Agente IA local con n8n

Introducción

Se buscaba la creación de un agente IA que funcionara totalmente en local, por eso se implementó todo lo necesario para este objetivo. El agente IA que se logró crear es capaz de leer archivos subidos a la base de datos vectorial en Qdrant y responder consultas basándose en estos.

Requisitos previos del sistema:

-Sistema operativo (Windows, Linux o macOS)

-Git (Ya que lo necesitaremos para clonar el repo de donde sacaremos el ambiente de desarrollo)

-Docker Desktop (Nos permite usar el ambiente de desarrollo para el agente en una forma aislada)

-WSL (Para usar Docker Desktop, ya que este permite que funcione de manera nativa en Windows)

-cURL (Este es opcional, pero es útil si se necesita arreglar algunas cosas desde la consola)

Ambiente de desarrollo:

Para generar nuestro agente, necesitaremos de algunas cosas, las cuales se nombrarán a continuación.

-n8n (Este nos permite crear los flujos de trabajo los cuales usaremos tanto para generar nuestro agente como para generar el flujo de trabajo que suba información a la base de conocimiento)

-Qdrant (Esta será nuestra base de conocimiento ya que nos permite almacenar información de manera vectorial la cual será interpretada por el agente)

-Ollama (Esta es la herramienta que nos permitirá correr modelos de lenguaje (LLMs) localmente en el equipo)

-Llama 3.2 (Será el modelo de lenguaje que usaremos)

-Nomic-embed-text (Será el encargado de convertir texto en vectores numéricos que capturan el significado o la estructura semántica del contenido, esto para que el agente entienda y trabaje el contenido de manera inteligente)

Paso a paso para la creación del agente

Paso 1: Se debe instalar tanto el Docker desktop como Git, esto lo hacemos desde las páginas oficiales tanto de Git como de Docker

Paso 2: se deben ejecutar la siguiente línea de comandos en la consola

```
git clone https://github.com/n8n-io/self-hosted-ai-starter-kit.git
cd self-hosted-ai-starter-kit
docker compose --profile cpu up
```

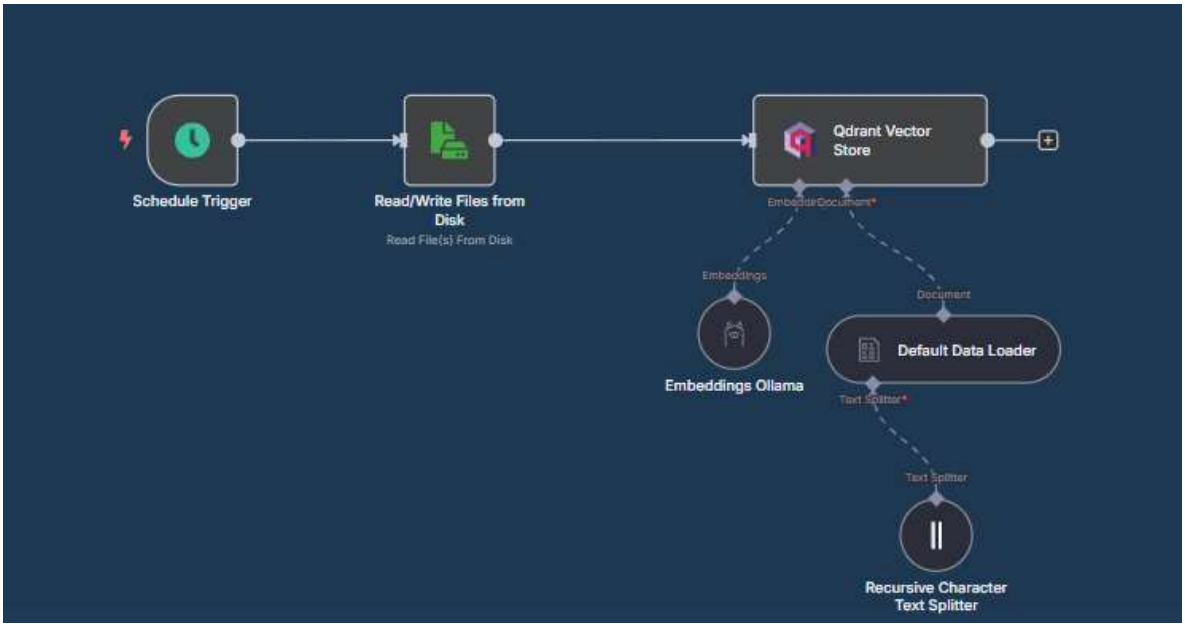
Esto lo que hará será instalarnos el ambiente de desarrollo en Docker Desktop, que trae n8n, Qdrant y Ollama, todo esto se juntará automáticamente en self-hosted-ai-starter-kit el cual es una agrupación de contenedores.

Paso 3: Iremos a el contenedor de Ollama y abriremos la consola, ya ahí descargaremos el modelo de IA de nuestra preferencia (Visitar la página de Ollama para buscar el modelo que se adapte a tus requerimientos o objetivos) en este caso se usará llama 3.2, para instalarlo usamos el comando: "ollama run llama3.2" en la consola.

Paso 4: De nuevo en la consola de Ollama ejecutaremos el siguiente comando "ollama run nomic-embed-text" para descargar el modelo de embedding.

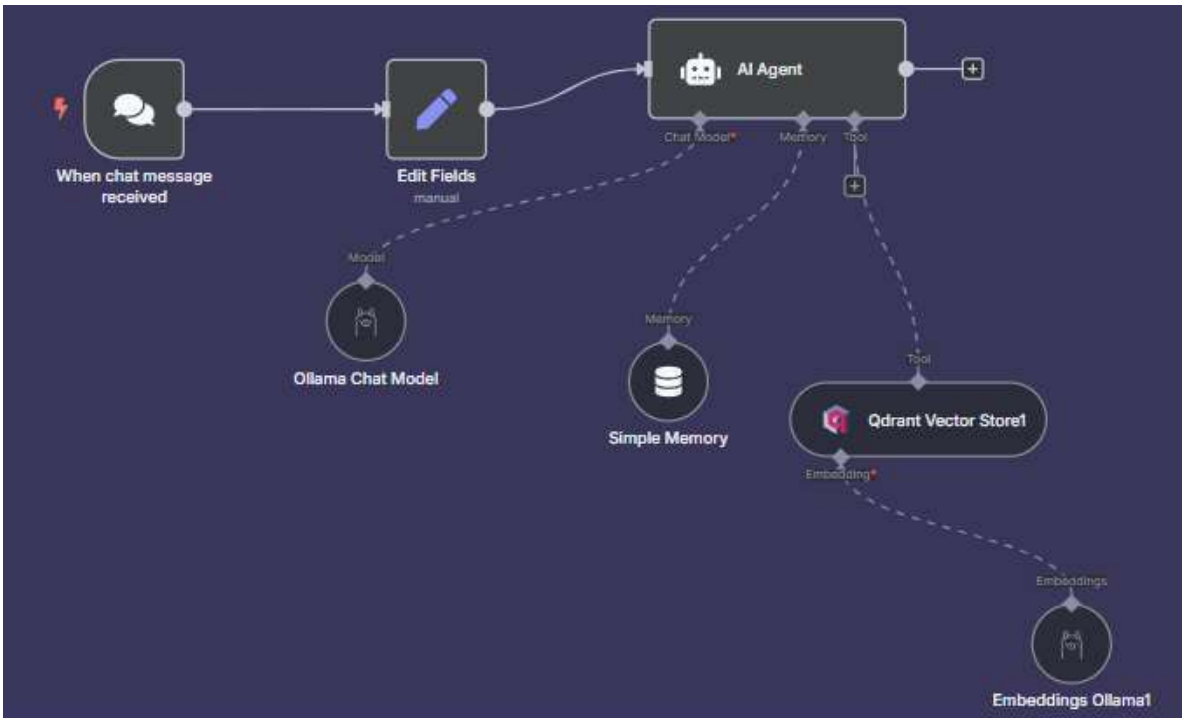
Paso 5: Iremos ahora a Docker Desktop y encenderemos la agrupación de contenedores self-hostedai-starter-kit, después procedemos a poner lo siguiente en nuestro buscador (ya sea Google u otro) http://localhost:5678 esto lo que hará será llevarnos al editor de n8n, donde empezaremos a crear nuestros flujos de trabajo.

Paso 6: Abrimos un nuevo flujo de trabajo, donde empezaremos uniendo los nodos que nos permitan subir archivos desde local hasta la base vectorial, en el caso de este agente fue hecho de la siguiente forma:



Donde el nodo "Schedule Trigger" será encargado de iniciar todo el flujo, el siguiente "Read/Write files from Disk" carga los archivos desde local y los siguientes nodos son encargados de subirlos a la colección de Qdrant que nosotros hayamos escogido o incluso crearla si lo configuramos adecuadamente.

Paso 7: Ya habiendo creado el flujo de trabajo encargado de subir archivos a la base de conocimiento, procedemos a crear el agente, que para este caso fue hecho de la siguiente forma:



Donde empezamos con un nodo de chat para hacer las consultas, un siguiente nodo que se encarga de asignarle una sessionId al prompt , y por último el agente al cual se le asigno como modelo de IA el llama 3.2, una memoria simple y como herramienta el acceso a la base de conocimiento.

Observaciones

-La velocidad con la que responde el agente varía dependiendo de la potencia de la maquina donde se esté ejecutando, por lo tal hay que ajustar los parámetros del agente de acorde a estas capacidades.

-Si se requiere acceder a la base de conocimiento y ver si hay realmente información adentro ó por otro motivo, iremos al siguiente link http://localhost:6333/dashboard#/collections y allí podemos ver las colecciones que tenemos con su información.