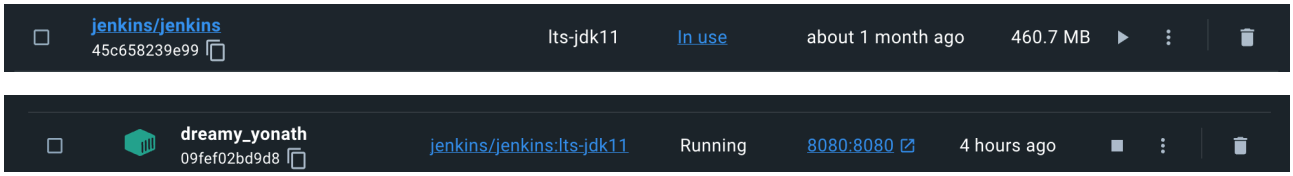
Endy Mehraein

B3-C1-TD Jenkins

1er Etape installation et Configuration

Tout d'abord j'ai fait le choix de faire ce TD via Docker, cela ma permis de monter mes compétences en docker de plus je trouve cela plus pratique.

Je suis donc aller récupéré l'image officielle de Jenkins sur le forum DockerHub https://hub.docker.com/_/jenkins



Nous avons donc notre image et notre conteneur que nous pouvons lancé.

Suite à cela nous pouvons nous diriger sur le port http://localhost:8080/

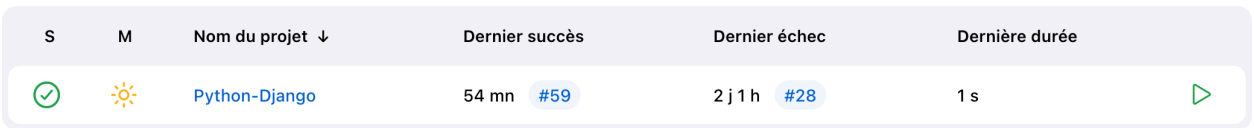
Un mot de passe d'administrateur nous sera demandé, le mot de passe est dans le fichier initialAdminPassword il suffit de le copié et le renseigné

Une fois chose faite nous pouvons créer un compte administrateur via le formulaire Jenkins

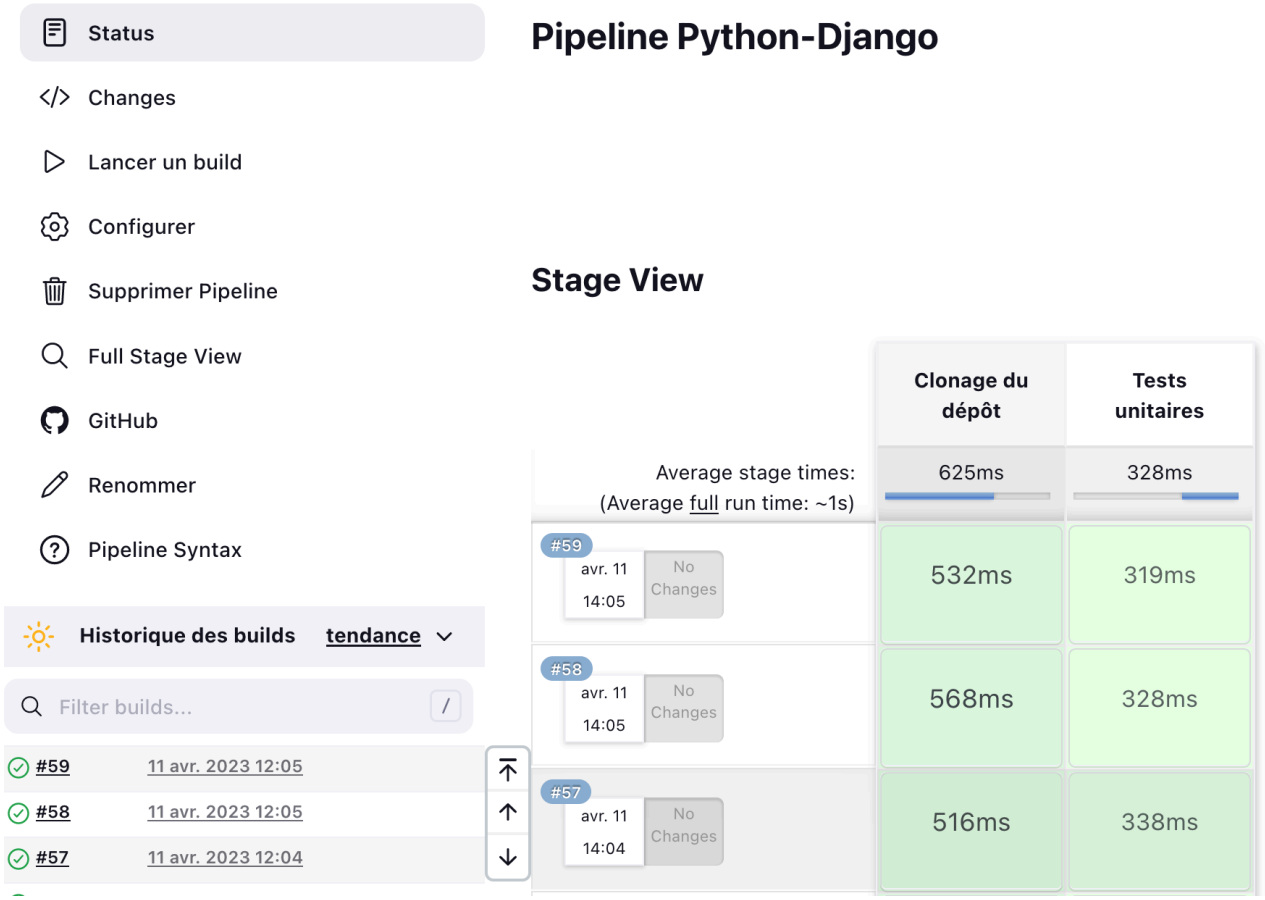
Passons en suite à la création de l'item ou du projet, on renseigne un nom

Et nous pouvons sélectionné « Pipeline » qui sera suffisant pour l'exécution de tests.

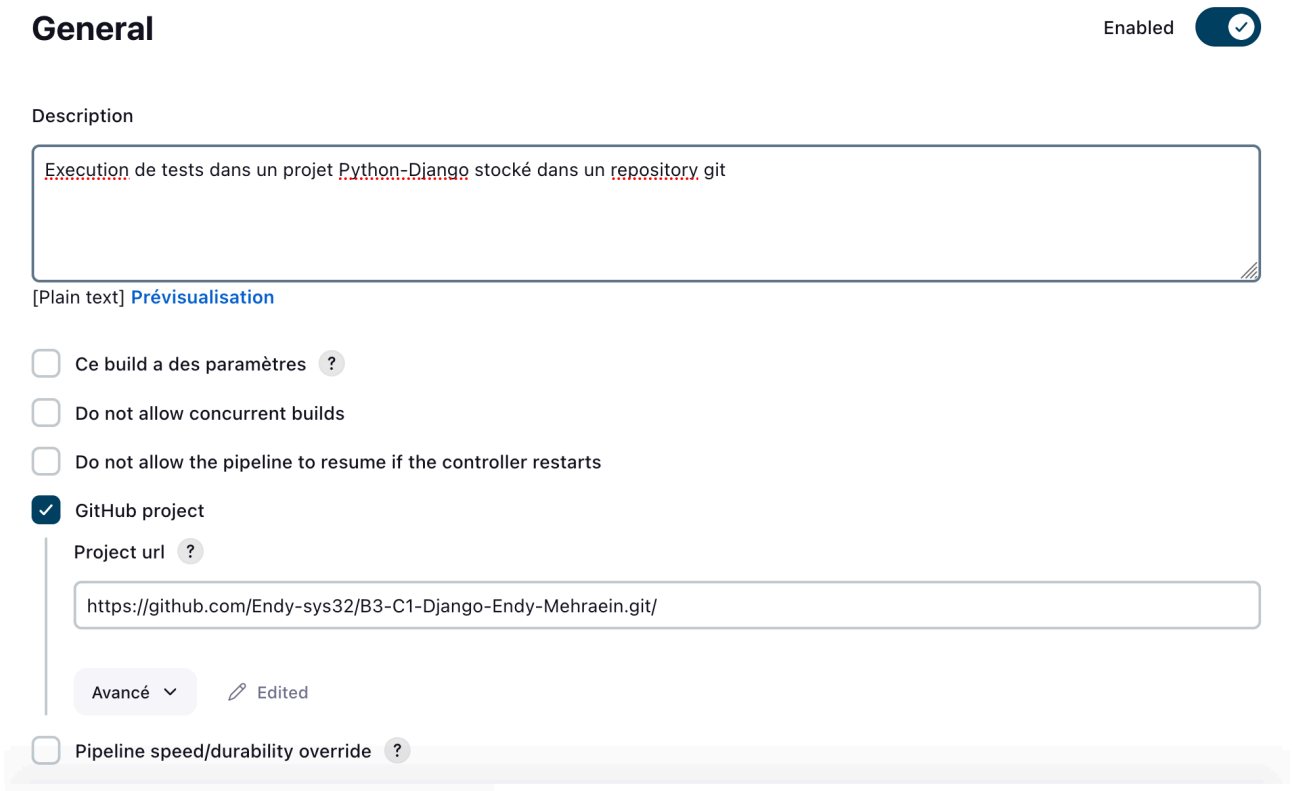
Nous retrouverons alors notre projet avec des métriques sur les précédent builds.



Passons maintenant à la configuration de notre pipeline



Tout d'abord nous devons récupéré le projet dans laquelle nous voulons faire nos tests,
Pour cela il faut spécifier à Jenkins l'adresse du repo cible.



Dans cette exemple le repo cible est https://github.com/Endy-sys32/B3-C1-Django-Endy-Mehraein qui est un projet Python-Django

Nous allons ensuite configuré la pipeline qui

Definition

Pipeline script

Script

```
pipeline {
    agent any
    stages {
        stage('Clonage du dépôt') {
            steps {
                git branch: 'test', url: 'https://github.com/Endy-sys32/B3-C1-Django-Endy-Mehraein.git'
            }
        }
        stage('Tests unitaires') {
            steps {
                dir('pilotage/src') {
                    sh 'python3 -m unittest tests.py'
                }
            }
        }
    }
}
```

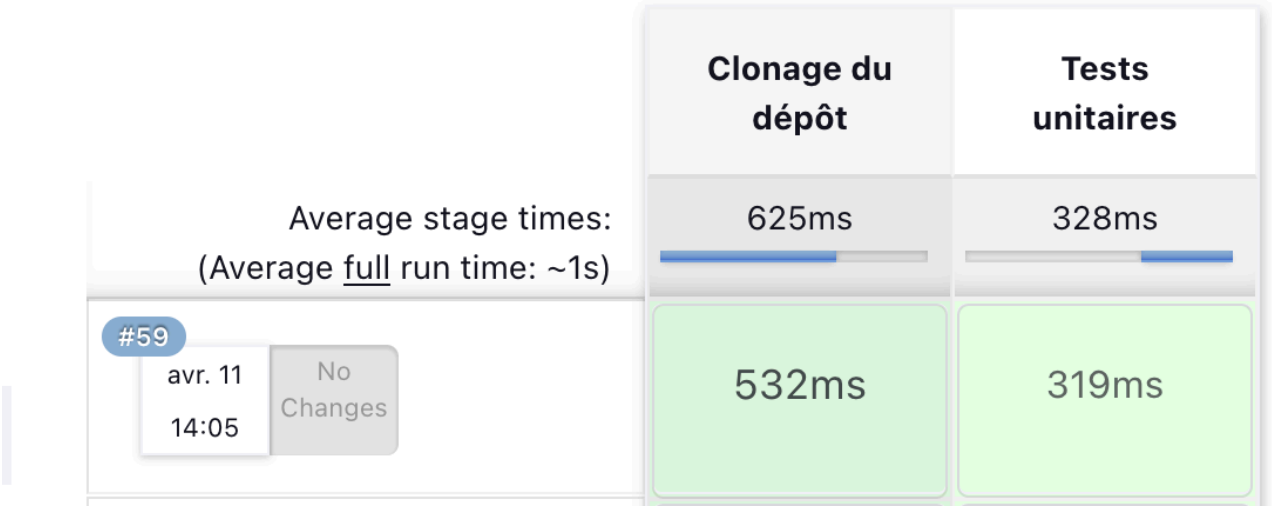
Use Groovy Sandbox

Pipeline Syntax

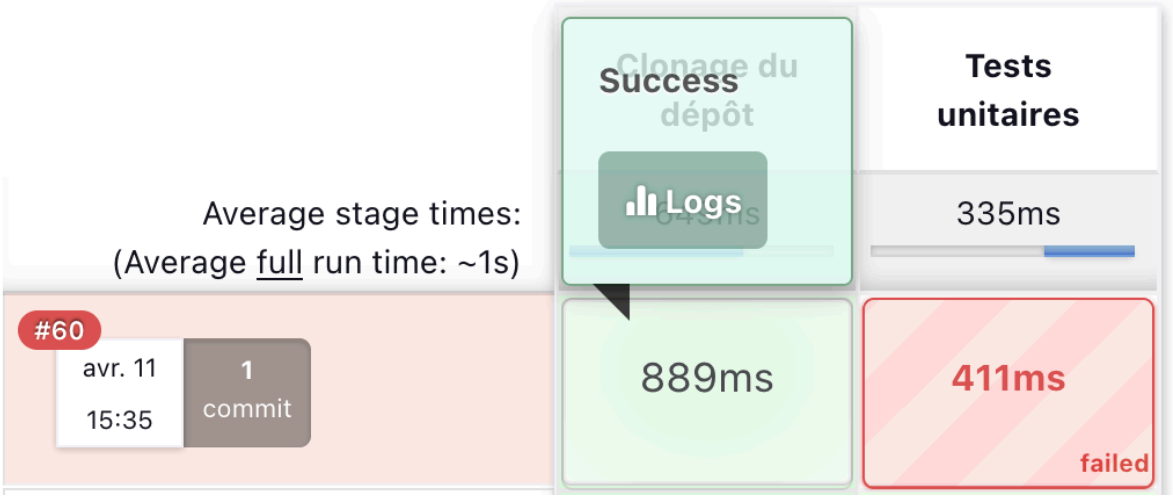
Nous pouvons voir deux stage différend, Clonage du dépôt et Tests unitaire

Dans la phase clonage du dépôt nous récupérons le code du repo et nous nous positionnons sur la branche test

Dans la phase Tests unitaires nous nous déplaçons dans l'arborescence pour accéder au dossier contenant les tests puis nous les éxécutons.



Lors de l'exécutions des tests nous pouvons donc voir les deux phases avec des logs sur le déroulement de la pipeline.



En cas de fail nous pouvons voir la phases à l'issue du fail
Et nous avons accès à des logs sur les causes de l'erreur

Stage Logs (Tests unitaires)

Shell Script -- python3 -m unittest tests.py (self time 282ms)

```
+ python3 -m unittest tests.py
........F........
======================================================================
FAIL: test_division5 (tests.Test)
----------------------------------------------------------------------
Traceback (most recent call last):
  File "/var/jenkins_home/workspace/Python-Django/pilotage/src/tests.py", line 42, in test_division5
    self.assertEqual(division(4, 4), 6)
AssertionError: 1.0 != 6

----------------------------------------------------------------------
Ran 17 tests in 0.000s

FAILED (failures=1)
```

Ici nous pouvons voir que la fonction test_division5 qui à pour paramètre 4 et 4 attend la réponse 6 alors que la réponse est 1 il nous précise alors que 1 est différend de 6 donc erreur.

Pour éxécuter le build à distance nous pouvons utilisé un trigger qui ce déclenchera lors d'un push sur le repo choisi

Et pour finir en cas d'échec ou du succès du build nous pouvons configuré la pipeline afin d'envoyé un mail.

```
    }
    success {
        steps {
            mail to: 'ensy.mehraein@epsi.fr',
                subject: 'Construcuction réussie'
                body: "Le build s'est terminé avec succés"
        }
    }
    failure {
        steps {
            mail to: 'ensy.mehraein@epsi.fr',
                subject: 'Échec de la construction Jenkins',
                body: "Il y a eu une erreur lors de la construction du projet Jenkins. Veuillez vérifier les journaux de build pour plus d'informations."
        }
    }
}
```