I - Intégration continue - Local

Afin de s'assurer que le développement de l'application se déroule bien, on a mis en place plusieurs outils.

1) L'espace de travail.

Utilisation de Lerna qui est un gestionnaire de mono-repo. C'est une façon de stocker son code, les modules et les bibliothèques sont tous réunis au même endroit. Cela nous permet entre autre de réunir l'utilisation de certains programmes pour nettoyer le code où même éviter de dupliquer certains modules.
Le défaut c'est que les dossiers `node modules` sont très importants et que certaines dépendances sont réutilisées mais pas de la même manière et pas toujours avec les mêmes versions ce qui peut créer des conflits de version.

2) Eslint.

Tout le monde code de manière différente, ce qui peut amener à une lecture et une difficile compréhension de ce code.
Afin de résoudre ce problème, on a mis en place une norme de codage grâce à Eslint basée sur des normes de Airbnb.
Cela permet un formatage de code générique et propre au groupe, pas à seulement une personne.

3) Git-cz et Husky.

3.1) Git-cz.

C'est un outil qui nous permet de faire des commit selon le style que l'on souhaite et en respectant une certaine sémantique.

3.2) Husky.

Outil qui facilite l'écoute des hooks de git et nous permet de faire des actions selon certaines commandes de git.

Il nous sert à capter le moment où un commit est fait et lancer des commandes de lint et de test juste avant.

II - Intégration continue - Github

1) Github action.

Github action nous permet de configurer des action sur des branches. Ce sont des machines mises à disposition par github.

1.1) Node.

L'action node nous permet de faire plusieurs actions sur une machine Ubuntu. Le but ici est de tester, comme dans des conditions réelles, le fonctionnement du programme sur une machine qui sera la même que sur le serveur hôte.
Lors de la merge request sur la branche de développement, l'action va lancer plusieurs commandes les unes après les autres.

- run: npm run install:all
  `Installation de tous les packages utilisés lors du projet. Le but est d'être sûr que l'installation se fait bien et qu'il n'y a pas de problèmes de version.`
- run: npm run build --if-present
  `Vérifier que les différentes application ont leur build qui fonctionne correctement.`
- run: npm run test
  `Vérifier que les test passent bien en ligne et qu'ils n'ont pas été évités lors du push.`
- run: npm run lint
  `On fini par reformater le code au cas où.`

1.2) Push Epitech

Le but de cette action est de push tous les commits qui sont fait et faire en sorte que le repo mis à disposition d'Epitech soit la copie conforme du notre.
Disposition prise car on estimait ne pas avoir assez de liberté concernant les paramètres du repo comme la sécurité de celui-ci, les actions, etc ...

III - Déploiement continue.

Lors du merge de la branche de dev qui a été normalement testée celle-ci est automatiquement push et déployée sur le serveur dans le cloud de Digital Ocean.

1) Digital Ocean.

Machine qui tourne sur ubuntu.
Dessus sont installés Mysql, pm2 pour gérer les serveurs node et Nginx pour les serveurs virtuels.