Lesson 08 Demo 03

Configuring Multibranch Jenkins Pipeline Job

Objective: To configure a Multibranch Jenkins pipeline job to facilitate managing multiple projects within a single job setup

Tools required: Jenkins, Git, and GitHub

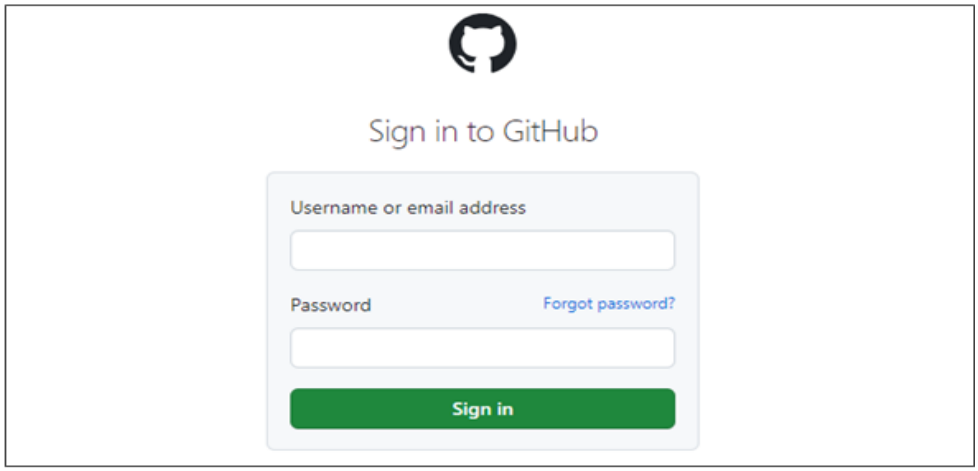
Prerequisites: None

Steps to be followed:

1. Create a Git repository
2. Push the code file into the Git repository
3. Create a Multibranch Pipeline

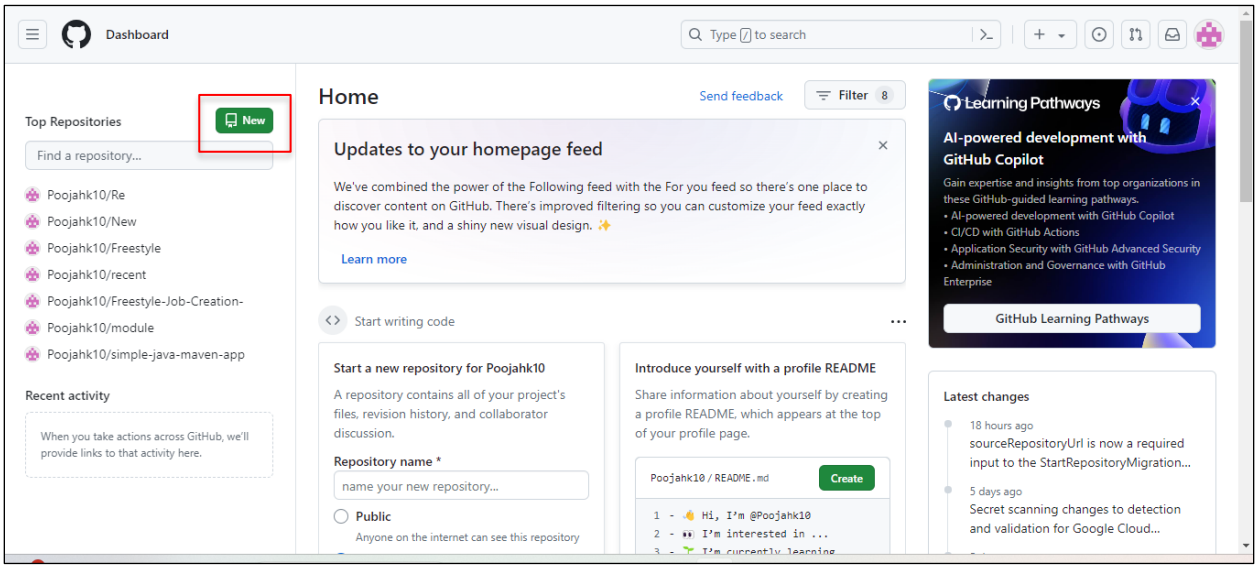
Step 1: Create a Git repository

1.1 Open the browser in your lab, go to **github.com**, and sign in to your account

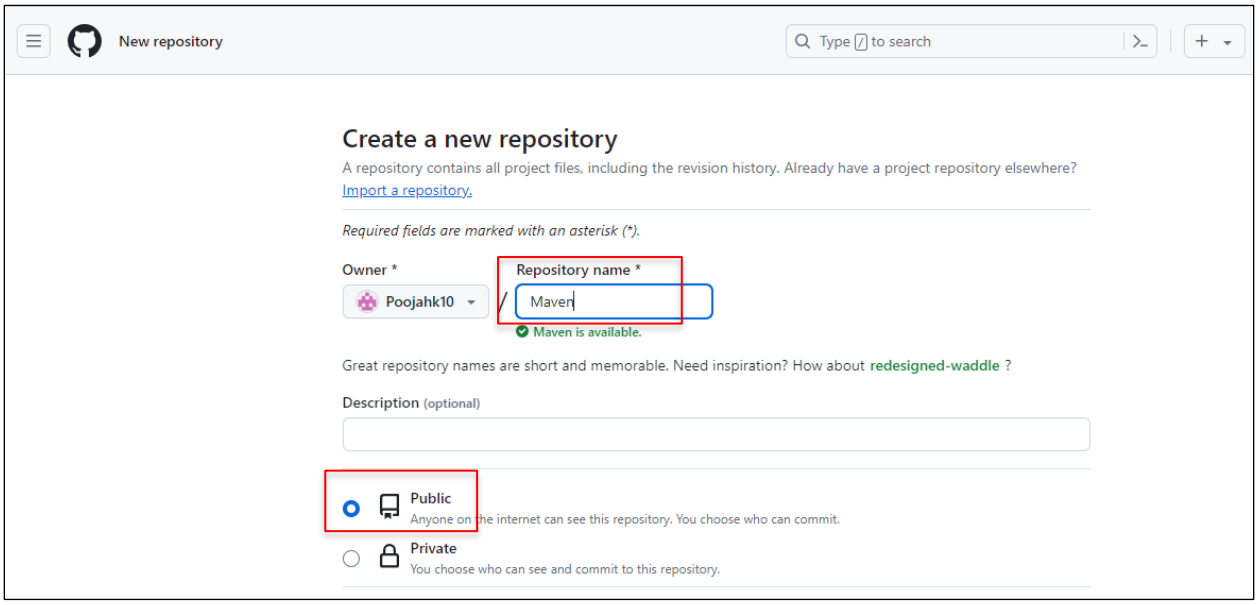


Note: If you do not have a GitHub account, visit the official website at https://github.com/signup and create a new account

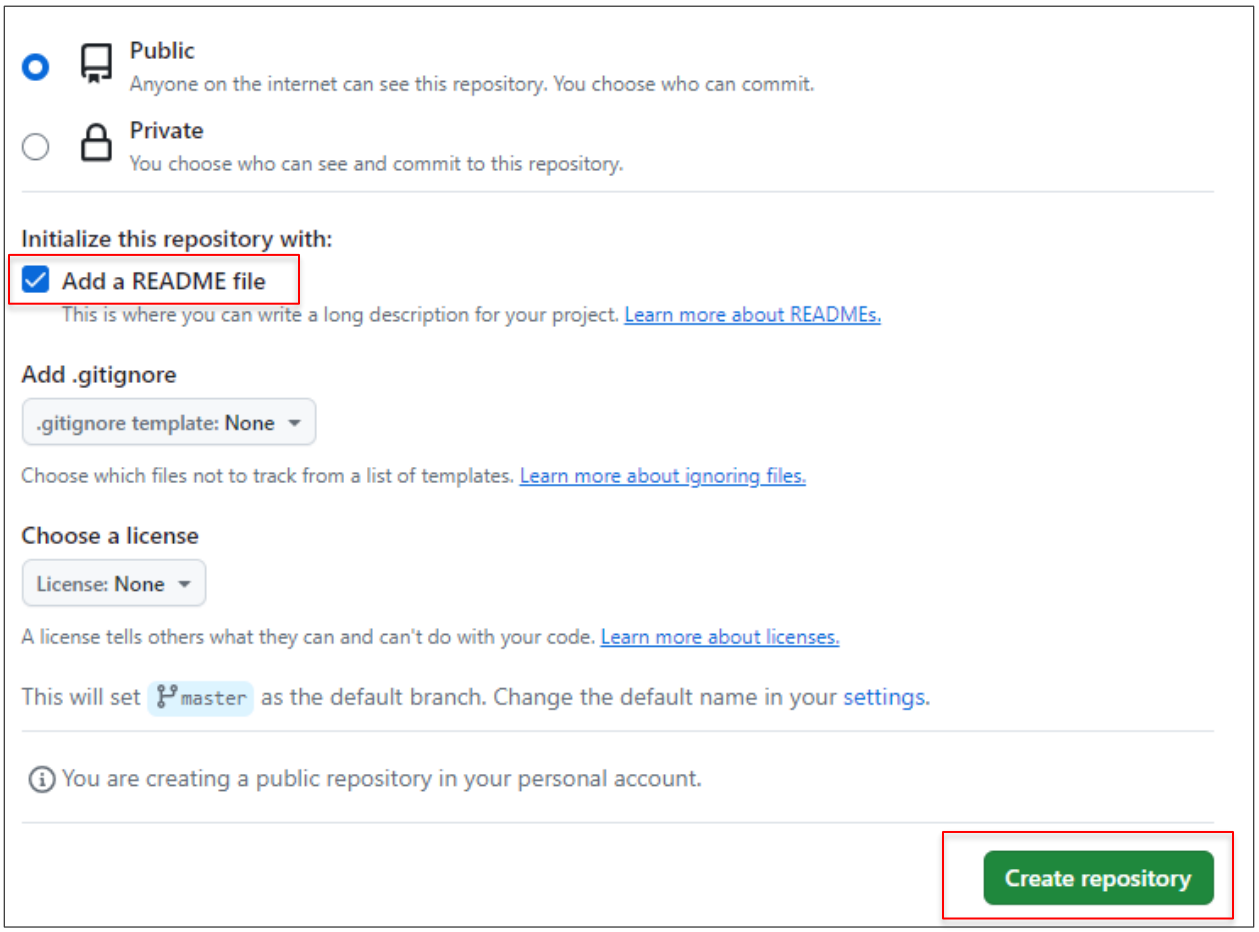
1.2 Click on **New** as shown in the screenshot below:



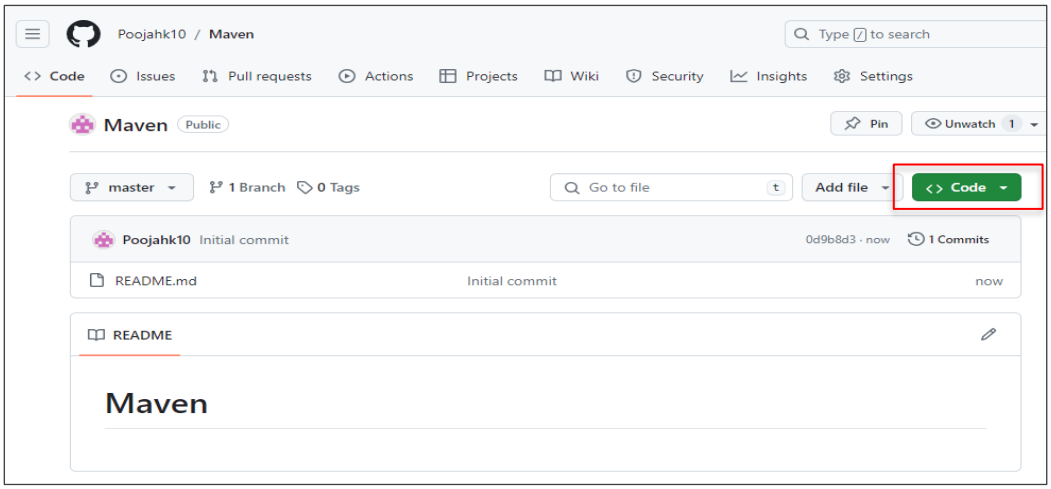
1.3 Enter a desired name for your repository and choose **Public** as shown in the screenshot below:



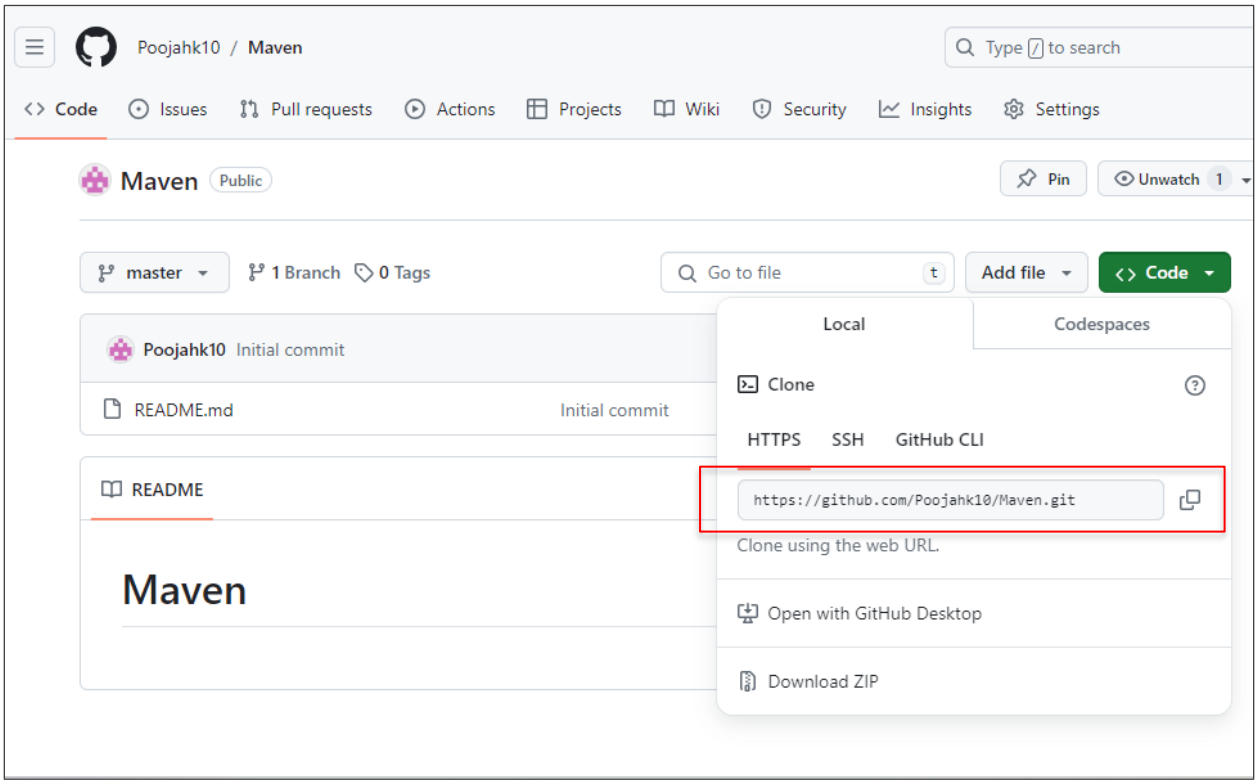
1.4 Click on **Add a README file** and then click on **Create repository** as shown in the screenshot below:



1.5 Click on **Code** as shown in the screenshot below:



1.6 Copy the repository URL as shown in the screenshot below:



Step 2: Push the code file into the Git repository

2.1 Open the Linux terminal in your lab and clone the repository using the below command:
git clone RepositoryURL

```
syedsharozsimpl@ip-172-31-40-171:~$ git clone https://github.com/[redacted]ven.git
Cloning into 'Maven'...
remote: Enumerating objects: 3, done.
remote: Counting objects: 100% (3/3), done.
remote: Total 3 (delta 0), reused 0 (delta 0), pack-reused 0
Receiving objects: 100% (3/3), done.
syedsharozsimpl@ip-172-31-40-171:~$
```

2.2 Navigate inside the repository that you had created using the below command:
cd RepositoryName

```
syedsharozsimpl@ip-172-31-40-171:~$ cd Maven
syedsharozsimpl@ip-172-31-40-171:~/Maven$
```

2.3 Initialize the Git using the below command:

git init

```
syedsharozsimpl@ip-172-31-40-171:~/Maven$ git init
Reinitialized existing Git repository in /home/syedsharozsimpl/Maven/.git/
syedsharozsimpl@ip-172-31-40-171:~/Maven$
```

2.4 Create a file using the below command:

nano demofile

```
syedsharozsimpl@ip-172-31-40-171:~/Maven$ nano demofile
```

2.5 Paste the below pipeline script inside the file as shown in the screenshot below:

```
pipeline {
  agent any
  stages {
    stage('Checkout') {
      steps {
        // Checkout your source code from version control
        git 'https://github.com/your/repository.git'
      }
    }
    stage('Build') {
      steps {
        // Use Maven to build your project
        sh 'mvn clean package'
      }
    }
    stage('Test') {
      steps {
        // Run tests if applicable
        sh 'mvn test'
      }
    }
    stage('Deploy') {
      steps {
        // Deploy your artifact, if necessary
        // Example: sh 'mvn deploy'
      }
    }
  }
```

```
    post {
        success {
            // This block will be executed if the pipeline runs successfully
            echo 'Pipeline executed successfully!'
        }
        failure {
            // This block will be executed if the pipeline fails
            echo 'Pipeline failed!'
        }
    }
}
```

Note: Ensure you provide your Git repository URL on line7, save, and exit the page by clicking on **ctrl+S** to save and **ctrl+X** to exit

```
syedsharozsimpl@ip-172-31-40-171: ~/Maven
File  Edit  View  Search  Terminal  Help
  GNU nano 6.2                                                    demofile *
pipeline {
    agent any

    stages {
        stage('Checkout') {
            steps {
                // Checkout your source code from version control
                git 'https://github.com/your/repository.git'
            }
        }
        stage('Build') {
            steps {
                // Use Maven to build your project
                sh 'mvn clean package'
            }
        }
        stage('Test') {
            steps {
                // Run tests if applicable
                sh 'mvn test'
            }
        }
        stage('Deploy') {
            steps {

^G Help        ^O Write Out   ^W Where Is    ^K Cut         ^T Execute     ^C Location    M-U Undo       M-A Set
^X Exit        ^R Read File   ^\ Replace     ^U Paste       ^J Justify     ^/ Go To Line  M-E Redo       M-6 Cop
```

2.6 Stage and commit the changes using the below commands:

git add .

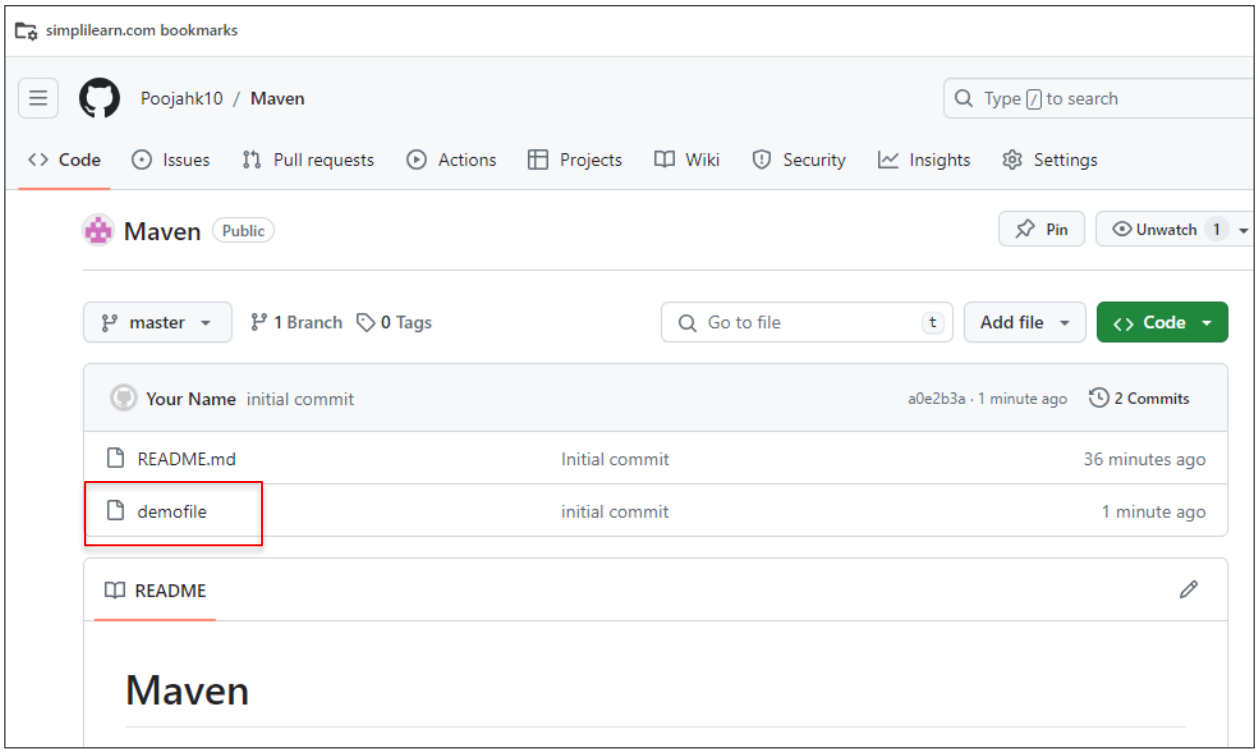
git commit -m “initial commit”

```
syedsharozsimpl@ip-172-31-40-171:~/Maven$ git add .
syedsharozsimpl@ip-172-31-40-171:~/Maven$ git commit -m "initial commit"
[master a0e2b3a] initial commit
 1 file changed, 41 insertions(+)
 create mode 100644 demofile
syedsharozsimpl@ip-172-31-40-171:~/Maven$
```

2.7 Push the file to the Git repository using the below command:
git push

```
syedsharozsimpl@ip-172-31-40-171:~/Maven$ git push
Username for 'https://github.com': Poojahk10
Password for 'https://Poojahk10@github.com':
Enumerating objects: 4, done.
Counting objects: 100% (4/4), done.
Delta compression using up to 4 threads
Compressing objects: 100% (3/3), done.
Writing objects: 100% (3/3), 614 bytes | 614.00 KiB/s, done.
Total 3 (delta 0), reused 0 (delta 0), pack-reused 0
To https://github.com/Poojahk10/Maven.git
   0d9b8d3..a0e2b3a  master -> master
syedsharozsimpl@ip-172-31-40-171:~/Maven$
```

2.8 Navigate to your Git repository to check for the file that is pushed:

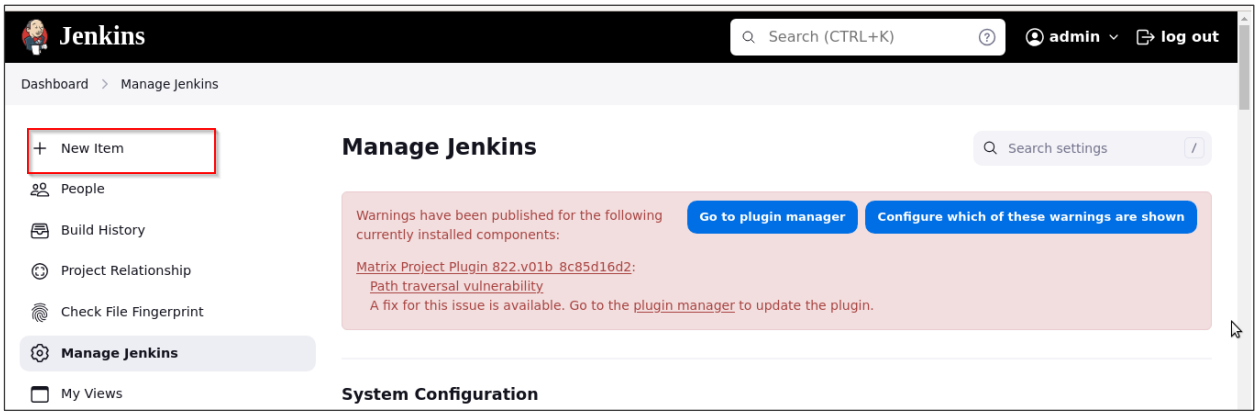


Step 3: Create a Multibranch Pipeline

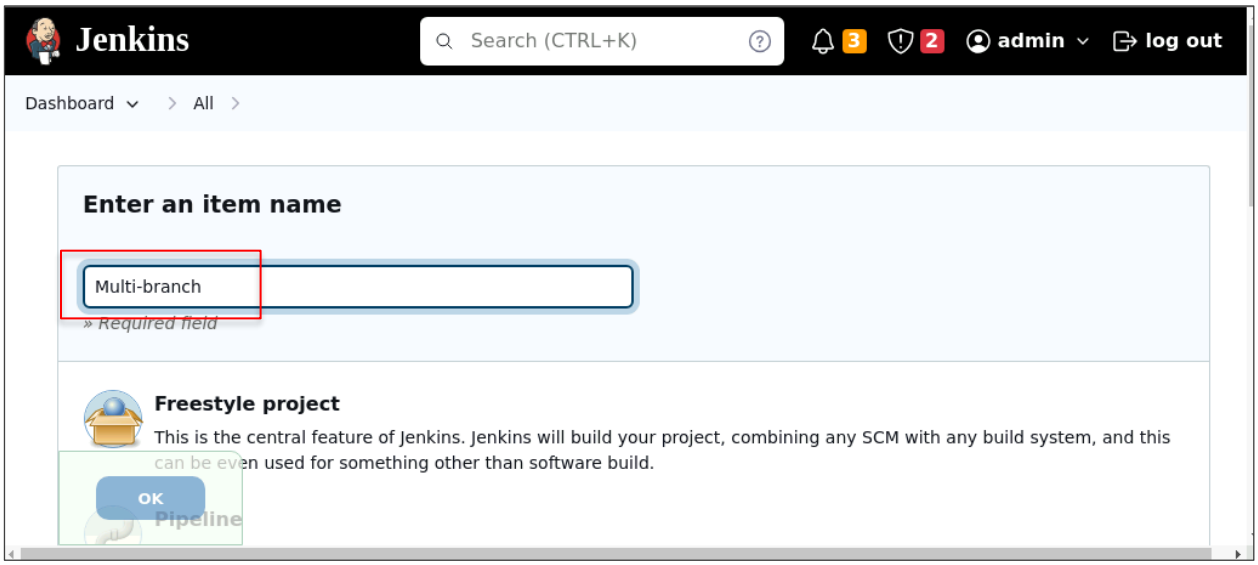
3.1 Open the browser, go to the Jenkins **Dashboard** by typing **localhost:8080** in your browser, provide the credentials, and click the **Sign in** button

Note: Use the given credentials to access Jenkins in the lab: **Username** is admin and **Password** is Root123$

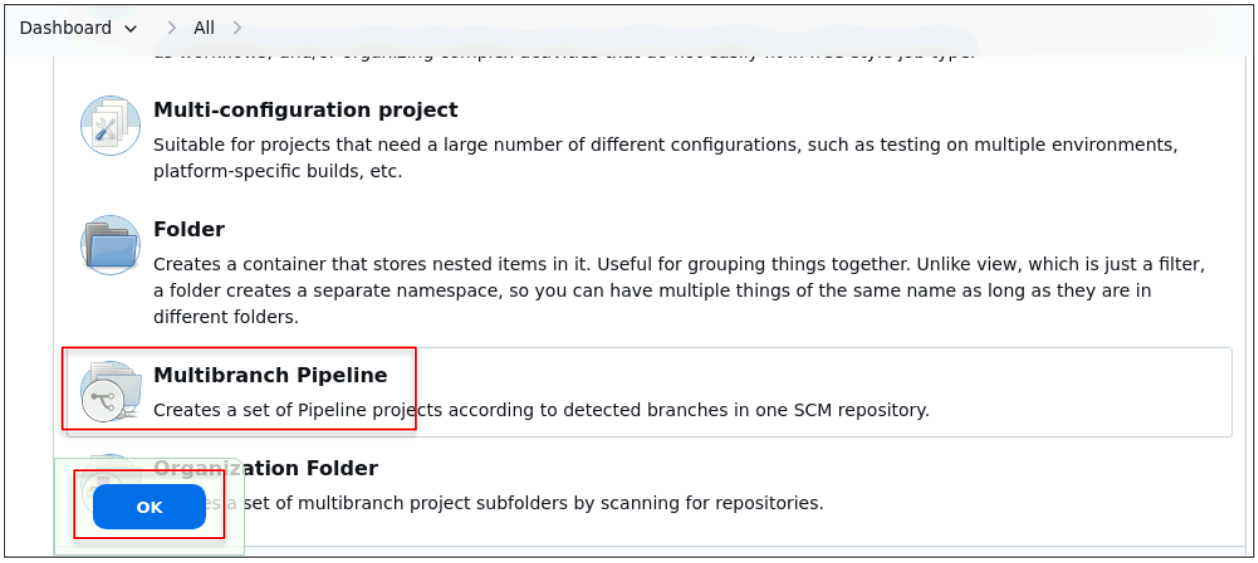
3.2 Click on **New Item** as show in the screenshot below:



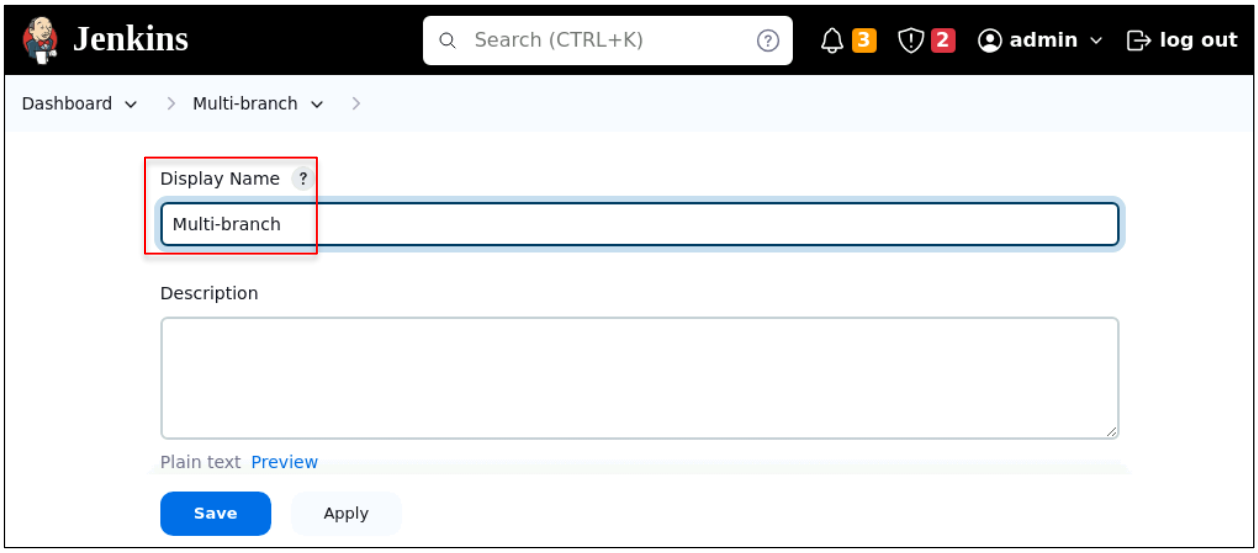
3.3 Enter a desired name for the project as shown in the screenshot below:



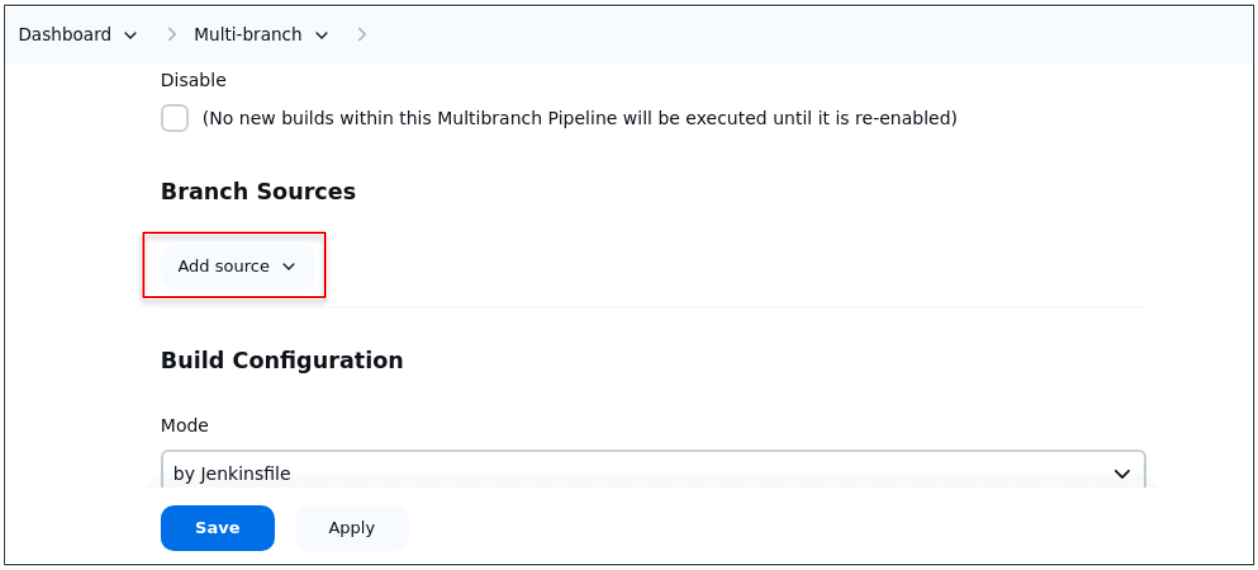
3.4 Scroll down to **Multibranch Pipeline**, click on it, and then click on the **OK** button as shown in the screenshot below:



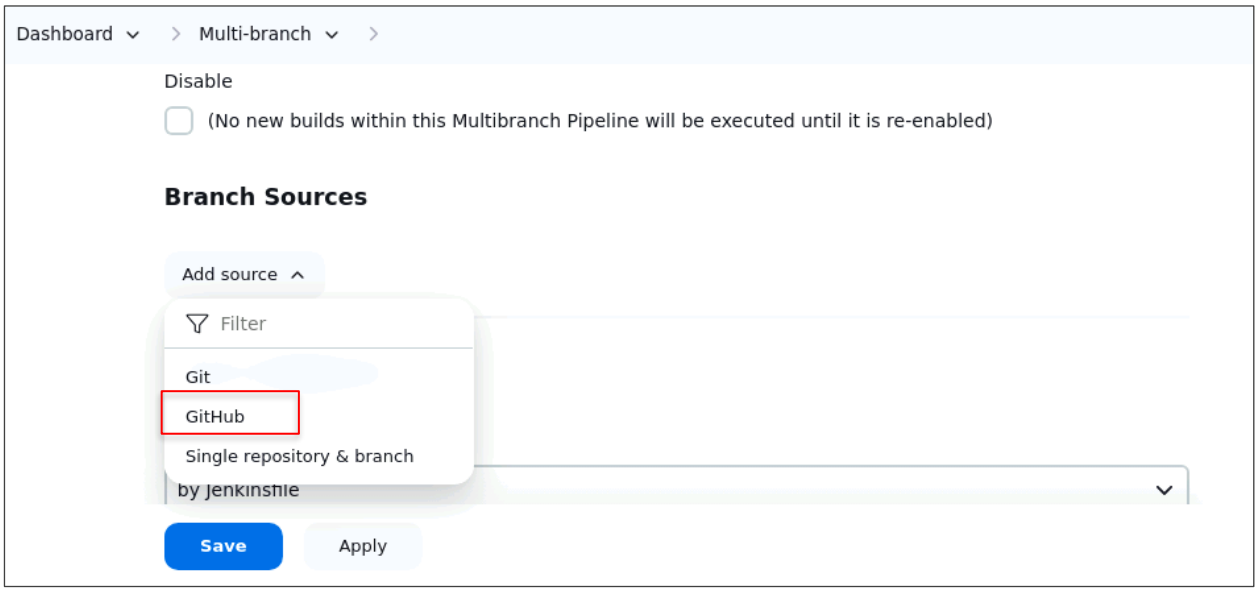
3.5 Enter a desired name for **Display Name** as shown in the screenshot below:



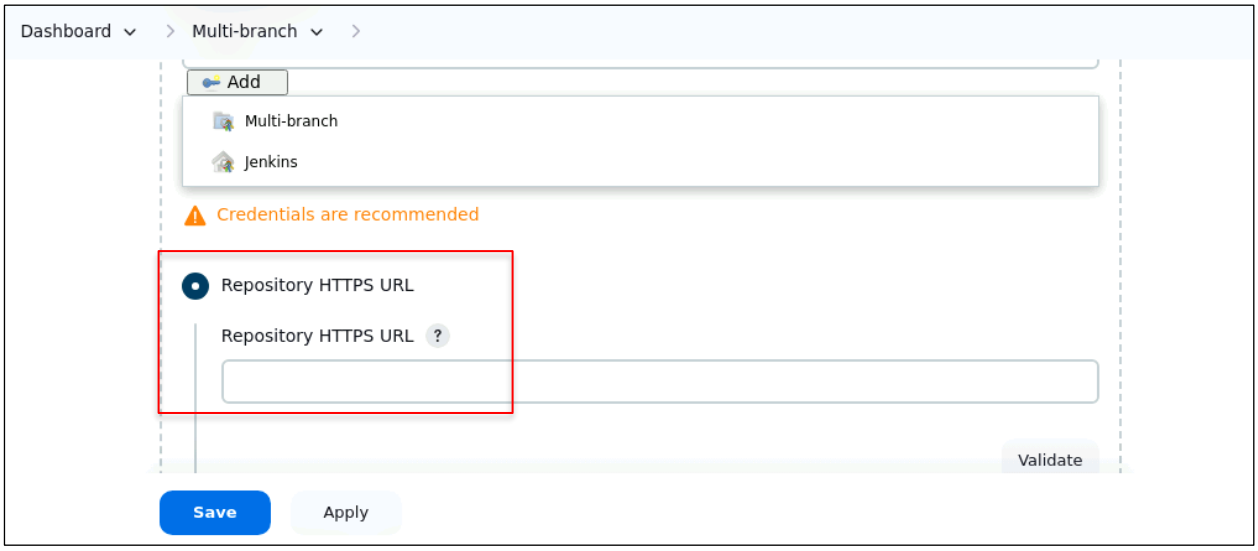
3.6 Scroll down to **Branch Sources** and then click on **Add source** as shown in the screenshot below:



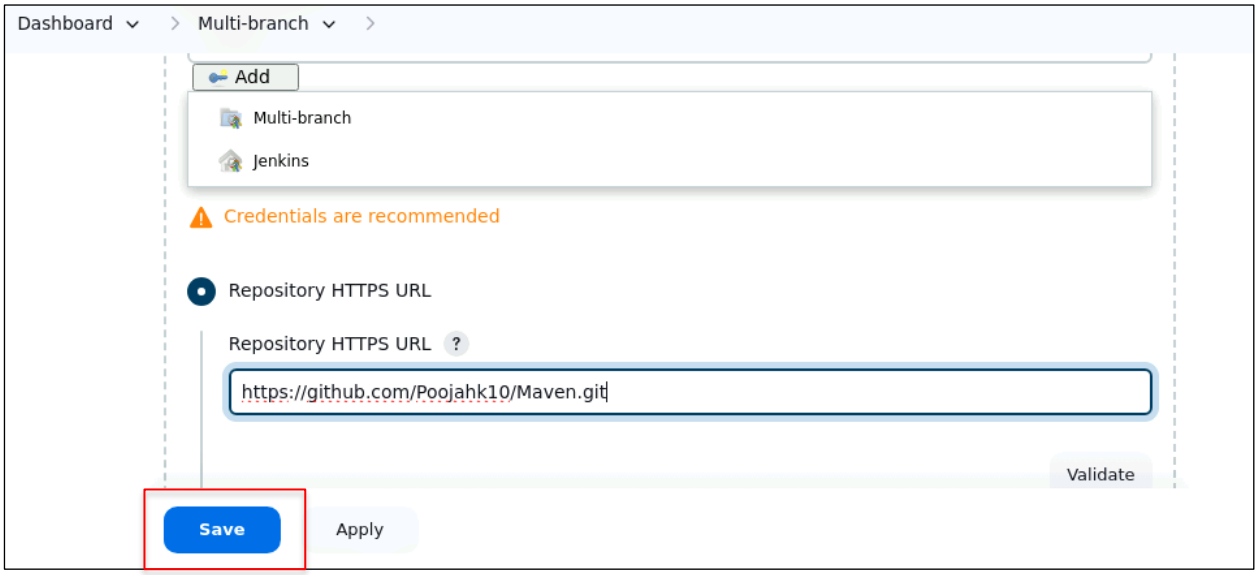
3.7 Click on **GitHub** as shown in the screenshot below:



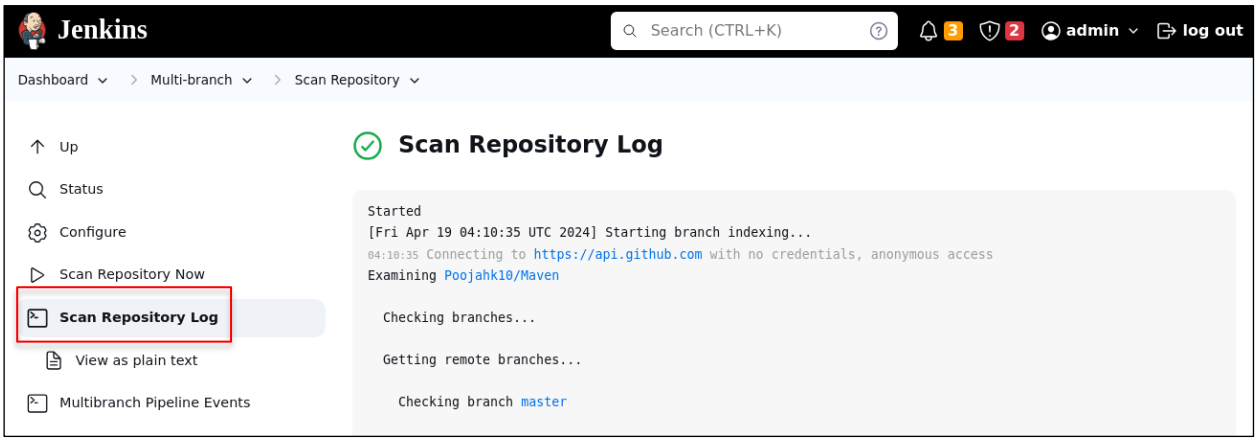
3.8 Scroll down to **Repository HTTPS URL** as shown in the screenshot below**:**



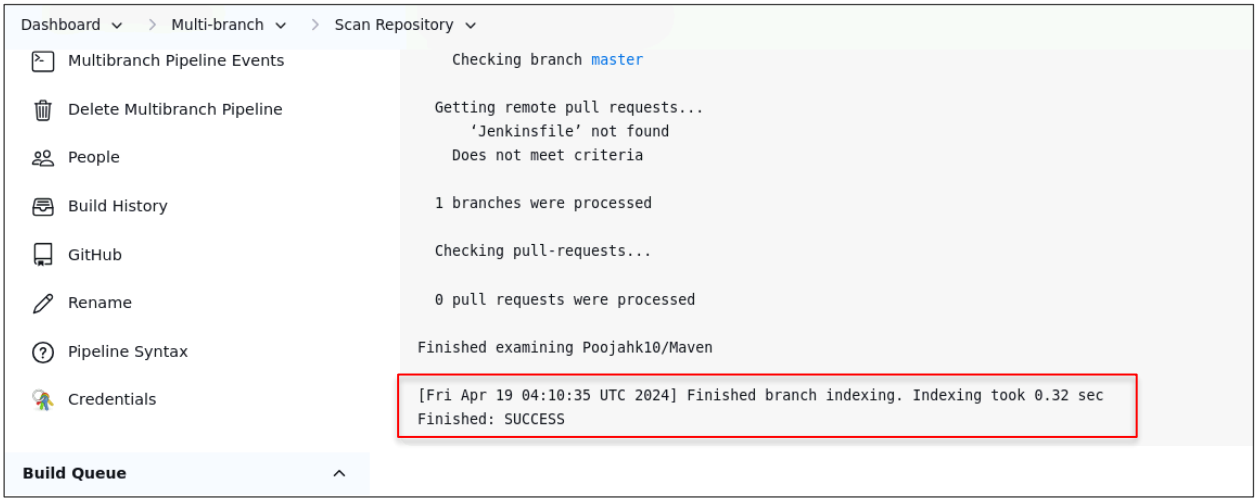
3.9 Enter the repository URL and then click on the **Save** button as shown in the screenshot below:



3.10 Click on **Scan Repository Log** as shown in the screenshot below:



3.11 Scroll down to verify that the status is displayed as **SUCCESS** as shown in the screenshot below:



By following these steps, you have successfully configured a Multibranch Jenkins pipeline job to facilitate managing multiple projects within a single job setup.