Laboratoire 1, partie 2 :

Analyse de performance

Tel que discuté dans le cours, un des aspects clés de l'approche DevOps dans la recherche d'amélioration continue est la **rétroaction** basée sur la télémétrie, tant au niveau du produit développé que du processus de développement.

Ce laboratoire se concentre sur la mise en place de télémétrie au niveau du processus. Il a pour objectif de développer une (ou plusieurs) application qui permet d'analyser les performances de l'équipe à partir des informations contenues dans le tableau Kanban et les "Pull Requests" (PR) en utilisant l'API GraphQL de GitHub. Les différentes mesures (métriques) [1] produites par cette application ont pour objectif d'aiguiller l'équipe dans l'amélioration de leurs processus de développement.

Pour ce faire, vous devez trouver des valeurs pertinentes à exposer quant aux performances de l'équipe en rapport avec leur projet.

Contenu

1. Création des applications

Vous devez développer une application avec une interface utilisateur (web, console, ou autres) ou une API permettant d'accéder à diverses données associées au tableau Kanban et aux pull-requests associés au projet. Vous pouvez utiliser le langage ou cadriciel de votre choix pour faire l'interface utilisateur.

** **IMPORTANT** :*

- *Vous devez vous assurer que votre projet est facilement exécutable.*
- *Aucun support ne sera fourni par rapport aux langages de programmation et choix technologiques.*

2. Calcul des métriques

2.1 Métriques Kanban

Vous devez implémenter des fonctions permettant l'extraction et le calcul de différentes métriques associées au tableau Kanban. Ces quatre (4) métriques devront être fournies par votre application.

- Temps (lead time) pour une tâche donnée
- Temps (lead time) pour les tâches terminées dans une période donnée
- Nombre de tâches actives pour une colonne donnée
- Nombre de tâches complétés pour une période donnée

Ces valeurs devront être affichées dans l'application.

2.2 Métriques “pull-request”

Vous devez aussi implémenter des fonctions permettant l’extraction et le calcul de différentes métriques associées aux pull-requests. Cinq (5) métriques devront être fournies par votre application, toutefois vous devrez choisir et justifier le choix de ces métriques dans un contexte DevOps.

2.3 Métrique nécessitant une couche de persistance

Vous avez aussi à implémenter la métrique suivante :

- Nombre de tâches dans chaque colonne pour une période donnée avec la date de création. (ex. quantités des colonnes filtrée avec de toutes les tâches créées entre le 1 et 7 janvier à chaque X temps)

Cette métrique n’est pas directement accessible par l’API de GitHub et nécessitera la création de votre propre logique. Par exemple, le déplacement des cartes du kanban est partiellement disponible avec l’API mais n’offre pas directement le nom des colonnes impliquées. Pour cette métrique, une stratégie impliquant une couche de persistance est une solution intéressante. Vous pouvez utiliser une base de données ou un simple fichier. De plus, votre couche de persistance peut être locale ou sur internet pour le tp1.

3. Tests et démonstration

Vous devrez faire une démonstration de votre application lors de la session suivant la remise de votre travail. L’horaire des démonstrations sera fourni une semaine à l’avance.

Votre application doit pouvoir être utilisée pour différents projets GitHub. Pour le tester et démontrer son fonctionnement (lors de la démo) vous devez utiliser, en plus de votre projet, au moins un autre projet pour avoir un échantillon de données plus grand (ex : https://github.com/orgs/ohmyzsh/projects/1). Certaines informations comme le nom des colonnes sont nécessaires au fonctionnement de votre application selon le projet. Ces informations peuvent faire partie d’un fichier de configuration ou automatiquement récupérés (plus complexe à implémenter).

4. Utilisation du Kanban et documentation

Comme mentionné dans la partie 1 du laboratoire, le Kanban doit être utilisé de façon systématique afin de suivre et documenter l’avancement du projet. La structure de branche doit être respecté avec “main”, “develop” et “features/*” (avec git flow) ou “main” et “feature” (avec github flow).

De plus, la documentation devrait être mise à jour pour refléter les changements au projet, par exemple indiquer comment exécuter le code, justifier les métriques sélectionnées, etc.

Consignes additionnelles

Les mêmes consignes qu'au premier laboratoire s'appliquent :

- La documentation faite par les membres du groupe doit être :
  - Claire;
  - Concise;
  - Pertinente;
  - Simple.
- Il est **très important d'utiliser les jalons (Milestones) pour chacune des tâches (et/ou « Pull Requests ») des laboratoires** (ex. Lab1). L'utilisation de ceux-ci simplifiera grandement les corrections puisqu'il sera possible de filtrer les tâches à partir de ce paramètre.

Rapport

Il y a un seul court rapport à rendre sur Moodle pour le TP1 (partie 1 et 2). Vous devez soumettre un document qui comprend une introduction, une description du projet, une explication des décisions (ex. justification du choix des métriques, comment chaque métrique est calculée, etc.) et une conclusion. Le rapport peut faire référence à votre wiki / documentation dans GitHub.

Évaluation

L'évaluation du travail sera basée sur les critères suivants :

- Métriques Kanban
- Métriques "pull request"
- Métriques nécessitant une couche de persistance
- Documentation et rapport
- Utilisation du Kanban, tâches et "pull requests"
- Cohérence et pertinence du projet*

* *Ces points permettent d'évaluer plus en détail la qualité, la cohérence, la pertinence du projet dans sa totalité. Par exemple, est-ce que les bonnes pratiques sont utilisées ou est-ce que le projet est utilisable lors de la remise.*