Consideraciones Actualizadas del Proyecto AtlasNode

1. Objetivo General

Crear un orquestador centralizado que gestione múltiples microsites legales ("unitarios"), unificando datos y lógica de negocio, y ofreciendo a cada nicho:

- Landing estática optimizada para SEO.
- Sección de noticias automáticas.
- Zona privada de cliente (perfil, chat, compartición de archivos, reportes).
- Portal de profesionales para abogados.
- Panel de administración central con supervisión, configuración y control de calidad.

2. Arquitectura de Datos (Prisma Schema)

Se utiliza PostgreSQL con Prisma como ORM. Modelo de datos principal:

- User: id, email, password, role (client, professional, admin), datos personales y timestamps.
- Client, Professional: vinculan usuarios a un UnitServer, almacenan casos, archivos y reportes.
- Case: relaciona cliente, profesional y servidor; estado (open, inProgress, pending, closed); chat, archivos, reportes.
- Chat y Message: historial de conversaciones por caso, con remitente y timestamp.
- File y Report: gestión de documentación y reportes de incidencias.
- UnitServer, Constellation, UnitConfig, Section, ManualArticle, AutoSource, Image: definiciones para servidores unitarios, configuraciones y contenidos manuales/automáticos.

3. Orquestador Central

El orquestador central expone el API GraphQL, gestiona autenticación y coordina despliegues.

- Tecnologías: Next.js + GraphQL + Prisma (+ NextAuth/JWT).
- Endpoints: Queries y Mutations de páginas, usuarios, chat, archivos, reportes, noticias.
- Fronts: /admin: administración global (configuración, validación IA, KPIs). /pro: portal para profesionales (gestión de casos, chat, archivos).
- Procesos: Webhooks: POST /api/refresh a unitarios cuando se actualiza configuración. Configuración y despliegue de unitarios a través de la API central.

4. Panel de Administración Central

El panel central permite gestionar servidores, usuarios y contenidos de forma unificada.

1. 1. Servidores Unitarios: listado, creación, tokens, prueba de conexión, analíticas.

2. 2. Configuraciones: CRUD de configuraciones (título, secciones, noticias, enlaces), asignación a servidores y flags de actualización.
3. 3. Profesionales: alta/baja, asignación a servidores, estadísticas.
4. 4. Clientes: listado global, filtros (servidor/profesional), asignaciones, incidencias.
5. 5. Supervisión de Casos: estado, historial, búsquedas por cliente/profesional/servidor.
6. 6. Contenido Legal: artículos manuales, secciones fijas, fuentes automáticas (AutoSources).
7. 7. Dashboard: KPIs, métricas por servidor, tiempos medios de respuesta.
8. 8. Seguridad y Ajustes: gestión de administradores, logs, backups, variables globales.

5. Portal de Profesionales

Este espacio está diseñado para que los profesionales (abogados, peritos, expertos) gestionen sus casos y comunicaciones de forma eficiente:

- Dashboard: métricas de casos asignados, mensajes pendientes y alertas de plazos.
- Gestión de casos: listado con filtros (cliente, estado, fecha) y acceso al detalle de cada caso.
- Detalle de caso: chat interno, documentos adjuntos, generación de informes y checklist de tareas.
- Repositorio de documentos: organización por caso, plantillas propias, carga masiva y versionado.
- Calendario y agenda: plazos legales, citas y reuniones con exportación a calendarios externos.
- Recursos legales: biblioteca de artículos manuales y configuración de fuentes automáticas (AutoSources).
- Perfil y configuración: datos personales, preferencias de notificaciones, gestión de credenciales y firma digital.
- Notificaciones y alertas: historial de avisos del sistema, con ajustes de frecuencia y canal.
- Soporte: acceso a FAQ, chat de ayuda o tickets, y logs de actividad para auditoría.

6. Flujo de Comunicación y Despliegue de Unitarios

Describe cómo el orquestador notifica y los unitarios actualizan su configuración de forma coordinada.

- Creación: registro de nuevo servidor unitario en el orquestador, generación de tokens (orquestador ↔ unitario) y persistencia en la BD.
- Despliegue: unitario se despliega en Vercel/Netlify con endpoint /config para introducir tokens.
- Prueba de Conexión: orquestador verifica autenticación bidireccional.
- Creación de Configuración: administrador define parámetros de landing (título, footer, secciones, enlaces, scraping de noticias) y guarda como UnitConfig.
- Envío de Configuración: se envía flag de actualización; el unitario consulta GraphQL, genera config.json, newsCache.json e HTML estático.

- Gestión Posterior: clonación, sobrescritura o reenvío de configuraciones a uno o varios servidores.

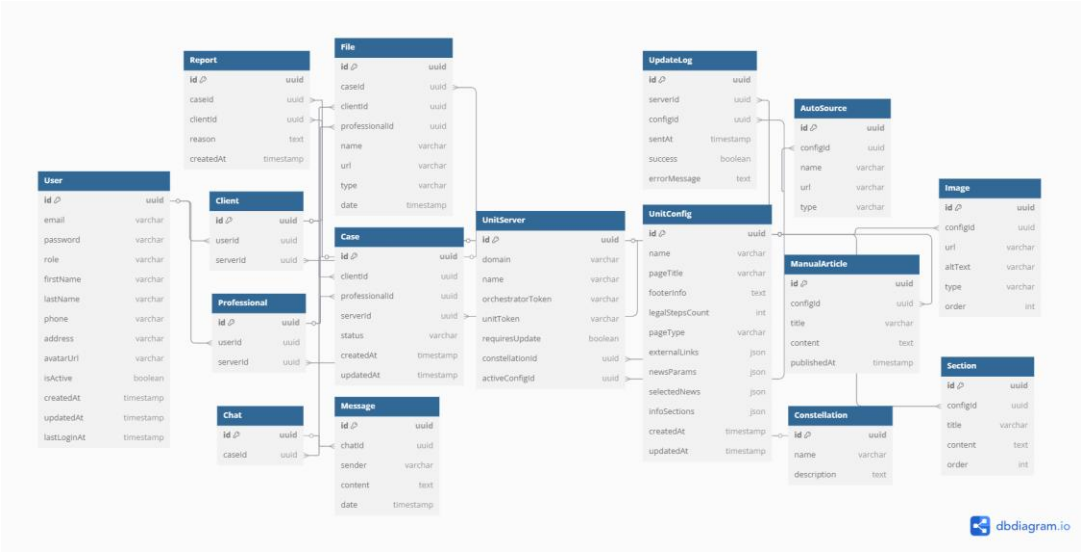
7. Microsites Unitarios

Cada microsite unitario consume configuración y datos del orquestador para generar contenido estático y dinámico.

- Generación de landing estática usando getStaticProps y datos de config.json y newsCache.json.
- ISR on-demand vía /api/refresh.
- Back-office ligero interno para edición de config.json y cron de scraping/RSS.
- Consumo de API GraphQL central para chat, perfil, archivos y reportes.

8. Diagrama de Arquitectura

Figura: Visión global de orquestador, BD central, microsites unitarios y flujos de actualización.



9. Despliegue y Operaciones

- Orquestador: Vercel (Serverless Functions), AWS Lambda/ECS-Fargate.
- Unitarios: Vercel, Netlify.
- Base de Datos: RDS/Railway con backups y réplicas.
- Escalabilidad: contenedores, balanceo y tolerancia a fallos.
- Monitorización: logs centralizados, alertas de incidencias y métricas de rendimiento.