Rétrospectives de projet

Réunion du 28 septembre :

Concernant les réunions :
- Une réunion par semaine à planifier à la fin de la réunion précédente
- Réunion courte et préparé : il faut préparer à l'avance une présentation sur l'avancement du sujet et les questions potentielles
- Faire une rétrospective / récapitulatif de chaque réunion
- Penser à répondre à tous lors d'un envoi de mail

Concernant le sujet :
- Site internet permettant à un joueur d'avoir les informations concernant les défis
- clé en main signifie que l'application doit être à la disposition du client sans qu'il ait besoin d'avoir des privilèges root pour l'utiliser
- L'application est constituée d'un ensemble de composants docker exécuté en parallèle. Chaque composant à un ou plusieurs points d'entrées accessibles par les autres composants ou par le joueur. Les composants peuvent donc communiquer entre eux sans nécessairement que le client le puisse.

Technologie sur lesquels se renseigner :
- podman : alternative à docker
- docker et docker-compose

Réunion du 7 octobre :

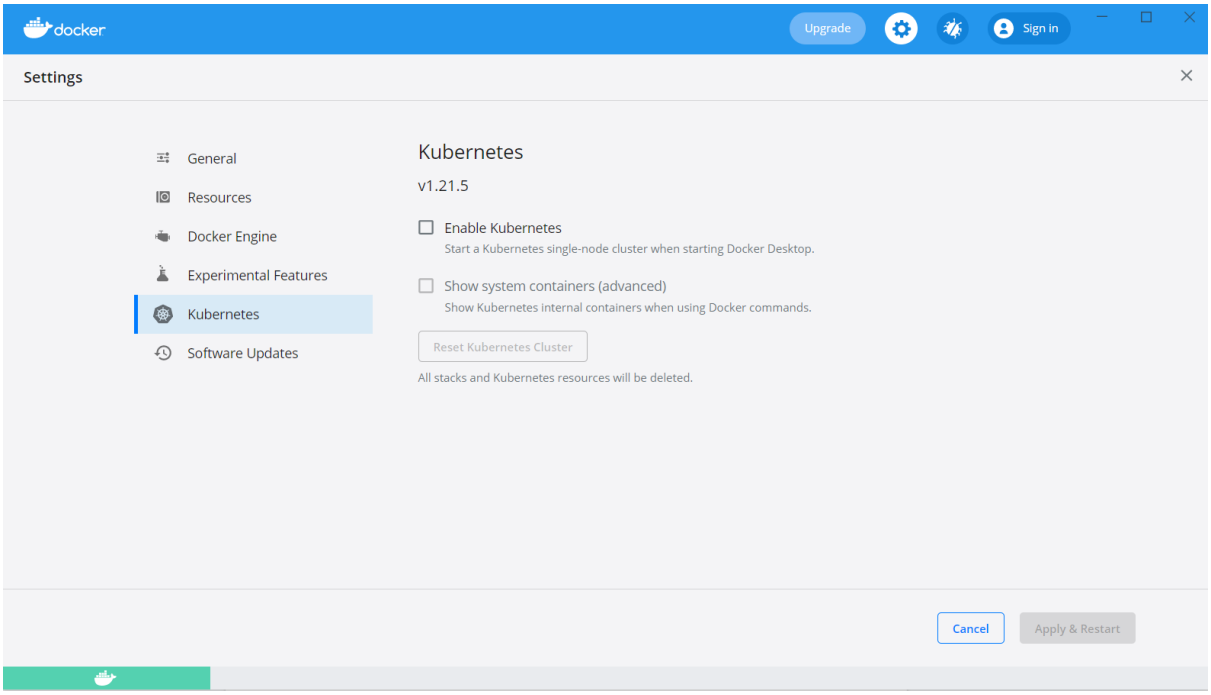
Défi ssh utilise la commande bash ssh ip [port] pour se connecter à l'image du docker (= container). Cette commande a pour effet de se connecter au bash de la machine distante, dans notre cas ce sera une machine virtuel linux lancer en docker.

Il faudrait savoir s'il est possible d'avoir une persistance des données contenues dans un docker sans avoir besoin d'une base de données.
Qu'est-ce que c'est les volumes docker ? Base de données ou volume ?

Lors d'un docker-compose down on ne clear par les volumes mais lors d'un docker-compose up on crée un nouveau volume donc il n'y a pas de persistance des données avec les volumes.
docker-run : relance le conteneur où il en était

Le projet est clé en main. C'est à dire qu'une fois que l'on pull le projet depuis git on a juste à exécuter une commande (docker-compose up par exemple) pour lancer tous les dockers sur notre machine et commencer les défis.

Il existe une option dans l'application docker desktop de windows pour autoriser les scripts kubernetes.



Les défis du projet doivent avoir :

- un aspect réseau avec ssh, api rest etc
- un aspect distribution avec des défis git
- un aspect systeme avec qq commande bash/shell

Si docker-compose ne permet pas de faire ce que l'on veut faire, une alternative avec podman ?
podman : comme docker-compose mais avec yml (podman kubes ?)

Réunion du 12 octobre :

Question de la persistance des données (base de données ou volume).
Le joueur à accès à un nombre de défis et pour accéder au défi numéro 2 il doit avoir fini le défi numéro 1 et ainsi de suite.
Tant que le joueur n'a pas coupé les dockers il peut accéder au défi auxquels il s'était arrêté. Cependant dès que le joueur coupe les dockers, il doit les relancer pour pouvoir jouer, et comme les dockers n'ont pas de persistance (à vérifier) sans base de données, il n'y a aucun moyen de savoir où est-ce que le joueur s'était arrêté.

Idée de défi GIT :

- récupérer le projet git-game et le mettre dans un docker
- retirer tout ce qui n'est pas nécessaire
- permettre de récupérer un mot clé, à la fin du défi, à entrer sur le site défistribué pour accéder au défi suivant

Idée de défi SSH

- lancer une image d'un micro système linux en docker (alpine ?)
- le joueur ouvre un terminal sur sa machine

- il doit entrer la commande bash ssh avec les bons arguments pour se connecter à la machine distante.
- lorsque sa connexion ssh avec le docker est réussi un script bash-rc est exécuté
- 2ème étape : le joueur doit se déconnecter et envoyer une clé public au docker pour pouvoir se connecter sans avoir à renseigner son mot de passe.
- on vérifie que le joueur se connecte de la bonne manière
- Une fois que c'est fait, on envoie la clé solution sur le terminal du joueur.

Est-ce qu'on a besoin de scruter le terminal ? Et si oui comment le faire ?
-> Processus daemon exécuté au lancement de la vm et qui envoie un message toutes les n secondes pour aider le joueur.

Réunion du 19 octobre :