The Magical Journey of Hello World

"Once upon a time, in the world of software magic, I had a special task at hand. My mission was to craft a simple Java Maven Application that would greet the world with a cheerful 'Hello.' This little application was going to be much more than lines of code – it was going to be a journey.

To start this journey, I needed to fetch the application's secret recipe from a magical place called GitHub. But there was a catch – my trusty sidekick, the Slave OS, needed a special tool called 'git' to fetch the recipe properly. So, I waved my wand and installed the 'git' software onto the Slave OS.

As I continued on my quest, I discovered that the heart of this adventure was Maven, a powerful wizard that could transform my code into a magical package. But Maven needed its own set of ingredients, which it found in a special scroll called 'pom.xml.' This scroll held the secrets to gathering all the necessary libraries for my application.

But my journey wasn't just about creating the package; I also needed to ensure that my creation was flawless. To do this, I wrote a special script – a sort of magical spell – to test the powers of my application. If the test spell worked its magic without a hitch, I knew my creation was ready to face the world.

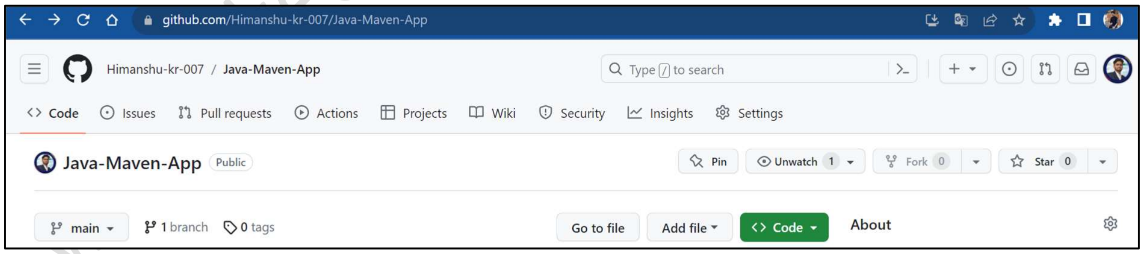
With a victorious smile, I finally reached the grand finale of my journey. Armed with a successful test, I used Maven to create a powerful enchanted package. This package contained all the spells and charms my application needed to work its magic on any other machine.

And that's not all – my package was versatile. It could be summoned on any machine that understood the language of Java. With a wave of my wand, the 'java' command brought my creation to life, spreading its 'Hello' message far and wide.

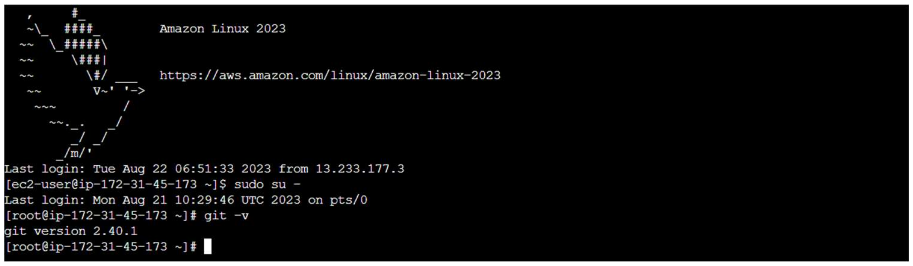
And so, my tale ends with a simple 'Hello,' a journey of code and magic, and the knowledge that even the smallest spells can have the grandest impact."

Now I am going to implement this as a practical by using manual Way and then Same thing I am going to show with the help of automation.

I have this Code in my GitHub Repository: https://github.com/Himanshu-kr-007/Java-Maven-App.git

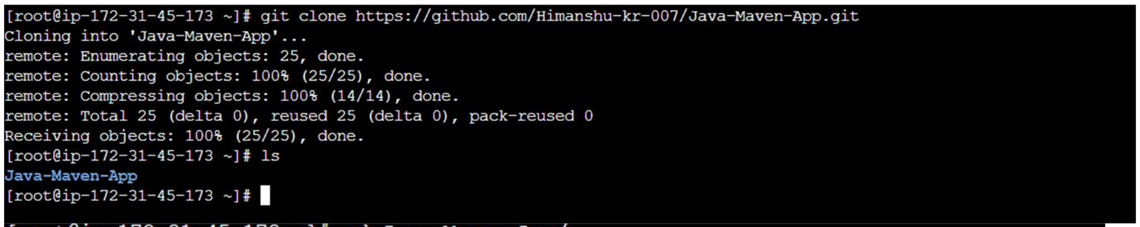


I am going to perform this practical on my Master Node of Jenkins. Which I Already created.

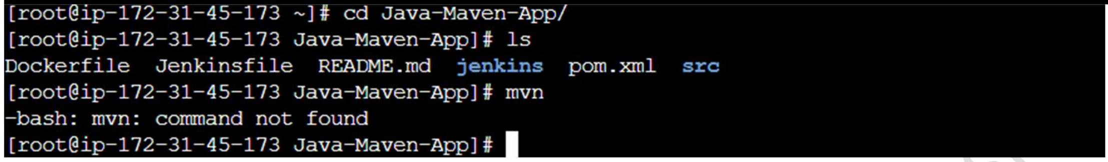


```
   ,     #_
   ~\_  ####_        Amazon Linux 2023
  ~~  \_#####\
  ~~     \###|
  ~~       \#/ ___   https://aws.amazon.com/linux/amazon-linux-2023
   ~~       V~' '->
    ~~~         /
      ~~._.   _/
         _/ _/
       _/m/'
Last login: Tue Aug 22 06:51:33 2023 from 13.233.177.3
[ec2-user@ip-172-31-45-173 ~]$ sudo su -
Last login: Mon Aug 21 10:29:46 UTC 2023 on pts/0
[root@ip-172-31-45-173 ~]# git -v
git version 2.40.1
[root@ip-172-31-45-173 ~]#
```

Here I have git Installed. Now I am going to download the code in my OS. by using git clone command.

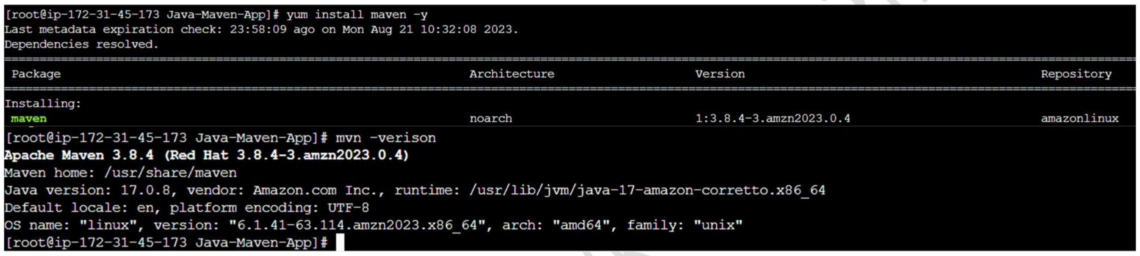


```
[root@ip-172-31-45-173 ~]# git clone https://github.com/Himanshu-kr-007/Java-Maven-App.git
Cloning into 'Java-Maven-App'...
remote: Enumerating objects: 25, done.
remote: Counting objects: 100% (25/25), done.
remote: Compressing objects: 100% (14/14), done.
remote: Total 25 (delta 0), reused 25 (delta 0), pack-reused 0
Receiving objects: 100% (25/25), done.
[root@ip-172-31-45-173 ~]# ls
Java-Maven-App
[root@ip-172-31-45-173 ~]#
```



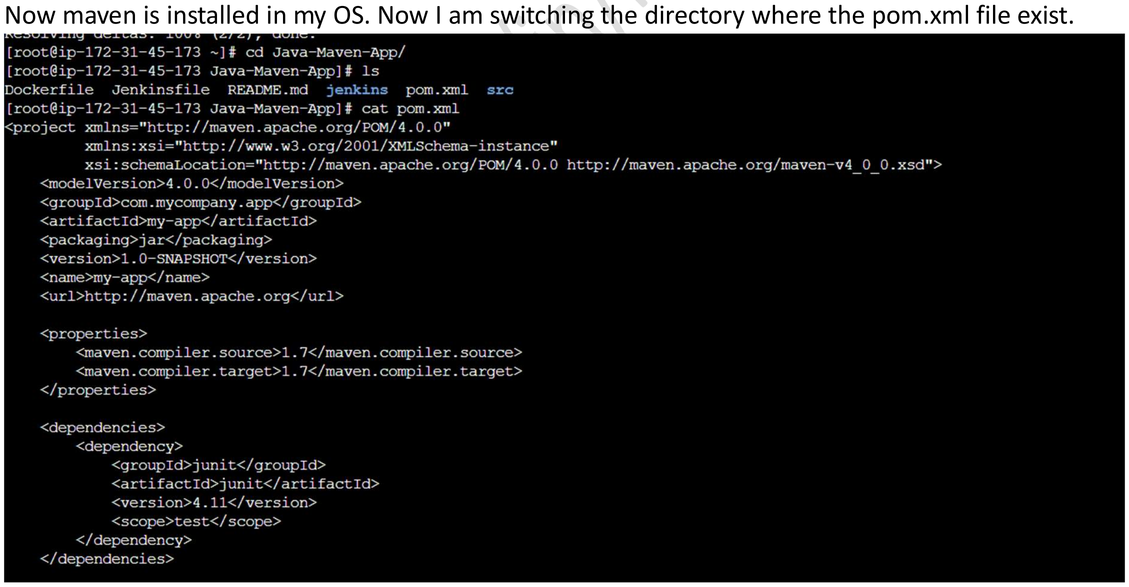
```
[root@ip-172-31-45-173 ~]# cd Java-Maven-App/
[root@ip-172-31-45-173 Java-Maven-App]# ls
Dockerfile  Jenkinsfile  README.md  jenkins  pom.xml  src
[root@ip-172-31-45-173 Java-Maven-App]# mvn
-bash: mvn: command not found
[root@ip-172-31-45-173 Java-Maven-App]#
```

For Building this Project I need maven command which is **mvn** but this command in not installed in my OS. So, I am going to install this command by using yum command.

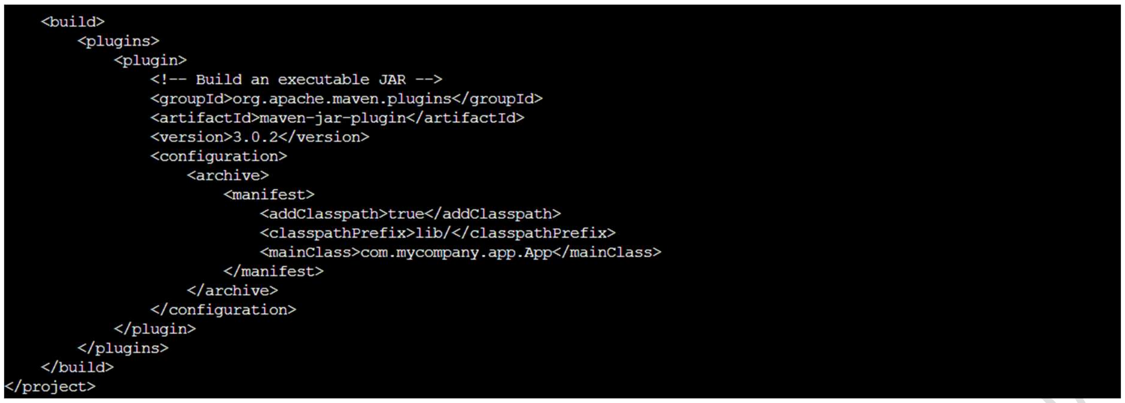


```
[root@ip-172-31-45-173 Java-Maven-App]# yum install maven -y
Last metadata expiration check: 23:58:09 ago on Mon Aug 21 10:32:08 2023.
Dependencies resolved.
==========================================================================================================================================
 Package                                         Architecture                 Version                                         Repository
==========================================================================================================================================
Installing:
 maven                                           noarch                       1:3.8.4-3.amzn2023.0.4                          amazonlinux
[root@ip-172-31-45-173 Java-Maven-App]# mvn -verison
Apache Maven 3.8.4 (Red Hat 3.8.4-3.amzn2023.0.4)
Maven home: /usr/share/maven
Java version: 17.0.8, vendor: Amazon.com Inc., runtime: /usr/lib/jvm/java-17-amazon-corretto.x86_64
Default locale: en, platform encoding: UTF-8
OS name: "linux", version: "6.1.41-63.114.amzn2023.x86_64", arch: "amd64", family: "unix"
[root@ip-172-31-45-173 Java-Maven-App]#
```

Now maven is installed in my OS. Now I am switching the directory where the pom.xml file exist.



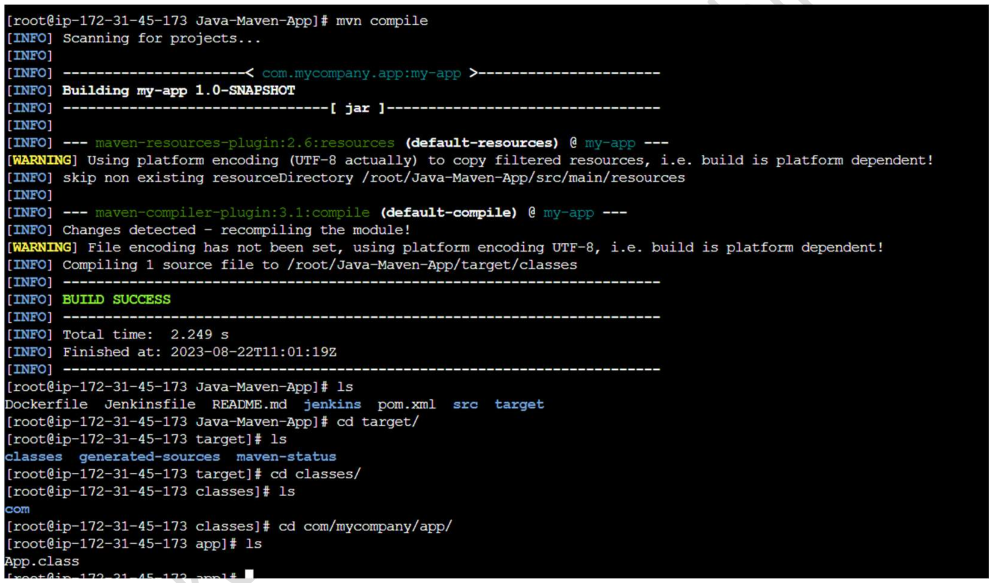
```
Resolving deltas: 100% (2/2), done.
[root@ip-172-31-45-173 ~]# cd Java-Maven-App/
[root@ip-172-31-45-173 Java-Maven-App]# ls
Dockerfile  Jenkinsfile  README.md  jenkins  pom.xml  src
[root@ip-172-31-45-173 Java-Maven-App]# cat pom.xml
<project xmlns="http://maven.apache.org/POM/4.0.0"
         xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
         xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 http://maven.apache.org/maven-v4_0_0.xsd">
    <modelVersion>4.0.0</modelVersion>
    <groupId>com.mycompany.app</groupId>
    <artifactId>my-app</artifactId>
    <packaging>jar</packaging>
    <version>1.0-SNAPSHOT</version>
    <name>my-app</name>
    <url>http://maven.apache.org</url>

    <properties>
        <maven.compiler.source>1.7</maven.compiler.source>
        <maven.compiler.target>1.7</maven.compiler.target>
    </properties>

    <dependencies>
        <dependency>
            <groupId>junit</groupId>
            <artifactId>junit</artifactId>
            <version>4.11</version>
            <scope>test</scope>
        </dependency>
    </dependencies>
```



```
    <build>
        <plugins>
            <plugin>
                <!-- Build an executable JAR -->
                <groupId>org.apache.maven.plugins</groupId>
                <artifactId>maven-jar-plugin</artifactId>
                <version>3.0.2</version>
                <configuration>
                    <archive>
                        <manifest>
                            <addClasspath>true</addClasspath>
                            <classpathPrefix>lib/</classpathPrefix>
                            <mainClass>com.mycompany.app.App</mainClass>
                        </manifest>
                    </archive>
                </configuration>
            </plugin>
        </plugins>
    </build>
</project>
```

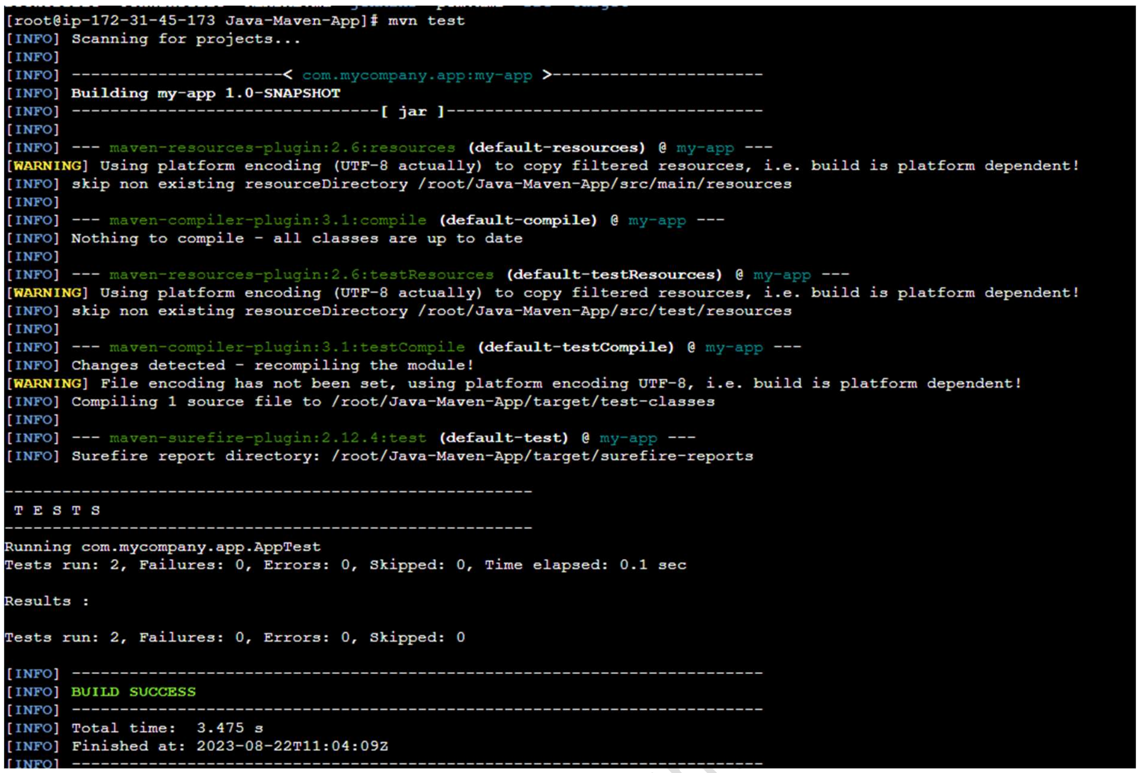
In the pom file, the dependencies are mentioned like for Building the application I need to do testing for that Here I have mentioned about junit, and for executing the application, here I have mentioned about maven-jar plugin.

Now I am going to run the command mvn Compile.

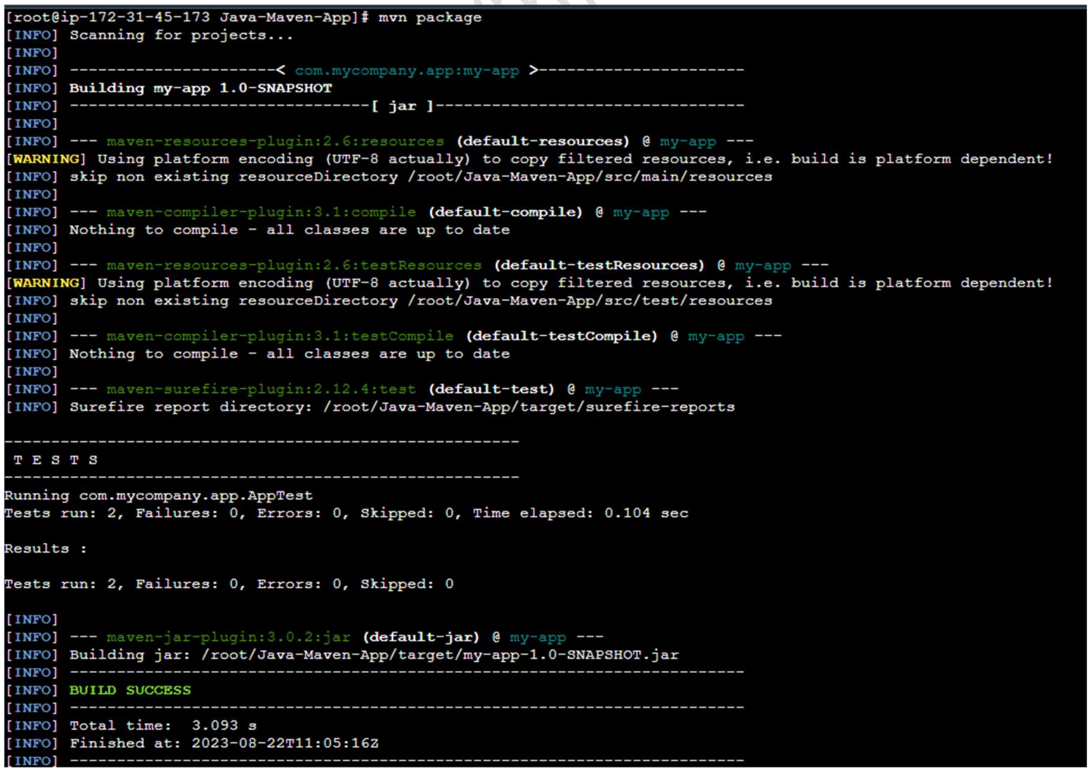


```
[root@ip-172-31-45-173 Java-Maven-App]# mvn compile
[INFO] Scanning for projects...
[INFO]
[INFO] ----------------------< com.mycompany.app:my-app >----------------------
[INFO] Building my-app 1.0-SNAPSHOT
[INFO] --------------------------------[ jar ]---------------------------------
[INFO]
[INFO] --- maven-resources-plugin:2.6:resources (default-resources) @ my-app ---
[WARNING] Using platform encoding (UTF-8 actually) to copy filtered resources, i.e. build is platform dependent!
[INFO] skip non existing resourceDirectory /root/Java-Maven-App/src/main/resources
[INFO]
[INFO] --- maven-compiler-plugin:3.1:compile (default-compile) @ my-app ---
[INFO] Changes detected - recompiling the module!
[WARNING] File encoding has not been set, using platform encoding UTF-8, i.e. build is platform dependent!
[INFO] Compiling 1 source file to /root/Java-Maven-App/target/classes
[INFO] ------------------------------------------------------------------------
[INFO] BUILD SUCCESS
[INFO] ------------------------------------------------------------------------
[INFO] Total time:  2.249 s
[INFO] Finished at: 2023-08-22T11:01:19Z
[INFO] ------------------------------------------------------------------------
[root@ip-172-31-45-173 Java-Maven-App]# ls
Dockerfile  Jenkinsfile  README.md  jenkins  pom.xml  src  target
[root@ip-172-31-45-173 Java-Maven-App]# cd target/
[root@ip-172-31-45-173 target]# ls
classes  generated-sources  maven-status
[root@ip-172-31-45-173 target]# cd classes/
[root@ip-172-31-45-173 classes]# ls
com
[root@ip-172-31-45-173 classes]# cd com/mycompany/app/
[root@ip-172-31-45-173 app]# ls
App.class
```

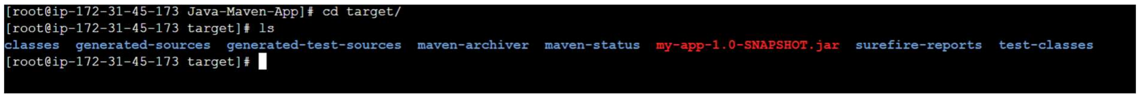
Now Maven compiled the code. Now go back to the directory where pom.xml is Present. And run the command **mvn test.**


```
[root@ip-172-31-45-173 Java-Maven-App]# mvn test
[INFO] Scanning for projects...
[INFO]
[INFO] ---------------------< com.mycompany.app:my-app >---------------------
[INFO] Building my-app 1.0-SNAPSHOT
[INFO] --------------------------------[ jar ]---------------------------------
[INFO]
[INFO] --- maven-resources-plugin:2.6:resources (default-resources) @ my-app ---
[WARNING] Using platform encoding (UTF-8 actually) to copy filtered resources, i.e. build is platform dependent!
[INFO] skip non existing resourceDirectory /root/Java-Maven-App/src/main/resources
[INFO]
[INFO] --- maven-compiler-plugin:3.1:compile (default-compile) @ my-app ---
[INFO] Nothing to compile - all classes are up to date
[INFO]
[INFO] --- maven-resources-plugin:2.6:testResources (default-testResources) @ my-app ---
[WARNING] Using platform encoding (UTF-8 actually) to copy filtered resources, i.e. build is platform dependent!
[INFO] skip non existing resourceDirectory /root/Java-Maven-App/src/test/resources
[INFO]
[INFO] --- maven-compiler-plugin:3.1:testCompile (default-testCompile) @ my-app ---
[INFO] Changes detected - recompiling the module!
[WARNING] File encoding has not been set, using platform encoding UTF-8, i.e. build is platform dependent!
[INFO] Compiling 1 source file to /root/Java-Maven-App/target/test-classes
[INFO]
[INFO] --- maven-surefire-plugin:2.12.4:test (default-test) @ my-app ---
[INFO] Surefire report directory: /root/Java-Maven-App/target/surefire-reports

-------------------------------------------------------
 T E S T S
-------------------------------------------------------
Running com.mycompany.app.AppTest
Tests run: 2, Failures: 0, Errors: 0, Skipped: 0, Time elapsed: 0.1 sec

Results :

Tests run: 2, Failures: 0, Errors: 0, Skipped: 0

[INFO] ------------------------------------------------------------------------
[INFO] BUILD SUCCESS
[INFO] ------------------------------------------------------------------------
[INFO] Total time:  3.475 s
[INFO] Finished at: 2023-08-22T11:04:09Z
[INFO] ------------------------------------------------------------------------
```

Now Implement the command **mvn package**. This command will download the library and build the package which is present in the target folder.

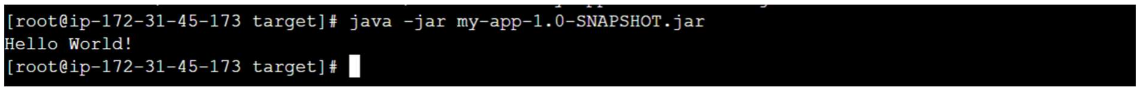

```
[root@ip-172-31-45-173 Java-Maven-App]# mvn package
[INFO] Scanning for projects...
[INFO]
[INFO] ---------------------< com.mycompany.app:my-app >---------------------
[INFO] Building my-app 1.0-SNAPSHOT
[INFO] --------------------------------[ jar ]---------------------------------
[INFO]
[INFO] --- maven-resources-plugin:2.6:resources (default-resources) @ my-app ---
[WARNING] Using platform encoding (UTF-8 actually) to copy filtered resources, i.e. build is platform dependent!
[INFO] skip non existing resourceDirectory /root/Java-Maven-App/src/main/resources
[INFO]
[INFO] --- maven-compiler-plugin:3.1:compile (default-compile) @ my-app ---
[INFO] Nothing to compile - all classes are up to date
[INFO]
[INFO] --- maven-resources-plugin:2.6:testResources (default-testResources) @ my-app ---
[WARNING] Using platform encoding (UTF-8 actually) to copy filtered resources, i.e. build is platform dependent!
[INFO] skip non existing resourceDirectory /root/Java-Maven-App/src/test/resources
[INFO]
[INFO] --- maven-compiler-plugin:3.1:testCompile (default-testCompile) @ my-app ---
[INFO] Nothing to compile - all classes are up to date
[INFO]
[INFO] --- maven-surefire-plugin:2.12.4:test (default-test) @ my-app ---
[INFO] Surefire report directory: /root/Java-Maven-App/target/surefire-reports

-------------------------------------------------------
 T E S T S
-------------------------------------------------------
Running com.mycompany.app.AppTest
Tests run: 2, Failures: 0, Errors: 0, Skipped: 0, Time elapsed: 0.104 sec

Results :

Tests run: 2, Failures: 0, Errors: 0, Skipped: 0

[INFO]
[INFO] --- maven-jar-plugin:3.0.2:jar (default-jar) @ my-app ---
[INFO] Building jar: /root/Java-Maven-App/target/my-app-1.0-SNAPSHOT.jar
[INFO] ------------------------------------------------------------------------
[INFO] BUILD SUCCESS
[INFO] ------------------------------------------------------------------------
[INFO] Total time:  3.093 s
[INFO] Finished at: 2023-08-22T11:05:16Z
[INFO] ------------------------------------------------------------------------
```

The package is successfully built and stored in the Target directory.



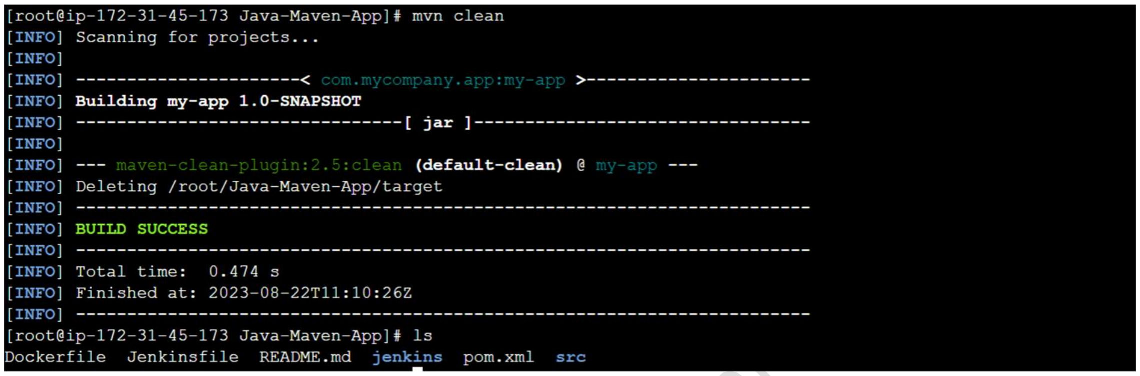
```
[root@ip-172-31-45-173 Java-Maven-App]# cd target/
[root@ip-172-31-45-173 target]# ls
classes  generated-sources  generated-test-sources  maven-archiver  maven-status  my-app-1.0-SNAPSHOT.jar  surefire-reports  test-classes
[root@ip-172-31-45-173 target]#
```

To run this package, we need to use java command. **java -jar package-name.jar**



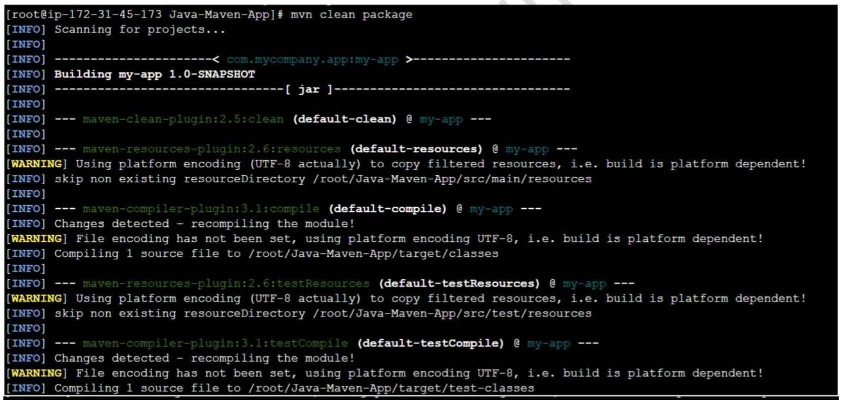
```
[root@ip-172-31-45-173 target]# java -jar my-app-1.0-SNAPSHOT.jar
Hello World!
[root@ip-172-31-45-173 target]#
```

After that I am going to clean the target file.

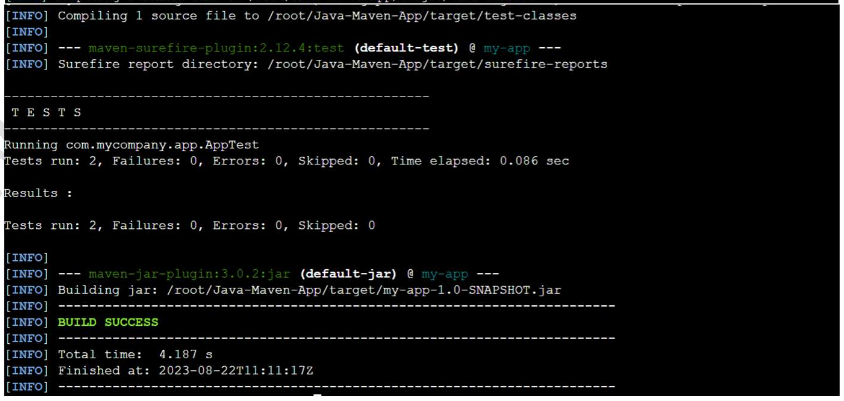


```
[root@ip-172-31-45-173 Java-Maven-App]# mvn clean
[INFO] Scanning for projects...
[INFO]
[INFO] ----------------------< com.mycompany.app:my-app >----------------------
[INFO] Building my-app 1.0-SNAPSHOT
[INFO] --------------------------------[ jar ]---------------------------------
[INFO]
[INFO] --- maven-clean-plugin:2.5:clean (default-clean) @ my-app ---
[INFO] Deleting /root/Java-Maven-App/target
[INFO] ------------------------------------------------------------------------
[INFO] BUILD SUCCESS
[INFO] ------------------------------------------------------------------------
[INFO] Total time:  0.474 s
[INFO] Finished at: 2023-08-22T11:10:26Z
[INFO] ------------------------------------------------------------------------
[root@ip-172-31-45-173 Java-Maven-App]# ls
Dockerfile  Jenkinsfile  README.md  jenkins  pom.xml  src
```

Here this command will remove the target directory.



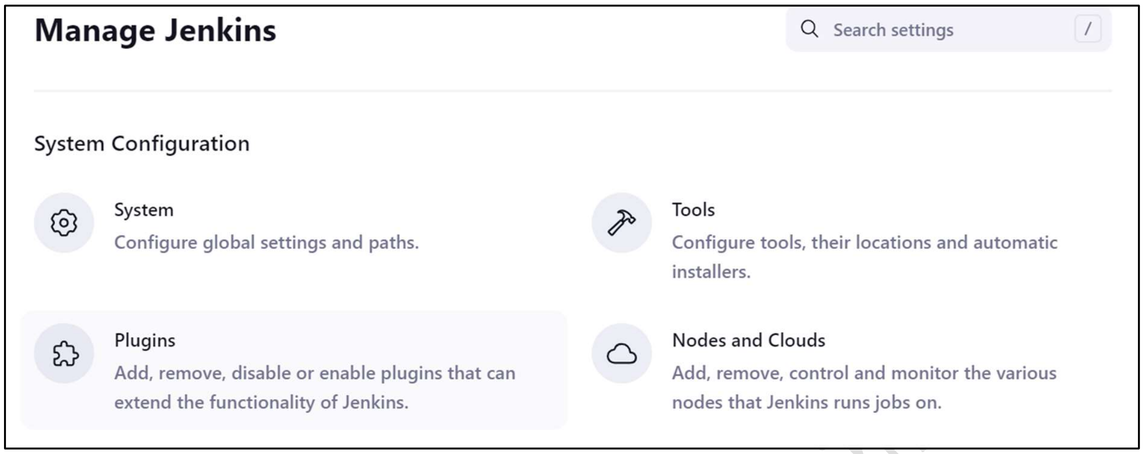
```
[root@ip-172-31-45-173 Java-Maven-App]# mvn clean package
[INFO] Scanning for projects...
[INFO]
[INFO] ----------------------< com.mycompany.app:my-app >----------------------
[INFO] Building my-app 1.0-SNAPSHOT
[INFO] --------------------------------[ jar ]---------------------------------
[INFO]
[INFO] --- maven-clean-plugin:2.5:clean (default-clean) @ my-app ---
[INFO]
[INFO] --- maven-resources-plugin:2.6:resources (default-resources) @ my-app ---
[WARNING] Using platform encoding (UTF-8 actually) to copy filtered resources, i.e. build is platform dependent!
[INFO] skip non existing resourceDirectory /root/Java-Maven-App/src/main/resources
[INFO]
[INFO] --- maven-compiler-plugin:3.1:compile (default-compile) @ my-app ---
[INFO] Changes detected - recompiling the module!
[WARNING] File encoding has not been set, using platform encoding UTF-8, i.e. build is platform dependent!
[INFO] Compiling 1 source file to /root/Java-Maven-App/target/classes
[INFO]
[INFO] --- maven-resources-plugin:2.6:testResources (default-testResources) @ my-app ---
[WARNING] Using platform encoding (UTF-8 actually) to copy filtered resources, i.e. build is platform dependent!
[INFO] skip non existing resourceDirectory /root/Java-Maven-App/src/test/resources
[INFO]
[INFO] --- maven-compiler-plugin:3.1:testCompile (default-testCompile) @ my-app ---
[INFO] Changes detected - recompiling the module!
[WARNING] File encoding has not been set, using platform encoding UTF-8, i.e. build is platform dependent!
[INFO] Compiling 1 source file to /root/Java-Maven-App/target/test-classes
```



```
[INFO] Compiling 1 source file to /root/Java-Maven-App/target/test-classes
[INFO]
[INFO] --- maven-surefire-plugin:2.12.4:test (default-test) @ my-app ---
[INFO] Surefire report directory: /root/Java-Maven-App/target/surefire-reports

-------------------------------------------------------
 T E S T S
-------------------------------------------------------
Running com.mycompany.app.AppTest
Tests run: 2, Failures: 0, Errors: 0, Skipped: 0, Time elapsed: 0.086 sec

Results :

Tests run: 2, Failures: 0, Errors: 0, Skipped: 0

[INFO]
[INFO] --- maven-jar-plugin:3.0.2:jar (default-jar) @ my-app ---
[INFO] Building jar: /root/Java-Maven-App/target/my-app-1.0-SNAPSHOT.jar
[INFO] ------------------------------------------------------------------------
[INFO] BUILD SUCCESS
[INFO] ------------------------------------------------------------------------
[INFO] Total time:  4.187 s
[INFO] Finished at: 2023-08-22T11:11:17Z
[INFO] ------------------------------------------------------------------------
```

I run the command mvn clean package and it is first cleaning the package then compile it then test and then create the package.

Here I have done the entire step manually, but now I am going to achieve this thing with the help of Jenkins. For this in slave nod we need to install git and maven and here we want Jenkins to contact maven for building the package. In this case we need plugins for maven.



Manage Jenkins

System Configuration

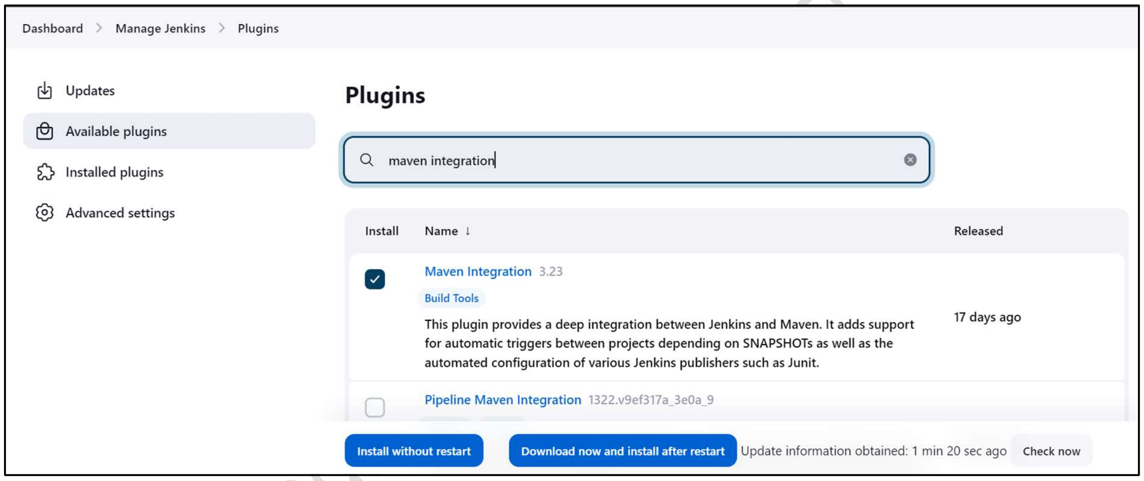
System
Configure global settings and paths.

Tools
Configure tools, their locations and automatic installers.

Plugins
Add, remove, disable or enable plugins that can extend the functionality of Jenkins.

Nodes and Clouds
Add, remove, control and monitor the various nodes that Jenkins runs jobs on.

From Manage Jenkins Click on Plugins and search maven integration from available plugins.



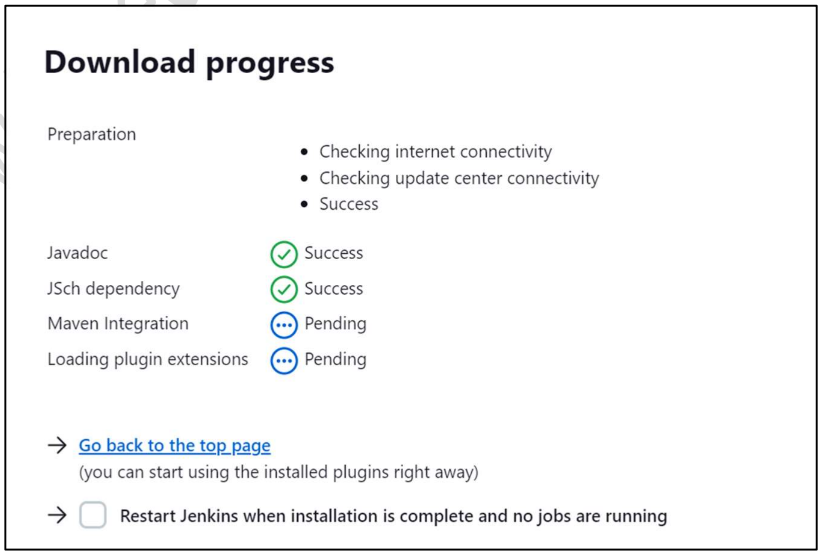
Dashboard > Manage Jenkins > Plugins

Plugins

maven integration

Install	Name ↓	Released
✓	Maven Integration 3.23 Build Tools This plugin provides a deep integration between Jenkins and Maven. It adds support for automatic triggers between projects depending on SNAPSHOTs as well as the automated configuration of various Jenkins publishers such as Junit.	17 days ago
☐	Pipeline Maven Integration 1322.v9ef317a_3e0a_9	

Update information obtained: 1 min 20 sec ago



Download progress

Preparation
- Checking internet connectivity
- Checking update center connectivity
- Success

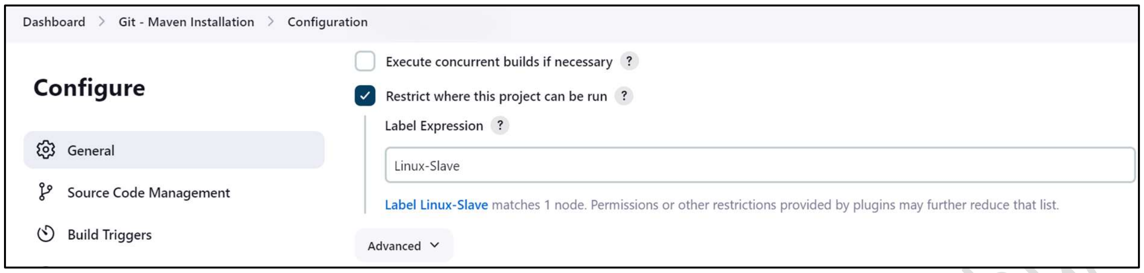
Javadoc	✓ Success
JSch dependency	✓ Success
Maven Integration	Pending
Loading plugin extensions	Pending

→ Go back to the top page
(you can start using the installed plugins right away)

→ ☐ Restart Jenkins when installation is complete and no jobs are running

After Installation, we need to restart the Jenkins so that we can use this plugin. Just check the box for that. Till that time, I am going to create one job to install git and maven in my Slave system.

Click on + Item to create a new job of free style.



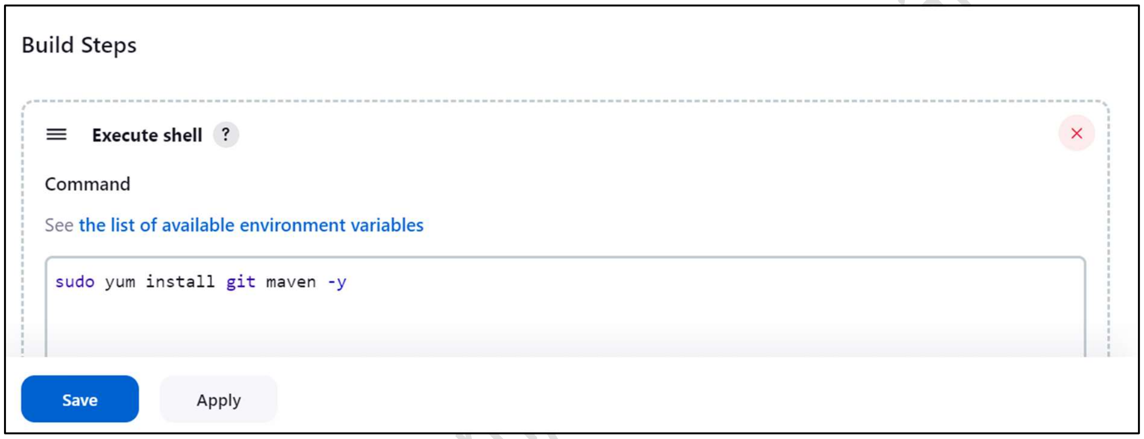
[ ] Execute concurrent builds if necessary

[✓] Restrict where this project can be run

Label Expression: Linux-Slave

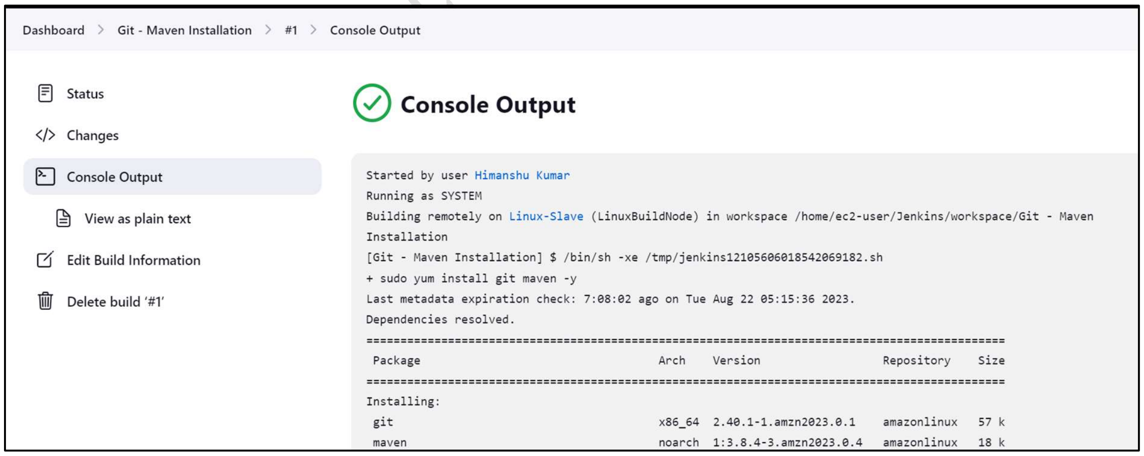
Label Linux-Slave matches 1 node. Permissions or other restrictions provided by plugins may further reduce that list.

Here I have selected this Job will only run in Linux-Slave and then in Build Step option I have given the command to install the git and maven.



```
sudo yum install git maven -y
```

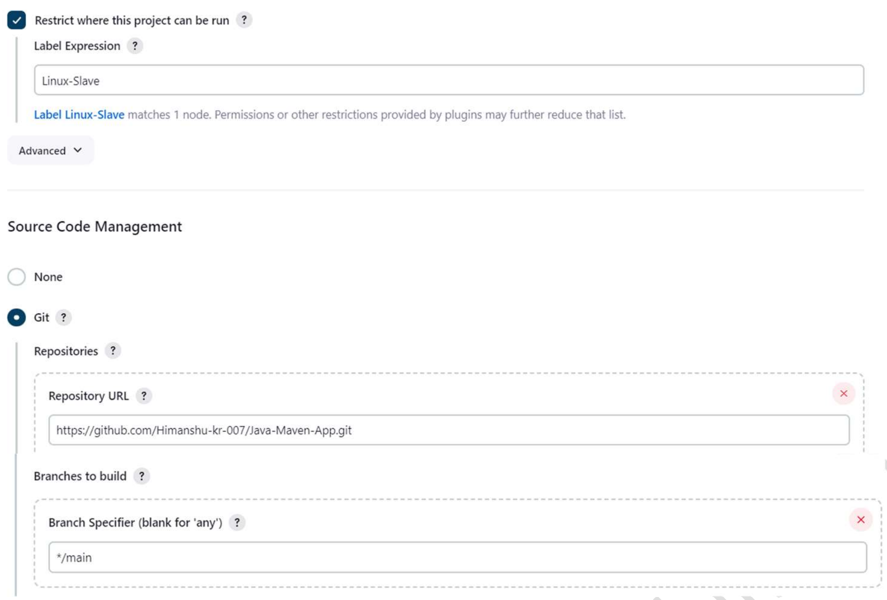
After that I click on build Now, then this job is successfully built and we can see the result.



```
Started by user Himanshu Kumar
Running as SYSTEM
Building remotely on Linux-Slave (LinuxBuildNode) in workspace /home/ec2-user/Jenkins/workspace/Git - Maven
Installation
[Git - Maven Installation] $ /bin/sh -xe /tmp/jenkins12105606018542069182.sh
+ sudo yum install git maven -y
Last metadata expiration check: 7:08:02 ago on Tue Aug 22 05:15:36 2023.
Dependencies resolved.
==============================================================================================
 Package                                  Arch     Version                    Repository    Size
==============================================================================================
Installing:
 git                                      x86_64  2.40.1-1.amzn2023.0.1      amazonlinux   57 k
 maven                                    noarch  1:3.8.4-3.amzn2023.0.4     amazonlinux   18 k
```

Now I am going to Create new Job where I am going to build the Project.

I am going to run this Job on my Slave Node for that I selected that.



Restrict where this project can be run

Label Expression

Linux-Slave

Label Linux-Slave matches 1 node. Permissions or other restrictions provided by plugins may further reduce that list.

Source Code Management

Repositories

Repository URL

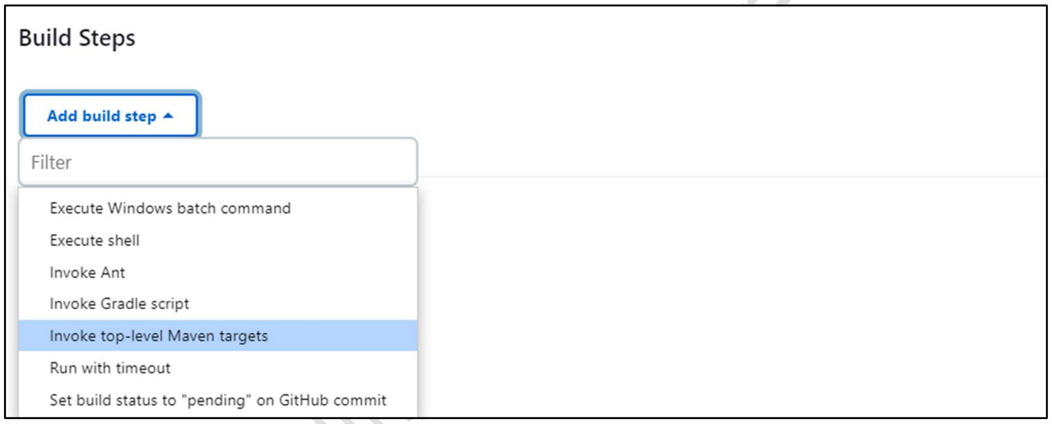
https://github.com/Himanshu-kr-007/Java-Maven-App.git

Branches to build

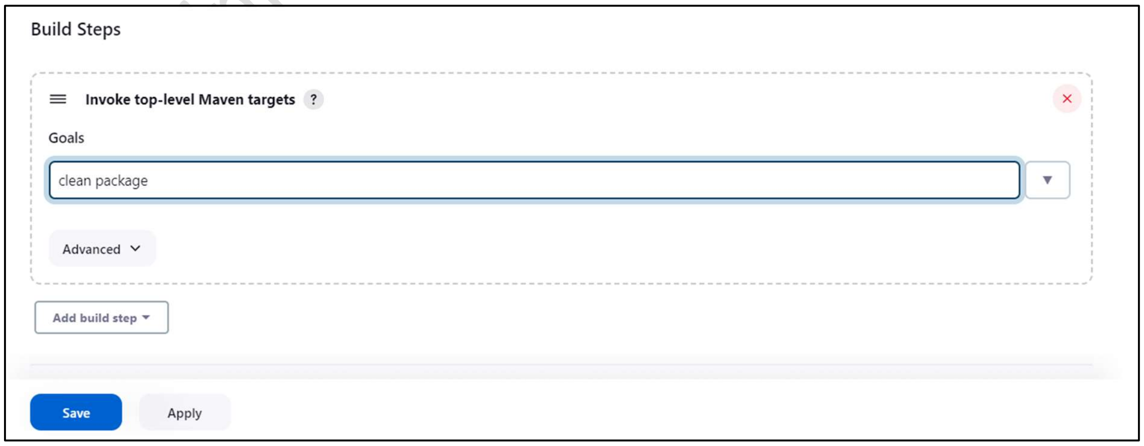
Branch Specifier (blank for 'any')

*/main

From the Build step I am going to select the Invoke top-level Maven target


Build Steps
Add build step
Filter
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

And in the input box I am going to write my final requirement which is **clean package**. Then clicking on save button.



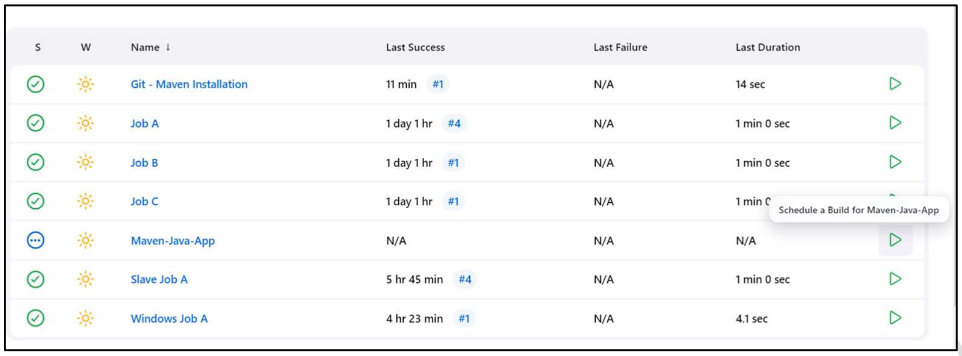
Build Steps

Invoke top-level Maven targets

Goals

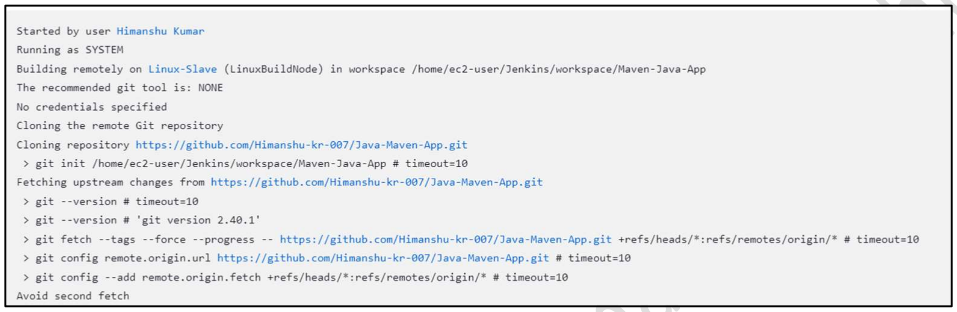
clean package

Now going to run this Job by clicking on Play button.



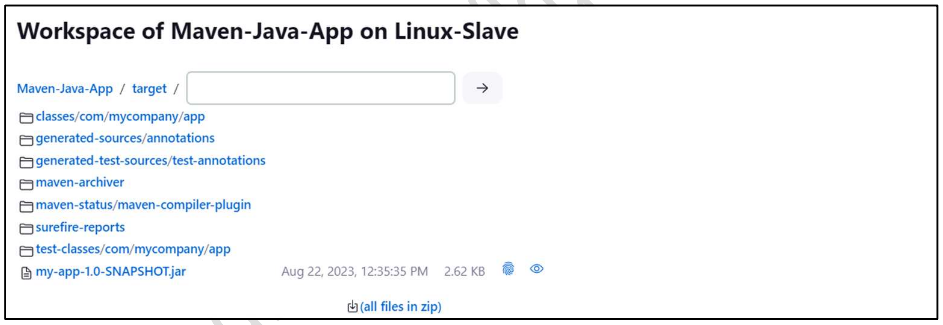
S	W	Name ↓	Last Success	Last Failure	Last Duration	
✓	☀	Git - Maven Installation	11 min #1	N/A	14 sec	▷
✓	☀	Job A	1 day 1 hr #4	N/A	1 min 0 sec	▷
✓	☀	Job B	1 day 1 hr #1	N/A	1 min 0 sec	▷
✓	☀	Job C	1 day 1 hr #1	N/A	1 min 0	
⋯	☀	Maven-Java-App	N/A	N/A	N/A	▷
✓	☀	Slave Job A	5 hr 45 min #4	N/A	1 min 0 sec	▷
✓	☀	Windows Job A	4 hr 23 min #1	N/A	4.1 sec	▷

And The Build is start and we can see the results also.



```
Started by user Himanshu Kumar
Running as SYSTEM
Building remotely on Linux-Slave (LinuxBuildNode) in workspace /home/ec2-user/Jenkins/workspace/Maven-Java-App
The recommended git tool is: NONE
No credentials specified
Cloning the remote Git repository
Cloning repository https://github.com/Himanshu-kr-007/Java-Maven-App.git
 > git init /home/ec2-user/Jenkins/workspace/Maven-Java-App # timeout=10
Fetching upstream changes from https://github.com/Himanshu-kr-007/Java-Maven-App.git
 > git --version # timeout=10
 > git --version # 'git version 2.40.1'
 > git fetch --tags --force --progress -- https://github.com/Himanshu-kr-007/Java-Maven-App.git +refs/heads/*:refs/remotes/origin/* # timeout=10
 > git config remote.origin.url https://github.com/Himanshu-kr-007/Java-Maven-App.git # timeout=10
 > git config --add remote.origin.fetch +refs/heads/*:refs/remotes/origin/* # timeout=10
Avoid second fetch
```

But this code is also generating a class file which we can retrieve by using Artifact from workspace.



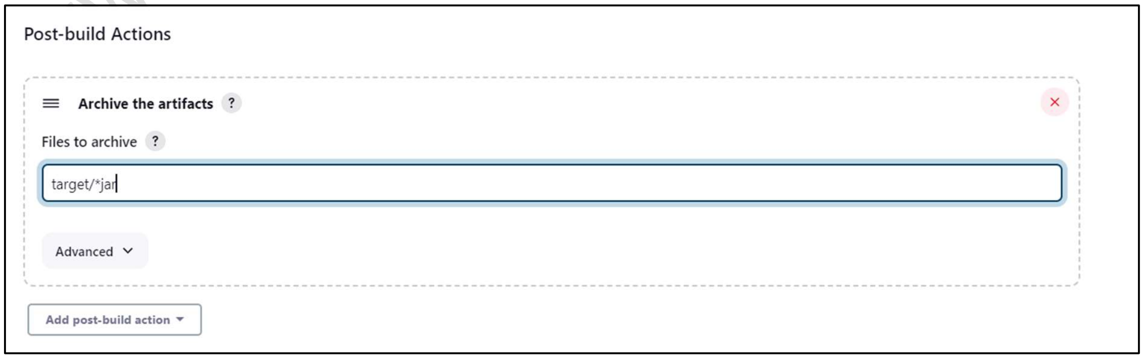
Workspace of Maven-Java-App on Linux-Slave

Maven-Java-App / target /

- classes/com/mycompany/app
- generated-sources/annotations
- generated-test-sources/test-annotations
- maven-archiver
- maven-status/maven-compiler-plugin
- surefire-reports
- test-classes/com/mycompany/app
- my-app-1.0-SNAPSHOT.jar   Aug 22, 2023, 12:35:35 PM   2.62 KB

(all files in zip)

Currently, these package are installed in the Slaved node and in real world we are not using the static System as slave, their we are using dynamic slave where after the execution of Job the slave was deleted. But their we need to save some executable file which we can use for other purpose. In that case we can use the help for Post Build Action. After execution of Job and before deletion of node. This feature will help to collect the artifact and store in the local workspace for Job.



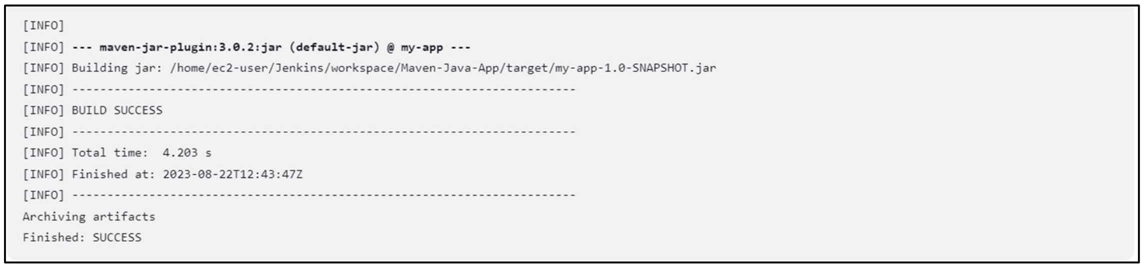
Post-build Actions

Archive the artifacts ?

Files to archive ?

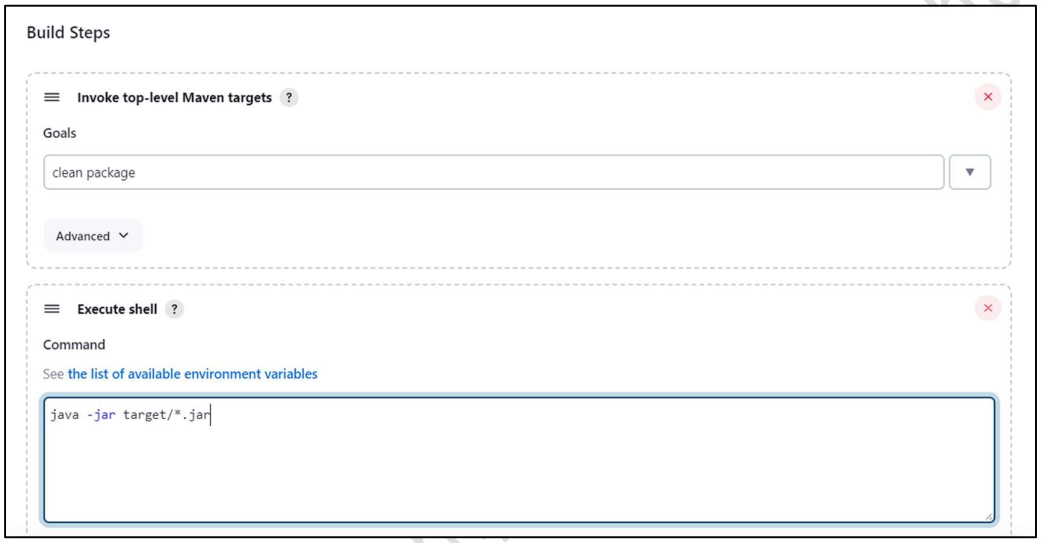
target/*jar

Now click on Build Now Job.



```
[INFO]
[INFO] --- maven-jar-plugin:3.0.2:jar (default-jar) @ my-app ---
[INFO] Building jar: /home/ec2-user/Jenkins/workspace/Maven-Java-App/target/my-app-1.0-SNAPSHOT.jar
[INFO] ------------------------------------------------------------------------
[INFO] BUILD SUCCESS
[INFO] ------------------------------------------------------------------------
[INFO] Total time:  4.203 s
[INFO] Finished at: 2023-08-22T12:43:47Z
[INFO] ------------------------------------------------------------------------
Archiving artifacts
Finished: SUCCESS
```

Then it will store the Artifacts for us. Now, I am going execute the artifacts. By adding the execute shell option.



Build Steps

Invoke top-level Maven targets

Goals: clean package

Advanced

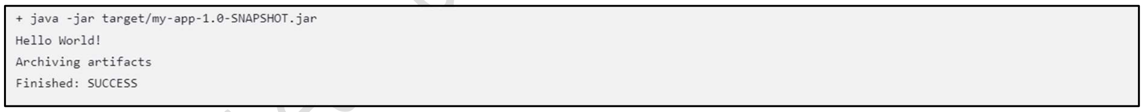
Execute shell

Command

See the list of available environment variables

```
java -jar target/*.jar
```

After Building this Job we can see the result in the output.



```
+ java -jar target/my-app-1.0-SNAPSHOT.jar
Hello World!
Archiving artifacts
Finished: SUCCESS
```

SMALL TIP: While Creating Slave Nodes. If You have created slave Nodes with Suppose Name "Webservers deploy" and if you have a requirement to run 10 webservers in 10 system in one go then you can deploy the webserver in all of them by using this label name.