**Lab Exercise 7**

**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

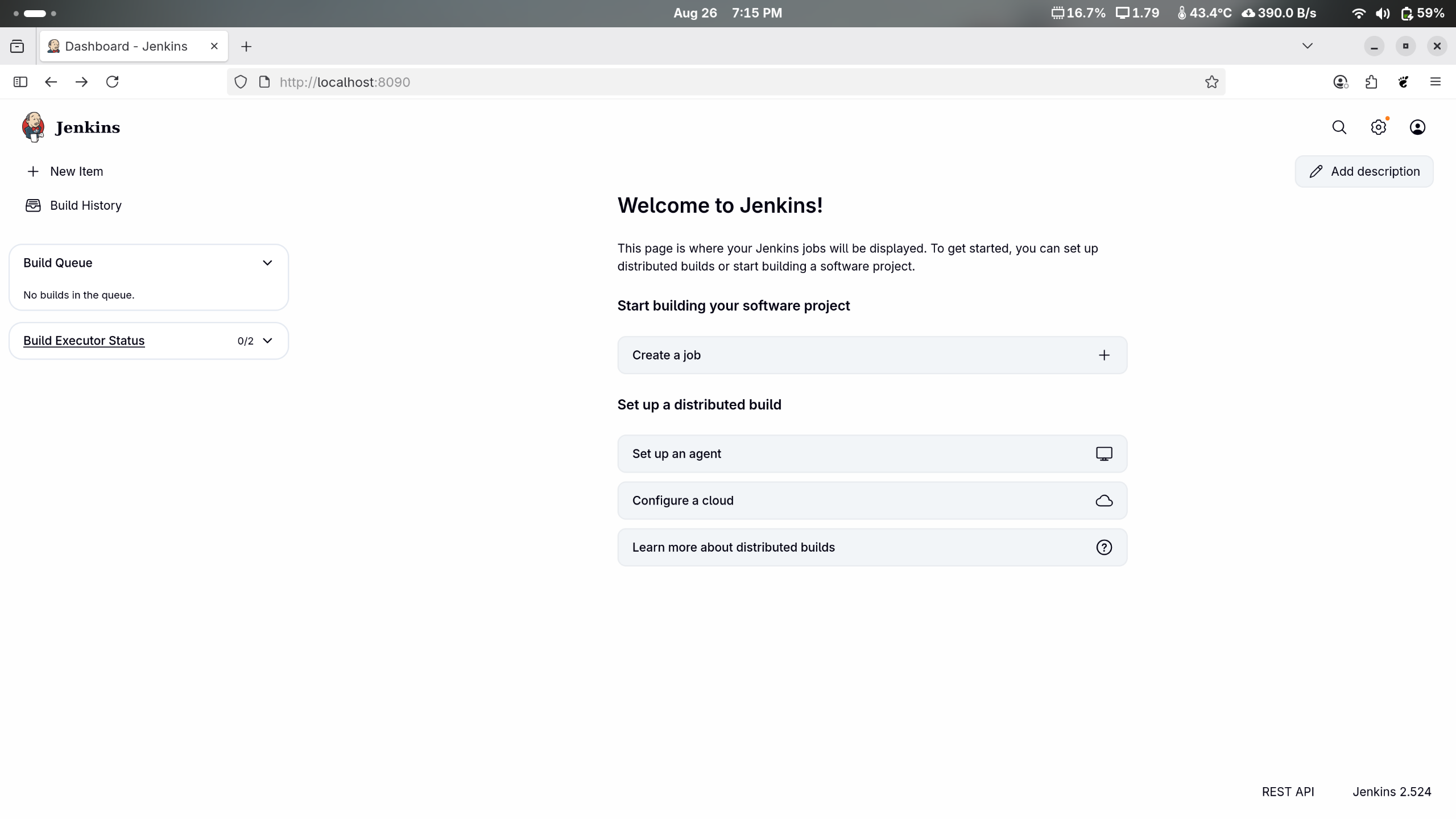
**Step 1:** **Install the Maven plugin**

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

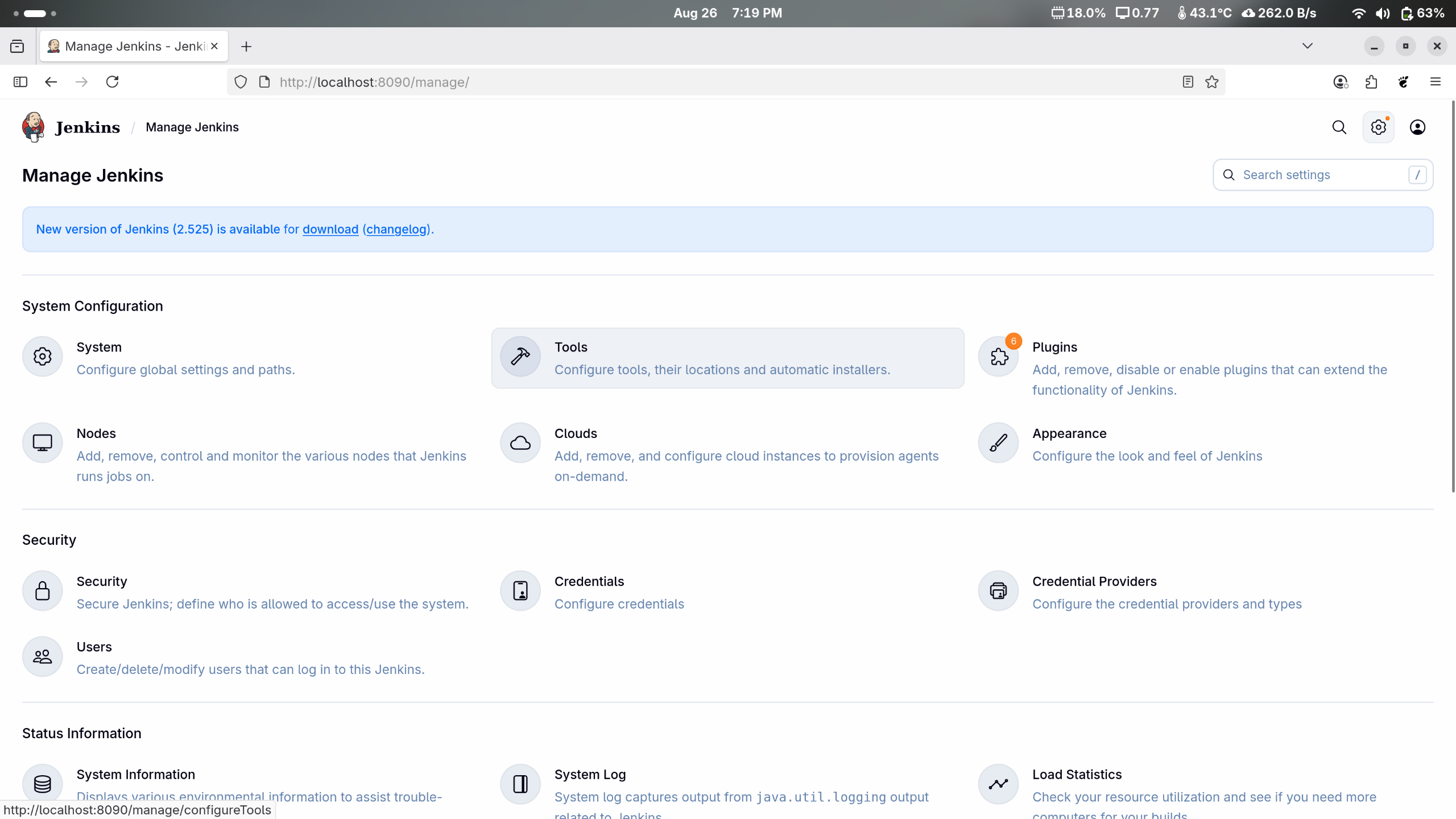
A screenshot of a login page

Description automatically generated

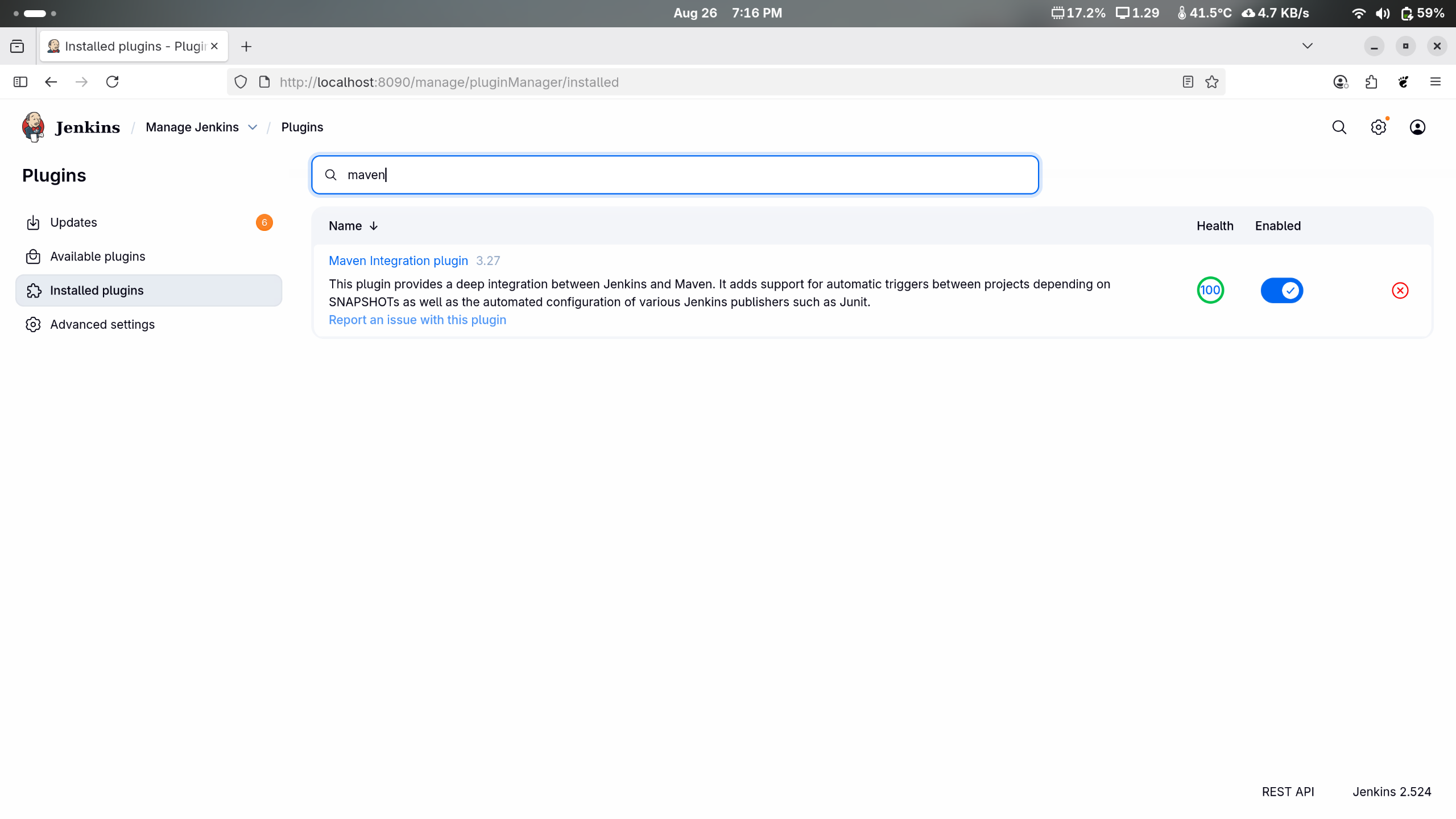
1. Click on the **Manage Jenkins** option as shown in the screenshot below:



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



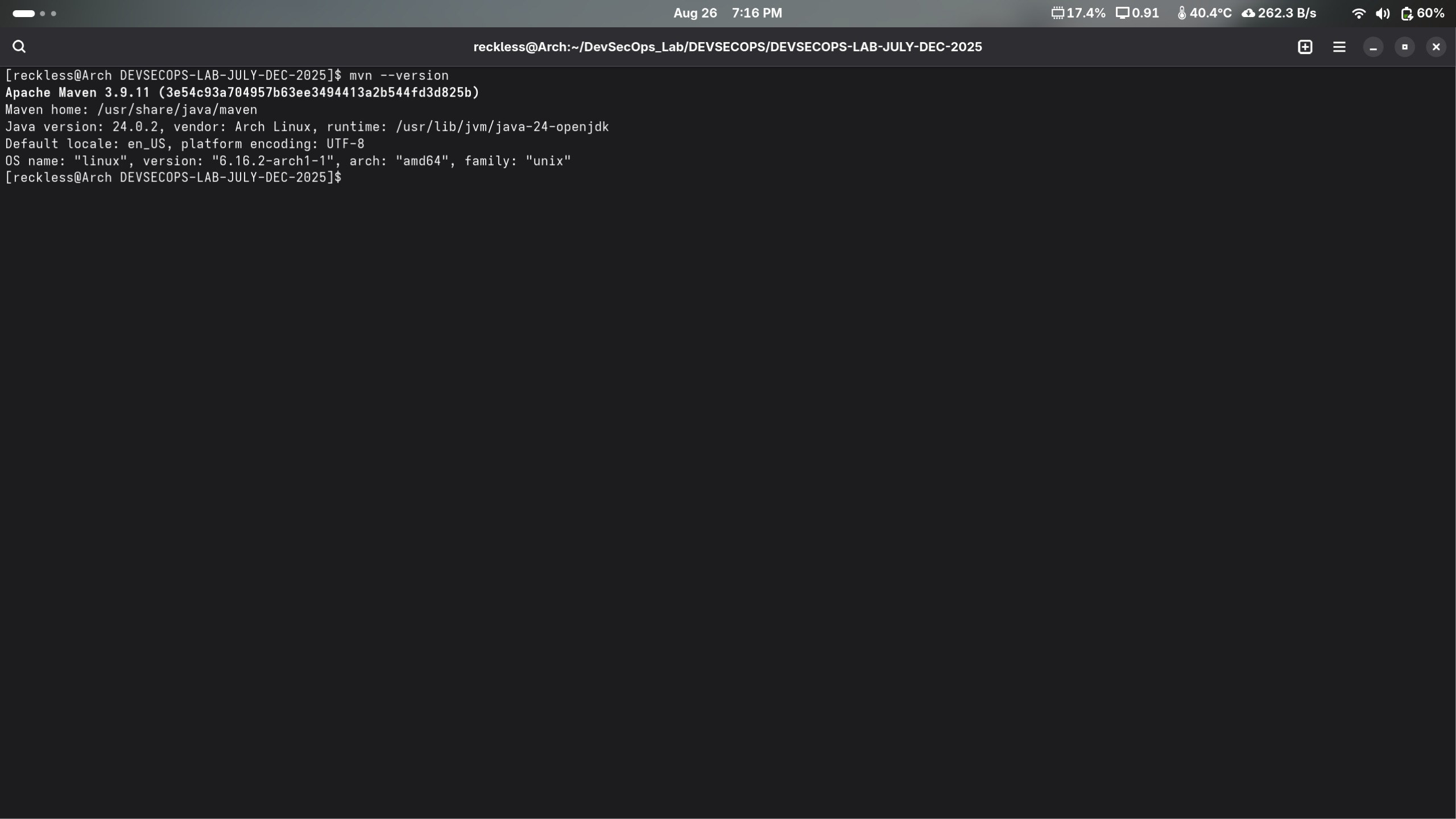
1. Click on **Installed plugins** to verify whether the **Maven Integration plugin** has been installed



|  |
| --- |
| **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. |

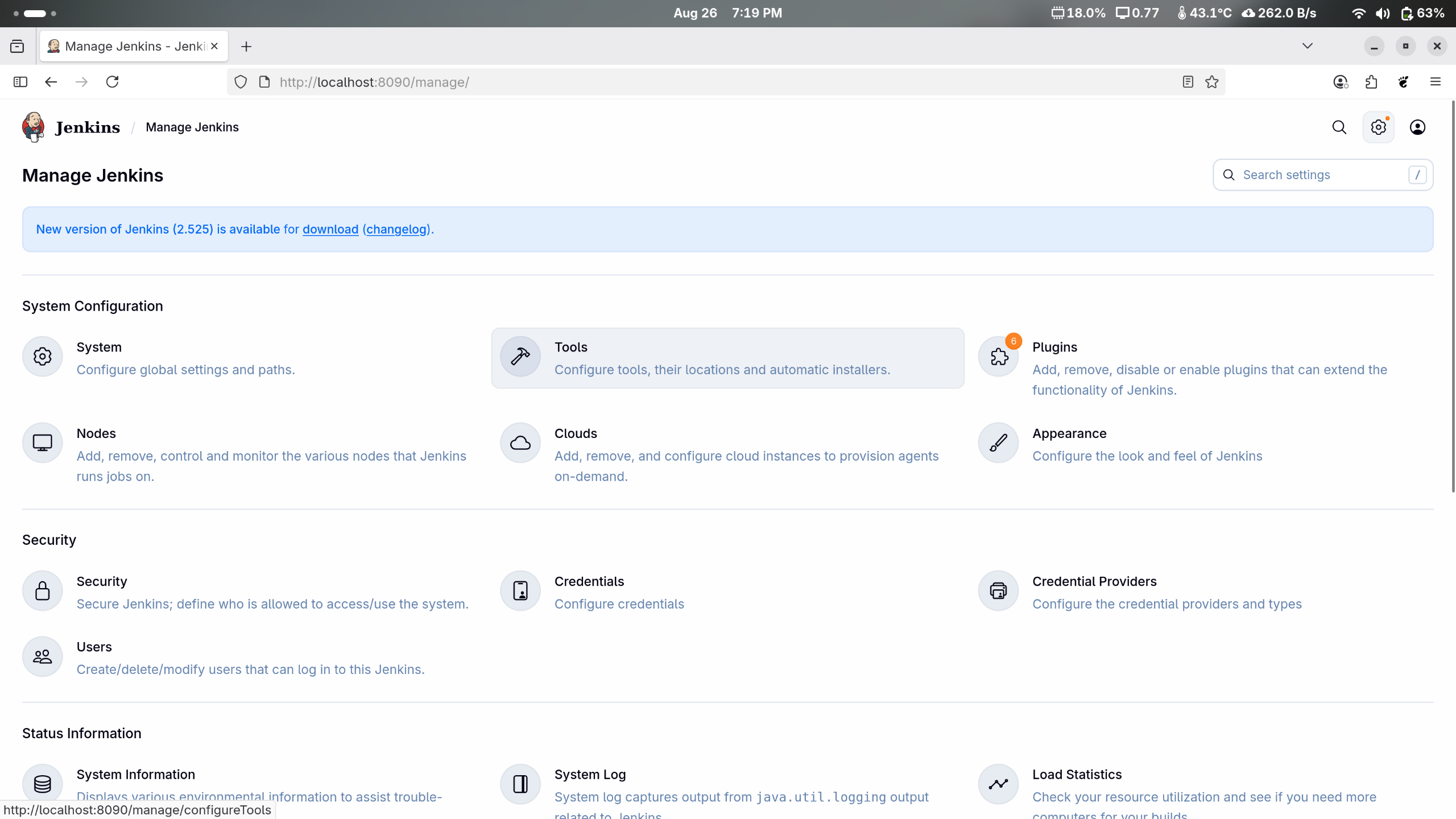
1. Use the following command to check the Maven version:

**mvn -version**

****

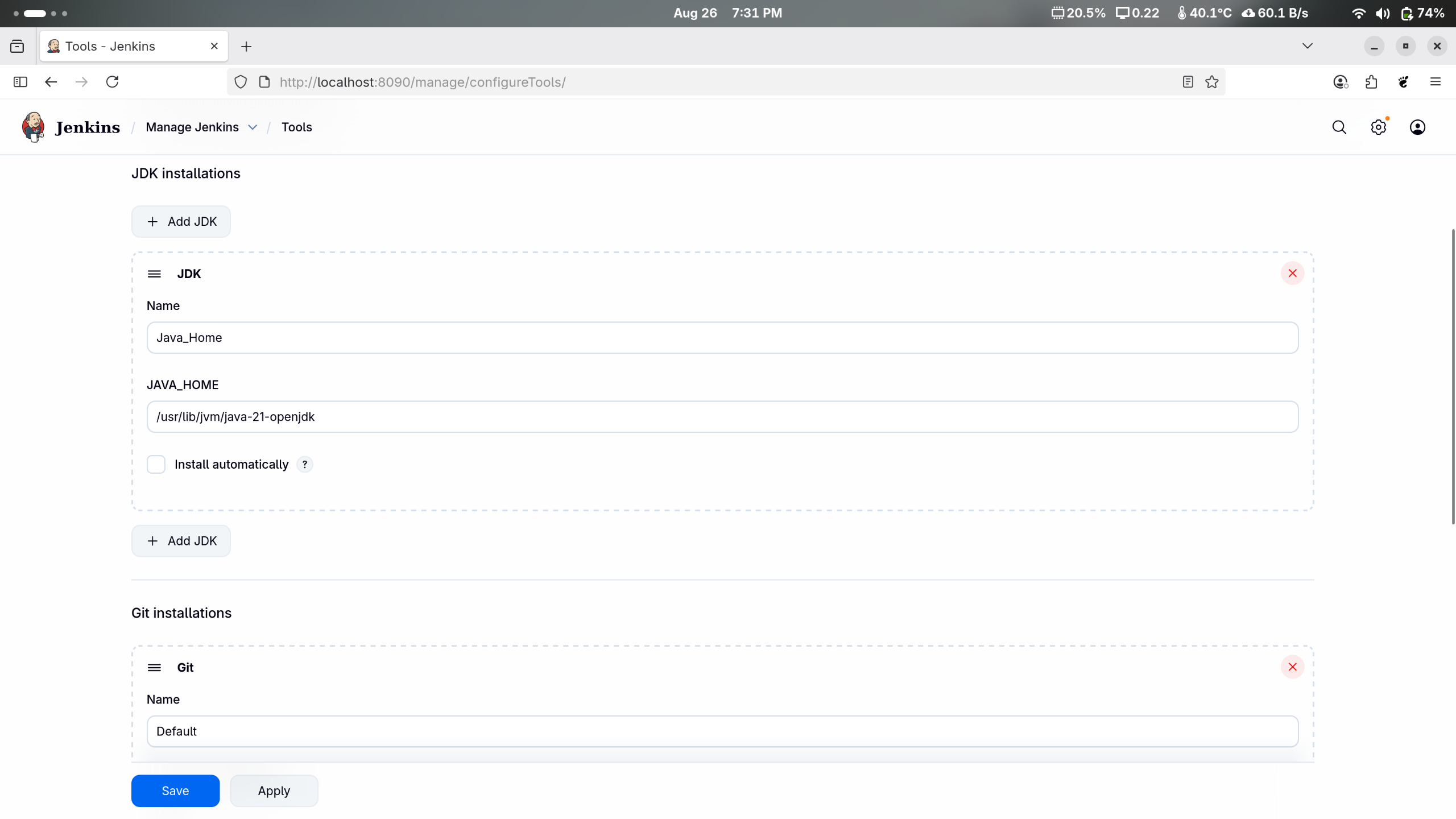
**Step 2: Set up Global Tool Configuration**

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



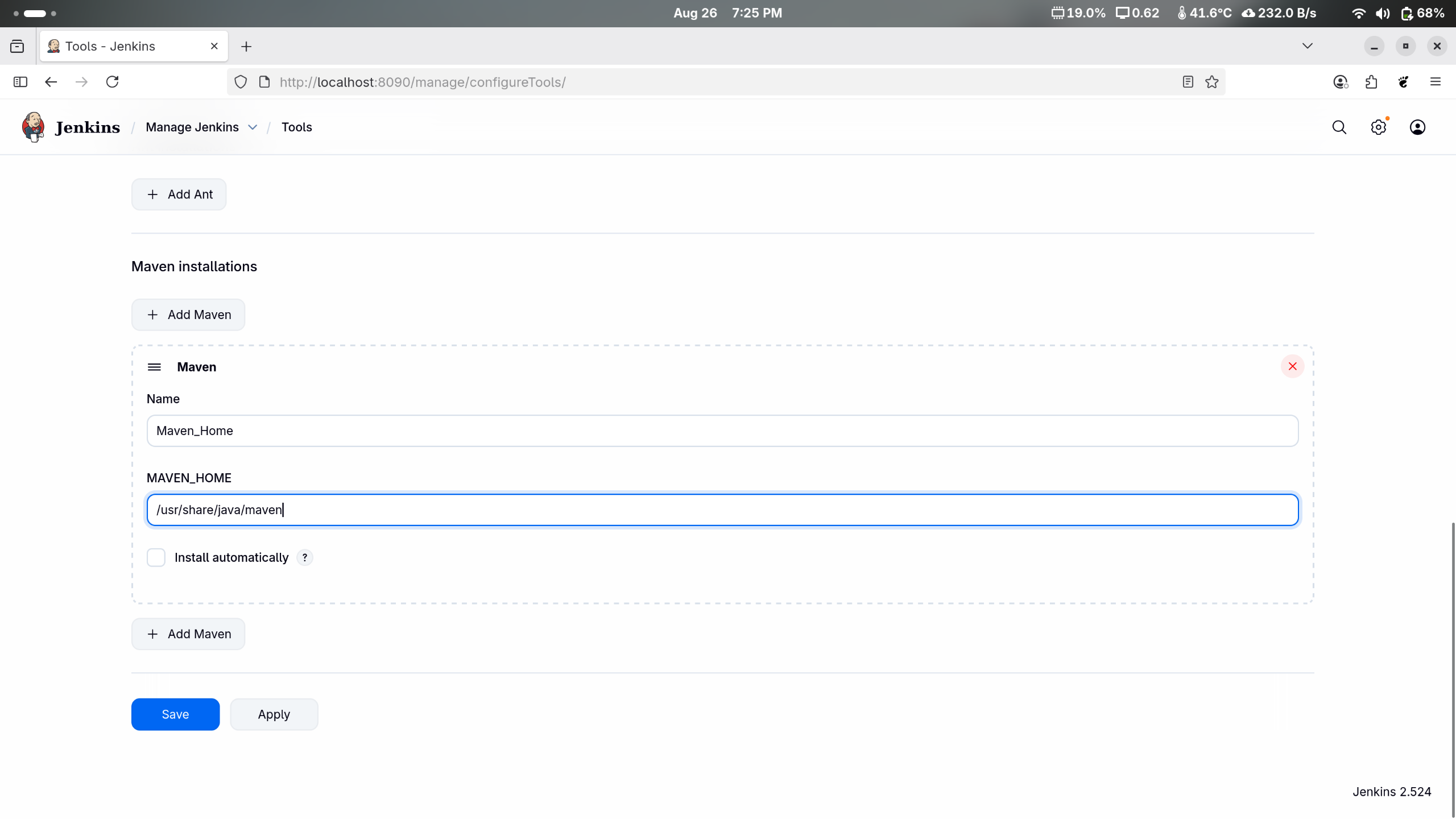
1. Click on **JDK installations** and provide the **Name** and **JAVA\_HOME** path

|  |
| --- |
| **Note:** Set the **JAVA\_HOME** environment variable to **/usr/lib/jvm/java-11-openjdk-amd64** |



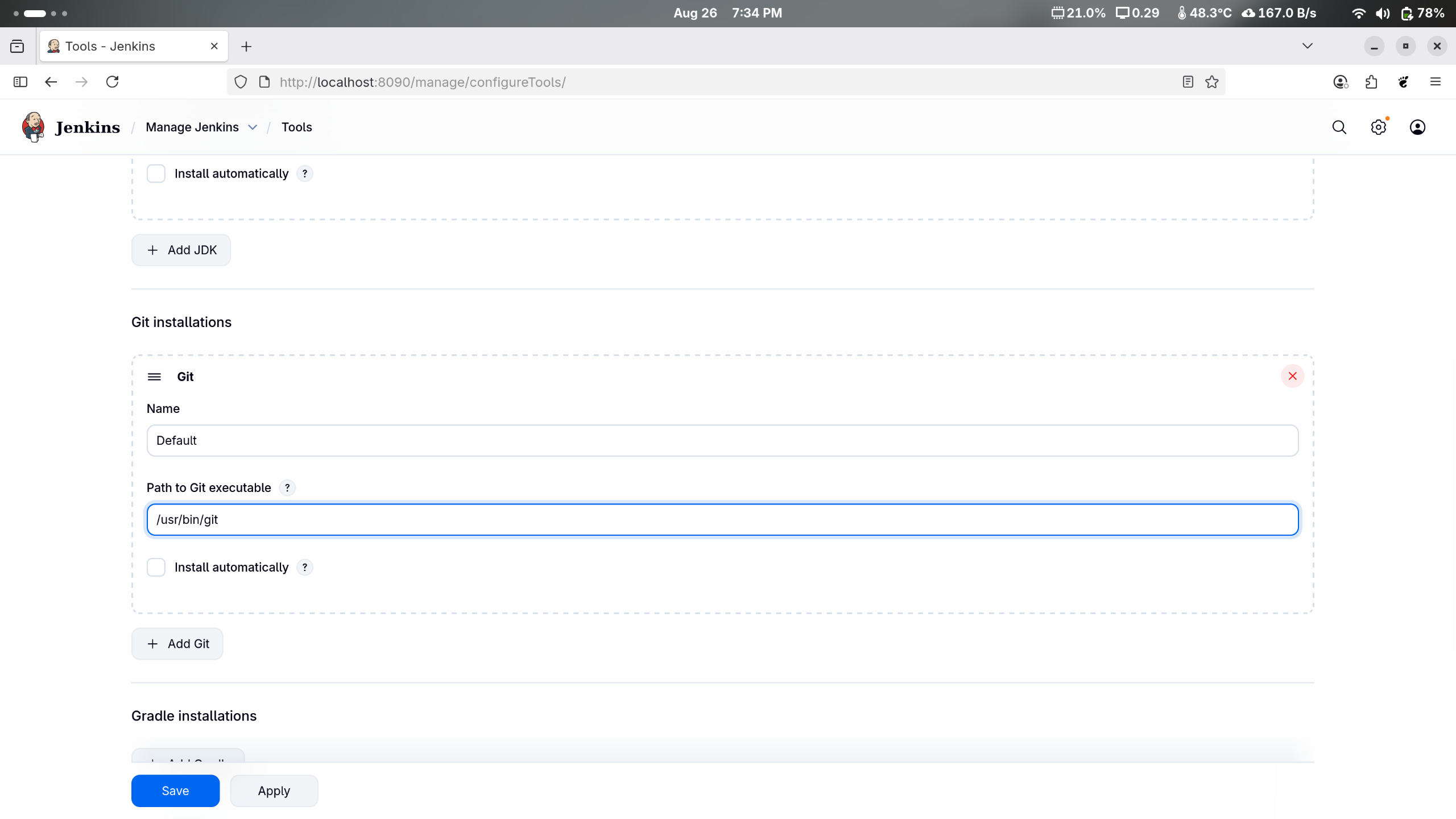
1. To configure Maven, click on the **Maven installations** button in the Maven section and enter a **Name** and **MAVEN\_HOME** path

|  |
| --- |
| **Note**: Set the **MAVEN\_HOME** environment variable to **/usr/share/maven** |



1. 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** |

****

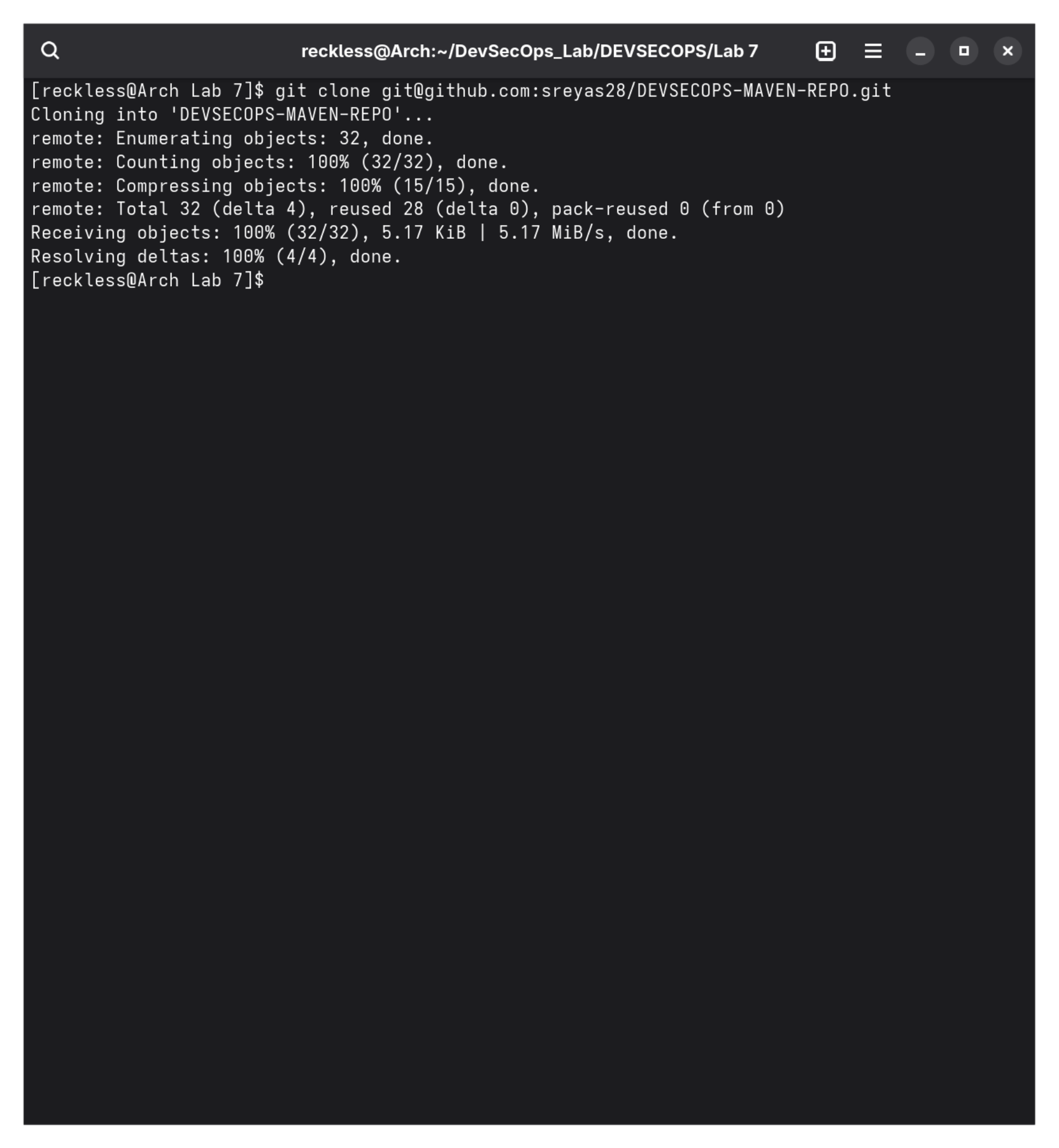
**Step 3: Fork a sample repository**

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

A screenshot of a computer

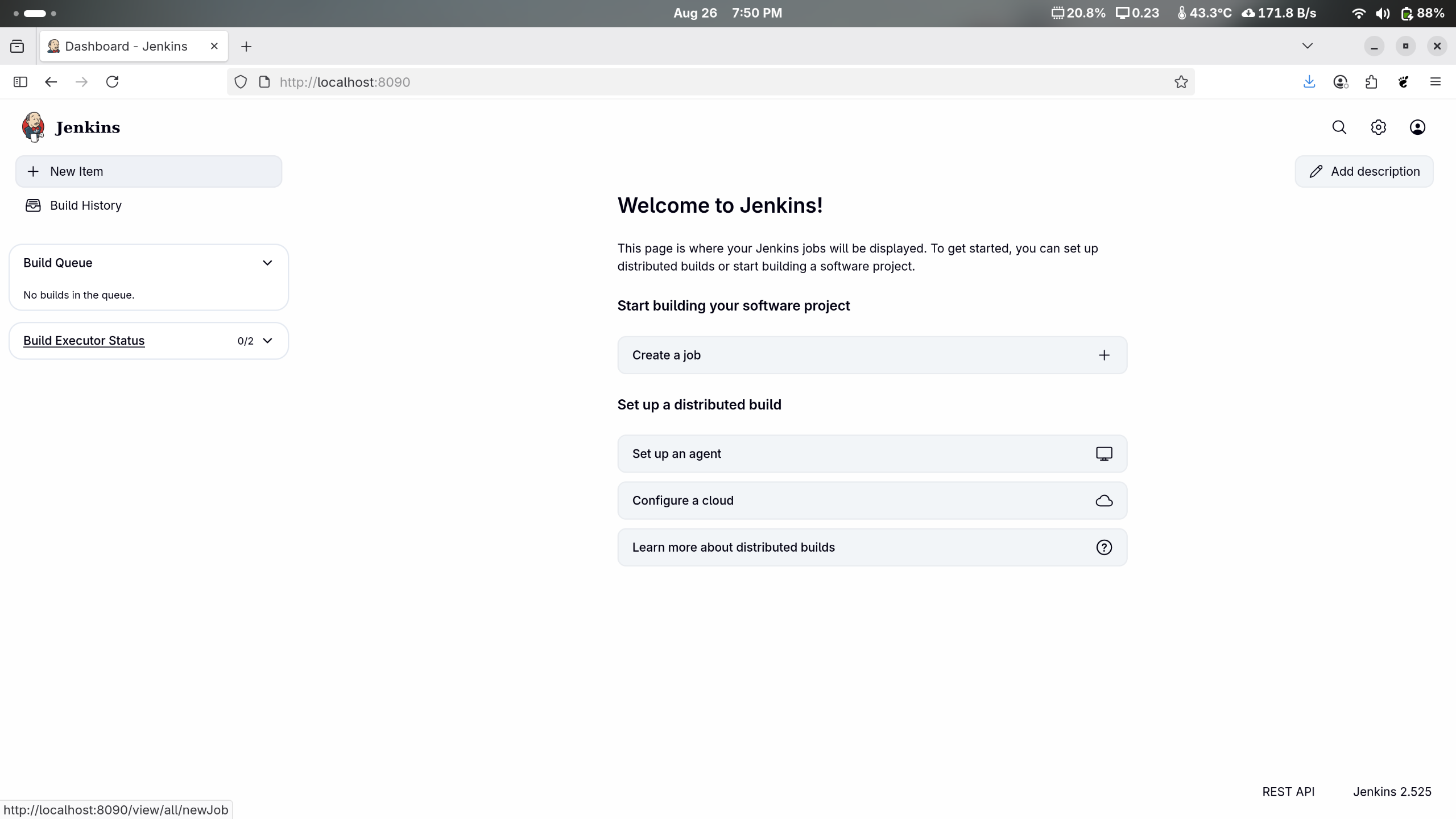
Description automatically generated

1. Run **git clone [Forked REPO URL]** in the terminal to clone the repository locally

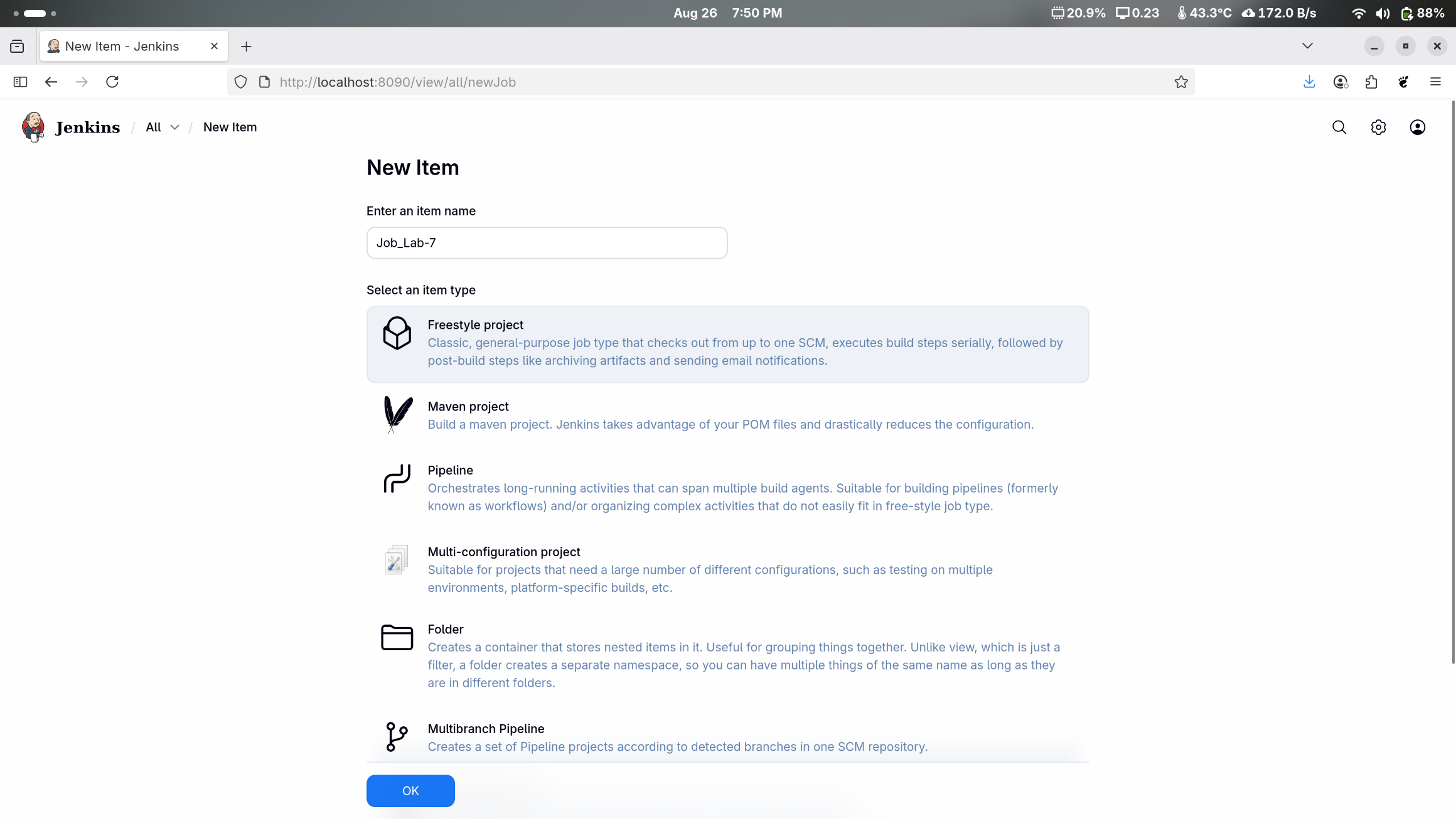


**Step 4: Integrate Maven with Jenkins**

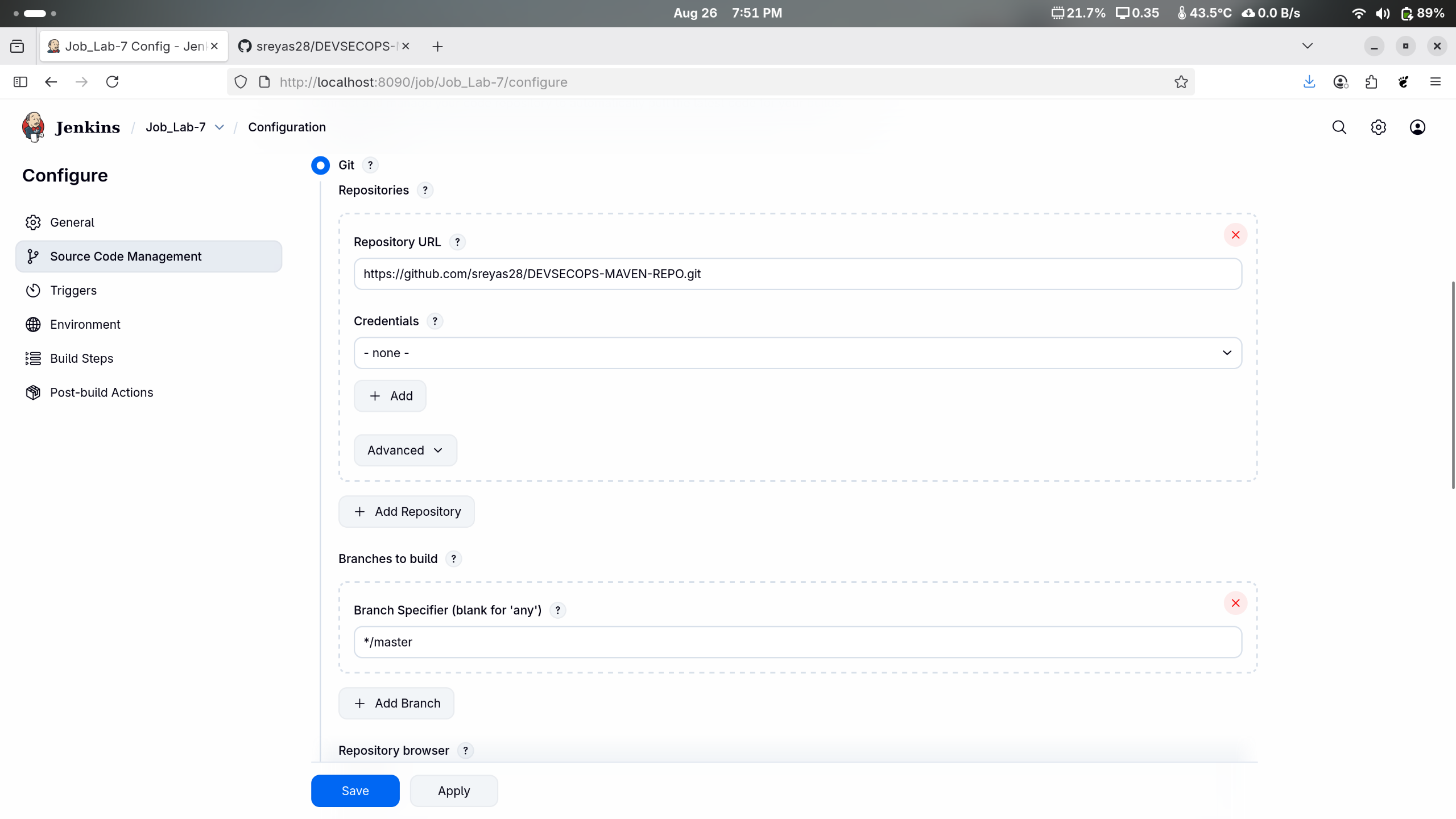
1. Click on **New Item** in the Jenkins Dashboard



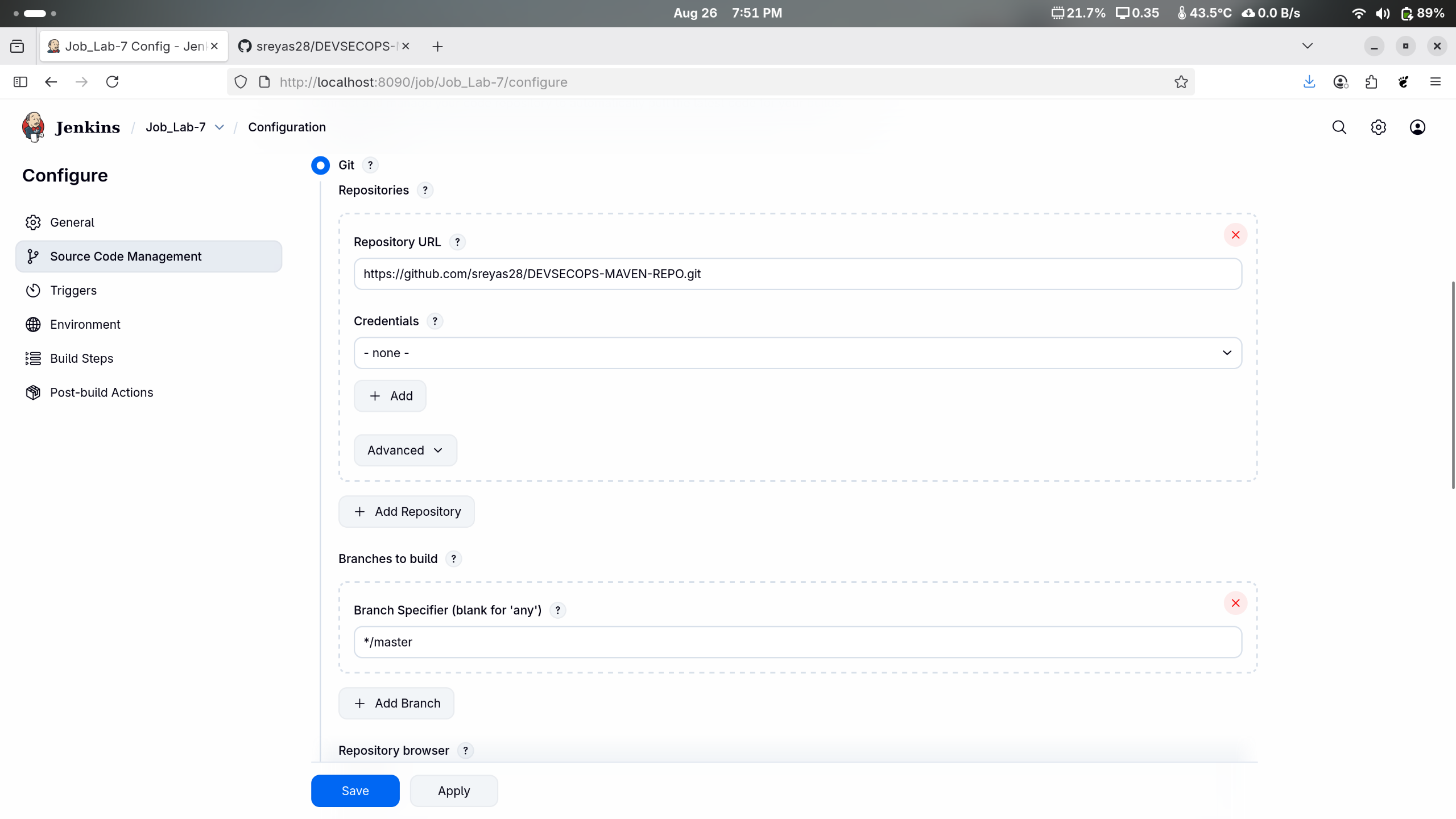
1. Enter a name for the project, select **Freestyle project** as the build job type, and click on the **OK** button as shown in the screenshot below:



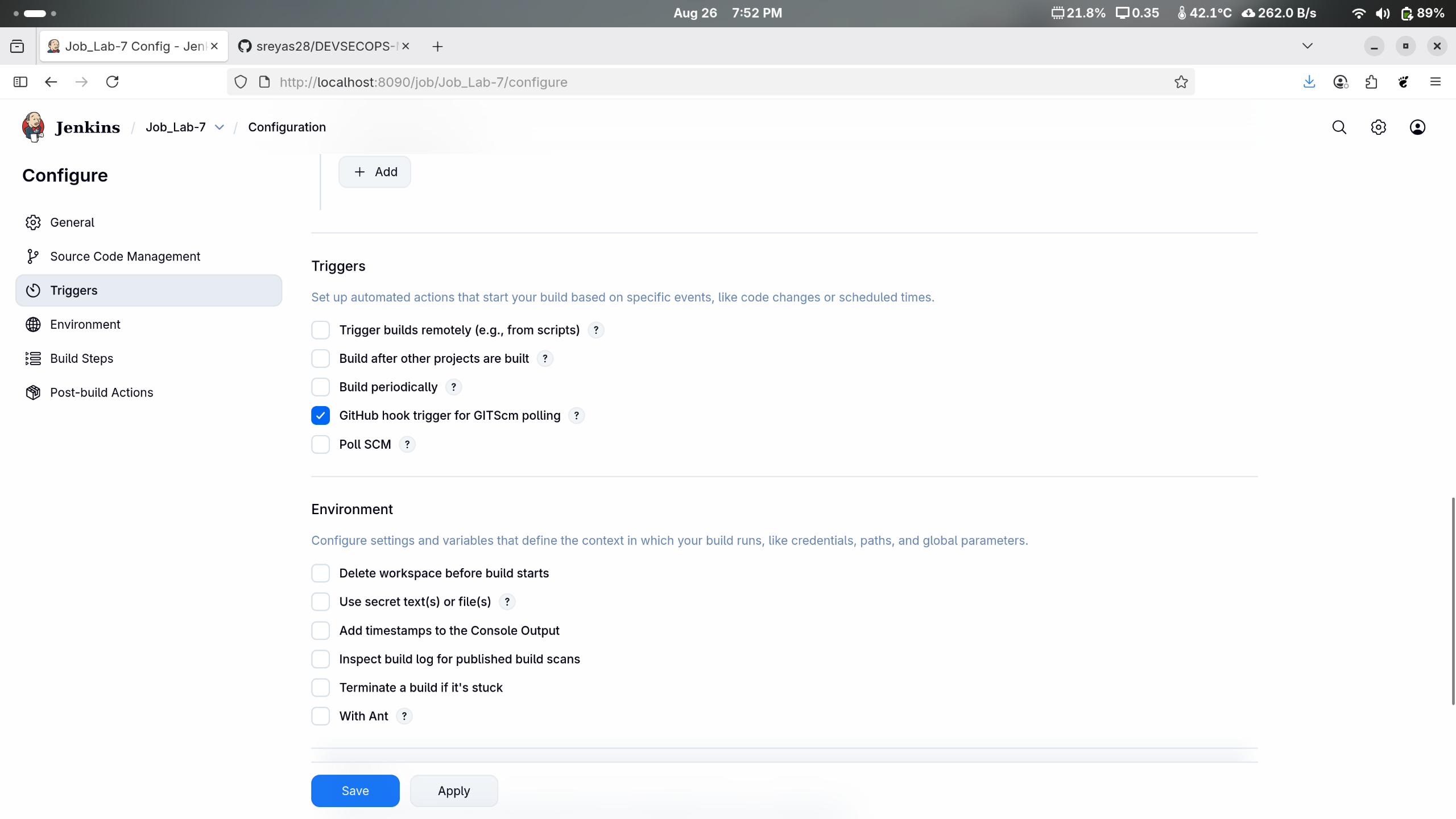
1. Click on **Source Code Management**



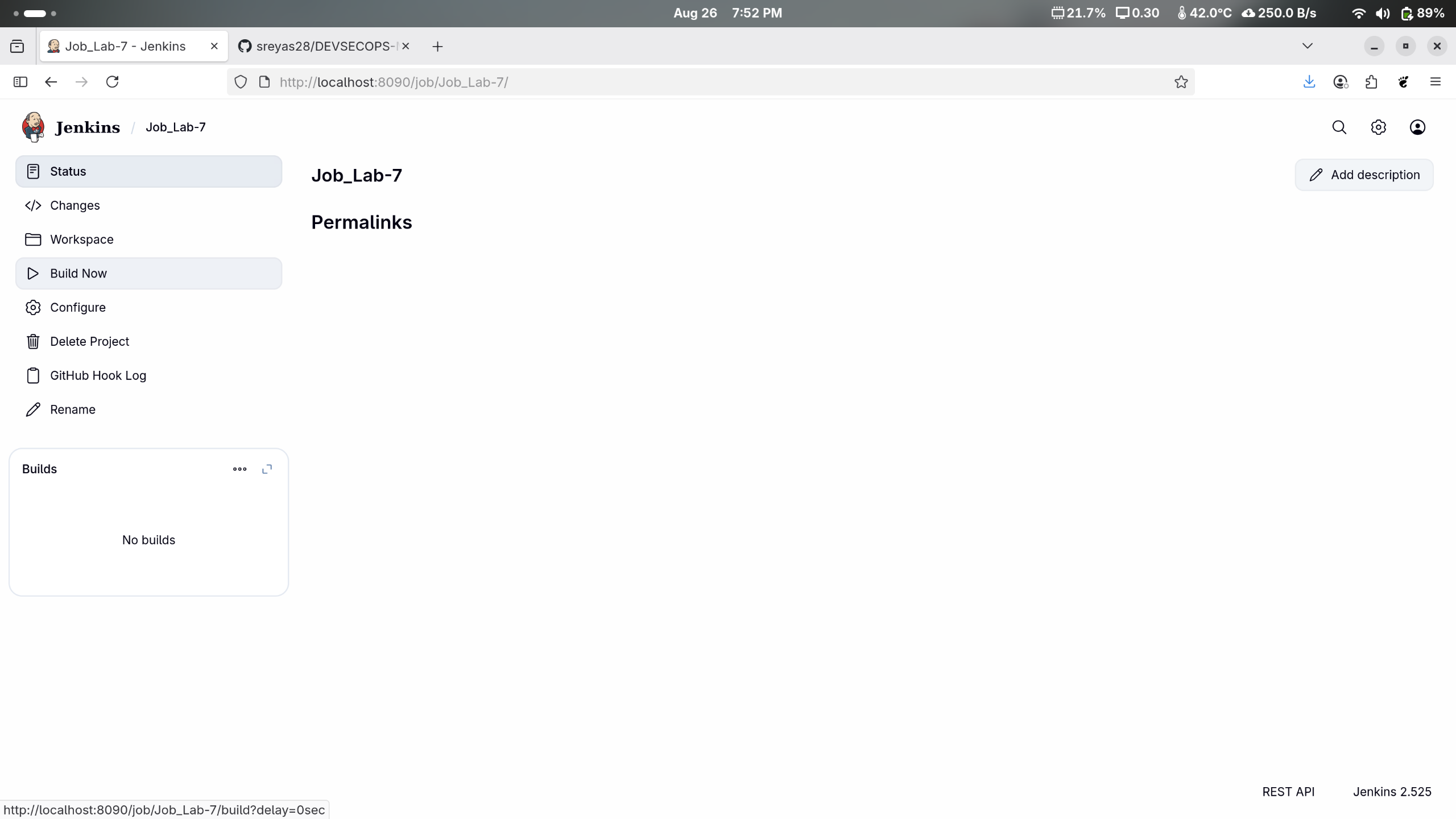
1. Select **Git** and enter the **Repository URL**



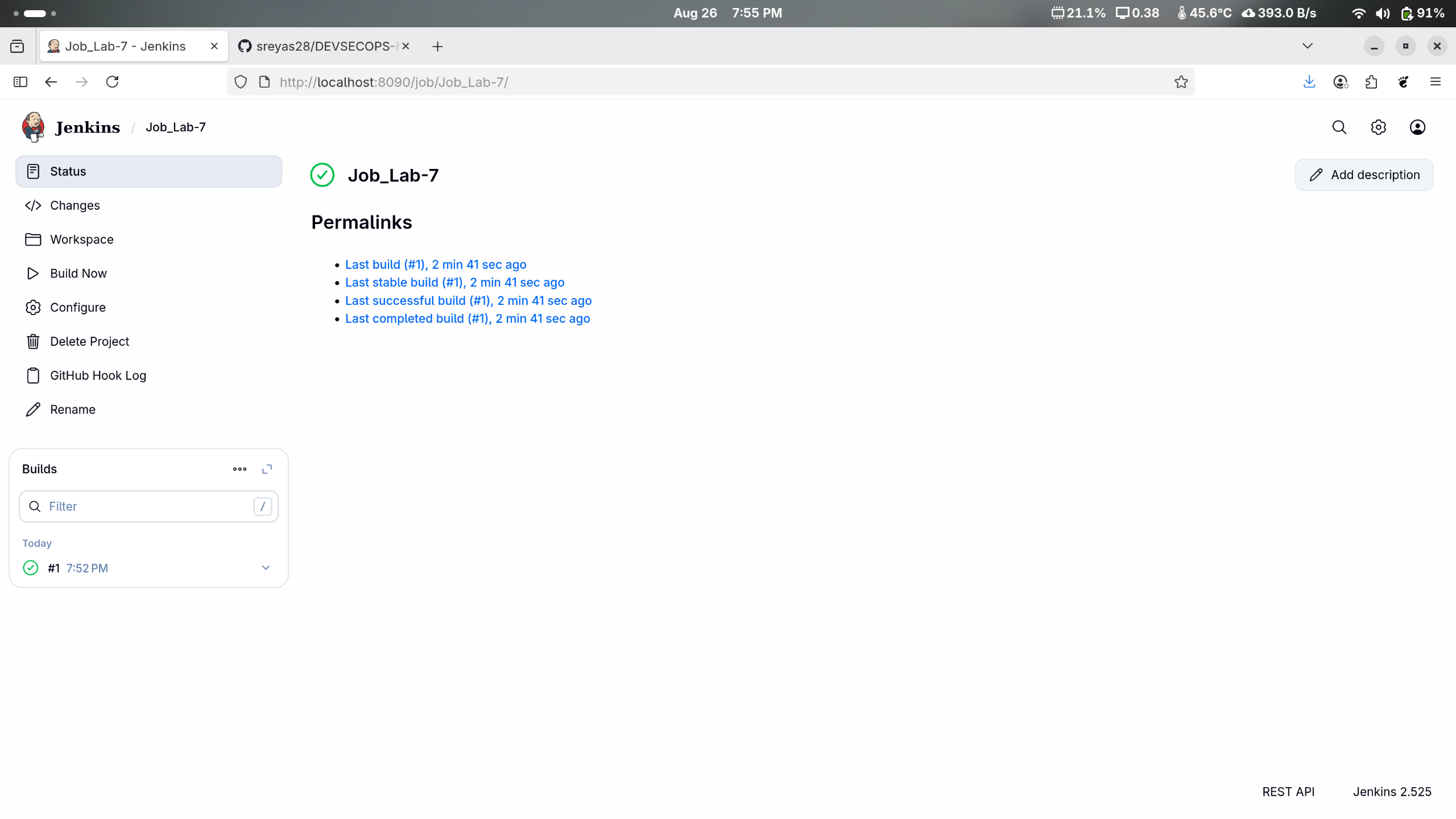
1. Click on **Build Triggers**, select the required option as shown in the screenshot below, and then click on **Save**



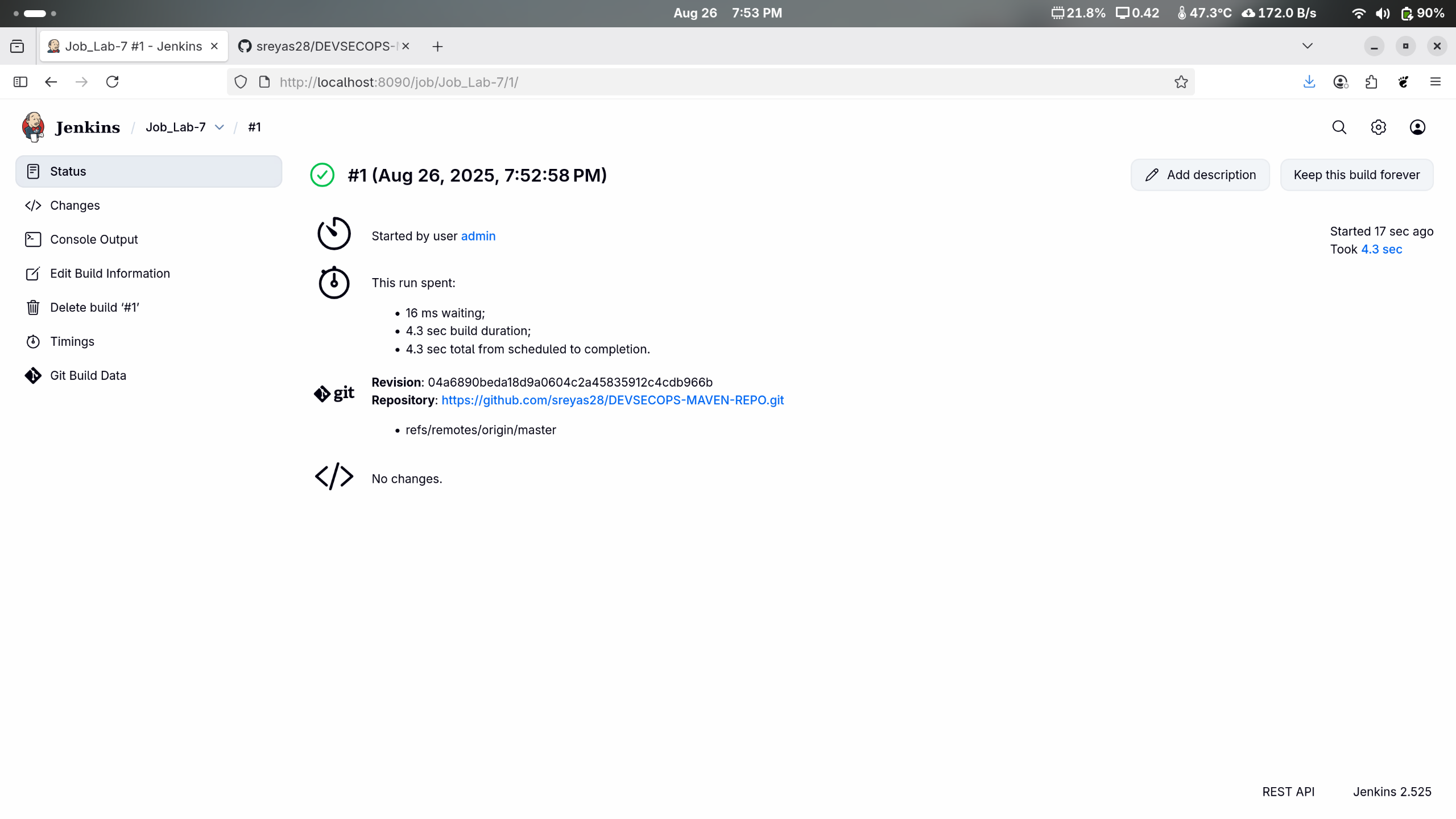
1. Click on **Build Now** to view the build results



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



1. 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.