

Name – Utkarsh Rawat

Sap Id – 500124586

Batch 3 – DevOps

Lab Exercise 6.2

Creating a New Jenkins Job to Checkout Source Code

Objective: To set up a Jenkins job to manage source code, specifically by configuring the Source Code Management section to check out code from a Git repository

Tools required: Jenkins

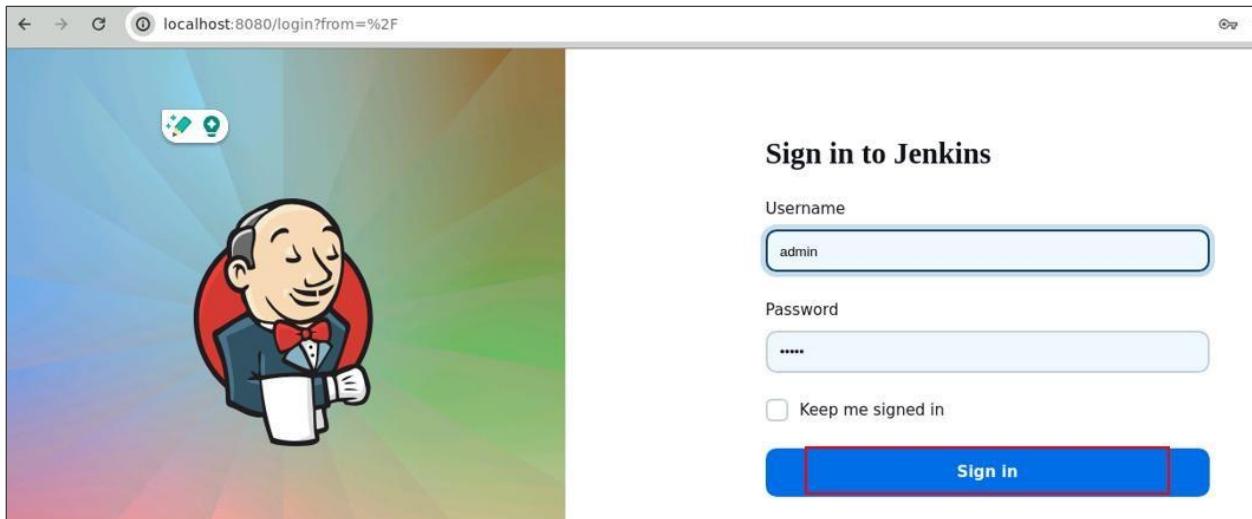
Prerequisites: Jenkins must be operational.

Steps to be followed:

1. Log in and create a Jenkins job
2. Configure source code management

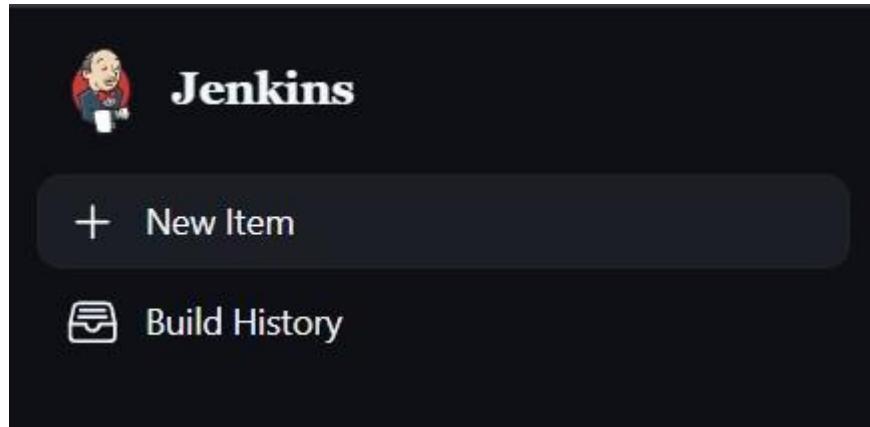
Step 1: Log in and create a Jenkins job

- 1.1 Navigate to **localhost:8080** in your web browser, enter your credentials, and click on **Sign In**



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

1.2 Create a new Jenkins job by clicking on **New Item**



1.3 Provide custom job name inside the field **Enter an item name**, select the **Freestyle project** option, and click on the **OK** button to save the job

New Item

Enter an item name

Freestyle

Select an item type

Freestyle project
Classic, general-purpose job type that checks out from up to one SCM, executes build steps serially, followed by post-build steps like archiving artifacts and sending email notifications.

Pipeline
Orchestrates long-running activities that can span multiple build agents. Suitable for building pipelines (formerly known as workflows) and/or organizing complex activities that do not easily fit in free-style job type.

Multi-configuration project
Suitable for projects that need a large number of different configurations, such as testing on multiple environments, platform-specific builds, etc.

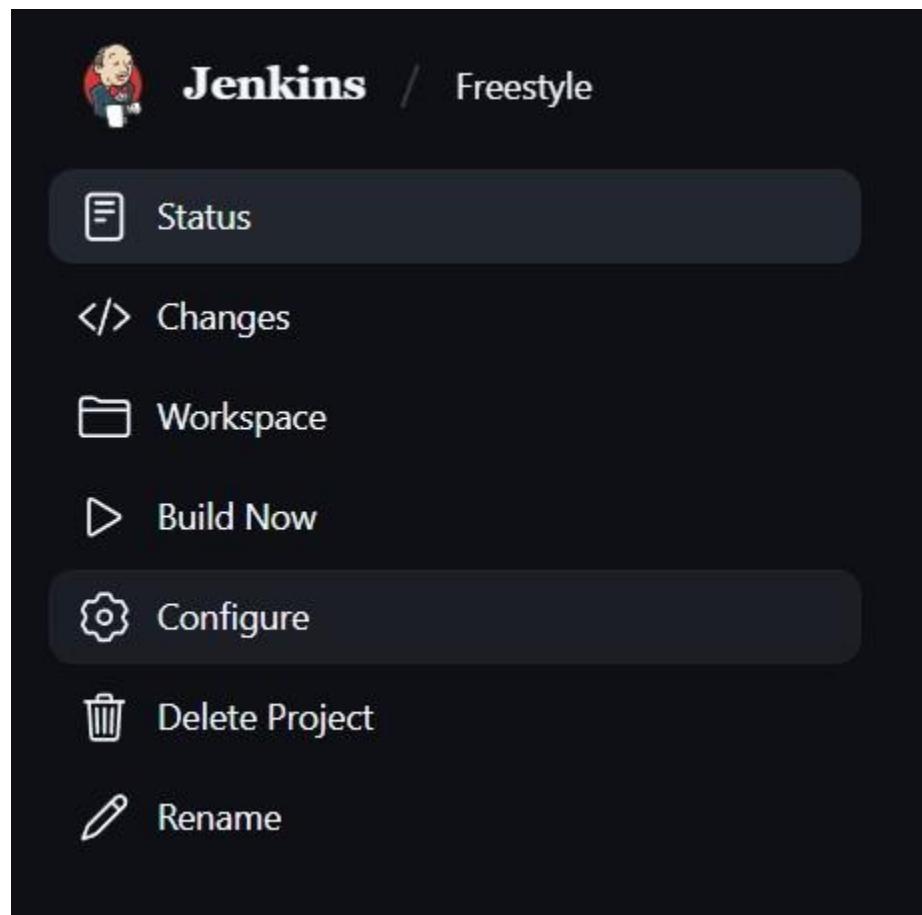
Folder
Creates a container that stores nested items in it. Useful for grouping things together. Unlike view, which is just a filter, a folder creates a separate namespace, so you can have multiple things of the same name as long as they are in different folders.

Multibranch Pipeline
Creates a set of Pipeline projects according to detected branches in one SCM repository.

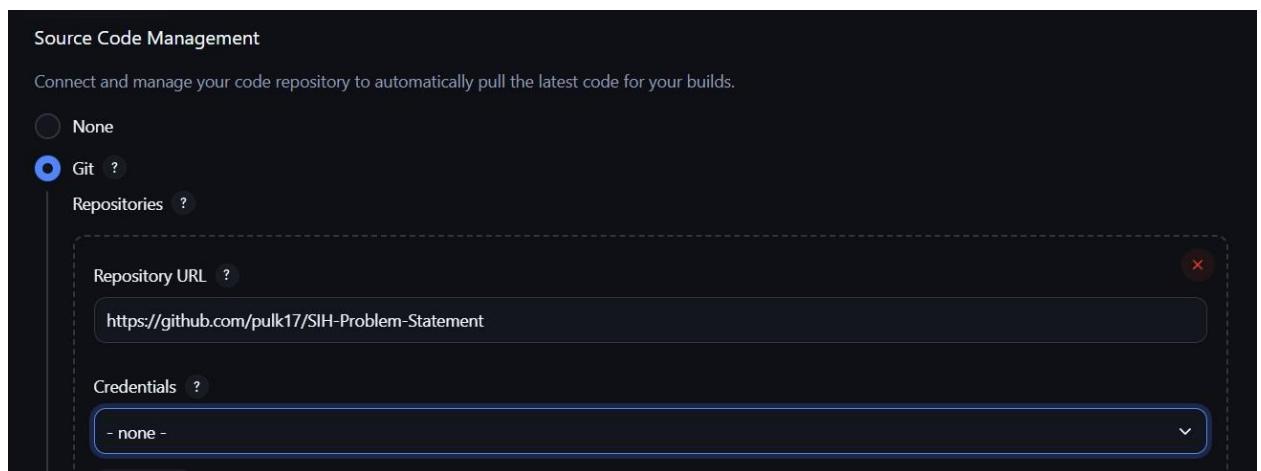
OK

Step 2: Configure source code management

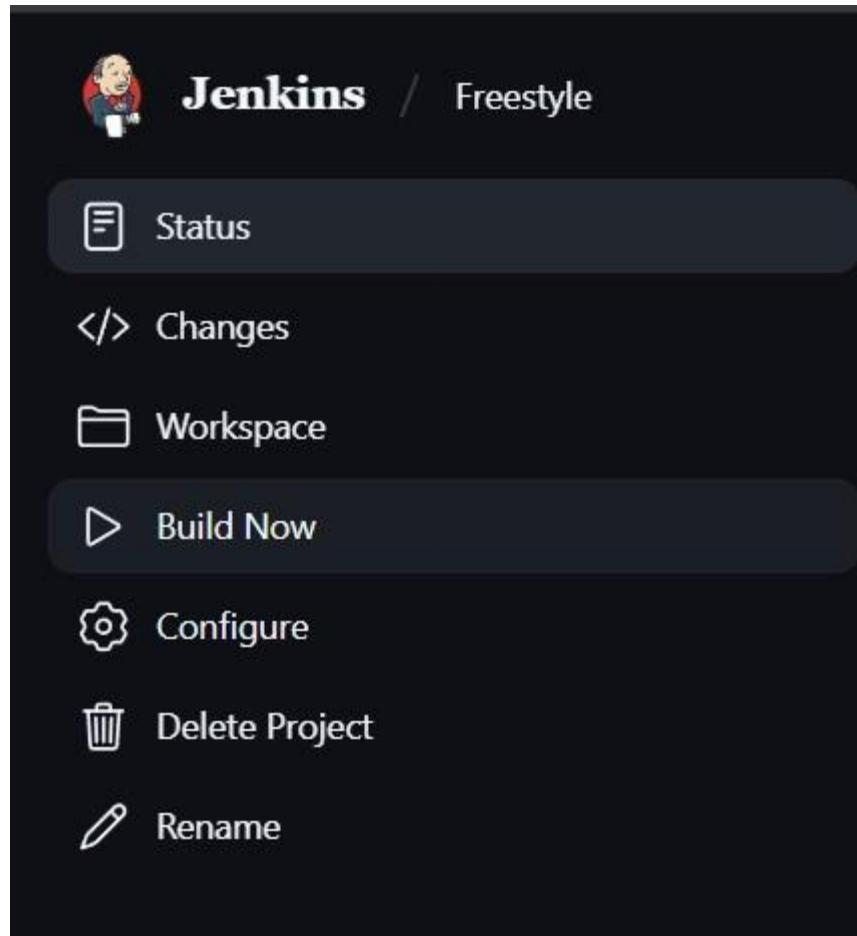
2.1 Access the newly created job's configuration screen by clicking on **Configure**



2.2 Navigate to the **Source Code Management** tab, provide Git repository configuration inside the **Repository URL** field, and click on the **Save** button



2.3 Then, click on the **Build Now** option to schedule a build



2.4 To schedule the build, click the required link under **Permalinks**

A screenshot of a Jenkins job run details page. At the top, it shows a green checkmark icon next to "#3 (Aug 27, 2025, 11:08:58 AM)". Below that is a circular progress bar icon with a play button symbol, followed by the text "Started by user Pulkit Chauhan". Underneath is another circular progress bar icon with a stopwatch symbol, followed by the text "This run spent:" and a bulleted list: "• 3 ms waiting; • 2.1 sec build duration; • 2.1 sec total from scheduled to completion." Further down, there's a "git" icon with a pull request symbol, followed by the text "Revision: e2ed0caf252c3ebadc7b5514ab5f49da6f4e05ce" and "Repository: https://github.com/pulk17/SIH-Problem-Statement" with a bulleted list: "• refs/remotes/origin/main". At the bottom, there's a double-angle bracket icon followed by the text "No changes."

2.5 Click on **Console Output** to check out the process during the build process

```
Running as SYSTEM
Building in workspace C:\ProgramData\Jenkins\.jenkins\workspace\Freestyle
The recommended git tool is: NONE
No credentials specified
> git.exe rev-parse --resolve-git-dir C:\ProgramData\Jenkins\.jenkins\workspace\Freestyle\.git # timeout=10
Fetching changes from the remote Git repository
> git.exe config remote.origin.url https://github.com/pulk17/SIH-Problem-Statement # timeout=10
Fetching upstream changes from https://github.com/pulk17/SIH-Problem-Statement
> git.exe --version # timeout=10
> git --version # 'git version 2.47.1.windows.2'
> git.exe fetch --tags --force --progress -- https://github.com/pulk17/SIH-Problem-Statement +refs/heads/*:refs/remotes/origin/* # timeout=10
> git.exe rev-parse "refs/remotes/origin/main^{commit}" # timeout=10
Checking out Revision e2ed0caf252c3ebadc7b5514ab5f49da6f4e05ce (refs/remotes/origin/main)
> git.exe config core.sparsecheckout # timeout=10
> git.exe checkout -f e2ed0caf252c3ebadc7b5514ab5f49da6f4e05ce # timeout=10
Commit message: "another fix"
First time build. Skipping changelog.
Finished: SUCCESS
```

2.6

By following these steps, you have successfully set up a Jenkins job to automatically check out source code from a Git repository, enabling seamless integration and automation in your CI/CD pipeline.