**Arquitectura del Sistema**

**1. Arquitectura General**

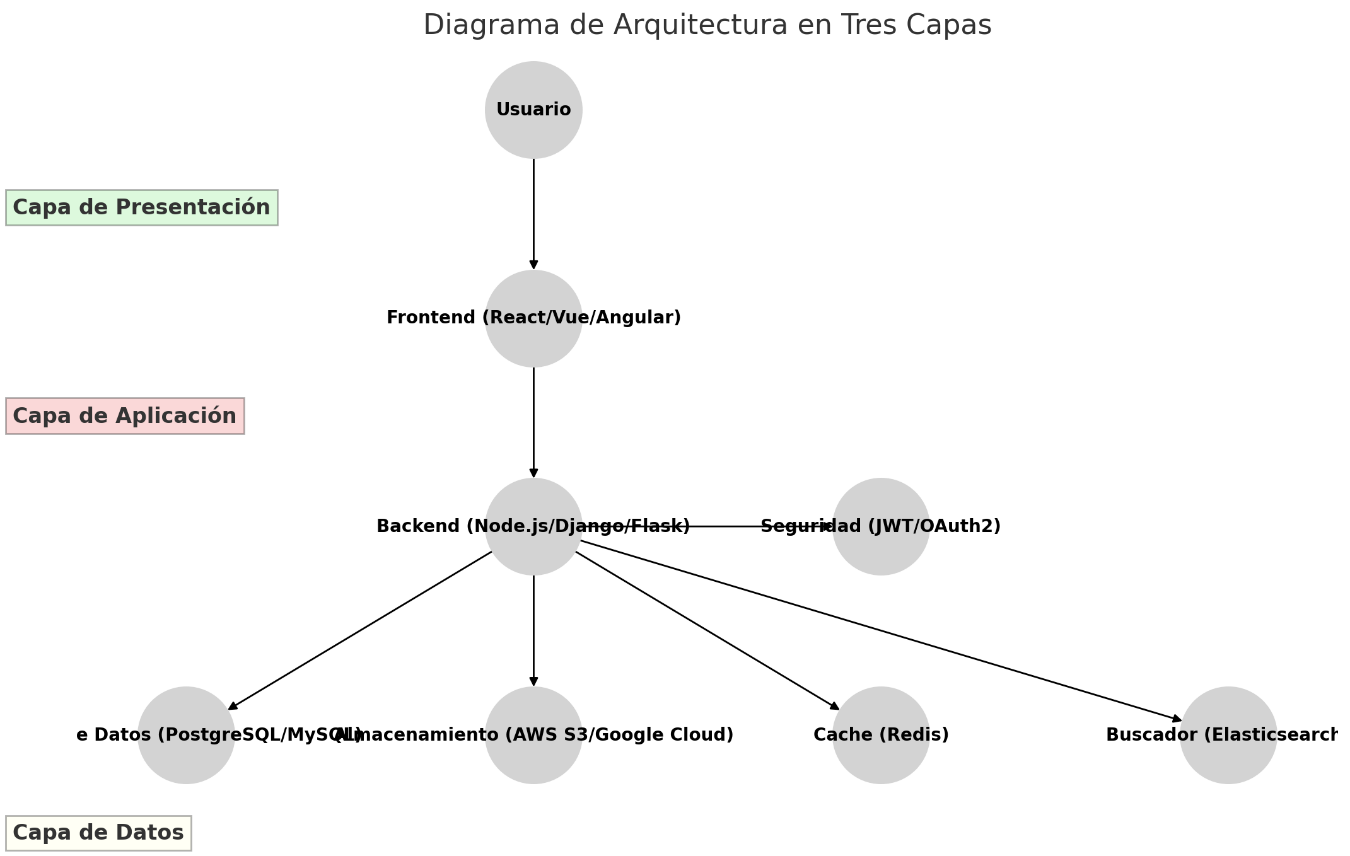
Se recomienda una arquitectura basada en **microservicios** o un **modelo de tres capas** para garantizar escalabilidad, seguridad y mantenibilidad.

**1.1 Capas de la Arquitectura**

1. **Capa de Presentación (Frontend)**
   * Framework recomendado: **React.js, Vue.js o Angular**
   * Funcionalidades clave:
     + Interfaz responsiva y amigable
     + Vista previa de datasets
     + Panel de administración
     + Gestión de usuarios y autenticación
     + Sistema de comentarios y calificación
2. **Capa de Aplicación (Backend)**
   * Lenguaje recomendado: **Node.js (Express), Python (Django o Flask)**
   * Funcionalidades clave:
     + API REST para la comunicación con el frontend
     + Autenticación y autorización (JWT o OAuth2)
     + Gestión de datasets (carga, visualización, descarga)
     + Control de comentarios y moderación
     + Búsqueda y filtrado de datasets
     + Seguridad y validación de datos
3. **Capa de Datos (Base de Datos y Almacenamiento)**
   * **Base de datos relacional**: PostgreSQL o MySQL (para datos estructurados como usuarios y datasets)
   * **Base de datos NoSQL**: MongoDB (para almacenamiento de comentarios y registros de actividad)
   * **Almacenamiento de archivos**: AWS S3, Google Cloud Storage o almacenamiento local para alojar datasets

**2. Infraestructura y Tecnologías**

| **Componente** | **Tecnología Recomendada** |
| --- | --- |
| **Frontend** | React.js / Vue.js / Angular |
| **Backend (API)** | Node.js con Express / Django / Flask |
| **Autenticación** | OAuth2 |
| **Base de Datos** | PostgreSQL / MySQL (datos estructurados), MongoDB (comentarios y logs) |
| **Almacenamiento de Archivos** | AWS S3 / Google Cloud Storage / Servidor local con redundancia |
| **Cache** | Redis (para mejorar la velocidad de búsqueda y acceso a datasets) |
| **Buscador** | Elasticsearch o PostgreSQL con índices full-text |
| **Servidor y Hosting** | AWS EC2, Google Cloud Compute Engine, o DigitalOcean |
| **Seguridad** | HTTPS, Encriptación de datos en reposo, Protección contra inyecciones SQL y XSS |



**Modelo de Datos**

Se define un esquema de base de datos relacional optimizado para almacenar los **datasets**, la información de los **usuarios** y los **comentarios**.

Tabla

El contenido generado por IA puede ser incorrecto.

**1. Entidades y Relaciones**

**1.1 Tabla: usuarios**

Almacena la información de los usuarios registrados en la plataforma.

CREATE TABLE usuarios (

id SERIAL PRIMARY KEY,

nombre VARCHAR(100) NOT NULL,

correo VARCHAR(150) UNIQUE NOT NULL,

contrasena\_hash TEXT NOT NULL,

rol ENUM('admin', 'usuario') DEFAULT 'usuario',

fecha\_registro TIMESTAMP DEFAULT CURRENT\_TIMESTAMP

);

**1.2 Tabla: datasets**

Almacena la información de cada dataset disponible en la plataforma.

CREATE TABLE datasets (

id SERIAL PRIMARY KEY,

titulo VARCHAR(255) NOT NULL,

descripcion TEXT NOT NULL,

categoria VARCHAR(100),

etiquetas TEXT,

usuario\_id INT REFERENCES usuarios(id) ON DELETE CASCADE,

url\_archivo TEXT NOT NULL, -- Ruta en almacenamiento

formato ENUM('CSV', 'JSON', 'Excel'),

tamano\_archivo BIGINT, -- Tamaño en bytes

fecha\_subida TIMESTAMP DEFAULT CURRENT\_TIMESTAMP

);

**1.3 Tabla: especificaciones\_dataset**

Almacena los detalles de las columnas que componen un dataset.

CREATE TABLE especificaciones\_dataset (

id SERIAL PRIMARY KEY,

dataset\_id INT REFERENCES datasets(id) ON DELETE CASCADE,

nombre\_columna VARCHAR(150) NOT NULL,

tipo\_dato VARCHAR(50) NOT NULL, -- Ej: INT, FLOAT, TEXT, BOOLEAN

descripcion TEXT

);

**1.4 Tabla: comentarios**

Guarda los comentarios y discusiones sobre un dataset.

CREATE TABLE comentarios (

id SERIAL PRIMARY KEY,

dataset\_id INT REFERENCES datasets(id) ON DELETE CASCADE,

usuario\_id INT REFERENCES usuarios(id) ON DELETE CASCADE,

contenido TEXT NOT NULL,

fecha\_creacion TIMESTAMP DEFAULT CURRENT\_TIMESTAMP

);

**1.5 Tabla: descargas**

Registra las descargas de cada dataset por los usuarios.

CREATE TABLE descargas (

id SERIAL PRIMARY KEY,

dataset\_id INT REFERENCES datasets(id) ON DELETE CASCADE,

usuario\_id INT REFERENCES usuarios(id) ON DELETE CASCADE,

fecha\_descarga TIMESTAMP DEFAULT CURRENT\_TIMESTAMP

);