



# Inception

*Résumé : Ce document est un exercice lié à l'administration système.*

*Version : 5.1*

# Contenu

<b>I</b>	<b>Préambule</b>	<b>2</b>
<b>II</b>	<b>Introduction</b>	<b>3</b>
<b>III</b>	<b>Directives générales</b>	<b>4</b>
<b>IV</b>	<b>Instructions relatives à l'IA</b>	<b>5</b>
<b>V</b>	<b>Partie obligatoire</b>	<b>7</b>
<b>VI</b>	<b>Lisez-moi Exigences</b>	<b>12</b>
<b>VII</b>	<b>Conditions préalables à la validation</b>	<b>14</b>
<b>VIII</b>	<b>Partie bonus</b>	<b>15</b>
<b>IX</b>	<b>Soumission et évaluation par les pairs</b>	<b>16</b>

# **Chapitre I**

## **Préambule**



# **Chapitre II**

## **Introduction**

Ce projet vise à élargir vos connaissances en administration système à l'aide de Docker. Vous allez virtualiser plusieurs images Docker en les créant dans votre nouvelle machine virtuelle personnelle.

# Chapitre III

## Directives générales

- Ce projet doit être réalisé sur une machine virtuelle.
- Tous les fichiers nécessaires à la configuration de votre projet doivent être placés dans un dossier srcs .
- Un fichier Makefile est également requis et doit être placé à la racine de votre répertoire. Il doit configurer l'ensemble de votre application (c'est-à-dire qu'il doit créer les images Docker à l'aide du fichier docker-compose.yml).
- Ce sujet nécessite la mise en pratique de concepts que, selon votre niveau, vous n'avez peut-être pas encore appris. Nous vous conseillons donc de ne pas hésiter à lire toute la documentation relative à l'utilisation de Docker, ainsi que tout autre document qui pourrait vous être utile pour réaliser ce travail.

# Chapitre IV

## Instructions IA

### ● Contexte

Au cours de votre apprentissage, l'IA peut vous aider dans de nombreuses tâches différentes. Prenez le temps d'explorer les différentes capacités des outils d'IA et la manière dont ils peuvent vous aider dans votre travail. Cependant, abordez-les toujours avec prudence et évaluez les résultats de manière critique. Qu'il s'agisse de code, de documentation, d'idées ou d'explications techniques, vous ne pouvez jamais être tout à fait sûr que votre question était bien formulée ou que le contenu généré est exact. Vos pairs sont une ressource précieuse pour vous aider à éviter les erreurs et les angles morts.

### ● Message principal

- ✚ Utilisez l'IA pour réduire les tâches répétitives ou fastidieuses.
- ✚ Développez des compétences en matière de prompting (avec ou sans codage) qui vous seront utiles dans votre future carrière.
- ✚ Apprenez comment fonctionnent les systèmes d'IA afin de mieux anticiper et éviter les risques courants, les biais et les problèmes éthiques.
- ✚ Continuez à développer vos compétences techniques et vos compétences clés en travaillant avec vos pairs.
- ✚ N'utilisez que du contenu généré par l'IA que vous comprenez parfaitement et dont vous pouvez assumer la responsabilité.

### ● Règles pour les apprenants :

- Vous devez prendre le temps d'explorer les outils d'IA et de comprendre leur fonctionnement afin de pouvoir les utiliser de manière éthique et réduire les biais potentiels.
- Vous devez réfléchir à votre problème avant de saisir une commande, ce qui vous aidera à rédiger des commandes plus claires, plus détaillées et plus pertinentes en utilisant un vocabulaire précis.
- Vous devez prendre l'habitude de vérifier, d'examiner, de remettre en question et de tester systématiquement tout ce qui est généré par l'IA.
- Vous devez toujours demander l'avis de vos pairs : ne vous fiez pas uniquement à votre propre validation.

## ● Résultats de la phase :

- Développez des compétences en matière de prompting à la fois générales et spécifiques à un domaine.
- Boostez votre productivité grâce à une utilisation efficace des outils d'IA.
- Continuez à renforcer votre pensée computationnelle, votre capacité à résoudre des problèmes, votre adaptabilité et votre esprit de collaboration.

## ● Commentaires et exemples :

- Vous serez régulièrement confronté à des situations (examens, évaluations, etc.) où vous devrez démontrer votre réelle compréhension. Soyez prêt, continuez à développer vos compétences techniques et interpersonnelles.
- Expliquer votre raisonnement et débattre avec vos pairs révèle souvent des lacunes dans votre compréhension. Faites de l'apprentissage entre pairs une priorité.
- Les outils d'IA manquent souvent de contexte spécifique et ont tendance à fournir des réponses génériques. Vos pairs, qui partagent votre environnement, peuvent vous offrir des informations plus pertinentes et plus précises.
- Alors que l'IA a tendance à générer la réponse la plus probable, vos pairs peuvent vous offrir des perspectives alternatives et des nuances précieuses. Fiez-vous à eux pour vérifier la qualité de vos réponses.

### ✓ Bonne pratique :

Je demande à l'IA : « Comment tester une fonction de tri ? » Elle me donne quelques idées. Je les essaie et examine les résultats avec un collègue. Nous affinons l'approche ensemble.

### ✗ Mauvaise pratique :

Je demande à l'IA d'écrire une fonction entière, je la copie-colle dans mon projet. Lors de l'évaluation par les pairs, je ne peux pas expliquer ce qu'elle fait ni pourquoi. Je perds ma crédibilité et mon projet échoue.

### ✓ Bonne pratique :

J'utilise l'IA pour m'aider à concevoir un analyseur syntaxique. Ensuite, je passe en revue la logique avec un pair. Nous détectons deux bogues et le réécrivons ensemble : il est désormais meilleur, plus propre et parfaitement compréhensible.

### ✗ Mauvaise pratique :

J'ai laissé Copilot générer mon code pour une partie essentielle de mon projet. Il compile, mais je ne peux pas expliquer comment il gère les pipes. Lors de l'évaluation, je ne parviens pas à justifier mon choix et j'échoue dans mon projet.

# Chapitre V Partie

## obligatoire

Ce projet consiste à mettre en place une petite infrastructure composée de différents services selon des règles spécifiques. L'ensemble du projet doit être réalisé dans une machine virtuelle. Vous devez utiliser Docker Compose.

Chaque image Docker doit porter le même nom que le service correspondant. Chaque service doit fonctionner dans un conteneur dédié. Pour des raisons de performances, les conteneurs doivent être construits à partir de l'avant-dernière version stable d'Alpine ou de Debian. Le choix vous appartient. Vous devez également écrire vos propres fichiers Dockerfiles, un par service. Les fichiers Dockerfiles doivent être appelés dans votre fichier docker-compose.yml par votre Makefile. Cela signifie que vous devez créer vous-même les images Docker de votre projet. Il est alors interdit de récupérer des images Docker prêtes à l'emploi, ainsi que d'utiliser des services tels que DockerHub (Alpine/Debian étant exclu de cette règle).

Vous devez ensuite configurer :

- Un conteneur Docker contenant NGINX avec TLSv1.2 ou TLSv1.3 uniquement.
- Un conteneur Docker contenant uniquement WordPress + php-fpm (qui doit être installé et configuré), sans nginx.
- Un conteneur Docker contenant uniquement MariaDB, sans nginx.
- Un volume contenant votre base de données WordPress.
- Un deuxième volume contenant les fichiers de votre site web WordPress.
- Un réseau Docker qui établit la connexion entre vos conteneurs.

Vos conteneurs doivent redémarrer en cas de panne.



Un conteneur Docker n'est pas une machine virtuelle. Il n'est donc pas recommandé d'utiliser des correctifs de fortune basés sur « tail -f » et des méthodes similaires lorsque vous essayez de l'exécuter. Renseignez-vous sur le fonctionnement des démons et demandez-vous s'il est judicieux de les utiliser ou non.





Bien sûr, l'utilisation de `network: host` ou `--link` ou `links:` est interdite. La ligne `network` doit être présente dans votre fichier `docker-compose.yml`. Vos conteneurs ne doivent pas être démarrés avec une commande exécutant une boucle infinie. Cela s'applique donc également à toute commande utilisée comme point d'entrée ou dans des scripts de point d'entrée. Voici quelques correctifs de fortune interdits : `tail -f`, `bash`, `sleep infinity`, `while true`.



Renseignez-vous sur le PID 1 et les meilleures pratiques pour écrire des fichiers Dockerfile.

- Votre base de données WordPress doit comporter deux utilisateurs, dont l'un est l'administrateur. Le nom d'utilisateur de l'administrateur ne peut pas contenir `admin/Admin` ou `admin-istrator/Administrator` (par exemple, `admin`, `administrator`, `Administrator`, `admin-123`, etc.).



Vos volumes seront disponibles dans le dossier `/home/login/data` de la machine hôte utilisant Docker. Bien sûr, vous devez remplacer `login` par votre nom d'utilisateur.

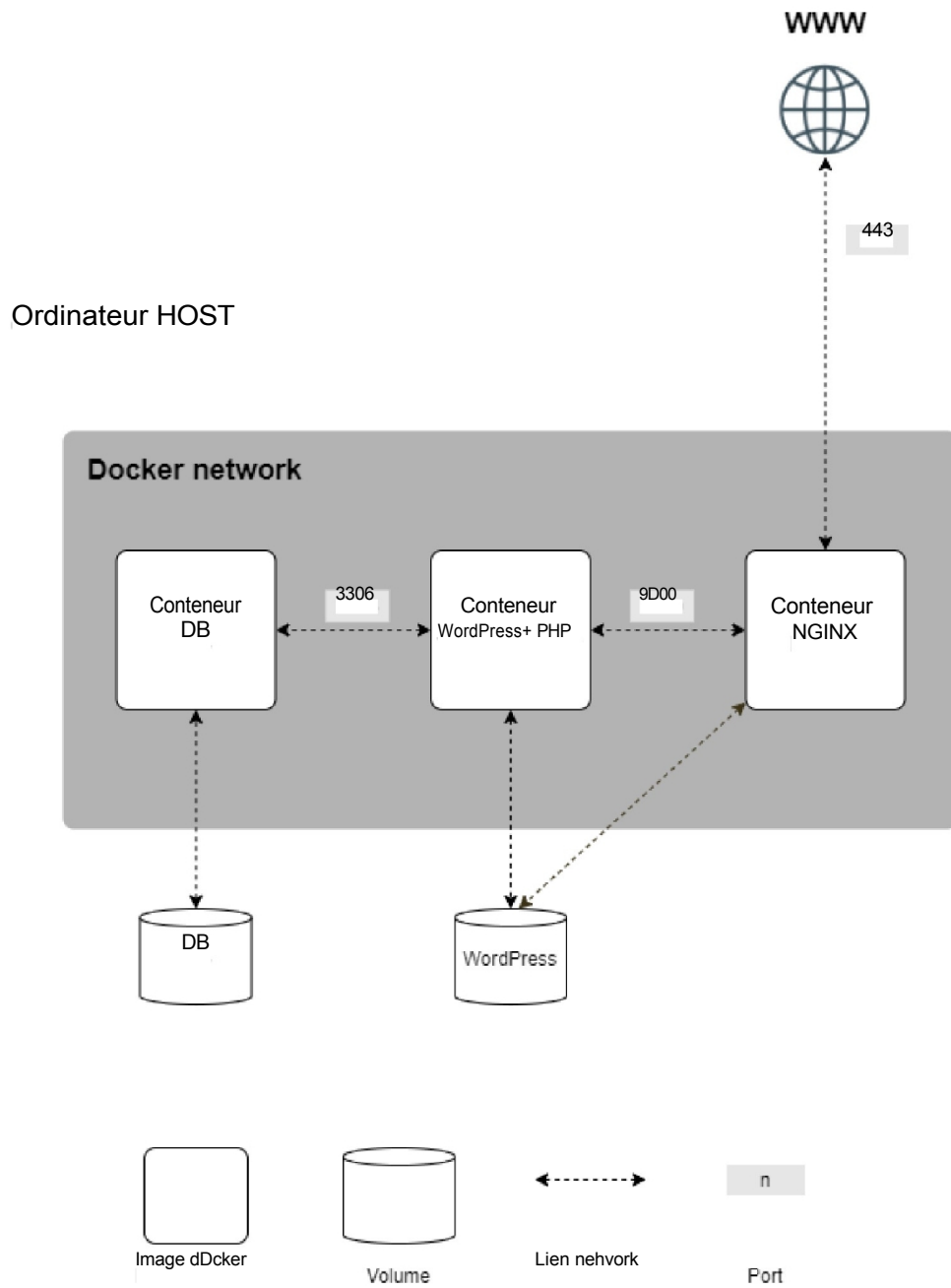
Pour simplifier les choses, vous devez configurer votre nom de domaine afin qu'il pointe vers votre adresse IP locale.

Ce nom de domaine doit être `login.42.fr`. Là encore, vous devez utiliser votre propre identifiant. Par exemple, si votre identifiant est `wil`, `wil.42.fr` redirigera vers l'adresse IP pointant vers le site web de `wil`.



La balise « `latest` » est interdite.  
Aucun mot de passe ne doit être présent dans vos fichiers Dockerfiles. L'utilisation de variables d'environnement est obligatoire.  
De plus, il est obligatoire d'utiliser un fichier `.env` pour stocker les variables d'environnement. Il est fortement recommandé d'utiliser les secrets Docker pour stocker toute information confidentielle. Toute information d'identification, clé API ou mots de passe trouvés dans votre dépôt Git (en dehors des secrets correctement configurés) entraîneront l'échec du projet.  
Votre conteneur NGINX doit être le seul point d'entrée dans votre infrastructure via le port 443 uniquement, en utilisant le protocole TLSv1.2 ou TLSv1.3.

Voici un exemple de schéma du résultat attendu :



Voici un exemple de la structure de répertoires attendue :

```
$> ls -alR
total XX
drwxrwxr-x 3 wil wil 4096 avril 42 20:42 .
drwxrwxrwt 17 wil wil 4096 avril 42 20:42 ..
-rw-rw-r-- 1 wil wil XXXX avril 42 20:42 Makefile
drwxrwxr-x 3 wil wil 4096 avril 42 20:42 secrets
drwxrwxr-x 3 wil wil 4096 avril 42 20:42 srcs

./secrets :
total XX
drwxrwxr-x 2 wil wil 4096 avril 42 20:42 .
drwxrwxr-x 6 wil wil 4096 avril 42 20:42 ..
-rw-r--r-- 1 wil wil XXXX avril 42 20:42 credentials.txt
-rw-r--r-- 1 wil wil XXXX avril 42 20:42 db_password.txt
-rw-r--r-- 1 wil wil XXXX avril 42 20:42 db_root_password.txt

./srcs :
total XX
drwxrwxr-x 3 wil wil 4096 avril 42 20:42 .
drwxrwxr-x 3 wil wil 4096 avril 42 20:42 ..
-rw-rw-r-- 1 wil wil XXXX avril 42 20:42 docker-compose.yml
-rw-rw-r-- 1 wil wil XXXX avril 42 20:42 .env
drwxrwxr-x 5 wil wil 4096 avril 42 20:42 requirements

./srcs/requirements:
total XX
drwxrwxr-x 5 wil wil 4096 avril 42 20:42 .
drwxrwxr-x 3 wil wil 4096 avril 42 20:42 ..
drwxrwxr-x 4 wil wil 4096 avril 42 20:42 bonus
drwxrwxr-x 4 wil wil 4096 avril 42 20:42 mariadb
drwxrwxr-x 4 wil wil 4096 avril 42 20:42 nginx
drwxrwxr-x 4 wil wil 4096 avril 42 20:42 outils
drwxrwxr-x 4 wil wil 4096 avril 42 20:42 wordpress

./srcs/requirements/mariadb :
total XX
drwxrwxr-x 4 wil wil 4096 avril 42 20:45 .
drwxrwxr-x 5 wil wil 4096 avril 42 20:42 ..
drwxrwxr-x 2 wil wil 4096 avril 42 20:42 conf
-rw-rw-r-- 1 wil wil XXXX avril 42 20:42 Dockerfile
-rw-rw-r-- 1 wil wil XXXX avril 42 20:42 .dockerignore
drwxrwxr-x 2 wil wil 4096 avril 42 20:42 outils [...]

./srcs/requirements/nginx:
total XX
drwxrwxr-x 4 wil wil 4096 avril 42 20:42 .
drwxrwxr-x 5 wil wil 4096 avril 42 20:42 ..
drwxrwxr-x 2 wil wil 4096 avril 42 20:42 conf
-rw-rw-r-- 1 wil wil XXXX avril 42 20:42 Dockerfile
-rw-rw-r-- 1 wil wil XXXX avril 42 20:42 .dockerignore
drwxrwxr-x 2 wil wil 4096 avril 42 20:42 outils [...]

$> cat srcs/.env
DOMAIN_NAME=wil.42.fr #
MYSQL_SETUP
MYSQL_USER=XXXXXXXXXXXX [...]
$>
```



Pour des raisons de sécurité évidentes, tous les identifiants, clés API, mots de passe, etc. doivent être enregistrés localement de différentes manières/dans différents fichiers et ignorés par git. Les identifiants stockés publiquement vous mèneront directement à l'échec du projet.



Vous pouvez stocker vos variables (sous forme de nom de domaine) dans un fichier de variables d'environnement tel que `.env`

# Chapitre VI

## Lisez-moi Configuration requise

Un fichier README.md doit être fourni à la racine de votre dépôt Git. Son objectif est de permettre à toute personne qui ne connaît pas le projet (collègues, personnel, recruteurs, etc.) de comprendre rapidement en quoi consiste le projet, comment l'exécuter et où trouver plus d'informations sur le sujet.

Le fichier README.md doit au moins contenir :

- La toute première ligne doit être en italique et se lire comme suit : *Ce projet a été créé dans le cadre du programme 42 par <login1>[, <login2>[, <login3>[...]]]*.
- Une section « Description » qui présente clairement le projet, y compris son objectif et un bref aperçu.
- Une section « Instructions » contenant toutes les informations pertinentes sur la compilation, l'installation et/ou l'exécution.
- Une section « Ressources » répertoriant les références classiques liées au sujet (documentation, articles, tutoriels, etc.), ainsi qu'une description de la manière dont l'IA a été utilisée, en précisant pour quelles tâches et quelles parties du projet.

► **Des sections supplémentaires peuvent être nécessaires en fonction du projet** (par exemple, exemples d'utilisation, liste des fonctionnalités, choix techniques, etc.).

Tout ajout requis sera explicitement indiqué ci-dessous.

- Une section « **Description du projet** » doit également expliquer l'utilisation de Docker et les sources incluses dans le projet. Elle doit indiquer les principaux choix de conception, ainsi qu'une comparaison entre :
  - Machines virtuelles vs Docker
  - Secrets vs variables d'environnement
  - Réseau Docker vs réseau hôte
  - Volumes Docker vs montages liés



Votre fichier README doit être rédigé en anglais.

# Chapitre VII

## Conditions préalables à la validation

En plus des exigences existantes, les fichiers de documentation suivants doivent être présents à la racine de votre référentiel. Ils doivent être rédigés au format Markdown (.md).

- **USER\_DOC.md — Documentation utilisateur** Ce fichier doit expliquer, en termes clairs et simples, comment un utilisateur final ou un administrateur peut :
  - Comprendre quels services sont fournis par la pile.
  - Démarrer et arrêter le projet.
  - Accéder au site web et au panneau d'administration.
  - Localiser et gérer les identifiants.
  - Vérifier que les services fonctionnent correctement.
- **DEV\_DOC.md — Documentation pour les développeurs** Ce fichier doit décrire comment un développeur peut :
  - Configurer l'environnement à partir de zéro (prérequis, fichiers de configuration, secrets).
  - Construire et lancer le projet à l'aide du Makefile et de Docker Compose.
  - Utiliser les commandes appropriées pour gérer les conteneurs et les volumes.
  - Identifier où les données du projet sont stockées et comment elles sont conservées.

# Chapitre VIII

## Partie bonus

Pour ce projet, la partie bonus se veut simple.

Un fichier Dockerfile doit être rédigé pour chaque service supplémentaire. Ainsi, chaque service s'exécutera dans son propre conteneur et disposera, si nécessaire, de son volume dédié.

Liste des bonus :

- Configurez le cache redis pour votre site WordPress afin de gérer correctement le cache.
- Configurez un conteneur de serveur FTP pointant vers le volume de votre site WordPress.
- Créez un site web statique simple dans le langage de votre choix, à l'exception du PHP (oui, le PHP est exclu). Par exemple, un site vitrine ou un site pour présenter votre CV.
- Configurez Adminer.
- Configurez un service de votre choix que vous jugez utile. Lors de la soutenance, vous devrez justifier votre choix.



Pour compléter la partie bonus, vous avez la possibilité de configurer des services supplémentaires. Dans ce cas, vous pouvez ouvrir davantage de ports en fonction de vos besoins.



La partie bonus ne sera évaluée que si la partie obligatoire est parfaitement réalisée. Parfaitement signifie que la partie obligatoire a été entièrement réalisée et fonctionne sans aucun dysfonctionnement. Si vous n'avez pas satisfait à TOUTES les exigences obligatoires, votre partie bonus ne sera pas évaluée.



# Chapitre IX

## Soumission et évaluation par les pairs

Soumettez votre devoir dans votre dépôt Git comme d'habitude. Seul le travail contenu dans votre dépôt sera évalué lors de la soutenance. N'hésitez pas à vérifier les noms de vos dossiers et fichiers pour vous assurer qu'ils sont corrects.

Au cours de l'évaluation, une brève **modification du projet** peut parfois être demandée. Il peut s'agir d'un changement mineur de comportement, de quelques lignes de code à écrire ou à réécrire, ou d'une fonctionnalité facile à ajouter.

Bien que cette étape **ne s'applique pas à tous les projets**, vous devez vous y préparer si elle est mentionnée dans les directives d'évaluation.

Cette étape vise à vérifier votre compréhension réelle d'une partie spécifique du projet. La modification peut être effectuée dans n'importe quel environnement de développement de votre choix (par exemple, votre configuration habituelle) et devrait être réalisable en quelques minutes, sauf si un délai spécifique est défini dans le cadre de l'évaluation.

On peut par exemple vous demander d'apporter une petite mise à jour à une fonction ou à un script, de modifier un affichage ou d'ajuster une structure de données pour stocker de nouvelles informations, etc.

Les détails (portée, objectif, etc.) seront précisés dans les **directives d'évaluation** et peuvent varier d'une évaluation à l'autre pour un même projet.