**Cas pratique**

**Déployer un environnement docker avec docker file et docker compose avec Django Python**

Prérequis :

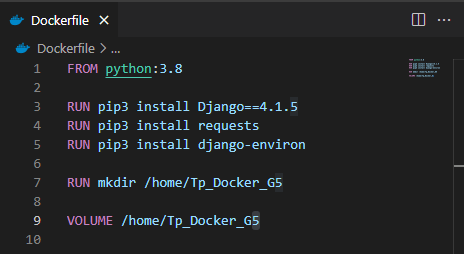
* Lancer Docker desktop
* Ouvrir un compte Docker Hub
* Installer python3 dans votre machine
* Libérer le port 80 de votre machine pour permettre à Docker de tourner

**Step - 1** Créer un dossier pour notre projet

**Step - 2** ouvrir le dossier avec un éditeur (Visual Code)

**Step - 3** Créer un fichier Dockerfile en respectant la syntaxe

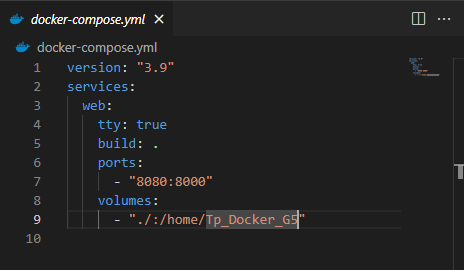
**Step - 4** Editer notre Dockerfile en créant notre image personnalisée à savoir un environnement docker qui va contenir notre application web Django python



* **From Python** c’est l’image de base d’où on va créer notre image personnalisé il se trouve dans le hub
* **RUN pip3 install Django==4.1.5** : permet d’installer Django 4.1.5 (il faut toujours avoir la reflexe de préciser les versions des applications
* Et on procède pour la même chose avec en installant les modules requests et Django-environ pour installer respectivement les requêtes et l’environnement Django
* **RUN mkdir /home/Tp\_Docker\_G5** : crée un dossier Tp\_Docker\_G5 dans le répertoire home dans notre conteneur
* **VOLUME /home/Tp\_Docker\_G5** : permet de se placer dans le dossier /home/Tp\_Docker\_G5

**Step - 5** on crée notre fichier docker-compose.yml

**Step - 6** On édite notre fichier docker-compose.yml



* On précise la version de notre docker-compose
* Apres on va lister nos services
* Par convention si c’est une application web on nomme cette service web mais on pouvait l’appeler autrement
* Tty : true permet de faire tourner notre conteneur continuellement
* Build . : permet de construire notre conteneur basé sur l’image qui se trouve dans notre Dockerfile
* Ports : - ‘’8080 :800’’ permet de mapper notre port de notre machine au port 8000 du conteneur qu’on va executer
* Volumes : - ‘’./:/home/Tp\_Docker\_G5’’ permet de faire persister les données de notre repertoire principal ./ au repertoire /home/Tp\_Docker\_G5

**Step - 7** on ouvre le terminal puis on se positionne dans notre dossier du projet ensuite on lance la commande docker compose up pour builder notre image et démarrer notre conteneur

**Step - 8** Apres build on peut aller vérifier dans notre docker desktop notre image et le conteneur qui est en train de tourner

**Step - 9** Ensuite on se connecte au terminal du conteneur puis se placer dans notre dossier. On vérifie notre version de python qu’on avait défini dans notre Dockerfile

**Step - 10** Maintenant on peut créer tranquillement notre projet Django avec la commande **django-admin startproject** testgroupe5

Apres avoir créer notre projet dans le terminal de notre conteneur on peut voir notre projet dans notre dossier grâce au mappage qu’on avait définit dans notre fichier docker compose.

**Step - 11** On se positionne dans notre dossier de projet qu’on vient de créer (testgroupe5) On tape la commande python3 manage.py runserver 0.0.0.0 :8000 pour démarrer notre application Django sur le port 8000 du conteneur

**Step - 12** on ouvre notre browser localhost :8080 et on verra notre application tourner

**Step – 13** On déploie notre image docker sur le docker hub

* Tout d’abord on crée le lien entre notre docker hub et notre environnement local en tappant la commande **docker tag docker\_tp\_docker\_G5-web karim2801/testgroupe5**
  + Docker\_tp\_groupe5-web c’est le nom de notre image
  + Karim2801 c’est l’identifiant de mon docker hub
  + Testgroupe5 c’est le repository que j’ai créé
* Ensuite on tape la commande **docker push karim2801/testgroupe5** pour déposer notre image sur le hub

Maintenant on peut aller vérifier notre image dans notre docker hub