Asistente Legal RAG

Código de Trabajo de Costa Rica

Bitácora técnica del proyecto

Sofia de la Cruz

Mayo 2026

asistente-laboral-cr.vercel.app

Este documento describe el proceso de construcción de un sistema RAG (Retrieval-Augmented Generation) sobre legislación laboral costarricense. El objetivo fue aprender el pipeline completo de primera mano: procesamiento de documentos, embeddings, búsqueda vectorial, prompt engineering, e integración en una interfaz conversacional deployada públicamente. La documentación cubre decisiones de diseño, alternativas evaluadas, errores encontrados y lecciones aprendidas en cada fase.

Contenido

Resumen ejecutivo

Los modelos de lenguaje de propósito general presentan limitaciones documentadas cuando se consultan sobre derecho laboral en jurisdicciones de menor representación en datos de entrenamiento, como Costa Rica: tienden a citar artículos incorrectos, inventar jurisprudencia, o responder con confianza desde información desactualizada. Un sistema RAG con corpus acotado y fuentes oficiales aborda este problema de manera específica: el modelo solo puede fundamentar respuestas en los artículos que tiene indexados, y cuando la respuesta no está en el corpus, declara el vacío en lugar de generar contenido no fundamentado.

El caso de uso elegido — asistente conversacional sobre el Código de Trabajo, Ley 9738 de Teletrabajo y Ley 2412 de Aguinaldo — combina elementos técnicamente convenientes para el aprendizaje: corpus extenso pero acotado, estructura jerárquica clara, fuentes públicamente disponibles, y dominio donde las consecuencias de una respuesta incorrecta son verificables.

Stack tecnológico

Procesamiento	Python, pdfplumber, requests, BeautifulSoup
Embeddings	Cohere embed-multilingual-v3.0 (1024 dimensiones)
Vector DB	Supabase + pgvector, índice HNSW, distancia coseno
Modelo de chat	Google Gemini 2.5 Flash vía API compatible con OpenAI
Backend	Next.js 16 + TypeScript + Vercel AI SDK v6
Rate limiting	Upstash Redis — 10 req/min, 50 req/día por IP
Hosting	Vercel, auto-deploy desde GitHub

Fase 1: Selección y obtención de fuentes

Antes de iniciar la implementación, se investigó si existían soluciones similares para usuarios costarricenses. Se encontraron versiones rudimentarias: un GPT personalizado en YesChat sin diseño de producto, una aplicación en Google Play que funciona como eBook con búsqueda de texto, y guías en inglés orientadas a empleadores extranjeros. Ninguna implementación bien ejecutada dirigida al usuario costarricense.

Documentos procesados

Documento	Fuente	Formato
Código de Trabajo (versión con Reforma Procesal Laboral)	MTSS oficial	PDF
Ley 9738 — Teletrabajo	MTSS oficial	PDF
Ley 2412 — Aguinaldo	PGR / SCIJ	HTML scraped

Exclusiones de scope y justificación

Decreto de salarios mínimos	Se actualiza semestralmente. Incluirlo en un corpus estático generaría respuestas desactualizadas con cada nuevo decreto.
Jurisprudencia (Sala II y Sala IV)	Limitación real del sistema. El derecho laboral CR en la práctica se interpreta vía jurisprudencia. Declarado explícitamente como limitación conocida y roadmap de v2.
Constitución Política	No relevante para consultas prácticas del usuario promedio.
Convenios OIT	Dominio especializado, fuera del alcance de v1.

> *Decisión arquitectónica: las fuentes se versionan localmente con git como artefactos de build-time. La aplicación en producción consulta únicamente la base vectorial; las fuentes no son consultadas en runtime. Esto da control explícito sobre qué versión de cada ley está reflejada en los embeddings.*

Fase 2: Procesamiento de documentos

Esta fase transforma las fuentes en una base estructurada de artículos que alimenta la base vectorial. Es el paso más crítico del pipeline: errores aquí se propagan sin señales visibles a todas las fases posteriores.

Enfoque de dos fases con revisión intermedia

En lugar de un parser monolítico de una sola pasada, se usó un enfoque en dos fases: primero extracción cruda a texto plano para inspeccionar particularidades del documento, luego parseo estructurado que produce el JSON final. La revisión manual entre fases detectó clases de error que las aserciones automáticas no habrían encontrado.

Schema de salida

kind	"articulo" o "transitorio" — los documentos legales contienen ambos y tratarlos como iguales es legalmente incorrecto.
article_suffix	"bis", "ter", "quater", "quinquies" o null. Artículos insertados por reformas se numeran con sufijos en lugar de renumerar el código.
chapter	Ocho campos nullable (libro, titulo, capitulo, seccion + sus títulos). Permite construir el string jerárquico en cualquier momento y habilita filtros reales.
title	Sumilla (encabezado corto) cuando existe. Controlado con flag has_sumilla por fuente.

Clases de error encontradas en validación manual

Las aserciones automáticas (conteos, IDs únicos, ausencia de cuerpos vacíos) pasaron en todas las rondas. Los errores críticos fueron detectados únicamente mediante validación manual sobre muestras dirigidas (artículos conocidos, casos límite, transiciones de capítulo):

— Heurísticas demasiado permisivas para detección de sumillas — falsos positivos en leyes que no las usan.

— Contaminación con boilerplate: notas al pie y bloques de firma formal del documento.

— Pérdida de cierres de citación cuando la detección de headers eliminaba paréntesis finales.

— Ocho artículos silenciosamente omitidos por casing inconsistente ("ARTÍCULO" vs "Artículo" en el PDF).

— Doce artículos con sufijos romanos (bis, ter, quater) absorbidos en el artículo padre.

Los veinte artículos afectados incluían el procedimiento de despido de mujer embarazada, el régimen de huelga en servicios esenciales, y la capacidad procesal — contenido que un usuario buscaría directamente.

Resultado final del corpus

Fuente	Artículos	Transitorios	Total
Código de Trabajo	708	11	719
Ley 9738 — Teletrabajo	10	1	11
Ley 2412 — Aguinaldo	10	4	14
Total	728	16	744

Fase 3: Embeddings y búsqueda vectorial

Un sistema RAG separa dos responsabilidades en modelos distintos: el modelo de embeddings convierte texto en vectores numéricos para búsqueda por similitud semántica, y el modelo de chat genera la respuesta en lenguaje natural a partir de los fragmentos recuperados. Estas decisiones son independientes.

Selección del modelo de embeddings

Se evaluaron cinco opciones: OpenAI text-embedding-3, Voyage voyage-3-large, Voyage voyage-law-2 (especializado en derecho), Cohere embed-multilingual-v3.0, y BAAI/bge-m3 (open-source).

Modelo seleccionado	Cohere embed-multilingual-v3.0
Justificación principal	El corpus está completamente en español. Cohere multilingüe v3 fue entrenado para retrieval cross-lingual desde su arquitectura base, a diferencia de modelos con multilingüismo añadido sobre un núcleo en inglés.
Alternativa descartada	Voyage law-2 ofrece especialización legal pero fue entrenado principalmente en corpus en inglés. La ventaja de especialización probablemente no transfiere limpiamente a español.
Nota metodológica	No se ejecutó comparación A/B entre modelos de embeddings. La optimización se difiere a v2, una vez existan consultas reales para evaluar retrieval de forma empírica.

Base de datos vectorial: Supabase + pgvector

Índice	HNSW con distancia coseno — más rápido que IVFFlat a la escala del proyecto.
Schema	Tabla plana (no por ley) con chapter almacenado como JSONB. Queries más simples, expansión sin joins.
Seguridad	Row Level Security con dos políticas: lectura pública (anon role), escritura solo vía service_role.

Estrategia de chunking

El 12% del corpus excede el límite de ~512 tokens de Cohere. La truncación simple perdería contenido en artículos clave (Artículo 95 sobre maternidad, Artículo 81 sobre causas de despido). Se implementó chunking inteligente por párrafos con dos mejoras:

Prefijo jerárquico	Cada chunk incluye la ruta completa: Libro > Título > Capítulo > Sección > Artículo. Mejora retrieval para consultas temáticas cuando el vocabulario de la consulta difiere del texto del artículo.
Sibling completion	Cuando retrieval retorna un chunk parcial de un artículo multi-chunk, una consulta de seguimiento obtiene los chunks hermanos faltantes antes de enviar el contexto al modelo. Previene que el modelo "complete" contenido faltante desde su memoria de entrenamiento — falla concreta observada sin esta mejora: respuesta citó el Artículo 81 inciso i como catch-all de falta grave cuando el correcto es el inciso l.

Validación end-to-end

Consulta de validación: "¿cuántos días de vacaciones me corresponden?" — Resultado: Artículo 153 como top match (similitud coseno 0.686); posiciones 2-5 ocupadas por Artículos 154-156, todos en la sección de vacaciones anuales. Cross-lingual semantic match confirmado: la consulta usó "días", el artículo usa "semanas".

Fase 4: Interfaz conversacional

Arquitectura multi-proveedor

El backend de chat se diseñó para ser agnóstico al proveedor del modelo: con tres variables de entorno (CHAT_MODEL, API key del proveedor, OPENAI_BASE_URL) es posible cambiar de modelo sin modificar código. La mayoría de proveedores modernos exponen APIs compatibles con el SDK de OpenAI. Esta decisión convirtió la selección del modelo en una decisión empírica iterable.

Proceso de selección del modelo de chat

Se iteró a través de tres modelos en secuencia, con el mismo prompt y el mismo conjunto de consultas de prueba:

GPT-4o-mini (iter. 1)	Errores de atribución en citas multi-artículo y fugas del disclaimer en rechazos. No se resolvieron con ajuste de prompt.
GPT-4o (iter. 2)	Errores de citación resueltos. Nueva falla: deriva hacia comentarios legales generales no fundamentados después de declarar un vacío, introducidos con frases como "En términos generales". Múltiples iteraciones del prompt redujeron la deriva pero no la eliminaron — característica estructural del modelo.
Gemini 2.5 Flash (iter. 3)	Seleccionado como modelo de producción por evidencia empírica.

Métrica de evaluación	GPT-4o	Gemini 2.5 Flash

Frase prohibida "En términos generales" (3 consultas)	2 / 3	0 / 3
Frase prohibida "En la práctica" (3 consultas)	1 / 3	0 / 3
Citaciones promedio por respuesta	1–2 artículos	3–4 con incisos
Costo estimado por consulta	~USD 0.0094	~USD 0.0006

Hipótesis sobre la diferencia: los modelos de Google y Anthropic tienden a ser entrenados con mayor énfasis en seguimiento estricto de instrucciones, mientras que OpenAI optimiza más fuertemente por "utilidad". Esta diferencia es estructuralmente relevante cuando las reglas del sistema son explícitamente restrictivas.

Diseño del system prompt

El system prompt tiene 11 secciones. Cada sección responde a una falla concreta observada durante la evaluación — no fue diseñado de una sola vez sino iterado contra evidencia empírica:

1. Persona + scope	Define qué corpus conoce el modelo y sus límites explícitos.
2. Regla fundamental	Toda respuesta sustantiva debe basarse únicamente en los artículos del contexto. Sin invención, sin inferencias, sin conocimiento general.
3. Conceptos no establecidos	Patrón de reconocimiento de vacíos para conceptos doctrinales (jurisprudencia, derechos adquiridos, jus variandi): declarar qué SÍ está en los artículos y qué requiere análisis legal más amplio.
4. Después de declarar un vacío	Blocklist de frases que señalan riesgo de alucinación: "En términos generales", "En la práctica", "Por lo general", "Normalmente", "Cabe destacar". La estructura correcta es declaración del vacío → cierre, sin paso intermedio.
5. Tono y registro	6 reglas numeradas: apertura situación-primero, sin fórmulas de cortesía vacías, lenguaje cotidiano con transformaciones explícitas, oraciones cortas, citas integradas naturalmente, "usted" consistente.
6. Citación de artículos	Nunca usar marcadores [N] internos del contexto. Formato: "Artículo X del Código de Trabajo". Cero ocurrencias de corchetes en la salida salvo en citas textuales del artículo.
7. Estructura de respuestas	80–250 palabras típicamente; viñetas para preguntas de múltiples partes; prosa para preguntas simples.
8. Manejo de casos específicos	Para situaciones personales: responder con ley general aplicable más caveat de que cada caso depende de circunstancias específicas.
9. Rechazo	Preamble de verificación (revisar retrieval antes de rechazar). Tipo A: pregunta laboral fuera del corpus — incluye referencias institucionales. Tipo B: fuera del tema laboral. Decisión basada en palabras clave concretas, no en criterios ambiguos.
10. Cierre	Dos párrafos verbatim al final de toda respuesta sustantiva: referencia al MTSS y Defensa Pública, luego disclaimer. Los rechazos omiten este cierre.

11. Calibración de tono	Referencia a transcripciones reales de @beatrizherreracr, abogada laboralista costarricense con 134.5K seguidores. El chatbot replica el registro conversacional pero no la autoridad expansiva.

Resultados de evaluación final (15 consultas)

Distribuidas en cuatro categorías: consultas comunes, consultas de frontera, consultas conversacionales extraídas de videos de usuarios reales, y consultas con conceptos doctrinales pesados.

— Cero fabricaciones: todas las citaciones verificadas contra los chunks retornados por retrieval.

— Disclaimer presente en toda respuesta sustantiva; ausente en rechazos.

— "Usted" consistente en el 100% de las respuestas.

— Cero ocurrencias de marcadores [N] en salidas.

— Cero frases de la blocklist en salidas.

— Hedge doctrinal correcto para preguntas que requieren análisis más allá del texto literal.

— Rechazo limpio con clasificación correcta (tipo A vs tipo B) en todos los casos de prueba.

Fase 5: Frontend, deployment y producto

Decisiones de arquitectura de frontend

Base	Plantilla Vercel AI Chatbot clonada y reducida: de 27 archivos en app/ a 5, de 70+ componentes a 24. El patrón de streaming chat es una solución resuelta; el valor del proyecto está en el backend RAG.
Port a TypeScript	Backend reescrito de Python a TypeScript dentro de Next.js para simplicidad de deployment — evita orquestación de dos lenguajes y dos deployments.
Multi-turno con query rewriting	Antes del retrieval, si existen mensajes previos, una llamada previa a Gemini reescribe el último mensaje como pregunta autocontenida. Resuelve falla concreta: "y si solo trabajé 6 meses" → "¿Cuántos días de vacaciones si trabajé 6 meses?" El retrieval usa la versión reescrita; el modelo de respuesta recibe el historial completo.

Problemas descubiertos en uso real (no en evaluación automatizada)

Al menos cuatro problemas críticos se encontraron únicamente al interactuar con el sistema como usuario en navegador:

Silenciamiento de errores	Errores mid-stream (cuota de proveedor agotada, timeouts) producían HTTP 200 con stream vacío — la UI mostraba nada. Resuelto separando errores pre-stream (→ HTTP 4xx/5xx) de errores mid-stream (→ callback onError del SDK).
Follow-ups rotos	El retrieval usaba solo el último mensaje del usuario como query, ignorando el historial. Resuelto con query rewriting descrito arriba.
Presunción de conflicto	El cierre original decía "para hacer valer sus derechos, puede acudir al MTSS" — presuponía un conflicto en consultas puramente informativas. Versión final: "si en algún momento necesita hacer valer sus derechos" — condicional.

Inconsistencia en rechazos	Los rechazos laborales no ofrecían recursos institucionales, a diferencia de las respuestas sustantivas. Resuelto implementando dos tipos de rechazo: tipo A (laboral, incluye MTSS y Defensa Pública), tipo B (fuera de tema, remite a profesional del área).

Métricas del sistema deployado

Métrica	Valor
Latencia al primer token	~2–4 segundos (pipeline completo: rewriting + embedding + retrieval + sibling completion + streaming)
Respuesta completa	~8–12 segundos
Costo por consulta (Gemini 2.5 Flash)	~USD 0.0016
Rate limiting	10 req/min, 50 req/día por IP (Upstash Redis)
Confiabilidad eval final	15/15 — cero alucinaciones de citación, cero fallos de rechazo
URL pública	asistente-laboral-cr.vercel.app

Anexo A: System Prompt completo

Se reproduce íntegramente el system prompt de producción. Las secciones están numeradas en el orden en que el modelo las recibe.

1. Persona + scope

```
Eres un asistente conversacional especializado en derecho laboral
costarricense. Tu base de conocimiento se limita ESTRICTAMENTE al
Código de Trabajo, la Ley 9738 de Teletrabajo y la Ley 2412 de
Aguinaldo. No tienes información sobre ninguna otra ley, jurisprudencia,
salarios mínimos vigentes, ni decretos.
```

2. REGLA FUNDAMENTAL

```
Toda respuesta sustantiva debe basarse ÚNICAMENTE en los artículos
provistos abajo en el contexto. No inventes información, no infieras
contenido de artículos que no aparecen en el contexto, no completes
con conocimiento general. Si el contexto no responde la pregunta,
aaplica la regla de rechazo.
```

3. CONCEPTOS NO ESTABLECIDOS EN EL TEXTO LITERAL

```
Algunos conceptos del derecho laboral (como "derechos adquiridos",
"jurisprudencia", "doctrina", "jus variandi", "principios generales
del derecho") NO están establecidos en el texto literal del Código
de Trabajo ni de las leyes en tu base. Si una pregunta requiere
estos conceptos para responderse completamente, indica claramente
qué SÍ está en los artículos y qué requiere análisis legal más
amplio que no está en tu base de datos.
```

4. DESPUÉS DE DECLARAR UN VACÍO LEGAL

```
Cuando declares que el Código no cubre algo, ABSOLUTAMENTE NO
continúes con consejo legal general. No agregues oraciones que
comiencen con frases como "En términos generales", "En la práctica",
"Sin embargo, [generalidad]", "Es importante considerar", "Por lo
general", "Normalmente", "Cabe destacar". Estas frases son señales
de que estás a punto de inventar contenido.

La estructura correcta es: declaración del gap → CIERRE. Sin paso intermedio.
```

5. TONO Y REGISTRO

1. Abre conectando con la situación del usuario, no con definiciones abstractas.
2. NO uses fórmulas como "Vea,", "Veamos,", "Bonita pregunta", o saludos.
3. Usa lenguaje cotidiano: "Tiene seis meses para presentar esta acción" en lugar de "El plazo de prescripción es de seis meses".
4. Evita cláusulas condicionales largas y anidadas.
5. Cita artículos integrándolos naturalmente.
6. Usa "usted" de forma consistente. Nunca "tú", "te", "tus".

6. CITACIÓN DE ARTÍCULOS

Cita SIEMPRE los artículos específicos. Formato natural: "Artículo X del Código de Trabajo". NUNCA uses marcadores [N] del contexto — son índices internos. La salida correcta tiene cero ocurrencias de corchetes "[" o "]" salvo dentro de citas textuales del artículo.

7. ESTRUCTURA DE RESPUESTAS

Típicamente 80–250 palabras. Viñetas para preguntas con múltiples partes. Prosa para preguntas simples.

8. MANEJO DE CASOS ESPECÍFICOS

Para situaciones personales: ley general aplicable más caveat de que cada caso depende de circunstancias específicas.

9. RECHAZO

Antes de rechazar, evalúa si los artículos recuperados ofrecen contenido útil temáticamente relacionado.

Tipo A (pregunta laboral fuera del corpus):
"No encuentro información específica sobre esto en mi base de datos, que cubre el Código de Trabajo, la Ley 9738 de Teletrabajo y la Ley 2412 de Aguinaldo. Para asuntos relacionados con sus derechos laborales, puede acudir gratuitamente al Ministerio de Trabajo (MTSS), donde la Dirección Nacional de Inspección del Trabajo recibe denuncias en sus oficinas o en línea. Si necesita representación legal y gana menos de aproximadamente ■800.000 mensuales, la Defensa Pública del Poder Judicial brinda asesoría gratuita."

Tipo B (fuera del tema laboral):
"No encuentro información sobre esto en mi base de datos, que cubre el Código de Trabajo, la Ley 9738 de Teletrabajo y la Ley 2412 de Aguinaldo. Para asuntos fuera de este alcance, le recomiendo consultar con un profesional en el área correspondiente."

10. CIERRE

Toda respuesta sustantiva termina con EXACTAMENTE estos dos párrafos verbatim:

Párrafo 1: "Si en algún momento necesita hacer valer sus derechos, puede acudir gratuitamente al Ministerio de Trabajo (MTSS), donde la Dirección Nacional de Inspección del Trabajo recibe denuncias en sus oficinas o en línea. Si necesita representación legal y gana menos de aproximadamente ■800.000 mensuales, la Defensa Pública del Poder Judicial brinda asesoría gratuita."

Párrafo 2: "Esta información es de carácter informativo y no constituye asesoría legal. Para casos específicos, consulte con un profesional."

Los rechazos NO incluyen estos párrafos.

Anexo B: Limitaciones conocidas del sistema

Documentadas explícitamente. Reconocer los límites de un sistema de IA es parte del trabajo responsable con estas tecnologías.

Jurisprudencia	No incluye votos de Sala Segunda y Sala Constitucional. En la práctica, el derecho laboral costarricense se interpreta vía jurisprudencia — esta es la limitación más relevante del sistema para casos reales.
Salarios mínimos	No incluye montos vigentes. El Decreto de Salarios Mínimos se actualiza semestralmente; incluirlo en un corpus estático generaría respuestas desactualizadas.
Fecha del corpus	Información actualizada hasta la versión de las fuentes oficiales utilizadas (agosto 2024 para el Código de Trabajo del MTSS).
Artículos tabulares	Artículos con listas enumeradas extensas (por ejemplo, Artículo 224 con la tabla de incapacidades por lesiones) se dividen en muchos chunks pequeños — funciona para consultas específicas pero no para recuperar la tabla completa.
Modelo de embeddings	No especializado en derecho. Una evaluación futura podría comparar el modelo multilingüe actual contra modelos legal-specific una vez existan consultas reales para medir retrieval empíricamente.
Asesoría legal	El sistema no constituye asesoría legal. Todas las respuestas incluyen disclaimer explícito. Para asuntos formales, se requiere consulta con un profesional.

Sofia de la Cruz · sofiadelacruz.com · sofiadelacruzgross@gmail.com