

# JC2R

## Elemental by JC2R

Documentación Técnica - Sistema de Monitoreo y Cuantificación de  
Carbono

### **Proyecto Open Source**

Sistema de Monitoreo y Cuantificación de Carbono Forestal

**Desarrollado por:** JC2R S.A.S

<https://jc2r.com>

<https://elemental.jc2r.com>

**Versión:** 1.0.1

**Fecha:** November 2025

# Elemental by JC2R

Sistema de monitoreo y cuantificación de biomasa y carbono almacenado para proyectos de conservación en la región Amazónica.

[!Elemental Logo](#)

## ■ Tabla de Contenidos

- [Descripción General](#)
- [Características Principales](#)
- [Arquitectura del Sistema](#)
- [Requisitos del Sistema](#)
- [Instalación](#)
- [Configuración](#)
- [Uso del Sistema](#)
- [API Endpoints](#)
- [Estructura de la Base de Datos](#)
- [Metodología de Cálculo](#)
- [Despliegue](#)
- [Desarrollo](#)
- [Licencia](#)

## ■ Descripción General

**Elemental** es una plataforma web integral open source diseñada para el monitoreo y cuantificación de biomasa y carbono almacenado en bosques tropicales. Desarrollada por **JC2R S.A.S**, la plataforma combina metodologías científicas validadas con tecnología moderna para facilitar inventarios forestales y análisis de carbono.

El sistema permite:

- Gestión de parcelas forestales de 0.1 hectáreas (20m x 50m)
- Registro y medición de especies arbóreas perennes
- Cuantificación de necromasa (biomasa muerta)
- Censo de vegetación herbácea
- Cálculo de biomasa aérea y subterránea
- Estimación de carbono almacenado usando modelos alométricos

- Análisis satelital mediante datos de NASA MODIS
- Visualización geoespacial con Google Maps
- Importación de datos desde archivos GPX y CSV

## ■ Características Principales

### 1. Gestión de Parcelas

- **Establecimiento de parcelas temporales** de 0.1 ha (20m x 50m)
- **Georreferenciación precisa** con coordenadas UTM (datum WGS84)
- Registro de 4 vértices y punto central
- Mapeo interactivo con Google Maps
- Selección de puntos mediante clic en mapa o entrada manual
- Importación de coordenadas desde archivos GPX

### 2. Inventario Forestal

- **Censo de especies arbóreas** ( $DAP \geq 10$  cm)
- Medición de variables dendrométricas:
- DAP (Diámetro a la Altura del Pecho) a 1.3m
- Altura total
- Identificación de especie
- Estado sanitario
- Posición en parcela
- Base de datos de especies con densidad de madera

### 3. Cuantificación de Biomasa

- **Necromasa:**
- Subparcelas de 5m x 5m
- Categorías: gruesa (>10cm) y fina
- Procesamiento en laboratorio (secado a 105°C)
- Relación peso seco/peso fresco
- **Vegetación Herbácea:**
- Cuadrantes de 1m x 1m
- Pesaje de biomasa fresca
- Determinación de biomasa seca
- Extrapolación por hectárea

## 4. Análisis Satelital

- **Integración con NASA MODIS:**
  - Índice de Vegetación de Diferencia Normalizada (NDVI)
  - Índice de Vegetación Mejorado (EVI)
  - Serie temporal de datos satelitales
  - Procesamiento automático de datos NASA AppEEARS
- **Carga de datos:**
  - Solicitud automática a NASA API
  - Importación directa desde archivos CSV
  - Soporte para formatos Statistics y Results
  - Valores pre-normalizados (0-1)

## 5. Cálculos Científicos

- **Modelos alométricos:**
  - Chave et al. para bosques húmedos tropicales
  - Ecuaciones IPCC
  - Modelos IDEAM adaptados
- **Factores de conversión:**
  - Factor de carbono: 0.47 (por defecto)
  - Factores específicos por especie
  - Biomasa subterránea según IPCC
- **Resultados:**
  - Biomasa aérea (t/ha)
  - Biomasa subterránea (t/ha)
  - Carbono total almacenado (t C/ha)
  - Series temporales de indicadores

## 6. Visualización y Reportes

- **Dashboard interactivo:**
  - Métricas consolidadas por parcela
  - Gráficos de serie temporal (Recharts)
  - Indicadores satelitales con tooltips informativos
  - Mapas temáticos de distribución de carbono
- **Análisis comparativo:**
  - Comparación entre parcelas
  - Evolución temporal de indicadores
  - Distribución por componentes de biomasa

## 7. Puntos de Referencia

- Gestión de waypoints georeferenciados
- Organización por zonas
- Importación masiva desde archivos GPX
- Selección interactiva en mapa
- Registro de fuente de datos

## ■ ■ Arquitectura del Sistema

### Stack Tecnológico

#### #### Backend

- **Framework:** FastAPI 0.109+
- **Base de Datos:** SQLite con SQLAlchemy 2.0+
- **Servidor:** Uvicorn con soporte ASGI
- **Procesamiento de Datos:**
  - Pandas 2.1+ (análisis de datos)
  - NumPy 1.26+ (cálculos numéricos)
  - GeoPandas 0.14+ (datos geoespaciales)
  - Scipy 1.11+ (cálculos científicos)

#### #### Frontend

- **Framework:** React 19.2
- **Build Tool:** Vite 7.2
- **Router:** React Router DOM 7.9
- **UI Components:**
  - Radix UI (componentes accesibles)
  - shadcn/ui (sistema de diseño)
  - Lucide React (iconos)
- **Estilos:** Tailwind CSS 3.4 + tailwindcss-animate
- **Mapas:** Google Maps API (@googlemaps/js-api-loader)
- **Gráficos:** Recharts 3.4
- **Estado:** React Hooks + Context API
- **Notificaciones:** Sonner (toast notifications)
- **Temas:** next-themes (dark/light mode)

## #### Infraestructura

- **Containerización:** Docker + Docker Compose
- **CI/CD:** GitHub Actions
- **Servidor:** VPS (147.93.10.133)
- **Dominio:** <https://elemental.jc2r.com>

## Estructura de Directorios

```
IAP/ ■■■■ frontend/ # Aplicación React ■■■■ src/ ■ ■■■■ components/ #
Componentes React ■ ■ ■■■■ Layout.jsx # Layout principal con sidebar ■ ■ ■
■■■■ Dashboard.jsx # Panel principal ■ ■ ■■■■ Mapa.jsx # Vista de mapa con
Google Maps ■ ■ ■■■■ Parcelas.jsx # Gestión de parcelas ■ ■ ■■■■
AnálisisSatelital.jsx # Análisis satelital ■ ■ ■■■■ HistorialSatelital.jsx
# Series temporales satelitales ■ ■ ■■■■ Especies.jsx # Catálogo de
especies ■ ■ ■■■■ FormularioPuntoReferencia.jsx # Formulario de puntos ■ ■
■■■■ MapSelectorPunto.jsx # Selector de puntos en mapa ■ ■ ■■■■ ui/ #
Componentes shadcn/ui ■ ■ ■■■■ services/ ■ ■ ■■■■ api.js # Cliente API ■ ■ ■
■■■■ lib/ ■ ■ ■■■■ utils.js # Utilidades (cn, etc.) ■ ■ ■■■■ App.jsx #
Componente raíz ■ ■ ■■■■ main.jsx # Entry point ■ ■■■■ public/ ■ ■ ■■■■
JC2R_LOGO.svg # Logo blanco (dark mode) ■ ■ ■■■■ JC2R_LOGO_BLACK.svg # Logo
negro (light mode) ■ ■■■■ package.json ■ ■■■■ vite.config.js ■ ■■■■
tailwind.config.js ■ ■■■■ src/ # Backend Python ■ ■■■■ api/ ■ ■ ■■■■ main.py
# Entry point FastAPI ■ ■ ■■■■ routes/ ■ ■ ■■■■ parcelas.py # CRUD parcelas
■ ■ ■■■■ especies.py # CRUD especies ■ ■ ■■■■ arboles.py # CRUD árboles
■ ■ ■■■■ necromasa.py # CRUD necromasa ■ ■ ■■■■ herbaceas.py # CRUD
herbáceas ■ ■ ■■■■ calculos.py # Cálculos de biomasa/carbono ■ ■ ■■■■
calculos_satelitales.py # Análisis satelital ■ ■ ■■■■ puntos_referencia.py
# Puntos de referencia ■ ■ ■■■■ schemas/ ■ ■ ■■■■ parcela_schema.py ■ ■ ■■■■
calculo_satelital_schema.py ■ ■■■■ models/ # Modelos SQLAlchemy ■ ■■■■
services/ # Lógica de negocio ■ ■■■■ utils/ # Utilidades ■ ■■■■ scripts/ #
Scripts de utilidad ■ ■■■■ import_gpx.py # Importador de archivos GPX ■ ■■■■
data/ ■ ■■■■ raw/ # Datos crudos (GPX, CSV) ■ ■■■■ processed/ # Datos
procesados ■ ■■■■ docs/ # Documentación ■ ■■■■ config/ # Configuraciones ■ ■■■■
tests/ # Tests unitarios ■ ■■■■ iap_database.db # Base de datos SQLite ■ ■■■■
requirements.txt # Dependencias Python ■ ■■■■ docker-compose.yml # Configuración
Docker ■ ■■■■ Dockerfile # Imagen Docker backend ■ ■■■■ .env # Variables de
entorno ■ ■■■■ init_db.py # Script de inicialización DB ■ ■■■■ start.sh # Script
de inicio ■ ■■■■ README.md # Este archivo
```

## ■ Requisitos del Sistema

### Para Desarrollo Local

- **Python:** 3.9 o superior
- **Node.js:** 18.x o superior
- **npm:** 9.x o superior
- **SQLite:** 3.x
- **Git:** 2.x

### Para Producción

- **Docker:** 20.x o superior

- **Docker Compose:** 2.x o superior
- Servidor Linux (Ubuntu 20.04+ recomendado)
- 2GB RAM mínimo
- 10GB espacio en disco

## APIs Externas

- **Google Maps API Key** (para mapas interactivos)
- **NASA Earthdata Account** (para análisis satelital MODIS)

## ■ Instalación

### Instalación Local

#### #### 1. Clonar el Repositorio

```
git clone <repository-url> cd IAP
```

#### #### 2. Configurar Backend

```
# Crear entorno virtual python3 -m venv venv # Activar entorno virtual # En
macOS/Linux: source venv/bin/activate # En Windows: venv\Scripts\activate #
Instalar dependencias pip install -r requirements.txt # Inicializar base de
datos python init_db.py
```

#### #### 3. Configurar Frontend

```
cd frontend npm install
```

#### #### 4. Configurar Variables de Entorno

Crear archivo **.env** en la raíz del proyecto:

```
# Base de Datos DATABASE_URL=sqlite:///./iap_database.db # Google Maps
VITE_GOOGLE_MAPS_API_KEY=tu_google_maps_api_key # NASA Earthdata
NASA_USERNAME=tu_nasa_username NASA_PASSWORD=tu_nasa_password # Backend
BACKEND_URL=http://localhost:8000 # Entorno ENVIRONMENT=development
```

#### #### 5. Ejecutar el Sistema

##### Terminal 1 - Backend:

```
source venv/bin/activate python -m uvicorn src.api.main:app --reload --host
0.0.0.0 --port 8000
```

##### Terminal 2 - Frontend:

```
cd frontend npm run dev
```

El sistema estará disponible en:

- **Frontend:** <http://localhost:5173>
- **Backend API:** <http://localhost:8000>
- **API Docs:** <http://localhost:8000/docs>

## Instalación con Docker

```
# Construir y levantar contenedores docker compose up -d # Ver logs docker  
compose logs -f # Detener contenedores docker compose down
```

## ■■ Configuración

### Google Maps API

1. Crear proyecto en [Google Cloud Console](#)
2. Habilitar APIs:
  - Maps JavaScript API
  - Geocoding API
  - Places API
3. Crear credenciales (API Key)
4. Agregar a `.env`: `VITEGOOGLEMAPSAPIKEY=tuapikey`

### NASA Earthdata

1. Registrarse en [NASA Earthdata](#)
2. Aprobar aplicaciones:
  - NASA AppEEARS
3. Agregar credenciales a `.env`:

```
NASA_USERNAME=tu_usuario NASA_PASSWORD=tu_contraseña
```

### Base de Datos

#### Inicialización

```
python init_db.py
```

Crea las siguientes tablas:

- `parcelas` - Parcelas forestales
- `especies` - Catálogo de especies
- `arboles` - Inventario de árboles



- [necromasa](#) - Mediciones de necromasa
- [herbaceas](#) - Mediciones de herbáceas
- [calculos\\_satelitales](#) - Análisis satelitales
- [puntos\\_referencia](#) - Puntos de referencia georeferenciados

#### #### Importar Datos GPX

```
# Colocar archivos GPX en data/raw/ python scripts/import_gpx.py
```

#### #### Backup Manual

```
# Crear backup cp iap_database.db iap_database_backup_$(date +%Y%m%d).db #
Restaurar backup cp iap_database_backup_20250117.db iap_database.db
```

## ■ Uso del Sistema

### 1. Dashboard

Vista principal con métricas consolidadas:

- Total de parcelas registradas
- Total de árboles inventariados
- Biomasa total estimada
- Carbono total almacenado
- Gráficos de distribución

### 2. Mapa

Visualización geoespacial interactiva:

- Ubicación de parcelas
- Puntos de referencia por zonas
- Filtros por zona/tipo
- Creación de nuevas zonas y puntos
- Selección de coordenadas mediante clic

### 3. Gestión de Parcelas

**Crear Parcela:**

1. Ir a "Parcelas" → "Nueva Parcela"
2. Completar datos:

- Código único
  - Nombre descriptivo
  - Zona priorizada
  - Fecha de establecimiento
3. Georreferenciación:
- **Opción A:** Ingresar coordenadas manualmente
  - **Opción B:** Seleccionar en mapa (clic o arrastrar marcador)
4. Registrar 4 vértices y punto central
5. Información adicional: pendiente, cobertura, accesibilidad

#### **Gestionar Datos de Parcela:**

- **Árboles:** Inventario forestal con mediciones DAP y altura
- **Necromasa:** Registro de biomasa muerta (peso fresco/seco)
- **Herbáceas:** Censo de vegetación en cuadrantes
- **Cálculos:** Ver estimaciones de biomasa y carbono
- **Análisis Satelital:** Series temporales NDVI/EVI

## **4. Análisis Satelital**

#### **Crear Análisis desde NASA:**

1. Seleccionar parcela
2. Ir a tab "Satelital"
3. Clic en "Nuevo Análisis NASA"
4. Configurar:
  - Rango de fechas
  - Productos satelitales (MOD13Q1, MOD13Q1)
  - Índices (NDVI, EVI)
5. Enviar solicitud a NASA AppEEARS
6. Esperar procesamiento (5-15 minutos)
7. Descargar resultados automáticamente

#### **Cargar Análisis desde CSV:**

1. Seleccionar parcela
2. Ir a tab "Satelital"
3. Clic en "Cargar CSV"
4. Seleccionar archivo CSV (formato NASA AppEEARS)
5. El sistema detecta automáticamente el formato:
  - **Statistics:** File Name, Date, Mean
  - **Results:** Fecha, NDVI, EVI (columnas directas)

## 6. Visualizar resultados inmediatamente

### Formatos CSV Soportados:

#### Formato Statistics:

```
File Name,Date,Mean MOD13Q1.061_250m_aid0001.csv,2020-01-01,0.8743  
MOD13Q1.061_250m_aid0001.csv,2020-01-17,0.8615
```

#### Formato Results:

```
Fecha,NDVI,EVI,Biomasa,Carbono 2020-01-01,0.8743,0.6592,245.3,115.3  
2020-01-17,0.8615,0.6421,238.1,111.9
```

### Interpretación de Indicadores:

#### NDVI (Normalized Difference Vegetation Index):

- **0.8 - 1.0:** Vegetación muy densa (bosques maduros)
- **0.6 - 0.8:** Vegetación densa (bosques secundarios)
- **0.4 - 0.6:** Vegetación moderada (cultivos, pastizales)
- **0.2 - 0.4:** Vegetación dispersa
- **< 0.2:** Suelo desnudo o agua

#### EVI (Enhanced Vegetation Index):

- Optimizado para áreas con alta biomasa
- Reduce efectos de saturación en bosques densos
- Rangos similares a NDVI pero más sensible

#### Biomasa Aérea Estimada:

- Calculada mediante modelos alométricos
- Unidades: toneladas por hectárea (t/ha)
- Rango típico bosques amazónicos: 150-400 t/ha

#### Carbono Almacenado:

- Factor de conversión: 0.47
- Unidades: toneladas de carbono por hectárea (t C/ha)
- Relevante para proyectos REDD+

## 5. Especies

### Catálogo de Especies:

- Nombre común y científico
- Familia botánica
- Densidad de madera (g/cm<sup>3</sup>)

- Factor de carbono específico
- Distribución geográfica
- Usos tradicionales

#### Agregar Nueva Especie:

1. Ir a "Especies" → "Nueva Especie"
2. Completar información
3. Especificar densidad de madera (crucial para cálculos)
4. Guardar

## 6. Puntos de Referencia

#### Crear Punto Manualmente:

1. Ir a "Mapa" → "Crear Punto de Referencia"
2. Seleccionar zona
3. Ingresar nombre y descripción
4. Coordenadas:
  - **Manual:** Escribir latitud/longitud
  - **Mapa:** Clic en ubicación o arrastrar marcador
5. Guardar

#### Importar desde GPX:

```
# Colocar archivos .gpx en data/raw/ python scripts/import_gpx.py
```

El script:

- Detecta todos los archivos GPX en `data/raw/`
- Extrae waypoints (lat, lon, nombre, descripción)
- Verifica duplicados (tolerancia 0.00001°)
- Importa a base de datos
- Muestra resumen de importación

## ■ API Endpoints

### Parcelas

```
GET /api/v1/parcelas # Listar todas las parcelas POST /api/v1/parcelas # Crear
parcela GET /api/v1/parcelas/{id} # Obtener parcela por ID PUT
/api/v1/parcelas/{id} # Actualizar parcela DELETE /api/v1/parcelas/{id} #
Eliminar parcela GET /api/v1/parcelas/{id}/calculos # Obtener cálculos de
parcela
```

## Especies

```
GET /api/v1/especies # Listar todas las especies POST /api/v1/especies # Crear especie GET /api/v1/especies/{id} # Obtener especie por ID PUT /api/v1/especies/{id} # Actualizar especie DELETE /api/v1/especies/{id} # Eliminar especie
```

## Árboles

```
GET /api/v1/arboles/parcela/{id} # Árboles de una parcela POST /api/v1/arboles # Registrar árbol GET /api/v1/arboles/{id} # Obtener árbol por ID PUT /api/v1/arboles/{id} # Actualizar árbol DELETE /api/v1/arboles/{id} # Eliminar árbol
```

## Necromasa

```
GET /api/v1/necromasa/parcela/{id} # Necromasa de una parcela POST /api/v1/necromasa # Registrar necromasa GET /api/v1/necromasa/{id} # Obtener registro por ID PUT /api/v1/necromasa/{id} # Actualizar registro DELETE /api/v1/necromasa/{id} # Eliminar registro
```

## Herbáceas

```
GET /api/v1/herbaceas/parcela/{id} # Herbáceas de una parcela POST /api/v1/herbaceas # Registrar herbácea GET /api/v1/herbaceas/{id} # Obtener registro por ID PUT /api/v1/herbaceas/{id} # Actualizar registro DELETE /api/v1/herbaceas/{id} # Eliminar registro
```

## Cálculos

```
GET /api/v1/calculos/parcela/{id} # Cálculos de parcela POST /api/v1/calculos/parcela/{id}/calcular-biomasa # Calcular biomasa POST /api/v1/calculos/parcela/{id}/calcular-carbono # Calcular carbono GET /api/v1/calculos/parcela/{id}/resumen # Resumen consolidado
```

## Análisis Satelital

```
GET /api/v1/calculos-satelitales/parcela/{id} # Listar análisis POST /api/v1/calculos-satelitales/parcela/{id} # Crear análisis NASA POST /api/v1/calculos-satelitales/parcela/{id}/desde-csv # Crear desde CSV GET /api/v1/calculos-satelitales/{id} # Obtener análisis DELETE /api/v1/calculos-satelitales/{id} # Eliminar análisis GET /api/v1/calculos-satelitales/{id}/serie-temporal # Serie temporal POST /api/v1/calculos-satelitales/{id}/procesar-csv # Procesar CSV NASA
```

## Puntos de Referencia

```
GET /api/v1/puntos-referencia # Listar todos los puntos POST /api/v1/puntos-referencia # Crear punto GET /api/v1/puntos-referencia/zona/{zona} # Puntos por zona GET /api/v1/puntos-referencia/{id} # Obtener punto por ID PUT /api/v1/puntos-referencia/{id} # Actualizar punto DELETE /api/v1/puntos-referencia/{id} # Eliminar punto
```

## Documentación Interactiva

- **Swagger UI:** <http://localhost:8000/docs>
- **ReDoc:** <http://localhost:8000/redoc>

## ■ Estructura de la Base de Datos

### Tabla: `parcelas`

```
id INTEGER PRIMARY KEY codigo VARCHAR(50) UNIQUE NOT NULL nombre VARCHAR(200)
zona_priorizada VARCHAR(100) fecha_establecimiento DATE latitud FLOAT
longitud FLOAT altitud FLOAT utm_x FLOAT utm_y FLOAT utm_zone VARCHAR(10)
vertice1_lat FLOAT vertice1_lon FLOAT vertice2_lat FLOAT vertice2_lon FLOAT
vertice3_lat FLOAT vertice3_lon FLOAT vertice4_lat FLOAT vertice4_lon FLOAT
pendiente FLOAT tipo_cobertura VARCHAR(100) accesibilidad VARCHAR(50)
observaciones TEXT croquis_url VARCHAR(500) responsable VARCHAR(100) estado
VARCHAR(50) created_at DATETIME DEFAULT CURRENT_TIMESTAMP updated_at DATETIME
```

### Tabla: `especies`

```
id INTEGER PRIMARY KEY nombre_comun VARCHAR(200) NOT NULL nombre_cientifico
VARCHAR(200) familia VARCHAR(100) densidad_madera FLOAT factor_carbono FLOAT
descripcion TEXT usos TEXT distribucion TEXT created_at DATETIME DEFAULT
CURRENT_TIMESTAMP updated_at DATETIME
```

### Tabla: `arboles`

```
id INTEGER PRIMARY KEY parcela_id INTEGER NOT NULL REFERENCES parcelas(id) ON
DELETE CASCADE especie_id INTEGER REFERENCES especies(id) numero_arbol
INTEGER NOT NULL codigo VARCHAR(50) dap FLOAT NOT NULL altura FLOAT posicion_x
FLOAT posicion_y FLOAT forma_fuste VARCHAR(50) estado_sanitario VARCHAR(100)
fecha_medicion DATE observaciones TEXT foto_url VARCHAR(500) created_at
DATETIME DEFAULT CURRENT_TIMESTAMP updated_at DATETIME
```

### Tabla: `necromasa`

```
id INTEGER PRIMARY KEY parcela_id INTEGER NOT NULL REFERENCES parcelas(id) ON
DELETE CASCADE subparcela_numero INTEGER subparcela_x FLOAT subparcela_y
FLOAT tipo VARCHAR(50) peso_fresco FLOAT peso_seco FLOAT diametro FLOAT
longitud FLOAT fecha_medicion DATE fecha_secado DATE observaciones TEXT
created_at DATETIME DEFAULT CURRENT_TIMESTAMP updated_at DATETIME
```

### Tabla: `herbaceas`

```
id INTEGER PRIMARY KEY parcela_id INTEGER NOT NULL REFERENCES parcelas(id) ON
DELETE CASCADE cuadrante_numero INTEGER cuadrante_x FLOAT cuadrante_y FLOAT
peso_fresco FLOAT peso_seco FLOAT especies_presentes TEXT cobertura_visual
FLOAT fecha_medicion DATE fecha_secado DATE observaciones TEXT created_at
DATETIME DEFAULT CURRENT_TIMESTAMP updated_at DATETIME
```

### Tabla: `calculos\_satelitales`

```
id INTEGER PRIMARY KEY parcela_id INTEGER NOT NULL REFERENCES parcelas(id) ON
DELETE CASCADE fecha_inicio DATE NOT NULL fecha_fin DATE NOT NULL fuente_datos
VARCHAR(50) producto VARCHAR(50) ndvi_promedio FLOAT evi_promedio FLOAT
biomasa_aerea_estimada FLOAT carbono_estimado FLOAT modelo_estimacion
VARCHAR(100) factor_carbono FLOAT num_imagenes_usadas INTEGER calidad_datos
FLOAT estado_procesamiento VARCHAR(50) serie_temporal JSON created_at
DATETIME DEFAULT CURRENT_TIMESTAMP updated_at DATETIME
```

### Tabla: `puntos\_referencia`

```
id INTEGER PRIMARY KEY zona VARCHAR(100) NOT NULL nombre VARCHAR(200)
descripcion TEXT latitud FLOAT NOT NULL longitud FLOAT NOT NULL fuente
VARCHAR(200) created_at DATETIME DEFAULT CURRENT_TIMESTAMP updated_at
DATETIME
```

## ■ Metodología de Cálculo

### 1. Biomasa Aérea (Arbórea)

Ecuación Alométrica de Chave et al. (2014):

$$AGB = 0.0673 \times (\rho \times D^2 \times H)^{0.976}$$

Donde:

- **AGB** = Biomasa aérea (kg)
- **$\rho$**  = Densidad de la madera (g/cm<sup>3</sup>)
- **D** = DAP (cm)
- **H** = Altura total (m)

Proceso:

1. Medir DAP a 1.3m del suelo
2. Medir altura total con hipsómetro
3. Identificar especie para obtener densidad de madera
4. Aplicar ecuación alométrica
5. Sumar biomasa de todos los individuos en la parcela
6. Extrapolar a hectárea (parcela = 0.1 ha)

### 2. Biomasa Subterránea (Raíces)

Factor de Conversión IPCC:

$$BGB = AGB \times R$$

Donde:

- **BGB** = Biomasa subterránea (t/ha)
- **AGB** = Biomasa aérea (t/ha)
- **R** = Factor de conversión (0.24 para bosques tropicales)

### 3. Necromasa

Cálculo por Subparcela (5m x 5m):

$$\text{Necromasa (t/ha)} = (\text{Peso\_seco} / \text{Área\_subparcela}) \times 10000$$

Donde:

- $\text{Peso\_seco}$  = Peso seco en kg (secado a 105°C)
- $\text{Área\_subparcela}$  = 25 m<sup>2</sup>
- 10000 = Factor de conversión a hectárea

## 4. Biomasa Herbácea

**Cálculo por Cuadrante (1m x 1m):**

$$\text{Herbácea (t/ha)} = (\text{Peso\_seco} / \text{Área\_cuadrante}) \times 10000$$

Donde:

- $\text{Peso\_seco}$  = Peso seco en kg
- $\text{Área\_cuadrante}$  = 1 m<sup>2</sup>

## 5. Biomasa Total

$$\text{Biomasa\_Total} = \text{Biomasa\_Aérea} + \text{Biomasa\_Subterránea} + \text{Necromasa} + \text{Herbácea}$$

## 6. Carbono Almacenado

**Factor de Carbono:**

$$\text{Carbono (t C/ha)} = \text{Biomasa\_Total (t/ha)} \times 0.47$$

El factor 0.47 representa que aproximadamente el 47% de la biomasa es carbono.

## 7. Estimación desde Índices Satelitales

**Modelo NDVI-Biomasa:**

$$\text{Biomasa} = a \times \text{NDVI}^2 + b \times \text{NDVI} + c$$

Donde los coeficientes a, b, c son calibrados para bosques amazónicos:

- $a$  = 500
- $b$  = -150
- $c$  = 50

**Modelo EVI-Biomasa:**

$$\text{Biomasa} = d \times \text{EVI}^2 + e \times \text{EVI} + f$$

Coeficientes calibrados:

- $d$  = 600
- $e$  = -200
- $f$  = 75



## ■ Despliegue

### Despliegue Automático (CI/CD)

El proyecto usa GitHub Actions para CI/CD automático:

**Workflow** ([.github/workflows/deploy.yml](#)):

1. Trigger: Push a `main`
2. Conecta a servidor vía SSH
3. Pull del código más reciente
4. Rebuild de contenedores Docker
5. Restart de servicios

■ ■ **Importante:** El CI/CD **NO** actualiza la base de datos automáticamente. La base de datos debe migrarse manualmente.

### Actualización Manual de Base de Datos

```
# 1. Crear backup de DB producción ssh root@147.93.10.133 "cd /srv/Elemental
&& cp iap_database.db iap_database_backup_$(date +%Y%m%d).db" # 2. Copiar DB
local a producción scp iap_database.db root@147.93.10.133:/srv/Elemental/ #
3. Reiniciar backend ssh root@147.93.10.133 "cd /srv/Elemental && docker
compose restart backend"
```

### Despliegue Manual Completo

```
# 1. Conectar al servidor ssh root@147.93.10.133 # 2. Ir al directorio del
proyecto cd /srv/Elemental # 3. Pull del código git pull origin main # 4.
Rebuild y restart docker compose down docker compose up -d --build # 5. Ver
logs docker compose logs -f
```

### Variables de Entorno en Producción

Configurar `.env` en el servidor:

```
# Base de Datos DATABASE_URL=sqlite:///./iap_database.db # Google Maps
VITE_GOOGLE_MAPS_API_KEY=production_google_maps_key # NASA Earthdata
NASA_USERNAME=production_nasa_username NASA_PASSWORD=production_nasa_password
# Backend BACKEND_URL=https://elemental.jc2r.com # Entorno
ENVIRONMENT=production
```

### Verificación de Despliegue

```
# Verificar contenedores docker compose ps # Verificar logs backend docker
compose logs backend # Verificar logs frontend docker compose logs frontend #
Verificar base de datos docker compose exec backend python -c "import sqlite3;
conn = sqlite3.connect('iap_database.db'); print('DB OK')"
```

### Monitoreo

### Logs en tiempo real:

```
docker compose logs -f
```

### Logs del backend:

```
docker compose logs -f backend
```

### Logs del frontend:

```
docker compose logs -f frontend
```

## Backup Automático (Recomendado)

Crear cron job para backups diarios:

```
# Editar crontab crontab -e # Agregar línea (backup diario a las 3 AM) 0 3 * *  
* cd /srv/Elemental && cp iap_database.db backups/iap_database_$(date  
+\"%Y\"%m\"%d\").db
```

## ■ ■ Desarrollo

### Estructura de Desarrollo

#### Branch Strategy:

- **main** - Producción (auto-deploy)
- **develop** - Desarrollo
- **feature/\*** - Features nuevas
- **fix/\*** - Bug fixes

## Agregar Nueva Funcionalidad

### Backend (FastAPI):

#### 1. Crear modelo en `src/models/`:

```
# src/models/nueva_entidad.py from sqlalchemy import Column, Integer, String,  
Float from src.database import Base class NuevaEntidad(Base): __tablename__ =  
"nueva_entidad" id = Column(Integer, primary_key=True) nombre =  
Column(String(200)) valor = Column(Float)
```

#### 2. Crear schema en `src/api/schemas/`:

```
# src/api/schemas/nueva_entidad_schema.py from pydantic import BaseModel  
class NuevaEntidadCreate(BaseModel): nombre: str valor: float class  
NuevaEntidadResponse(NuevaEntidadCreate): id: int class Config:  
from_attributes = True
```

### 3. Crear rutas en `src/api/routes/`

```
# src/api/routes/nueva_entidad.py from fastapi import APIRouter, Depends from sqlalchemy.orm import Session from src.database import get_db router = APIRouter(prefix="/nueva-entidad", tags=["nueva-entidad"]) @router.get("/") def listar_entidades(db: Session = Depends(get_db)): return db.query(NuevaEntidad).all()
```

### 4. Registrar router en `src/api/main.py`

```
from src.api.routes import nueva_entidad app.include_router(nueva_entidad.router, prefix="/api/v1")
```

## Frontend (React):

### 1. Crear servicio API en `frontend/src/services/api.js`

```
export async function obtenerEntidades() { const response = await fetch(`${API_BASE_URL}/nueva-entidad`) if (!response.ok) throw new Error('Error al obtener entidades') return await response.json() }
```

### 2. Crear componente en `frontend/src/components/`

```
// frontend/src/components/NuevaEntidad.jsx import { useState, useEffect } from 'react' import { obtenerEntidades } from '@services/api' export default function NuevaEntidad() { const [entidades, setEntidades] = useState([]) useEffect(() => { cargarEntidades(), [] } async function cargarEntidades() { const data = await obtenerEntidades() setEntidades(data) } return ( <div> /* UI aquí */ </div> ) }
```

### 3. Agregar ruta en `frontend/src/App.jsx`

```
import NuevaEntidad from './components/NuevaEntidad' <Route path="/nueva-entidad" element={ <NuevaEntidad /> } />
```

### 4. Agregar navegación en `frontend/src/components/Layout.jsx`

```
const navigation = [ // ... { name: 'Nueva Entidad', href: '/nueva-entidad', icon: IconComponent }, ]
```

## Testing

### Backend:

```
# Ejecutar tests pytest # Con cobertura pytest --cov=src --cov-report=html # Test específico pytest tests/test_parcelas.py
```

### Frontend:

```
cd frontend npm run test
```

## Code Quality

### Python (Black + Flake8):

```
# Formatear código black src/ # Linting flake8 src/ # Type checking mypy src/
```

### JavaScript (ESLint):

```
cd frontend npm run lint
```

## Debugging

### Backend con debugger:

```
import pdb; pdb.set_trace()
```

### Frontend con React DevTools:

- Instalar [React DevTools](#)
- Inspeccionar componentes en navegador

## ■ Licencia

### MIT License

Copyright (c) 2025 JC2R S.A.S

Este es un proyecto open source. Consulta el archivo LICENSE para más detalles.

## ■ Contacto y Soporte

**Desarrollado por:** JC2R S.A.S

**Sitio web corporativo:** <https://jc2r.com>

**Demo del sistema:** <https://elemental.jc2r.com>

**Email:** [info@jc2r.com](mailto:info@jc2r.com)

## ■ Agradecimientos

- Comunidades locales de la Amazonía (Patruyeros, Santa Sofía)
- NASA Earthdata por acceso a datos MODIS
- Chave et al. por modelos alométricos
- IPCC por factores de conversión
- Comunidad open source

## ■ Referencias

1. Chave, J., et al. (2014). "Improved allometric models to estimate the aboveground biomass of tropical trees." *Global Change Biology*, 20(10), 3177-3190.
2. IPCC (2006). "2006 IPCC Guidelines for National Greenhouse Gas Inventories." Volume 4: Agriculture, Forestry and Other Land Use.
3. IDEAM (2011). "Protocolo para la estimación nacional y subnacional de biomasa - carbono en Colombia."
4. NASA AppEEARS Documentation: <https://appeears.earthdatacloud.nasa.gov/>
5. MODIS Vegetation Index Products (MOD13): <https://modis.gsfc.nasa.gov/data/dataproduct/mod13.php>

**Versión:** 1.0.1

**Última actualización:** Noviembre 2025