TP MGPI- CI/CD GitLab

L'objectif de ce TP est de mettre en place une CI/CD sur la forge gitlab de l'établissement. Cette CI validera des tests unitaires puis un audit statique de code et le CD produira la documentation au format HTML.

Ce TP sera réalisé en binome. Vous n'utiliserez aucun IDE, toute action `git` sera réalisée depuis l'invite de commandes. L'un des étudiants, désigné par AA, sera l'administrateur du projet. L'autre étudiant, désigné par BB, sera développeur. AA et BB seront les initiales (Prénom, Nom) des membres du binome. Vous aurez besoin de deux sessions de navigateur Web, l'un pour AA et l'autre pour BB. De même vous aurez deux terminaux d'ouvert, l'un avec une session de AA, l'autre de BB [1].

Suivez les procédures suivantes, où chaque action commence par les initiales de la personne qui la réalise.

1 Mise en place du projet

AA Créez un projet (vide) sur la forge en respectant la nomenclature suivante pour le nom du projet : `MGPI-TP-CICD-XX-YY-AA-BB`, tel que `XX-YY` représente l'année universitaire en cours (par exemple `22-23`) ;

AA Activez l'intégration continue pour la projet (Paramètres › Général › Visibilité, fonctionnalités du projet, autorisations › CI/CD et surtout enregistrer) ;

AA Ajoutez BB comme développeur ;

AA et BB Si ce n'est déjà fait, enregistrez les clés SSH (au moins de BB) pour éviter de devoir taper constamment votre login et votre mot de passe, vous trouvez de la documentation à ce sujet sur :
— `https://wiki.insa-rouen.fr/doku.php?id=insa:gitlab:authentification`
— `https://gitlab.insa-rouen.fr/help/user/ssh.md`

BB Clonez votre projet en SSH ;

AA Ajoutez un ticket (*issue*) « initialiser le projet » en l'assignant à BB ;

AA Créez, depuis le site, la branche correspondante sur le serveur ;

BB Mettez vous localement sur cette branche ;

BB Créez un fichier `.gitignore` en le complétant des gitignore de référence nécessaires (python, votre IDE, etc.) : `https://github.com/github/gitignore`

BB Téléchargez l'archive disponible sur moodle, et ajoutez les fichiers suivant à la racine de votre projet :
— `huffman/compteur.py` une partie de la logique métier
— `tests/test_compteur.py` des tests unitaires
— `.pylintrc` le fichier de configuration de pylint, l'analyseur de code

1. Utilisez la commande `su`

— `requirements.txt` les fichiers permettant d'installer facilement les dépendances

Commitez [2] et pushez ces modifications ;

BB Effectuez la demande de fusion avec la branche *main* ;

AA Réalisez cette fusion à partir du terminal et supprimer la branche ;

AA Fermez le ticket.

2 Mise en place de l'intégration continue

AA Sur la branche *main*, depuis l'éditeur du menu CI/CD de gitlab, configurez l'intégration continue (fichier `.gitlab-ci.yml`) tel que :

— l'image de référence sera `python:3.10` ;

— il y aura un *stages* `tests` pour les tests ;

— trois *jobs* seront créés :

.config qui permettra d'initialiser l'environnement python en installant toutes les dépendances (`pip install -r requirements.txt`);

pylint qui étend le job `.config`, associé au stage `test` et qui exécute la commande : `pylint --rcfile=.pylintrc huffman`;

pytest qui étend le job `.config`, associé au stage `tests` et qui exécute la commande : `python -m pytest`

Cette intégration devrait échouer ;

AA Créez un nouveau ticket pour résoudre les problèmes d'intégration continue en l'assignant à BB ;

BB Créez une nouvelle branche pour résoudre les problèmes. Une fois terminé, demandez la fusion avec la branche *main* ;

AA Réalisez la fusion à partir de l'interface Web en supprimant la branche et fermez le ticket.

3 Mise en place du déploiement continu

Pour la branche `main`, nous allons générer automatiquement la documentation HTML du module *compteur.py* à l'aide *pydoc*. Cette documentation sera disponible à l'adresse suivante : `https://zz.pages.insa-rouen.fr/mgpi-tp-cicd-XX-YY-aa-bb/compteur.html` (tout en minuscule) ou `zz` représente *login* de AA.

AA Ajoutez un nouveau *stage* ;

AA Créez un *job* nommé *pages*, associé à ce *stage*, tel que :

— le script doit effectuer les actions suivantes :

```
- mkdir .public
- pydoc -w huffman/compteur.py
- mv *.html .public/
- mv .public public
```

— le résultat est un *articfact* qui sera généré dans un répertoire `public` avec une durée de vie de 7 jours :

2. Tout *commit* sera accompagné d'un message qui commencera par # suivi du numéro de l'*issue*

```
  artifacts:
    expire_in: 7 days
    paths:
      - public
```

4 Suppression du projet

Une fois le TP terminé, supprimez votre projet : Settings > General > Advanced > Delete this project