MANUAL TÉCNICO

Marco Ramirez

Tabla de contenidos

Introducción

Este documento presenta el desarrollo técnico del sistema de gestión de indicios. Se describen las tecnologías empleadas, la estructura del backend, las rutas disponibles, la lógica de autenticación y autorización, así como aspectos relevantes del modelo de datos y arquitectura general.

Tecnologías utilizadas

- **Node.js** con **Express** para la creación de la API RESTful.
- **Prisma ORM** para la manipulación de base de datos.
- **SQL Server** como sistema de gestión de base de datos, ejecutado en contenedor Docker.
- **JWT** (JSON Web Tokens) para la autenticación de usuarios.
- **Docker y Docker Compose** para la orquestación de servicios.
- **Bun** y **Next.js** para el frontend (en desarrollo).

Arquitectura general

El sistema se compone de:

- **Frontend (Next.js)**: Interfaz de usuario.
- **Backend (Express)**: API con lógica de negocio.
- **Base de datos (SQL Server)**: Almacena usuarios, expedientes e indicios.

El backend expone endpoints REST protegidos por JWT, que son consumidos por el frontend.

Modelo de datos

Entidades principales:

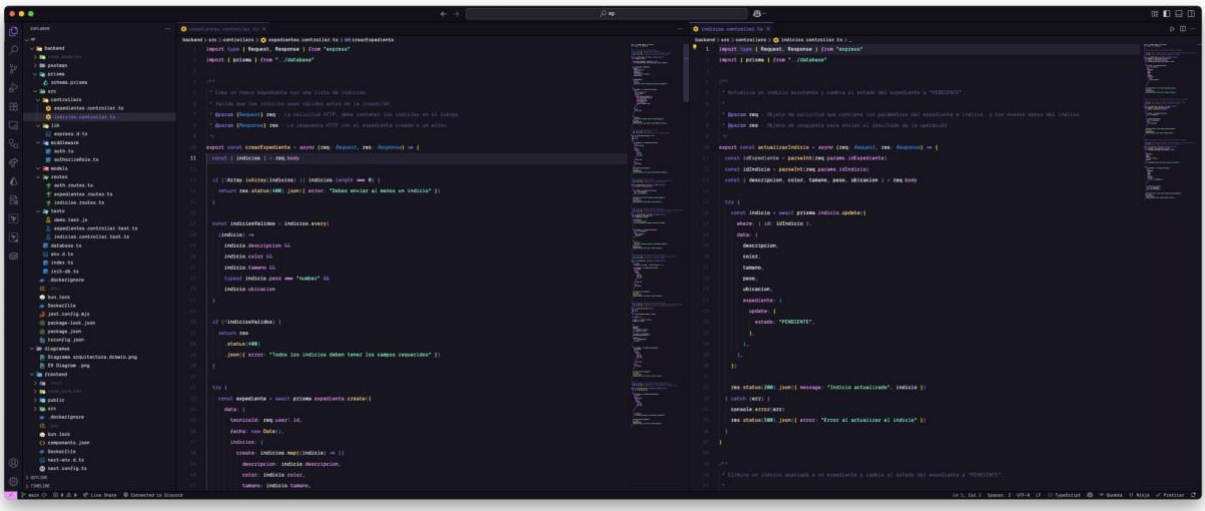
- **User**: contiene id, email, name, password, role.
- **Expediente**: asociado a un técnico; contiene estado, fecha, justificación.
- **Indicio**: relacionado a un expediente; incluye campos como descripción, color, tamaño, peso y ubicación.

Estructura del backend

Ruta: backend/src

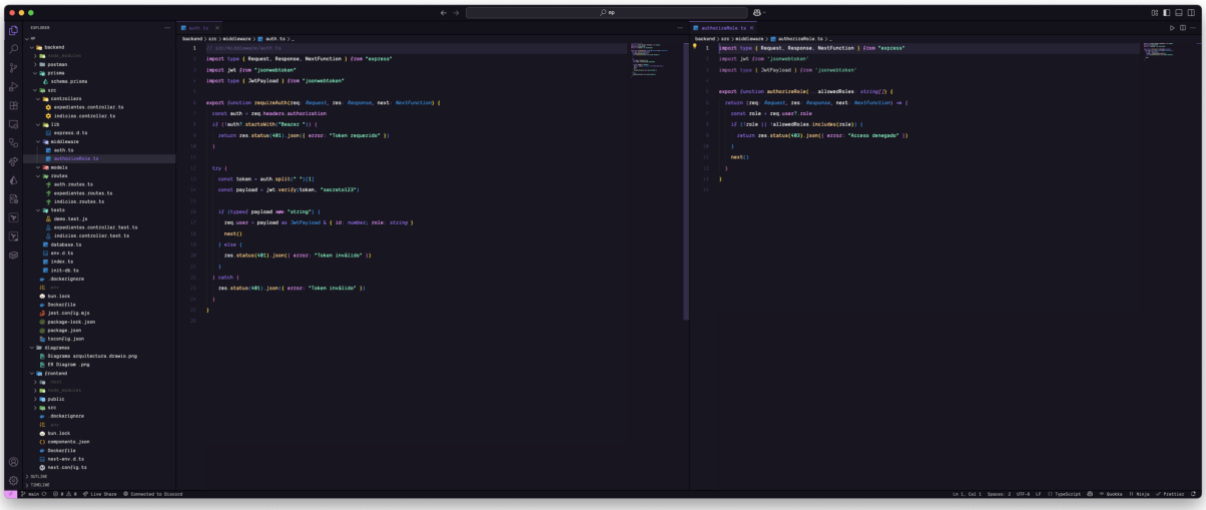
5.1 Controladores

- expedientes.controller.ts: CRUD completo de expedientes, con validaciones y filtrado por rol.
- indicios.controller.ts: Alta, edición y eliminación de indicios. Cualquier cambio marca el expediente como “PENDIENTE”.



5.2 Middleware

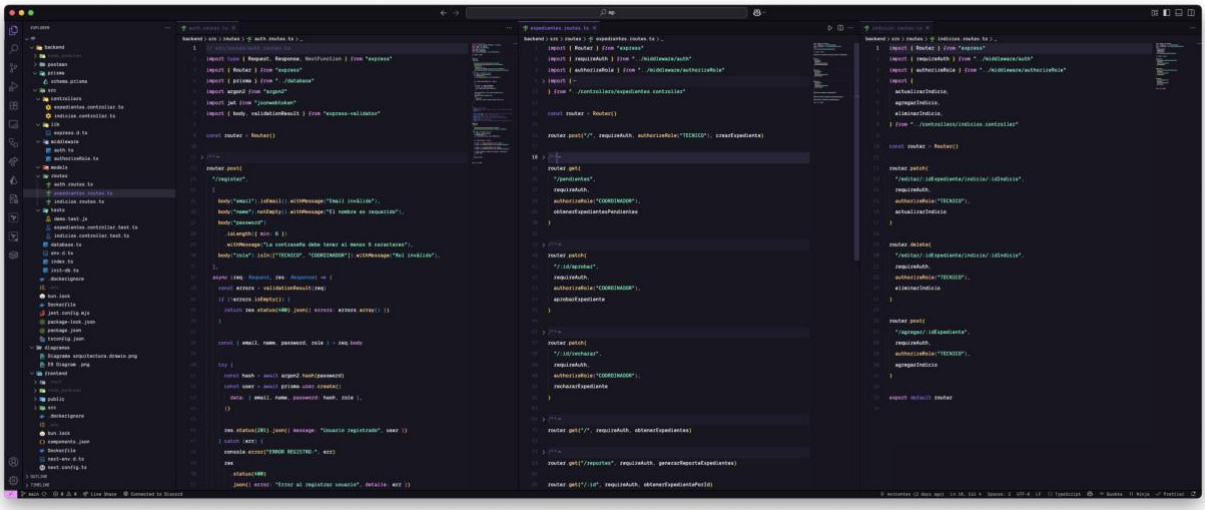
- auth.ts: Verifica JWT, inyectando req.user si es válido.
- authorizeRole.ts: Restringe rutas por rol (TECNICO o COORDINADOR).



5.3 Rutas

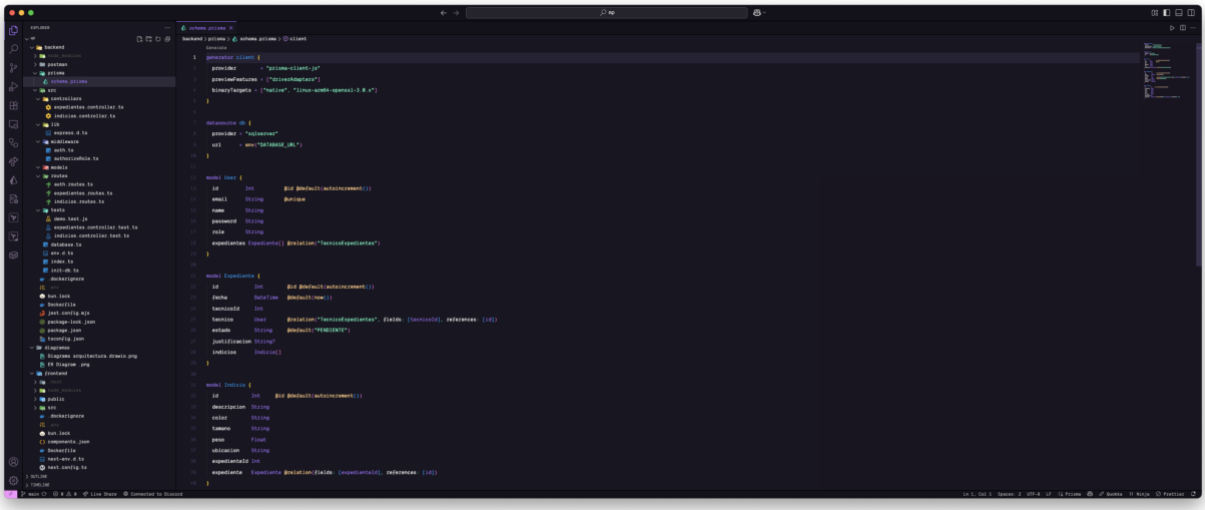
- auth.routes.ts:
    - POST /api/auth/register: registro con validación.
    - POST /api/auth/login: login con generación de JWT.
- expedientes.routes.ts:
    - POST /: crear expediente.
    - GET /pendientes: listado para coordinadores.
    - PATCH /:id/aprobar y /rechazar: cambia estado.
    - GET /: lista general filtrada por rol.
    - GET /reportes: reportes por estado y fechas.
- indicios.routes.ts:
    - POST /agregar/:idExpediente

- PATCH /editar/:idExpediente/indicio/:idIndicio
- DELETE /editar/:idExpediente/indicio/:idIndicio



5.4 Prisma

- Archivo schema.prisma define los modelos y relaciones.
- Comandos prisma generate y prisma migrate permiten sincronizar el esquema con la DB.



Seguridad y autenticación

- Los usuarios se registran con contraseña encriptada con **argon2**.
- El login genera un JWT firmado que incluye id y role.
- Todas las rutas protegidas requieren el header: