

Asistente Asesor de Alarmas para Incendio por Telegram

- Documentación Tarea Final Miguel Flores -

Descripción General del Proyecto

El proyecto consiste en el desarrollo e implementación de un **Agente de Inteligencia Artificial (RAG)** integrado en **Telegram**, diseñado para asistir a técnicos e instaladores en consultas sobre sistemas de detección de incendios de la marca INIM y normativas vigentes.

El sistema resuelve la necesidad de consultar documentación técnica extensa (manuales y normas) de forma inmediata y conversacional, permitiendo interacciones tanto por **texto** como por **mensajes de voz**.

El sistema se divide en dos flujos de trabajo automatizados (Workflows) que operan de manera complementaria:

1. **Workflow de Ingesta y Sincronización (Backend):** Se ejecuta automáticamente para monitorear una carpeta en **Supabase Storage**. Detecta nuevos manuales técnicos, extrae su texto, genera "embeddings" (vectores) y actualiza la base de conocimientos vectorial.
2. **Workflow de Asistencia (Frontend - Telegram):** Interfaz de usuario que procesa mensajes de texto y audios de voz. Utiliza un **Agente de IA** capaz de consultar la base vectorial para generar respuestas técnicas precisas basadas exclusivamente en la documentación cargada.

Arquitectura Lógica del Flujo

El sistema consta de dos arquitecturas separadas:

A) Workflow de Ingesta y Mantenimiento (Carga de Documentos)

Archivo fuente: Tarea Final Miguel Flores20251120final2.json

Este flujo asegura que el asistente siempre tenga la información actualizada sin duplicar datos.

#	Nodo	Función / Lógica Principal
1	Schedule Trigger	Inicia el flujo automáticamente en intervalos definidos (polling) para buscar cambios.
2	Supabase (Get All)	Consulta el estado actual de la base de datos para saber qué archivos ya fueron procesados.
3	HTTP Request	Lista los archivos reales presentes en el Storage de Supabase (carpeta "almacenamiento").
4	Code (Comparar Estados)	Ejecuta un script JS que compara los archivos del Storage vs. la Base de Datos. Clasifica los archivos en tres listas: <i>Agregar, Modificar y Eliminar</i> .
5	Switch / IF	Evalúa si existen archivos nuevos para procesar. Si no hay cambios, detiene el flujo.
6	Split Out / Loop	Separa los archivos nuevos para procesarlos uno por uno.
7	HTTP Request (Download)	Descarga el archivo binario (PDF, CSV, etc.) desde el Storage para su lectura.
8	Extract from File	Extrae el texto plano del archivo (soporta PDF, XLSX, CSV).
9	Embeddings Cohere	Convierte el texto extraído en vectores numéricos (embeddings) utilizando el modelo <code>embed-multilingual-v3.0</code> .
10	Supabase Vector Store	Guarda los vectores y el texto en la tabla <code>auxiliar</code> para ser consultados por el Agente.

b) Workflow de Interfaz de Usuario (Telegram + Agente IA)

Archivo fuente: *TpFinaLn8nMiguelFlores20251120final2.json*

Este flujo maneja la interacción directa con el técnico o usuario final.

#	Nodo	Función / Lógica Principal
1	Telegram Trigger	Recibe el mensaje del usuario. Detecta si es Texto o Nota de Voz .
2	IF (¿Mensaje Audio?)	Discrimina el tipo de entrada. Si es texto, pasa directo al Agente. Si es audio, desvía al flujo de transcripción.
3	Get Audio (Telegram)	Si es voz, descarga el archivo de audio temporalmente.
4	Google Gemini (Transcribe)	Utiliza el modelo <code>gemini-2.0-flash-lite</code> para transcribir el audio a texto con alta precisión técnica.
5	AI Agent (LangChain)	<p>Cerebro del sistema. Orquesta la conversación.</p> <ul style="list-style-type: none"> • Prompt: Configurado como experto técnico INIM. • Memory: Recuerda los últimos 7 mensajes para mantener contexto.
6	Supabase Vector Store (Tool)	Herramienta que el Agente utiliza para buscar información técnica específica en la base de datos vectorial (tabla <code>auxiliar</code>).
7	Cohere Chat Model	LLM que procesa la información recuperada y genera la respuesta final en lenguaje natural.
8	Telegram Send Message	Envía la respuesta técnica generada de vuelta al usuario.

Configuración del Cerebro del Asistente (System Prompt)

El comportamiento del Agente de IA está regido estrictamente por la siguiente instrucción de sistema (extraída directamente del nodo *AI Agent* del workflow de producción):

"Eres un asistente técnico especializado en sistemas de alarma de incendio INIM, incluyendo centrales, módulos, dispositivos, topologías de lazo, parametrización, normas IRAM y criterios de instalación.

Tu función es responder consultas utilizando exclusivamente la información contenida en el CONTEXTO proporcionado. No uses conocimientos externos, no completes lagunas y no infieras información no presente en el CONTEXTO.

Si el CONTEXTO no contiene información suficiente, respondé estrictamente: "No poseo información sobre eso en los documentos cargados."

Las respuestas deben ser:

- Técnicas, precisas y orientadas a resolución.
- Redactadas en español técnico estándar.
- Breves y estructuradas (máximo 12–14 líneas).
- Adecuadas para ser enviadas por Telegram (sin bloques largos, sin formato complejo).
- Sin repreguntas, sin ambigüedad y sin introducir temas no solicitados.

Respondé siempre de forma cerrada, directa y finalizada en el primer intento."

Requerimientos Externos y Base Documental

Base Documental (RAG)

La "inteligencia" del bot reside en los documentos técnicos procesados y almacenados en la tabla `auxiliar` de Supabase. Los documentos activos actualmente son:

1. **Manual de Instalación:** *Previdia Micro-S_guida_111.pdf*
 - Contiene esquemas de conexión, especificaciones de terminales, cableado de lazos y resistencias.
2. **Normativa Técnica:** *CIR-GT5-Sistemas de deteccion-Junio 2018 Rev00.pdf*
 - Define clasificación de detectores, alturas máximas de montaje y espaciamientos según normas IRAM/NFPA.
3. **Reglamentación Local:** *PE-RES-MJGGC-SSREGIC-99-20-ANX-2.pdf*
 - Establece requisitos legales para instalaciones contra incendio en CABA, funciones de control y mantenimiento.
4. **Datasheets:** *Previdia Compact_datasheet_120_ES*, *IC0020_datasheet_110_S0*, *ES2000_datasheet_120_ES* y *ED200_datasheet_120_ES*.

Estos documentos aportan conocimiento sobre:

- Centrales INIM, módulos y lazos.
- Condiciones de instalación.
- Normativa IRAM aplicable.
- Modos operativos.
- Criterios de programación.

Todos los fragmentos fueron insertados en la tabla `auxiliar` con embeddings de dimensión 1024.

Credenciales y Requisitos Técnicos

Para el correcto funcionamiento de los workflows, se han configurado las siguientes integraciones:

Supabase:

- URL del proyecto y claves de acceso (*anon key* o *service_role key*).
- Tabla auxiliar configurada con campo `embedding` tipo `vector(1024)`.
- Función RPC `match_documents()` habilitada para la búsqueda por similitud.
- Extensión `pgvector` instalada en la base de datos PostgreSQL.

Telegram Bot API:

- Token del bot generado por BotFather.
- Webhook configurado y vinculado al nodo Trigger de n8n.

Modelo de Lenguaje (LLM):

- Se ha configurado el uso de modelos como **Cohere** o **Google Gemini Flash** (según disponibilidad de créditos/API Key activa) para la generación de texto y embeddings.

Speech-to-Text:

- Para el procesamiento de audios se utiliza la API de **Google Gemini** (modelo flash-lite) capaz de entender terminología técnica en español.

Lógica Temporal y Condicional

El sistema implementa lógica condicional avanzada para manejar diferentes tipos de entrada y asegurar la eficiencia del procesamiento:

A. Lógica de Selección de Medio (Audio vs. Texto):

- El flujo evalúa el objeto JSON entrante desde Telegram.
- **Condición:** Si el campo `message.voice` existe, el flujo se desvía a la rama de procesamiento de audio (descarga del archivo binario + transcripción con IA).
- **Condición:** Si el campo `message.text` existe, el flujo omite la transcripción y pasa el texto directamente al Agente RAG.

B. Lógica de Ingesta de Documentos (Comparación de Estados):

- Para evitar re-procesar documentos (ahorrando costos de cómputo), el sistema no sube todo ciegamente.
- **Comparación:** Un script compara la lista de archivos actuales en el Storage contra el registro histórico en la base de datos.

- **Bifurcación (Switch):**

- *Archivos Nuevos:* Se envían a la rama de "Split Out" para extracción y vectorización.
- *Archivos Eliminados:* Se ejecuta una acción de limpieza en la base de datos para borrar los vectores obsoletos.
- *Sin Cambios:* El flujo termina su ejecución automáticamente.

C. Lógica de Respuesta del Agente:

- El Agente tiene una instrucción estricta que si la búsqueda vectorial en Supabase no devuelve fragmentos con suficiente similitud, el LLM tiene prohibido alucinar una respuesta.
- **Condición:** Si `Contexto == Null` o irrelevante → **Salida:** "No poseo información sobre eso en los documentos cargados."
- **Respuesta de Fallo:** Si el *Vector Store* no encuentra información con suficiente similitud, el *System Prompt* del Agente está instruido para responder: "*No poseo información sobre eso en los documentos cargados*", evitando alucinaciones.

Evidencia de Funcionamiento y Validación de Casos de Uso

A continuación, se documentan visualmente las interacciones en tiempo real con el Asistente Técnico a través de la interfaz de Telegram. Las siguientes capturas de pantalla acreditan el cumplimiento de los objetivos funcionales del sistema, destacando específicamente:

- La precisión de las respuestas generadas por el motor RAG ante consultas técnicas complejas.
- La eficacia del módulo de transcripción automática para mensajes de voz.



