Introduction au laboratoire

Choix technologiques

Ce laboratoire a été codé en c#. Le langage n'est pas imposé et vous pouvez créer ou réécrire vos services dans le langage de votre choix (à vos risques et périls).

Le Framework utilisé pour les microservices est ASP.NET Core.

Bien que le système n'utilise pas Docker Swarm ou Kubernetes, certaines des fonctionnalités ont été reproduites. Nous avions besoin d'une solution facile à déployer avec un minimum de configuration. De plus ce système permet l'ajout de "Chaos Engineering" et d'évaluations spécifiques au cours de LOG430 en un seul composant déployable sur votre Docker Desktop.

Docker Desktop: Vous devez utiliser Docker Desktop puisqu'il permet une gestion de conteneur local et devez activer dans les "Settings" => "General" => "Expose daemon on tcp://localhost:2375 without TLS". Votre ordinateur doit permettre la virtualisation.

Un ServiceMeshHelper Nuget package à été ajouté à vos service pour faciliter l'accès au service mesh (et au service discovery). Cependant, il est possible pour vous de faire les mêmes appels par vous-même par l'API du NodeController

La librairie MassTransit a été choisie pour les interactions avec RabbitMQ puisqu'elle englobe des attributs de qualité intéressants pour nos besoins.

Les services (incluant le NodeController) utilisent Swagger (OpenAPI) pour la documentation. Quand vos services sont déployés, vous y avez accès à cette adresse: @http://localhost:[Port Number]/swagger/index.html

Le code qui vous est donné comporte des erreurs. Certaines sont intentionnelles pour vous permettre de prendre de meilleures décisions architecturales. D'autres sont moins intentionnelles... Vous êtes responsable des répercussions de ces erreurs sur votre système.

Le NodeController est «Pull» de DockerHub. Si vous pensez qu'il contient une erreur, informez-en votre chargé de laboratoire en fournissant le contexte exact. Si le problème peut être reproduit, il sera corrigé et une nouvelle version sera publiée. Votre version du NodeController sera affichée sur l'interface utilisateur (dashboard) lors de l'évaluation. Il sera donc facile de repérer que vous n'avez pas la dernière version. Dans ce cas, vous pourrez simplement redéployer avec la dernière version en supprimant auparavant l’image dans Docker Desktop.

Pointeurs généraux

Vous êtes encouragés à expérimenter!

N'ayez pas peur de redéployer souvent.

Vous pouvez modifier le nombre de Nano CPUs (1 000 000 000 équivaut à un cœur) alloué à chaque conteneur à l'aide de l'API du NodeController si une expérience n'est pas en cours.

Avant une évaluation, je vous conseille de relancer votre Docker Daemon (Dockerd) afin de nettoyer tous les memory leaks et autres problèmes potentiels.

Vos Dockerfiles sont en mode «Debug» pour simplifier votre debugging, il pourrait être intéressant de les mettre en mode «Release» pour les évaluations 2 et 3.

Les tests ont été déprecated dû à un manque de maintenance et de temps sur les services que vous utilisez. Plutôt que de vous remettre une couverture et qualité de test douteuse, ils ont été enlevés. Le projet de test demeure dans la solution. Sentez-vous libre d'écrire des tests au début afin de vous assurer que vos changements n'impactent pas la logique. Les «mocks» seront essentiels pour vos tests, si vous en faites.

Suite 1- Prenez connaissance du Wiki du projet, disponible sur GitHub.