Caja de Herramientas JAL — Especificación Funcional & Técnica (v0.3)

> Inspiración: **Normograma de Medellín** (https://www.medellin.gov.co/normograma/docs/index.html). La idea es crear una plataforma similar, pero enfocada en **archivos propios del proyecto de asistencia a las JAL**, con búsqueda avanzada en todos los documentos, filtros y visualización de resultados.

---

1) Referencia del Normograma & Adaptación JAL

El Normograma de Medellín permite: - Buscar normas por palabras clave. - Filtrar por categoría, tipo de norma, año. - Ver detalle de cada norma con metadatos y texto completo.

Nuestra Caja de Herramientas JAL tendrá un enfoque similar pero con estas diferencias: - Archivos propios (PDF, DOCX, XLSX, PPTX, imágenes, CSV, enlaces). - Filtros ajustados a **procesos JAL** (tipo de documento, etapa, año, etiquetas). - Interfaz visual moderna (React + Tailwind + shadcn/ui). - Roles diferenciados (ver sección 2). - **Colecciones (kits)** temáticos para facilitar la navegación (ej. Kit de posesión, Kit de planeación, Kit de control político).

---

2) Roles y público objetivo

Roles principales

- **Comunidad (Ediles y ciudadanos)**: pueden buscar, visualizar y descargar documentos.
- **Administrador**: sube y gestiona nuevos archivos, administra usuarios, colecciones, sinónimos y cuenta con un **dashboard** para ver descargas, actividad y métricas de uso.

Casos de uso

1. Un edil busca “plan de desarrollo comunal” y accede a los documentos relevantes.
2. El administrador sube un nuevo formato de control político, añade metadatos y lo publica.
3. El administrador revisa en el dashboard cuántas descargas tuvo un documento.

---

3) Arquitectura técnica

Frontend

- **React + Vite + TypeScript**
- **UI**: Tailwind + shadcn/ui
- **Visor de documentos**: pdf.js integrado

Backend & Buscador

- **API**: Node/Express o NestJS

- **BD**: Postgres (metadatos, usuarios, métricas)
- **Buscador avanzado**: Meilisearch/Typesense
- Con sinónimos y relevancia ajustada.
- Posibilidad de integrar modelo de **IA semántica** para mejorar resultados (ej. embeddings con OpenAI o Cohere, similar a la búsqueda de YouTube: tolerancia a variaciones, relevancia semántica, autocompletado inteligente).
- **Indexación**: Apache Tika/Textract + OCR (Tesseract) para PDFs escaneados.
- **Almacenamiento**: Azure Blob Storage (reemplaza S3/R2).

Hosting en Azure

- Frontend: Azure Static Web Apps.
- Backend/API: Azure App Service o Azure Functions.
- Base de datos: Azure Database for PostgreSQL.
- Buscador: desplegado en VM/Container de Azure (Meilisearch/Typesense).
- Archivos: Azure Blob Storage.

---

4) Modelo de datos simplificado

Documentos - `id` - `title` - `summary` - `type` (guía, formato, plantilla, normativa, informe, link, otro) - `stage` (diagnóstico, planeación, ejecución, seguimiento, control político) - `keywords`, `tags` - `year` - `owner` - `storage_key` - `file_ext` - `version` - `permissions` (público/interno) - `created_at`, `updated_at`

Usuarios - `id`, `name`, `email`, `role` (Comunidad, Administrador)

Métricas - `document_id`, `views`, `downloads`, `last_access`

---

5) Funcionalidades principales

- **Búsqueda avanzada híbrida**:
- Full-text con Meilisearch/Typesense.
- **IA semántica** (opcional) para consultas más naturales y relevantes.
- Autocompletado inteligente.
- **Filtros**: tipo, etapa, año, etiquetas.
- **Resultados**: lista con fragmentos resaltados.
- **Detalle de documento**: metadatos + visor PDF + descargas.
- **Colecciones/Kits**: agrupación temática de documentos.
- **Historial & favoritos** (por usuario).
- **Dashboard del Administrador**:
- Descargas y vistas por documento.
- Términos de búsqueda más frecuentes.
- Documentos más consultados.
- Búsquedas sin resultados (para detectar brechas).

---

6) UI/UX inspirada en Normograma

- **Home**: buscador central + accesos rápidos a colecciones.
- **Resultados**: listado con filtros activos y paginación.
- **Detalle**: visor de documento, metadatos, descargas.
- **Admin**: dashboard de descargas y métricas, gestión de usuarios, documentos y sinónimos.

---

7) Roadmap

- **MVP**: carga de documentos + buscador avanzado (Meilisearch) + filtros + ficha de documento + roles básicos.
- **Fase 2**: dashboard de métricas, colecciones, OCR.
- **Fase 3**: integración de IA semántica para búsqueda estilo YouTube, búsquedas guardadas y alertas.

---

8) Próximos pasos

1. Confirmar filtros prioritarios (tipo, etapa, año, etiquetas).
2. Definir kits iniciales (colecciones temáticas).
3. Preparar 50–100 documentos para carga piloto.
4. Montar front React + backend Node en Azure + Meilisearch.
5. Implementar dashboard para administradores.
6. Explorar integración de búsqueda semántica (IA) en la fase 3.
