Sección 4 – Flujo Principal del Algoritmo en consulta.js

Esta sección describe el flujo principal del algoritmo que maneja las consultas en el sistema. El proceso consta de seis pasos principales, asegurando que la consulta sea procesada de manera eficiente y precisa.

Paso 1: Pulido de la Pregunta

El sistema recibe la consulta original del usuario y la somete a un proceso de refinamiento para mejorar su claridad y precisión.

Se realiza el llamado a la función pulirPregunta() la cual entrega un **prompt perfecto** para continuar con la ejecución del algoritmo en sus próximos pasos, sin necesidad de mantener el historial para cada próximo paso.

- Se obtiene el historial de consultas anteriores.
- Se usa una SmartFunction para reestructurar la consulta considerando el contexto permitido.
- Se registran métricas sobre los tokens de entrada y salida.

Paso 2: Identificación de Archivos Relevantes

Con la consulta optimizada, el algoritmo busca los archivos más relevantes en el índice de documentos.

Se realiza el llamado a la smartFunction identificarArchivos(); cuyo objetivo es que la misma IA nos diga, teniendo en su mano la pregunta y el índice de los chunks "***¿cuáles son los chunks a los cuales irías a buscar esta información?"***

- Se accede al índice de documentos disponible en "*_Document_Index.txt*"
- Se filtran los archivos que contienen información relevante para la consulta.
- Se limita la cantidad de archivos a evaluar con el parámetro MAX_CHUNKS.
- Se actualizan métricas de tokens.

Paso 3: Construcción de la Respuesta

El sistema realiza la búsqueda de respuestas a la consulta realizada, dentro de cada uno de los chunks listados en el paso anterior. Se combina la pregunta original con la versión refinada prompt perfecto para enriquecer la búsqueda.

- Se extrae información clave de los documentos relevantes.
- Se usa MAX_ANSWERS para limitar el número de respuestas obtenidas.
- Se registran los tokens utilizados en esta fase.

La resultante de este paso es un string que contiene el texto de una o varias respuestas válidas encontradas. Las que serán pulidas en el próximo paso.

Paso 4: Pulido de la Respuesta

Si se encuentran respuestas relevantes, el sistema refina la salida antes de mostrarla al usuario.

- Se utiliza una Smart-Function para mejorar la coherencia y legibilidad de la respuesta final.
- Se aplican formato Markdown resaltando las ideas clave con negritas y viñetas.
- Se contabilizan nuevamente los tokens consumidos en este paso.

Paso 5: Actualización del Historial

El sistema guarda la consulta y la respuesta en el historial para mejorar futuras interacciones.

- Se almacena la consulta refinada y su respuesta en la base de datos.
- Este historial permite que el sistema mantenga el contexto en sesiones prolongadas.

La cantidad de iteraciones almacenadas en el Historial depende de la configuración MAX_HISTORIAL obrante en el archivo “.env”

Paso 6: Cálculo de Costos y Devolución de la Respuesta

Finalmente, se calcula el costo computacional y se envía la respuesta al usuario.

- Se determinan los tokens de entrada y salida utilizados en la consulta, en todos sus pasos.
- Se estima el costo total de la ejecución.
- Se envía la respuesta refinada al usuario en formato JSON.

Con esta estructura, el sistema garantiza eficiencia, precisión y optimización en cada consulta procesada.

A continuación una captura de pantalla de **los seis pasos** del algoritmo en ejecución: