

# Guide de dépannage pour la Toolbox

## Introduction

Ce guide de dépannage présente les problèmes courants que vous pourriez rencontrer lors de l'utilisation de la Toolbox et leurs solutions. Il couvre l'installation, la configuration, le démarrage/arrêt des conteneurs, et la gestion des données persistantes.

## Table des matières

1. [Démarrage et arrêt des conteneurs](#)
2. [Gestion des données persistantes](#)
3. [Problèmes courants avec Keycloak](#)
4. [Problèmes courants avec PostgreSQL et pgAdmin](#)
5. [Problèmes courants avec Portainer](#)
6. [Erreurs de permission](#)
7. [Résolution des conflits de ports](#)

## Démarrage et arrêt des conteneurs

### Démarrage des conteneurs

Pour démarrer tous les conteneurs et l'application Flask:

```
bash  
poetry run bash ./start_all.sh
```

Si vous rencontrez une erreur de permission:

```
bash  
chmod +x start_all.sh  
poetry run bash ./start_all.sh
```

Alternativement, vous pouvez utiliser directement Docker Compose pour les conteneurs seuls:

```
bash  
docker-compose up -d
```

Puis démarrer l'application Flask séparément avec Poetry:

```
bash
```

```
poetry run python main.py
```

## Arrêt des conteneurs

Pour arrêter tous les conteneurs:

```
bash
```

```
./stop_all.sh
```

Si vous rencontrez une erreur de permission:

```
bash
```

```
chmod +x stop_all.sh
```

```
./stop_all.sh
```

Alternativement, vous pouvez utiliser directement Docker Compose:

```
bash
```

```
docker-compose down
```

## Vérification des conteneurs en cours d'exécution

Pour vérifier que tous les conteneurs sont en cours d'exécution:

```
bash
```

```
docker ps
```

## Gestion des données persistantes

### Comprendre les volumes Docker


Les données persistantes sont stockées dans des volumes Docker qui sont mappés aux répertoires locaux. Ces dossiers sont créés automatiquement lors du premier démarrage des conteneurs.

### Dossiers créés automatiquement

- `./postgresql/postgres_data`: Stocke les données PostgreSQL
- `./postgresql/pgadmin_volume`: Stocke les configurations de pgAdmin
- `./keycloak/keycloak-data`: Stocke les données Keycloak
- `./portainer/portainer_data`: Stocke les données Portainer

- `./rapports`: Dossier partagé pour les rapports (utilisé par plusieurs services)

## Attention aux permissions

 **Important:** Les dossiers créés par les conteneurs Docker peuvent avoir des propriétaires et des permissions spécifiques qui les rendent difficiles à supprimer. Par exemple:

```
bash
drwx----- 19 70 root 4096 May 17 14:44 postgres_data
drwxrwxr-x 5 5050 5050 4096 May 17 14:52 pgadmin_volume
```


Ces dossiers sont créés avec des utilisateurs spécifiques à Docker (ID 70 pour postgres, 5050 pour pgAdmin) et peuvent nécessiter des privilèges root pour être supprimés.

## Solution pour supprimer les dossiers avec permissions spéciales

Si vous devez supprimer ces dossiers (par exemple pour un nettoyage complet):

```
bash
sudo rm -rf ./postgresql/postgres_data
sudo rm -rf ./postgresql/pgadmin_volume
```

## Problème de GitHub et données sensibles

 **Attention:** Ces dossiers contiennent des données sensibles et ne doivent **PAS** être ajoutés à Git ou GitHub. Assurez-vous qu'ils sont listés dans votre fichier `.gitignore`.

Exemple de `.gitignore`:

```
# Données persistantes Docker
postgresql/postgres_data/
postgresql/pgadmin_volume/
keycloak/keycloak-data/
portainer/portainer_data/
rapports/
```

## Problèmes courants avec Keycloak

### Échec du démarrage de Keycloak

Si Keycloak ne démarre pas correctement:

1. Vérifiez les logs:

bash

`docker logs keycloak`

## 2. Problèmes courants:

- Problèmes de permissions sur le volume des données
- Conflit de port (8080 déjà utilisé)
- Problèmes de mémoire insuffisante

## 3. Solutions:

- Recréer le dossier de données: `sudo rm -rf ./keycloak/keycloak-data && mkdir -p ./keycloak/keycloak-data`
- Vérifier si le port 8080 est déjà utilisé: `sudo lsof -i :8080`
- Augmenter la mémoire disponible dans les paramètres Docker

# Problèmes courants avec PostgreSQL et pgAdmin

## PostgreSQL ne démarre pas

Si PostgreSQL ne démarre pas:

### 1. Vérifiez les logs:

bash

`docker logs toolbox_postgres`

## 2. Problèmes courants:

- Corruption des données
- Problèmes de permissions sur le volume des données
- Conflit de port (5432 déjà utilisé)

## 3. Solutions:

- Recréer le dossier de données: `sudo rm -rf ./postgresql/postgres_data && mkdir -p ./postgresql/postgres_data`
- Vérifier si le port 5432 est déjà utilisé: `sudo lsof -i :5432`

## pgAdmin n'affiche pas les serveurs

Si vous ne voyez pas vos serveurs dans pgAdmin:

1. Vérifiez que PostgreSQL est bien démarré
2. Essayez d'ajouter manuellement une connexion au serveur avec ces paramètres:

- Nom: Toolbox PostgreSQL
- Hôte: postgres (nom du service dans docker-compose)
- Port: 5432
- Base de données: toolbox\_db
- Utilisateur: toolbox\_user
- Mot de passe: secure\_password

## Dossiers créés par PostgreSQL et pgAdmin

⚠ **Important:** Lorsque vous connectez PostgreSQL via pgAdmin, des dossiers de données persistantes sont automatiquement créés:

```
./postgresql/postgres_data ... # Données PostgreSQL (propriétaire: ID 70, permissions: 700)
./postgresql/pgadmin_volume ... # Données pgAdmin (propriétaire: ID 5050, permissions: 755)
```

Ces dossiers:

- Sont nécessaires au bon fonctionnement des services
- Ne peuvent pas être supprimés sans privilèges root
- Ne doivent pas être ajoutés à Git/GitHub car ils contiennent des données sensibles

## Problèmes courants avec Portainer

### Échec du démarrage de Portainer

Si Portainer ne démarre pas:

1. Vérifiez les logs:

```
bash
```

```
docker logs portainer
```

2. Problèmes courants:

- Problèmes d'accès au socket Docker
- Conflit de port (9000 déjà utilisé)

3. Solutions:

- Vérifier les permissions: `sudo chmod 666 /var/run/docker.sock`
- Vérifier si le port 9000 est déjà utilisé: `sudo lsof -i :9000`

## Utilisation de Portainer pour gérer les conteneurs

Portainer offre une interface graphique pour gérer vos conteneurs Docker sans utiliser la ligne de commande:

1. Accédez à Portainer via votre navigateur à l'adresse <http://localhost:9000>
2. Lors de la première connexion:
  - Créez un utilisateur administrateur
  - Choisissez l'environnement "Local"
3. Pour gérer vos conteneurs:
  - Allez dans la section "Containers" dans le menu de gauche
  - Vous verrez la liste de tous vos conteneurs avec leur statut
  - Utilisez les boutons d'action pour:
    - Démarrer/arrêter des conteneurs individuels
    - Redémarrer des conteneurs
    - Supprimer des conteneurs
    - Voir les logs en temps réel
    - Accéder à un terminal dans le conteneur
4. Avantages de Portainer:
  - Interface visuelle intuitive
  - Pas besoin de se souvenir des commandes Docker
  - Visualisation facile des statistiques (CPU, mémoire)
  - Gestion centralisée de tous vos conteneurs

Pour supprimer un conteneur via Portainer:

1. Cochez la case à côté du conteneur
2. Cliquez sur "Remove" dans la barre d'actions
3. Choisissez si vous voulez également supprimer les volumes associés (attention: cela supprimera les données persistantes)

## Erreurs de permission

### Erreurs lors du démarrage des conteneurs

Si vous rencontrez des erreurs de permission lors du démarrage:

1. Vérifiez les logs du conteneur spécifique
2. Assurez-vous que les dossiers ont les bonnes permissions

Solutions courantes:

```
bash
```

```
# Donner des permissions au dossier des rapports
```

```
mkdir -p ./rapports
```

```
chmod 777 ./rapports
```

```
# Donner des permissions au dossier de données Keycloak
```

```
mkdir -p ./keycloak/keycloak-data
```

```
chmod 777 ./keycloak/keycloak-data
```

## Erreurs lors de l'accès aux fichiers créés par Docker

Les fichiers créés par les conteneurs Docker appartiennent souvent à l'utilisateur root ou à des utilisateurs spécifiques à Docker:

Solution:

```
bash
```

```
# Pour accéder temporairement aux fichiers
```

```
sudo ls -la ./postgresql/postgres_data
```

```
# Pour changer Le propriétaire (à utiliser avec précaution)
```

```
sudo chown -R votre_utilisateur:votre_groupe ./postgresql/postgres_data
```

## Résolution des conflits de ports

Si certains ports sont déjà utilisés sur votre système, vous devrez modifier le fichier docker-compose.yml.

Ports utilisés par défaut:

- PostgreSQL: 5432
- pgAdmin: 5050
- Keycloak: 8080
- Portainer: 9000

Pour changer un port (exemple pour PostgreSQL):

```
yaml
```

```
postgres:
```

```
... # ...
```

```
... ports:
```

```
... - "5433:5432" # Utiliser Le port 5433 à la place de 5432
```

Après avoir modifié le fichier, redémarrez les conteneurs:

```
bash
```

```
docker-compose down
```

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

## Conclusion

Ce guide devrait vous aider à résoudre les problèmes courants avec votre Toolbox. Si vous rencontrez des problèmes non répertoriés ici, n'hésitez pas à consulter la documentation officielle de Docker et des services spécifiques, ou à demander de l'aide à votre équipe de support.