ZHOU Eric

XU Paul

RAPPORT INTERMEDIAIRE

Pour l'objectif de ce projet nous avons pour le début décidé de partir sur la base des corrections de TME dont on a modifié et ajouter du code au fur et à mesure du projet.

Nous nous sommes aidés en partie de l'IA pour avancer sur ce projet.

Au niveau de la base de données Redis, nous sommes partis sur un kind «stateful set » pour permettre une facilité de la persistance des données même si le pod est recrée et aussi une image redis-exporter pour la partie metrics qui sera utilisé.

L'implémentation de grafana/prometheus ont été faite à l'aide d'un fichier yml pour les ajouter dans le cluster, de ce fait cela nous permet par défaut d'ajouter prometheus à grafana directement sans le faire manuellement. Prometheus et Grafana ont tous les deux une persistance des données d'une session à l'autre.

Prometheus a également une fonction de découverte automatique des pods donc celui du backend pour notre cas présent par exemple.

Pour la lecture du fichier conf.js du frontend par App.js nous avons choisis d'utiliser une variable d'environnement donc process.env … pour faire une mise à jour de la variable en fonction de l'IP externe donné par « minikube tunnel ». Pour implémenter cette solution nous avons dû partir sur une ConfigMap qui injecte une variable d'environnement lors du déploiement du frontend.

Pour le serveur node.js nous avons ajouter des annotations de prometheus directement dans le fichier yml du serveur pour permettre une automisation de la collecte des métriques.

Nous avons également ajouté des kind « horizontalpod autoscaler » pour la mise en échelle automatiques du backend et de la base de données avec un certains nombres de replicas.