Plantilla de Documentación de Prompts

Campo	Contenido
ID del Prompt	P003-Fase 3
Fecha de Uso	25/03/2025
Modelo de IA	ChatGPT o3 mini-high
Categoría de la Tarea	Frontend/Backend/Base de Datos/Autenticación/etc.
Tecnología	HTML/CSS/JavaScript/PHP/SQL/etc.
Texto Completo del Prompt	V.1 Objetivo: Generar el código necesario para la Fase 3: Gestión de productos en un sistema basado en Laravel. Requerimientos CRUD de productos para vendedores y proveedores. Los vendedores pueden gestionar productos que venden directamente a los clientes. Los proveedores pueden gestionar productos que venden directamente a los vendedores. Validaciones necesarias en la creación y edición de productos (ejemplo: precio positivo, stock mínimo). Cada usuario solo puede gestionar sus propios productos. Implementar paginación en la vista de listado de productos. Gestión de stock Opción para modificar el stock de un producto. Restricciones para evitar valores negativos. Mensajes de alerta cuando el stock esté bajo. Categorización y búsqueda Filtros por categoría, precio y disponibilidad. Búsqueda por nombre y descripción. Ordenación por precio (ascendente/descendente) y fecha de creación. Estructura del Código Separación por archivos: Cada controlador y modelo debe estar en su propio archivo. El código generado debe organizarse en múltiples archivos en lugar de incluirlo todo en un solo bloque. Extras (Opcionales pero recomendados) Middleware para restringir acceso a vendedores/proveedores según su rol. Mensajes de éxito/error en las vistas. Formato de Entrega Controladores en archivos separados, con sus respectivos métodos. Modelos con sus relaciones necesarias. Vistas Blade organizadas para cada funcionalidad. Rutas web organizadas en web.php. Migraciones y Seeders en archivos distintos. Documentos adjuntos: Documentación de la especificación y DB del proyecto 1º Iteración: Dame el método show de VendorProductController y su vista 2º Iteración: Da error la vista show: Property [name] does not exist on the Eloquent builder instance. 3º Iteracion: Con firstOrFail da error: SQLSTATE[42S22]: Column not found: 1054 Unknown column 'products.deleted_at' in 'where clause' select * from `products` where `id` = index and `user_id` = 2 and `products.deleted_at` is null 4º Iteracion: Añade los enlaces para ver los productos: @extends('layouts.app') @section('content') <link href="css/app.css" rel="stylesheet"> Perfil del Vendedor <form action="{{ route('vendor.profile.update') }}" method="POST"> @csrf <!-- Usar método POST (o PUT mediante spoofing si es necesario) --> <label for="name">Nombre</label> <input type="text" name="name" value="{{ old('name', $user->name) }}" class="form-control" required> <label for="phone">Teléfono</label> <input type="text" name="phone" value="{{ old('phone', $user->phone) }}" class="form-control"> <!-- Agregar otros campos según se requiera --> <button type="submit" class="btn btn-primary">Actualizar Perfil</button> </form> @endsection 5º Iteración: Dame el código necesario para los puestos de los vendedores y proveedores: vistas, controladores, etc... 6º Iteracion: Añade a la vista los enlaces para ver el puesto o crearlo si no tiene uno: @extends('layouts.app') @section('content') <link href="css/app.css" rel="stylesheet">

	Perfil del Vendedor Ver mis productos Crear nuevo producto <form action="{{ route('vendor.profile.update') }}" method="POST"> @csrf <!-- Usar método POST (o PUT mediante spoofing si es necesario) --> <label for="name">Nombre</label> <input type="text" name="name" value="{{ old('name', $user->name) }}" class="form-control" required> <label for="phone">Teléfono</label> <input type="text" name="phone" value="{{ old('phone', $user->phone) }}" class="form-control"> <!-- Agregar otros campos según se requiera --> <button type="submit" class="btn btn-primary">Actualizar Perfil</button> </form> @endsection 7º Iteracion: dame el comando para hacer un enlace simbolico entre public y storage 8º Iteracion: Modifica el codigo para que borre la foto antigua: public function update(Request $request, $id) { $stand = Stand::where('id', $id) ->where('user_id', Auth::id()) ->firstOrFail(); $data = $request->validate(['name' => 'required\|string\|max:255', 'description' => 'nullable\|string', 'location' => 'required\|string\|max:255', 'category' => 'nullable\|string\|max:255', 'stand_picture'=> 'nullable\|image\|max:2048',]); if ($request->hasFile('stand_picture')) { $file = $request->file('stand_picture'); $path = $file->store('stands', 'public'); $data['stand_picture'] = $path; } $stand->update($data); return redirect()->route('stand.show', $stand->id) ->with('success', 'Puesto actualizado correctamente.'); } 9º Iteracion: Dame el comando para instalar boostrap en laravel

V.1.1 **Objetivo General**: Diseñar e implementar una solución completa y modular para la gestión de productos en una plataforma web basada en Laravel, considerando distintos roles de usuario (vendedores y proveedores), buenas prácticas de desarrollo web y arquitectura MVC.

🧩 **Requerimientos Funcionales**

1. **Gestión de Productos (CRUD)**
   - Vendedores: pueden gestionar productos que venden a clientes finales.
   - Proveedores: gestionan productos destinados a los vendedores.
   - Cada usuario solo puede ver y modificar sus propios productos.
   - Validaciones:
     - Precio mayor a 0.
     - Stock no negativo.

2. **Gestión de Stock**
   - Opción para actualizar el stock.
   - Restricción para evitar valores negativos.
   - Alerta visual en la vista si el stock es bajo (menos de 5 unidades).
3. **Categorización, Filtros y Búsqueda**
   - Filtros: por categoría, precio, disponibilidad.
   - Búsqueda por nombre o descripción.
   - Ordenación por precio y fecha de creación.
4. **Paginación**
   - Implementar paginación en la vista de productos.

🧱 **Requerimientos Técnicos**

- **Estructura del Código**
  - Cada modelo y controlador en su propio archivo.
  - Vistas Blade separadas por funcionalidad.
  - Rutas organizadas en `web.php`.
  - Migraciones y seeders en archivos distintos.
- **Middleware**
  - Controlar el acceso mediante middleware por roles (`vendedor`, `proveedor`).
- **Mensajes**
  - Incluir alertas de éxito o error en las vistas.

📦 **Entregables**

- Migración de productos con `SoftDeletes`.
- Modelo `Product` con relaciones y `fillable`.
- Controladores separados para vendedores y proveedores con todas las operaciones CRUD.
- Vistas Blade:
  - Listado con filtros.
  - Formularios para crear, editar, ver y gestionar stock.
- Rutas: definidas por grupo y middleware.
- Seeder de productos (opcional pero recomendado).

Long. del Prompt	V.1 4923 caracteres	V.1.1 1703 caracteres
Documentos incluidos	Sí, Documentación de la especificación y DB del proyecto	
Ejemplos Incluidos	Sí, código que requería de modificaciones	
Calidad de la Respuesta	V.1 5	V.1.1 5
P. Ponderada	V.1 5	V.1.1 5
Prob. de Código	Falta de vistas y rutas clave Mala implementación de los Modelos Mala definición de las migraciones Falta de ejecución de comandos necesarios	
Iteraciones	9	
Mejoras Recomendadas	Estructuración más detallada, especificación de los entregables.	
Resultados de aprendizaje vinculados	V1 R1-R4: Comprensión e implementación de servicios web completos. R5, R11, R12: Autonomía en el desarrollo, trabajo colaborativo y ética profesional. R6-R10, R13-R15: Documentación, presentación, gestión del tiempo y aplicación autónoma.	
	V1.1 R1, R2, R3, R4, R5, R6, R10, R12, R14, R15	
Competencias relacionadas	V1 CETM-6, CETM-7, CETM-1, CETM-4: Diseño e implementación de servicios y arquitecturas web. CG-5: Programación en entorno de telecomunicaciones. CB-4, CB-5, CT-1, CT-2, CR-2, CR-3: Comunicación efectiva, autonomía, uso de recursos, colaboración y búsqueda de información.	
	V1.1 CETM6, CETM7, CG5, CB4, CT1, CT2, CR3	
Objetivos de la asignatura	V1 OBJ-1: Uso de tecnologías web (Laravel, Blade, etc.). OBJ-2: Desarrollo e implementación completa de un sistema web. OBJ-3: Aplicación práctica de herramientas de comercio electrónico.	
	V1.1 OBJ-1, OBJ-2, OBJ-3	

Tabla de Comparación de Versiones de Prompt

Versión	Camb. Clave	Cla.	Esp.	Prec. Téc.	Ctx.	Estruc.	Cal. Cód.	Punt. Pond.	Mej.
1.0	Inicial	5	5	5	5	5	5	5	-
1.1	Se agregaron especificaciones técnicas	5	5	5	5	5	5	5	0

Leyenda de Abreviaturas

- **Camb. Clave** = Cambios Clave
- **Cla.** = Claridad
- **Esp.** = Especificidad
- **Prec. Téc.** = Precisión Técnica
- **Ctx.** = Contexto
- **Estruc.** = Estructura
- **Cal. Cód.** = Calidad del Código
- **Punt. Pond.** = Puntuación Ponderada
- **Mej.** = Mejora