

Lab Exercise 8

Setting up Maven Build Job in Jenkins

Objective: To set up Maven build job in Jenkins for automating the build process, enabling continuous integration to enhance the software development lifecycle

Tools required: Jenkins

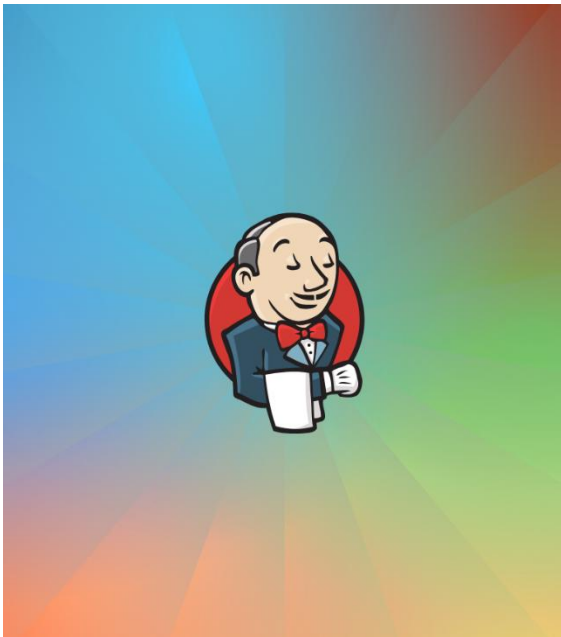
Prerequisites: You need to have a Jenkins up and running.

Steps to be followed:

1. Log in to Jenkins CI tool and configure Maven freestyle job

Step 1: Log in to Jenkins CI tool and configure Maven freestyle job

1.1 Log in to Jenkins using your credentials



Sign in to Jenkins

Username

namit53

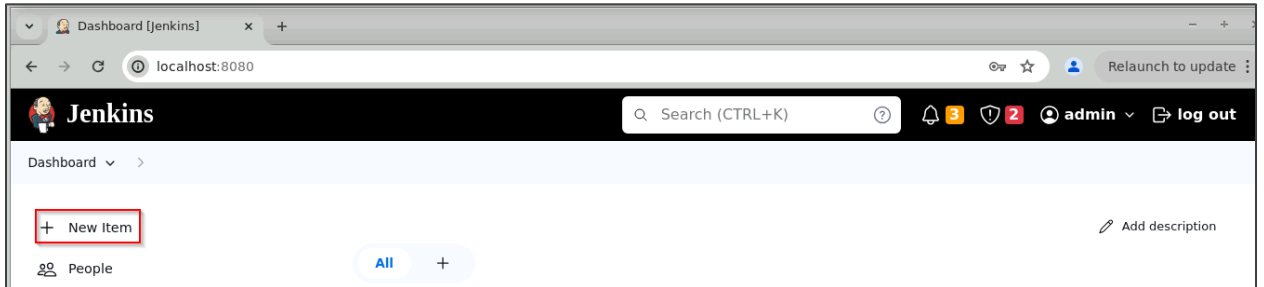
Password

☐ Keep me signed in

Sign in

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

1.2 In the Jenkins dashboard, click on **New Item**



1.3 Select the **Freestyle project** while creating a Jenkins job, provide a custom job name, and click on **OK**

New Item

Enter an item name

LabExercise 8

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.



Maven project

Build a maven project. Jenkins takes advantage of your POM files and drastically reduces the configuration.



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

OK

Configure

General

Source Code Management

Triggers

Environment

Build Steps

Post-build Actions

Source Code Management

Connect and manage your code repository to automatically pull the latest code for your builds.

☐ None

☒ Git ?

Repositories ?

Repository URL ?

https://github.com/namit53/DEVSECOPS-MAVEN-REPO

Please enter Git repository.

Credentials ?

- none -

+ Add

Advanced

Save

Apply

1.4 Now, in the Configure page, navigate to **Source Code Management** in the left navigation bar, select **Git**, and then provide the Git repository URL

Source Code Management

Connect and manage your code repository to automatically pull the latest code for your builds.

☐ None
☒ **Git** ?

Repositories ?

Repository URL ?

`https://github.com/hkshitesh/DEVSECOPS-MAVEN-REPO.git`

Credentials ?

- none -

+ Add

Advanced ▾

Add Repository

Note: Here, the repository URL is <https://github.com/hkshitesh/DEVSECOPS-MAVEN-REPO.git>.

1.5 Now, navigate to **Build Steps**, click on **Add build step**, and then select the option **Invoke top-level Maven targets**

Jenkins / LabExercise 8 / Configuration / Poll SCM ?

Configure

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

Environment

Configure settings and variables that define the context in which your build runs, like credentials, paths, and global parameters.

☐ Delete workspace before build starts
☐ Use secret text(s) or file(s) ?

Filter Matches to the Console Output

- Execute Windows batch command
- Execute shell
- Invoke Ant
- Invoke Gradle script
- Invoke top-level Maven targets**
- Run with timeout
- Set build status to "pending" on GitHub commit

Add build step ^

Post-build Actions

Define what happens after a build completes, like sending notifications, archiving artifacts, or triggering other jobs.

Save Apply

1.6 Provide clean install under **Goals** section and then click on **Save**

Jenkins / LabExercise 8 / Configuration

Configure

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

Build Steps

Automate your build process with ordered tasks like code compilation, testing, and deployment.

Invoke top-level Maven targets ?

Maven Version

(Default)

Goals

clean install

Advanced ▾

Add build step ▾

Save Apply

1.7 You will be navigated to the project after clicking on Save. Now, click on **Build Now** to initiate a new build, and the build logs will display the progress of the build process.

Jenkins / LabExercise 8 / #4 / Console Output

```

Progress (2): 127/193 kB | 49/157 kB
Progress (2): 144/193 kB | 49/157 kB
Progress (2): 144/193 kB | 66/157 kB
Progress (2): 144/193 kB | 82/157 kB
Progress (2): 160/193 kB | 82/157 kB
Progress (2): 160/193 kB | 98/157 kB
Progress (2): 176/193 kB | 98/157 kB
Progress (2): 176/193 kB | 115/157 kB
Progress (2): 193 kB | 115/157 kB
Progress (2): 193 kB | 131/157 kB
Progress (2): 193 kB | 147/157 kB
Progress (2): 193 kB | 157 kB

Downloaded from central: https://repo.maven.apache.org/maven2/org/codehaus/plexus/plexus-utils/4.0.1/plexus-utils-4.0.1.jar (193 kB at 1.4 MB/s)
Downloaded from central: https://repo.maven.apache.org/maven2/org/apache/maven/resolver/maven-resolver-api/1.9.18/maven-resolver-api-1.9.18.jar (157 kB at 1.0 MB/s)
[INFO] Installing C:\ProgramData\Jenkins\jenkins\workspace\LabExercise 8\pom.xml to
C:\WINDOWS\system32\config\systemprofile\.m2\repository\upes\devops\upes.devops\0.0.1-SNAPSHOT\upes.devops\0.0.1-SNAPSHOT.pom
[INFO] Installing C:\ProgramData\Jenkins\jenkins\workspace\LabExercise 8\target\upes-calc.jar to
C:\WINDOWS\system32\config\systemprofile\.m2\repository\upes\devops\upes.devops\0.0.1-SNAPSHOT\upes.devops\0.0.1-SNAPSHOT.jar
[INFO] -----
[INFO] BUILD SUCCESS
[INFO] -----
[INFO] Total time: 41.486 s
[INFO] Finished at: 2025-08-30T20:59:01+05:30
[INFO] -----
Finished: SUCCESS
  
```

**Jenkins**

/ LabExercise 8 / #4

Status

Changes

Console Output

Edit Build Information

Delete build '#4'

Timings

Git Build Data

Previous Build

**#4 (Aug 30, 2025, 8:58:15 PM)**Started by user [Namit Rampal](#)

This run spent:

- 14 ms waiting;
- 46 sec build duration;
- 46 sec total from scheduled to completion.

**Revision:** 04a6890beda18d9a0604c2a45835912c4cdb966b**Repository:** <https://github.com/namit53/DEVSECOPS-MAVEN-REPO.git>

- refs/remotes/origin/master



No changes.

You can see that the build is configured successfully.

By following these steps, you have successfully set up Maven build job in Jenkins for automating the build process, enabling continuous integration to enhance the software development lifecycle.