

1) Objetivo

Este sistema es una plataforma SaaS multi-tenant para gestionar internacion domiciliaria. Integra clinica, logistica, inventario y facturacion, con control de permisos por rol y por plan.

2) Stack tecnologico

- Next.js 16 (App Router) y React 19
- TypeScript
- NextAuth con credenciales
- Prisma + PostgreSQL
- Stripe (suscripciones y portal de pagos)
- S3/MinIO (evidencias y adjuntos)
- PDFKit (PDFs de remitos y exportaciones)
- Tailwind CSS + shadcn/ui
- Vitest

3) Estructura del repositorio

- src/: codigo fuente de la app.
- src/app/: rutas UI y API (App Router).
- src/lib/: logica de negocio compartida (auth, rls, billing, etc).
- src/components/: componentes UI y de aplicacion.
- src/types/: tipos y augmentations (NextAuth).
- prisma/: esquema, migraciones y seed.
- docs/: documentacion (incluye docs/rls.md).
- public/: assets estaticos.
- tests/: pruebas automatizadas.
- docker-compose.yml: Postgres + MinIO para entorno local.
- Configuracion root: package.json, tsconfig.json, eslint.config.mjs, next.config.ts, postcss.config.mjs, components.json.

4) Rutas y modulos (UI)

- /login (grupo (auth)): inicio de sesion con credenciales.
- /dashboard: KPIs operativos basicos (pacientes, episodios, visitas, entregas, incidentes).
- /patients: alta y listado de pacientes.
- /episodes: alta de episodios, workflow, plan de cuidado, alta/egreso.
- /episodes/[id]: detalle de episodio con timeline y visitas.
- /agenda: programacion de visitas, checklist, notas clinicas, consumos y adjuntos.
- /inventory/products: catalogo de productos con reposicion minima.
- /inventory/warehouses: depositos y ubicaciones.
- /inventory/stock: movimientos, lotes y alertas de stock/vencimientos.
- /logistics/orders: ordenes aprobadas y kits.
- /logistics/picklists: asignacion de depositos, congelado, incidentes y packing.
- /logistics/deliveries: firmas, evidencia, PDF de remito y cierre.
- /payers: obras sociales, planes y requisitos.
- /authorizations: autorizaciones por paciente y carga de requisitos.
- /billing: estado del plan, checkout y portal Stripe, bloqueos por mora.
- /billing/preliquidation: control autorizado vs realizado vs evidenciado.
- /billing/invoices: generacion y exportacion de facturas.
- /billing/rules: reglas de facturacion por payer/plan y producto.

- /billing/debits: debitos o rechazos sobre facturas.
- /billing/payments: pagos y conciliacion.
- /billing/aging: aging de saldos por rangos.
- /kpis: analitica y KPIs clinicos, logísticos y financieros.
- /onboarding (superadmin): alta de tenants y admins.
- /superadmin/tenants (superadmin): monitoreo basico de tenants.

5) API (endpoints principales)

- POST /api/auth/[...nextauth]: autenticacion.
- POST /api/stripe/webhook: eventos de suscripcion y estado de tenant.
- GET /api/billing/preliquidation/export: export CSV de preliquidacion.
- GET /api/billing/invoices/export: export CSV o PDF de facturas.
- GET /api/kpis/export: export CSV de KPIs.
- GET /api/deliveries/[id]/pdf: PDF de remito/acta de entrega.
- POST /api/deliveries/[id]/evidence: subida de evidencia.
- POST /api/visits/[id]/attachments: adjuntos clinicos (imagen o PDF).
- POST /api/authorizations/requirements/[id]/upload: adjunto de requisito.

6) Seguridad, roles y multi-tenant

- Middleware protege rutas privadas en src/middleware.ts.
- Roles disponibles: SUPERADMIN, ADMIN_TENANT, COORDINACION, DEPOSITO, LOGISTICA, PROFESIONAL, FACTURACION, AUDITOR.
- Control de modulos por plan en src/lib/tenant-access.ts.
- RLS: withTenant y withSuperadmin setean app.tenant_id y app.is_superadmin.

7) Base de datos (resumen de modelos)

- Tenant y User: multi-tenant y roles.
- Patient, Episode, Visit, ClinicalNote, ClinicalAttachment: nucleo clinico.
- Product, Warehouse, Batch, StockMovement: inventario.
- ApprovedOrder, PickList, Delivery, Incident: logistica y despacho.
- Payer, PayerPlan, PayerRequirement, Authorization: obras sociales y autorizaciones.
- Invoice, InvoiceItem, BillingRule, DebitNote, Payment: facturacion.
- AuditLog: trazabilidad de acciones.

8) Integraciones externas

- Stripe: suscripciones, precios y portal.
- S3/MinIO: almacenamiento de evidencias y adjuntos.
- PDFKit: generacion de PDFs (remitos y exportaciones).

9) Variables de entorno (principales)

- DATABASE_URL: conexion PostgreSQL.
- STRIPE_SECRET_KEY, STRIPE_WEBHOOK_SECRET, STRIPE_PRICE_*, STRIPE_*_URL: Stripe.
- S3_BUCKET, S3_REGION, S3_ENDPOINT, S3_ACCESS_KEY_ID, S3_SECRET_ACCESS_KEY, S3_PUBLIC_URL, S3_FORCE_PATH_STYLE: S3/MinIO.
- DELIVERY_MIN_EVIDENCE, VISIT_SLA_MINUTES, DELIVERY_SLA_HOURS: reglas operativas.
- SUPERADMIN_EMAIL, SUPERADMIN_PASSWORD, DEFAULT_TENANT_SLUG, DEFAULT_TENANT_ADMIN_EMAIL, DEFAULT_TENANT_ADMIN_PASSWORD: seed y onboarding.

10) Scripts utiles

- npm run dev: levantar frontend.
- npm run build / npm run start: build y produccion.
- npm run prisma:migrate, npm run prisma:generate, npm run db:push, npm run seed: base de datos.
- npm run test: ejecutar pruebas.

11) Operacion local recomendada

- Levantar servicios: docker-compose up -d.
- Configurar .env con las variables necesarias.
- Ejecutar migraciones y seed.
- Iniciar npm run dev.

12) Tests

- tests/smoke.test.ts: pruebas basicas de formateo de numeros.