

TP Kubernetes, (Event Driven Architecture) EDA

Étape 1 : manipulations

Objectifs :

Après avoir installé un environnement conforme kubernetes avec le bundle MINIKUBE, voir le document « [document tp kubernetes.pdf](#) », vous présenterez dans un premier temps, dans un document PDF ou Markdown (*) les premiers tests (Questions Kubernetes ci dessous) de l'environnement Kubernetes en répondant aux questions qui suivent.

()Vous mettrez la ou les commande(s) utilisée(s) et une preuve de bon fonctionnement de celle ci (capture écran de la CLI par exemple) pour chacune des questions ci dessous.*

Questions Kubernetes.

Démarrer **minikube** pour pouvoir travailler sur **kubernetes** avec **kubectl**

Gestion de minikube

- Vérifiez que minikube pointe correctement vers le moteur docker ?
- Quels sont les addons actuellement installé ?
- Installez celle qui vous semble intéressante, pourquoi ?
- Lister les profils actif sous minikube avec toutes ses caractéristiques ?
- Quels sont les profils en cours ?
- Comment puis je créer un nouveau profile, que représente un profile ?
- Afficher le statut de minikube ?
- Comment puis je accéder au dashboard de minikube ?
- Qu'est ce que le DashBoard, que présente t'il ?
- Lister les nœuds d'un profil ?
- Ajouter un nœud à un profil minikube, supprimer ce même nœud. ?
- Consulter les logs de minikube, comment faire ?

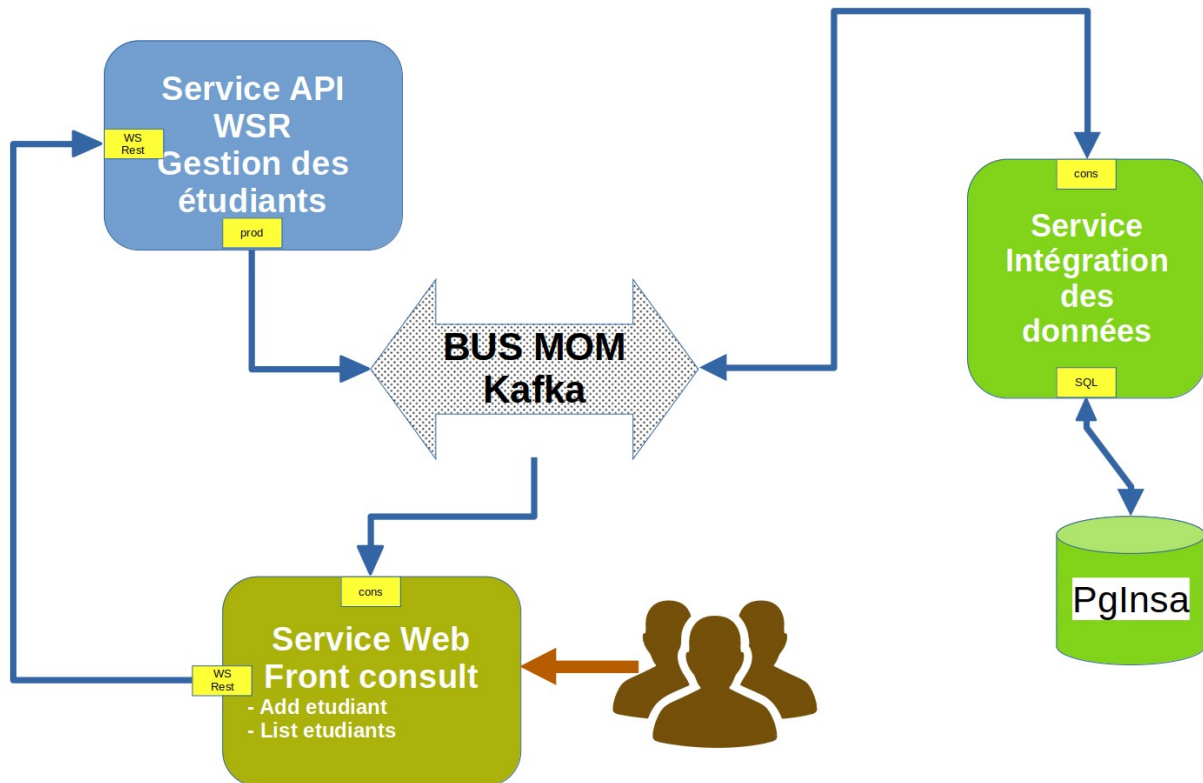
Gestion des pods et services sous kubernetes

- Lister les images actuellement en exécution dans votre environnement minikube ?
- Lancer une image nginx dans un pod et/ou un deployment en mode impératif. ?
- Créer un service en mode impératif permettant d'accéder à votre service nginx. ?
- Visualiser les informations du pod et du service. ?
- Obtenir l'url du service. ?
- Exécuter le service dans lynx ou Firefox ou tout autre browser. ?
- Comment puis je lancer une commande bash directement dans mon conteneur nginx ?
- Lister les logs du conteneur nginx du pod ?

Étape 2 : Présentation d'un projet d'une architecture de type Event Driven Architecture (EDA)

Vous allez proposer une architecture de type EDA qui permettra les éléments suivants :

A partir du projet du TD précédent (WS Rest étudiants), vous aller transformer le code source pour que l'on puisse a partir de cette interface EndPoint Rest API alimenter une base de données (postgresQl) via un service d'événements comme le schéma ci dessous le propose (**ceci reste une proposition**).



Cette petite architecture, permettra via les « endpoints » disponible d'envoyer des messages vers le bus kafka qui seront consommés par le service d'intégrations des données vers la base postgresql et exécuterons les ordres. Ce service d'intégration retournera des messages vers le bus qui sera consommé par le service Web Front pour retourner à l'utilisateur la réussite ou la liste. On implémentera également l'api qui permet de lister l'ensemble des étudiants et une interface web qui permet de se connecter a cette api pour introduire un nouvel étudiant.

Vous pouvez adapter cette architecture selon votre besoin lors du développement, veuillez a rester conforme à une architecture de type EDA.

En options : Vous pourrez également implémenter les autres « endpoints » présenté précédemment, modifier, supprimer.

Vous adapterez le projet présenté et modifierez vos codes sources de vos différents modules pour qu'ils s'adaptent parfaitement aux besoins d'un type de déploiement sous "kubernetes" et surtout reste conforme à une architecture de type EDA.

Vous allez avec tous les codes sources, construire pour chacun de vos module un fichier "**dockerfile**", permettant de "**builder**" chaque module en une seule commande de build avec docker, idéalement permettre en une seule commande de compiler l'ensemble des sources et construire les images pour un déploiement docker et permettre le déploiement "kubernetes".

Toutes les images qui seront "buildé", devront être stockée dans une "registry" (docker push registry), afin de permettre un déploiement simplifié par la suite dans "kubernetes".

Une fois ces opérations réalisées, vous allez construire les fichiers "manifest" kubernetes nécessaire qui permettront de déployer complètement votre solution dans votre environnement "minikube".

N'oubliez pas la base de données et le bus "MOM" type kafka ou autres qui devront faire partie du projet de déploiement "kubernetes".

Pour cela vous travaillerez au niveau pods, services, etc.

Pour ce projet, vous fournirez

- L'ensemble des fichiers "dockerfile multiStage".
- L'ensemble des commandes de "buid" et de "push".
- Le ou les fichiers "manifest kubernetes".
- Des captures d'écrans dans un document PDF démontrant le bon déroulement et fonctionnement de votre application déployé dans Kubernetes (Minikube).

Livraisons attendues

Objet : L'IA générative, oui... mais avec discernement !

Bonne nouvelle : vous pouvez utiliser des outils d'IA générative pour vos travaux pratiques. Ces technologies sont puissantes et peuvent vous aider à gagner du temps et à explorer de nouvelles idées. Mais attention, cela ne remplace pas votre réflexion personnelle.

Voici les règles essentielles :

=> Analyse critique obligatoire

Les réponses fournies par l'IA ne sont pas toujours exactes. Vous devez vérifier, corriger et valider chaque information en vous appuyant sur ce que nous avons vu en cours et rester dans ce cadre.

=> Compréhension avant tout

L'objectif n'est pas d'avoir une solution « prête à l'emploi », mais de comprendre les mécanismes. Si je vous demande d'expliquer votre travail, vous devez être capables de le faire sans l'IA.

=> Responsabilité

L'IA est un outil, pas un pilote automatique. Vous restez responsables de la qualité et de la pertinence de votre travail.

En résumé : utilisez l'IA pour aller plus loin, pas pour éviter de réfléchir. C'est votre curiosité et votre esprit critique qui feront la différence.

Bon courage et soyez créatifs... mais toujours rigoureux !

Date limite de remise le : Vendredi 16 Janvier 2026

Remise : Les fichiers sources, dockerfile, Yaml et les rapports PDF (comportant les captures d'écrans et vos remarques) via un Git public ou privé(*) adresse du Git qui sera retransmis par mail aux adresses tondeur.herve@yahoo.fr & herve.tondeur@uphf.fr avec les noms des développeurs dans le markdown README qui accompagnera obligatoirement ce projet.

(*) Adresse d'invitation pour le github « tondeur-h » tondeur.herve@yahoo.fr