Requisitos Funcionales y Técnicos – Proyecto E■commerce

Este documento define los requisitos funcionales y técnicos necesarios para un sistema de comercio electrónico robusto, escalable y preparado para producción. Se basa en las funcionalidades implementadas y en buenas prácticas de arquitectura, seguridad y experiencia de usuario.

1. Gestión del ciclo de vida del producto

- El sistema debe manejar estados de producto: activo, inactivo y borrador.
- Solo los productos activos pueden visualizarse y añadirse al carrito.
- Cada cambio de estado debe registrarse con auditoría (usuario, fecha y motivo).

2. CRUD completo de productos

- Crear, editar, listar y eliminar productos mediante interfaces administrativas.
- Soportar carga, actualización y eliminación de imágenes asociadas al producto.
- Eliminaciones lógicas recomendadas para preservar historial y métricas.

3. Variantes e inventario

- Un producto puede tener múltiples variantes (talla, color u otros atributos).
- Cada variante debe contar con SKU único, precio, stock independiente y atributos en formato JSON.
- La selección de variante es obligatoria antes de añadir un producto al carrito.

4. Carrito de compras

- El carrito debe permitir agregar, actualizar y eliminar productos y variantes.
- No se permiten cantidades menores a 1 ni superiores al stock disponible.
- El carrito debe persistir por sesión o usuario autenticado.

5. Filtros y vistas de productos

- El sistema debe permitir filtrar productos por precio, categoría, estado y atributos.
- Las vistas deben ser optimizadas para rendimiento y experiencia de usuario.
- Soporte para paginación y carga eficiente de imágenes.

6. Integración con Cloudinary

- Las imágenes deben almacenarse en Cloudinary mediante uploads firmados.
- Las credenciales deben gestionarse exclusivamente desde variables de entorno.
- Guardar en base de datos la URL pública y el public_id de cada imagen.

7. Estructura de base de datos

- Modelo normalizado con tablas separadas para productos, variantes, imágenes y carrito.
- Uso de campos JSON para atributos flexibles como colores y tallas.
- Uso de transacciones para operaciones críticas como actualización de stock.

8. Seguridad y validaciones

- Validación de datos tanto en frontend como backend.

- Protección contra manipulación de precios y cantidades.
- Sanitización de inputs y control de acceso por roles.

9. Arquitectura y estructura de código

- Código organizado por módulos y responsabilidades claras.
- Separación entre lógica de negocio, vistas y acceso a datos.
- Preparado para escalabilidad y mantenimiento a largo plazo.

10. Pruebas, monitoreo y calidad

- Pruebas unitarias para lógica crítica (carrito, precios, stock).
- Pruebas de integración para APIs y servicios externos.
- Logs, monitoreo y manejo de errores en entorno productivo.