

Lesson 06 Demo 02

Integrating Maven with Jenkins

Objective: To install the Maven plugin in Jenkins for smooth integration and automation of Maven-based build processes within the Jenkins environment

Tools required: Git, GitHub, and Jenkins

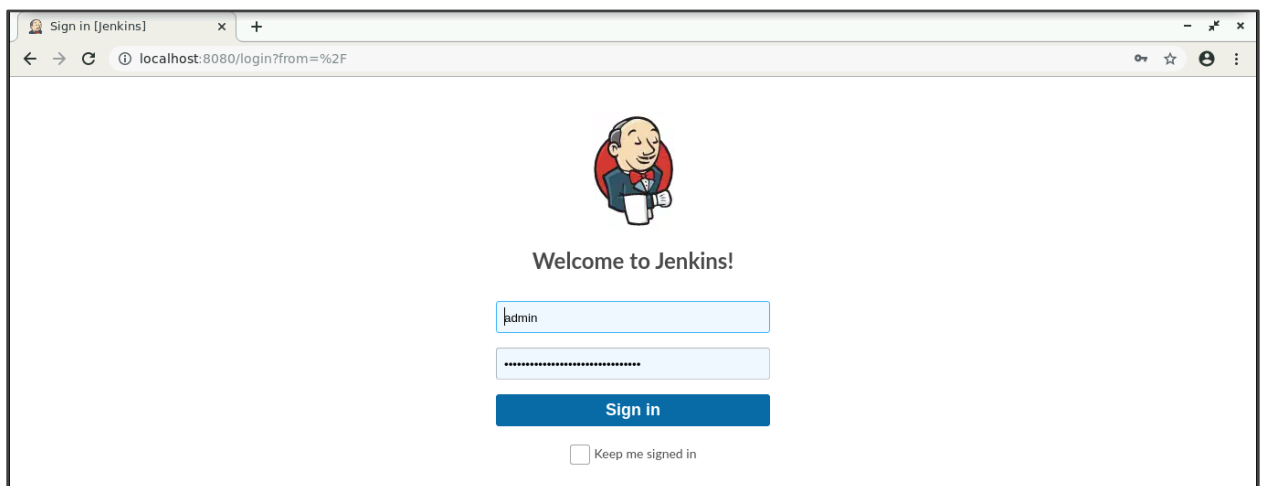
Prerequisites: None

Steps to be followed:

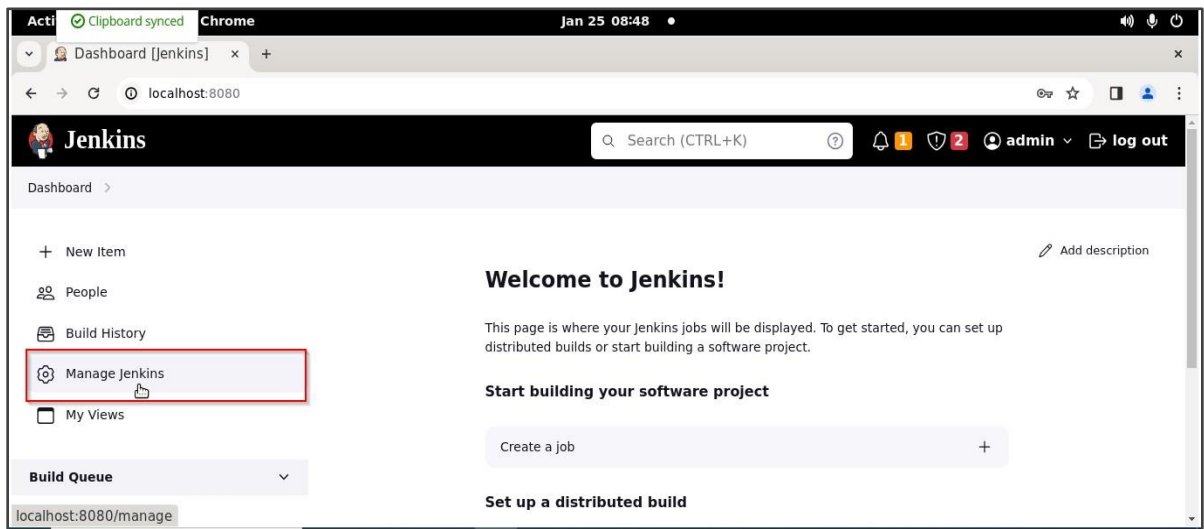
1. Install the Maven plugin
2. Set up Global Tool Configuration
3. Fork a sample repository
4. Integrate Maven with Jenkins

Step 1: Install the Maven plugin

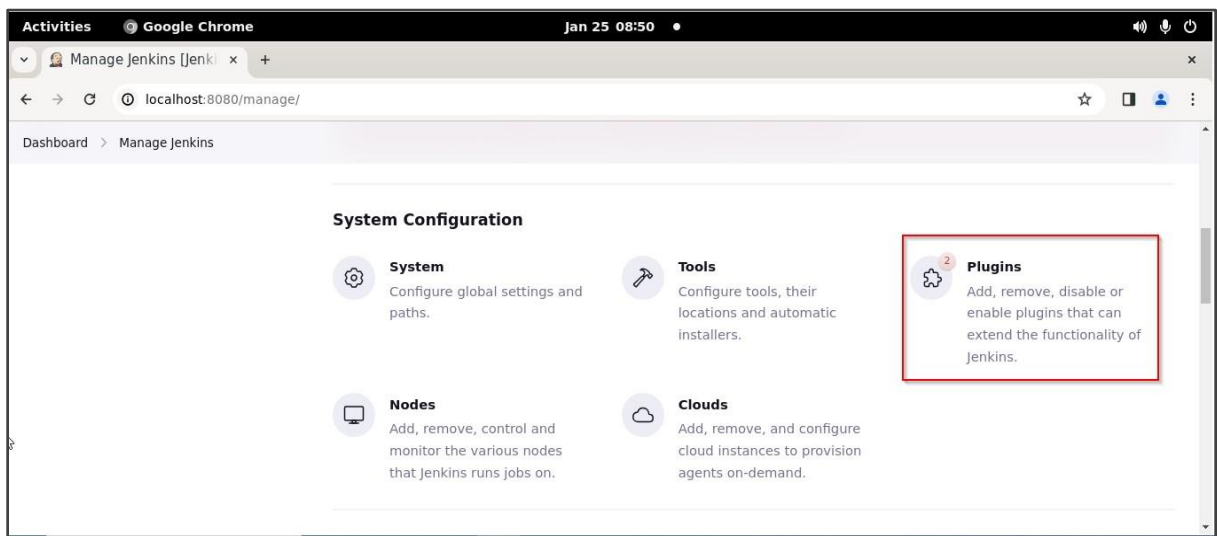
- 1.1 Open the browser, go to the Jenkins Dashboard by typing **localhost:8080** in your browser, provide the credentials, and click the **Sign in** button



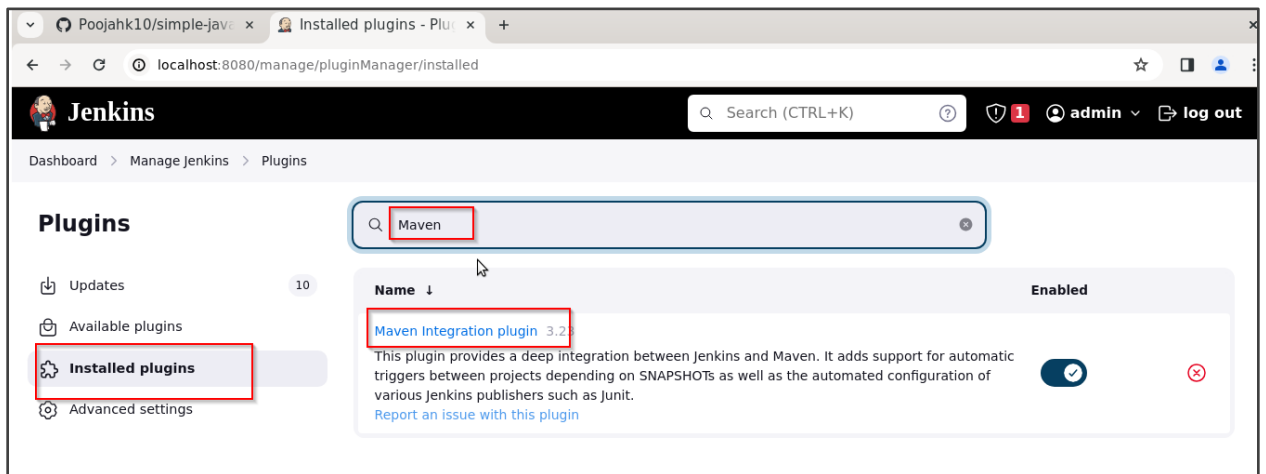
1.2 Click on the **Manage Jenkins** option as shown in the screenshot below:



1.3 Click on the **Plugins** option as shown in the screenshot below:



1.4 Click on **Installed plugins** to verify whether the **Maven Integration plugin** has been installed



Note: Maven is already installed in your practice lab environment. If not, click on **Available plugins**, search for the Maven Integration plugin, and install it.

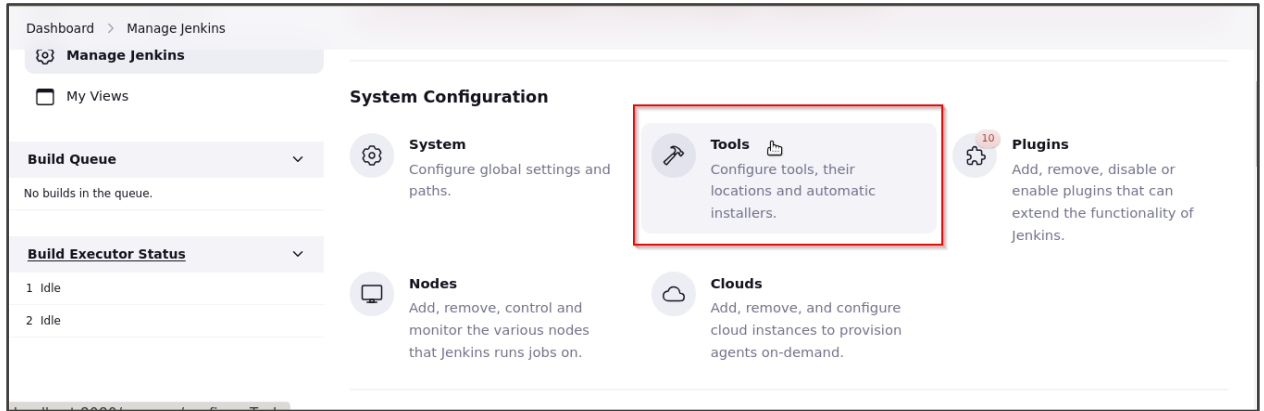
1.5 Use the following command to check the Maven version:

mvn -version

```
labsuser@ip-172-31-39-225:~$ mvn -version
Apache Maven 3.6.3
Maven home: /usr/share/maven
Java version: 11.0.21, vendor: Ubuntu, runtime: /usr/lib/jvm/java-11-openjdk-amd64
Default locale: en, platform encoding: UTF-8
OS name: "linux", version: "6.2.0-1018-aws", arch: "amd64", family: "unix"
labsuser@ip-172-31-39-225:~$
```

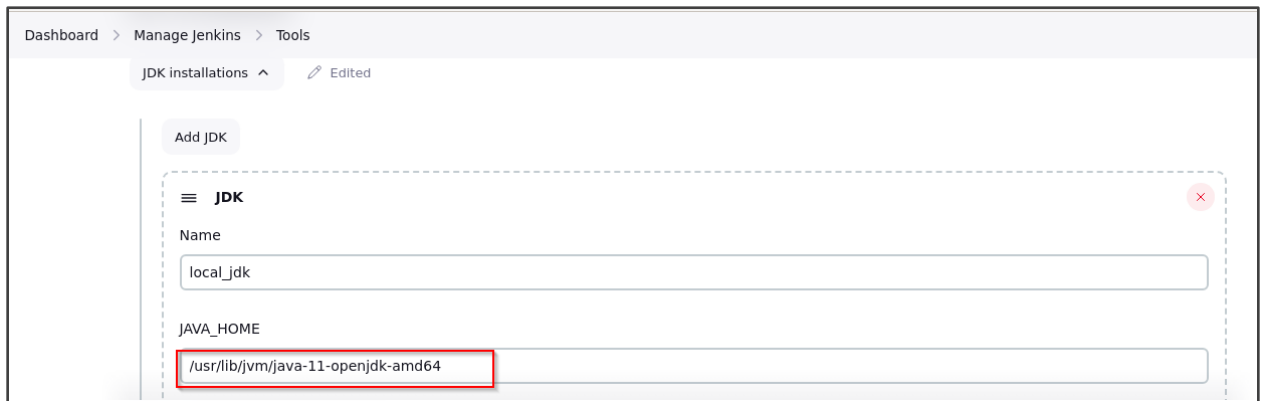
Step 2: Set up Global Tool Configuration

2.1 Go to the Jenkins Dashboard, click on **Manage Jenkins**, and then select **Tools** from the list of options



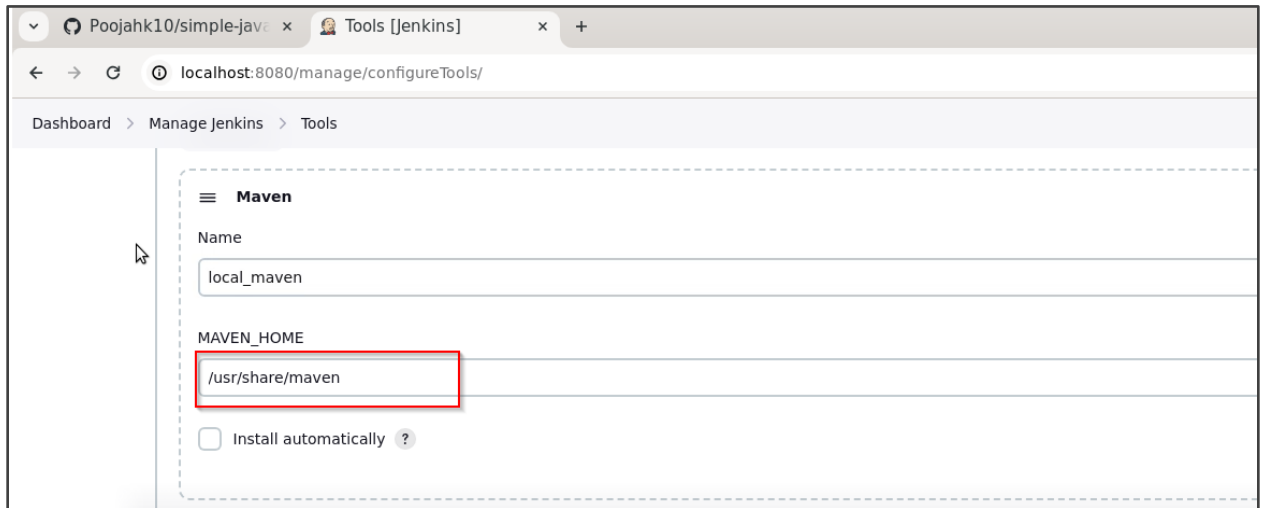
2.2 Click on **JDK installations** and provide the **Name** and **JAVA_HOME** path

Note: Set the **JAVA_HOME** environment variable to **/usr/lib/jvm/java-11-openjdk-amd64**



2.3 To configure Maven, click on the **Maven installations** button in the Maven section and enter a **Name** and **MAVEN_HOME** path

Note: Set the **MAVEN_HOME** environment variable to **/usr/share/maven**



The screenshot shows the Jenkins web interface at the URL `localhost:8080/manage/configureTools/`. The breadcrumb navigation is `Dashboard > Manage Jenkins > Tools`. On the left sidebar, the `Tools` section is selected. The main content area is titled `Maven` and contains the following fields:

- Name:** `local_maven`
- MAVEN_HOME:** `/usr/share/maven` (this field is highlighted with a red rectangle)
- Install automatically:** ☐ (with a help icon)

2.4 To configure Git, click on **Git installations** and add the **Name** and **Path to Git executable**

Note: Set the **Path to Git executable** environment variable to **/bin/git** and click on **Save**

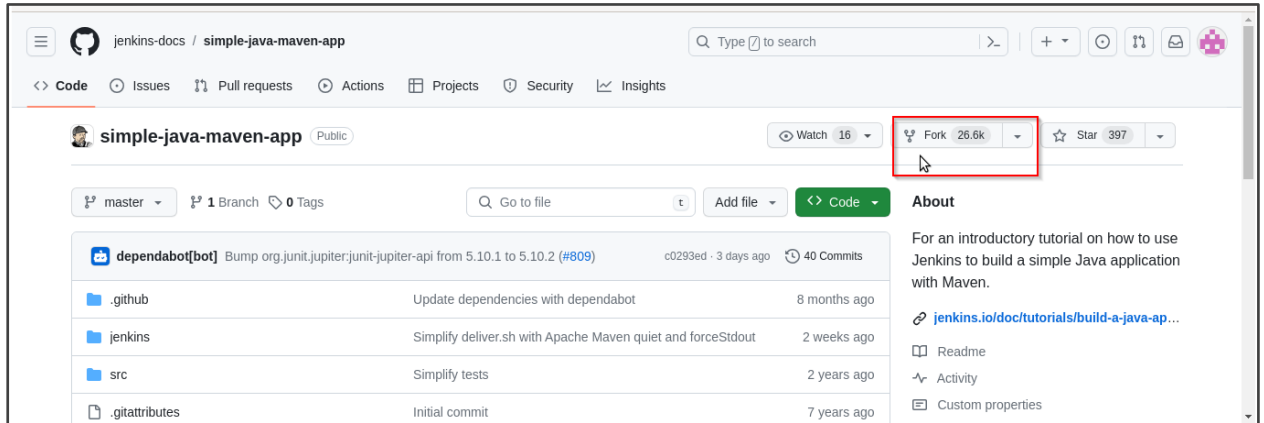


The screenshot shows the Jenkins web interface for the `Git` tool configuration. The breadcrumb navigation is `Dashboard > Manage Jenkins > Tools`. On the left sidebar, the `Tools` section is selected. The main content area is titled `Git` and contains the following fields:

- Name:** `local_git`
- Path to Git executable:** `/bin/git` (this field is highlighted with a red rectangle)
- Install automatically:** ☐ (with a help icon)

Step 3: Fork a sample repository

- 3.1 Log in to your GitHub account, navigate to <https://github.com/jenkins-docs/simple-java-maven-app>, and click on **Fork**

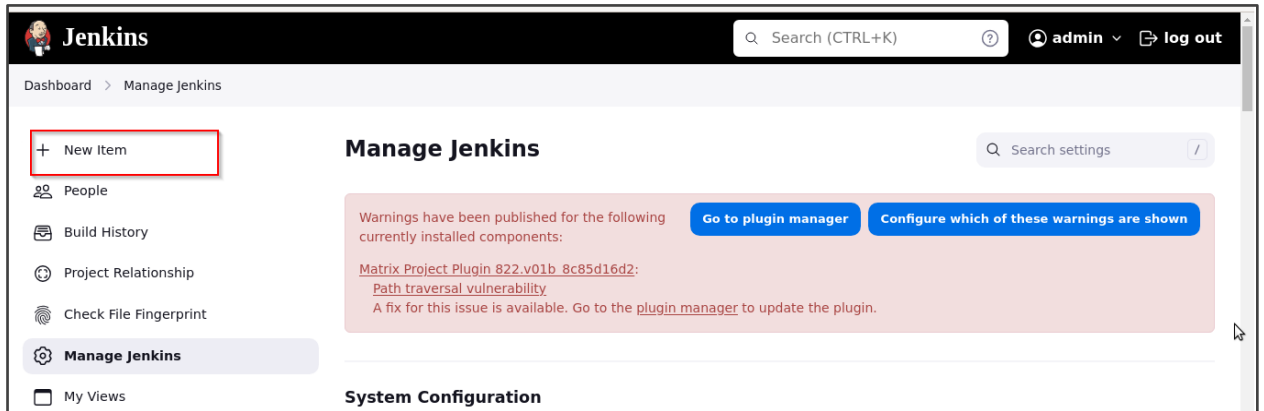


- 3.2 Run **git clone [Forked REPO URL]** in the terminal to clone the repository locally

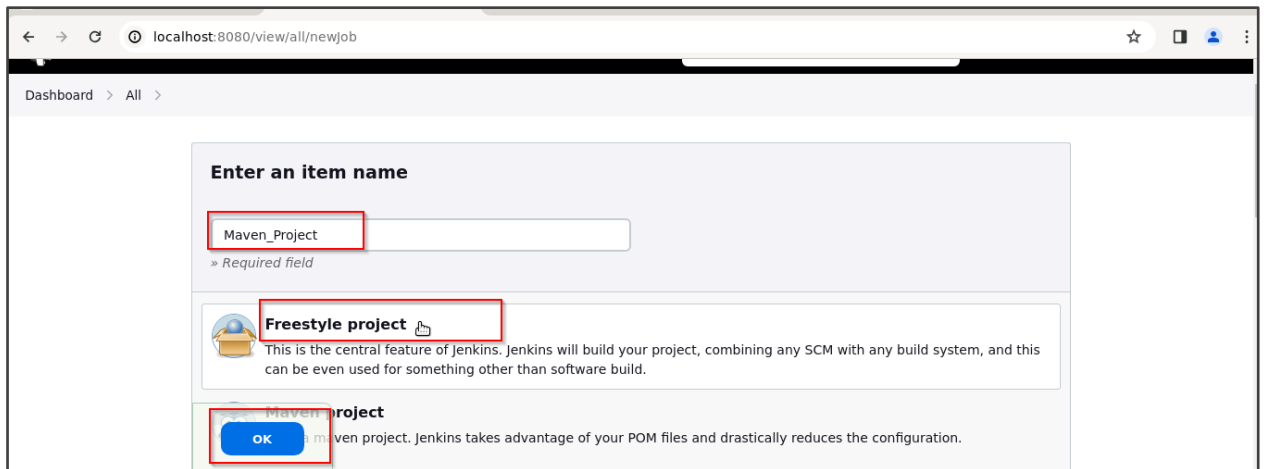
```
labsuser@ip-172-31-39-225:~$ git clone https://github.com/jenkins-docs/simple-java-maven-app.git
Cloning into 'simple-java-maven-app'...
remote: Enumerating objects: 173, done.
remote: Counting objects: 100% (9/9), done.
remote: Compressing objects: 100% (8/8), done.
remote: Total 173 (delta 2), reused 4 (delta 0), pack-reused 164
Receiving objects: 100% (173/173), 33.22 KiB | 3.32 MiB/s, done.
Resolving deltas: 100% (51/51), done.
labsuser@ip-172-31-39-225:~$
```

Step 4: Integrate Maven with Jenkins

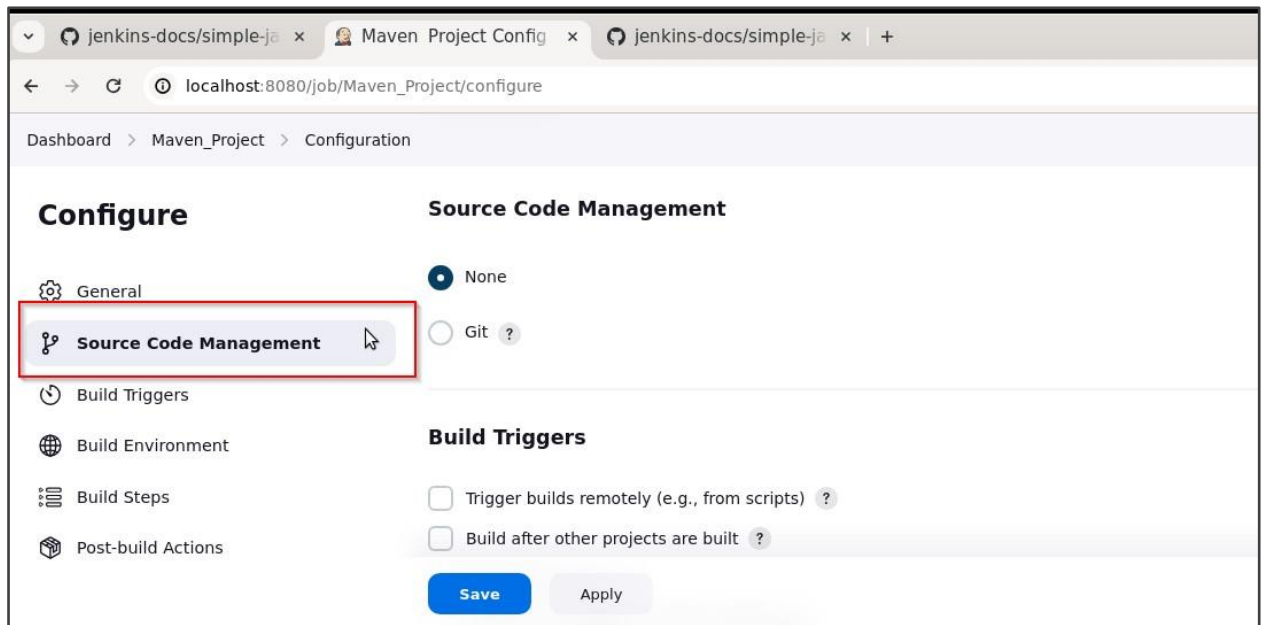
4.1 Click on **New Item** in the Jenkins Dashboard



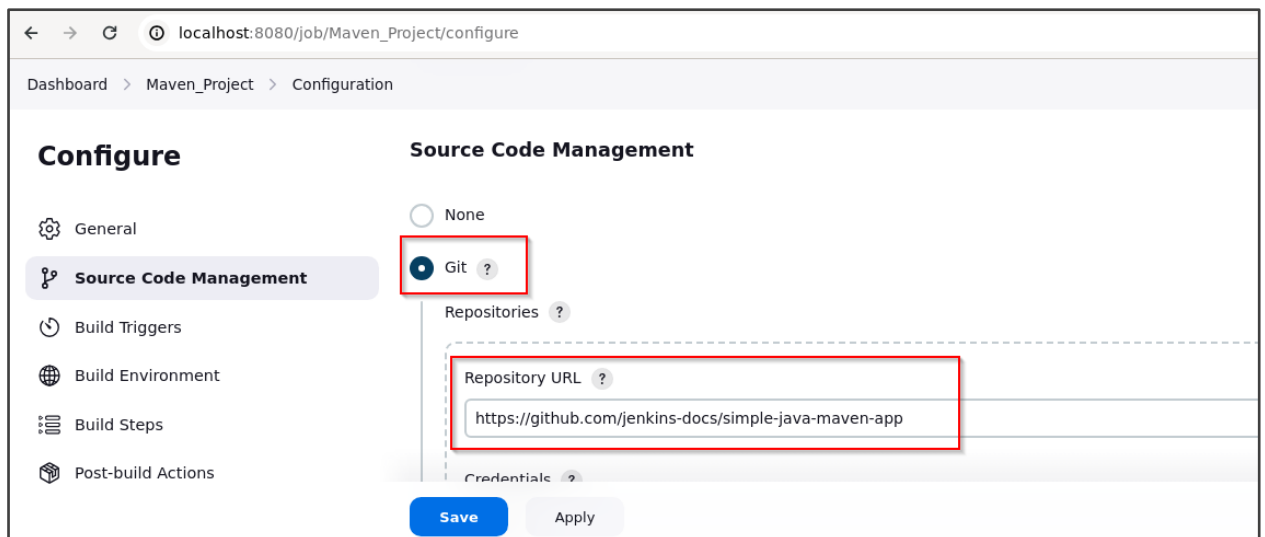
4.2 Enter a name for the project, select **Freestyle project** as the build job type, and click on the **OK** button as shown in the screenshot below:



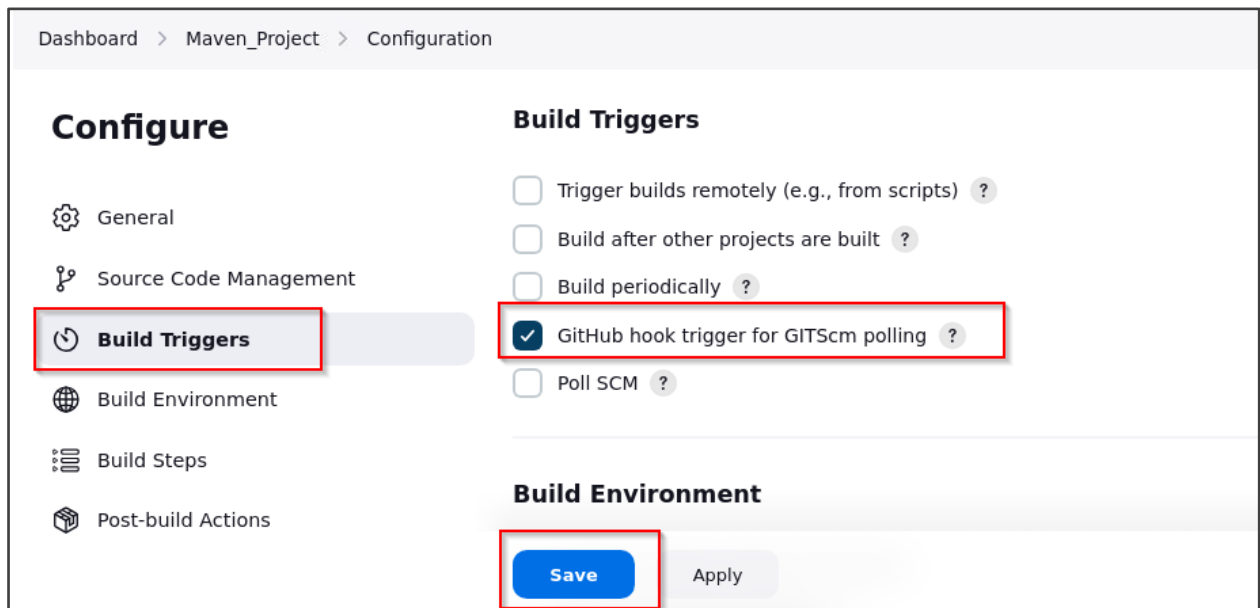
4.3 Click on **Source Code Management**



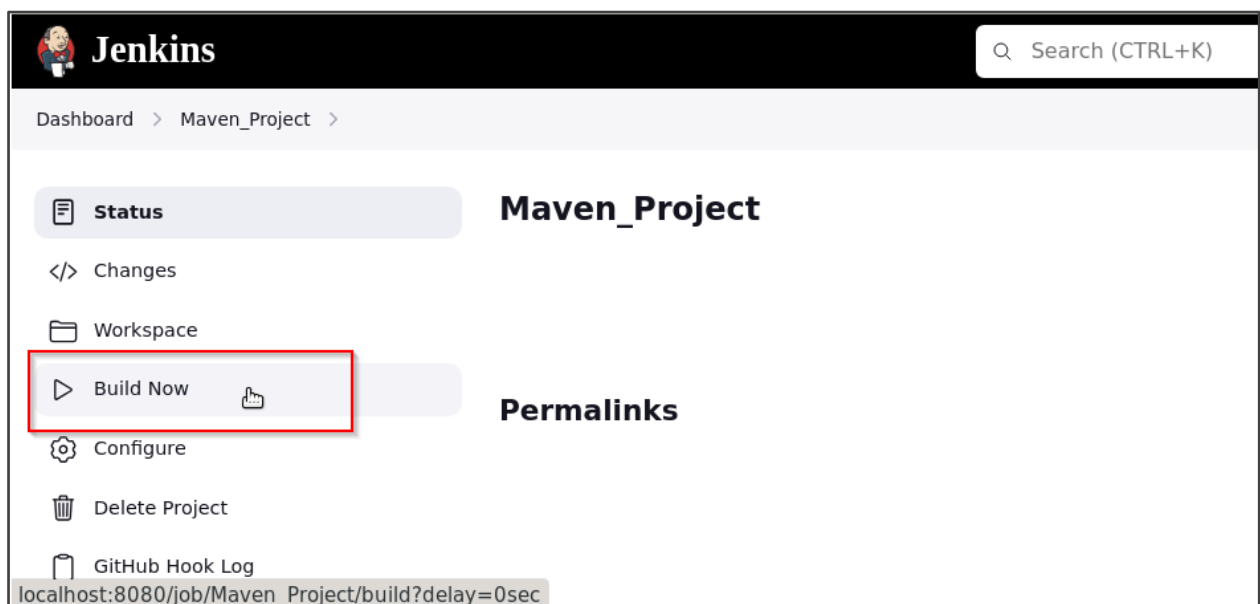
4.4 Select **Git** and enter the **Repository URL**



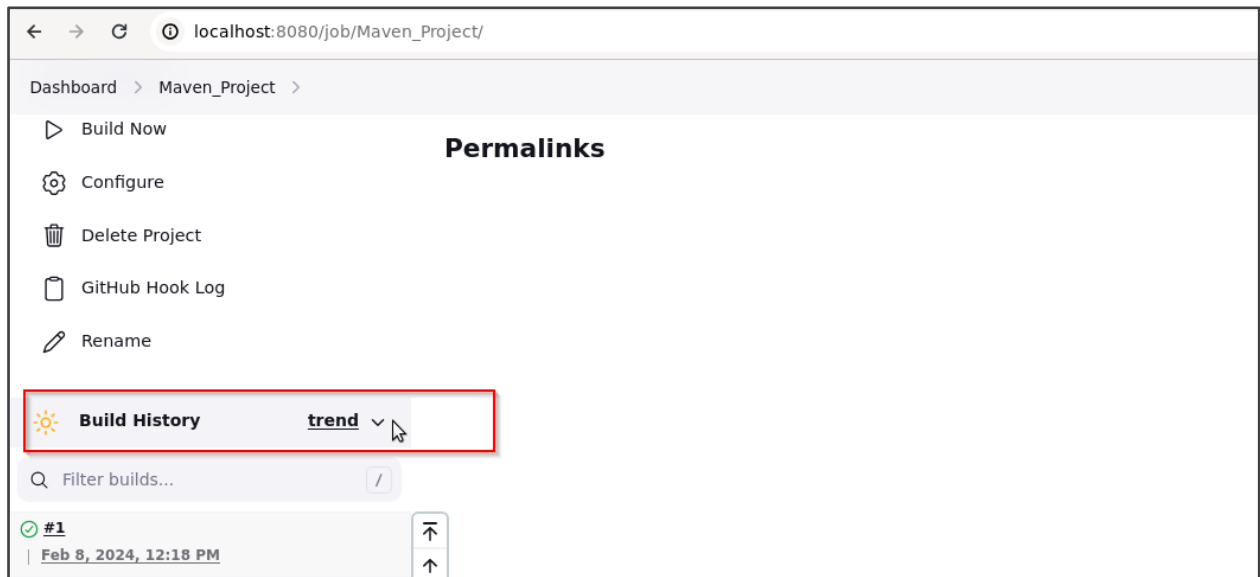
4.5 Click on **Build Triggers**, select the required option as shown in the screenshot below, and then click on **Save**



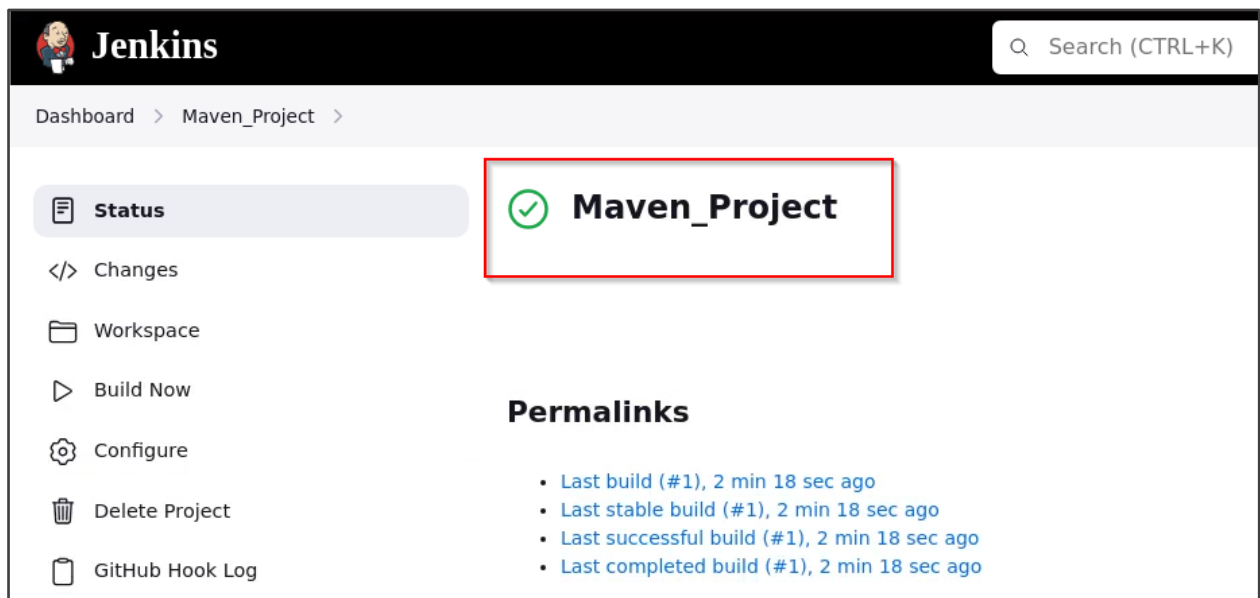
4.6 Click on **Build Now** to view the build results



4.7 Click on **trend** in the **Build History** as shown in the screenshot below:



4.8 Click on **Status** to view the build logs



By following these steps, you have successfully installed the Maven plugin in Jenkins, making it easier to automate Maven-based build tasks within the Jenkins environment for smoother integration and workflow automation.