Documento informativo Proyecto RAG con agente de IA

Especificaciones del Equipo

- El proyecto fue desarrollado en un equipo con las siguientes especificaciones técnicas:
- Procesador: AMD Ryzen 5 7535HS
- Tarjeta gráfica: NVIDIA GeForce RTX 2050 (utilizada opcionalmente)
- Memoria RAM: 16 GB DDR5
- Almacenamiento: SSD NVMe 512 GB
- Sistema operativo: Windows 11 Pro
- Virtualización y contenedores: Docker Desktop con soporte para WSL2 -- Ubuntu

Contexto del Proyecto

Este proyecto fue creado con el propósito de implementar un sistema de RAG (Retrieval-Augmented Generation) que permita procesar documentos PDF y responder preguntas sobre su contenido de forma automatizada.
La motivación principal es contar con una herramienta académica que facilite la consulta de información de documentos extensos sin necesidad de leerlos completamente, haciendo uso de modelos de lenguaje localizados e integraciones automatizadas.

Desarrollo del Proyecto

A continuación, se describe el paso a paso del desarrollo del sistema RAG:

- Preparación del entorno: instalación de Docker, Python, n8n, Ollama y dependencias.
- Instalación del modelo local: se usó DeepSeek R1 1.5B mediante Ollama.
- Lectura de documentos: se habilitó un sistema para cargar PDFs y dividirlos en fragmentos.
- Generación de embeddings: usando "sentence-transformer´s", se convirtieron los fragmentos a vectores.
- Almacenamiento vectorial: se integró Qdrant como base de datos para los vectores.
- Backend de consulta: se montó un servidor FastAPI para realizar preguntas y obtener respuestas.
- Automatización: con n8n se conectaron los servicios para un flujo automático desde el PDF hasta la respuesta.

Definiciones y Conceptos Clave

- RAG (Retrieval-Augmented Generation): Técnica que combina recuperación de información y generación de texto.
- Embedding: Representación numérica de un texto para facilitar su comparación o búsqueda.
- PDF chunking: Proceso de dividir un PDF en fragmentos de texto más pequeños.
- Qdrant: Base de datos vectorial especializada para búsquedas semánticas.
- Ollama: Plataforma que permite ejecutar modelos de lenguaje grandes de forma local.
- sentence-transformers: Librería para generar embeddings semánticos en Python.
- n8n: Herramienta de automatización de flujos de trabajo similar a Zapier, pero open source.
- FastAPI: Framework web para construir APIs rápidas y eficientes en Python.
- Docker: Plataforma para ejecutar aplicaciones en contenedores, asegurando portabilidad y aislamiento.