**Rapport Projet *Tweetoscope***

Pour arriver à mettre une application minimale permettant de prédire la popularité d’un tweet, nous avons dans un premier temps créé le dépôt git distant, ensuite nous nous sommes réparti les tâches, enfin l’application a été développée en utilisant une approche incrémentale c’est-à-dire en ajoutant différentes briques élémentaires de l’application petit à petit.

Notre application a à chaque évolution suit 4 cycles (que l’on peut dire qu’il s’agit du processus CI/CL vu en cours) :

* Test : comme son nom l’atteste, ce premier stage consiste à réellement s’assurer que l’application marche et a un comportement voulu. Dans notre pipeline sur Gitlab, nous avons tenté en vain d’inclure ce stage dans le fichier .gitlab-ci.yml en faisant en sorte de faire tourner Kafka sur gitlab mais rien ne marchait, du coup ce stage était effectué en local, viens ensuite :
* Qualité (quality) : ce deuxième stage consistait à mesurer la qualité de notre code. Pour ce faire, à chaque push sur le dépôt master distant, ce job s’exécutait pour évaluer la qualité de notre code et nous avons créé, un dossier nommé codequality-results. Les critères pris en jeux pour évaluer la qualité du code sont dans un premier de vérifier les redondances et les répétitions inutiles.
* Build : Viens ensuite le stage Build dont l’objectif principal est de construire les images Docker et de les pusher au niveau de notre répertoire git.
* Deploy : le dernier stage consiste à déployer ou à faire de la livraison continue en déployant l’application sur le cluster de l’école. Pour ce faire, nous savions faire du déploiement via Gitlab en utilisant d’autres services tiers mais sur le cluster de l’école, la procédure à faire était de récupérer les deux fichiers qui se trouve dans le dossier Deploiement/cluster et d’exécuter la commande kubectl apply -f ….

Pour s’assurer que notre solution est tolérable aux pannes et scalable, nous avons utilisé dans un premier temps en local chez nous et ensuite pour passer à l’échelle en utilisant les ordinateurs de l’école nous avons utilisé Kubernetes. Dans le fichier yaml permettant d’utiliser Kubernetes, nous avons systématique répliquer toutes les images docker sur différents nœuds (au minimum 4 fois) ce qui nous permettait de dire que la solution était au max tolérable à 3 pannes.