

Manual Técnico

Documentación

Sistema de Información para el Cambio Climático del Ministerio
de Minas y Energía

GearsMap

25 de junio de 2026

Versión 1.0

Tabla de contenido

1. Arquitectura técnica	1
2. Estructura del código fuente	3
3. Configuración de rutas API	4
4. Flujo de autenticación	6
5. Motor de derivación MRV	7
6. Conectores de datos	8
7. Pipeline de procesamiento Geoespacial M&E	9
8. Sincronización automática (cron diario)	11

1. Arquitectura técnica

1.1. Visión general

SICMME sigue una arquitectura de tres capas con dos backends asimétricos:

```
[Navegador] ↔ [Next.js :3000] ↔ [PostgreSQL :5432]
                               ↔ [FastAPI :8001] ↔ [PostgreSQL :5432]
                               ↔ [TiTiler :8090]
```

1.2. Frontend (Next.js)

Aspecto	Detalle
Framework	Next.js 16.1.3 con App Router
Lenguaje	TypeScript 5.9
Estilos	Tailwind CSS 4
Componentes	shadcn/ui sobre Radix primitives
Mapas	Leaflet (vectorial), MapLibre GL (ráster)
Gráficos	Apache ECharts 6
Autenticación	better-auth 1.6.18
Validación	react-hook-form + zod
URL state	nuqs

1.3. Backend MRV/INGEI (FastAPI)

Aspecto	Detalle
Framework	FastAPI (Python 3.11+)
Servidor	uvicorn
Base de datos	asyncpg (pool asíncrono)
Cálculos	pandas (IPCC 2006)
Conectores	pydataxm (XM SinerGox)

Programación	APScheduler (cron in-process)
--------------	-------------------------------

1.4. Base de datos

Aspecto	Detalle
Motor	PostgreSQL 14+ con PostGIS
Host	172.17.2.81 (VPN MinEnergía)
Base de datos	dbmrvme
Esquemas	auth, mye, mrv, public

2. Estructura del código fuente

```

src/
├─ app/                                # Páginas y layouts (App Router)
│  ├─ (auth)/                          # Grupo de autenticación
│  ├─ (dashboard)/                    # Grupo de dashboard
│  │  ├─ cambio-climatico/
│  │  │  ├─ mye/                      # Páginas M&E
│  │  │  └─ mitigacion/mrv/          # Páginas MRV
│  │  └─ prediccion-climatica/
│  └─ api/                            # Route handlers
│     ├─ auth/
│     ├─ mye/                         # API M&E (~30 endpoints)
│     ├─ mrv/                         # API MRV (rewrite a FastAPI)
│     ├─ ingei/
│     └─ ...
├─ components/                        # Componentes React
│  ├─ admin/                          # AdminGate, panel admin
│  ├─ mapas/                          # Leaflet, MapLibre
│  ├─ mrv/                            # Catálogo, fichas, calculadora
│  └─ ui/                             # shadcn/ui
├─ lib/                              # Utilidades y lógica
│  ├─ db.ts                           # Pool pg M&E
│  ├─ auth.ts                         # Configuración better-auth
│  └─ mrv/                            # Catálogo estático MRV
├─ types/                            # Definiciones TypeScript
└─ hooks/                            # Hooks personalizados

backend/
├─ main.py                           # Punto de entrada FastAPI
├─ cambio_climatico/
│  ├─ mye/back/                       # Pipeline M&E (Python offline)
│  └─ mitigacion/
│     ├─ mrv/                         # Router MRV FastAPI
│     └─ ingei/                      # Router INGEI FastAPI
└─ requirements.txt

scripts/
├─ cron_daily.py                     # Sincronización automática MRV
└─ seed-auth-users.ts                # Usuarios de semilla

```

3. Configuración de rutas API

3.1. Rutas nativas de Next.js

Ruta	Método	Propósito
/api/mye/geometrias	GET, POST	Geometrías de municipios y departamentos
/api/mye/valores	POST	Valores de indicadores M&E
/api/mye/estadisticas/*	GET, POST	Estadísticas cuantitativas y cualitativas
/api/mye/admin/*	GET, POST, PUT, DELETE	Administración M&E
/api/auth/*	GET, POST	Autenticación (better-auth)
/api/ingei/estimate	POST	Cálculo INGEI (proxy a FastAPI)
/api/visibilidad-indicadores	GET, PUT	Visibilidad MRV (Next nativa)
/api/fuentes-indicadores	GET, PUT	Fuentes MRV (Next nativa)
/api/adaptation-actions/*	GET, POST, PUT	Acciones de adaptación

3.2. Rutas reescritas a FastAPI

```
// next.config.ts
async rewrites() {
  return [
    {
      source: '/api/mrv/:path*',
      destination: `${INGEI_SERVICE_URL}/mrv/:path*`,
    },
  ];
}
```

Ruta FastAPI	Método	Propósito
--------------	--------	-----------

/mrv/indicadores	GET	Lista de indicadores
/mrv/indicadores/{cod}	GET	Ficha de indicador
/mrv/indicadores/{cod}/valores	GET	Valores históricos
/mrv/indicadores/{cod}/rango	GET	Rango de fechas disponible
/mrv/derivados/*	GET	Indicadores derivados
/mrv/sync/{cod}	POST	Ejecutar conector
/mrv/admin/*	GET, POST, PUT, DELETE	CRUD administración
/healthz	GET	Health check

4. Flujo de autenticación

4.1. Better Auth

La autenticación se maneja mediante `better-auth`, una biblioteca que gestiona:

1. **Registro:** Creación de usuarios con correo y contraseña (hash bcrypt).
2. **Inicio de sesión:** Validación de credenciales, creación de sesión.
3. **Sesión:** Cookie HTTP-only con duración de 30 días.
4. **Roles:** Verificación de rol en cada solicitud protegida.

4.2. Roles

Rol	Valor en BD	Capacidades
Administrador	<code>admin</code>	Acceso completo, incluyendo administración y pipeline
Analista	<code>analista</code>	Lectura, reporte de acciones, cálculo INGEI
Público	<code>publico</code>	Solo lectura, acceso como invitado

4.3. Protección de rutas

```
// AdminGate component – envuelve layouts administrativos
function AdminGate({ children }: { children: React.ReactNode }) {
  const session = useSession();
  if (session.data?.user?.role !== 'admin') {
    return <AccessDenied />;
  }
  return <>{children}</>;
}
```

5. Motor de derivación MRV

5.1. Descripción

El motor de derivación recorre el grafo acíclico dirigido (DAG) de dependencias entre indicadores y materializa los valores calculados en la tabla `mrv.hecho_indicador_calculado`.

5.2. Flujo de ejecución

1. Obtener todos los indicadores derivados activos
2. Construir el DAG desde `mrv.indicador_parametro`
3. Ordenar topológicamente los indicadores
4. Para cada indicador en orden:
 - a. Obtener la fórmula
 - b. Resolver valores de indicadores fuente
 - c. Evaluar la expresión matemática
 - d. Persistir en `hecho_indicador_calculado`

5.3. Evaluación de fórmulas

Las fórmulas se almacenan como texto (ej. `"HtRt = Q / W"`) y se evalúan mediante un analizador sintáctico que construye un árbol de expresión (AST). Los operadores soportados incluyen suma, resta, multiplicación, división, potenciación y funciones agregadas (promedio, suma, mínimo, máximo).

6. Conectores de datos

6.1. Conectores implementados

Conector	Fuente	Indicadores	Tecnología
XM SinerGox	XM	W, DCdEE, Q, CRest, InPr, CpInG, UndGn, IAGPE	pydataxm
DANE IPI	DANE	Índice de producción industrial	HTTP API
UPME FNCE	UPME	Capacidad FNCE	HTTP API
RENARE	RENARE	Reducción de emisiones	HTTP API

6.2. Configuración de un conector

Cada conector se configura mediante la tabla `mrv.fuente_conexion` con los siguientes parámetros:

```
{
  "tipo": "API_XM",
  "endpoint": "https://services6.xm.com.co/api/v1/...",
  "configuracion": {
    "variable": "W",
    "periodicidad": "DIARIA",
    "timeout": 30,
    "reintentos": 3
  }
}
```

7. Pipeline de procesamiento Geoespacial M&E

7.1. Descripción

El pipeline M&E procesa datos bioclimáticos en bruto (WorldClim, IDEAM) para generar capas ráster de amenaza, vulnerabilidad y riesgo climático para el sector minero-energético.

7.2. Pasos del pipeline

Paso	Descripción	Entrada	Salida
step00	Datos bioclimáticos en bruto	WorldClim, IDEAM	Ráster crudo
step01	Cálculo de indicadores a 1 km	Ráster crudo	19 variables bioclimáticas
step02	Ráster unificado	19 variables	Ráster multibanda
step03	Píxeles por municipio	Ráster unificado + geometrías	Estadísticas por municipio
step04	Poblado de base de datos	Estadísticas	Tablas mye.*
step05	Cálculo de deltas	Valores históricos + futuros	Deltas por variable
step06	Rangos y leyendas	Deltas	Tabla mye_rangos_db
step07	Amenaza a 1 km	Variables seleccionadas	Ráster de amenaza
step08	Base a 90 m	Amenaza 1 km + topografía	Ráster 90 m
step09	Susceptibilidad	Geología, suelos, pendiente	Ráster de susceptibilidad
step10	Amenaza a 90 m	Susceptibilidad + amenaza 1 km	Ráster consolidado
step11	Consolidación de amenaza	Múltiples amenazas	Ráster final de amenaza
step12	Vulnerabilidad	Exposición + sensibilidad + capacidad adaptativa	Ráster de vulnerabilidad

step13	Riesgo	Amenaza + vulnerabilidad	Ráster de riesgo
step14	Estadísticas ráster	Rásteres + geometrías	Tablas de estadísticas
step15	Bandas derivadas a 1 km	Rásteres existentes	Bandas calculadas

7.3. Ejecución

```
# Desde el panel de administración M&E
# O mediante línea de comandos en el servidor:
python gestionar.py --lote <id_lote> --paso <step_name>
```

8. Sincronización automática (cron diario)

8.1. Descripción

El programador de tareas (APScheduler) ejecuta el script `scripts/cron_daily.py` que sincroniza datos desde conectores y materializa indicadores derivados.

8.2. Configuración

Las variables de entorno controlan la ejecución del cron:

```
MRV_CRON_ENABLED=true
MRV_CRON_HOURS=5,9,13,17,21,1 # Horas del día (hora Colombia)
MRV_CRON_DAYS=30               # Días hacia atrás a sincronizar
```

8.3. Mecanismo de bloqueo

El cron utiliza un archivo de bloqueo (`data/cron_daily.lock`) para evitar ejecuciones simultáneas. Si el bloqueo está activo, la ejecución se omite y se registra en el archivo de registro correspondiente.