OpenWebUI + Ollama による日本語Web検索付き生成AIチャット構築手順

🧩 構成概要

本プロジェクトでは、Qiita記事「OpenWebUI + OllamaによるローカルLLM実行環境構築」を参考に、以下を目標として環境を一から構築する。

🎯 目的

- ローカルLLM（例：LLaMA3, Nekomata）を用いた日本語対応生成AIチャットの実装
- Web検索機能を組み込んだRAG風チャット体験の実現

🏗 使用技術スタック

項目	内容
チャットUI	OpenWebUI（Docker）
LLMエンジン	Ollama（ローカル推論）
日本語対応モデル	llama3, nekomata, nous-hermes2-jp など
Web検索連携	外部検索APIまたはLangChainツール
OS想定	Windows 10/11 or Linux（WSL対応）

☑ ステップ1：OpenWebUI + Ollama の環境構築

1-1. Ollamaとは？

Ollamaはローカル環境で大規模言語モデル（LLM）を手軽に実行できる軽量な実行エンジンです。

- GPU・CPU両対応（GPUがあれば高速）
- モデルを `ollama pull モデル名` で取得して実行可能
- CLIやREST APIで操作可能

公式サイト：https://ollama.com

1-2. Ollamaのインストール

```
# Mac/Linux
curl -fsSL https://ollama.com/install.sh | sh

# Windows（WSL推奨）
wget https://ollama.com/download/OllamaSetup.exe
# または公式から手動ダウンロード：https://ollama.com/download
```

1-3. モデルの取得（日本語モデル）

```
# llama3 モデルの取得 ( 初回 )
ollama pull llama3

# 日本語対応モデル（任意で切替 )
ollama pull gemma3:4b
```

1-4. モデル動作確認（CLIによる検証）

モデルが正しく動作するかをCLIで確認：

```
ollama run gemma3:4b
```

プロンプトが表示されたら、日本語で入力してみる：

```
C:\Users\kura>ollama run gemma3:4b
>>> Send a message (/? for help)
```

問題なく応答が返ってくれば、モデルは正常に動作しています。 Ctrl + d（または /bye または /exit）で終了。

1-5. Dockerとは？

Dockerは、アプリケーションを軽量なコンテナ（仮想環境）として実行するためのプラットフォームです。

☑ **特徴：**

- OSやライブラリ依存を気にせず、どこでも同じ動作を保証
- `Dockerfile` や `docker-compose.yml` で環境構成を定義可能
- ローカルでもクラウドでも同一の実行環境を再現可能

1-6. docker-compose によるOpenWebUIのコンテナ定義と起動

以下の内容で `docker-compose.yml` ファイルを作成します：

```
version: '3.8'
services:
  openwebui:
    image: ghcr.io/open-webui/open-webui:main
    container_name: openwebui
    ports:
      - "3000:8080"
    volumes:
      - openwebui-data:/app/backend/data
```

```
    environment:
      - OLLAMA_BASE_URL=http://host.docker.internal:11434
    restart: unless-stopped

volumes:
  openwebui-data:
```

🔍 コンテナ定義の構文解説：

項目	説明
`version`	Composeファイルのバージョン（推奨：3.8）
`services`	実行するコンテナ群を定義するブロック
`image`	使用するDockerイメージ（OpenWebUIの公式）
`container_name`	任意のコンテナ名を指定（openwebui）
`ports`	ホストとコンテナのポートマッピング（例：3000）
`volumes`	永続化するデータボリュームの設定（チャット履歴など）
`environment`	コンテナ内で使用される環境変数（OLLAMAのURL）
`restart`	コンテナの再起動方針（unless-stopped: 停止しない限り再起動）

1-7. コンテナの起動

以下のコマンドでOpenWebUIのDockerコンテナを起動：

```
docker compose up -d
```

起動後、ブラウザで以下にアクセス：

```
http://localhost:3000
```

初回起動時はアカウント作成画面が表示されます。

☑ ステップ2：OpenWebUI上で日本語モデルの検証

2-1. OpenWebUIにアクセス

```
http://localhost:3000
```

ブラウザからアクセスし、初回はアカウントを作成する。

2-2. モデルの切り替えと日本語動作確認

- 「Model」設定から `llama3`, `nekomata`, `nous-hermes2` を選択
- 日本語で以下のようなプロンプトを入力して検証：
    - 「中京大学について教えてください」
    - 「生成AIとは何ですか？」

2-3. 応答品質の比較メモ

モデル	日本語精度	応答速度	コメント
llama3	△（一部英語混じり）	◎	英語ベース、文脈は強い
nekomata	◎	○	日本語特化、安定動作
nous-hermes2	○	○	会話風に強いが一部冗長

🚧 次ステップ（予定）

次は以下の機能を追加予定：

- Web検索機能の組み込み（SerpAPI or LangChain）
- 検索結果とLLMを組み合わせたRAG風応答の実装
- 日本語における情報正確性の評価

➡ 続きは次回、ステップ3として追記予定