Jenkins

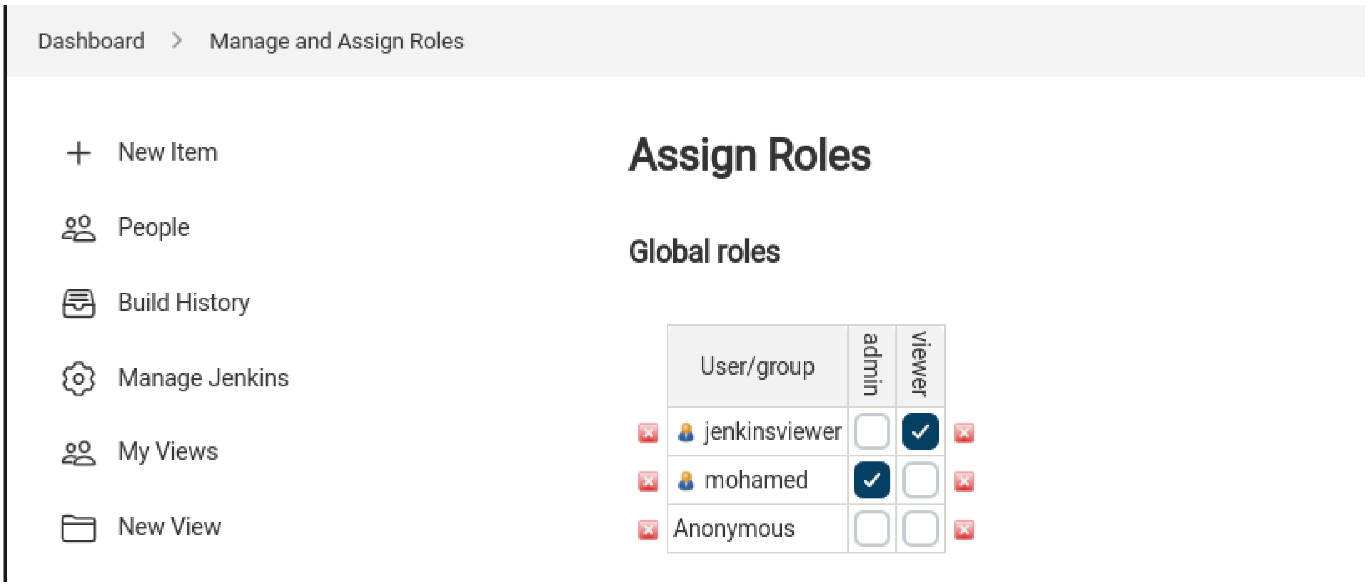
1. install jenkins with docker image

```
└─ mohamed@DevOps:$ docker pull jenkins/jenkins:lts
└─ mohamed@DevOps:$ docker run --name jenkins -d -p8080:8080
jenkins/jenkins:lts
0c8869f7380bb9687a0c628cc2b07bb2d92ac43877c25ac51d16e485267d5ec8
```

2. install role based authorization plugin

3. create new user

4. create read role and assign it to the new user

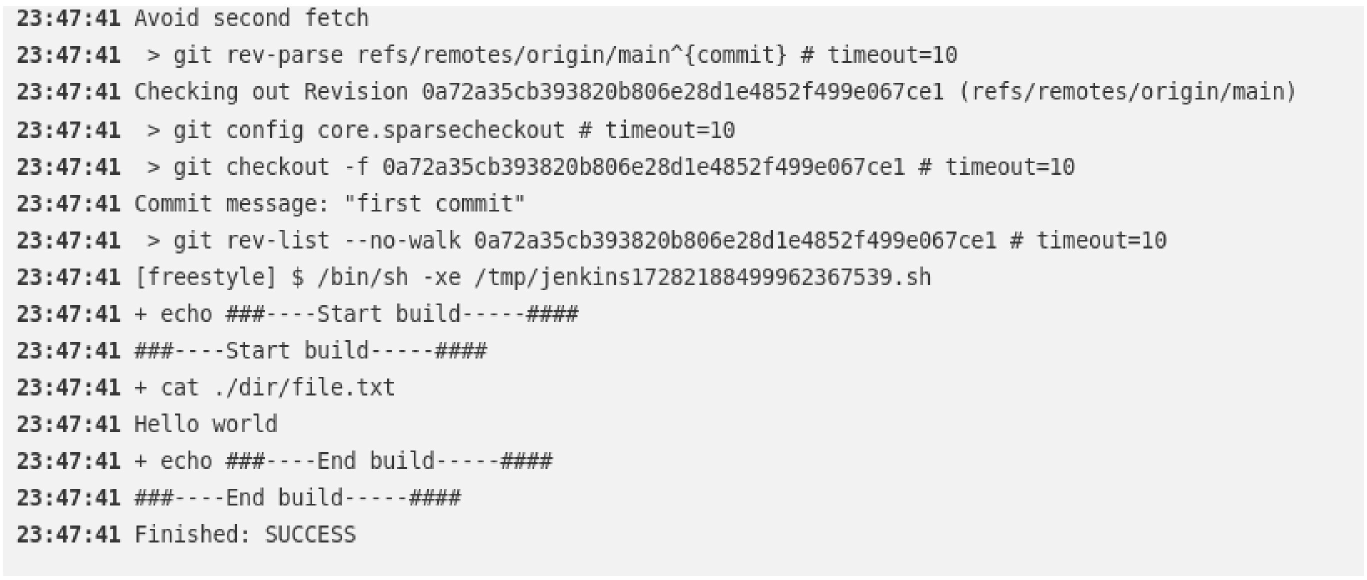


Assign Roles

Global roles

User/group	admin	viewer
jenkinsviewer	☐	✓
mohamed	✓	☐
Anonymous	☐	☐

5. create free style pipeline and link it to private git repo(inside it create directory and create file with "hello world")



```
23:47:41 Avoid second fetch
23:47:41  > git rev-parse refs/remotes/origin/main^{commit} # timeout=10
23:47:41 Checking out Revision 0a72a35cb393820b806e28d1e4852f499e067ce1 (refs/remotes/origin/main)
23:47:41  > git config core.sparsecheckout # timeout=10
23:47:41  > git checkout -f 0a72a35cb393820b806e28d1e4852f499e067ce1 # timeout=10
23:47:41 Commit message: "first commit"
23:47:41  > git rev-list --no-walk 0a72a35cb393820b806e28d1e4852f499e067ce1 # timeout=10
23:47:41 [freestyle] $ /bin/sh -xe /tmp/jenkins17282188499962367539.sh
23:47:41 + echo ###----Start build-----####
23:47:41 ###----Start build-----####
23:47:41 + cat ./dir/file.txt
23:47:41 Hello world
23:47:41 + echo ###----End build-----####
23:47:41 ###----End build-----####
23:47:41 Finished: SUCCESS
```

---

1. create declarative in jenkins GUI pipeline for your own repo to do "ls"

```
pipeline {
    agent any

    stages {
        stage('Preparation') {
            steps {
                checkout changelog: false, poll: false, scm: [$class:
'GitSCM', branches: [[name: '*/main']], extensions: [], userRemoteConfigs:
[[credentialsId: 'github', url: 'https://github.com/mohamedanwer006/iti-
lab.git']]]

            }
        }

        stage('Build') {
            steps {
                 sh 'ls -R'
            }

        }
    }
}
```

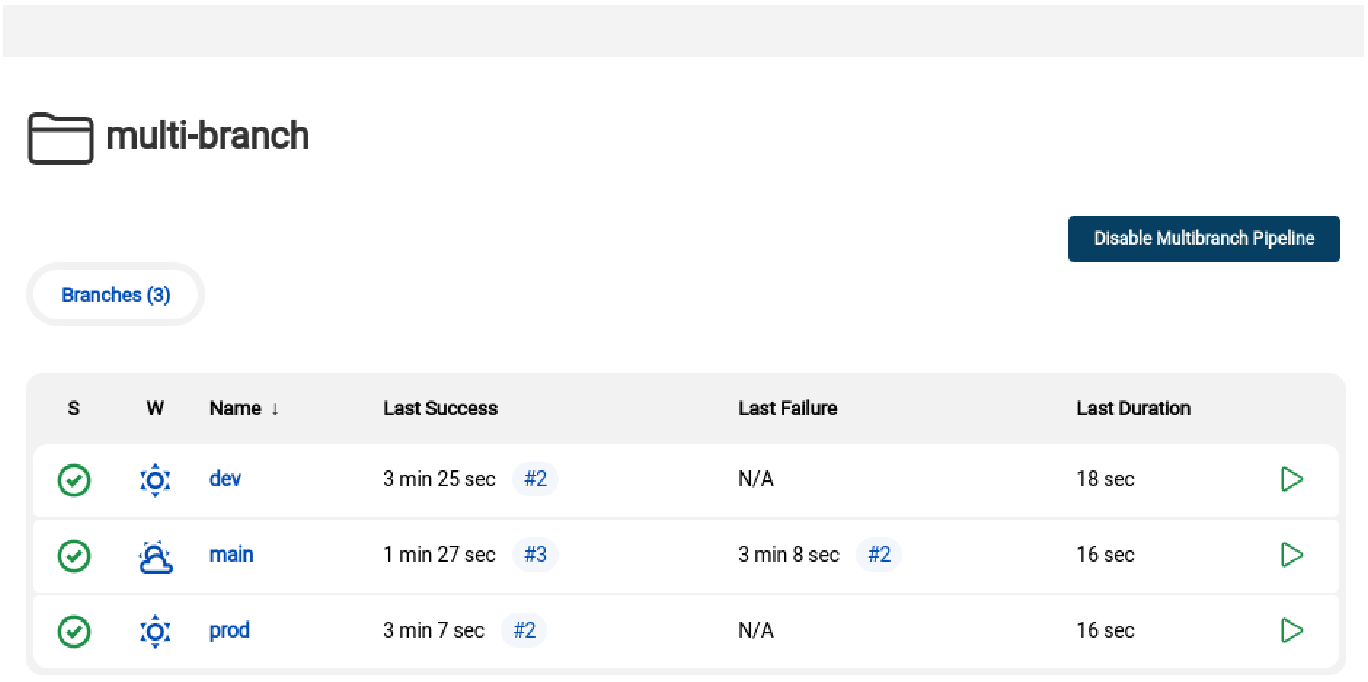
2. create scripted in jenkins GUI pipeline for your own repo to do "ls"

```


node {
    stage('Preparation') { // for display purposes
        // Get some code from a GitHub repository
        checkout changelog: false, poll: false, scm: [$class: 'GitSCM',
branches: [[name: '*/main']], extensions: [], userRemoteConfigs:
```

```
[[credentialsId: 'github', url: 'https://github.com/mohamedanwer006/iti-
lab.git']]]
        }
        stage('Build') {
            sh 'ls -R'
        }
    }
```

```
The recommended git tool is: NONE
using credential github
 > git rev-parse --resolve-git-dir /var/jenkins_home/workspace/declarative/.git # timeout=10
Fetching changes from the remote Git repository
 > git config remote.origin.url https://github.com/mohamedanwer006/iti-lab.git # timeout=10
Fetching upstream changes from https://github.com/mohamedanwer006/iti-lab.git
 > git --version # timeout=10
 > git --version # 'git version 2.30.2'
using GIT_ASKPASS to set credentials
 > git fetch --tags --force --progress -- https://github.com/mohamedanwer006/iti-lab.git
+refs/heads/*:refs/remotes/origin/* # timeout=10
 > git rev-parse refs/remotes/origin/main^{commit} # timeout=10
Checking out Revision 40c0b8b2e21d6c76e206d1302c51302d5f78daaf (refs/remotes/origin/main)
 > git config core.sparsecheckout # timeout=10
 > git checkout -f 40c0b8b2e21d6c76e206d1302c51302d5f78daaf # timeout=10
Commit message: "Create README.md"
[Pipeline] }
[Pipeline] // stage
[Pipeline] stage
[Pipeline] { (build)
[Pipeline] sh
+ ls -R
.:
README.md
dir

./dir:
file.txt
```

3. create the same with jenkinsfile in your branches as multibranch pipeline



S	W	Name ↓	Last Success	Last Failure	Last Duration
✓		dev	3 min 25 sec #2	N/A	18 sec
✓		main	1 min 27 sec #3	3 min 8 sec #2	16 sec
✓		prod	3 min 7 sec #2	N/A	16 sec

4. try to create jenkins image with casc and install slack notification plugin and main suggested plugins inside it
5. casc will contain creation of user and creation of credential for your dockerhub

```yaml
jenkins:
  securityRealm:
    local:
      allowsSignup: false
      users:
        - id: "admin"
          password: "admin"
        - id: "dev"
          password: "dev"

credentials:
  system:
    domainCredentials:
      - credentials:
          - usernamePassword:
              scope: GLOBAL
              id: "dockerhub"
              username: "mohameddev006"
              password: "1234456789"
              description: "Username/Password Credentials for DockerHub"
```