

Lesson 06 Demo 01

Setting up Git Configuration in Jenkins Job

Objective: To set up Git configuration in Jenkins for streamlined version control and automated build processes

Tools required: Jenkins

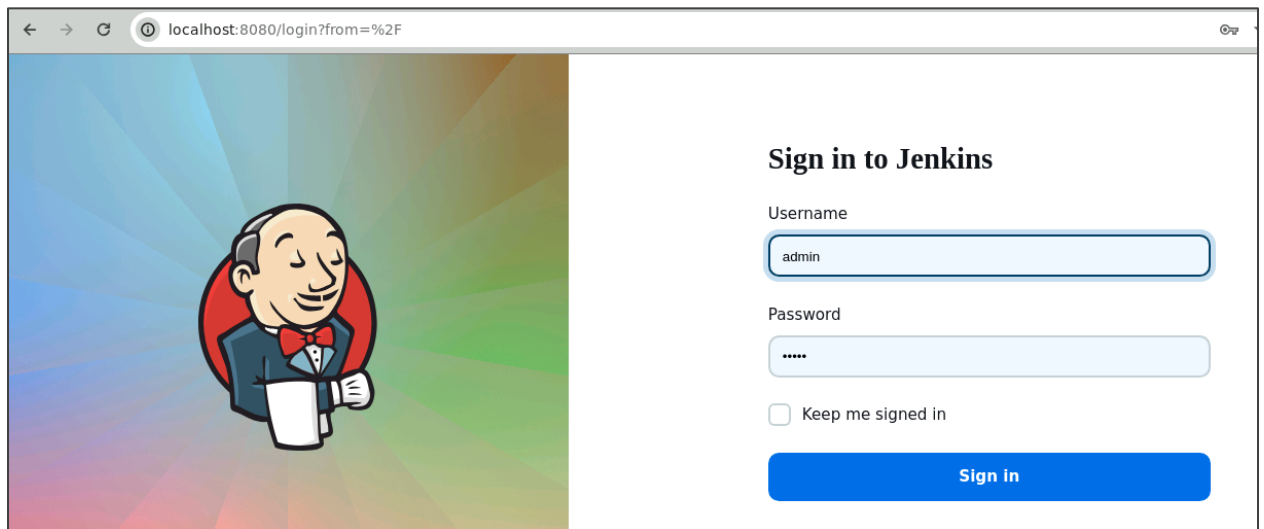
Prerequisites: You need to have a Jenkins up and running.

Steps to be followed:

1. Create new credentials in Jenkins for Git configuration
2. Create a Jenkins freestyle project
3. Configure Git in Jenkins

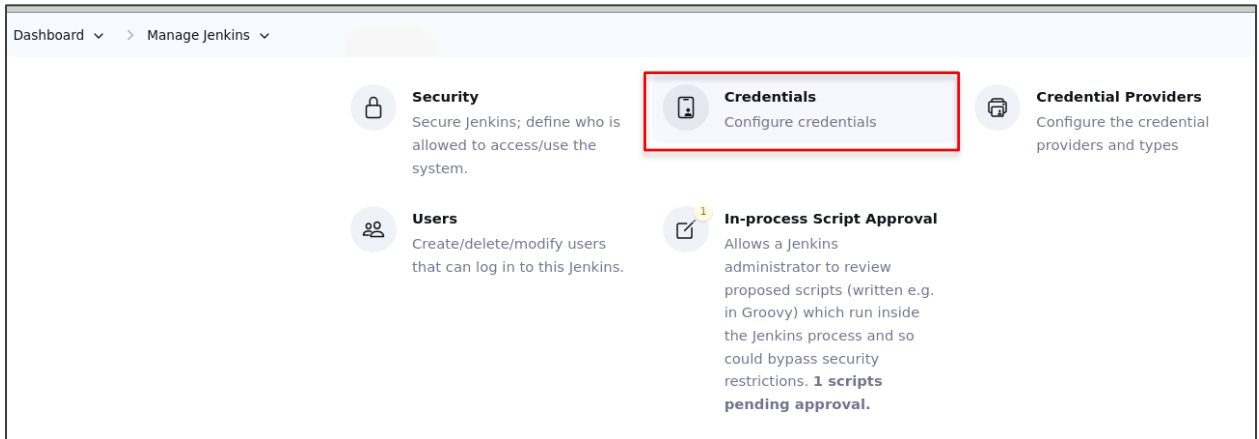
Step 1: Create new credentials in Jenkins for Git configuration

1.1 Log in to Jenkins using credentials

A screenshot of a web browser window showing the Jenkins login page. The browser's address bar displays 'localhost:8080/login?from=%2F'. The page features a large, colorful background image of the Jenkins mascot, a cartoon man in a tuxedo, holding a white cup. To the right of the image, the text 'Sign in to Jenkins' is displayed. Below this, there are two input fields: 'Username' with the text 'admin' and 'Password' with masked characters '.....'. A checkbox labeled 'Keep me signed in' is present below the password field. At the bottom right, there is a blue button labeled 'Sign In'.

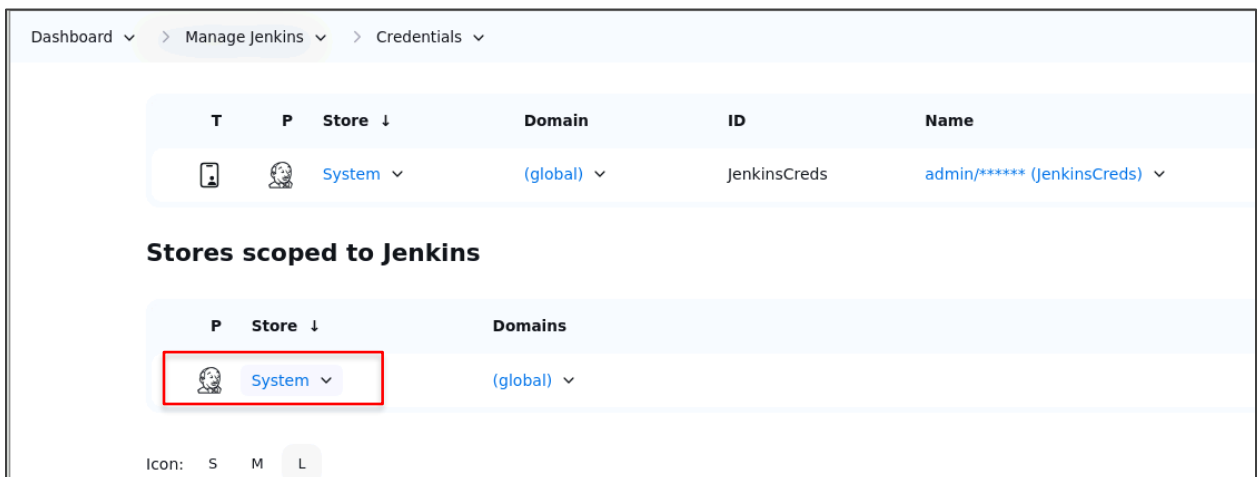
Note: The credentials for accessing Jenkins in the lab are Username: **admin** and Password: **Root123\$**.

1.2 Navigate to **Manage Jenkins** and click on **Credentials**

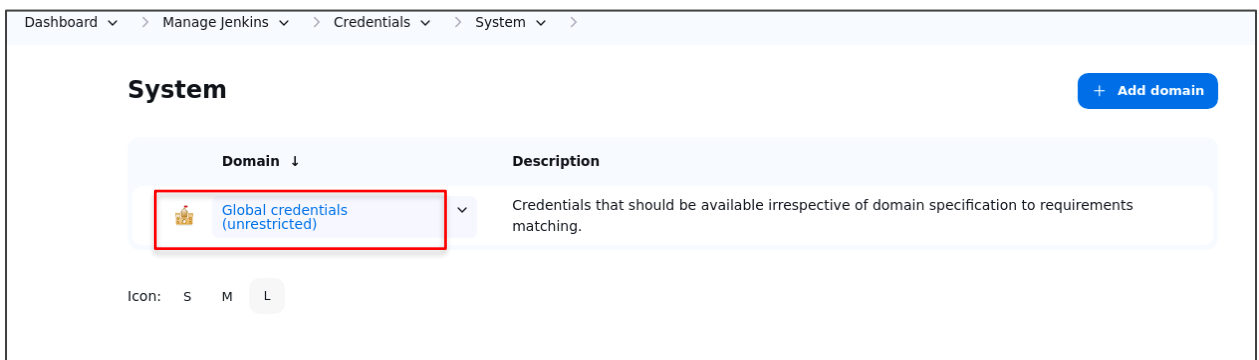


The screenshot shows the Jenkins 'Manage Jenkins' page. The breadcrumb trail is 'Dashboard > Manage Jenkins'. There are four main sections: 'Security' (Secure Jenkins; define who is allowed to access/use the system.), 'Users' (Create/delete/modify users that can log in to this Jenkins.), 'Credentials' (Configure credentials), and 'Credential Providers' (Configure the credential providers and types). The 'Credentials' section is highlighted with a red box. Below it, there is a section for 'In-process Script Approval' with a notification badge showing '1 scripts pending approval'.

1.3 Click on **System** under **Stores scoped to Jenkins** and then click on **Global credentials**

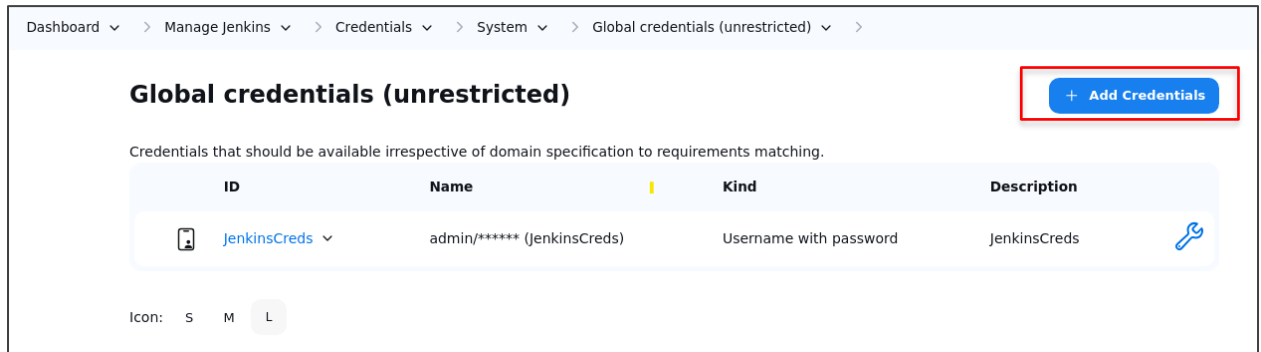


The screenshot shows the Jenkins 'Credentials' page. The breadcrumb trail is 'Dashboard > Manage Jenkins > Credentials'. There is a table with columns: 'T', 'P', 'Store', 'Domain', 'ID', and 'Name'. The table contains one entry: a lock icon, a user icon, 'System', '(global)', 'jenkinsCreds', and 'admin/***** (jenkinsCreds)'. Below the table is a section titled 'Stores scoped to Jenkins'. It contains a table with columns: 'P', 'Store', and 'Domains'. The table contains one entry: a user icon, 'System', and '(global)'. The 'System' entry is highlighted with a red box. At the bottom, there is a legend for 'Icon: S M L'.



The screenshot shows the Jenkins 'System' credentials page. The breadcrumb trail is 'Dashboard > Manage Jenkins > Credentials > System'. There is a button '+ Add domain'. Below it is a table with columns: 'Domain' and 'Description'. The table contains one entry: 'Global credentials (unrestricted)' and 'Credentials that should be available irrespective of domain specification to requirements matching.'. The 'Global credentials (unrestricted)' entry is highlighted with a red box. At the bottom, there is a legend for 'Icon: S M L'.

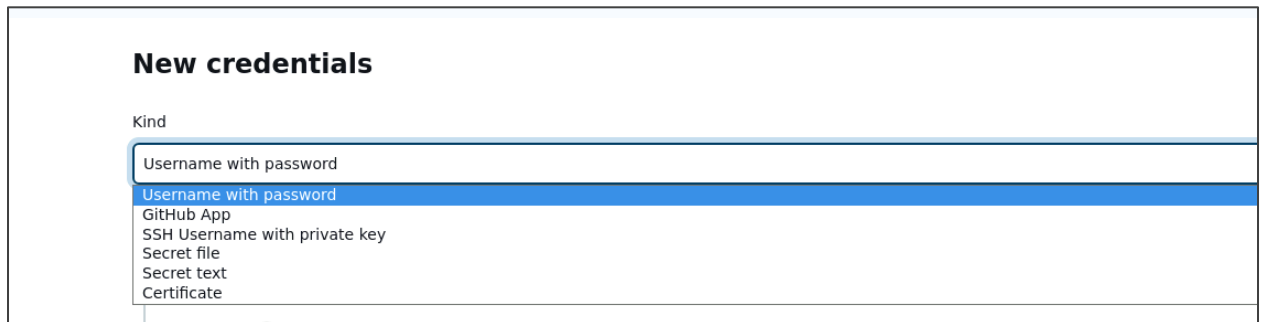
1.4 Click on + Add Credentials



The screenshot shows the Jenkins 'Global credentials (unrestricted)' page. At the top right, a blue button labeled '+ Add Credentials' is highlighted with a red rectangle. Below the header, a table lists existing credentials. The table has columns for ID, Name, Kind, and Description. One credential is listed: ID 'JenkinsCreds', Name 'admin/***** (JenkinsCreds)', Kind 'Username with password', and Description 'JenkinsCreds'. Below the table, there are icons for 'Icon: S M L'.

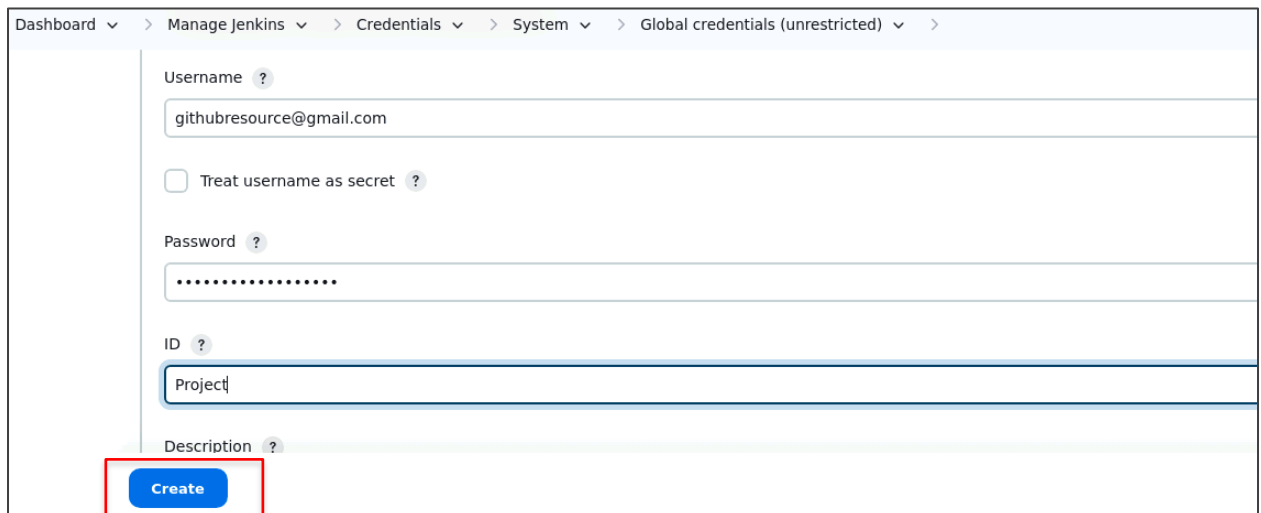
ID	Name	Kind	Description
JenkinsCreds	admin/***** (JenkinsCreds)	Username with password	JenkinsCreds

1.5 Under Kind, select the Username with password option



The screenshot shows the 'New credentials' dialog in Jenkins. The 'Kind' dropdown menu is open, showing a list of options: 'Username with password', 'GitHub App', 'SSH Username with private key', 'Secret file', 'Secret text', and 'Certificate'. The 'Username with password' option is highlighted with a blue background.

1.6 Add your GitHub Username and Password, assign Project as the ID, and click on Create



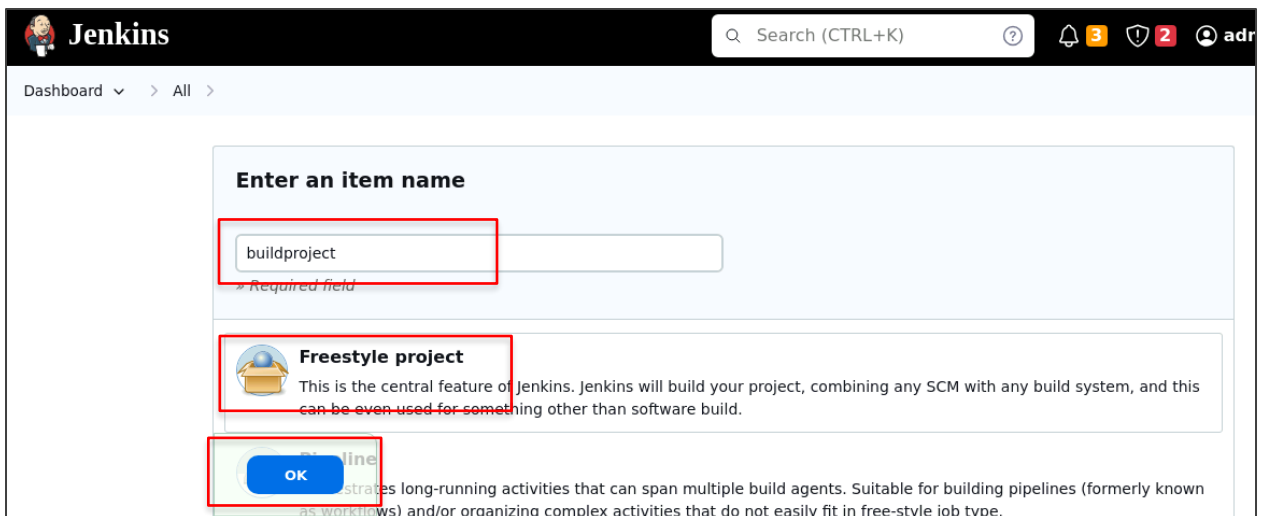
The screenshot shows the 'Create new credentials' form in Jenkins. The form has fields for 'Username', 'Password', 'ID', and 'Description'. The 'Username' field contains 'githubresource@gmail.com'. The 'Password' field is masked with dots. The 'ID' field contains 'Project'. The 'Description' field is empty. At the bottom left, a blue 'Create' button is highlighted with a red rectangle.

Step 2: Create a Jenkins freestyle project

2.1 On the Jenkins dashboard, click on **+ New Item**

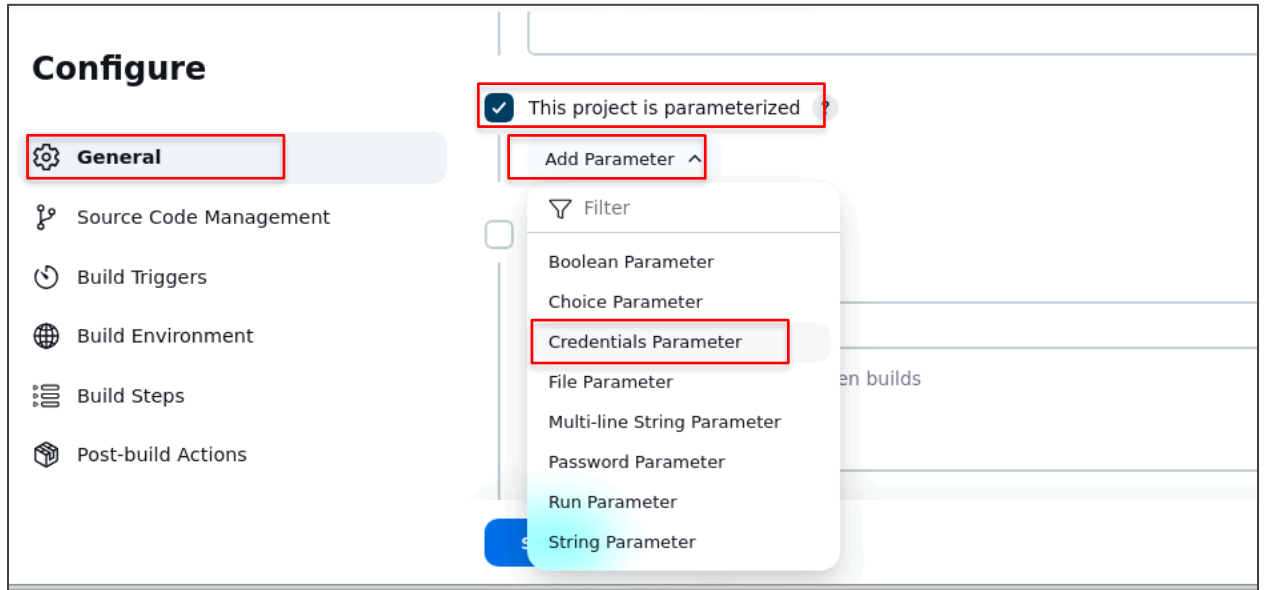


2.2 Enter the project name as **buildproject**, select the **Freestyle project** option, and click on **OK**

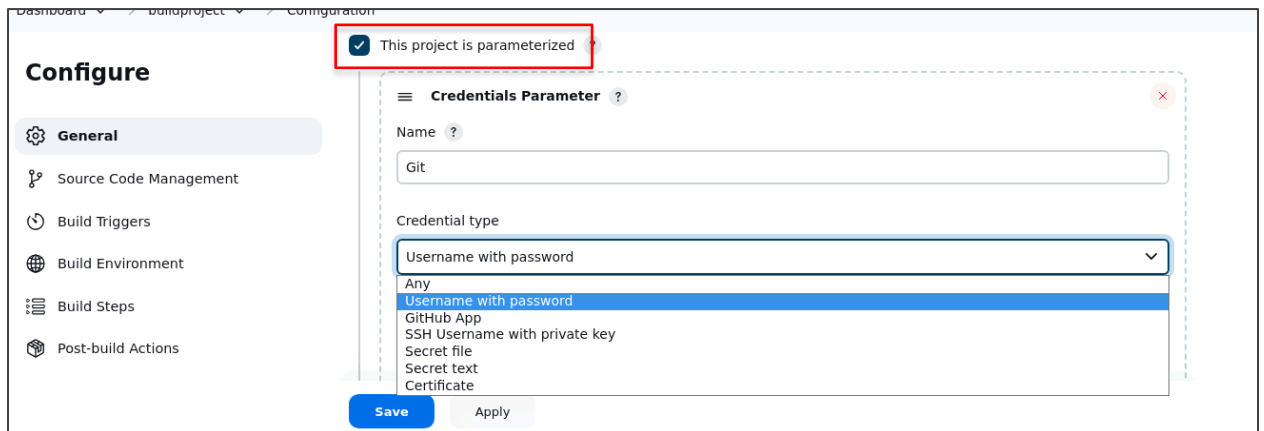


Step 3: Configure Git in Jenkins

3.1 Navigate to the **General** section, select **This project is parameterized**, click on **Add Parameter**, and select **Credentials Parameter**



3.2 Add **Name** as **Git**, select **Username with password** as the **Credential type**, and then choose your created credentials as the **Default value**



Configure

General

- Source Code Management
- Build Triggers
- Build Environment
- Build Steps
- Post-build Actions

☐ Required ?

Default Value ?

githubresource@gmail.com/*****

- none -
admin/***** (JenkinsCreds)
githubresource@gmail.com/*****

Description ?

Git

Save Apply

3.3 Navigate to the **Source Code Management** section, provide the GitHub **Repository URL**, select the added GitHub credentials, and click **Apply** and **Save**

Dashboard > buildproject > Configuration

Configure

Source Code Management

- General
- Build Triggers
- Build Environment
- Build Steps
- Post-build Actions

☐ Git ?

Repositories ?

Repository URL ?

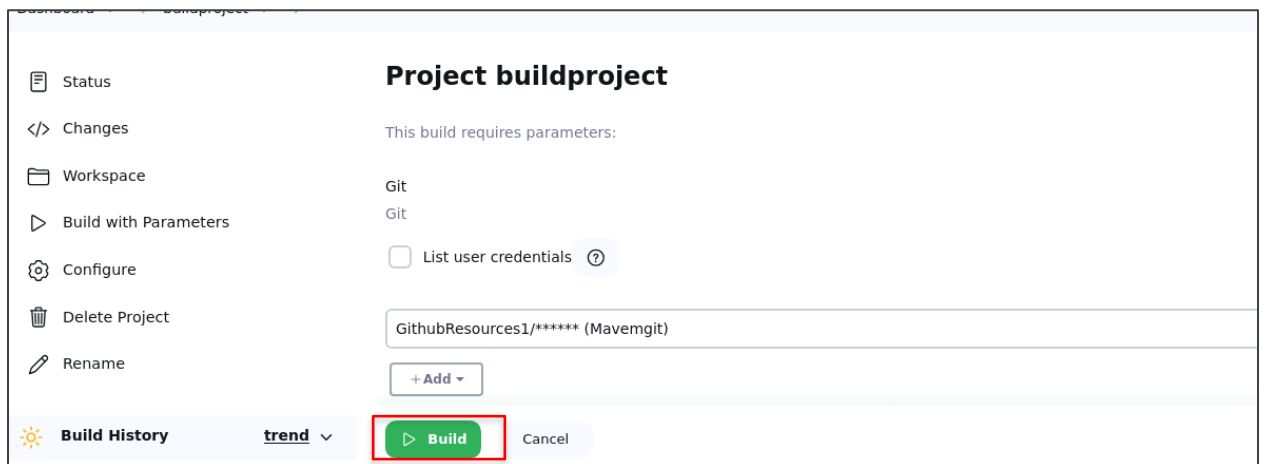
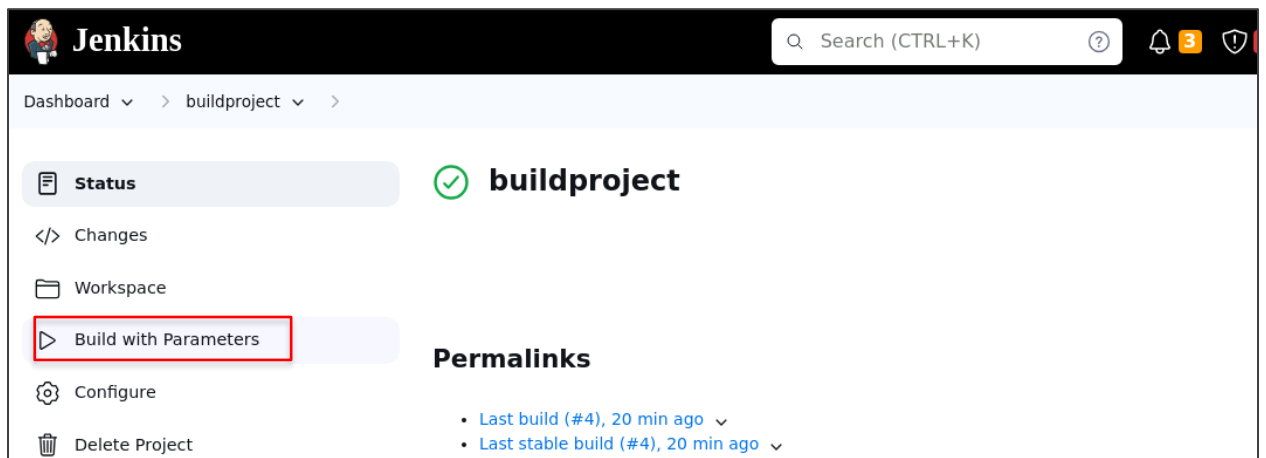
https://github.com/GithubResources1/Maven-Build.git

Credentials ?

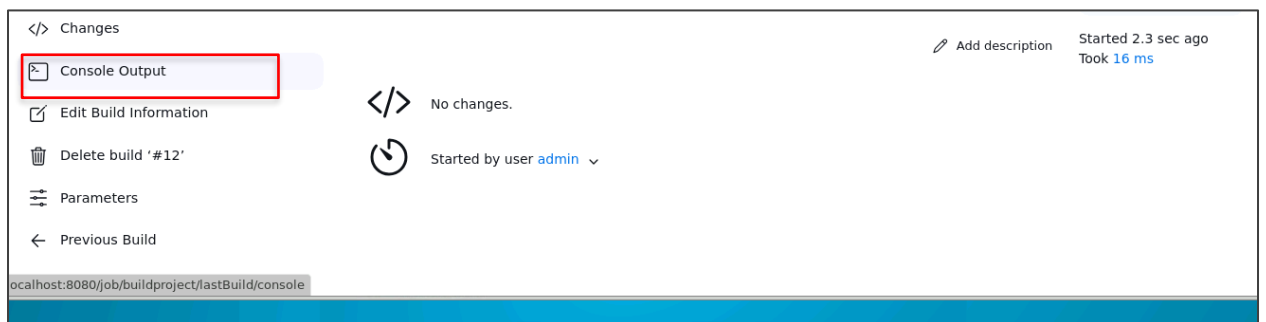
- none -
- none -
admin/***** (JenkinsCreds)
githubresource@gmail.com/*****

Save Apply

3.4 In the Jenkins dashboard, click on **Build with Parameters** under the **Status**, then click on **Build**



3.5 Click on **Console Output** to see the output



Dashboard > buildproject > #1 > Console Output

Status

Changes

Console Output

View as plain text

Edit Build Information

Delete build '#1'

Parameters

Git Build Data

Next Build

✓ Console Output

Started by user [admin](#)
Running as SYSTEM
Building in workspace /var/lib/jenkins/workspace/buildproject
The recommended git tool is: NONE
using credential Maven
Cloning the remote Git repository
Cloning repository <https://github.com/GithubResources1/Maven-Build.git>
> git init /var/lib/jenkins/workspace/buildproject # timeout=10
Fetching upstream changes from <https://github.com/GithubResources1/Maven-Build.git>
> git --version # timeout=10
> git --version # 'git version 2.43.2'
using GIT_ASKPASS to set credentials Mavengit
> git fetch --tags --force --progress -- <https://github.com/GithubResources1/Maven-Build.git>
+refs/heads/*:refs/remotes/origin/* # timeout=10

You can see the output.

By following these steps, you have successfully configured Git in Jenkins enabling streamlined version control and automated build processes.