Maven Project  
structure

Maven Project

src

Pom.xml

main

com

Company\_name

app

test

com

Company\_name

app

App.java

AppTest.java

On Jenkins server create the structure

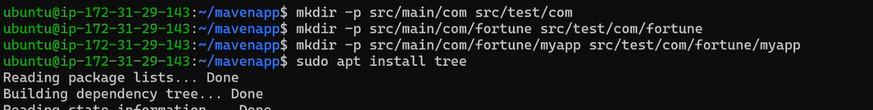
mkdir mavenapp

cd mavenappcd

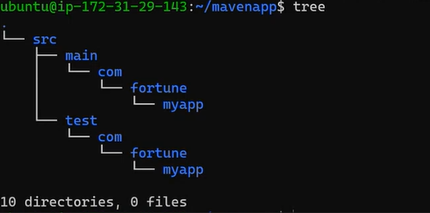


mkdir -p src/main/com/fortune/myapp src/test/com/fortune/myapp

sudo apt install tree



tree



Copy the code of pom.xml, App.java and AppTest.java from below github link and make changes as per your requirements

<https://github.com/jenkins-docs/simple-java-maven-app/blob/master/pom.xml>

**nano 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>https://maven.apache.org</url>

<dependencies>

<dependency>

<groupId>org.junit.jupiter</groupId>

<artifactId>junit-jupiter-api</artifactId>

<version>5.11.4</version>

<scope>test</scope>

</dependency>

</dependencies>

<properties>

<project.build.sourceEncoding>UTF-8</project.build.sourceEncoding>

</properties>

<build>

<pluginManagement>

<plugins>

<plugin>

<groupId>org.apache.maven.plugins</groupId>

<artifactId>maven-compiler-plugin</artifactId>

<version>3.13.0</version>

<configuration>

<release>17</release>

</configuration>

</plugin>

</plugins>

</pluginManagement>

<plugins>

<plugin>

<!-- Build an executable JAR -->

<groupId>org.apache.maven.plugins</groupId>

<artifactId>maven-jar-plugin</artifactId>

<version>3.4.2</version>

<configuration>

<archive>

<manifest>

<addClasspath>true</addClasspath>

<classpathPrefix>lib/</classpathPrefix>

<mainClass>com.mycompany.app.App</mainClass>

</manifest>

</archive>

</configuration>

</plugin>

<plugin>

<groupId>org.apache.maven.plugins</groupId>

<artifactId>maven-enforcer-plugin</artifactId>

<version>3.5.0</version>

<executions>

<execution>

<id>enforce-maven</id>

<goals>

<goal>enforce</goal>

</goals>

<configuration>

<rules>

<requireMavenVersion>

<version>[3.9.2,)</version>

</requireMavenVersion>

<requireJavaVersion>

<version>[17,)</version>

</requireJavaVersion>

</rules>

</configuration>

</execution>

</executions>

</plugin>

</plugins>

nan </build>

</project>

**nano src/main/com/fortune/myapp/App.java**

package com.mycompany.app;

/\*\*

\* Hello world!

\*/

public class App {

private static final String MESSAGE = "Hello World!";

public App() {}

public static void main(String[] args) {

System.out.println(MESSAGE);

}

public String getMessage() {

return MESSAGE;

}

}

**nano src/test/com/fortune/myapp/AppTest.java**

package com.mycompany.app;

import org.junit.jupiter.api.Test;

import static org.junit.jupiter.api.Assertions.assertEquals;

/\*\*

\* Unit test for simple App.

\*/

public class AppTest

{

@Test

public void testAppConstructor() {

App app1 = new App();

App app2 = new App();

assertEquals(app1.getMessage(), app2.getMessage());

}

@Test

public void testAppMessage()

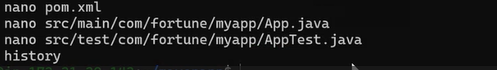
{

App app = new App();

assertEquals("Hello World!", app.getMessage());

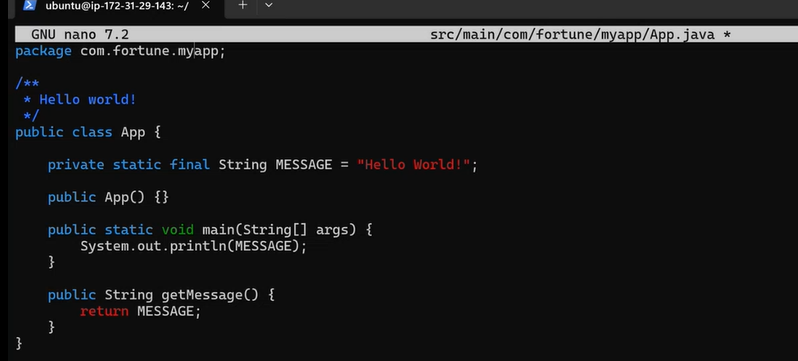
}

}



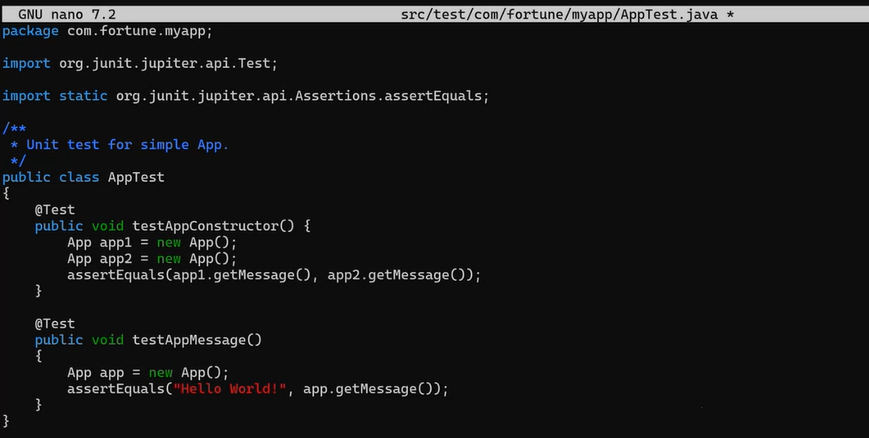
Open the App.java file and make changes in 1st line (mycompany -> fortune and app -> myapp)

**nano src/main/com/fortune/myapp/App.java**



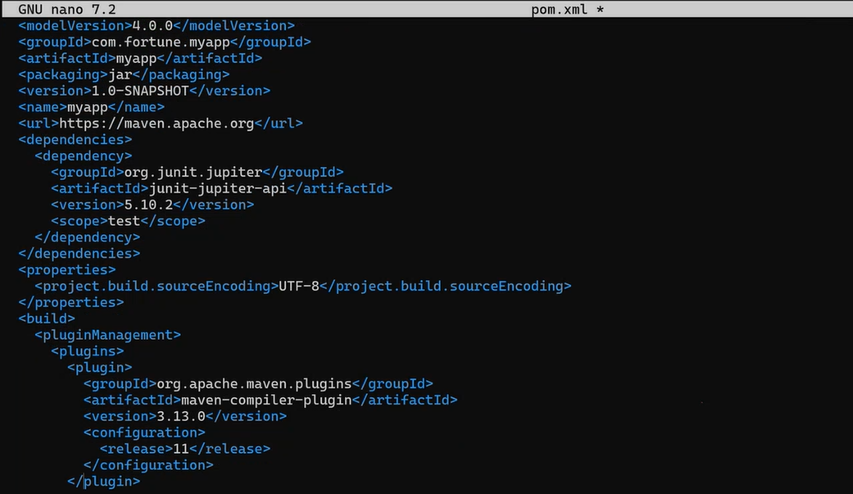
Open the AppTest.java file and make the same changes as we did in App.java in 1st line (mycompany -> fortune and app -> myapp)

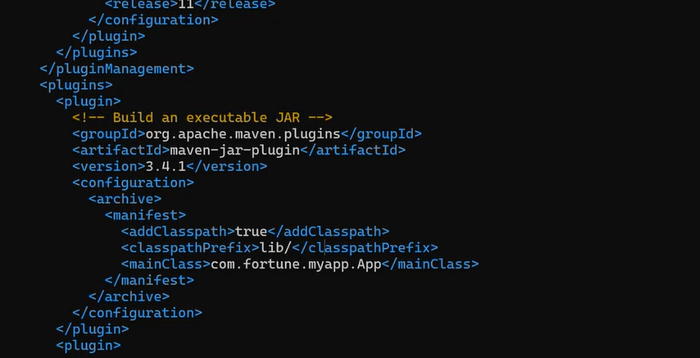
**nano src/test/com/fortune/myapp/AppTest.java**



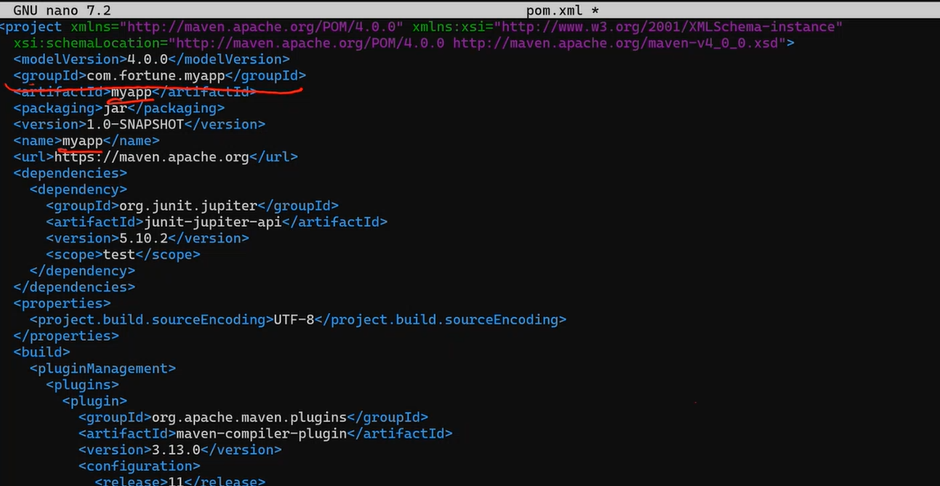
Now open pom.xml (changes in 4 lines)

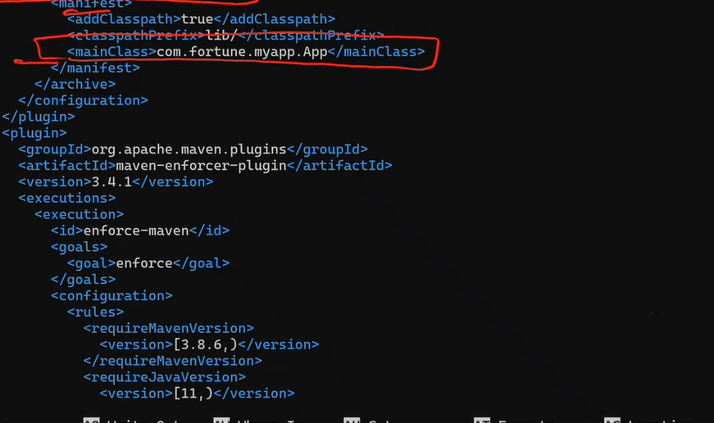
**nano pom.xml**





Changes highlighted





In mavenapp/ dir

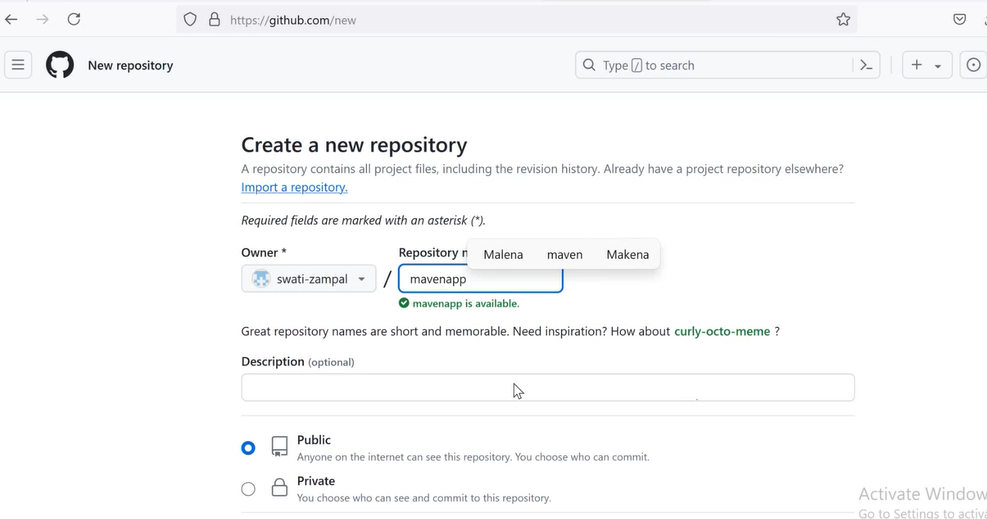
git init

git config - -global user.name “usernamei”

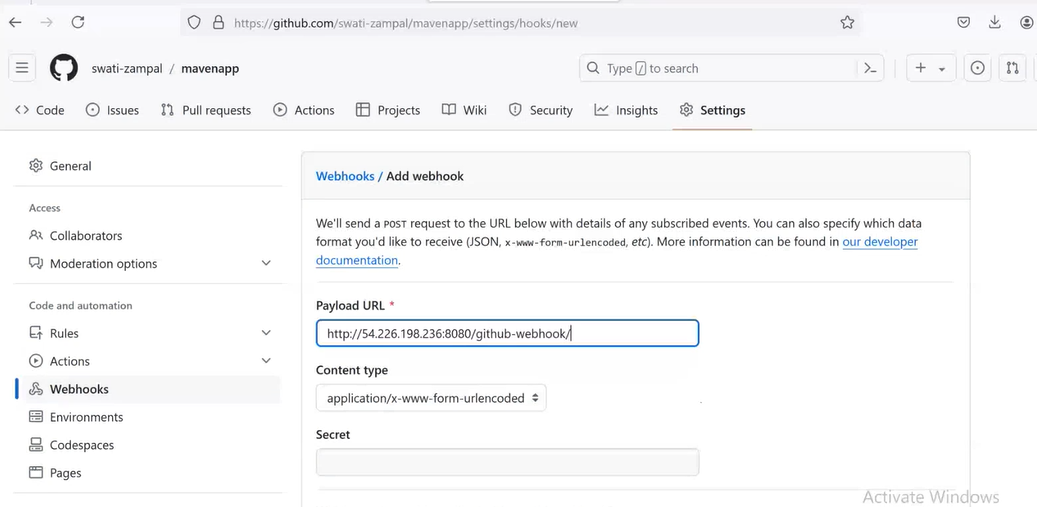
git config - -global user.email “email-id”

git remote add origin git-repo-link

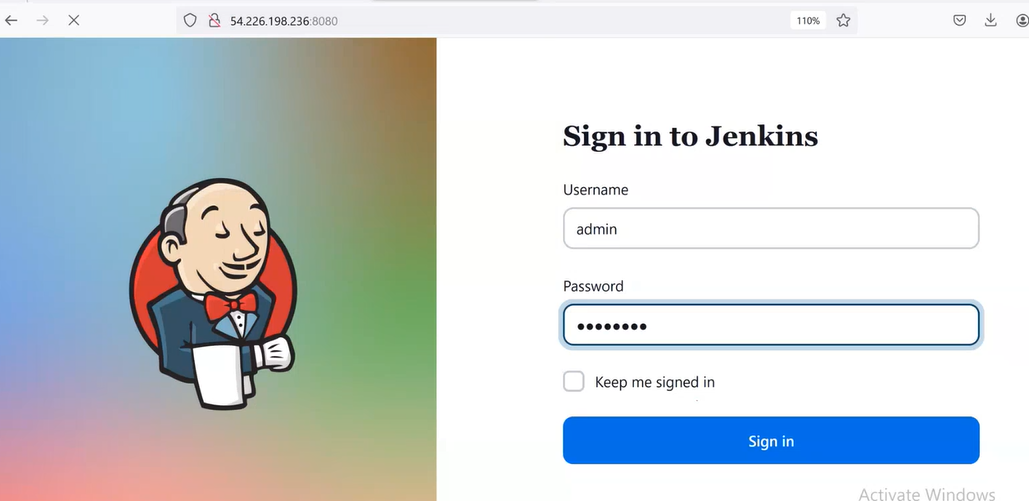
Create a new repository on Github



Set webhook to trigger Jenkins



Login to Jenkins through browser



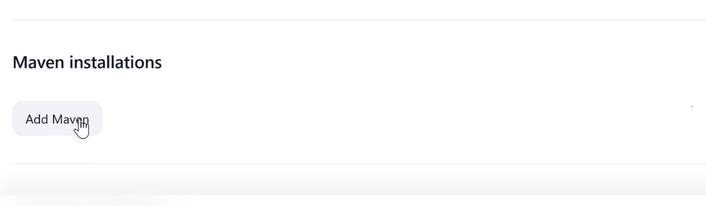
Do setup for Maven and JDK tool which is installed already while configuring Jenkins 1st time.

Dashboard à Manage Jenkins à Tools à JDK installations à Add JDK à Enter Name à install automatically





Below



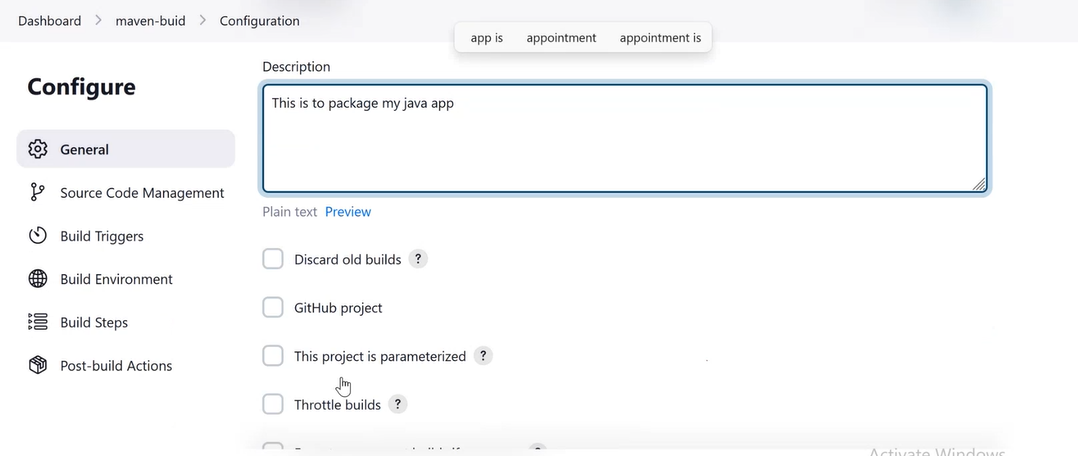


Save

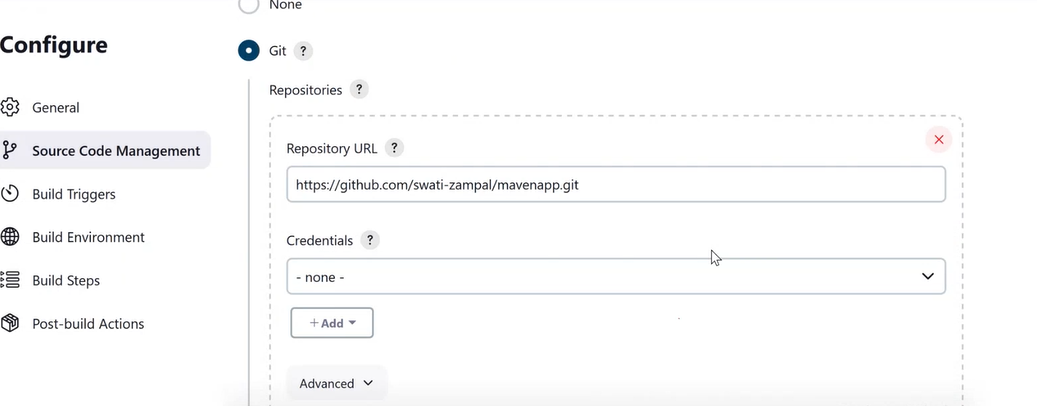
Create new freestyle Project

Dashboard à New Items à

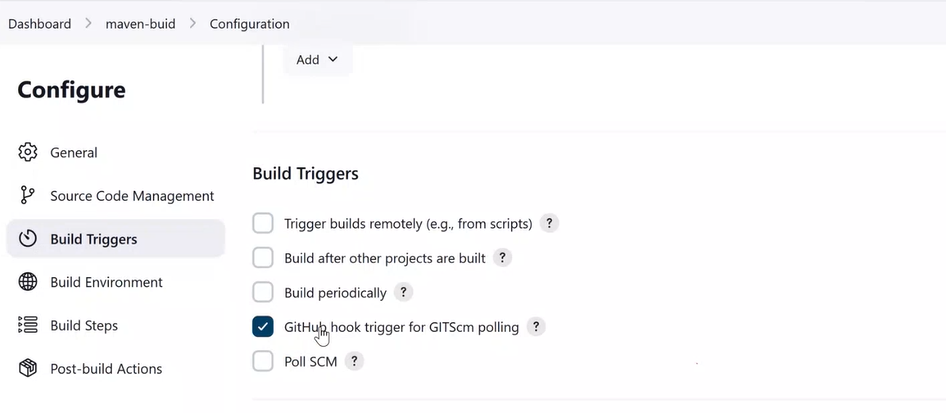
Step 1: General



Step 2: SCM



Step 3: Build Trigger

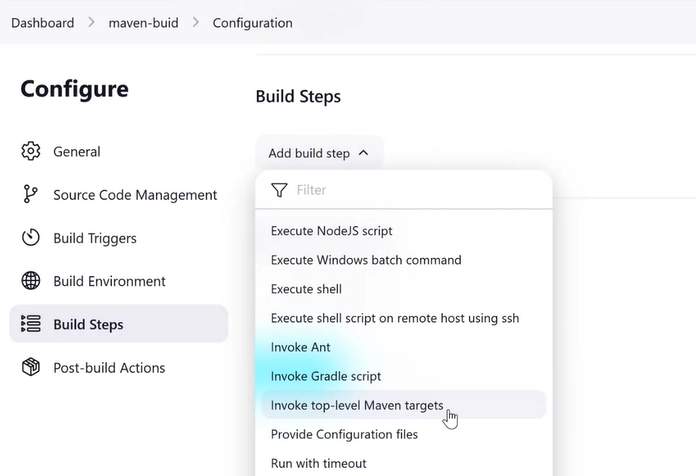


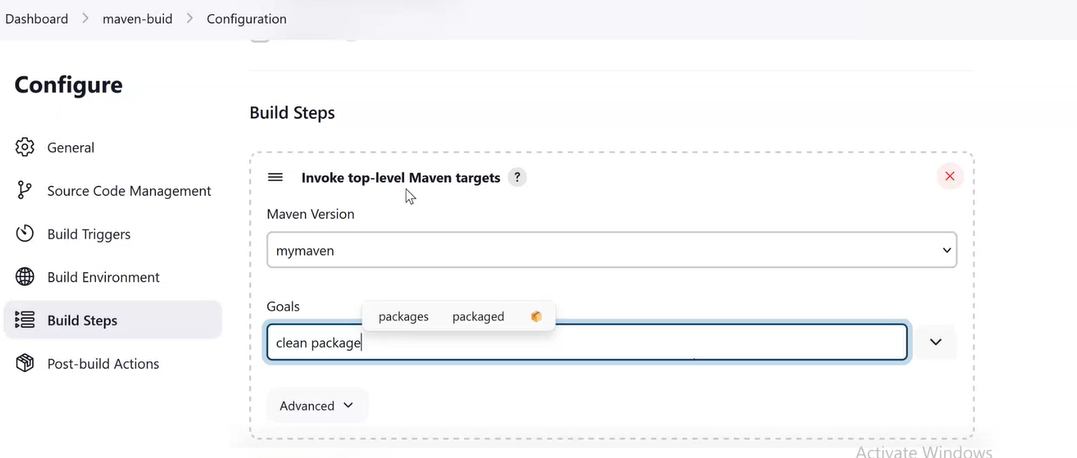
Step 4: Build Environment

(No changes)

Step 5: Build Step

Add build step à Invoke top-level Maven targets





Save it.

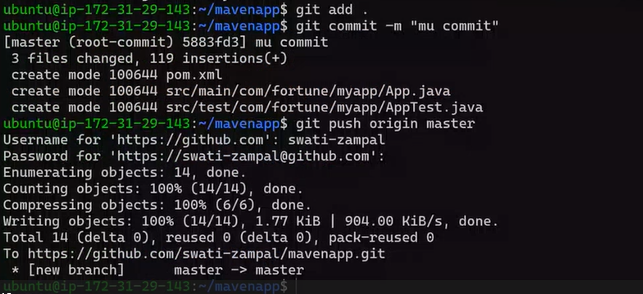
Now push the code files to github repo

cd mavenapp

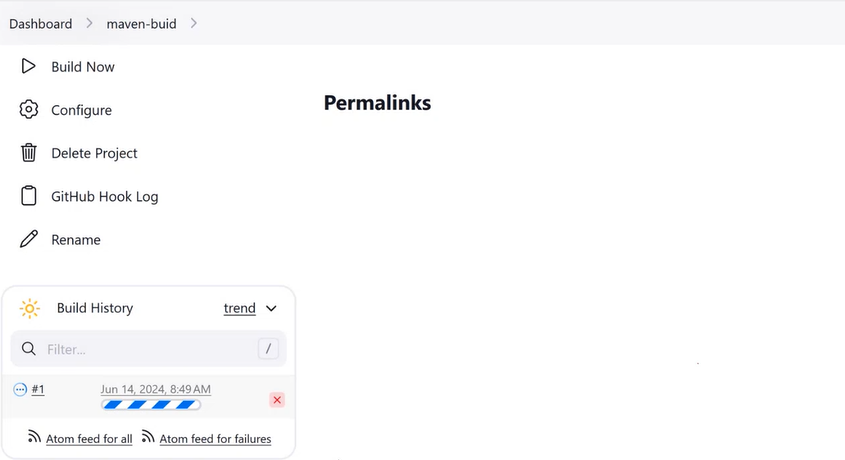
git add .

git commit –m “maven commit”

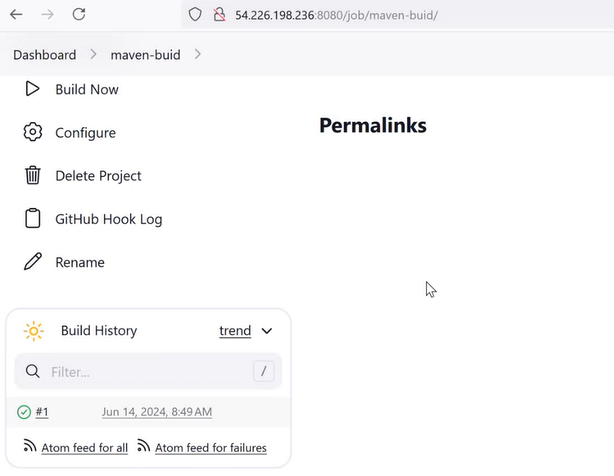
git push origin master

git

Check on browser, your project started building:

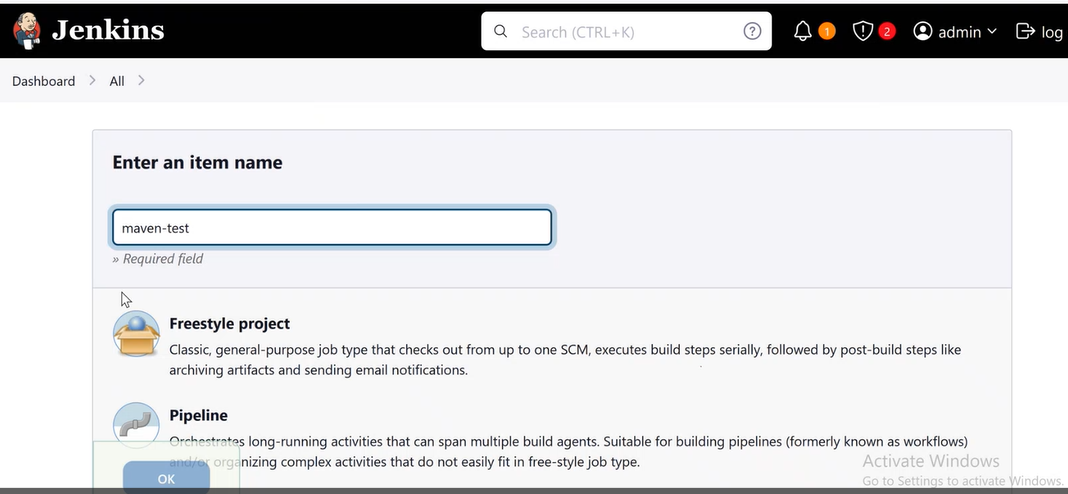


Build succeed

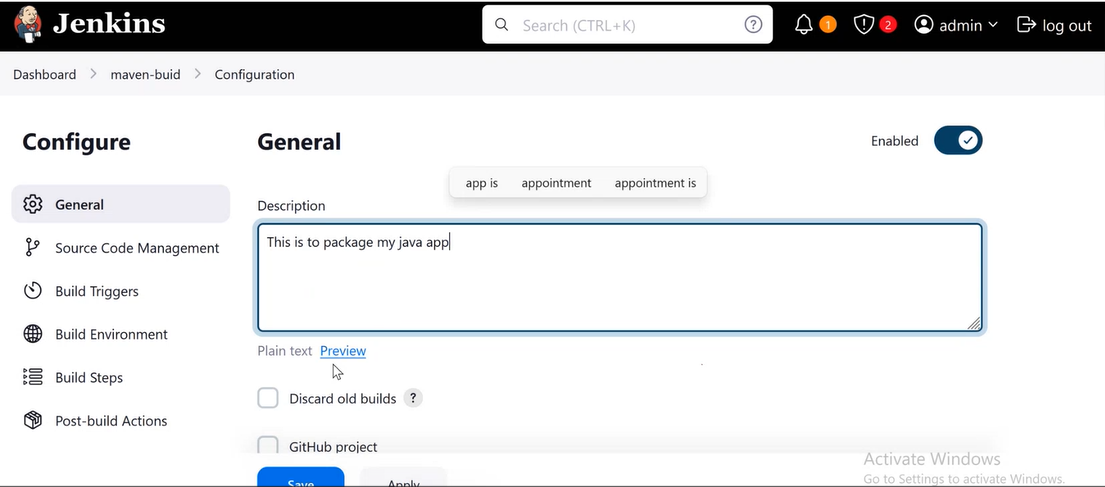


Check the workspace you will get .jar file there.

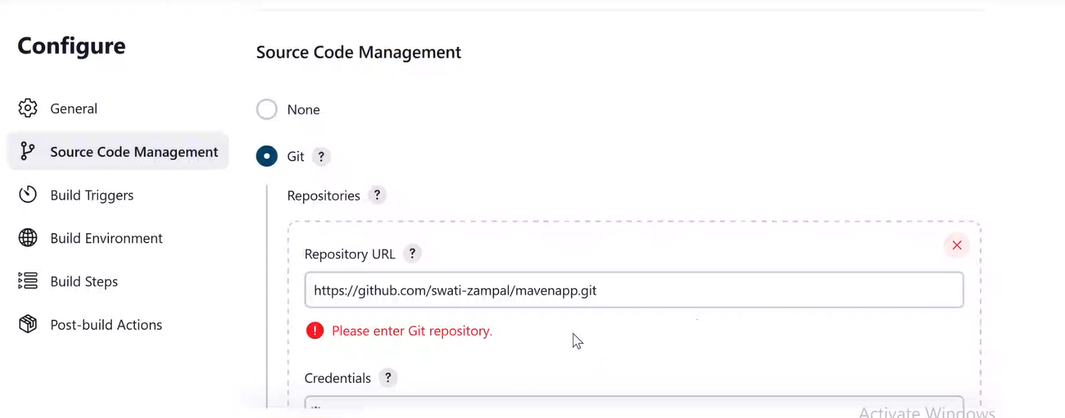
Project 2: maven-test



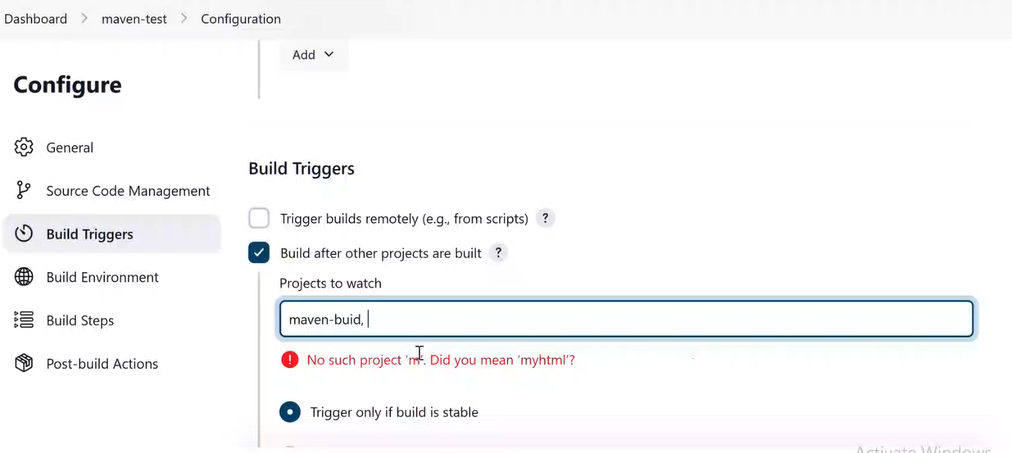
Step 1: General



Step 2: SCM



Step 3: Build Trigger

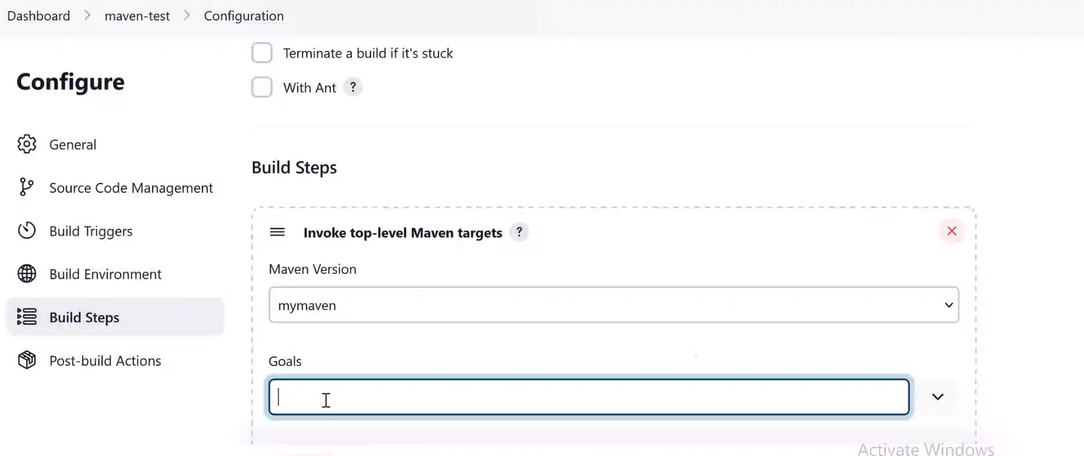


Step 4: Build Environment

(No changes)

Step 5: Build steps

Add build step à Invoke top-level maven targets



Save it and build your 1st project maven-build and check.

