Guide complet pour créer et lancer un projet Django avec Docker

Ce guide montre comment créer un projet Django directement dans un conteneur Docker, pour garantir un environnement isolé, cohérent et facilement déployable.

Étape 1 : Créer le Dockerfile

Ici, on part d'une image Python Alpine légère, on installe bash (utile pour scripts), on crée un dossier de travail /app, on copie la liste des dépendances et on installe un environnement virtuel Python avec les dépendances, on copie ensuite tout le code de notre projet dans l'image, et enfin on définit le shell comme commande par défaut.

```
FROM python:3.9-alpine
RUN apk add --no-cache bash
WORKDIR /app
COPY requirements.txt /tmp/
RUN python -m venv /py && /py/bin/pip install --upgrade pip && /py/bin/pip
install -r /tmp/requirements.txt
ENV PATH="/py/bin:\$PATH"
COPY . .
CMD ["sh"]
EOF
```

Étape 2 : Créer requirements.txt

Ce fichier liste les dépendances Python, ici juste Django, pour que Docker puisse les installer dans l'environnement virtuel.

Étape 3 : Créer docker-compose.yml

Ce fichier décrit comment lancer le service app :

- build : construire l'image Docker depuis le Dockerfile,
- volumes : monter le dossier local dans le conteneur pour synchroniser le code,
- ports : exposer le port 8000 du conteneur sur le port 8000 local,
- command : démarrer un shell qui tourne en boucle (permet de garder le conteneur vivant).

```
version: "3.9"
services:
  app:
    build: .
```

```
    volumes:
      - .:/app
    ports:
      - "8000:8000"
    command: sh -c "while true; do sleep 1000; done"
```

Étape 4 : Construire l'image Docker

Compile l'image Docker à partir du Dockerfile, télécharge la base Python, installe les dépendances, prépare tout l'environnement isolé.

```
docker compose build
```

Étape 5 : Créer le projet Django dans le conteneur

Exécute temporairement un conteneur basé sur l'image créée,et lance la commande Django pour créer un nouveau projet dans le dossier monté, ce qui initialise la structure Django dans votre dossier local.

```
docker compose run --rm app sh -c "django-admin startproject app ."
```

Étape 6 : Lancer le conteneur en mode détaché (background)

Démarre le service app défini dans docker-compose.yml,avec le code synchronisé, prêt à faire tourner Django.

```
docker compose up -d
```

Étape 7 : Démarrer le serveur Django à l'intérieur du conteneur

Se connecte au conteneur en cours d'exécution et lance le serveur de développement Django, accessible sur toutes les interfaces réseau (0.0.0.0) pour que l'hôte local puisse s'y connecter.

```
docker compose exec app sh -c "python manage.py runserver 0.0.0.0:8000"
```

Étape 8 : Accéder à votre application Django

Ouvrez votre navigateur à l'adresse http://localhost:8000. Vous verrez la page d'accueil de votre projet Django en marche,prouvant que tout fonctionne dans Docker !

Pour lancer exécuter une action dans notre conteneur Django, il suffit d'utiliser

```
docker compose run --rm app sh -c
```

Par exemple, pour créer une nouvelle application Django on peut juste mettre

```
docker compose run --rm app sh -c "python manage.py startapp core"
```