

Introducción a Docker

UD 06. Caso práctico

04 - Whisper AI y Web

Whisper



Autor: Sergi García Barea

Actualizado Febrero 2025

Licencia





Reconocimiento – NoComercial - CompartirIgual (BY-NC-SA): No se permite un uso comercial de la obra original ni de las posibles obras derivadas, la distribución de las cuales se debe hacer con una licencia igual a la que regula la obra original.

Nomenclatura

A lo largo de este tema se utilizarán distintos símbolos para distinguir elementos importantes dentro del contenido. Estos símbolos son:

 **Importante**

 **Atención**

 **Interesante**

1. Introducción	2
2. Pasos a previos para poner en marcha Whisper AI y Whisper Web	3
3. Poniendo en marcha Whisper AI y Whisper Web	3
4. Bibliografía	3

UD06. CASO PRÁCTICO 04

1. INTRODUCCIÓN

Whisper AI es un software que permite transcribir audio a Texto. Este software fue creado por OpenAI (empresa creadora de ChatGPT) y liberado junto con varios modelos de procesamiento de lenguaje muy precisos. Actualmente, es una de las mejores soluciones “self-hosted” para convertir audio a texto.

Whisper Web es una web, que proporciona una interfaz gráfica web para mandar audios a transcribir a Whisper AI. Su repositorio es <https://codeberg.org/pluja/web-whisper>

En este caso práctico vamos a poner en marcha una versión Dockerizada extraído de su repositorio siguiendo las instrucciones de <https://codeberg.org/pluja/web-whisper/wiki/Self-Hosting>

2. PASOS A PREVIOS PARA PONER EN MARCHA WHISPER AI Y WHISPER WEB

En el propio sitio web, nos piden que clonemos el repositorio con el comando:

```
git clone https://codeberg.org/pluja/web-whisper
```

Nota: en el curso os proporcionamos el resultado de clonar este repositorio en un fichero .zip.

Tras ello, copiamos el fichero de ejemplo a un “docker-compose.yml” con un comando similar a:

```
cp example.docker-compose.yml docker-compose.yml
```

Tras ello, también copiaremos y renombraremos el fichero “.env” de forma similar a:

```
cp example.env .env
```

Para adaptar la configuración, deberemos adaptar el fichero “.env”. Como adaptarlo se detalla en

<https://codeberg.org/pluja/web-whisper/wiki/Self-Hosting#user-content-configuration>

Pero con los parámetros con defecto, podemos tener un funcionamiento correcto.

3. PONIENDO EN MARCHA WHISPER AI Y WHISPER WEB

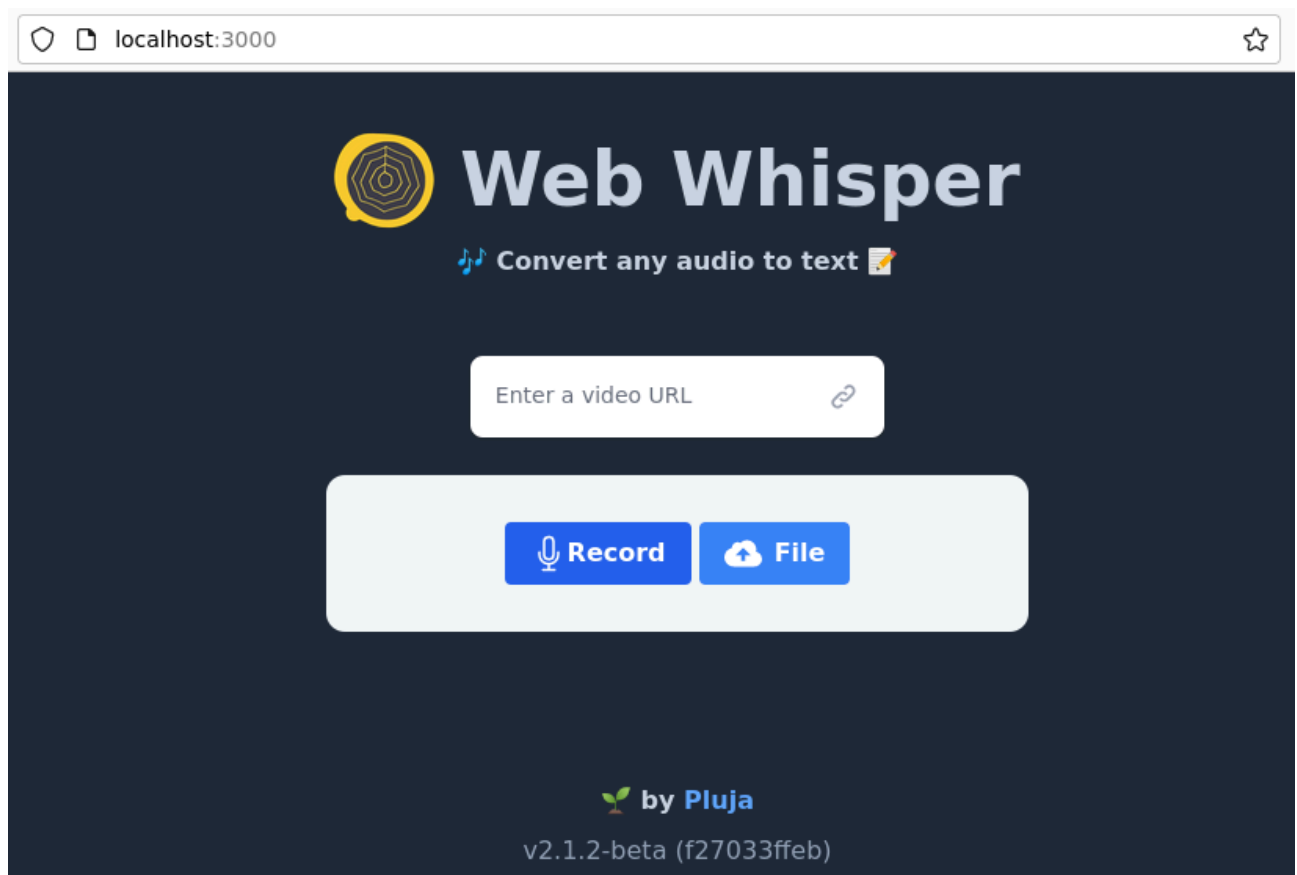
Una vez preparados los ficheros “docker-compose.yml” y “.env”, ejecutaremos:

```
docker compose up --build -d
```

Obteniendo algo similar a:

```
sergi@casa:~/Desktop/web-whisper$ docker compose up --build -d
[+] Building 0.0s (0/2)
[+] Building 57.3s (18/57)
=> => extracting sha256:90648ba22c191b969d8db393bf6138e7d6ecc1b70b3865d1717ec4d663192d89 0.6s
=> [web-whisper-backend whisper_builder 1/12] FROM docker.io/library/gcc:bullseye@sha256:c442526820a5e458c95785da89 51.9s
=> => resolve docker.io/library/gcc:bullseye@sha256:c442526820a5e458c95785da89da395e56630fd7ba51d8ab39f7eb8647b25ca2 1.2s
=> => sha256:c442526820a5e458c95785da89da395e56630fd7ba51d8ab39f7eb8647b25ca2 1.43kB / 1.43kB 0.0s
=> => sha256:5a5d5b051dee6bbcd3de3c91a994546c663976cdceff57f13ce7f78b051d5747 2.22kB / 2.22kB 0.0s
=> => sha256:c6aa7ca27d67e57111f305f696dd6c034312559a99fe45813bf39ccae1f0f91b 9.18kB / 9.18kB 0.0s
=> => sha256:5d25d039b6b6a3a0923cc617e1a7afa88767ad65a7f2f98b55ae58e3d203e092 16.14kB / 16.14kB 10.9s
=> => extracting sha256:5d25d039b6b6a3a0923cc617e1a7afa88767ad65a7f2f98b55ae58e3d203e092 0.0s
=> => sha256:0d0b93d00653ff9600d5aa73ad664f61daf21247989a37d7656a8960b41d1710 65.01MB / 138.30MB 50.1s
```

Cuando haya finalizado, tendremos Whisper Web con Whisper AI funcionando en <http://localhost:3000> obteniendo algo similar a:



4. BIBLIOGRAFÍA

- [1] Docker Docs <https://docs.docker.com/>
- [2] Docker Compose Docs <https://docs.docker.com/compose/>