

Open-WebUI

Interface IA conviviale (prend en charge Ollama, l'API OpenAI, ...)

- [Installation avec pip](#)
- [Installation avec docker](#)
- [Mise à jour](#)

Installation avec pip

Source : [GitHub - open-webui/open-webui: User-friendly AI Interface \(Supports Ollama, OpenAI API, ...\)](#)

ATTENTION !

L'installation avec `pip` rend la mise à jour du logiciel complexe. Faites cette installation pour des tests, ou du développement spécifique et non une utilisation en production.

Installation via Python pip

Open WebUI peut être installé via pip, l'installateur de paquets Python. Avant de procéder, assurez-vous d'utiliser Python 3.11 pour éviter les problèmes de compatibilité.

Installer Open WebUI :

Ouvrez votre terminal et exécutez la commande suivante pour installer Open WebUI :

```
pip install open-webui
```

Exécuter Open WebUI :

Une fois l'installation terminée, vous pouvez démarrer Open WebUI en exécutant la commande suivante :

```
open-webui serve
```

Cela démarrera le serveur Open WebUI, que vous pourrez accéder à l'adresse suivante : <http://localhost:8080>.

Erreur possible :

Erreur lors de l'installation de `open-webui` via `pip` :

```
root@srv-ia:/home/ia# pip install open-webui
error: externally-managed-environment

× This environment is externally managed
↳ To install Python packages system-wide, try apt install
```

python3-xyz, where xyz is the package you are trying to install.

If you wish to install a non-Debian-packaged Python package, create a virtual environment using `python3 -m venv path/to/venv`. Then use `path/to/venv/bin/python` and `path/to/venv/bin/pip`. Make sure you have python3-full installed.

If you wish to install a non-Debian packaged Python application, it may be easiest to use `pipx install xyz`, which will manage a virtual environment for you. Make sure you have pipx installed.

See `/usr/share/doc/python3.11/README.venv` for more information.

note: If you believe this is a mistake, please contact your Python installation or OS distribution provider. You can override this, at the risk of breaking your Python installation or OS, by passing `--break-system-packages`.

hint: See PEP 668 for the detailed specification.

Utiliser un environnement virtuel

La meilleure pratique pour installer des paquets Python sans affecter l'environnement système est de créer un **environnement virtuel**. Cela permet d'avoir une installation isolée pour vos paquets Python.

Étapes pour créer un environnement virtuel :

1. **Installez** `python3-venv` (si ce n'est pas déjà fait) :

```
sudo apt install python3-venv
```

2. **Créez un environnement virtuel :**

Dans le répertoire où vous souhaitez installer vos paquets, créez un environnement virtuel :

```
python3 -m venv mon_venv
```

Cette commande crée un dossier `mon_venv` qui contient une installation isolée de Python.

3. **Activez l'environnement virtuel :**

```
source mon_venv/bin/activate
```

Vous verrez probablement un changement dans l'invite de commande, indiquant que l'environnement virtuel est activé (par exemple, `(mon_venv)` au début de la ligne).

4. Installez `open-webui` dans l'environnement virtuel :

Une fois l'environnement activé, vous pouvez installer `open-webui` comme d'habitude :

```
pip install open-webui
```

5. Lancez l'application avec `nohup` et redirigez la sortie vers un fichier de log, puis lancez-le en arrière-plan :

```
nohup open-webui serve > open-webui.log 2>&1 &
```

- `nohup` permet au processus de ne pas être tué lorsqu'une session se termine.
- `> open-webui.log 2>&1` redirige les sorties standard (stdout) et les erreurs (stderr) vers un fichier `open-webui.log`.
- `&` place le processus en arrière-plan.

Vous pouvez quitter la session SSH ou votre terminal normalement, l'application continuera de fonctionner.

Pour voir si le processus est toujours en cours d'exécution :

```
ps aux | grep open-webui
```

6. Désactivez l'environnement virtuel quand vous avez terminé :

```
deactivate
```

Cela vous ramènera à votre environnement système Python.

Installation avec docker

Avant de faire l'installation de Open-WebUI, merci de vous référer à l'installation de docker.

Installation avec la configuration par défaut

Si Ollama est sur votre ordinateur, utilisez cette commande :

```
docker run -d -p 3000:8080 --add-host=host.docker.internal:host-gateway -v open-webui:/app/backend/data --name open-webui --restart always ghcr.io/open-webui/open-webui:main
```

Si Ollama est sur un serveur différent, utilisez cette commande :

Pour vous connecter à Ollama sur un autre serveur, changez l'URL OLLAMA_BASE_URL par l'URL du serveur :

```
docker run -d -p 3000:8080 -e OLLAMA_BASE_URL=https://example.com -v open-webui:/app/backend/data --name open-webui --restart always ghcr.io/open-webui/open-webui:main
```

Pour exécuter Open WebUI avec le support GPU Nvidia, utilisez cette commande :

```
docker run -d -p 3000:8080 --gpus all --add-host=host.docker.internal:host-gateway -v open-webui:/app/backend/data --name open-webui --restart always ghcr.io/open-webui/open-webui:cuda
```

Installation pour une utilisation uniquement avec l'API OpenAI

Si vous utilisez uniquement l'API OpenAI, utilisez cette commande :

```
docker run -d -p 3000:8080 -e OPENAI_API_KEY=your_secret_key -v open-webui:/app/backend/data --name open-webui --restart always ghcr.io/open-webui/open-webui:main
```

Installation d'Open WebUI avec le support intégré d'Ollama

Cette méthode d'installation utilise une image de conteneur unique qui regroupe Open WebUI avec Ollama, permettant une configuration simplifiée avec une seule commande. Choisissez la commande appropriée en fonction de votre configuration matérielle :

Avec le support GPU : Utilisez les ressources GPU en exécutant la commande suivante :

```
docker run -d -p 3000:8080 --gpus=all -v ollama:/root/.ollama -v open-webui:/app/backend/data --name open-webui --restart always ghcr.io/open-webui/open-webui:ollama
```

Pour un usage uniquement avec le CPU : Si vous n'utilisez pas de GPU, utilisez cette commande à la place :

```
docker run -d -p 3000:8080 -v ollama:/root/.ollama -v open-webui:/app/backend/data --name open-webui --restart always ghcr.io/open-webui/open-webui:ollama
```

Les deux commandes facilitent une installation intégrée et sans tracas de Open WebUI et Ollama, garantissant une mise en place rapide de l'ensemble.

Après l'installation, vous pouvez accéder à Open WebUI à l'adresse <http://localhost:3000>. Profitez-en ! ☑☑

Erreur de connexion Docker

Si vous rencontrez une erreur de connexion lorsque vous essayez d'accéder à Ollama, cela peut être dû au fait que le conteneur WebUI Docker ne peut pas communiquer avec le serveur Ollama qui fonctionne sur votre hôte. Voici comment résoudre cela :

Ajustez les paramètres réseau ☑☑ : Utilisez le flag `--network=host` dans votre commande Docker. Cela relie directement votre conteneur au réseau de votre hôte.

Changez le port : N'oubliez pas que le port interne passe de 3000 à 8080.

Exemple de commande Docker :

```
docker run -d --network=host -v open-webui:/app/backend/data -e OLLAMA_BASE_URL=http://127.0.0.1:11434 --name open-webui --restart always ghcr.io/open-webui/open-webui:main
```

☑☑ Après avoir exécuté cette commande, votre WebUI devrait être disponible à l'adresse <http://localhost:8080>.

Mise à jour

Mise à jour vers Open WebUI tout en conservant vos données

Si vous souhaitez mettre à jour vers la nouvelle image tout en migrant tous vos paramètres précédents tels que les conversations, les invites, les documents, etc.

Vous pouvez effectuer les étapes suivantes :

```
docker rm -f ollama-webui
docker pull ghcr.io/open-webui/open-webui:main
# Crée un nouveau volume et utilise un conteneur temporaire pour copier d'un volume à un autre
selon https://github.com/moby/moby/issues/31154#issuecomment-360531460
docker volume create --name open-webui
docker run --rm -v ollama-webui:/from -v open-webui:/to alpine ash -c "cd /from ; cp -av . /to"
[insérez la commande équivalente que vous avez utilisée pour installer avec le nouveau nom de l'image Docker]
```

Une fois que vous avez vérifié que toutes les données ont été migrées, vous pouvez supprimer l'ancien volume avec la commande suivante :

```
docker volume rm ollama-webui
```

Passage depuis un dépôt Git local

Si vous veniez d'une installation Git où vous avez utilisé `docker compose up` dans le répertoire du projet, vos volumes auront un préfixe avec le nom du dossier. Par conséquent, si votre chemin OpenWebUI était : `/home/myserver/ollama-webui/`, les volumes seraient nommés "ollama-webui_open-webui" et "ollama-webui_ollama".

Pour copier le contenu vers une installation Docker conventionnelle, vous pouvez exécuter des commandes de migration similaires. **Dans notre cas, les commandes seraient les suivantes :**

```
docker rm -f open-webui
docker rm -f ollama
docker pull ghcr.io/open-webui/open-webui:main
docker pull ghcr.io/open-webui/open-webui:ollama
docker volume create --name open-webui
docker volume create --name ollama
docker run --rm -v ollama-webui_open-webui:/from -v open-webui:/to alpine ash -c "cd /from ;
```

```
cp -av . /to"
docker run --rm -v ollama-webui_ollama:/from -v ollama:/to alpine ash -c "cd /from ; cp -av .
/to"
```

Selon que vous ayez installé Ollama ou que vous ayez déjà les mêmes noms de volumes en place, certaines des commandes pourraient générer des erreurs, mais elles peuvent généralement être ignorées en toute sécurité car nous procédons à un écrasement.

Ensuite, lancez les deux conteneurs comme d'habitude, comme décrit dans le guide de démarrage :

```
docker run -d -p 3000:8080 --add-host=host.docker.internal:host-gateway -v open-
webui:/app/backend/data --name open-webui --restart always ghcr.io/open-webui/open-webui:main
```

Une fois que vous avez vérifié que toutes les données ont été migrées, vous pouvez supprimer l'ancien volume avec la commande `docker volume rm` mentionnée ci-dessus.