Université de Bordeaux
Semestre 2

2021-2022

TD8 : Intégation continue

Compétences

- Découvrir le principe d'intégration continue en génie logiciel
- Mise en oeuvre de l'intégration continue sur GitLab
- Savoir lire les instructions élémentaires d'un fichier yml

Introduction

Lecture avant de commencer :

- https://fr.wikipedia.org/wiki/Int%C3%A9gration_continue

Référence pour GitLab :

- https://docs.gitlab.com/ee/ci/

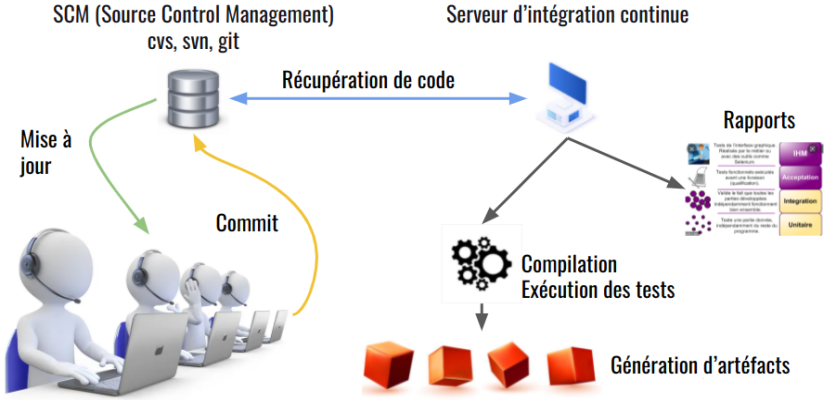
Avant l'intégration continue

Autrefois, les développeurs d'une même équipe avaient tendance à travailler séparément pendant de longues périodes et à n'intégrer leurs modifications au référentiel centralisé qu'après avoir fini de travailler. Cela rendait la fusion des différents codes difficile et chronophage. Cette méthode entraînait également des bogues difficiles à détecter. La combinaison de ces différents facteurs empêchait de livrer rapidement des mises à jour au propriétaire.

Avec l'intégration continue

L'intégration continue a pour but de tester l'ensemble du code déjà écrit. À mesure que le développement logiciel se poursuit, à chaque publication de code, des tests sont réalisés en parallèle du developpement et ce de manière continue.

Comment fonctionne l'intégration continue ?

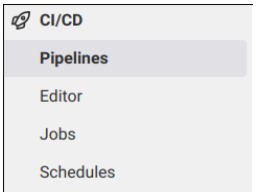


Explication du fonctionnement de l'intégration continue :

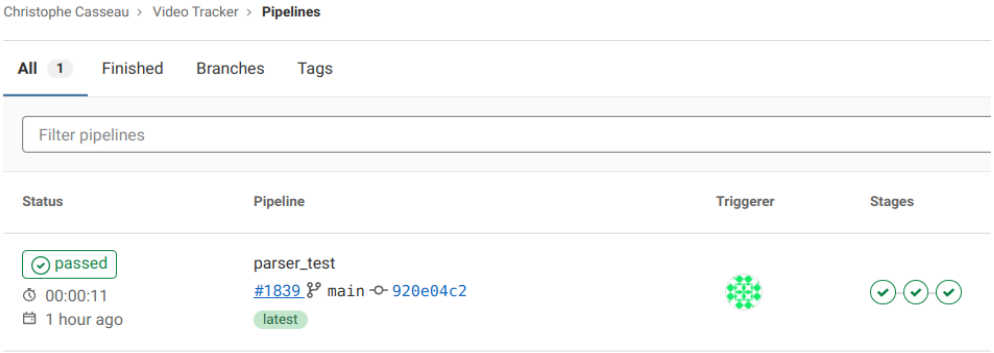
- Le développeur code les tâches qui lui ont été attribuées. Il réalise ensuite les tests unitaires correspondants afin de s'assurer que tout fonctionne correctement dans son environnement. Il fusionne son code avec celui du dépôt distant et résoud les eventuels conflits, teste de nouveau son code sur sa machine et apporte des corrections si nécessaires. Si tout est ok, il publie alors son code sur le dépôt distant.
- Le serveur d'intégration continue associé au dépôt distant détecte le commit et lance les opérations de compilation, de tests et de couverture. Ces opérations sont décrites sous forme de **jobs**. Les jobs sont associés à des **stages** : build, test, coverage
- Une fois les jobs terminés, une notification est envoyée à tous les membres de l'équipe. Elle contient notamment le statut du dernier *build and test*.
- L'équipe peut alors consulter les rapports émis lors de ce dernier build, les analyser, traiter des bugs s'il y en a, puis continuer à développer les autres phases du projet.

Intégration continue avec GitLab CREMI

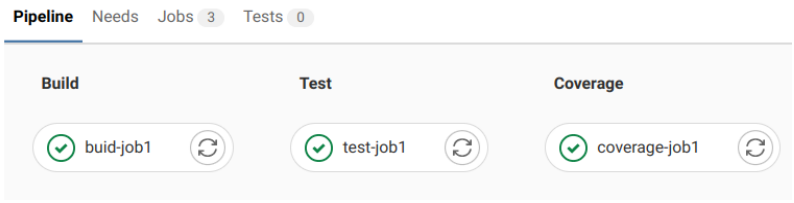
Sur le bandeau gauche de votre projet GitLab vous avez un menu CI/CD. Ce menu va nous permettre de mettre en place un ensemble de tests unitaires et d'intégration pour s'assurer que le code ajouter au projet répond bien à toutes les attentes des différents scénarios conçues pour évaluer le code.



1. Télécharger l'archive `src-sudent.zip`
2. Créer un nouveau projet GitLab pour ce TD par exemple projetTD8
3. Cloner le projet sur ordinateur et y copier le contenu de l'archive `src-student.zip`
4. Décrire précisément chacune des lignes du fichier `.gitlab-ci.yml`
5. Lire le code contenu dans l'archive et indiquer en quelques lignes ce qu'il est censé faire.
6. Faire votre premier commit et mettre à jour le dépôt distant.
7. Cliquer sur le menu d'intégration continue **CI/CD**, puis choisir *Pipelines*. L'image ci-dessous montre le résultat d'un commit sur le projet. On peut voir que tous les jobs ont été réalisés avec succès.



Pour accéder aux rapports des différents jobs, cliquer sur *passed*. Vous devriez obtenir la fenêtre suivante. On y retrouve les noms des jobs que nous avons déclaré dans le fichier `.gitlab-ci.yml`. Il n'y a plus qu'à choisir votre job pour obtenir le rapport. Chaque job peut également être relancé de manière indépendante.



8. À l'aide des rapports indiquer :
   - Le nom du serveur utilisé pour l'intégration continue.
   - La version de Python utilisée ainsi que celle du package coverage.
   - Le pourcentage de couverture de code par fichier.

9. Dans le fichier `test_parser` de votre dépôt local, décommenter le dernier test. Ce test n'est pas complètement fonctionnel. Ne pas corriger le code dans un premier temps. Faire un commit puis un push sur GitLab. Vérifier l'état de votre *Pipelines* dans le menu **CI/CD** et relever la ou les erreurs consignées dans les rapports.
10. Sur votre dépot local corriger ces erreurs jusqu'à ce que l'ensemble des tests soit fonctionnel et faire un nouveau commit.
11. Mettre en oeuvre le principe d'intégration continue sur le projet Video Tracker