.

# Lesson-End Project

# Sending Test Reports Using Email Notification

**Project agenda:** To configure Jenkins pipeline, which compiles code, execute JUnit

**Description**: Imagine a development team using Jenkins for automated email notifications of test reports, improving code quality and speed. Upon code commits, Jenkins triggers a pipeline that compiles, tests, and emails test results via an SMTP setup. This quick feedback loop is vital for agile practices, ensuring that developers immediately identify and address issues.

**Tools required:** Jenkins

**Prerequisites:** You must have Jenkins and GitHub access in the lab to proceed.

**Expected deliverables:** A fully configured Jenkins pipeline that compiles code, executes JUnit tests, and sends detailed test reports via email upon each build's completion

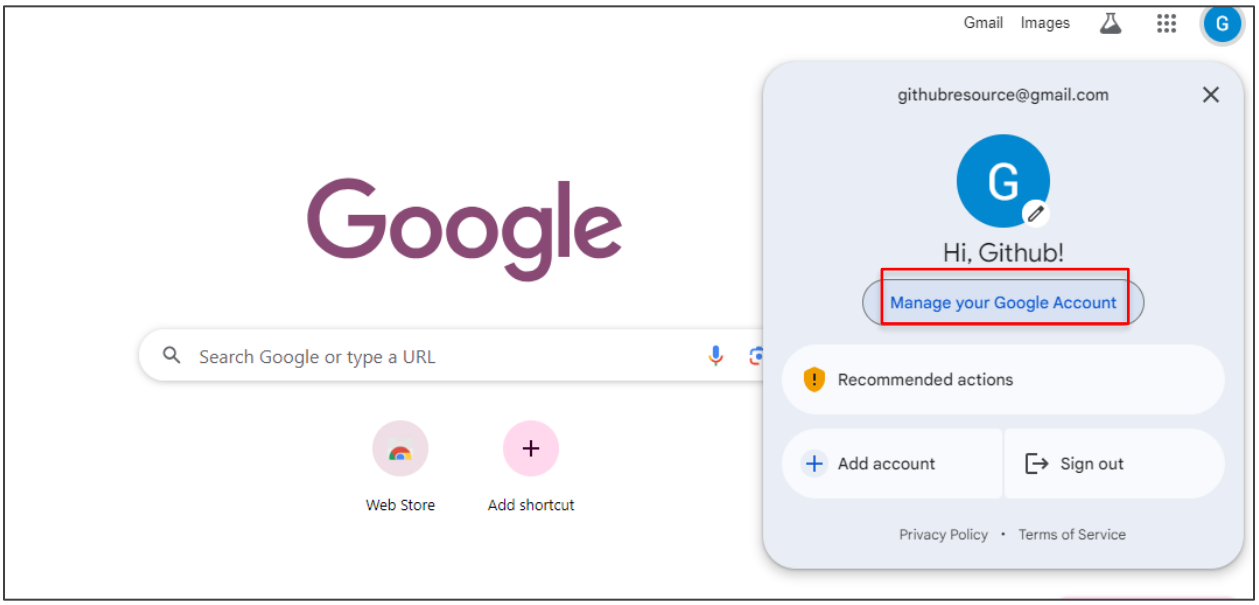
Steps to be followed:

1. Create an app password for SMTP configuration
2. Configure SMTP configurations in Jenkins
3. Configure Maven in Jenkins

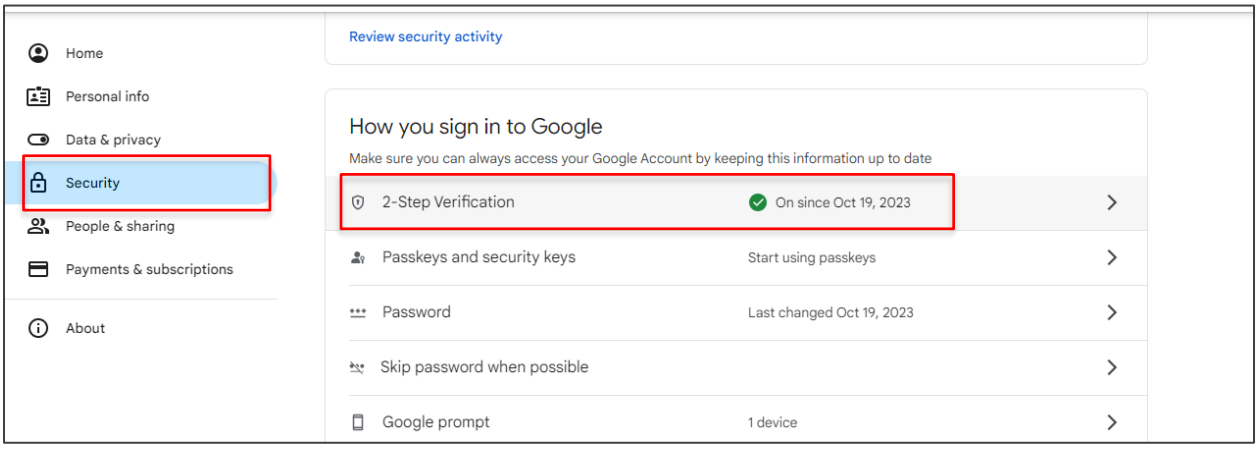
.

## Step 1: Create an app password for SMTP configuration

1.1 Navigate to your Google account and click on **Manage your Google Account**

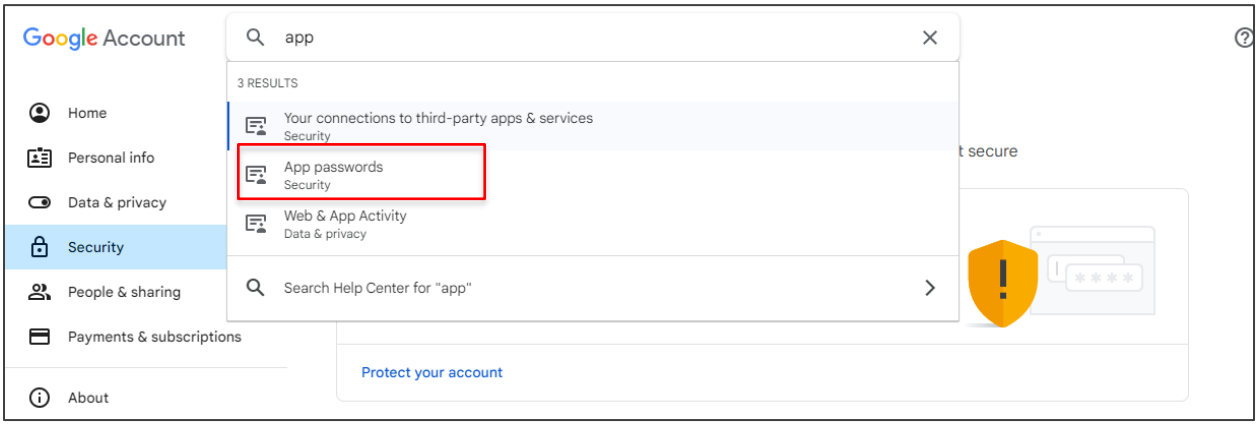



1.2 Now, navigate to **Security** and make sure you have **2-Step Verification** enabled

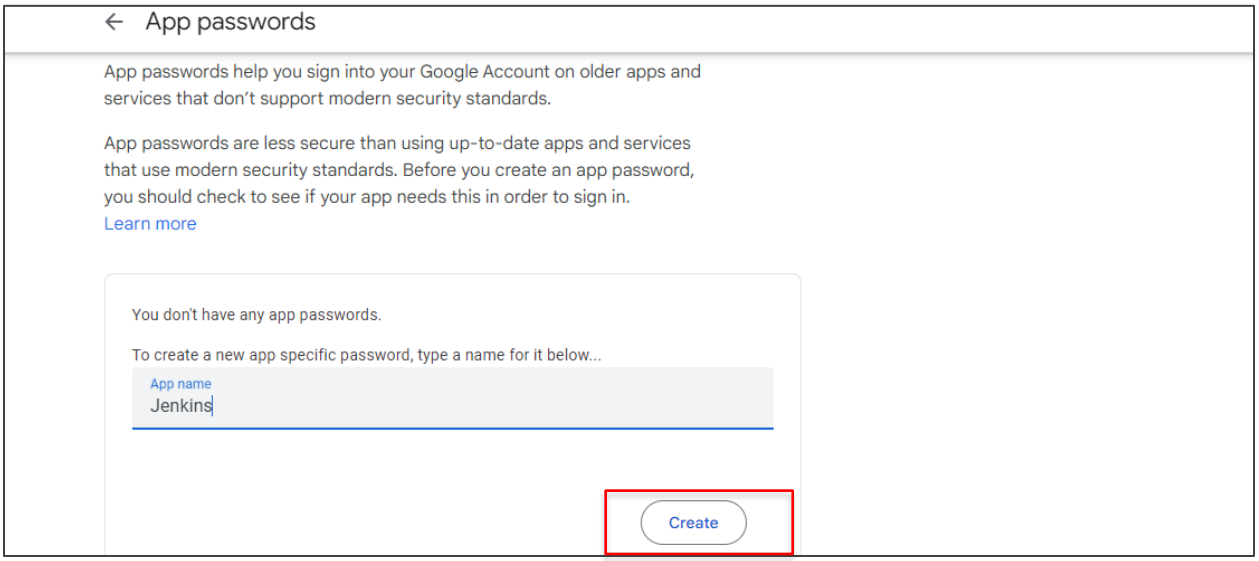


.

1.3 In **Security**, search for **App passwords** and click on it

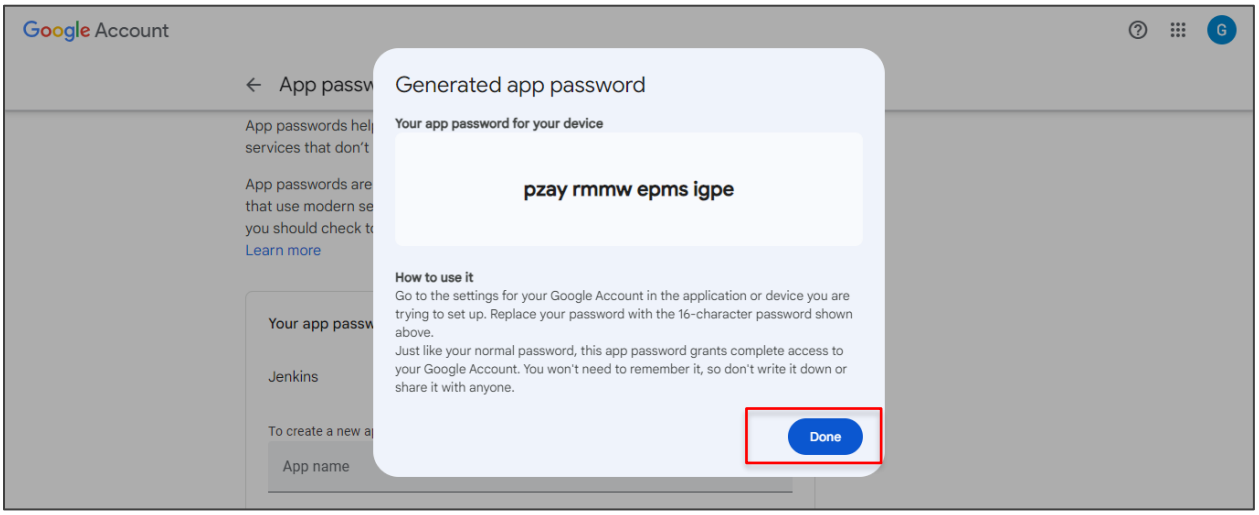



1.4 Provide the app name as **Jenkins** and click on **Create**



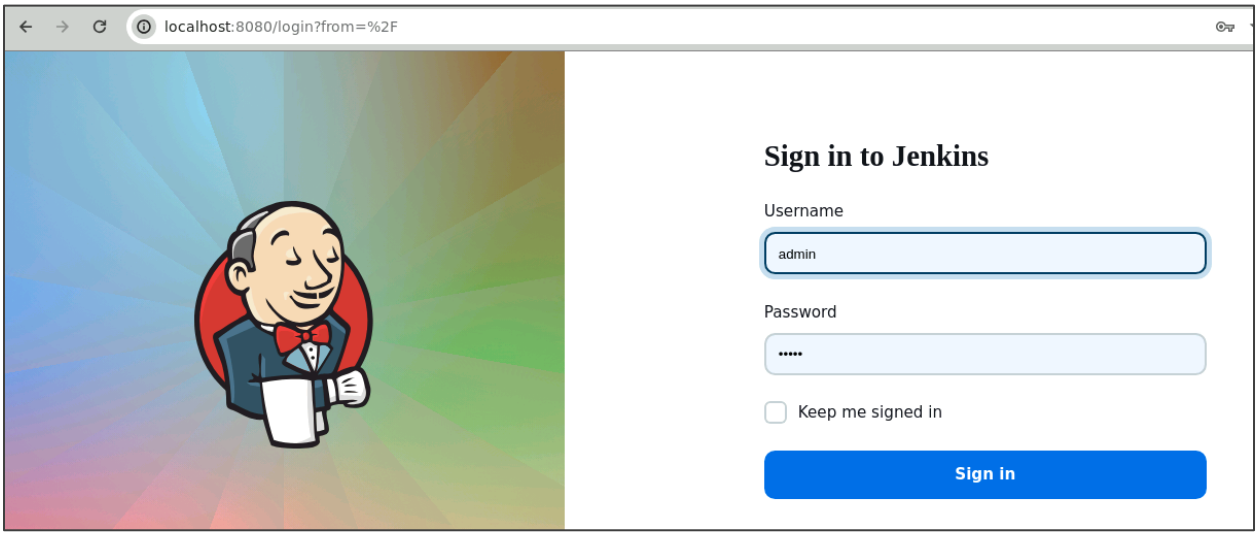
.

1.5 It will generate the app password. Copy the password and save it for future reference, then click on **Done**.




## Step 2: Configure SMTP configurations in Jenkins

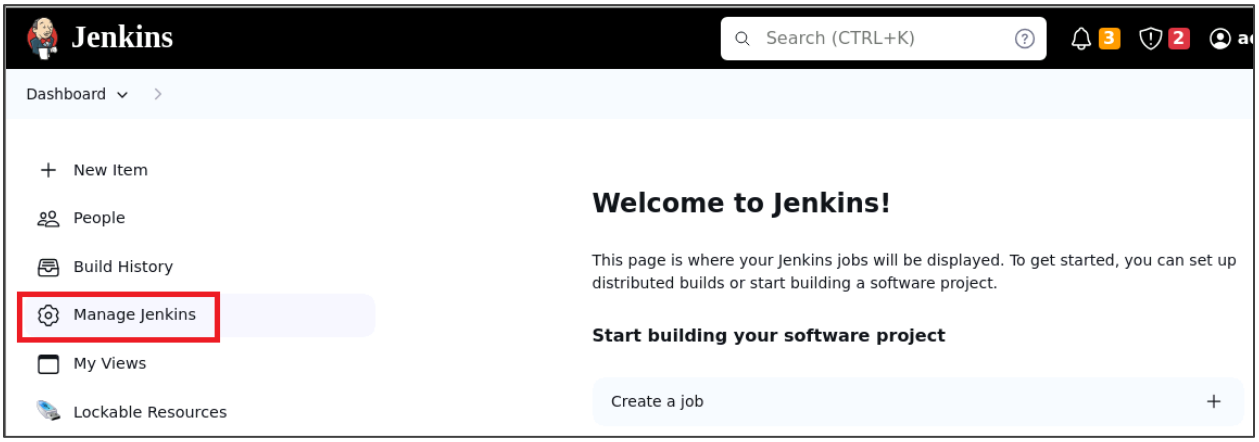
2.1 Sign in to **Jenkins** using your credentials



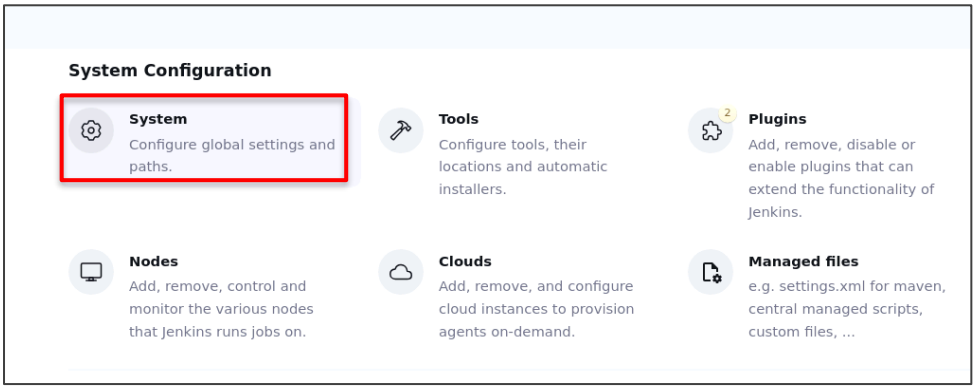

**Note:** The credentials for accessing Jenkins in the lab are Username: **admin** and Password: **admin**.

.

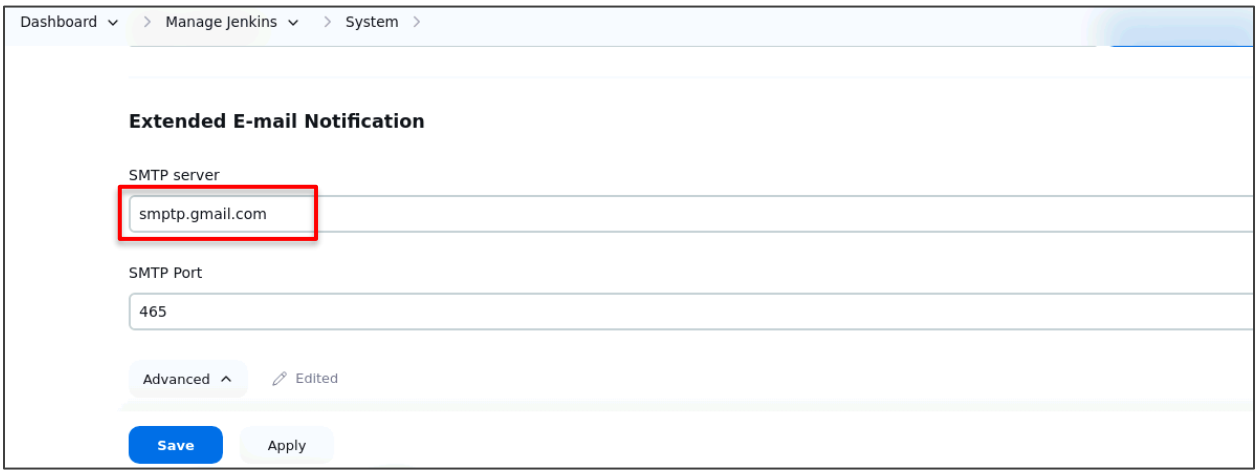
2.2 Navigate to **Manage Jenkins** in the Jenkins dashboard




2.3 Now, click on **System** under **System Configuration**

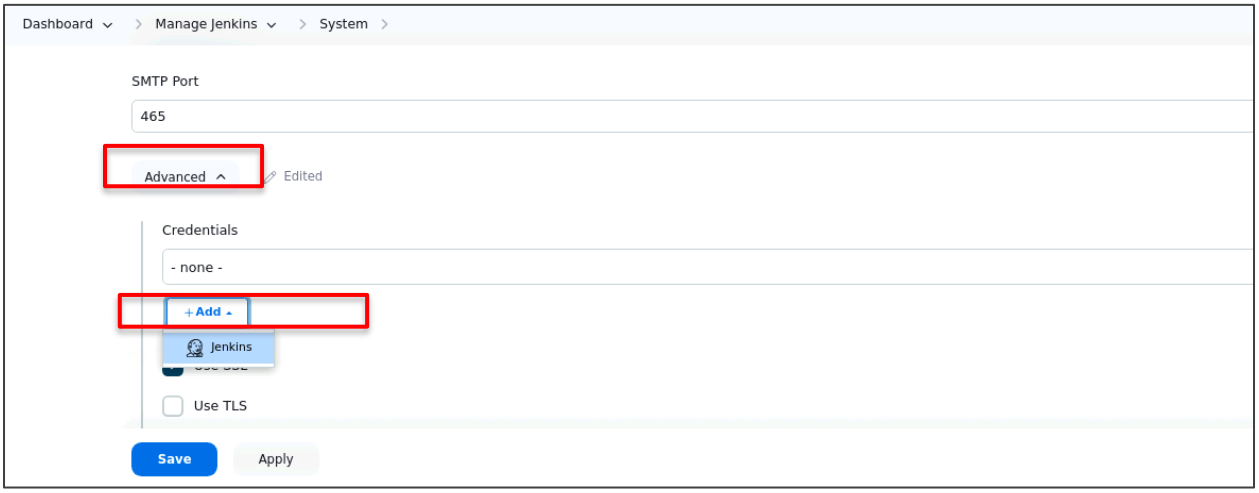



2.4 In the **System** section, scroll down to **Extend E-mail Notification** section and provide **smtp.gmail.com** as **the SMTP server** and **465** as **SMTP Port**

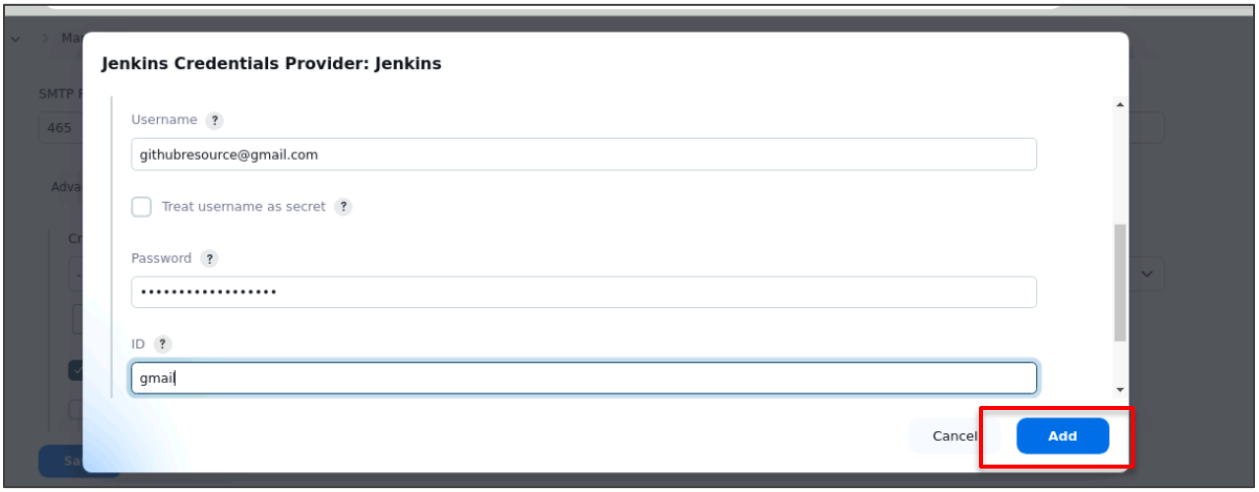


.

2.5 Click on **Advanced**, and in the **Credentials,** click on **Add** to add your credentials

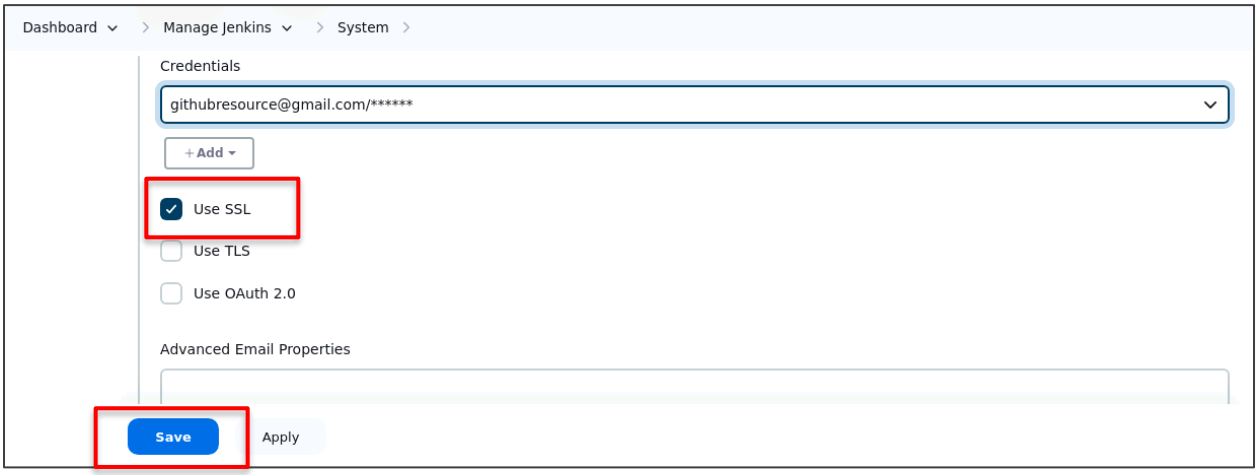



2.6 In the **Username,** provide your email ID; for **Password,** add the previously created app password, and then for **ID**, write **gmail** and click on **Add**



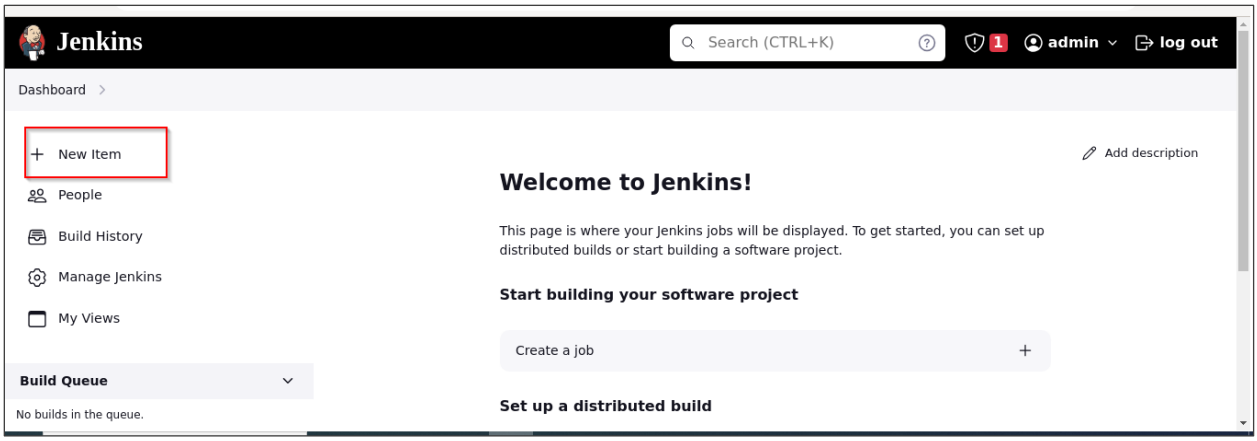
.

2.7 Select **Use SSL** and click on **Save** to save the configurations



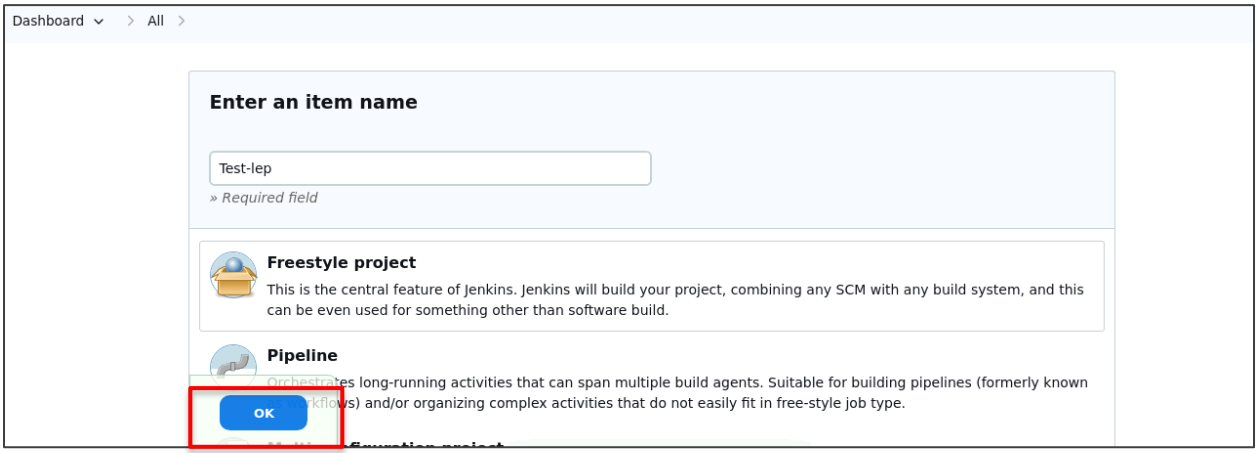

## Step 3: Configure Maven in Jenkins

3.1 Navigate back to Jenkins **Dashboard** and click on **New Item**



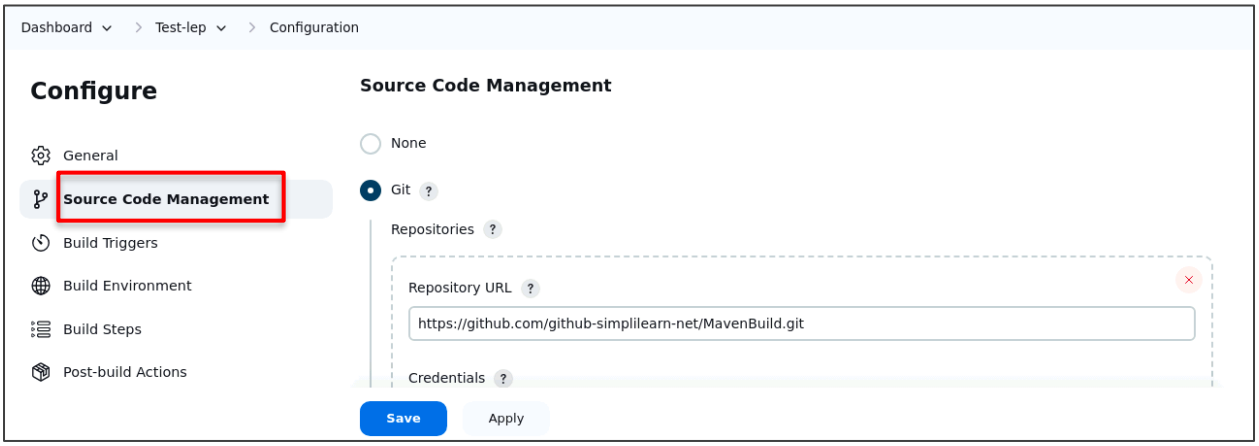
.

3.2 Enter the name of your project as **Test-lep**, select **Freestyle project**, then click on **OK**



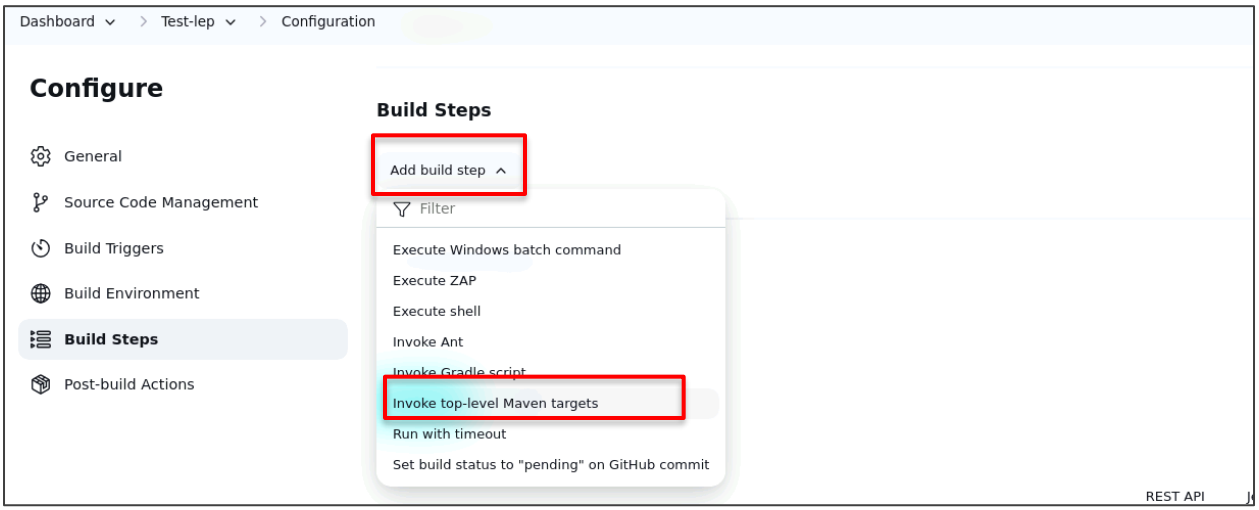

3.3 Navigate to **Source Code Management** and provide the below Git repository under **Repository URL**:

**https://github.com/github-simplilearn-net/MavenBuild.git**

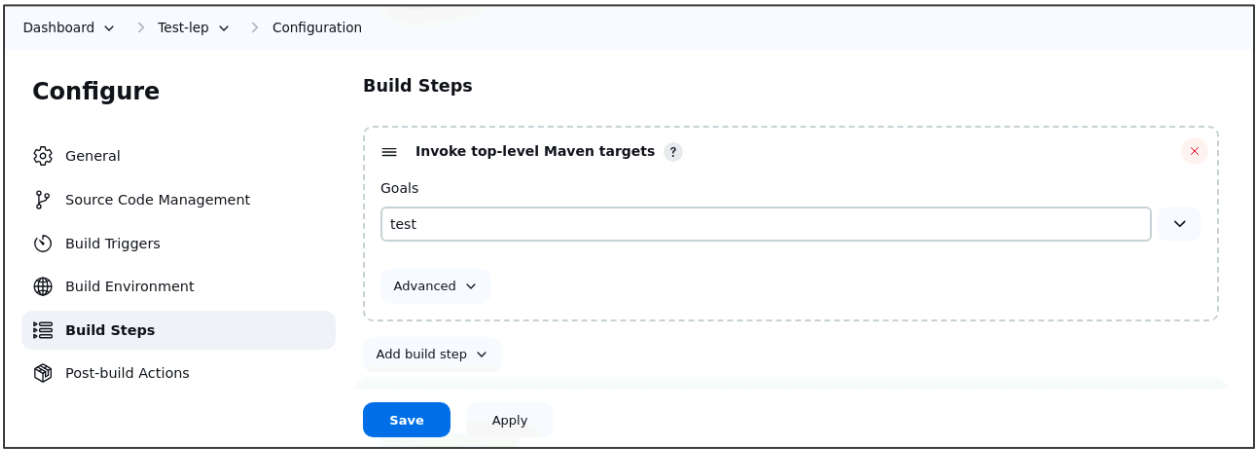


.

3.4 Navigate to **Build Steps**, click on **Add build step,** and select **Invoke top-level Maven targets**

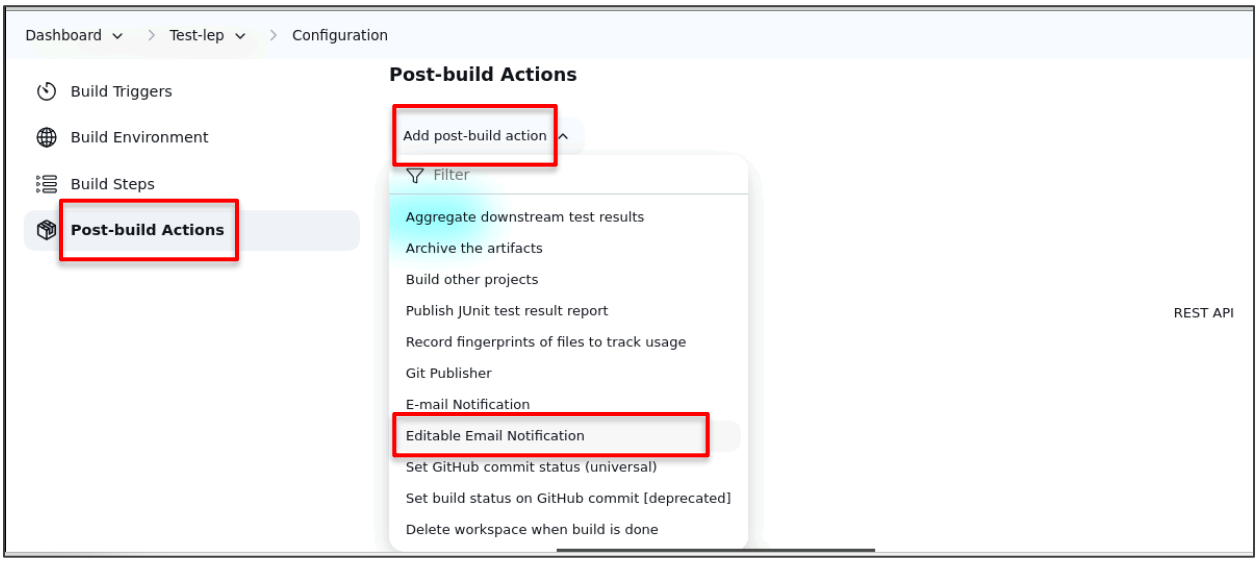



3.5 Write **test** in the **Goals**

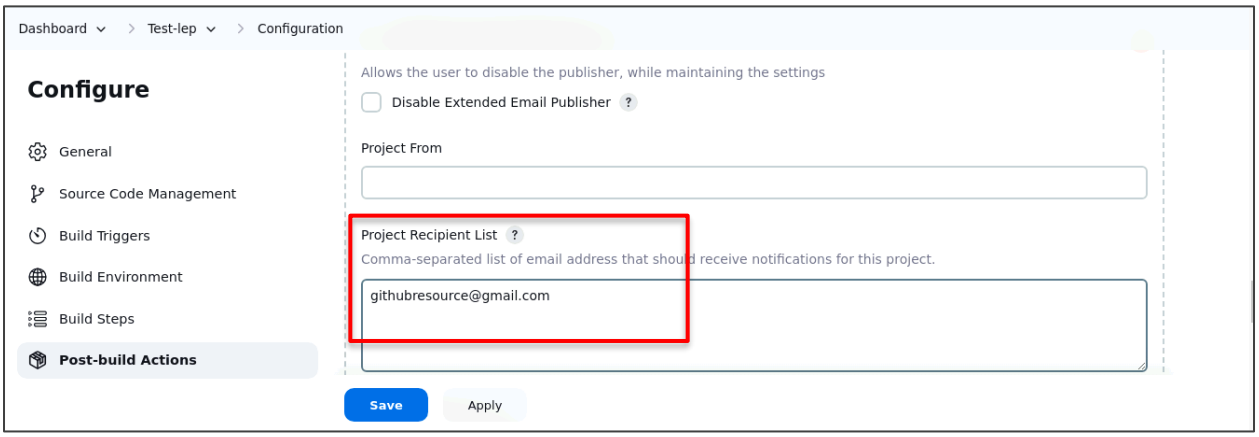


.

3.6 Navigate to **Post-build Actions**, click on **Add post-build action**, and select **Editable Email Notification**



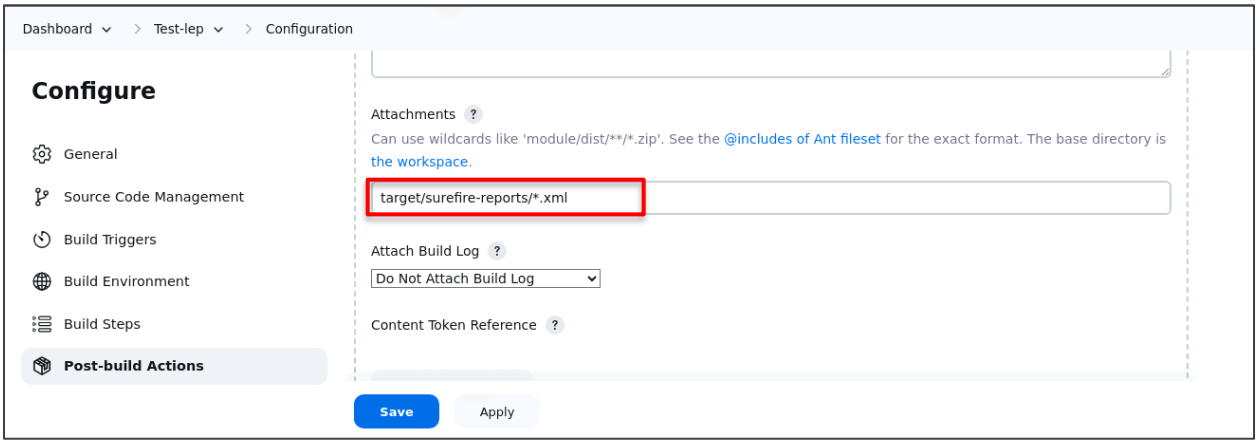

3.7 Under **Project Recipient List,** write your email address



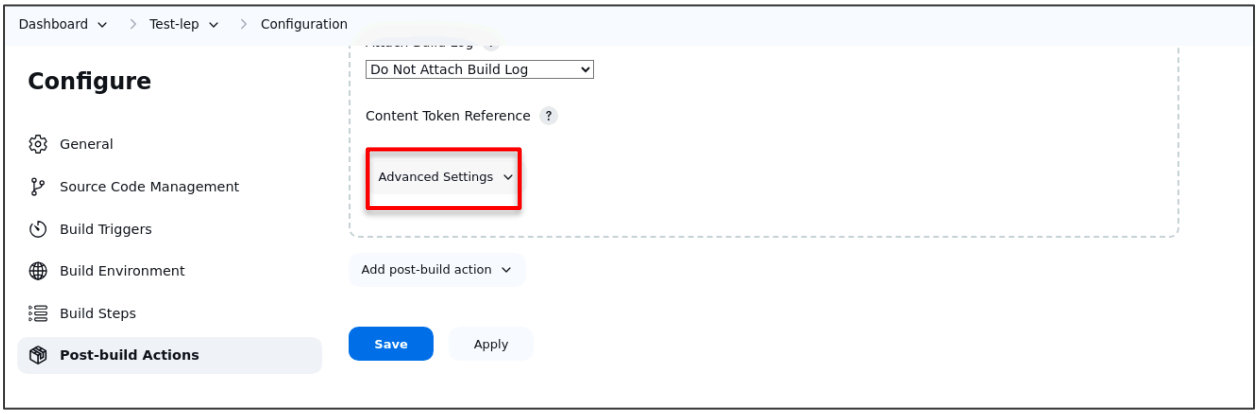
.

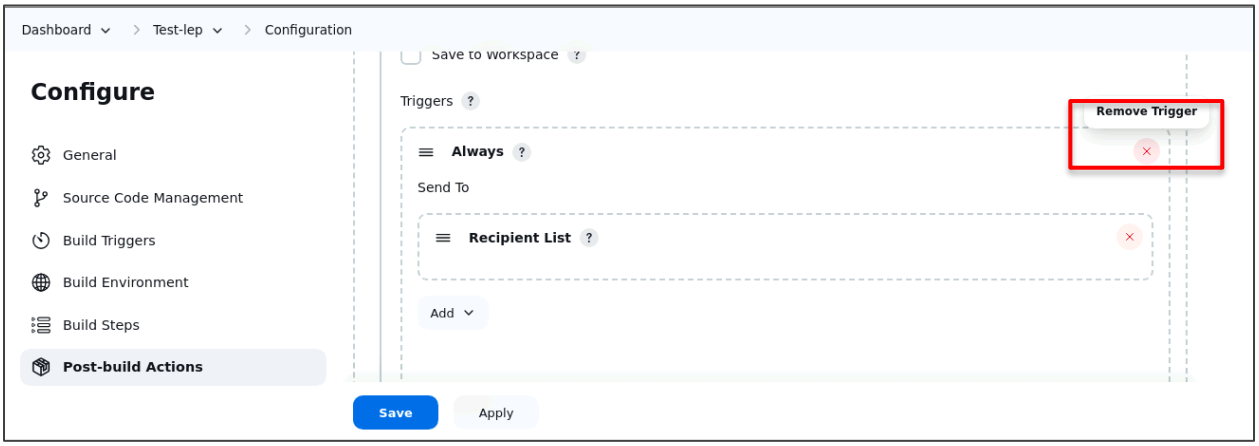
3.8 Scroll down to **Attachments** and provide the below details:

**target/surefire-reports/*.xml**



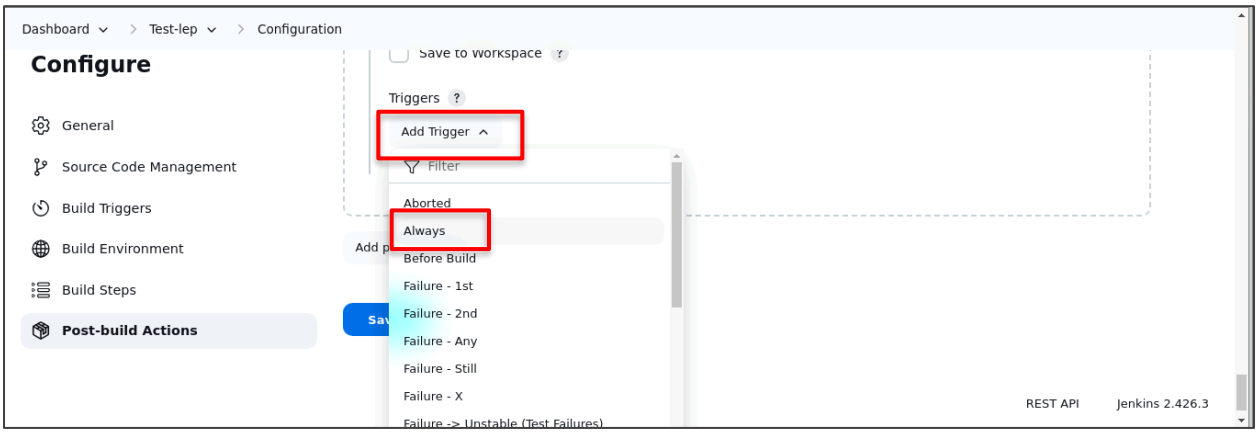

3.9 Click on **Advanced Settings**, scroll down to **Triggers,** and remove the existing trigger



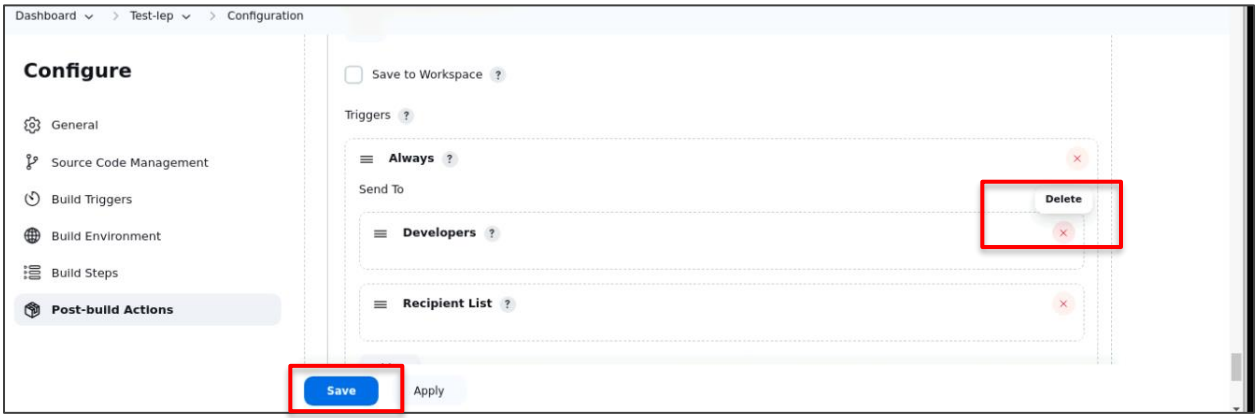



.

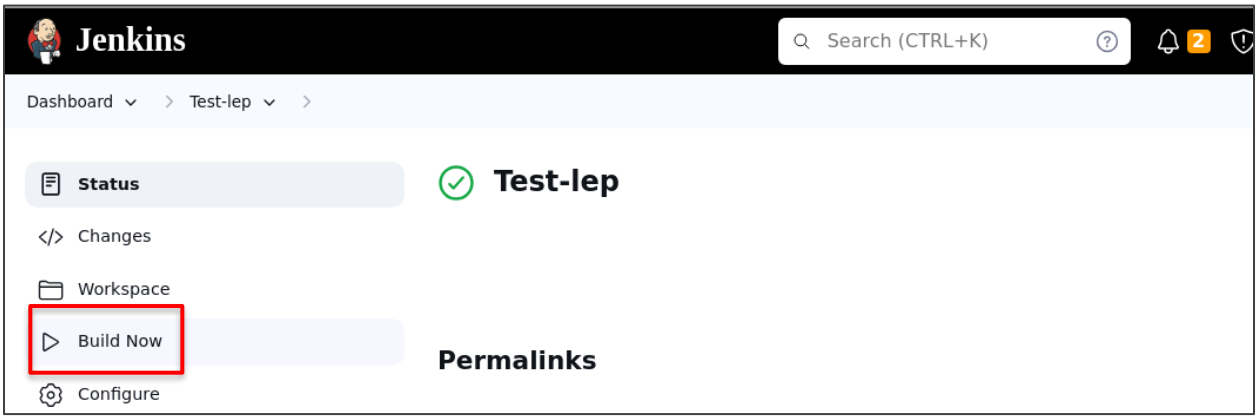
3.10 Click on **Add Trigger** and select **Always**




3.11 Delete the **Developers** and keep the **Recipient** list, then click on **Save**

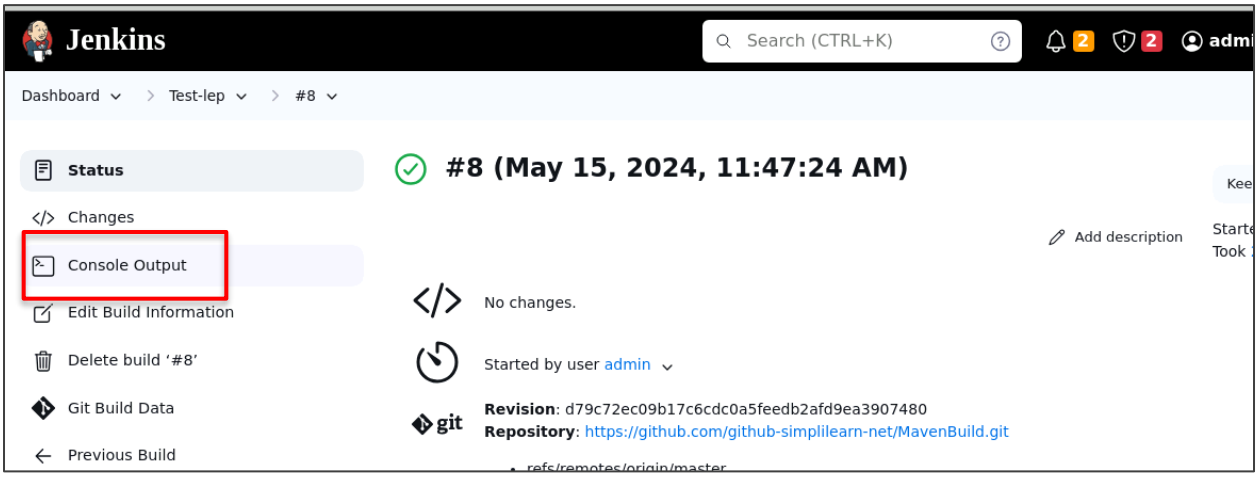


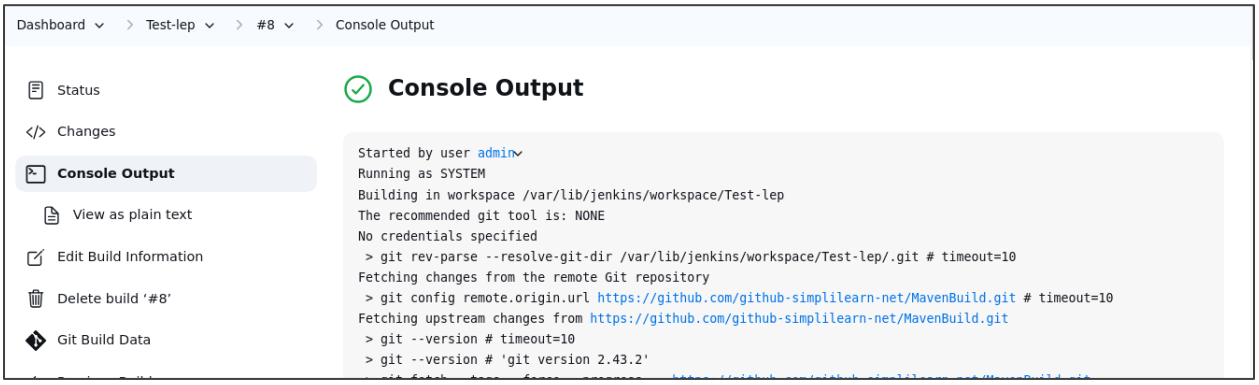

3.12 Now, click on **Build Now** to build the job



.

3.13 Once the job is built, click on **Console Output** to see the output



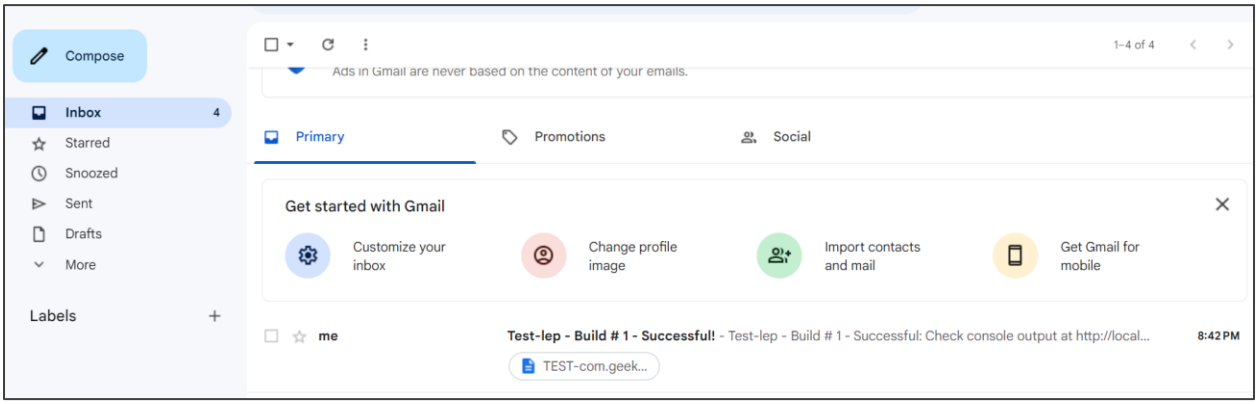



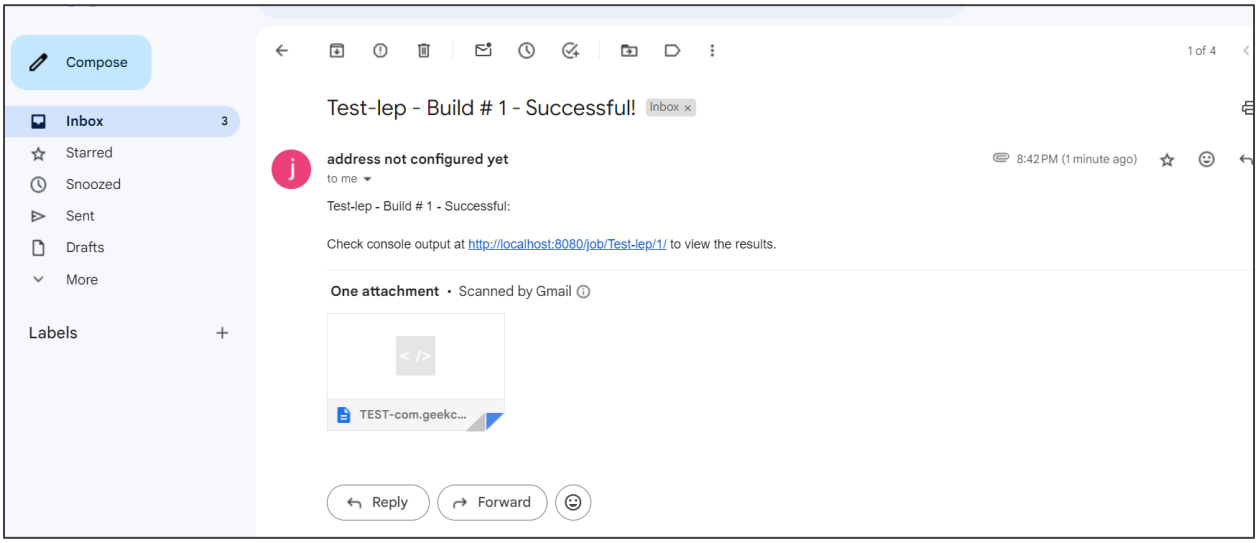


You can see the build is successful.

.

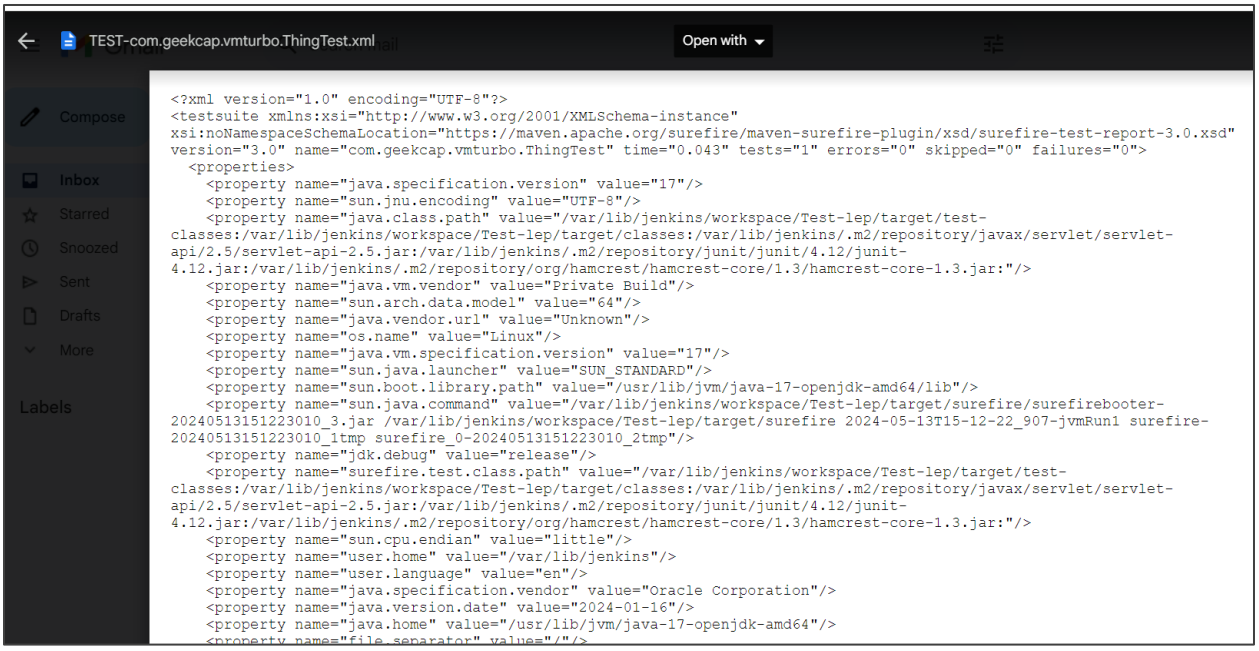
3.14 Navigate to your email account to validate if the email has been sent




You can see the email has been sent successfully.

.





```
<?xml version="1.0" encoding="UTF-8"?>
<testsuite xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="https://maven.apache.org/surefire/maven-surefire-plugin/xsd/surefire-test-report-3.0.xsd"
version="3.0" name="com.geekcap.vmturbo.ThingTest" time="0.043" tests="1" errors="0" skipped="0" failures="0">
  <properties>
    <property name="java.specification.version" value="17"/>
    <property name="sun.jnu.encoding" value="UTF-8"/>
    <property name="java.class.path" value="/var/lib/jenkins/workspace/Test-lep/target/test-
classes:/var/lib/jenkins/workspace/Test-lep/target/classes:/var/lib/jenkins/.m2/repository/javax/servlet/servlet-
api/2.5/servlet-api-2.5.jar:/var/lib/jenkins/.m2/repository/junit/junit/4.12/junit-
4.12.jar:/var/lib/jenkins/.m2/repository/org/hamcrest/hamcrest-core/1.3/hamcrest-core-1.3.jar:"/>
    <property name="java.vm.vendor" value="Private Build"/>
    <property name="sun.arch.data.model" value="64"/>
    <property name="java.vendor.url" value="Unknown"/>
    <property name="os.name" value="Linux"/>
    <property name="java.vm.specification.version" value="17"/>
    <property name="sun.java.launcher" value="SUN_STANDARD"/>
    <property name="sun.boot.library.path" value="/usr/lib/jvm/java-17-openjdk-amd64/lib"/>
    <property name="sun.java.command" value="/var/lib/jenkins/workspace/Test-lep/target/surefire/surefirebooter-
20240513151223010_3.jar /var/lib/jenkins/workspace/Test-lep/target/surefire 2024-05-13T15-12-22_907-jvmRun1 surefire-
20240513151223010_1tmp surefire_0-20240513151223010_2tmp"/>
    <property name="jdk.debug" value="release"/>
    <property name="surefire.test.class.path" value="/var/lib/jenkins/workspace/Test-lep/target/test-
classes:/var/lib/jenkins/workspace/Test-lep/target/classes:/var/lib/jenkins/.m2/repository/javax/servlet/servlet-
api/2.5/servlet-api-2.5.jar:/var/lib/jenkins/.m2/repository/junit/junit/4.12/junit-
4.12.jar:/var/lib/jenkins/.m2/repository/org/hamcrest/hamcrest-core/1.3/hamcrest-core-1.3.jar:"/>
    <property name="sun.cpu.endian" value="little"/>
    <property name="user.home" value="/var/lib/jenkins"/>
    <property name="user.language" value="en"/>
    <property name="java.specification.vendor" value="Oracle Corporation"/>
    <property name="java.version.date" value="2024-01-16"/>
    <property name="java.home" value="/usr/lib/jvm/java-17-openjdk-amd64"/>
    <property name="file.separator" value="/"/>
```


You can see the Junit report.

By following these steps, you have successfully set up Maven in Jenkins to run JUnit test cases and automatically distribute the test reports via email notifications to specified recipients after each build.