Implementación 2 - Flowise:

Sumario

Instalación

Ollama:

- Instalar Ollama: https://ollama.com
- Probar Ollama: ejecutar en terminal *ollama run llama3.2:3b*
- Salir con *Ctrl+d* o */bye*
- Probar funcionamiento en http://localhost:11434
- Si queremos una interfaz web para Ollama: Open WebUI https://openwebui.com
    - **Atención al puerto que abrimos, el 3000 dará problemas con Flowise**

Flowise:

- Instalar Node.js: https://nodejs.org/en
- Instalar Flowise: ejecutar en terminal *npx flowise start*
- Probar funcionamiento en http://localhost:3000

Instalación alternativa con docker:

Flowise y Ollama están disponibles en docker, y esto facilita su instalación. Como contrapartida, pueden surgir problemas al comunicar los distintos contenedores, y el rendimiento, o acceso a recursos como GPU puede ser inferior.

Crear chatbot sencillo contra Ollama

- Crear nuevo chatflow
- Añadir nodo Chains ==> Conversation Chain
- Añadir nodo Chat Models ==> ChatOllama
    - Base Path: http://127.0.0.1:11434
    - Model name: llama3.2:3b
    - Additional Parameters: Streaming = true
- Añadir nodo Memory ==> Buffer Memory
    - Additional Parameters: Memory Key = chat_history
- Conectar los nodos, guardar el chatflow y probar

Crear chatbot con RAG contra Ollama

- Crear nuevo Document Store
- Add Document Loader ==> File Loader
    - Select Text Splitter ==> Recursive Character Text Splitter
    - Upload File(s)
    - Process
    - More Actions == Upsert All Chunks
        - Select Embeddings ==> Ollama Embeddings
            - Base URL: http://localhost:11434
            - Model Name: nomic-embed-text
            - Use MMap = True
        - Select Vector Store: Faiss
            - Base Path to load: Carpeta local
        - Save Config
        - Upsert
- Crear nuevo chatflow
- Añadir nodo Chains ==> Conversational Retrieval QA Chain
- Añadir nodo Chat Models ==> ChatOllama
    - Base Path: http://127.0.0.1:11434
    - Model name: llama3.2:3b
    - Additional Parameters: Streaming = true
- Añadir nodo Memory ==> Buffer Memory
    - Additional Parameters: Memory Key = chat_history
- Añadir nodo Vector Stores ==> Document Store (Vector)
- Conectar los nodos, guardar el chatflow y probar

Crear agente sencillo contra Ollama

- Crear nueva credencial ==> Serp API
- Crear nuevo chatflow a partir de Marketplaces ==> Conversational Agent
- Use Template
- Añadir nodo Chains ==> Conversation Chain
- Cambiar nodo ChatOpenAI ==> ChatOllama
    - Base Path: http://127.0.0.1:11434
    - Model name: llama3.2:3b
    - Additional Parameters:
        - Streaming = false
        - JSON Mode = true
- Añadir credencial al nodo Serp API
- Conectar los nodos, guardar el chatflow y probar

Crear chatbot sencillo contra GPT4All

- Crear nuevo chatflow
- Añadir nodo Chains ==> LLM Chain
- Añadir nodo Chat Models ==> ChatLocalAI
    - Base Path: http://127.0.0.1:4891/v1
    - Model name: Llama 3.2 3B Instruct
    - Additional Parameters: Max Tokens: 2048
- Añadir nodo Prompts ==> Chat Prompt Template
    - System Message: "You are a helpful teacher at a Data Science Bootcamp, and you will answer the student's question, or say you don't know if you can't find the answer."
    - Human Message: {question}
    - Format prompt values: **{**

      1 item

      question:

      "{{question}}"

      }
- Conectar los nodos, guardar el chatflow y probar