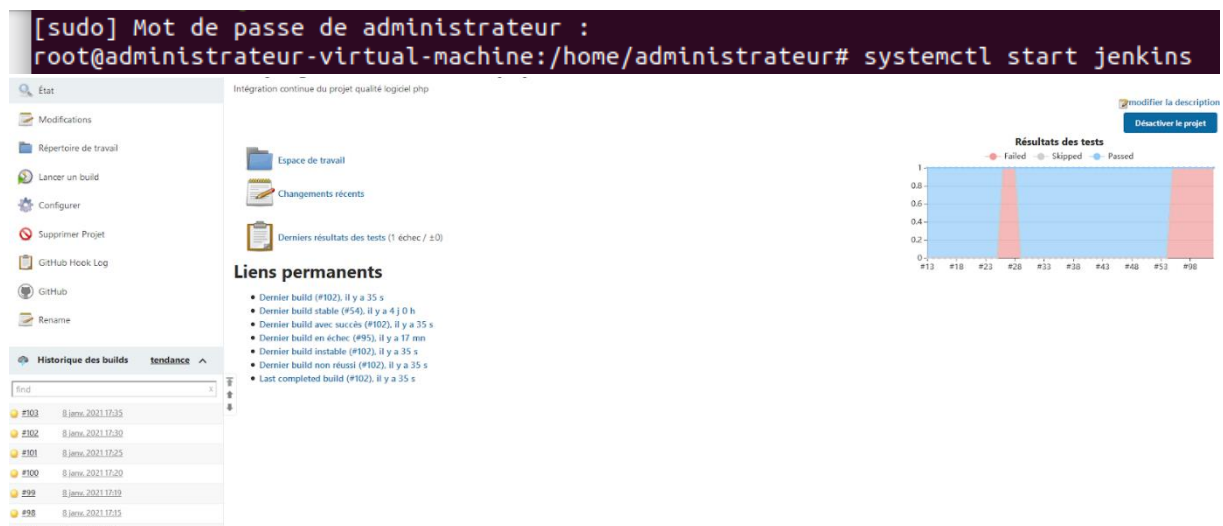


# Rapport de mise en place de l'intégration continue

Après avoir installé toutes les dépendances de Jenkins, on démarre le serveur Jenkins qu'on a installé dans une machine virtuelle dans le but de simuler l'intégration continue. Le seul problème est que pour lancer des tests unitaires qui interagissent avec la BDD on a une erreur car Jenkins préfère gérer des BDD sous forme de fichiers, ou il pourrait éventuellement exister un plugin pour associer MySQL à Jenkins mais nous n'avons pas réussi à le mettre en place.

Cependant quand le test unitaire est simple, on peut lancer un build, ce qui simule un lancement de test unitaire quand on push dans une branche.



Quand on va dans l'onglet « configurer » de Jenkins on peut associer le job à un git (celui-ci sera importé dans le serveur), on spécifie la branche sur laquelle les tests vont se lancer.

The screenshot shows the 'Gestion de code source' configuration page in Jenkins. It has two main sections: 'Repositories' and 'Branches to build'. In the 'Repositories' section, 'Git' is selected as the provider. The 'Repository URL' is set to `https://github.com/mhabaj/GestionEmpruntsMaterielInfo.git`. The 'Credentials' dropdown is set to '- aucun -' with an 'Ajouter' button. There are 'Avancé...' and 'Add Repository' buttons. In the 'Branches to build' section, the 'Branch Specifier (blank for \'any\')' is set to `*/adrien`. There is a red 'X' icon and an 'Add Branch' button.

Ci-dessous, on spécifie la fréquence ou les tests seront lancés (ici en l'occurrence on a configuré un test unitaire toutes les 5 minutes).

☒ Construire périodiquement

Planning

⚠ Étaler la charge de façon régulière en utilisant 'H/5 \* \* \* \*' plutôt que '\* /5 \* \* \* \*'  
Aurait été lancé à vendredi 8 janvier 2021 à 22:40:18 heure normale d'Europe centrale; prochaine exécution à vendredi 8 janvier 2021 à 22:45:18 heure normale d'Europe centrale.

☒ GitHub hook trigger for GITScm polling

Ci-dessous, on spécifie l'action que va effectuer le job, ici on lance les tests unitaires qui font les actions et la logique est spécifiée dans le fichier phpunit.xml ou on met en place une suite de tests,

```
<?xml version="1.0" encoding="UTF-8"?>
<phpunit colors="true"
  convertErrorsToExceptions="true"
  convertNoticesToExceptions="true"
  convertWarningsToExceptions="true"
  processIsolation="false"
  stopOnFailure="false">
  <testsuites>
    <testsuite name="Test suite">
      <directory>./</directory>
    </testsuite>
  </testsuites>
</phpunit>
```

ci-dessous on dit que Jenkins doit lancer les tests, dans action à la suite du build on spécifie le fichier qui doit être lu, celui qui contient les résultats des tests. Le principe est de convertir PHPUnit en JUnit, ce que comprends Jenkins.

**Exécuter un script shell**

Commande

[Voir la liste des variables d'environnement disponibles](#)

**Actions à la suite du build**

**Publier le rapport des résultats des tests JUnit**

XML des rapports de test

Une configuration du type **Fileset 'includes'** qui indique où se trouvent les fichiers XML des rapports de test, par exemple 'myproject/target/test-reports /'.xml'. Le répertoire de base (basedir) du fileset est **la racine du workspace**.

☐ Retain long standard output/error

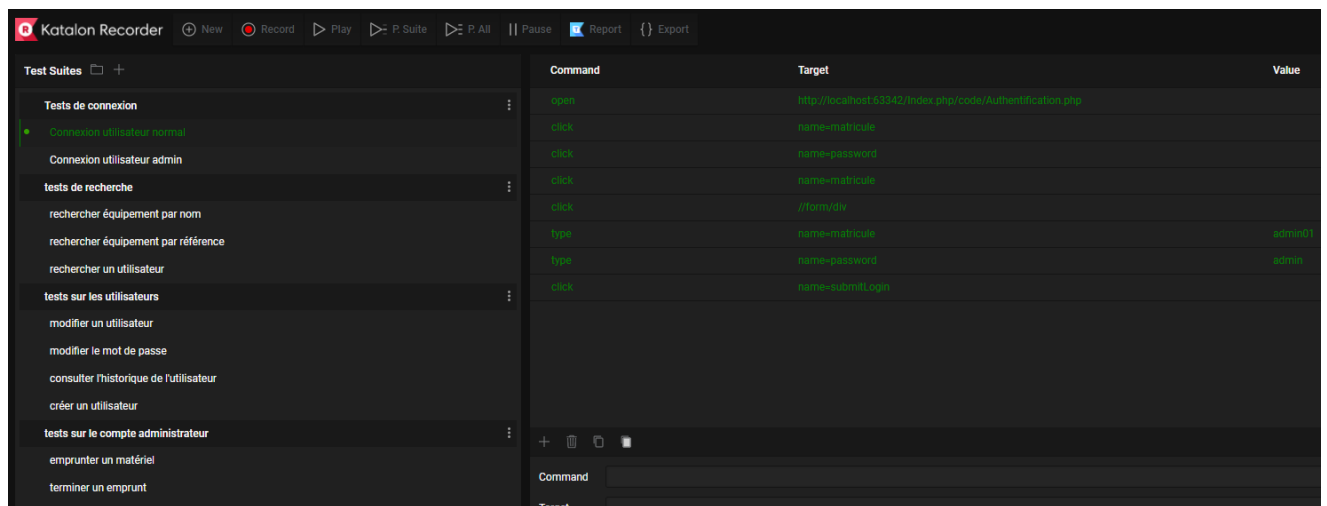
Health report amplification factor

Sources : <https://www.youtube.com/watch?v=Gy4Nk2pluNs>

<https://www.digitalocean.com/community/tutorials/how-to-install-jenkins-on-ubuntu-18-04>

# Partie sur les tests fonctionnels avec Katalon IDE de Selenium

Ci-dessous la liste des tests fonctionnels du projet, chaque cas contient tous les cas possibles, veuillez-nous excuser s'il en manque, nous étions sous pression à cause du temps, mais en tout cas nous avons compris comment utiliser l'outil et comment exporter les tests sous JUnit.



The screenshot displays the Katalon Recorder interface. On the left, a sidebar lists 'Test Suites' including 'Tests de connexion', 'Connexion utilisateur normal', 'Connexion utilisateur admin', 'tests de recherche', 'tests sur les utilisateurs', and 'tests sur le compte administrateur'. The main area shows a table of commands and targets for a test suite.

Command	Target	Value
open	http://localhost:83342/index.php/code/authentication.php	
click	name=matricule	
click	name=password	
click	name=matricule	
click	/formuliv	
type	name=matricule	admin1
type	name=password	admin
click	name=submit_login	