Apuntes de topicos

jueves, 6 de noviembre de 2025 00:52

Paso 1: Instalar Ollama en tu Computadora
Primero, necesitas instalar el programa Ollama. Este se encargará de gestionar y servir los modelos de lenguaje en tu máquina.

1. Ve a la página oficial de descargas de Ollama: https://ollama.com/download
2. Descarga el instalador para tu sistema operativo (Windows, macOS o Linux) y sigue las instrucciones. El README.md del proyecto también incluye los comandos para instalarlo usando winget (Windows) o brew (macOS). [README.md]
    - **Para Windows (usando winget en PowerShell):**

      powershell
      ```
      winget install Ollama.Ollama
      ```

Paso 2: Descargar un Modelo de Lenguaje
Una vez instalado Ollama, necesitas descargar un modelo para trabajar. El proyecto recomienda llama3.1, que es potente y versátil.

1. Abre una terminal (PowerShell, CMD o Terminal de macOS).
2. Ejecuta el siguiente comando. Esto descargará el modelo (puede tardar unos minutos dependiendo de tu conexión) y luego iniciará una sesión de chat para probarlo.

   bash
   ```
   ollama run llama3.1
   ```
3. Cuando veas el mensaje >>> Send a message (/? for help), significa que el modelo está funcionando. Puedes hacerle una pregunta como "Hola, ¿quién eres?" para probarlo.
4. Para salir de este chat, escribe /bye y presiona Enter.

Paso 3: Dejar Corriendo el Servidor de Ollama
Tu aplicación NestJS necesita conectarse al servidor de Ollama, que debe estar activo en segundo plano.

1. En la **misma terminal** que usaste antes, ejecuta:

   bash
   ```
   ollama serve
   ```
2. Verás algunos logs, pero la terminal se quedará "ocupada". **¡No cierres esta terminal!** Déjala abierta. Este es el motor de tu LLM local. Por defecto, estará escuchando en http://localhost:11434. [chat.service.ts]

Paso 4: Configurar tu Proyecto NestJS para Usar Ollama
Ahora, vamos a configurar tu código para que sepa que debe hablar con Ollama.

1. Abre una **nueva terminal** (recuerda, la otra está ocupada con ollama serve).
2. Navega al directorio de tu backend:

   bash
   ```
   cd"d:\practicas avanzadas\clases topicos\5 de noviembre\topicos-2025-02-project-2\examples\backend-nodejs"
   ```
3. Si aún no lo has hecho, instala las dependencias del proyecto:

   bash
   ```
   npm install
   ```
4. Crea el archivo de configuración .env a partir del ejemplo:

   powershell
   ```
   Copy-Item.env.example .env
   ```
5. Abre el nuevo archivo .env en tu editor de código (VS Code) y modifícalo para que se vea **exactamente** así:

   env
   ```
   # Proveedor de LLM a utilizar. Debe ser 'ollama'.
   LLM_PROVIDER=ollama
   ```

```
# Modelo local de Ollama que descargaste.
OLLAMA_MODEL=llama3.1

# --- No necesitas las variables de OpenAI, puedes borrarlas o dejarlas
comentadas ---
# OPENAI_API_KEY=
# OPENAI_MODEL=
```

Esto le dice al ChatService que debe usar la lógica de callOllama en lugar de callOpenAI. [chat.service.ts]

Paso 5: Ejecutar y Probar tu Backend

Con el servidor de Ollama corriendo en una terminal y tu proyecto configurado, es hora de iniciar tu aplicación.

1. En la **segunda terminal** (la que está en la carpeta backend-nodejs), ejecuta el comando para iniciar tu servidor NestJS:

   bash
   ```
   npm run start:dev
   ```
2. Espera a que aparezca el mensaje [Nest] Nest application successfully started.
3. ¡Es hora de la prueba final! Abre una **tercera terminal** y envía una petición curl a tu backend. Tu backend, a su vez, le preguntará a Ollama.

   bash
   ```
   curl -X POST http://localhost:3000/chat -H "Content-Type: application/json"-
   d '{"message": "Hola, ¿puedes recomendarme un buen libro de ciencia
   ficción?"}'
   ```

Resultado Esperado

Si todo salió bien, deberías ver una respuesta en formato JSON directamente en tu terminal, generada por el modelo llama3.1 que corre en tu máquina. La respuesta se verá similar a esto:

json

```
{"reply":"¡Claro! Un clásico que nunca falla es \"Dune\" de Frank Herbert. Es una
obra maestra de la ciencia ficción con un universo increíblemente detallado,
intriga política y personajes memorables. Si buscas algo más moderno, te podría
recomendar \"El problema de los tres cuerpos\" de Cixin Liu. ¿Te interesa alguno
de estos?"}
```


¡Felicidades! Has configurado y probado con éxito tu entorno de desarrollo local completo usando NestJS y Ollama. Ahora tienes dos terminales activas que son clave para tu desarrollo:

1. **Terminal 1:** ollama serve (El motor del LLM).
2. **Terminal 2:** npm run start:dev (Tu aplicación de backend).