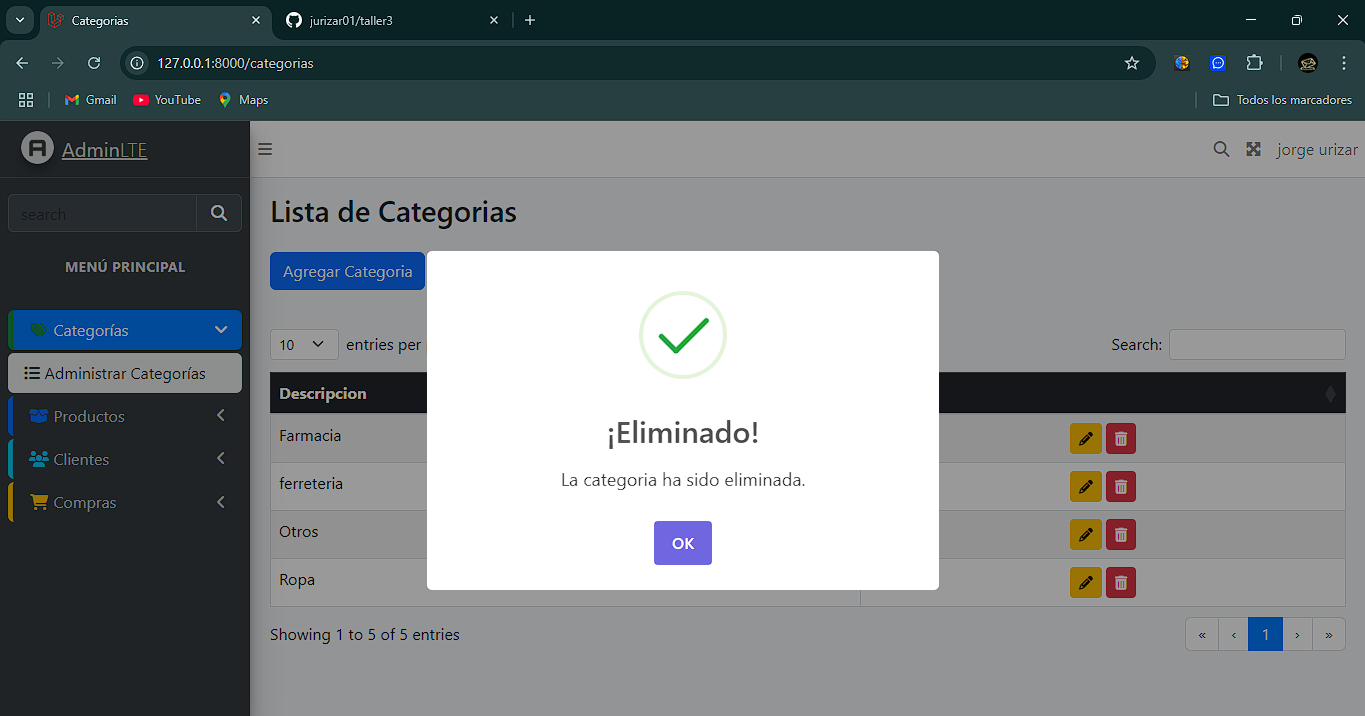
****

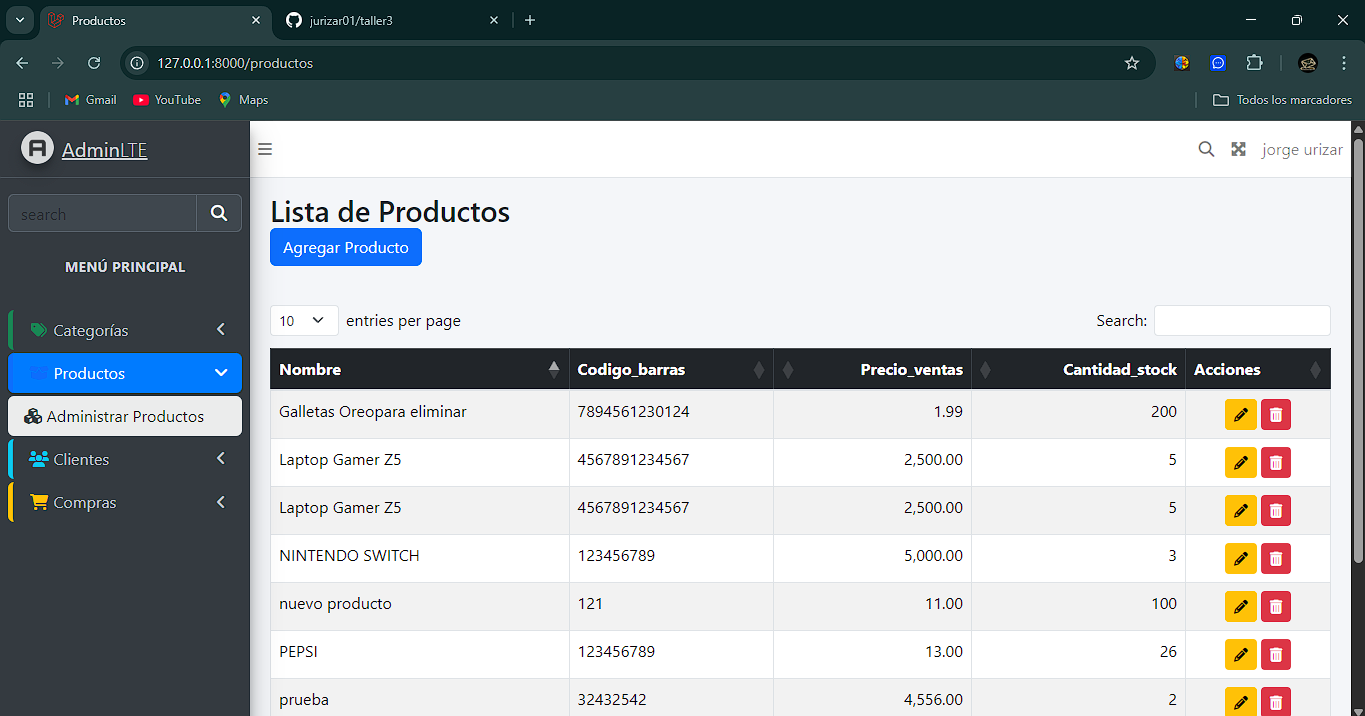
****

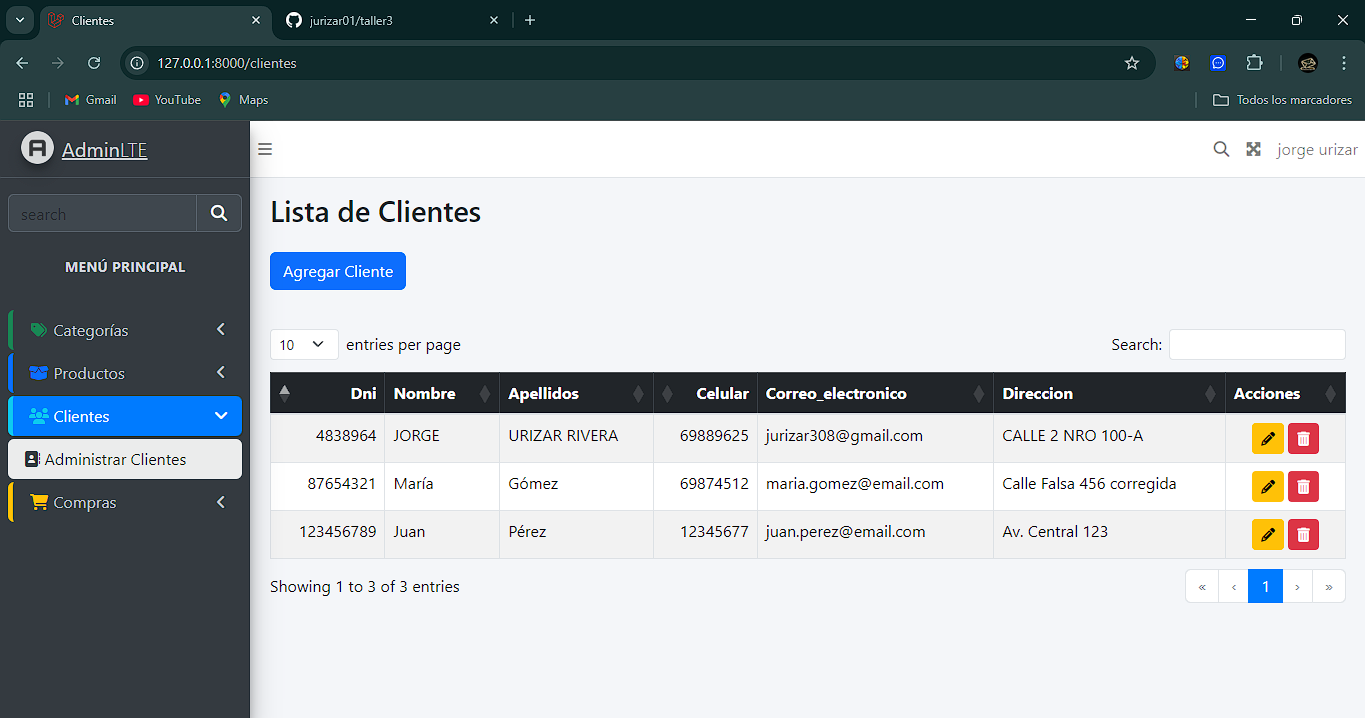
****

****

****

****

****

****

Estructura de Controladores

php

Copy

Download

*// app/Http/Controllers/CompraController.php*

class CompraController extends Controller {

public function create() {

return view('compras.create', [

'clientes' => Cliente::all(),

'productos' => Producto::where('stock', '>', 0)->get()

]);

}

public function store(CompraRequest $request) {

*// Lógica para crear nueva compra*

}

}

Modelo con Relaciones

php

Copy

Download

*// app/Models/Compra.php*

class Compra extends Model {

public function cliente() {

return $this->belongsTo(Cliente::class);

}

public function productos() {

return $this->belongsToMany(Producto::class)

->withPivot('cantidad', 'total')

->withTimestamps();

}

}

Configuración AdminLTE

php

Copy

Download

*// config/adminlte.php*

'menu' => [

[

'text' => 'Dashboard',

'url' => '/home',

'icon' => 'fas fa-tachometer-alt'

],

[

'text' => 'Gestión Comercial',

'icon' => 'fas fa-shopping-cart',

'submenu' => [

['text' => 'Productos', 'url' => '/productos'],

['text' => 'Compras', 'url' => '/compras'],

['text' => 'Clientes', 'url' => '/clientes']

]

]

]

**Repositorio GitHub:**  
<https://github.com/jurizar01/taller3.git>