NicoReparaciones - Documentación histórica y técnica (verificada con comandos Artisan)

Plataforma Web de Tienda + Reparaciones (corte: 12 de diciembre de 2025)

Documento actualizado con salidas reales de **php artisan about**, **migrate:status** y **route:list**.

Ficha técnica (artesan)

Item	Valor
Laravel Version	12.42.0
PHP Version	8.2.12
Environment	local
Debug	ENABLED
URL	localhost
Timezone	UTC
Cache	database
Database	mysql
Queue	database
Session	file
Views	CACHED
Public storage linked	NOT LINKED (public/storage)

Índice

1. Visión y objetivos

- **Qué es:** una plataforma web propia para un local de reparación de celulares y venta de accesorios, pensada para ciudad chica, con foco en simplicidad y control total (sin depender de SaaS cerrados).
- **Objetivos del dueño (Nico):** centralizar stock, pedidos y (próximamente) reparaciones; evitar pedidos falsos; medir costos y ganancias; mejorar visibilidad local.
- **Objetivos del cliente:** navegar productos desde el celular, armar carrito, hacer pedidos y consultar estado (pedidos y reparaciones).

2. Estado actual (verificado)

- Proyecto ejecutando en entorno **local** con **Laravel 12.42.0** y **PHP 8.2.12**.
- Debug Mode: **ENABLED** (correcto para desarrollo; en producción debe ir desactivado).
- Base URL: **localhost**.
- Timezone actual: **UTC** (recomendación: pasar a *America/Argentina/Cordoba* para que pedidos/reparaciones tengan horas locales).
- Sesiones: **file**. Cache: **database**. Queue: **database**. DB: **mysql**.
- Estado storage público: **public/storage NOT LINKED** (falta symlink).

3. Alcance implementado (MVP funcional)

- Autenticación (login/registro/logout).
- Tienda (home + listado /tienda, categorías y páginas de producto por slug).
- Carrito por sesión (agregar/quitar/modificar/vaciar) con límite por stock.
- Checkout (GET /checkout) y confirmación (POST /checkout/confirmar).
- Pedidos (cliente: /mis-pedidos y /mis-pedidos/{order}).
- Panel Admin de pedidos (listar, ver detalle y cambiar estado).

4. Migraciones y tablas (confirmado por migrate:status)

- Todas las migraciones están en estado **Ran** (ejecutadas).
- Core Laravel: users, cache, jobs.
- Negocio: categories, products, orders, order_items.
- **Lista exacta:** 0001_01_01_000000_create_users_table, 0001_01_01_000001_create_cache_table, 0001_01_01_000002_create_jobs_table, 2025_12_11_151734_create_categories_table, 2025_12_11_152136_create_products_table, 2025_12_11_172006_create_orders_table, 2025_12_11_172007_create_order_items_table.

5. Rutas reales (route:list)

- Rutas principales de tienda:
- • GET / (home) -> StoreController@index
- • GET /tienda -> StoreController@index
- • GET /tienda/categoria/{slug} -> StoreController@category
- • GET /producto/{slug} -> StoreController@product
- Carrito y checkout:
- • GET /carrito -> CartController@index
- • POST /carrito/agregar/{product} -> CartController@add
- • POST /carrito/actualizar/{product} -> CartController@update
- • POST /carrito/eliminar/{product} -> CartController@remove
- • POST /carrito/vaciar -> CartController@clear

- • GET /checkout -> CartController@checkout
- • POST /checkout/confirmar -> OrderController@confirm
- Pedidos cliente:
- • GET /mis-pedidos -> OrderController@index
- • GET /mis-pedidos/{order} -> OrderController@show
- Auth:
- • GET /login -> AuthController@showLogin
- • POST /login -> AuthController@login
- • POST /logout -> AuthController@logout
- • GET/POST /registro -> AuthController@showRegister / register
- Admin pedidos:
- • GET /admin/pedidos -> AdminOrderController@index
- • GET /admin/pedidos/{order} -> AdminOrderController@show
- • POST /admin/pedidos/{order}/estado -> AdminOrderController@updateStatus
- Nota: existe GET /storage/{path} (storage.local) típico de entorno local.

6. Detalles de UX/negocio ya implementados

- Cliente no ve cantidades exactas de stock (solo estado).
- Popup al agregar al carrito (tipo MercadoLibre) para evitar redirecciones.
- UI minimalista mobile-first.

7. Checklist de pruebas recomendadas (alineado a tus rutas)

- **Tienda:** /, /tienda, /tienda/categoria/{slug}, /producto/{slug} (slugs válidos e inválidos).
- **Carrito:** agregar/actualizar/eliminar/vaciar. Intentar exceder stock (no debe permitir).
- **Checkout:** GET /checkout sin login (si lo protegiste con auth, debe redirigir a /login). Confirmar pedido con carrito vacío (debe bloquear).
- **Pedidos:** /mis-pedidos muestra solo del usuario; probar abrir /mis-pedidos/{order} de otro usuario (debe 403 o 404).
- **Admin:** usuario común intentando /admin/pedidos (403). Admin: listar, ver y cambiar estado.
- **Persistencia:** al confirmar pedido: orders + order_items se crean, totales correctos, y (si corresponde) stock actualizado.

8. Fixes rápidos detectados (por artisan about)

- **Storage link:** falta symlink (public/storage NOT LINKED). Solución: *php artisan storage:link* (con tu php.exe de XAMPP).
- **Timezone:** está en UTC. Si querés horas locales, cambiar en config/app.php (timezone) o en .env si lo estás leyendo desde ahí.
- **Producción:** recordar APP_DEBUG=false y cachear config/routes cuando se despliegue.

9. Próximo módulo: Reparaciones (roadmap técnico)

- Agregar modelo y migración repairs (y opcional historial de estados).
- Estados recomendados: recibido -> diagnosticando -> esperando_aprobación -> en_reparación -> listo_retirar -> entregado -> cancelado.
- Campos mínimos: código, cliente (user_id o nombre/teléfono), marca/modelo, falla, diagnóstico, costos (repuestos/mano_obra), precio_final, estado, garantía_días, fechas, notas.
- Vistas: admin (CRUD + filtros + detalle) y cliente (consulta de estado por código/teléfono o por login).

- Reglas clave: auditoría de cambios de estado y cálculo de ganancia = precio_final - costos.

10. Estado actual y siguiente paso

- Estado: estable y usable en local; rutas y DB listas.
- Siguiente paso lógico: implementar **MVP de reparaciones** y conectarlo con consulta de estado del cliente.