

# SAE 23 Mettre en place une solution informatique pour l'entreprise



## TD Mise en place de l'environnement Wordpress

### Objectif :

*Dans ce TD vous allez créer un environnement de travail pour la SAé 23 cet environnement de travail devra pouvoir être recrée et restauré automatiquement à l'aide de vos scripts.*

**Vous effacerez les conteneurs et images que vous aurez créées ainsi que les images que vous avez téléchargées avant de partir. Vous conserverez une sauvegarde des données vous permettant de les recréer.**

---

### Compte rendu :

Vous ferez état de votre travail dans cette SAé dans le blog WordPress que vous allez construire : prenez des copies d'écran au début.

**Tout travail rendu mais non montré à l'enseignant sera considéré comme non réalisé.**

---

**Si vous ne pouvez pas faire des docker pull (docker affiche que l'on a dépassé le nombre maximum de "pull request"), il faudra créer un compte docker hub (choisir la version gratuite !!!) et vous logger dans docker desktop. Le quota sera alors lié à votre compte et pas à la machine.**

**N'oubliez pas à la fin du TP de vous déconnecter.**

---

Prérequis : TP n°1 sur Docker fait en R202

Apprentissages critiques ciblés par ce TD :

- RT3 :
  - AC0311 | Utiliser un système informatique et ses outils
  - AC0312 | Lire, exécuter, corriger et modifier un programme
  - AC0313 | Traduire un algorithme, dans un langage et pour un environnement donné
  - AC0314 | Connaître l'architecture et les technologies d'un site Web
  - AC0315 | Choisir les mécanismes de gestion de données adaptés au développement de l'outil
  - AC0316 | S'intégrer dans un environnement propice au développement et au travail collaboratif

## Sommaire

<b>1</b>	<b>INTRODUCTION – CAHIER DES CHARGES</b>	<b>3</b>
<b>2</b>	<b>INSTALLATION DE WORDPRESS</b>	<b>3</b>
2.1	TELECHARGEMENT DES IMAGES	5
2.2	CREATION DU CONTENEUR POUR LA BASE DE DONNEES	5
2.3	CREATION D'UN RESEAU DEDIE	6
2.4	CREATION D'UN CONTENEUR PHPMYADMIN	6
2.5	CREATION D'UN CONTENEUR WORDPRESS	7
2.6	PERSISTANCE DES DONNEES	7
<b>3</b>	<b>PREMIERS PAS AVEC WORDPRESS</b>	<b>8</b>
3.1	PARAMETRAGE	8
3.2	PREMIERE PAGE DU BLOG	8
<b>4</b>	<b>SUPPRESSION DU SITE ET RECONSTRUCTION</b>	<b>9</b>

## 1 Introduction – Cahier des charges

*Vous venez de finir des études et avez décidé de devenir autoentrepreneur. Vous décidez de créer un serveur web client faisant votre promotion et permettant à vos clients d'évaluer automatiquement le coût de vos interventions en fonction du type d'intervention et de leur demande.*

*Comme vous avez très peu de temps, vous décidez de réutiliser le site du portfolio que vous aviez créé en BUT que vous aviez sauvegardé sur github.*

*Vous décidez également qu'un blog vous permettant de décrire la construction de votre serveur montrera également vos compétences. Cependant, vous ne voulez pas utiliser Symfony mais plutôt un CMS comme Wordpress spécialisé dans les blogs.*

*Le but de cette SAé sera donc de construire ce site en utilisant des conteneurs. Votre site devra pouvoir être recréé sur n'importe quel ordinateur muni de Docker à l'aide d'un script Powershell et de fichiers de sauvegardes.*

*Dans cette première partie (TD), nous allons utiliser Docker pour créer le site Wordpress où vous rédigerez vos comptes rendus à chaque séance et un mode d'emploi en anglais.*

*A l'issue de chaque séance, vous sauvegarderez les données (répertoire site web + base de données) et détruirez les conteneurs. Ces conteneurs seront reconstruits à l'issue de chaque séance.*

### Portainer

Une image docker, portainer/portainer-ce, disponible sur le hub, permet de gérer au mieux l'environnement Docker.

Il est adapté aux trois systèmes d'exploitation principaux que sont Windows, Mac OS et Linux, puisqu'il se lance dans un navigateur Web. Pour créer un container à partir de l'image, copiez le code suivant dans un terminal, en une seule ligne :

```
docker run -d -p 8000:8000 -p 9443:9443 --name portainer --restart=always -v
```

```
/var/run/docker.sock:/var/run/docker.sock -v portainer_data:/data portainer/portainer-ce
```

Pour utiliser le container, écrivez <https://localhost:9443> dans un navigateur Web.

## 2 Installation de Wordpress

Vous vous aiderez de la fiche de synthèse réalisée lors du TP n°1.

Comme pour le 1<sup>er</sup> TP sur Docker, vous taperez les commandes dans Powershell. Vous réaliserez en parallèle un fichier wordpress\_monnom.ps1 où vous les recopierez au fur et à mesure.

L'utilisation de Visual Studio Code est conseillée pour l'édition du script : il existe une extension pour interagir avec PowerShell. Sinon, il existe aussi notepad, notepad++, l'éditeur PowerShell de Microsoft, ...

Le but est de pouvoir tout recréer en lançant ce script.

Pour installer Wordpress, nous avons besoin :

- de l'image officielle Wordpress,
- de l'image officielle de mariadb (notre base de donnée).

## 2.1 Téléchargement des images

Retrouvez-les sur le docker hub et téléchargez les images (si cela marche, recopiez les commandes dans le script .ps1).

## 2.2 Création du conteneur pour la base de données

Regardez la documentation associée dans le docker hub et retrouvez comment démarrer un conteneur de mariadb (une instance).

Nous allons essayer de comprendre les paramètres :

- `--detach` : vous avez déjà utilisé cette commande dans sa version abrégée `-d` : elle signifie que le conteneur continuera de fonctionner en tâche de fond. Sans cette commande, on attendrait que le conteneur ait terminé sa tâche (ici jamais) pour rendre la main.
- `--env` permet d'affecter des valeurs à différentes variables d'environnement :
  - `MARIADB_USER=example_user` signifie que l'on va créer un utilisateur nommé `example_user` pour accéder à la base de données,
  - `MARIADB_PASSWORD` permet d'indiquer le mot de passe associé cet utilisateur,
  - `MARIADB_DATABASE` permet d'indiquer de créer une base de données (on indique son nom) et de l'associer à notre utilisateur, (vous devrez rajouter cet élément qui ne figure pas dans l'exemple),
  - `MARIADB_ROOT_PASSWORD` correspond au mot de passe de l'administrateur (`root`).
- `mariadb:latest` est le nom de l'image.

Rajouter le paramètre `-p` pour faire correspondre le port 3306 (base de données) du conteneur au 3306 du PC.

Lancez la commande en personnalisant les paramètres.

Aller dans l'interpréteur de commande de la base de données (à l'aide de `docker exec`) :

- retrouvez l'adresse ip du conteneur,
- vérifiez que la commande `ping` ne marche pas (`ping` n'est pas installé),
- installez le paquet `iputils-ping`,
- vérifiez que `ping xxx` (où `xxx` est le nom de votre conteneur) ne marche pas).

On a donc maintenant l'adresse ip de la base de données mais on a constaté qu'il n'y a pas de serveur DNS répondant au nom du conteneur. Cela signifie que si l'on doit désigner l'adresse de notre l'ip conteneur à un autre, on devra donner une ip. C'est gênant car l'ip étant affectée par le dhcp de Docker elle peut changer à chaque fois. Un nom serait plus intéressant qu'une ip.

### 2.3 Création d'un réseau dédié

Il est possible de demander à Docker de construire un réseau dédié à certains conteneurs. Ce réseau est muni d'un serveur de nom utilisable par les conteneurs qui s'y trouve (mais pas par notre PC). Cela permet également d'isoler un groupe de conteneur des autres dans leur propre réseau.

Utilisez la commande

```
docker network create sae23_votrenom
```

pour créer un réseau dédié sae23\_votrenom.

N'oubliez pas d'insérer cette commande dans votre fichier script ps1.

Mettez à jour le fichier de synthèse que vous avez réalisé au TP1.

Supprimez le conteneur mariadb que vous avez créé, et recréez-le en rajoutant le paramètre

```
--network sae23_votrenom
```

Répétez les étapes permettant d'obtenir l'ip et de tester le ping. Le ping associé au nom du conteneur devrait fonctionner dans le conteneur (mais pas à partir de votre PC qui n'est pas dans le même réseau). On pourra donc désigner l'adresse du conteneur en utilisant son nom au lieu de son IP si on est dans le même réseau sae23\_votrenom.

N'oubliez pas de mettre à jour la commande docker run dans votre script !

Utilisez des # pour ajouter des commentaires dans votre script !!!

### 2.4 Création d'un conteneur phpmyadmin

Chercher dans le dockerhub comment créer un conteneur phpmyadmin (image officielle).

On adaptera l'exemple :

- remplacer le paramètre `--link mysql_db_server:db` qui est obsolète par `--network sae23_votrenom` afin que le conteneur soit sur le même réseau que la base de donnée,
- affecter à la variable d'environnement `PMA_HOST` le nom associé à au conteneur base de donnée (on utilise le serveur de nom qui le fera correspondre à l'IP),
- placer le port 80 du conteneur sur le 9000 du PC.

Exécuter la commande, connectez-vous à phpmyadmin et utilisez le pour accéder à votre base de donnée avec le compte root et le compte personnel.

Ajouter la commande dans votre script.

## 2.5 Création d'un conteneur Wordpress

En regardant la documentation dans le docker hub, retrouver l'image Wordpress officielle et téléchargez là.

Regardez sa documentation et adaptez là pour :

- utiliser le réseau sae23\_votrenom,
- faire correspondre le port 80 du conteneur au port 80 du PC,
- indiquer à l'aide des variables d'environnement :
  - le nom de l'hôte hébergeant la base de donnée,
  - le nom de l'utilisateur de la base de donnée,
  - le mot de passe de cet utilisateur,
  - le nom de la base de donnée à utiliser.

Connectez-vous avec votre navigateur internet à Wordpress et vérifiez que cela marche (ne faites pas de configuration).

P.S. : Wordpress a besoin d'un tout petit peu de temps pour démarrer.

## 2.6 Persistance des données

Le site Wordpress est stocké dans le conteneur Wordpress (répertoire /var/www/html) et dans le conteneur mariadb (répertoire /var/lib/mysql).

Si nous supprimons le conteneur nous les supprimons aussi. C'est-à-dire que l'on supprime votre compte rendu tapé dans Wordpress. C'est un peu gênant.

Nous avons donc besoin que les données survivent à la destruction du conteneur : persistance. Pour cela, il faut que docker les stocke dans un volume situé soit sur votre disque dur (ou vous voulez) soit dans un emplacement géré par docker.

Cela est réalisé en rajoutant l'option `--volume repPC:repCONTENEUR` à la commande `docker run`. Attention vous devez écrire complètement le répertoire (ne pas utiliser . pour le répertoire en cours).

Supprimez vos conteneurs. Créez deux répertoires : badowordpress et htmlwordpress et faites le nécessaire pour y stocker les données (répertoire indiquez au début de cette question).

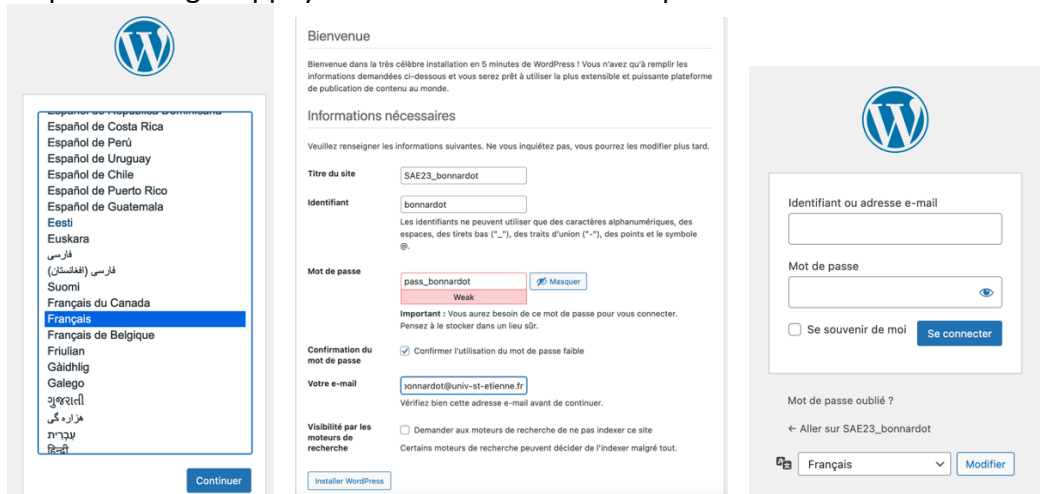
Relancez votre script et vérifiez que des données sont bien stockées dans ces répertoires.

Si les répertoires ne sont pas vides à la création du conteneur, on prend en compte leur contenu au lieu de les remplir avec le contenu par défaut. Vous conserverez précieusement le contenu de ces deux répertoires qui correspondent à votre site Wordpress.

## 3 Premiers pas avec WordPress

### 3.1 Paramétrage

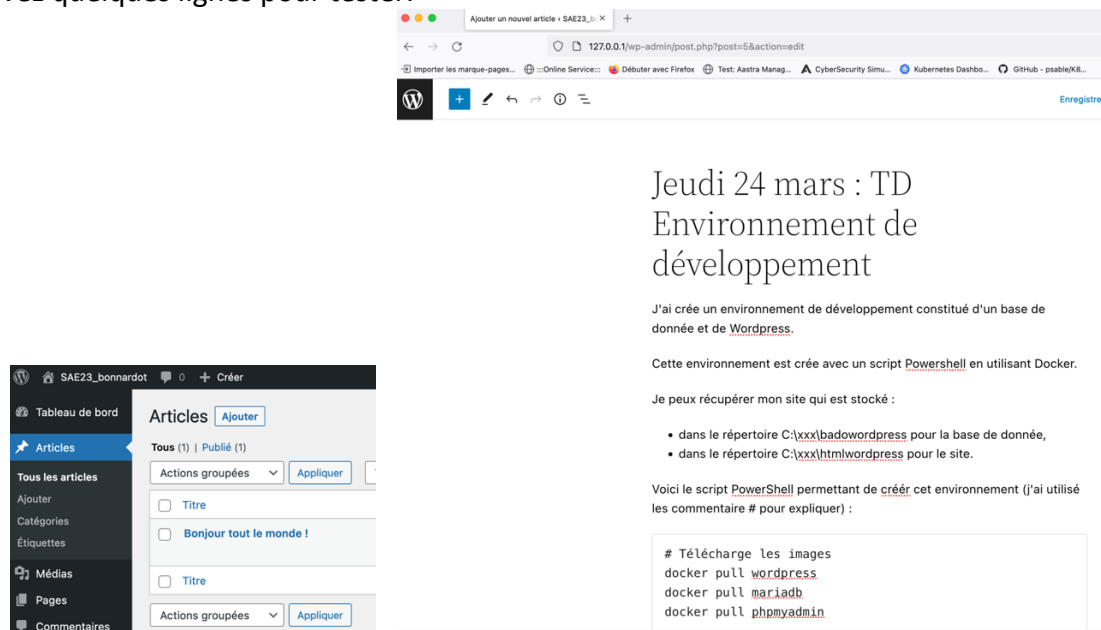
Connectez-vous à Wordpress avec votre navigateur web. Indiquez votre langue. Faites les premiers paramétrages appuyez sur Installer WordPress puis sur se connecter.



Vous noterez que l'url de la page de connexion est 127.0.0.1/wp-login.php  
Connectez-vous avec le compte qui vous avez défini.

### 3.2 Première page du blog

Cliquez sur l'onglet article puis sur Ajouter.  
Écrivez quelques lignes pour tester.



Jeudi 24 mars : TD  
Environnement de développement

J'ai crée un environnement de développement constitué d'un base de donnée et de Wordpress.

Cette environnement est crée avec un script Powershell en utilisant Docker.

Je peux récupérer mon site qui est stocké :

- dans le répertoire C:\xxx\badowordpress pour la base de donnée,
- dans le répertoire C:\xxx\htmlwordpress pour le site.

Voici le script PowerShell permettant de créer cet environnement (j'ai utilisé les commentaire # pour expliquer) :

```
# Télécharge les images
docker pull wordpress
docker pull mariadb
docker pull phpmyadmin
```

La barre + permet d'avoir accès à des outils d'insertion de code, d'image, ...  
N'oubliez pas de cliquer soit sur enregistrer le brouillon soit sur publier pour sauvegarder.

Cliquer sur le W en haut à gauche pour revenir à l'éditeur.  
Cliquer sur la maison pour voir votre site bas qui se trouve en 127.0.0.1.



## 4 Suppression du site et reconstruction

Fermez le site web

Stopper et supprimer vos conteneurs

Relancez votre script.

Vous devriez une fois que Wordpress aura démarré :

- Accéder à votre site Web en <http://127.0.0.1>
- Accéder à l'espace d'administration en <http://127.0.0.1/wp-login.php>
- Accéder à phpmyadmin en <http://127.0.0.1:9000>

Ne faites pas la suite tant que cela ne marche pas : vous ferez votre rapport sur WordPress. Essayer également de déployer votre site sur un autre PC (de l'IUT).

Vous savez pouvez donc maintenant sauvegarder ce que vous avez fait : il faut sauvegarder les deux dossiers.

Comme deux précautions valent mieux qu'une vous trouverez sur le docker hub une commande permettant de faire une sauvegarde de votre base de données dans un fichier texte du type :

```
docker exec bado sh -c 'exec mysqldump --all-databases -uroot -p"$MARIADB_ROOT_PASSWORD"' > "bado.sql"
```

Vous trouverez une commande pour la restauration.

Il est également possible de faire un export de la base de données avec phpmyadmin (on obtient un fichier sql). On peut alors recréer un conteneur mariadb associé à un volume vide et importer la base de données avec phpmyadmin. Mais attention, si vous devez faire un import votre fichier devra avoir une taille inférieure à 2 Mo. Si il est plus grand l'astuce est de compresser au format zip (on obtient alors un .sql.zip) que phpmyadmin sait gérer.

**ATTENTION A LA CORRUPTION DE BASE DE DONNEES MYSQL (cela m'est arrivé) : faites des sauvegardes de ce dossier avant et après avoir travaillé sous WordPress. Donnez une date à chaque sauvegarde et conservez l'historique. Comme cela en cas de problème, vous pouvez revenir à une version précédente. Doublez la sauvegarde avec un mysqldump. Arrêtez vos conteneurs avant de faire la sauvegarde.**

En cas de corruption ou de problème de fichier, vous serez responsable car vous avez été prévenu.

---

Document sous Licence Creative Commons Attribution - Pas d'Utilisation Commerciale - Partage dans les Mêmes Conditions 4.0 International

CC BY-NC-SA 4.0 