TP : Intégration Continue

Part 2 : Tests unitaires sur Jenkins & GitLab

Objectif :

Intégration continue avec Jenkins et GitLab.

Protocole :

Le but de ce TP est d'appliquer l'intégration continue aux projets **JAVA** et **PHP** avec **Jenkins** et **GitLab**.

Exercice I : Lancement des tests unitaires sur Jenkins

Lancement des tests unitaires pour des projets **Java** / **MAVEN** sur **Jenkins**. Et l'automatisation des lancements des builds sur **Jenkins**.

Exercice II : Lancement des tests unitaires pour une BDD de Cours & Prof

Définition des tests unitaires pour un projet **PHP** de gestion des profs et des cours sur **GitHub**. Et lancement des tests unitaires sur **Jenkins**.

Exercice III : Lancement des tests unitaires sur GitLab

Lancement des tests pour un projet **MAVEN** sur **GitLab**. Et création du fichier « **.gitlab-ci.yml** » décrivant les actions à exécuter après chaque commit

Exercice I : Lancement des tests unitaires sur Jenkins

- **Importation d'un projet MAVEN dans GitHub.**
- **Lancement des tests unitaires depuis Jenkins.**
- **Automatisation des lancements des builds sur Jenkins.**

Enoncé :

L'objectif de cet exercice est de lancer les tests unitaires des projets **Java** / **Maven** développés lors du TP1 mais cette fois-ci depuis **Jenkins**.

Partie 1 : Création des jobs et lancement des builds sur Jenkins

Tout d'abord, nous allons créer un référentiel (**Repository**) sur **GitHub** pour pousser (push) le code du projet Java **Calculs_Geo** développé dans le TP1 (**Codes de base calcul géo** sur moodle). Ensuite nous allons lier ce projet à un job **Jenkins** dans lequel seront lancés les tests unitaires.

Question 1 : Se connecter à GitHub et créer un nouveau référentiel (Repository) vide sans README en mode **public**.

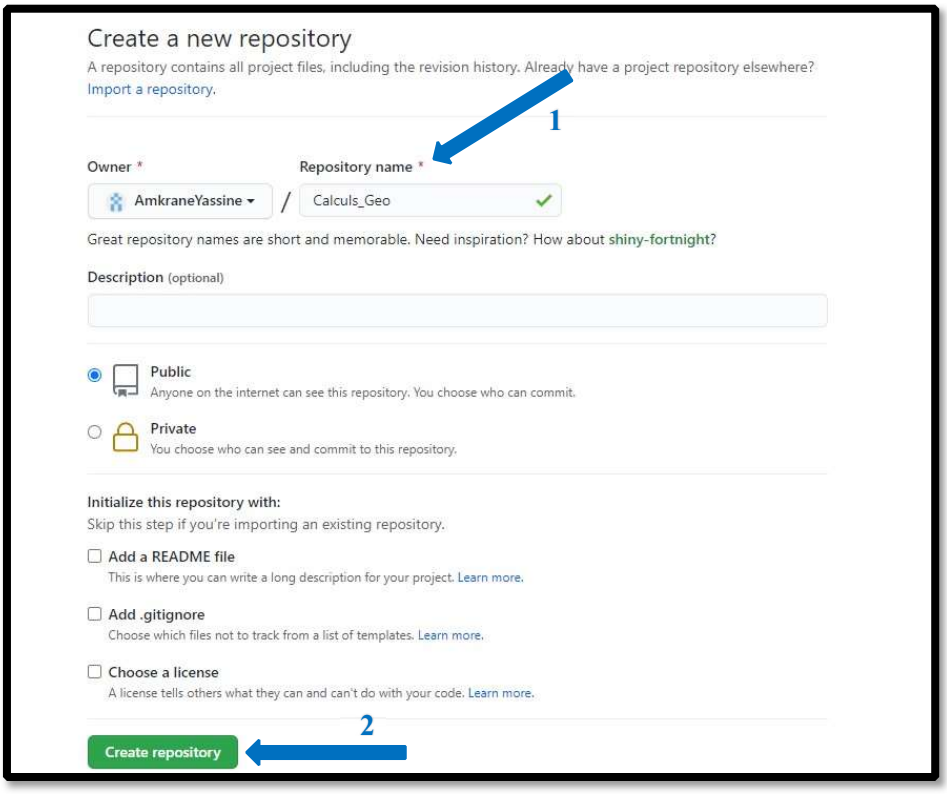

Figure 1 : Création d'un projet vide sur GitHub

Question 2 : Suivez les instructions GitHub pour pousser le code du projet sur le repository. **Note :** sous Windows, On pout utiliser Git Bash (https://gitforwindows.org/)

Les commandes doivent être lancées dans le répertoire du projet

Une fois fait, actualisez la page GitHub et vérifier si le projet a bien été envoyé.

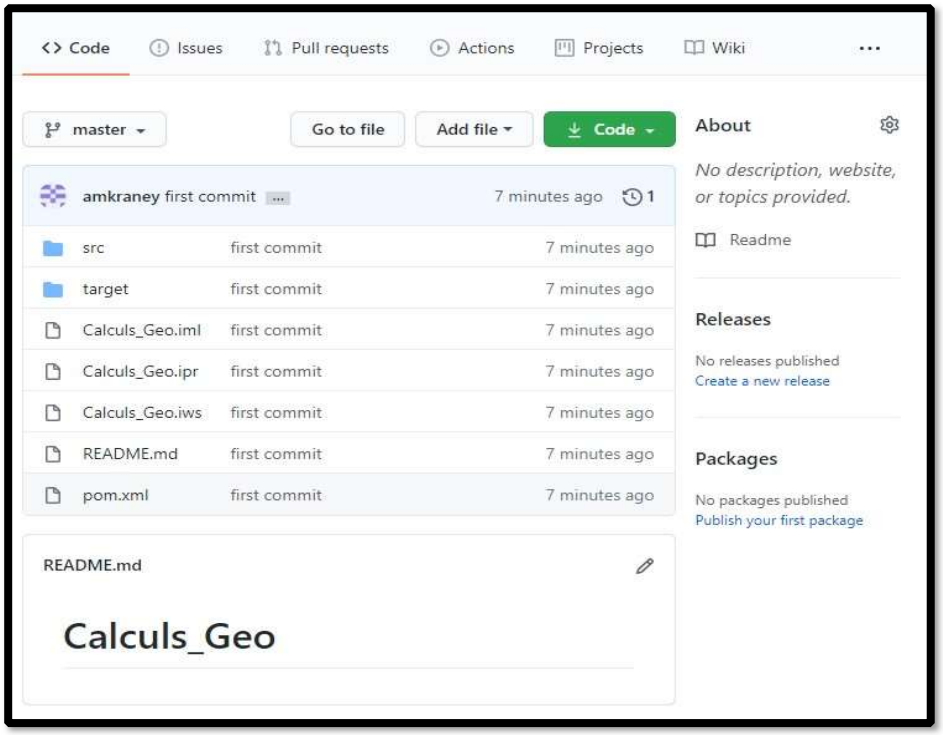

Figure2 : Dépôt du projet dans GitHub

Question 3 : Connectez-vous au site : https://jenkins.ig.umons.ac.be/. Votre login et mot de passe sont envoyés par email (par **Adriano GUTTADAURIA**).

Question 4 : Créez un job Jenkins et indiquez le lien du projet sur GitHub.

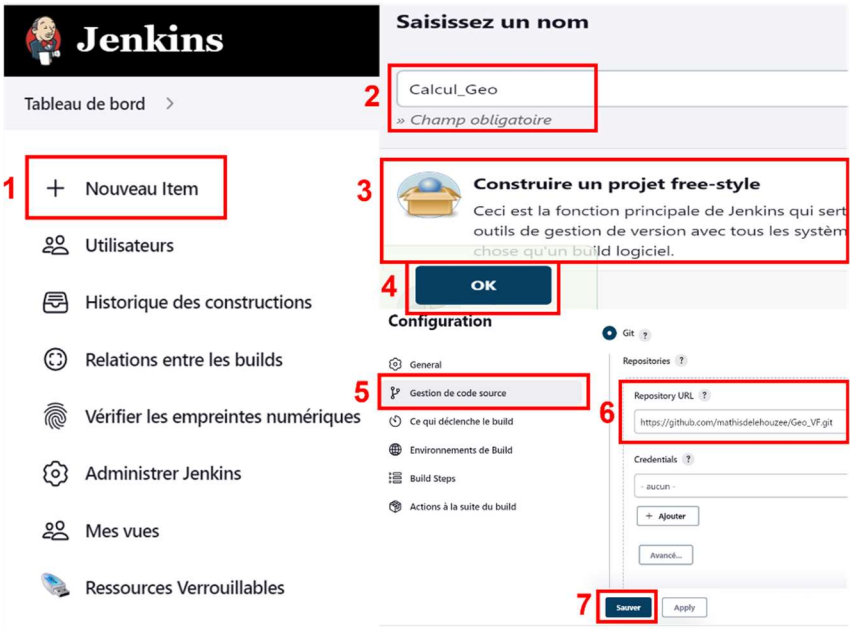

Figure 3 : Création d'un job Jenkins

Question 5 : Allez sur l'onglet « **Build steps** » pour « **Invoquer les cibles Maven**».

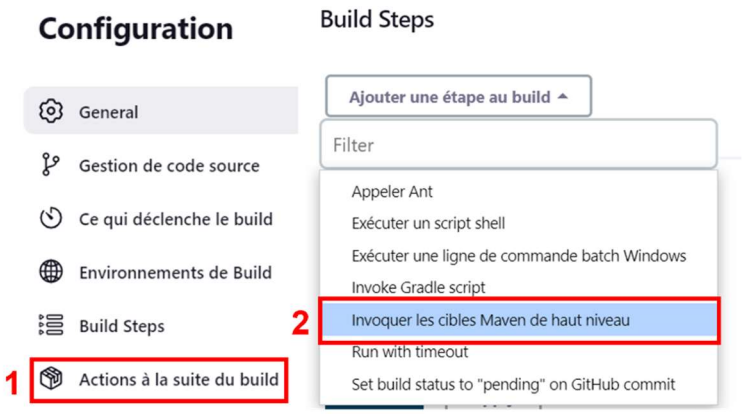

Figure 4 : Ajout d'une étape au build

Question 6 : Choisir la version de Maven (**3.8.6)** et invoquer « **test »** comme « **cibles Maven** ».

Ensuite, appliquer et sauver les configurations.

***Figure 5** : Invocation des cibles Maven et sauvegarde de la configuration*

Question 7 : Démarrez le job Jenkins en lançant un build. Puis aller dans le build. (Voir la figure 6).

Question 8 : Consultez les résultats en cliquant sur le build lancé. (Voir la figure 6).

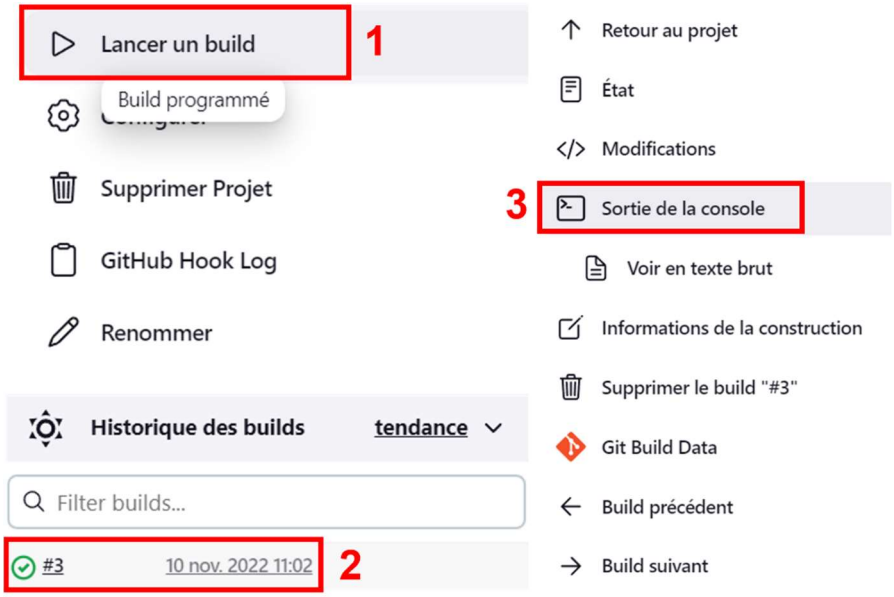

Figure 6 : Lancement d'un build

- La figure 7 montre un exemple de résultats obtenus.

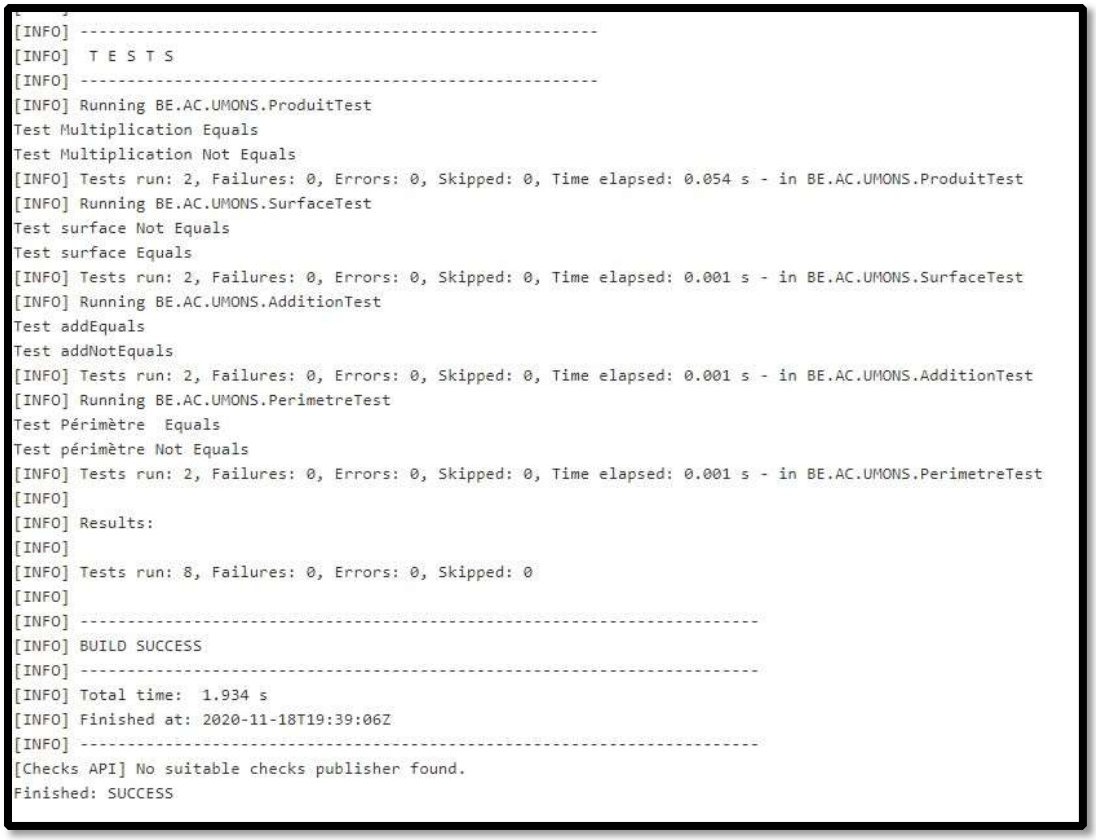

***Figure 7** : Résultats des tests unitaires sur Jenkins*

Partie 2 : Automatisation de lancement des tests unitaires sur Jenkins

Cette partie permet d'automatiser le lancement des tests unitaires lorsqu'une modification est apportée au projet (un nouveau push vers GitHub). C'est ce que nous allons appliquer au projet **Maven** de gestion des profs et cours développé dans le TP1 (Cours Prof Maven sur moodle).

Question 1 : Connectez-vous à GitHub et créez un nouveau référentiel (Repository) vide sans README en mode **public.**

Question 2 : Suivez les instructions GitHub pour pusher le code du projet **Maven** sur Github.

Question 3 : Connectez-vous sur le site : https://jenkins.ig.umons.ac.be/.

Question 4 : Créez et configurez un job Jenkins avec l'url du projet sur GitHub.

- Dans «Build Step » choisissez la version de Maven (**3.8.6)** et invoquez

« **-Dtest=Ig19Suite test » comme Cible.** Ensuite, appliquer et sauver les configurations.

Question 5 : Démarrez le job Jenkins en lançant un build. Puis consultez les résultats en cliquant sur le build lancé.

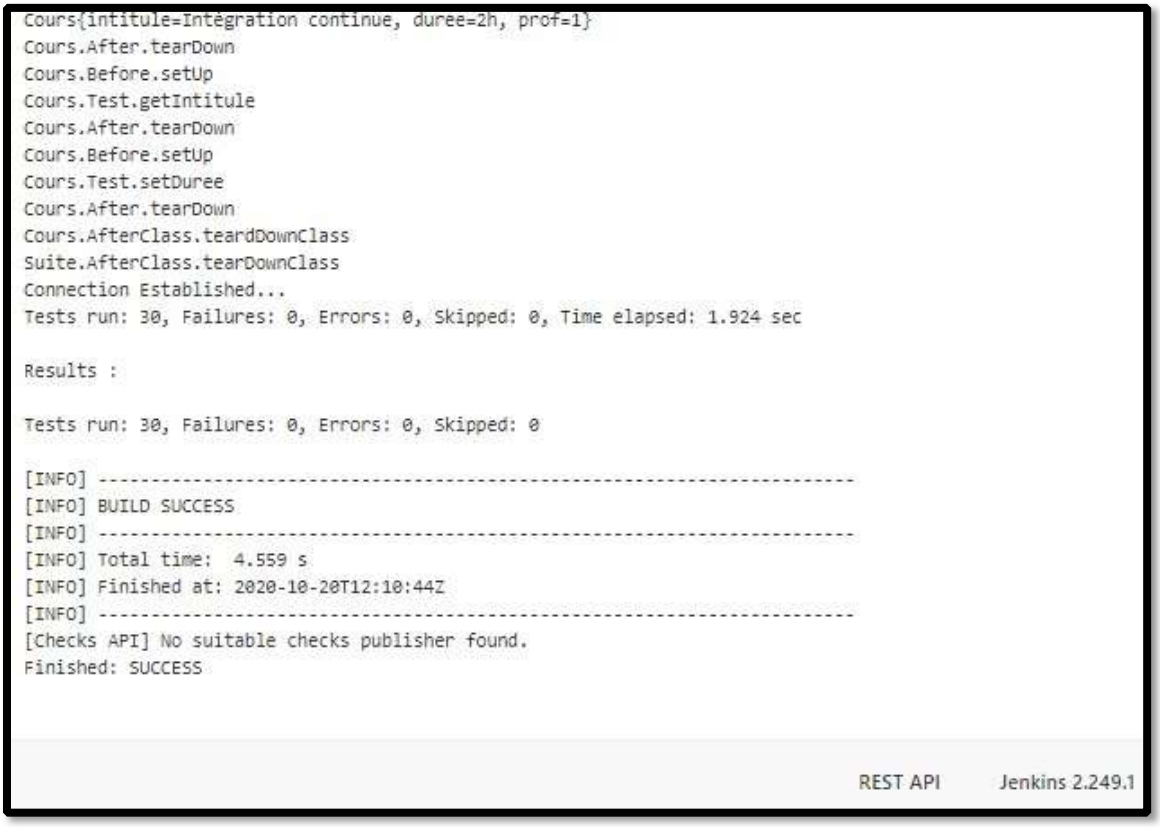

***Figure 7** : Résultats des tests unitaires sur Jenkins*

Question 6 : Automatisez le lancement des builds en suivant les étapes suivantes :

1- Dans le projet que nous avons créé sur GitHub, dans la première partie, cliquez sur **Settings**.

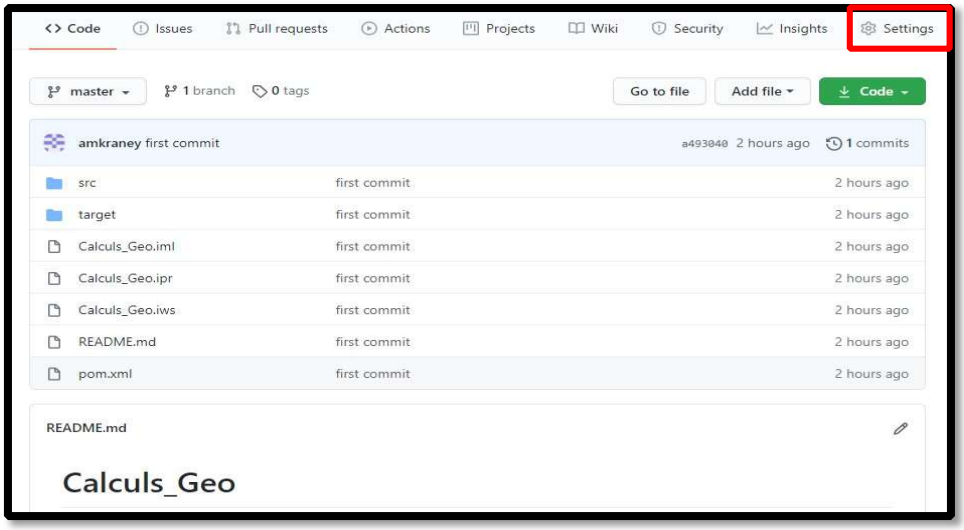

Figure 8 *: Tests unitaires automatiques (1/5)*

2- Cliquez sur **Webhooks**.

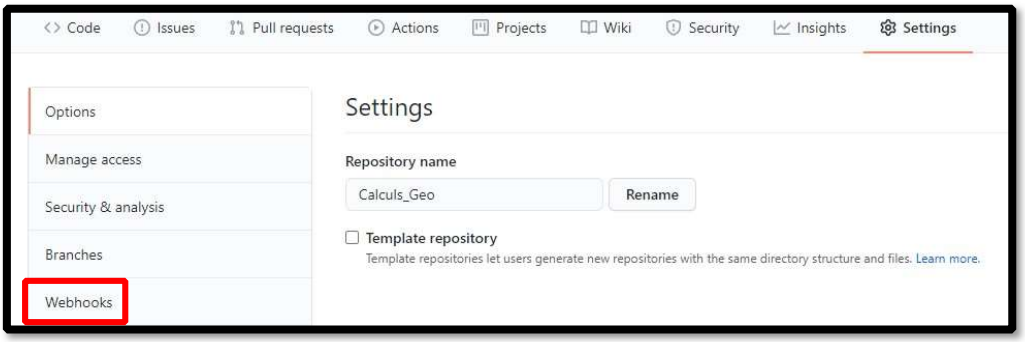

Figure 9 *: Tests unitaires automatiques (2/5)*

3- Cliquez sur **Add webhook**.

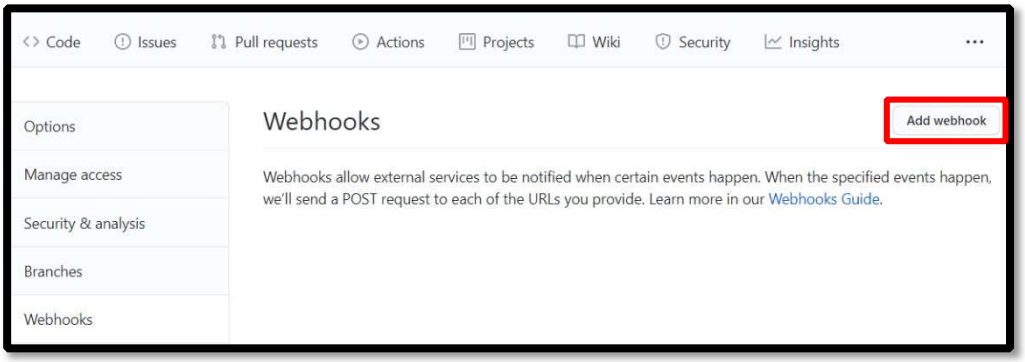

Figure 10 *: Tests unitaires automatiques (3/5)*

4- Tapez https://jenkins.ig.umons.ac.be//github-webhook// dans « **URL de la charge utile** » pour connecter **GitHub** avec **Jenkins**. Et cliquer sur **Add Webhook**.

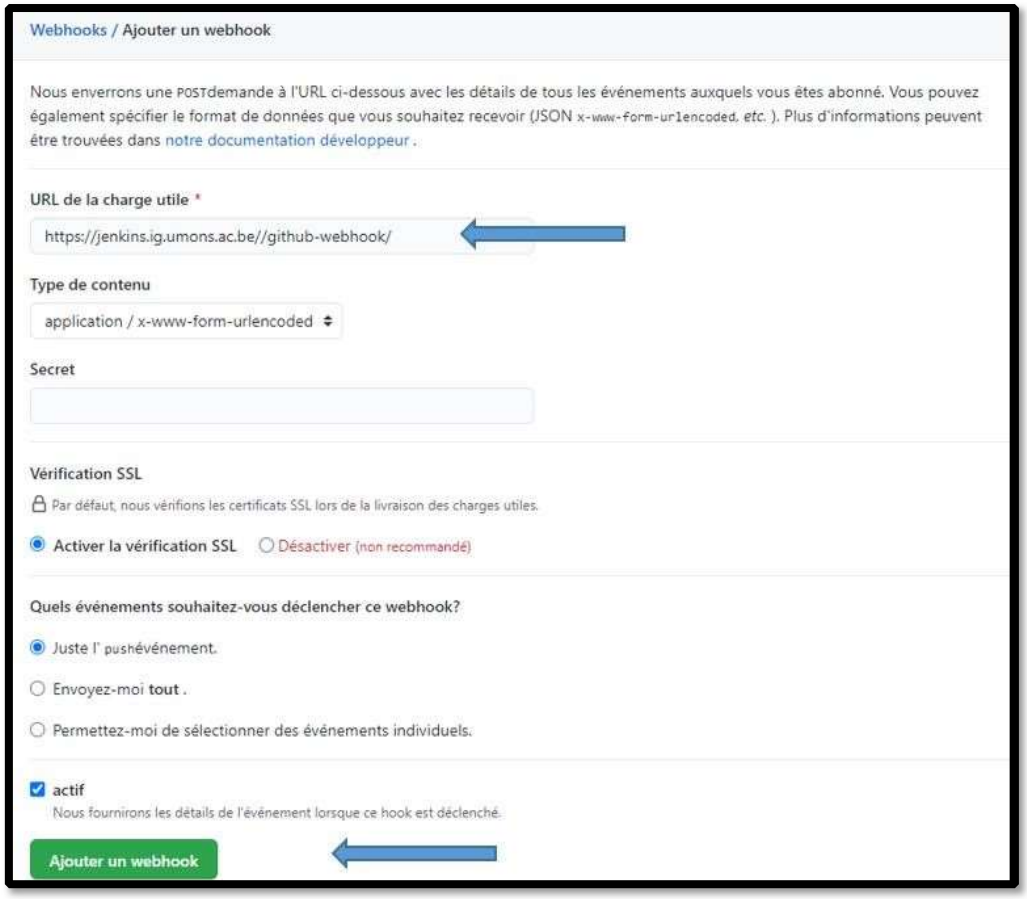

Figure 11 *: Tests unitaires automatiques (4/5)*

5- Une fois que c'est fait, allez sur https://jenkins.ig.umons.ac.be/ et sélectionnez votre projet, puis cliquez sur configurer. Dans la partie « Ce qui déclenche le buid », cochez la case : **GitHub hook trigger for GITScm polling**.

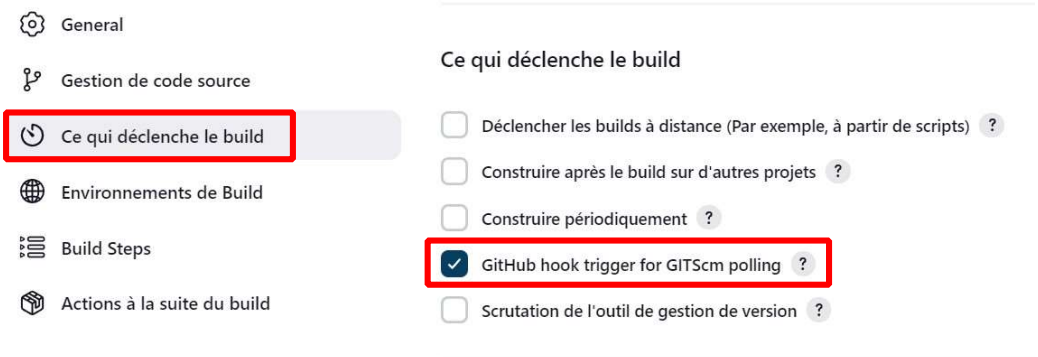

Figure 12 *: Tests unitaires automatiques (5/5)*

6- Appliquez et sauvez les changements.

Maintenant, après chaque **commit** (modification du code) sur GitHub, un build se lancera automatiquement sur Jenkins. On peut consulter les résultats des builds sur Jenkins après chaque push sur GitHub (ou sur l'éditeur + push & commit).

Exercice II : Gestion d'une BDD de Cours & Prof

- **Création d'un projet PHP sur GitHub.**
- **Gestion de la base de données et lancement des tests unitaires sur Jenkins.**

Enoncé :

L'objectif de cet exercice est de finaliser un projet PHP de gestion d'une base de données des profs et cours. Il nous restera ensuite à définir les tests unitaires et les automatiser sous Jenkins.

Partie 1 : Création du projet sur GitHub et création d'un job Jenkins

Question 1 : Télécharger le projet non finalisé « **IC_PHP_PROF_COURS** » de Moodle.

Question 2 : Sur GitHub, créez un projet vide sans README en mode **public** et suivre les instructions GitHub pour pusher le code du projet « **IC_PHP_PROF_COURS** » sur GitHub.

Question 3 : Créez un job Jenkins en indiquant le lien du projet sur GitHub.

Question 4 : Configurez le job Jenkins en suivant la configuration montrée dans les figures 15-16.

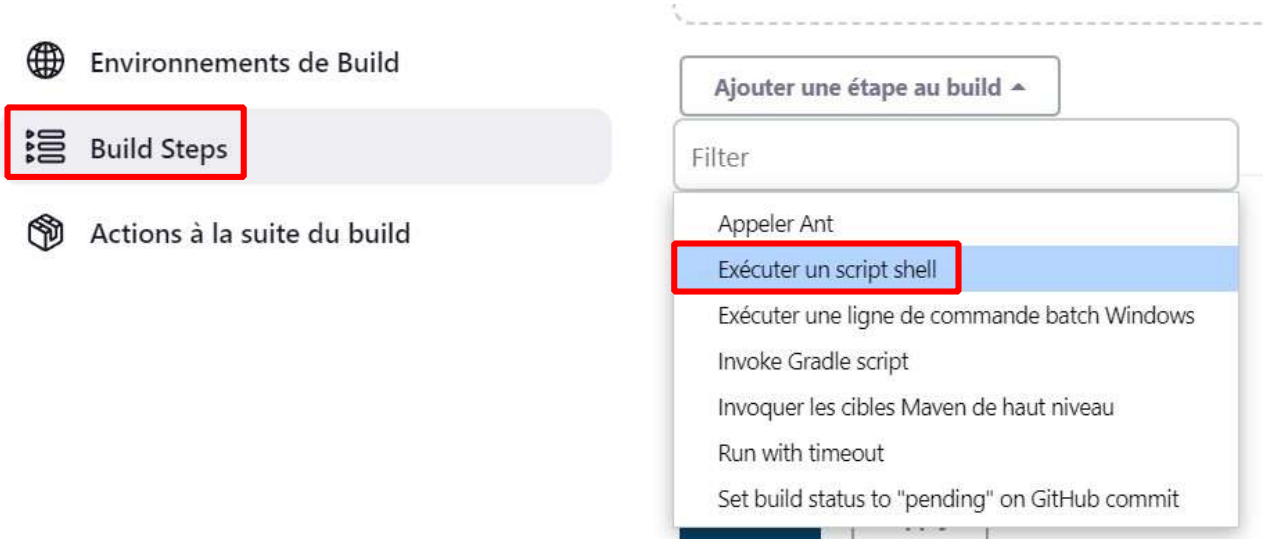

Figure 13 : Ajout d'une étape au build (1)

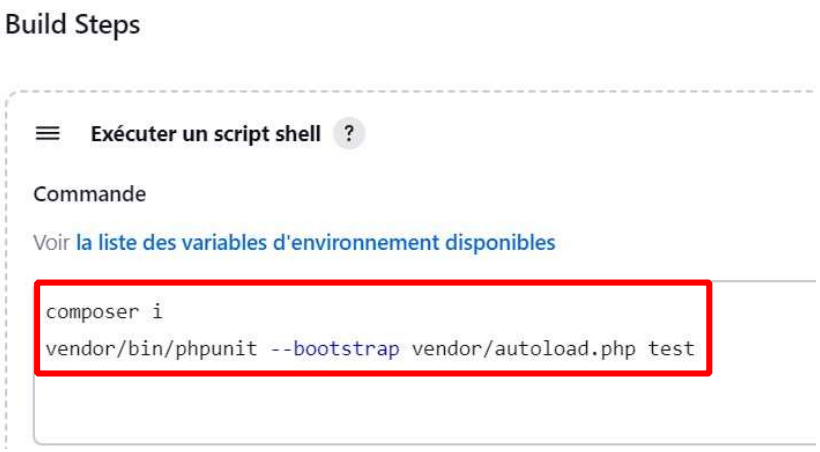

Figure 14 : Ajout d'une étape au build (2)

- Enfin, cliquez sur le bouton bleu « **Sauver** » situé en bas.

Question 5 : Démarrez le job Jenkins en lançant un build et Consulter les résultats.

```
Cours {intitule = Intégration continue, duree = 2h, prof = 1}
Cours.After.tearDown
Cours.Before.setUp
Cours.Test.select_intitule
Connection établie...
Cours {intitule = Intégration continue, duree = 2h, prof = 1}
Cours.After.tearDown
Cours.Before.setUp
Cours.Test.setIntitule
Cours.After.tearDown
Cours.Before.setUp
Cours.Test.select_id
Connection établie...
Cours {intitule = Intégration continue, duree = 2h, prof = 1}
Cours.After.tearDown
Cours.Before.setUp
Cours.Test.getIntitule
Cours.After.tearDown
Cours.Before.setUp
Cours.Test.setDuree
Cours.After.tearDown
Cours.AfterClass.teardDownClass
Suite.AfterClass.tearDownClass
Connection établie...
Tests exécutés: 30, échecs: 0, erreurs: 0, ignorés: 0, temps écoulé: 2,051 s

Résultats :

Tests exécutés: 30, échecs: 0, erreurs: 0, ignorés: 0

[INFO] ------------------------------------------------ ------------------------
[INFO] CONSTRUIRE LE SUCCÈS
[INFO] ------------------------------------------------ ------------------------
[INFO] Durée totale: 4,606 s
[INFO] Terminé à: 2020-11-05T11: 50: 16Z
[INFO] ------------------------------------------------ ------------------------
[Checks API] Aucun éditeur de chèques adapté trouvé.
Terminé: SUCCÈS
```

Figure 15 : Résultats des tests unitaires

Partie 3 : Gestion de la base de données et lancement des tests unitaires

Dans cette partie, nous allons définir les fonctions qui permettent la gestion de la base de données. Ensuite, nous lancerons les tests unitaires de ces fonctions.

Le code PHP permet de se connecter à une base de données et tester différentes requêtes (insertion, suppression, mise à jour, etc…).

Dans notre cas le MLD de la base de données est le suivant :

Prof (idProf, nom, prenom, date_naissance, lieu_naissance)

Cours (idCours, intitule, duree, #id_prof)

NB : Nous pouvons faire un commit/push après chaque modification du fichier **ProfCoursTest.php** pour vérifier que les tests soient tout le temps « **SUCCESS »**.

Question 6 : Insérez les enregistrements suivants dans la table **prof** :

Nom	Prénom	Date de naissance	Lieu de naissance
Nom_Prof1	Prenom_Prof1	10/01/1982	lieu_prof1
Nom_Prof2	Prenom_Prof2	10/02/1982	lieu_prof2
Nom_Prof3	Prenom_Prof3	10/03/1982	lieu_prof3

***Tableau 1** : Enregistrements à insérer dans la table Prof*

Question 7 : Insérez les enregistrements suivants dans la table **cours** :

Intitulé	Durée	Prof
IoT	10	1
IA	12	3
EDL	5	6

***Tableau 2** : Enregistrements à insérer dans la table Cours*

Question 8 : Dans la fonction « ***testAdd()*** », inspirez-vous du test d'ajout des profs pour tester l'ajout des cours.

Question 9 : Dans la fonction « ***testPrintAll()*** », inspirez-vous du test de la sélection et de

l'affichage des profs pour tester la sélection et l'affichage des cours.

Question 10 : Consultez les résultats sur Jenkins.

```
=> Nom_prof2 Prenom_prof2 | 10/02/1982 - lieu_prof2
=> Nom_prof3 Prenom_prof3 | 10/03/1982 - lieu_prof3
=> Nom_prof4 Prenom_prof4 | 10/04/1982 - lieu_prof4
=> Nom_prof5 Prenom_prof5 | 10/05/1982 - lieu_prof5
=> Nom_prof6 Prenom_prof6 | 10/06/1982 - lieu_prof6
=> Nom_prof7 Prenom_prof7 | 10/07/1982 - lieu_prof7
=> Nom_prof9 Prenom_prof9 | 10/09/1982 - lieu_prof9
=> TATSUM REVERGIE | 22/07/1984 - Toulouse, France
##################################################################

Suppression du cours num 7 REUSSIE.
@@@@@@@@@@@@@ - LISTE DES COURS APRES SUPPRESSION - Vérifiez le cours num 7 @@@@@@@@@@@@@
=> Intégratoin continue - 3h | 10
=> Cours2 - 2.5h | 3
=> Cours3 - 3h | 5
=> Cours4 - 2h | 3
=> Cours5 - 3h | 3
=> Cours6 - 2h | 4
=> Cours8 - 4h | 5
=> Intégratoin continue - 3h | 10
@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@

Test\ProfCoursTest::tearDown

.                                              8 / 8 (100%)Test\ProfCoursTest::setUp
Test\ProfCoursTest::testDeleteOne_2
Suppression du prof num 1 REUSSIE.
########## - LISTE DES PROFS APRES SUPPRESSION- Vérifier avec celui juste avant (1e supprimer) ##########
=> Nom_prof2 Prenom_prof2 | 10/02/1982 - lieu_prof2
=> Nom_prof3 Prenom_prof3 | 10/03/1982 - lieu_prof3
=> Nom_prof4 Prenom_prof4 | 10/04/1982 - lieu_prof4
=> Nom_prof5 Prenom_prof5 | 10/05/1982 - lieu_prof5
=> Nom_prof6 Prenom_prof6 | 10/06/1982 - lieu_prof6
=> Nom_prof7 Prenom_prof7 | 10/07/1982 - lieu_prof7
=> Nom_prof9 Prenom_prof9 | 10/09/1982 - lieu_prof9
=> TATSUM REVERGIE | 22/07/1984 - Toulouse, France
##################################################################

Suppression du cours num 1 REUSSIE.
@@@@@@@@@@@@@ - LISTE DES COURS APRES SUPPRESSION - Vérifier avec celui juste avant (1e supprimer) @@@@@@@@@@@@@
=> Cours2 - 2.5h | 3
=> Cours3 - 3h | 5
=> Cours4 - 2h | 3
=> Cours5 - 3h | 3
=> Cours6 - 2h | 4
=> Cours8 - 4h | 5
=> Intégratoin continue - 3h | 10
@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@

Test\ProfCoursTest::tearDown

Test\ProfCoursTest::tearDownAfterClass
SUPPRESSION DE LA BASE DONNEE user01_test_php REUSSIE
SUPPRESSION DES VARIABLES.
```

Figure 16 : Résultats du build

Exercice III : Lancement des tests unitaires sur GitLab

- **Création du compte GitLab et importation des projets dans GitLab**
- **Lancement des tests unitaires sur GitLab**

Enoncé :

Dans cet exercice, nous allons lancer les tests unitaires du projet **MAVEN** de transfert de monnaie développé dans le TP1 (Cours de base projet Money sur moodle) mais cette fois-ci sur **GitLab**.

Questions :

Question 1 : Allez sur le cluster IG via le lien https://gitlab.ig.umons.ac.be/.

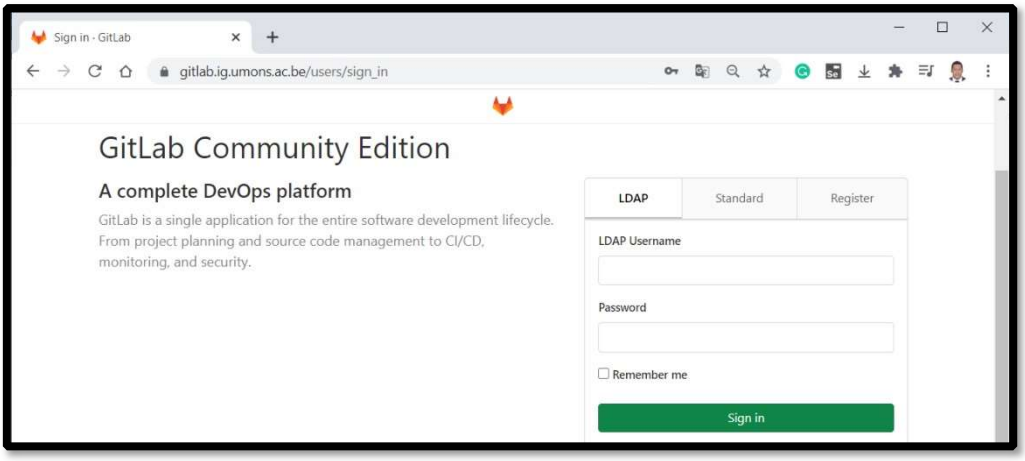

Figure 17 : Création d'un compte sur GitLab IG (1)

Question 2 : Cliquez sur l'onglet « **Register** » et saisir les informations du nouveau compte.

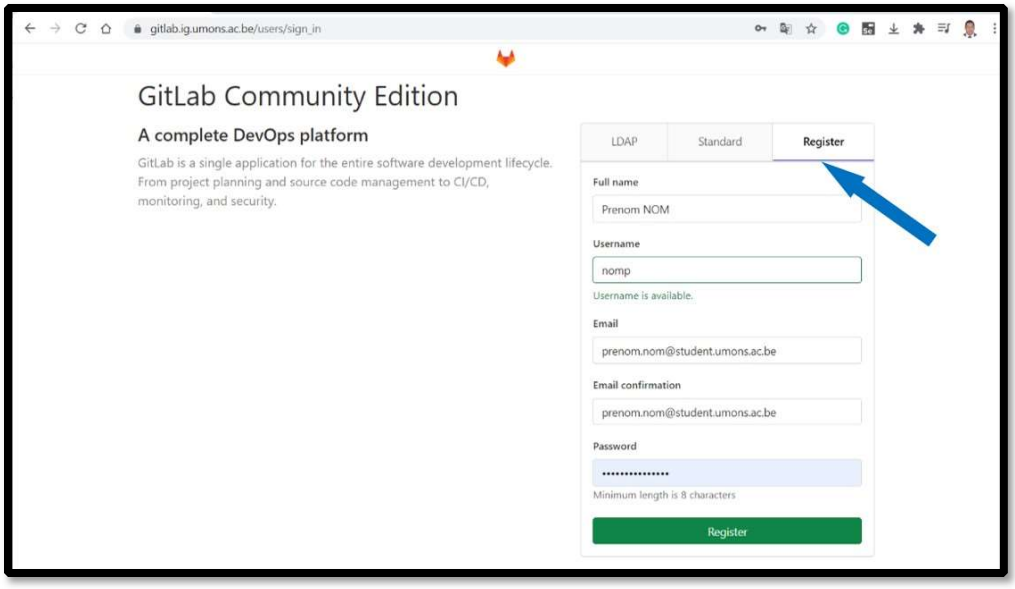

Figure 18 : Création d'un compte sur GitLab IG (2)

- Cliquez sur le bouton « **Register** » pour créer le nouveau compte.
- Un mail de confirmation du compte est envoyé.

- Dans votre boite mail, confirmez le compte en cliquant sur le lien « ***Confirm your account*** ».

Question 3 : Connectez-vous maintenant à **GitLab**.

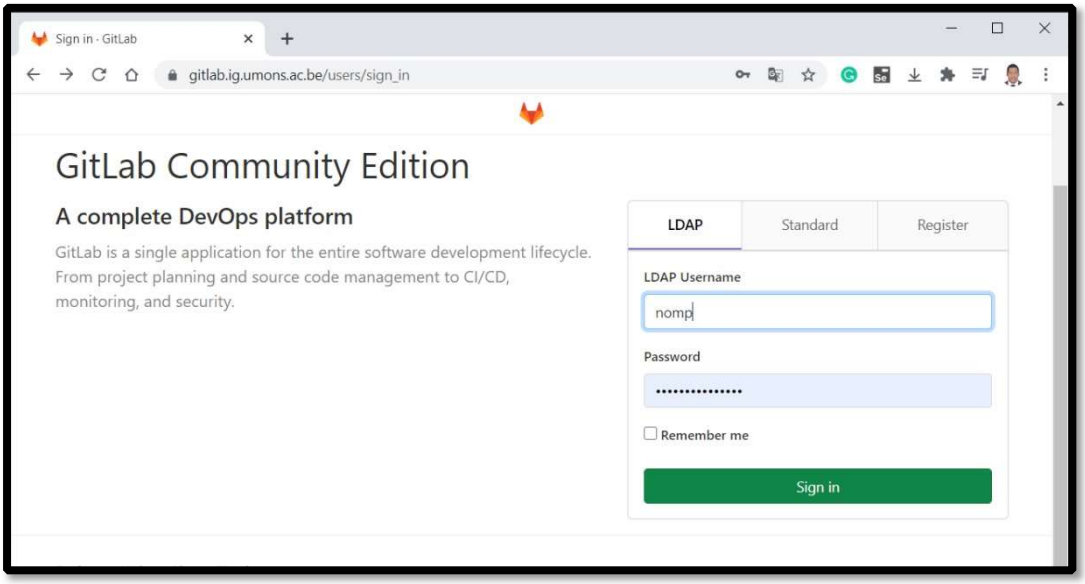

Figure 19 : Connexion à GitLab IG

Question 4 : Sur la page d'accueil de **GitLab IG**. Cliquez sur le bouton « **Create blank project** » pour créer un nouveau projet.

- Ensuite, sur la page « **New Project** », entrez le nom du projet, définissez le niveau de visibilité (**Public**, par exemple). Enfin, selectionnez « **Create Project** ».

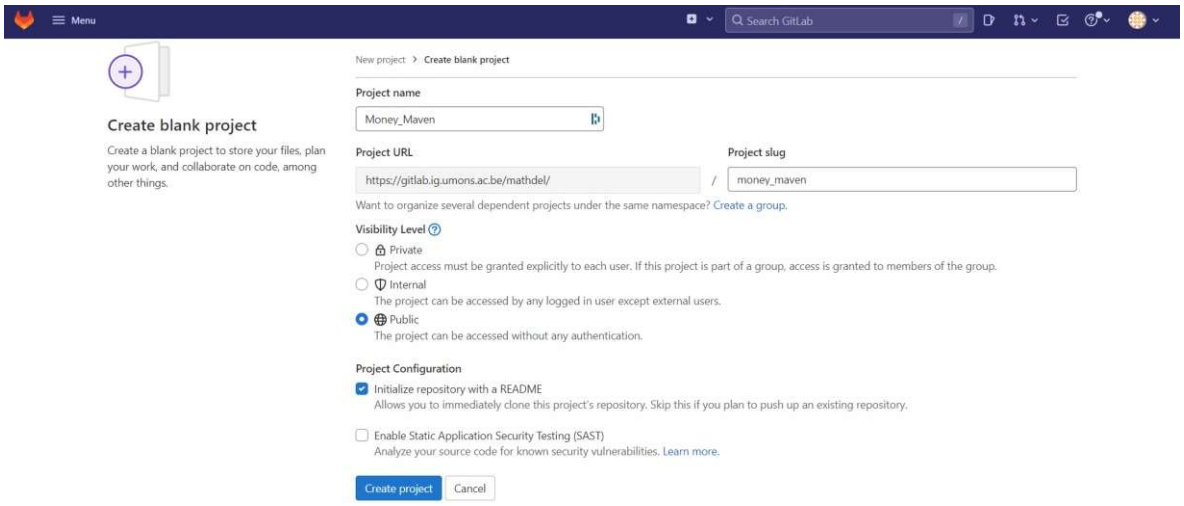

***Figure 20** : Création d'un projet sur GitLab IG (1)*

Question 5 :

- Créez un nouveau projet. Une fois le projet créé, Suivez les instructions GitLab pour importer le code du projet sur le dépôt GitLab du projet.

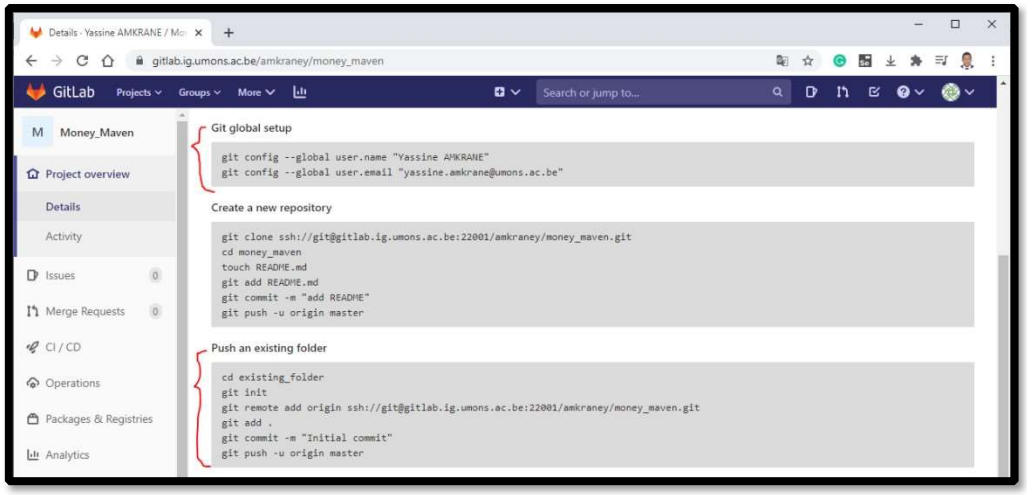

***Figure 21** : Création d'un projet sur GitLab IG (2)*

- Actualisez la page de **GitLab IG** pour voir si le projet est bien importé.

Question 6 : Vérifier que le fichier **pom.xml** spécifie bien la classe qui contient le **main()** pour que le run puisse se réaliser : (sachant qu'ici la classe qui contient le main est **Money** et le package est **be.ac.umons**) .

```
<artifactId>maven-jar-plugin</artifactId>
<version>3.0.2</version>
<configuration>
 <archive>
  <manifest>
    <mainClass>  be.ac.umons.Money</mainClass>
  </manifest>
 </archive>
</configuration>
```

Question 7 :

Crée le fichier « .gitlab-ci.yml » en cliquant sur « **Set up CI/CD** ». (Voir la figure 24)

Editor - > create new CI/CD pipeline

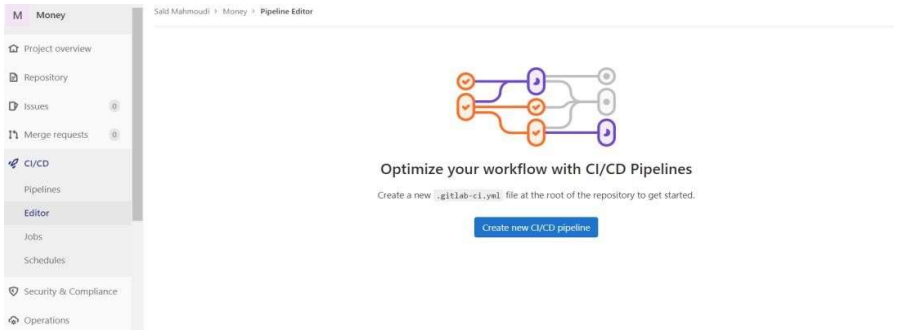

***Figure 22** : Exemple du fichier de configuration*

Question 8 : Définir le fichier "**.gitlab-ci.yml**" en s'inspirant de la figure 25.

```yaml
image: maven:3.6.0-jdk-8-alpine

stages:          # List of stages for jobs, and their order of execution
  - build
  - test
  - deploy

build-job:       # This job runs in the build stage, which runs first.
  stage: build
  script:
    - mvn compile
    - mvn package
  artifacts:
      name: "$CI_JOB_NAME-$CI_COMMIT_REF_NAME"
      paths:
      - target
      when: on_success

unit-test-job:   # This job runs in the test stage.
  stage: test    # It only starts when the job in the build stage completes successfully.
  script:
    - mvn test
  dependencies:
    - build-job


deploy-job:      # This job runs in the deploy stage.
  stage: deploy  # It only runs when *both* jobs in the test stage complete successfully.
  environment: production
  script:
    - java -jar ./target/Money-1.0-SNAPSHOT.jar
  dependencies:
    - build-job

```

***Figure 23** : Code du fichier .gitlab-ci;yml*

Question 9 : Cliquez sur Commit changes (un build se lance automatiquement)

***Figure 24** : Commit du fichier .gitlab.cí-yml*

Question 10 : Consultez les logs des builds. (Voir les figures 27-30)

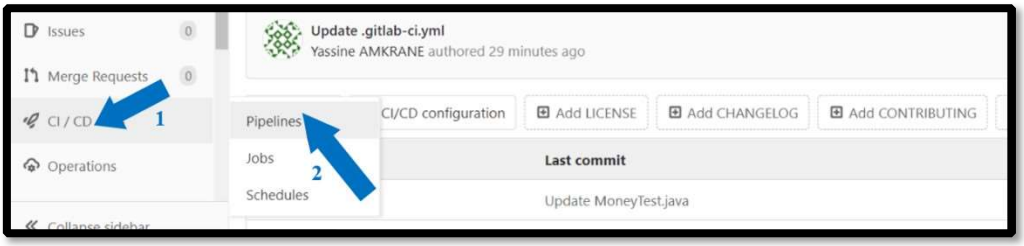

***Figure 25** : Consultation des builds (1)*

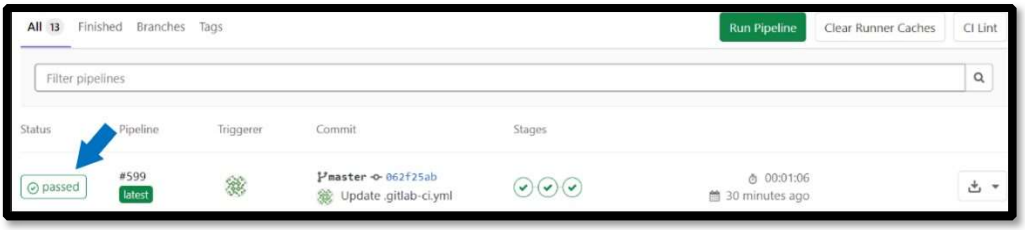

Figure 26 : Consultation des builds (2)

Figure 27 : Consultation des builds (3)

```
508 [INFO]  T E S T S
509 [INFO] -------------------------------------------------------
510 [INFO] Running be.ac.umons.MoneyTest
511 Add Test
512 Add Money Test
513 Amount Test
514 Currency Test
515 [INFO] Tests run: 4, Failures: 0, Errors: 0, Skipped: 0, Time elapsed: 0.02 s - in be.ac.umons.MoneyTest
516 [INFO]
517 [INFO] Results:
518 [INFO]
519 [INFO] Tests run: 4, Failures: 0, Errors: 0, Skipped: 0
520 [INFO]
521 [INFO] ------------------------------------------------------------------------
522 [INFO] BUILD SUCCESS
523 [INFO] ------------------------------------------------------------------------
524 [INFO] Total time:  7.694 s
525 [INFO] Finished at: 2020-11-07T13:37:40Z
526 [INFO] ------------------------------------------------------------------------
528 Cleaning up file based variables                                        00:01
530 Job succeeded
```

Figure 28 : Résultat des tests unitaires

Vous pouvez ajouter ou modifier les tests unitaires et lancer des builds.

FIN de l'énoncé