WorkShop Cloud Campus Campus DevOps Laboratoire 1 :

Découverte et configuration de l'environnement GitHub

Ce laboratoire a pour objectif de découvrir l'environnement de développement GitHub qui sera utilisé pour supporter votre travail durant les trois laboratoires du cours.

Dans ce laboratoire, vous allez découvrir les différentes possibilités qu'offre GitHub sur le suivi de projet, mais vous allez aussi avoir la possibilité de découvrir GraphQL, le langage de requêtes dévoilé par Facebook en 2015.

Afin de vous initier à l'utilisation de ces outils, vous aller devoir mettre en place une structure basique de projet que vous allez ensuite utiliser pour extraire de l'information à l'aide d'une interface graphique qui a été préalablement développée : https://github.com/MarioGith/metricstemi.git.

Puis, dans la deuxième partie de ce laboratoire, nous allons nous intéresser à la mise en place de télémétrie au niveau du processus. Cela a pour objectif de développer une (ou plusieurs) application qui permet d'analyser les performances d'un projet à partir des informations contenues dans le tableau Kanban et les "Pull Requests" (PR) en utilisant l'API de GitHub. Les différentes mesures (métriques) produites par cette application ont pour objectif d'aiguiller l'équipe dans l'amélioration de leurs processus de développement.

Prérequis

- Le laboratoire demande d'avoir un compte GitHub
- L'étudiant devra, pour les laboratoires, avoir des bases avec l'utilisation du système de contrôle de version **Git**.
- L'étudiant doit avoir une connaissance de l'utilisation du langage de balisage Markdown.

Ressources

Le lien ci-dessous fourni un ensemble de ressources (écrites et vidéos) utiles pour les laboratoires : https://itsemi.notion.site/LOG-680-32bf411835ff4566991ff11d1762c2b3.

Des ressources additionnelles seront ajoutées durant la session selon les besoins (et questions) des étudiants.

Si vous avez quelconques recommandations, merci d'en faire part au chargé de laboratoires qui les ajouteras.

Lien du Discord afin d'échanger avec le chargé de laboratoire : https://discord.gg/vrK5x7Qr.

Contenu

1. Création des répertoires GitHub

La première étape consiste à mettre en place deux répertoires pour y déposer le code source des projets.

Une personne de chaque équipe devra créer les repos GitHub et y ajouter ses coéquipiers et le chargé de laboratoire. Le nom de votre repo GitHub doit minimalement contenir votre numéro de groupe et d'équipe (ex : **metrics-eq1** et **hvac-eq1**). Assurez-vous de créer des repos privés.

Nous vous proposons d'utiliser le code du projet GitHub suivant pour votre repo GitHub metrics : https://github.com/MarioGith/metricstemi.git.

Les repos privés PRO permettent l'automatisation des politiques de branches, si cette fonctionnalité vous intéresse. Cependant, les repos basés sur une organisation (plutôt qu'un compte individuel) ne permettent pas l'automatisation de branche, même avec un compte PRO étudiant.

Vous êtes libre de choisir le type de repo (organisation ou individuel) que vous voulez du moment qu'il soit privé. Les comptes étudiants PRO ne sont pas obligatoires.

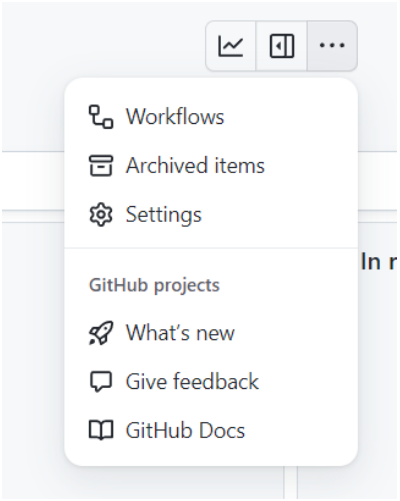
Néanmoins, je vous suggère fortement de les utiliser car ils offrent de nombreux avantages, et cela en dehors de la suite GitHub (e.g. Digital Ocean, Microsoft Azure, JetBrains, Namecheap…).

2. Création d'un projet et du tableau Kanban dans GitHub

Ensuite, vous devez créer un projet dans GitHub (voir onglet « Projects »). Ce projet vient avec un tableau Kanban. Celui-ci sera utilisé durant la session.

Le tableau doit comporter **au moins cinq colonnes** :

- Backlog
- À faire
- En cours
- Revue
- Terminée



Vous devez ajouter des "déclencheurs" ("trigger") sur la colonne Backlog lors de l'ouverture de nouvelles tâches.

Vous devez aussi ajouter des "déclencheurs" ("trigger") sur la colonne Terminée lors de la fermeture de tâches.

GitHub continuant d'évoluer avec son temps, les déclencheurs sont maintenant remplacés par des workflows plus visuels que vous trouverez directement dans vos projets.

3. Ajout de modèles et d'étiquettes

Pour le projet, vous devrez ajouter des modèles ("template") pour les tâches, des modèles pour les « Pull Requests », et des étiquettes (labels) pour les tâches.

Le projet devra comporter **au moins deux modèles pour les tâches** et **un modèle pour les « Pull Requests »**. Le projet doit aussi avoir **un nombre approprié d'étiquettes**. Il est possible de les utiliser pour définir des priorités, des types de tâches, etc. **Vous devez justifier vos choix** pour que les modèles et étiquettes soient pertinents au projet.

4. Création de tâches

À partir de maintenant, vous devez utiliser votre Kanban pour la gestion de votre projet. Pour ce faire, vous devez créer une tâche ("issue" dans GitHub) dans votre Kanban pour chacune des tâches de votre de projet. Vous devez utiliser les modèles ajoutés au préalable. Bien que les tâches **doivent respecter le modèle utilisé**, il n'est pas nécessaire de mettre une description exhaustive pour celle-ci.

5. Politique de branches

Vous devez adopter une politique de branches pour votre repo GitHub. Nous suggérons l'utilisation de politique de branches connues dans l'industrie :

https://www.atlassian.com/git/tutorials/comparing-workflows/gitflow-workflow

https://nvie.com/posts/a-successful-git-branching-model

https://guides.github.com/introduction/flow/

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.

6. Création de documentation

Vous devrez documenter les différents aspects de votre projet durant la session. Pour ce faire, vous pouvez utiliser le fichier README.md du projet et créer un répertoire appelé *Wiki* contenant la documentation dans différents fichiers en format Markdown. Ces fichiers doivent contenir toutes les diverses informations pertinentes au projet.

7. Calcul des métriques

7.1 Métriques Kanban

Des métriques peuvent être tirée de

Ces quatre (4) métriques devront être fournies par votre application, mais par soucis de temps, elles sont déjà implémentées dans le projets fournis plus haut :

- 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

7.2 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.

8. Consignes additionnelles

- 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.

- Bien que GitHub propose une liste (générique) de "templates" et "labels", ceux-ci ne sont pas tous pertinents aux besoins spécifiques de votre projet. Vous devez créer une liste de "templates" et "labels" qui sont adaptés aux besoins de votre projet, qui peut inclure certains proposés par GitHub.

- Imaginez que votre projet est *OpenSource* et qu'une nouvelle personne peut s'intégrer au projet facilement. Tout doit être documenté dans les fichiers Markdown du *Wiki* : comment exécuter le projet, libellé, modèles, structure de branche, etc.