LAB 4 - Gestion du réseau et des données pour les conteneurs

Partie 1 : Mappage des ports

Exemple 1 : Créez un conteneur en se basant sur la description suivante :

- Image : nodejscn/node
- Port à publier :
  - Sur le host : 8080
  - Dans le conteneur : 80
- Nom du conteneur : testAppNode

Exemple 2 : Créez un conteneur basé sur l'image jenkins:2.60.3 en mappant les port 8080 et 50000.

Exemple 3 : Depuis le docker hub, cherchez une image correspondante à la version 7.9 du **sonarqube** et créez un conteneur pour mapper les ports nécessaires.

Partie 2 : Persistance des données

Exemple 1 : Définition d'un volume dans un Dockerfile

- Créez le Dockerfile suivant

```
FROM alpine:3.8
VOLUME ["/data"]
```

- Construisez l'image imgvol à partir de ce Dockerfile
- lancez un shell interactif dans un container, nommé c2, basé sur l'image imgvol.
- créez le fichier /data/hello.txt dans le conteneur
- inspectez pour récupérer la clé **`Mounts`** afin d'avoir le chemin d'accès du volume sur la machine hôte.
- Supprimez le conteneur créé
- Vérifiez l'existence du fichier hello.txt

Exemple 2 : Définition d'un volume au lancement d'un container

- Lancez un container avec les caractéristiques suivantes:
  - basé sur l'image alpine:3.8
  - nommé c3
  - exécution en background (option -d)
  - définition de /data en tant que volume (option -v)
  - spécification d'une commande qui écrit dans le volume ci-dessus
- Inspectez le container et repérez notamment le chemin d'accès du volume sur la machine hôte.
- Supprimez le container c3.

Exemple 3 : Utilisation des volumes via la CLI

- Créez un volume nommé html
- Listez les volumes existants et vérifiez que le volume html est présent
- Inspectez le volume créé
- Lancer un container basé sur nginx et monter le volume html sur le point de montage /usr/share/nginx/html du container. En publiant également le port 80 du container sur le port 8080 de l'hôte.
- Affichez le contenu du volume html.