Plantilla de Documentación de Prompts

Campo	Contenido
ID del Prompt	P006-Fase 6
Fecha de Uso	28/03/2025
Modelo de IA	ChatGPT o3 mini-high
Categoría de la Tarea	Frontend/Backend
Tecnología	HTML/CSS/JavaScript/PHP
Texto Completo del Prompt	V.1 Prompt: Objetivo: Implementar la fase 6 del proyecto, centrada en integraciones API y actualización dinámica de la interfaz con AJAX en Laravel. Requerimientos Técnicos Desarrollo de API interna Crear un controlador en Laravel (ApiController) que proporcione endpoints relacionados con elementos del cms(usuarios, productos, puestos…) Implementar los modelos y relaciones necesarios en Laravel para estas funcionalidades. Asegurar que las respuestas sean en formato JSON estructurado correctamente. Integración con API de Clima (OpenWeatherMap) Implementar un método en ApiController (GET /api/clima) que recupere los datos climáticos de San Mateo, Gran Canaria. Utilizar la API de OpenWeatherMap con una clave de API configurable en .env. Implementar caché en Laravel para almacenar los datos por 30 minutos y reducir llamadas innecesarias. El resultado debe formatearse en JSON, incluyendo temperatura, condición climática, humedad y velocidad del viento. Implementación de AJAX para actualización dinámica Implementar en JavaScript (jQuery) las siguientes funcionalidades con AJAX: Widget de clima: Recuperar datos desde /api/clima y actualizar dinámicamente la interfaz sin recargar la página. Carga de productos destacados: Cargar dinámicamente los productos destacados desde /api/productos/destacados. Búsqueda de productos: Implementar un formulario de búsqueda interactivo que use AJAX para consultar /api/productos/buscar. Botón "Añadir al carrito": Implementar eventos dinámicos en botones de productos destacados y buscados para enviar la información a /carrito/agregar sin recargar la página. Asegurar que no haya conflictos entre los scripts de featured-products.js y product-search.js. Manejo adecuado de errores y estados de carga en la interfaz. Estructura de Vistas (Blade Templates en Laravel) app.blade.php (Layout principal): Debe incluir la carga correcta de Bootstrap, jQuery y otros scripts de terceros en el orden adecuado. Mantener una estructura limpia sin carga redundante de librerías. home.blade.php (Página de inicio): Incluir el widget de clima, la sección de productos destacados y el formulario de búsqueda de productos en la misma vista. Asegurar que el indicador de carga solo aparezca cuando una búsqueda esté activa. Solo cargar featured-products.js y product-search.js cuando la página de inicio esté activa. Consideraciones Adicionales Depuración: Incluir mensajes de consola (console.log) en cada función AJAX para facilitar la identificación de errores. Seguridad: Utilizar tokens CSRF en las solicitudes POST. Optimización: Minimizar el número de llamadas API innecesarias utilizando caché y eventos bien gestionados en JavaScript. Documentos adjuntos: Documentación de la especificación y DB del proyecto 1º Iteracion: da error la api de clima: GET http://mercado-local-gpt.test/api/clima 500 (Internal Server Error) 2º Iteracion: en el log sale: [2025-03-28 22:09:32] local.ERROR: Error en OpenWeatherMap API: {"cod":401, "message": "Invalid API key. Please see https://openweathermap.org/faq#error401 for more info."} 3º Iteracion: ahora sale este error en el log: [2025-03-28 22:13:41] local.ERROR: Illuminate\Http\Client\PendingRequest::get(): Argument #1 ($url) must be of type string, null given, called in C:\laragon\www\mercado-local-gpt\vendor\laravel\framework\src\Illuminate\Http\Client\Factory.php on line 461 {"exception":"[object] (TypeError(code: 0): Illuminate\\Http\\Client\\PendingRequest::get(): Argument #1 ($url) must be of type string, null given, called in C:\\laragon\\www\\mercado-local-gpt\\vendor\\laravel\\framework\\src\\Illuminate

	\\Http\\Client\\Factory.php on line 461 at C:\\laragon\\www\\mercado-local-gpt\\vendor\\laravel\\framework\\src\\Illuminate\\Http\\Client\\PendingRequest.php:768) 4º Iteracion: sigue dando error
	V.1.1 🧩 **Objetivo:** Implementar la **Fase 6** del proyecto en Laravel, que incluye: ● Desarrollo de API interna. ● Integración con API externa de clima (OpenWeatherMap). ● Actualización dinámica de la interfaz con **AJAX (jQuery)**. ● Organización modular de vistas (Blade). --- ⚙️ **Requerimientos Técnicos:** **1. API Interna en Laravel** ● Crear un `ApiController` en `app/Http/Controllers`. ● Implementar los siguientes endpoints REST con respuestas JSON bien formateadas: ○ `GET /api/productos/destacados`: Productos con `is_available = true`, ordenados por `created_at DESC`, máximo 6. ○ `GET /api/productos/buscar?q=`: Búsqueda por nombre o descripción. ○ `GET /api/usuarios` y `GET /api/puestos`: Información básica con relaciones si aplica. **Modelos necesarios**: `Product`, `User`, `Stand` con relaciones Eloquent bien definidas. --- **2. Integración con API de Clima (OpenWeatherMap)** ● Endpoint `GET /api/clima` que: ○ Use la URL y clave API desde `.env` (`OPENWEATHERMAP_URL`, `OPENWEATHERMAP_KEY`). ○ Obtenga datos del clima para **San Mateo, Gran Canaria**. ○ Implemente caché (`Cache::remember`) de 30 minutos. ○ Devuelva JSON con: temperatura, condición climática, humedad, velocidad del viento. ○ Maneje errores con `try/catch` y registre logs (`\Log::error()`). **Validar que config/services.php esté correctamente configurado** y ejecutar `php artisan config:cache` tras cualquier cambio en `.env`. --- **3. AJAX con jQuery** Implementar los siguientes scripts JS en `public/js/`:

	• `featured-products.js`: Carga de productos destacados desde `/api/productos/destacados`.
	• `product-search.js`: Formulario de búsqueda dinámica con resultados desde `/api/productos/buscar?q=`.
	• Botones `.add-to-cart`: Envío `POST /carrito/agregar` con CSRF token incluido.
	
	Todos los scripts deben incluir `console.log()` para depuración y manejar errores visualmente en la interfaz.
	
	--- 	
	
	4. Vistas Blade (Laravel)
	
	• **Layout principal** (`app.blade.php`):
	  ○ Incluir Bootstrap, jQuery y `@yield('scripts')`.
	  ○ Carga correcta y sin redundancia de librerías.
	• **Página de Inicio** (`home.blade.php`):
	  ○ Incluir:
	  ■ Widget de clima.
	  ■ Sección de productos destacados.
	  Buscador interactivo.
	  ○ Cargar `featured-products.js`, `product-search.js` y script inline para `/api/clima` sólo en esta vista.
	  ○ Mostrar indicadores de carga únicamente cuando una operación AJAX esté en progreso.
	
	--- 	
	
	✅ **Consideraciones Adicionales:**
	
	• **Depuración**: Incluir logs de consola y verificar mensajes en `storage/logs/laravel.log`.
	• **Seguridad**: Usar CSRF en todos los formularios POST.
	• **Optimización**: Minimizar llamadas innecesarias gracias al uso de caché y control de eventos en JS.
	• **Validación final**: Asegurarse de que todas las variables `.env` estén correctamente leídas por Laravel y que `config/services.php` esté bien definido.

Long. del Prompt	V.1 Sin: 2742 caracteres Junto iteraciones: 3805 caracteres / V.1.1 2643 caracteres
Documentos incluidos	Sí, Documentación de la especificación y DB del proyecto
Ejemplos Incluidos	Sí, mensajes de error a solucionar
Calidad de la Respuesta	V.1 5 / V.1.1 5
P. Ponderada	V.1 5 / V.1.1 5
Prob. de Código	Problemas con el manejo de APIs y uso de javascript
Iteraciones	4
Mejoras Recomendadas	Mayor especificaciones para evitar futuras iteraciones y/o errores
Resultados de aprendizaje vinculados	V1 R2: Conoce diferentes arquitecturas o modelos de software de comunicación. R3: Conoce y usa tecnologías, lenguajes y protocolos web (AJAX, API REST, Laravel). R4: Diseña, implementa y valida servicios web. R15: Aplica las competencias adquiridas de forma autónoma.
	V1.1 R1 Comprende la relación entre los elementos del sistema: API, AJAX, vistas, caché, configuración. R2 Distingue arquitecturas de comunicación (API REST, AJAX, MVC en Laravel). R3 Utiliza tecnologías web actuales: Laravel, jQuery, JSON, HTTP, entorno .env.

	R4 Diseña e implementa un sistema web funcional con servicios integrados. R5 Muestra interés y aplica tecnologías clave en programación web y telecomunicaciones. R6-R7 Exige redacción de código comprensible, documentación clara y posibilidad de presentación oral. R8, R12 Se requiere leer y comprender documentación en inglés (API externa) y validar fuentes. R14, R15 Requiere autonomía, cumplimiento de tareas y planificación de pruebas sin iteraciones futuras.
Competencias relacionadas	V1 CETM-6: Diseña arquitecturas de redes y servicios telemáticos. CETM-1: Construcción y gestión de servicios telemáticos. CETM-4: Describe, valida y optimiza protocolos. CETM-7: Programación de servicios y aplicaciones en red.
	V1.1 Básica CB2, CB3, CB5 Aplicación de conocimientos, análisis de datos, autonomía. General CG5 Diseño y programación en entornos de telecomunicaciones. Transversal CT1, CT2, CT3 Comunicación técnica, colaboración, mejora continua. Específica (Telemática) CETM-1, CETM-4, CETM-6, CETM-7 Arquitectura de servicios web, programación distribuida, validación de interfaces y protocolos.
Objetivos de la asignatura	V1 OBJ-1: Uso de tecnologías web tanto del lado cliente como servidor. OBJ-2: Desarrollo de un sistema web completo con acceso a base de datos. OBJ-3: Aplicación práctica dentro del contexto de una tienda de comercio electrónico.
	V1.1 OBJ-1 Uso de tecnologías web del lado cliente y servidor. OBJ-2 Desarrollo completo de un sistema web funcional. OBJ-3 Aplicación a un modelo de tienda (uso de carrito, productos, clima, búsquedas).

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