Lesson 4 Demo 2: Continuous Integration with Jenkins, Git, and Maven

This section will guide you to:

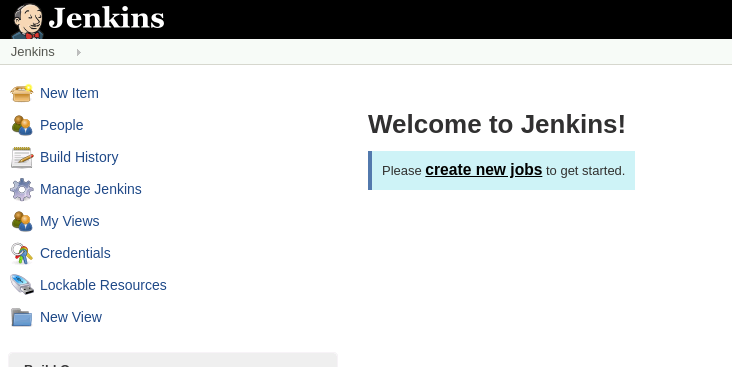
* Create Jenkins job for Maven project
* Poll Git for commits
* Build trigger using Push mechanism

This lab has 5 subsections:

1. Create first Jenkins job
2. Install and configure Maven
3. Configure Jenkins to work with Java, Git, and Maven
4. Create a Jenkins job for your Maven project and run the project
5. Poll Git for commits and automatically trigger the build

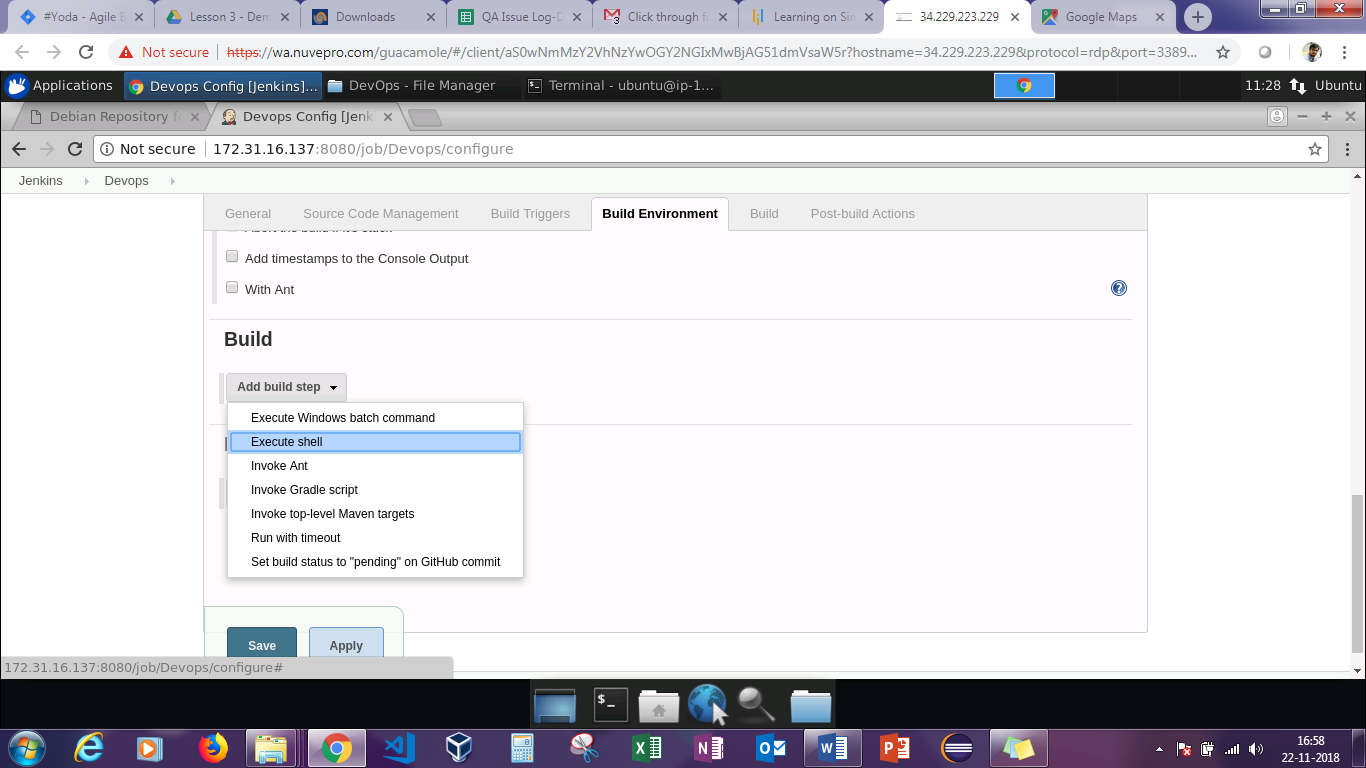
**Step 1 :** Create first Jenkins job

* Log in to Jenkins on [http://localhost:8080](http://x.x.x.x:8080/) using credentials: ***admin/password,*** and click **create new jobs**.

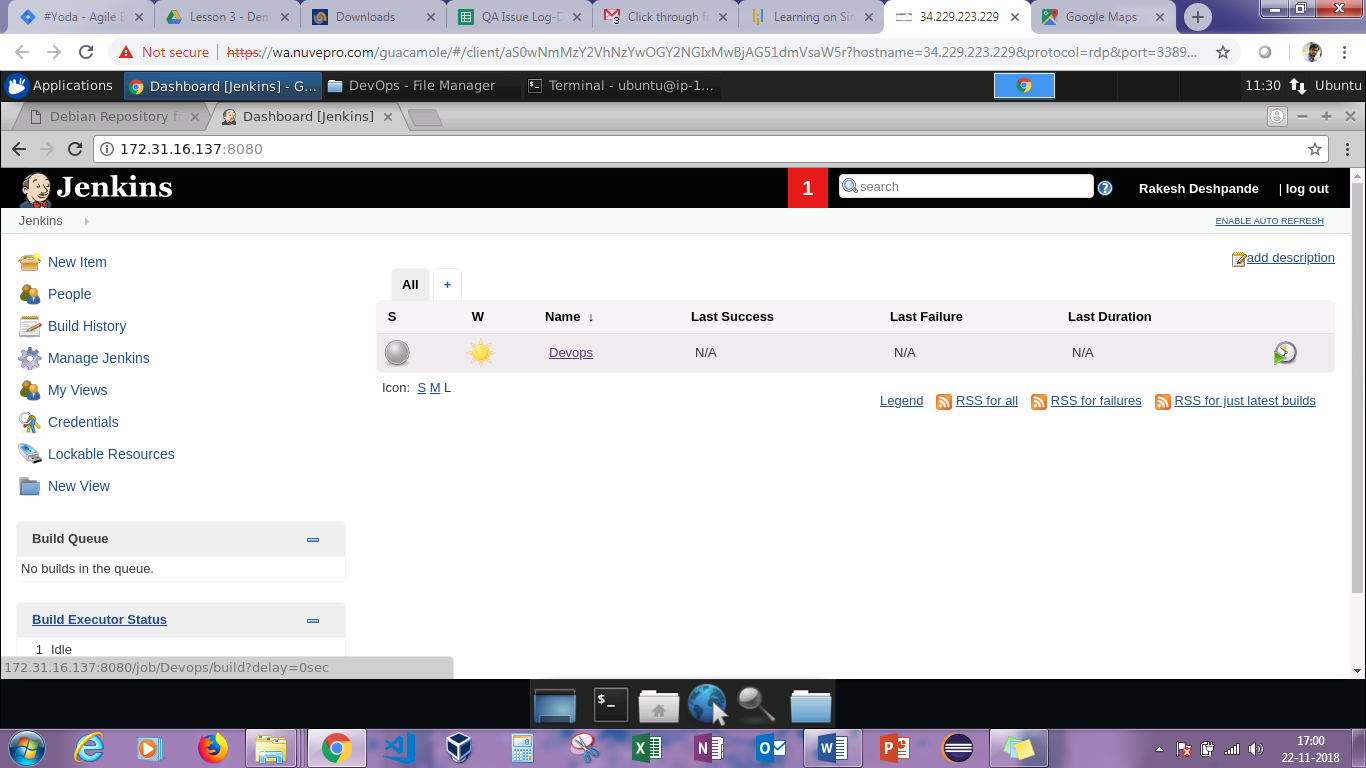


* Provide a name without any space, and select **Freestyle project.**
* Click **OK**.
* Type in the **description** for the job.
* Under **Build** section, click on ***Execute shell*** and type out the command:

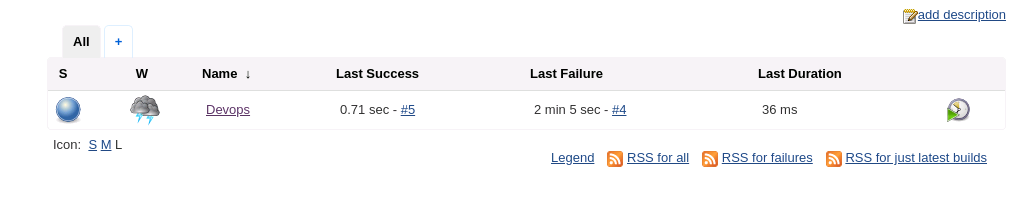
**echo “hello Jenkins”**



* Click **Save** and go back to the landing page to see the new job appear there.
* Click the **Schedule a build** button.



* Click on the job name to look at the **Build history**.
* The blue ball indicates success. Click on the blue ball to look at the **console output** as shown in the screenshot.



* You can see the echo command successfully run.
* Under the **Last Success** column, there’s the link to the build section.
* **W** column indicates sunshine, which means that from the last few commands everything has been successfully executed.

**Step 2:** Install and configure Maven.

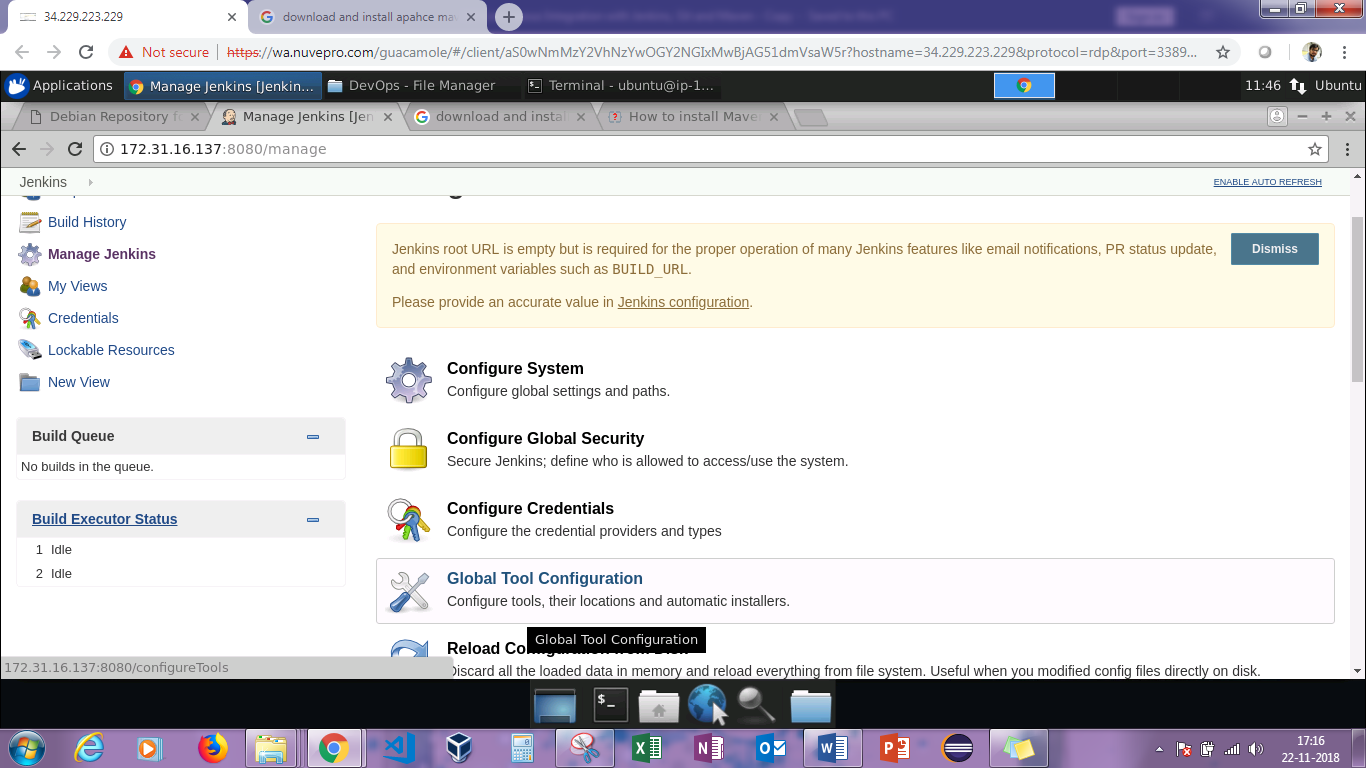
* Run these commands terminal to install Maven:

**sudo apt-get install maven**

**mvn --version**

**Step 3:** Configure Jenkins to work with Java, Git, and Maven

* Go to **Jenkins landing page** ➔ **Manage Jenkins** ➔ **Global Tool Configuration**. This loads Global Jenkins configuration.



* Now, JDK, Git, and Maven configurations need to be added.
* Complete the JDK configuration. Click on the **Add JDK** button. Specify the name as **localJDK**.
* If you already have JDK installed, uncheck the checkbox **Install automatically**. Under **JAVA HOME**, set the **JAVA HOME** path. You can find JAVA\_HOME PATH by using the following command:  
  **export JAVA\_HOME**

**echo $JAVA\_HOME**

* Add the Git settings. Specify the name as **localGit**, and you can find path to Git executable using the following command:

**which git**

* Add the Maven settings. Specify the name as **localMaven**, and find **MAVEN\_HOME** by using the following command:

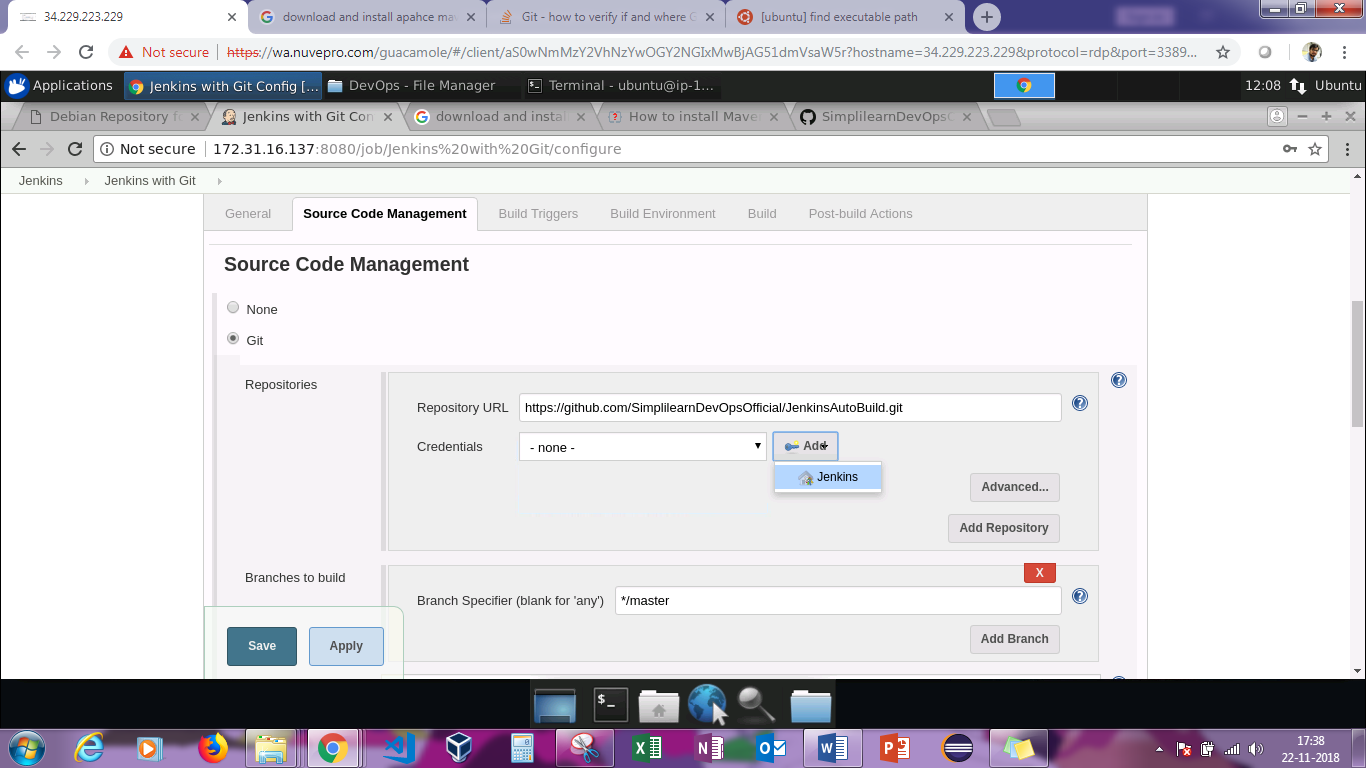
**mvn -v**



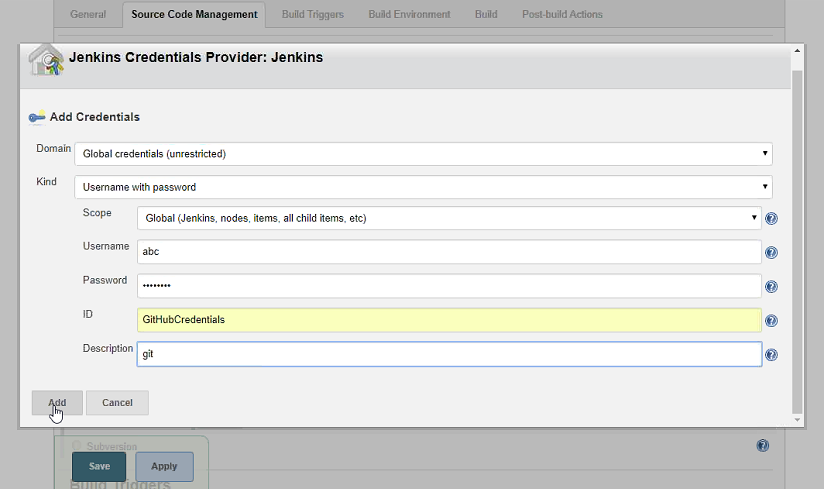
**Step 4:** Create a Jenkins job for your Maven project, and run the project.

**Please Note**: Make sure you have a Github repository. If not, create one.

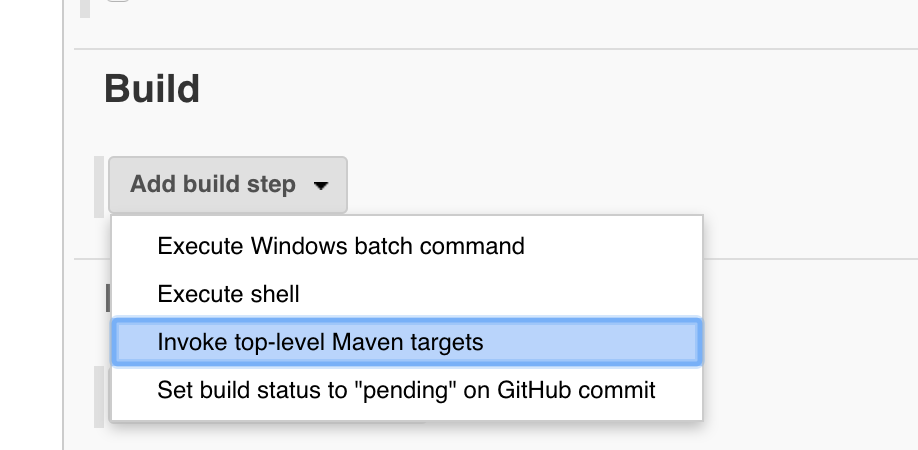
* Create a Jenkins job for your Maven project.
* Go to Jenkins landing page, and click on **New Item**.
* Set the name as **maven-project** of type **Freestyle project**, and click **OK**.
* Set the description as **first maven project**.
* Under source code management, set the details.
* Fill in your repository URL which you can get by clicking on **clone**



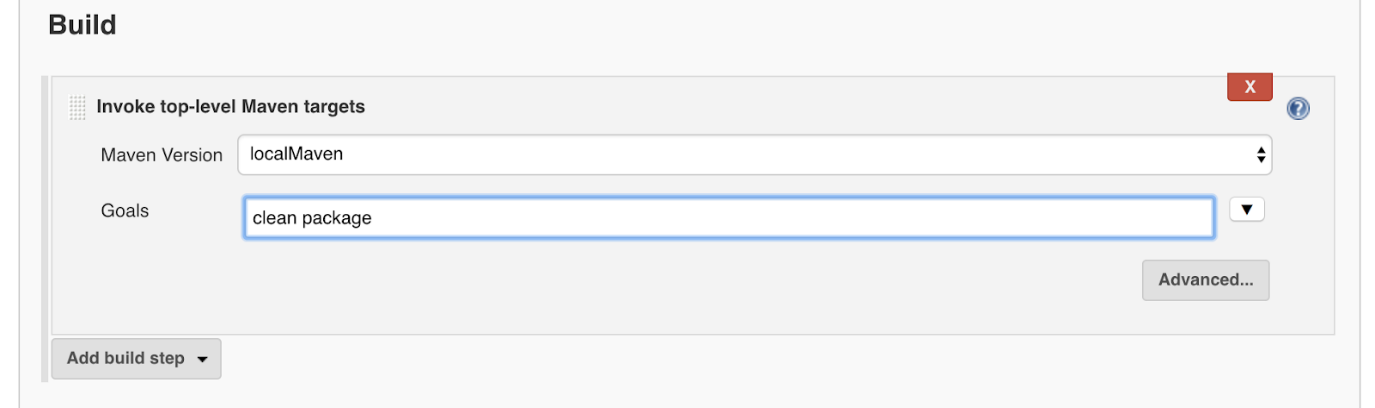
* Under **Credentials**, click on Jenkins and then enter the **username** and **password** of the Git account. Then click on **Add**.
* After adding it, select it from the dropdown box.



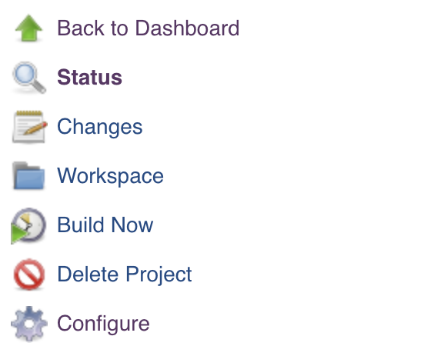
* Under the **Build** section, select **Invoke top level Maven target** in **Add a Build** step.



* Set the **Build** steps. Set the Maven version as **localMaven** and Goals as a clean **package**.



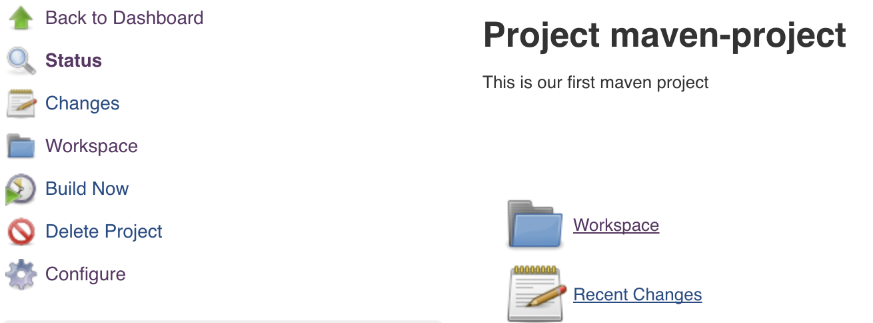
* Click **Save**.
* Run the project.
* Go to the dashboard, select the job, and click on **Build** **Now**.



* Wait till you see the blue ball as a success sign.
* Click on the **Build** icon when it is complete, and observe the ***console output***.

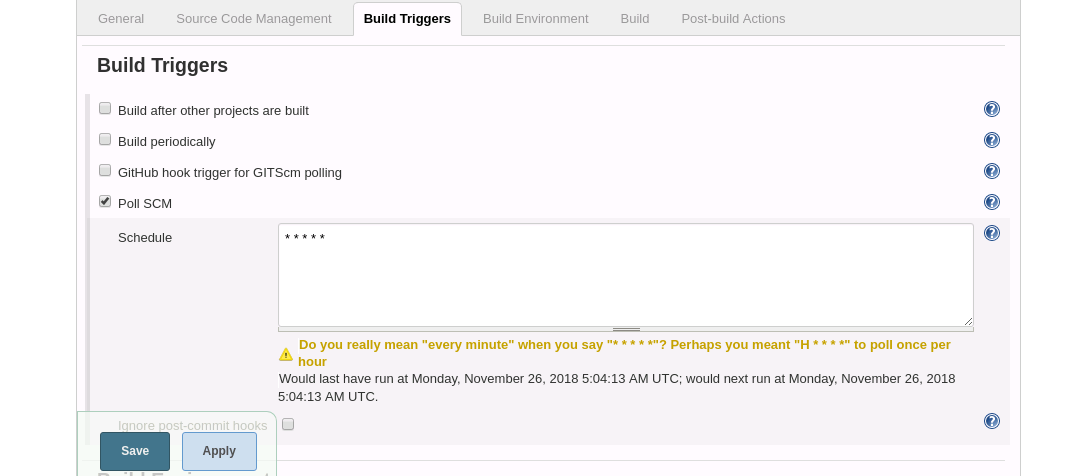
**Step 5**: Poll Git for commits and automatically trigger the build

* Poll Git for commits:
* Go to the project and click ***configure.***

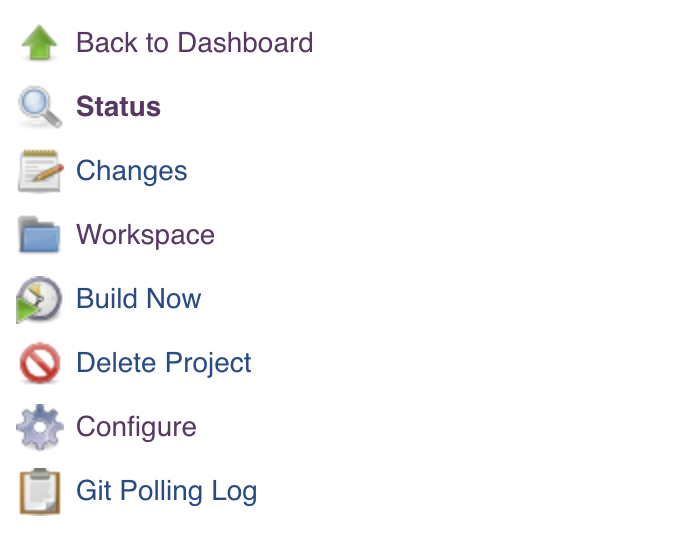


* Scroll to the **Build Trigger** section, and select **Poll SCM**. Set the schedule as **\* \* \* \* \***

**Note**: Make sure there is a space after each \*. Make sure there is no space after the last \*



* Click Save.
* You end up with an extra item in the side menu: **Git Polling Log**.



* Click on ***Git Polling Log*** after a minute to see that Jenkins has already polled Git for more commits.
* Automatically trigger a new **Build**.
* On your system, clone your Maven project from GitHub using the **clone** command.
* If it asks for a password, enter Git password.

**git log**

**vim README.md**

* Make a change, and save the file.

**git add .**

**git commit -m “another commit”**

**git push <Repository\_URL>**

* Now go back to Git to check if there is another commit. Go back to Jenkins (Git Polling Log), and check if the new build has been done.