PRACTICAL 3.1

Aim: Continuous Integration with Jenkins.

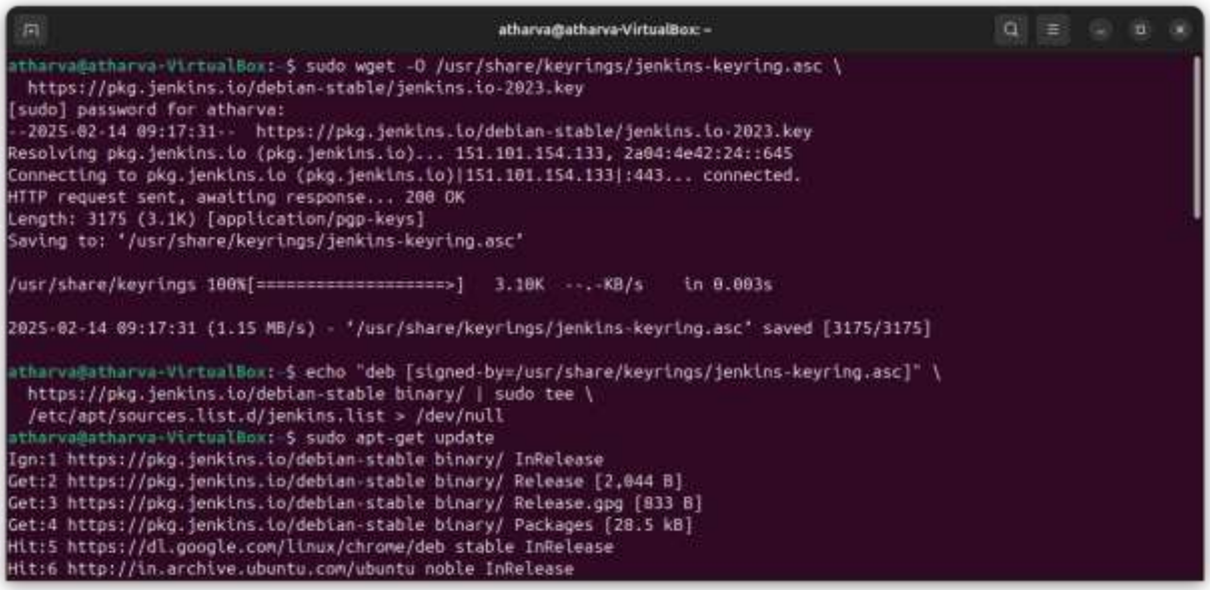
Objectives:
● The objective of Jenkins is to automate the building, testing, and deployment of software projects.
● To understand creation of Jobs

Tools Used: Virtual box manager, Ubuntu, Jenkins

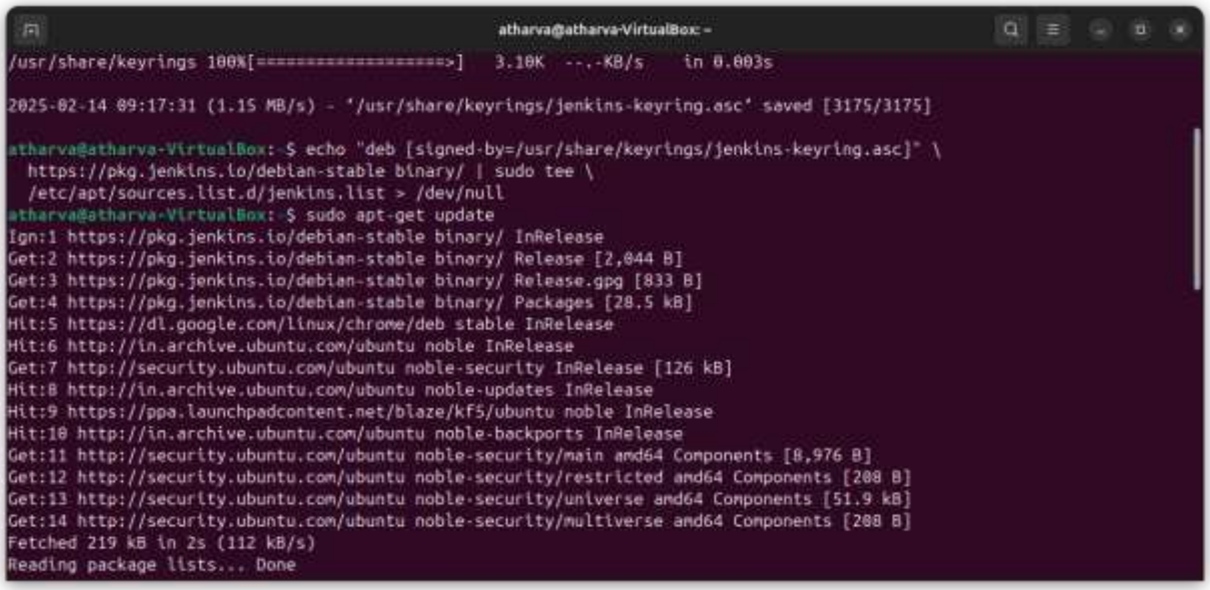
Concepts:
● Compile: This stage involves compiling your source code into executable code or binaries. This will generate a class files for java program
● Code Review: This stage involves reviewing the code changes made by developers before deploying them.Jenkins can trigger external tools such as PMD plugin that automates code reviews by scanning for errors and enforcing coding standards, enhancing code quality before deployment.
● Unit Test: This stage involves running automated unit tests to ensure that individual components or units of code function correctly in isolation. Unit tests are crucial for verifying the behavior of small sections of code and catching bugs early in the development process.
● Metrics Check: This stage involves analyzing code quality metrics and performing static code analysis to ensure adherence to coding standards, identify potential bugs, and maintain code maintainability. Tools like Jacoco can be integrated into Jenkins pipelines to perform these checks.
● Package: This stage involves packaging your application or project into a deployable artifact. Depending on the type of application, packaging could involve creating a WAR file

Problem Statement on Jenkins :
You are working as a DevOps Engineer in a company named Sanders & Fresco Pvt ltd. You have
been asked by your manager to create a Maven Project using Jenkins and build a war file of that
project. As a proof of concept, you have been given a web application to build.
Steps to solve:
● Open Jenkins and create a Maven project using it.
● You will have to create the following jobs,which are as follow:
1. Compile
2. Code Review
3. Unit test
4. Package
5. Metric Check

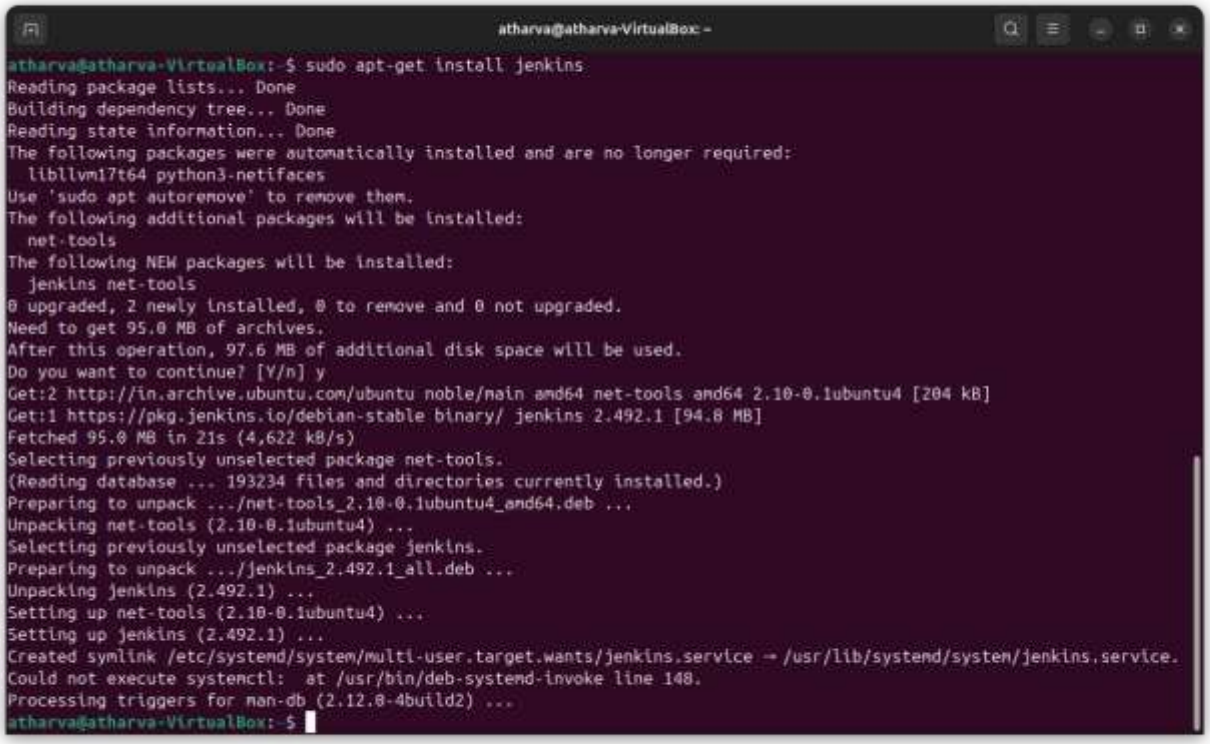
Installation of jenkins



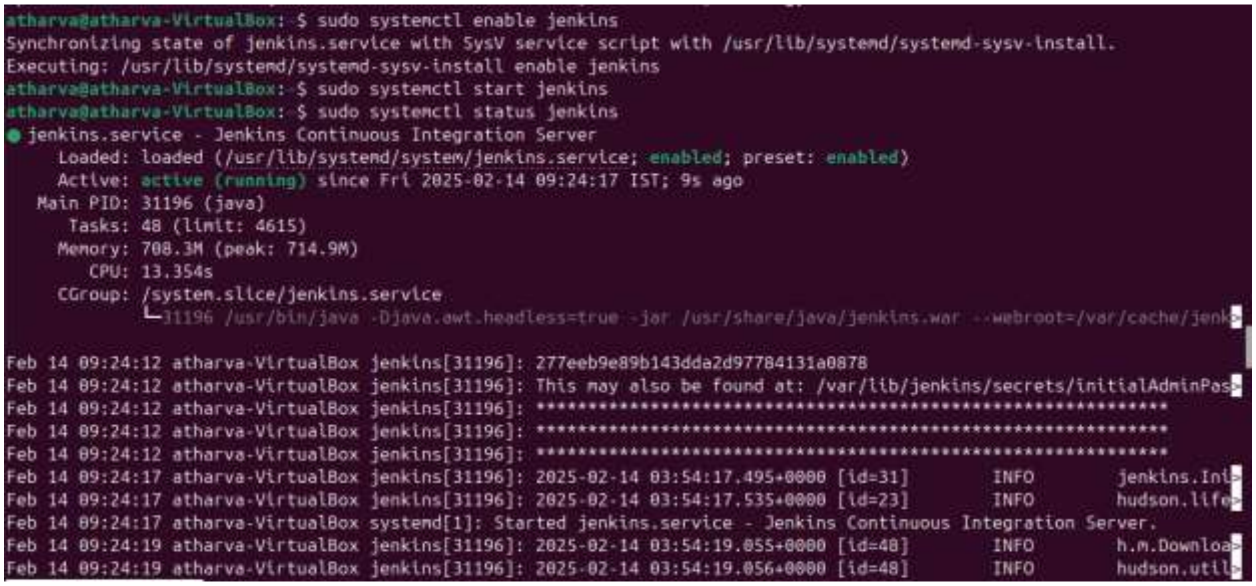
```
atharva@atharva-VirtualBox:~$ sudo wget -O /usr/share/keyrings/jenkins-keyring.asc \
  https://pkg.jenkins.io/debian-stable/jenkins.io-2023.key
[sudo] password for atharva:
--2025-02-14 09:17:31--  https://pkg.jenkins.io/debian-stable/jenkins.io-2023.key
Resolving pkg.jenkins.io (pkg.jenkins.io)... 151.101.154.133, 2a04:4e42:24::645
Connecting to pkg.jenkins.io (pkg.jenkins.io)|151.101.154.133|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 3175 (3.1K) [application/pgp-keys]
Saving to: '/usr/share/keyrings/jenkins-keyring.asc'

/usr/share/keyrings 100%[===================>]   3.10K  --.-KB/s    in 0.003s

2025-02-14 09:17:31 (1.15 MB/s) - '/usr/share/keyrings/jenkins-keyring.asc' saved [3175/3175]

atharva@atharva-VirtualBox:~$ echo "deb [signed-by=/usr/share/keyrings/jenkins-keyring.asc]" \
  https://pkg.jenkins.io/debian-stable binary/ | sudo tee \
  /etc/apt/sources.list.d/jenkins.list > /dev/null
atharva@atharva-VirtualBox:~$ sudo apt-get update
Ign:1 https://pkg.jenkins.io/debian-stable binary/ InRelease
Get:2 https://pkg.jenkins.io/debian-stable binary/ Release [2,044 B]
Get:3 https://pkg.jenkins.io/debian-stable binary/ Release.gpg [833 B]
Get:4 https://pkg.jenkins.io/debian-stable binary/ Packages [28.5 kB]
Hit:5 https://dl.google.com/linux/chrome/deb stable InRelease
Hit:6 http://in.archive.ubuntu.com/ubuntu noble InRelease
```



```
/usr/share/keyrings 100%[===================>]   3.10K  --.-KB/s    in 0.003s

2025-02-14 09:17:31 (1.15 MB/s) - '/usr/share/keyrings/jenkins-keyring.asc' saved [3175/3175]

atharva@atharva-VirtualBox:~$ echo "deb [signed-by=/usr/share/keyrings/jenkins-keyring.asc]" \
  https://pkg.jenkins.io/debian-stable binary/ | sudo tee \
  /etc/apt/sources.list.d/jenkins.list > /dev/null
atharva@atharva-VirtualBox:~$ sudo apt-get update
Ign:1 https://pkg.jenkins.io/debian-stable binary/ InRelease
Get:2 https://pkg.jenkins.io/debian-stable binary/ Release [2,044 B]
Get:3 https://pkg.jenkins.io/debian-stable binary/ Release.gpg [833 B]
Get:4 https://pkg.jenkins.io/debian-stable binary/ Packages [28.5 kB]
Hit:5 https://dl.google.com/linux/chrome/deb stable InRelease
Hit:6 http://in.archive.ubuntu.com/ubuntu noble InRelease
Get:7 http://security.ubuntu.com/ubuntu noble-security InRelease [126 kB]
Hit:8 http://in.archive.ubuntu.com/ubuntu noble-updates InRelease
Hit:9 https://ppa.launchpadcontent.net/blaze/kf5/ubuntu noble InRelease
Hit:10 http://in.archive.ubuntu.com/ubuntu noble-backports InRelease
Get:11 http://security.ubuntu.com/ubuntu noble-security/main amd64 Components [8,976 B]
Get:12 http://security.ubuntu.com/ubuntu noble-security/restricted amd64 Components [208 B]
Get:13 http://security.ubuntu.com/ubuntu noble-security/universe amd64 Components [51.9 kB]
Get:14 http://security.ubuntu.com/ubuntu noble-security/multiverse amd64 Components [208 B]
Fetched 219 kB in 2s (112 kB/s)
Reading package lists... Done
```



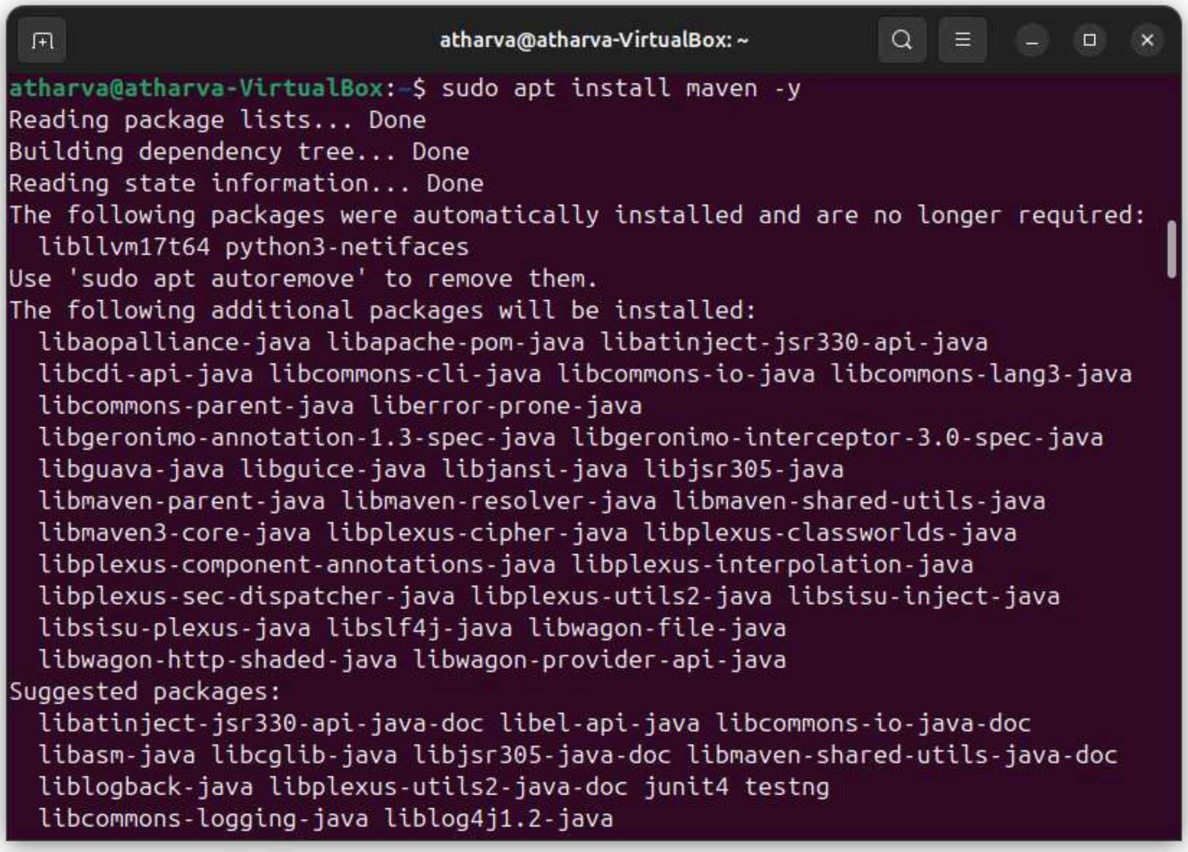
```
atharva@atharva-VirtualBox:~$ sudo apt-get install jenkins
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following packages were automatically installed and are no longer required:
  libllvm17t64 python3-netifaces
Use 'sudo apt autoremove' to remove them.
The following additional packages will be installed:
  net-tools
The following NEW packages will be installed:
  jenkins net-tools
0 upgraded, 2 newly installed, 0 to remove and 0 not upgraded.
Need to get 95.0 MB of archives.
After this operation, 97.6 MB of additional disk space will be used.
Do you want to continue? [Y/n] y
Get:2 http://in.archive.ubuntu.com/ubuntu noble/main amd64 net-tools amd64 2.10-0.1ubuntu4 [204 kB]
Get:1 https://pkg.jenkins.io/debian-stable binary/ jenkins 2.492.1 [94.8 MB]
Fetched 95.0 MB in 21s (4,622 kB/s)
Selecting previously unselected package net-tools.
(Reading database ... 193234 files and directories currently installed.)
Preparing to unpack .../net-tools_2.10-0.1ubuntu4_amd64.deb ...
Unpacking net-tools (2.10-0.1ubuntu4) ...
Selecting previously unselected package jenkins.
Preparing to unpack .../jenkins_2.492.1_all.deb ...
Unpacking jenkins (2.492.1) ...
Setting up net-tools (2.10-0.1ubuntu4) ...
Setting up jenkins (2.492.1) ...
Created symlink /etc/systemd/system/multi-user.target.wants/jenkins.service → /usr/lib/systemd/system/jenkins.service.
Could not execute systemctl:  at /usr/bin/deb-systemd-invoke line 148.
Processing triggers for man-db (2.12.0-4build2) ...
atharva@atharva-VirtualBox:~$ 
```



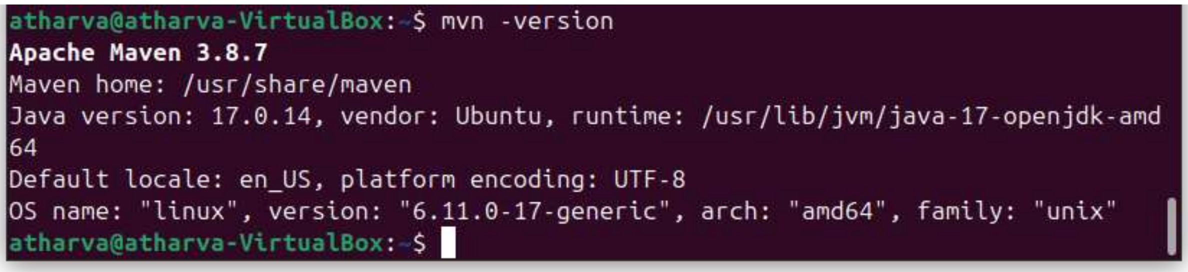
```
atharva@atharva-VirtualBox:~$ sudo systemctl enable jenkins
Synchronizing state of jenkins.service with SysV service script with /usr/lib/systemd/systemd-sysv-install.
Executing: /usr/lib/systemd/systemd-sysv-install enable jenkins
atharva@atharva-VirtualBox:~$ sudo systemctl start jenkins
atharva@atharva-VirtualBox:~$ sudo systemctl status jenkins
● jenkins.service - Jenkins Continuous Integration Server
     Loaded: loaded (/usr/lib/systemd/system/jenkins.service; enabled; preset: enabled)
     Active: active (running) since Fri 2025-02-14 09:24:17 IST; 9s ago
   Main PID: 31196 (java)
      Tasks: 48 (limit: 4615)
     Memory: 708.3M (peak: 714.9M)
        CPU: 13.354s
     CGroup: /system.slice/jenkins.service
             └─31196 /usr/bin/java -Djava.awt.headless=true -jar /usr/share/java/jenkins.war --webroot=/var/cache/jenk>

Feb 14 09:24:12 atharva-VirtualBox jenkins[31196]: 277eeb9e89b143dda2d97784131a0878
Feb 14 09:24:12 atharva-VirtualBox jenkins[31196]: This may also be found at: /var/lib/jenkins/secrets/initialAdminPas>
Feb 14 09:24:12 atharva-VirtualBox jenkins[31196]: *************************************************************
Feb 14 09:24:12 atharva-VirtualBox jenkins[31196]: *************************************************************
Feb 14 09:24:12 atharva-VirtualBox jenkins[31196]: *************************************************************
Feb 14 09:24:17 atharva-VirtualBox jenkins[31196]: 2025-02-14 03:54:17.495+0000 [id=31]        INFO        jenkins.Ini>
Feb 14 09:24:17 atharva-VirtualBox jenkins[31196]: 2025-02-14 03:54:17.535+0000 [id=23]        INFO        hudson.life>
Feb 14 09:24:17 atharva-VirtualBox systemd[1]: Started jenkins.service - Jenkins Continuous Integration Server.
Feb 14 09:24:19 atharva-VirtualBox jenkins[31196]: 2025-02-14 03:54:19.055+0000 [id=48]        INFO        h.m.Downloa>
Feb 14 09:24:19 atharva-VirtualBox jenkins[31196]: 2025-02-14 03:54:19.056+0000 [id=48]        INFO        hudson.util>
```

Start jenkins
sudo systemctl enable jenkins
sudo systemctl start jenkins
sudo systemctl status jenkins

Also make sure to install maven



```
atharva@atharva-VirtualBox:~$ sudo apt install maven -y
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following packages were automatically installed and are no longer required:
  libllvm17t64 python3-netifaces
Use 'sudo apt autoremove' to remove them.
The following additional packages will be installed:
  libaopalliance-java libapache-pom-java libatinject-jsr330-api-java
  libcdi-api-java libcommons-cli-java libcommons-io-java libcommons-lang3-java
  libcommons-parent-java liberror-prone-java
  libgeronimo-annotation-1.3-spec-java libgeronimo-interceptor-3.0-spec-java
  libguava-java libguice-java libjansi-java libjsr305-java
  libmaven-parent-java libmaven-resolver-java libmaven-shared-utils-java
  libmaven3-core-java libplexus-cipher-java libplexus-classworlds-java
  libplexus-component-annotations-java libplexus-interpolation-java
  libplexus-sec-dispatcher-java libplexus-utils2-java libsisu-inject-java
  libsisu-plexus-java libslf4j-java libwagon-file-java
  libwagon-http-shaded-java libwagon-provider-api-java
Suggested packages:
  libatinject-jsr330-api-java-doc libel-api-java libcommons-io-java-doc
  libasm-java libcglib-java libjsr305-java-doc libmaven-shared-utils-java-doc
  liblogback-java libplexus-utils2-java-doc junit4 testng
  libcommons-logging-java liblog4j1.2-java
```

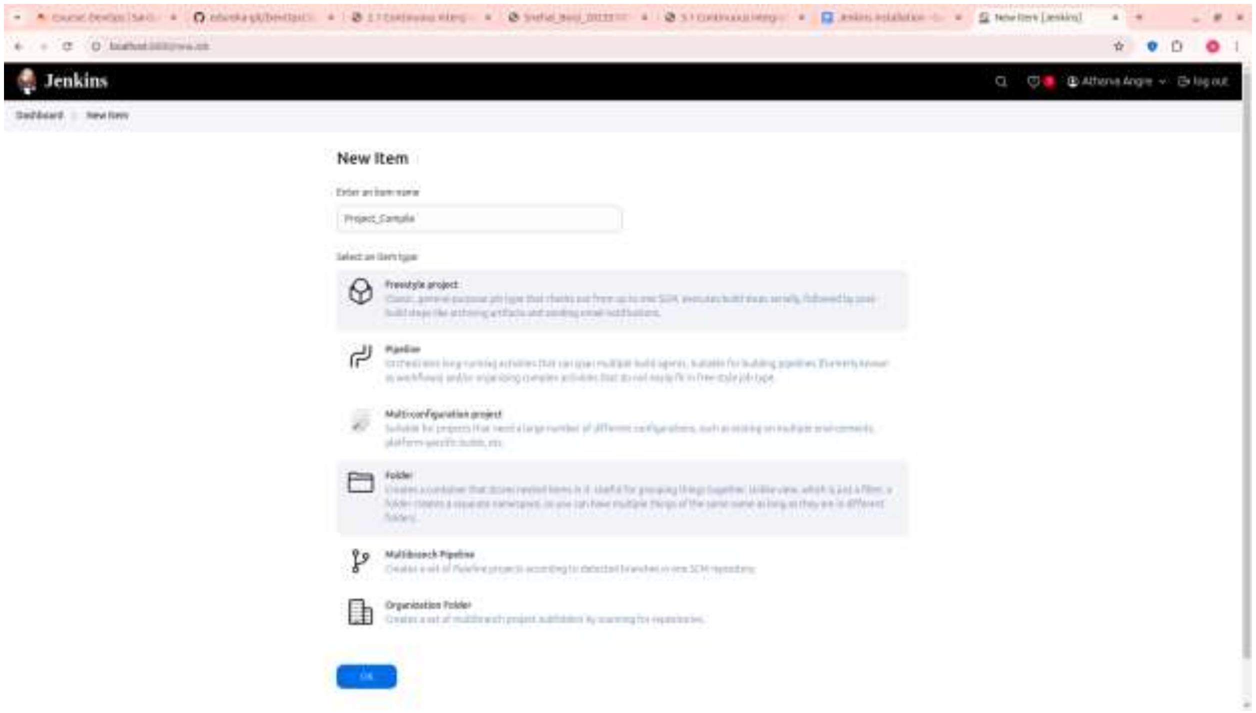


```
atharva@atharva-VirtualBox:~$ mvn -version
Apache Maven 3.8.7
Maven home: /usr/share/maven
Java version: 17.0.14, vendor: Ubuntu, runtime: /usr/lib/jvm/java-17-openjdk-amd
64
Default locale: en_US, platform encoding: UTF-8
OS name: "linux", version: "6.11.0-17-generic", arch: "amd64", family: "unix"
atharva@atharva-VirtualBox:~$ 
```

On browser go to http://localhost:8080/
And Sign in with the credentials

Compile

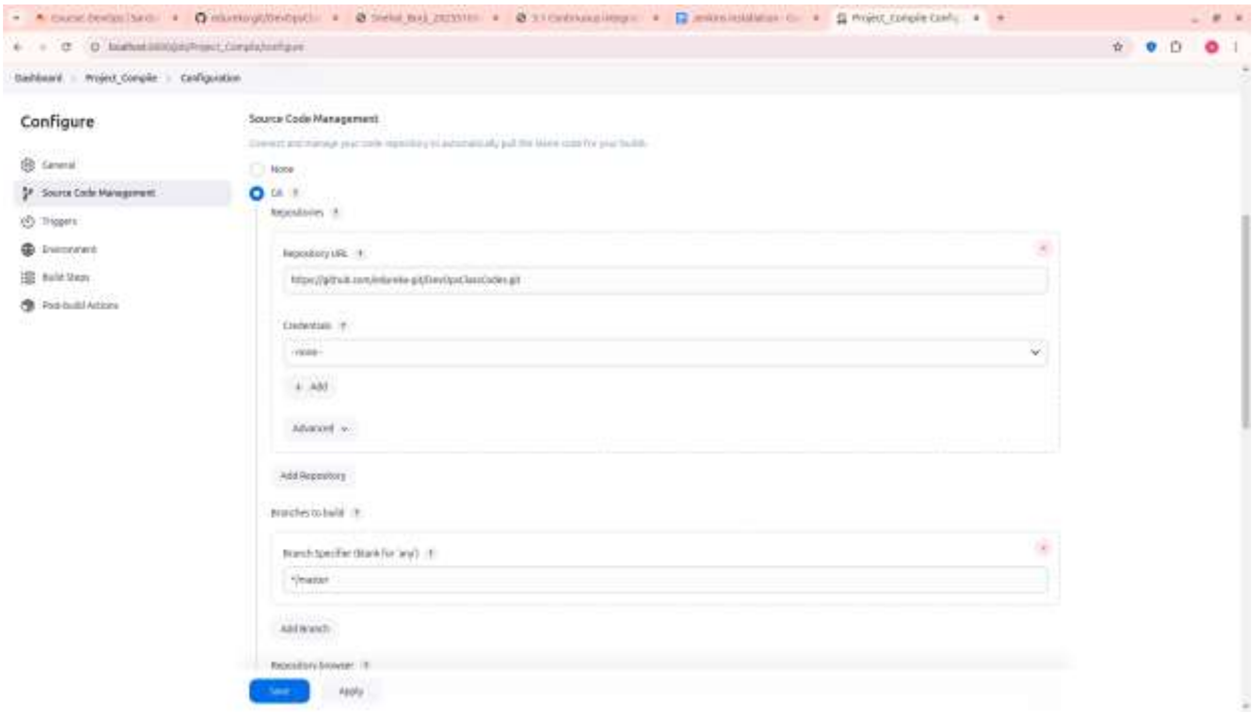
1. Created a new job as Compile and connected github repository of sample project



New Item

Enter an item name: Project_Compile

Select an item type: Freestyle project



Dashboard > Project_Compile > Configuration

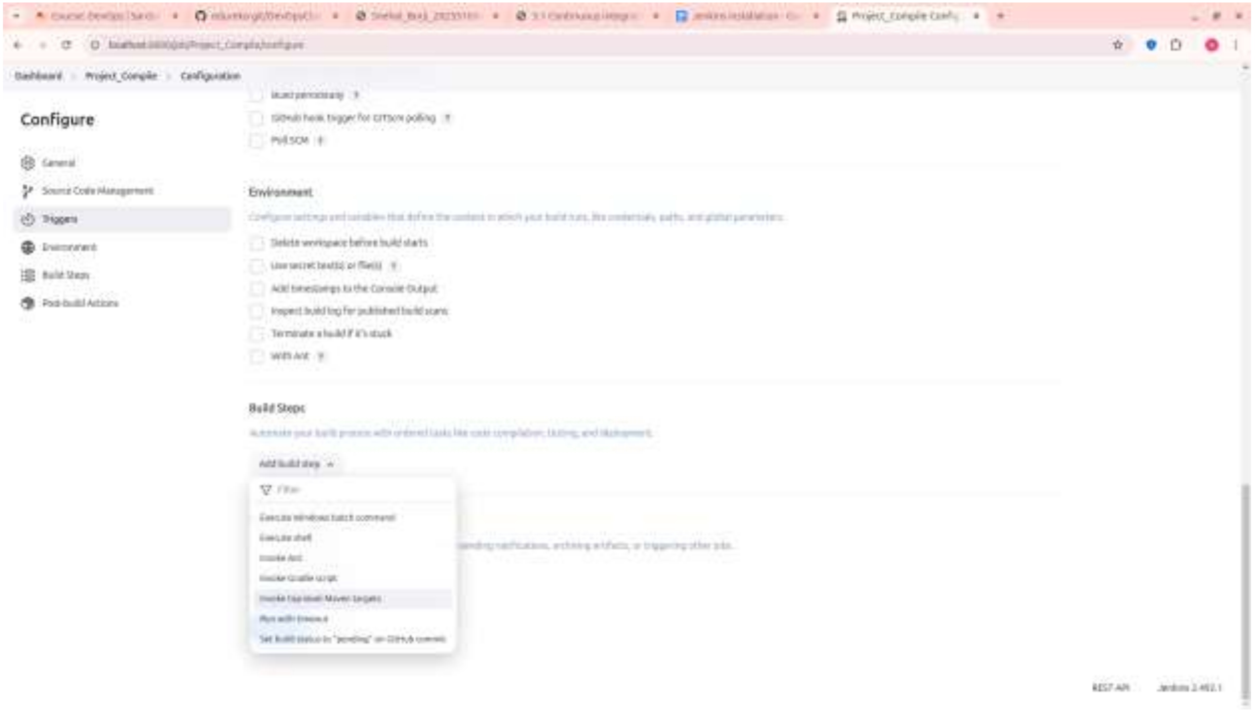
Source Code Management: Git

Repository URL: https://github.com/edureka-git/DevOpsClassCodes.git

Credentials: - none -

Branch Specifier (blank for 'any'): */master

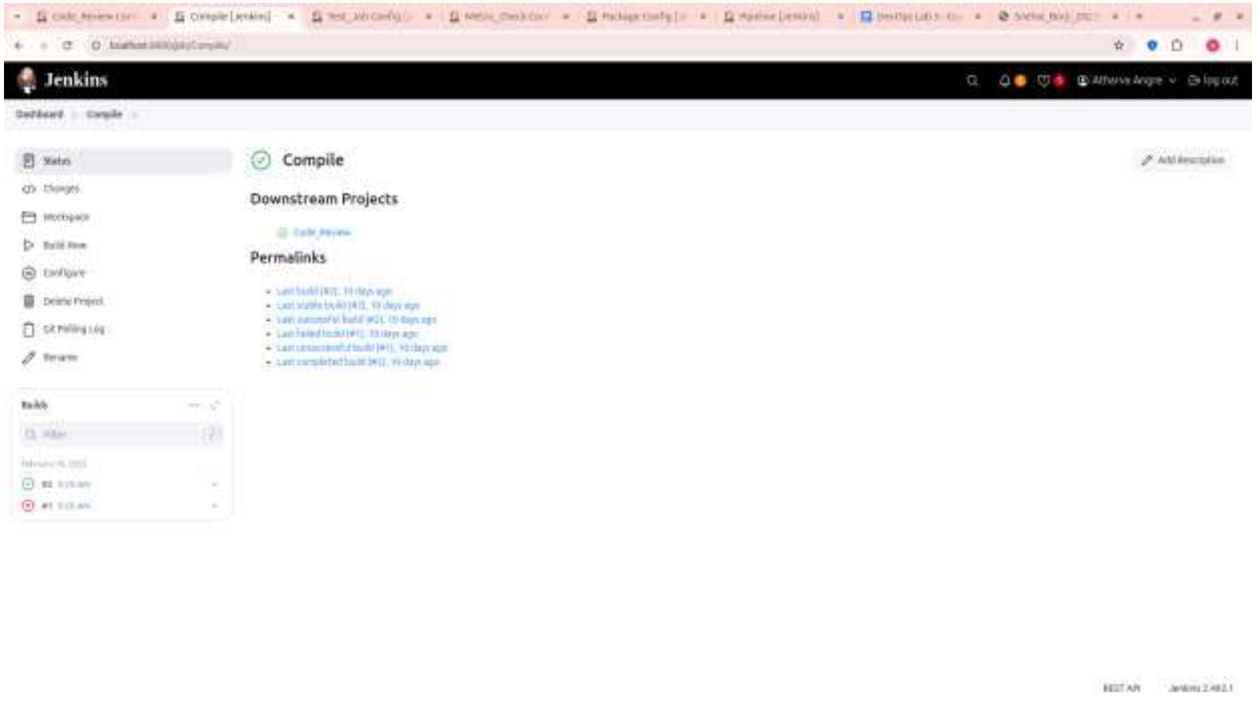
2. In build steps, selected a Invoke top-level Maven target and in it selected mymaven and in goals selected a compile plugin .


Dashboard
Project_Compile
Configuration
Configure
General
Source Code Management
Triggers
Environment
Build Steps
Post-build Actions
Build periodically
GitHub hook trigger for GITScm polling
Poll SCM
Environment
Delete workspace before build starts
Use secret text(s) or file(s)
Add timestamps to the Console Output
Inspect build log for published build scans
Terminate a build if it's stuck
With Ant
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
REST API
Jenkins 2.492.1

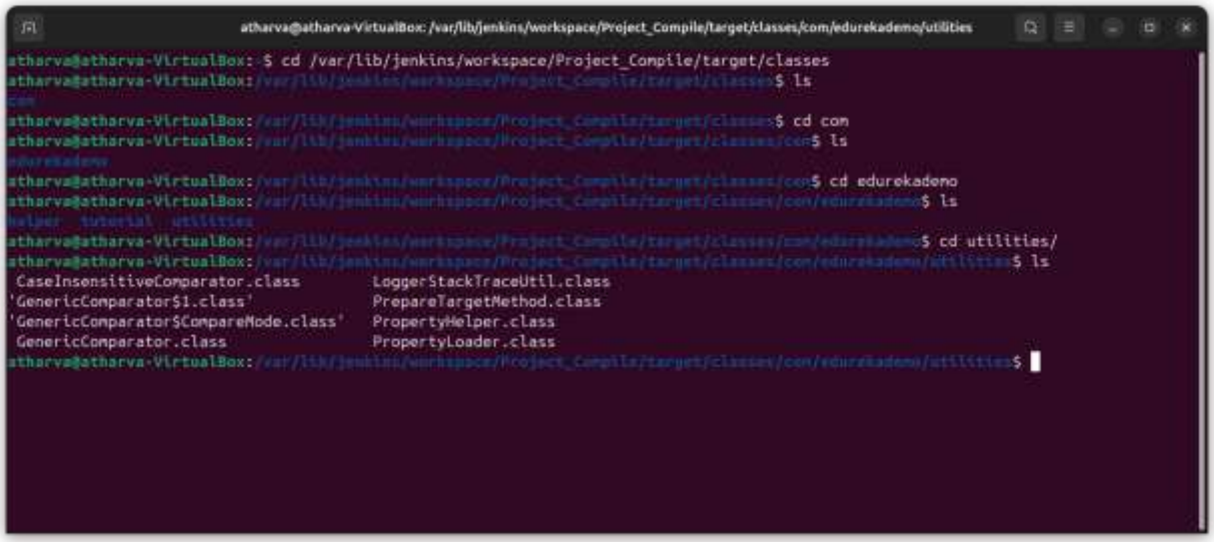
In the Goal type compile

Then Click save Button

It will run successfully



To check it
Copy the path which is as below
Compiling 13 source files to /var/lib/jenkins/workspace/Project_Compile/target/classes

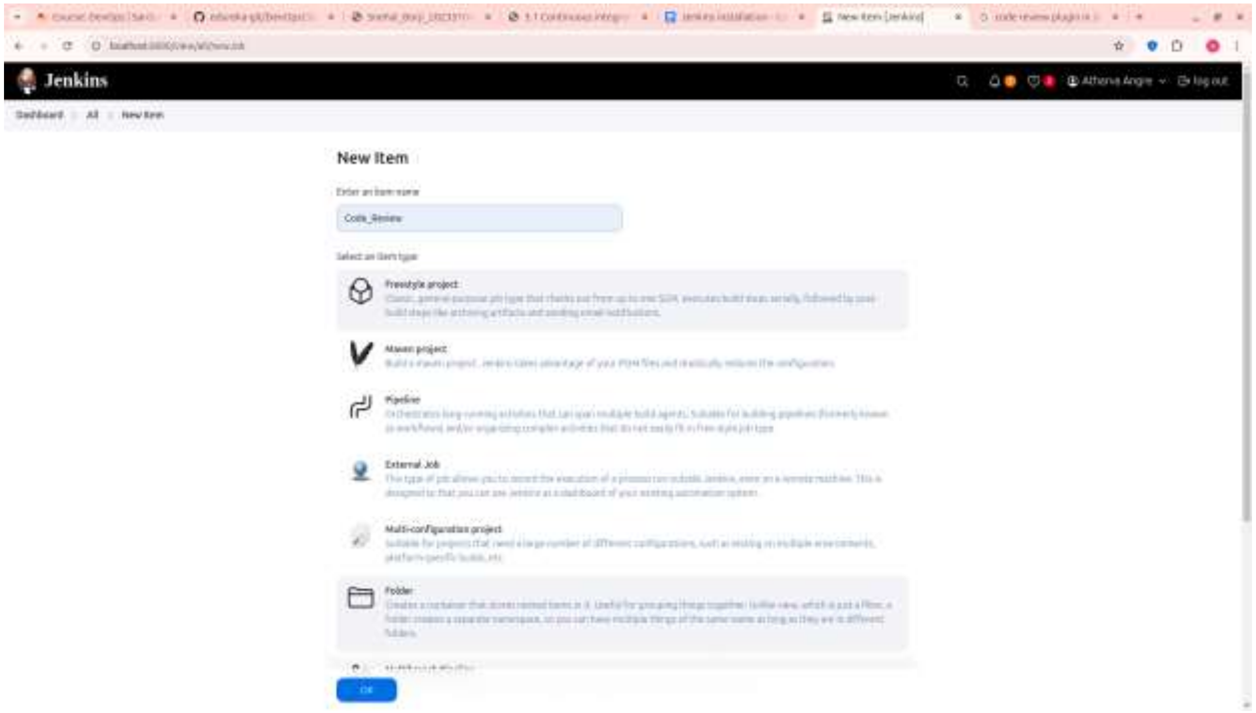


atharva@atharva-VirtualBox: /var/lib/jenkins/workspace/Project_Compile/target/classes/com/edurekademo/utilities

```
atharva@atharva-VirtualBox:~$ cd /var/lib/jenkins/workspace/Project_Compile/target/classes
atharva@atharva-VirtualBox:/var/lib/jenkins/workspace/Project_Compile/target/classes$ ls
com
atharva@atharva-VirtualBox:/var/lib/jenkins/workspace/Project_Compile/target/classes$ cd com
atharva@atharva-VirtualBox:/var/lib/jenkins/workspace/Project_Compile/target/classes/com$ ls
edurekademo
atharva@atharva-VirtualBox:/var/lib/jenkins/workspace/Project_Compile/target/classes/com$ cd edurekademo
atharva@atharva-VirtualBox:/var/lib/jenkins/workspace/Project_Compile/target/classes/com/edurekademo$ ls
helper  tutorial  utilities
atharva@atharva-VirtualBox:/var/lib/jenkins/workspace/Project_Compile/target/classes/com/edurekademo$ cd utilities/
atharva@atharva-VirtualBox:/var/lib/jenkins/workspace/Project_Compile/target/classes/com/edurekademo/utilities$ ls
 CaseInsensitiveComparator.class        LoggerStackTraceUtil.class
'GenericComparator$1.class'             PrepareTargetMethod.class
'GenericComparator$CompareMode.class'   PropertyHelper.class
 GenericComparator.class                PropertyLoader.class
atharva@atharva-VirtualBox:/var/lib/jenkins/workspace/Project_Compile/target/classes/com/edurekademo/utilities$ 
```

Code review

1. Created a new job as **Code_Review** and connected the github repository to it
Now install pmd plugin
New Item for Code Review



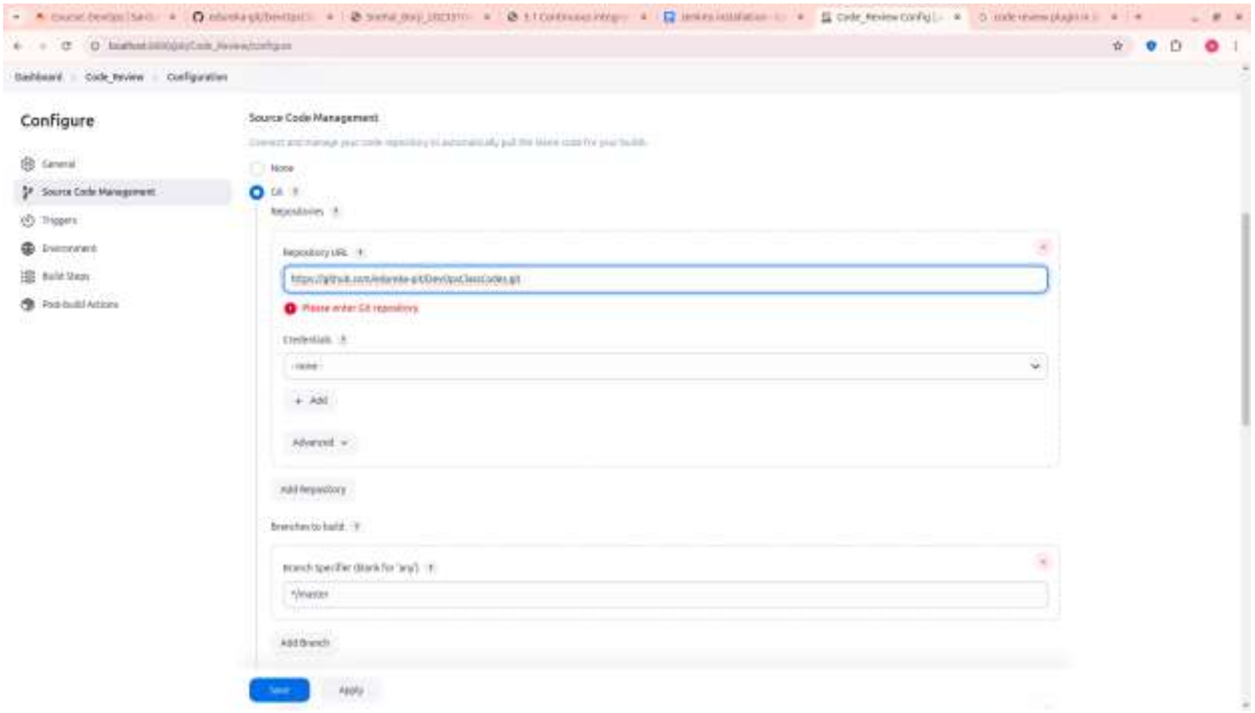
New Item

Enter an item name

Code_Review

Select an item type

In Source Code Management



Source Code Management

None [✓ Git]

Repositories

Repository URL

https://github.com/edureka-git/DevOpsClassCodes.git

Please enter Git repository.

Credentials

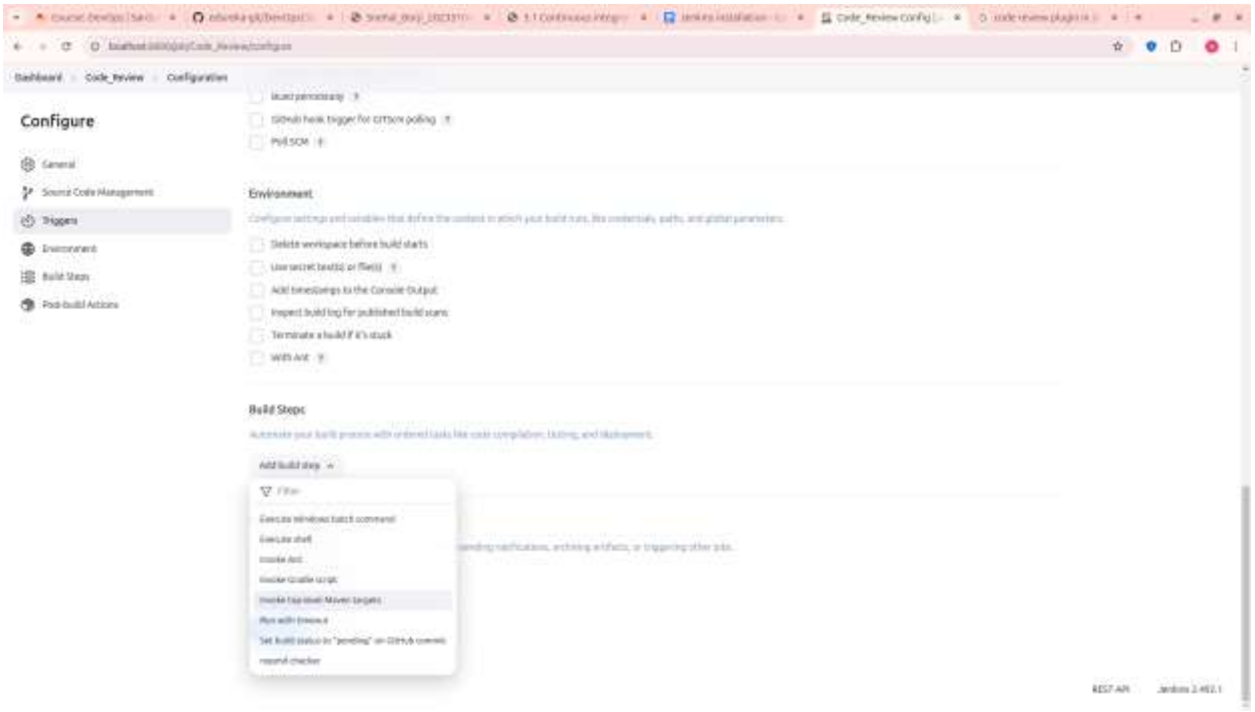
- none -

Branches to build

Branch Specifier (blank for 'any')

*/master

Build Step - > Invoke > type pmd:pmd



Build periodically
GitHub hook trigger for GITScm polling
Poll SCM

Environment

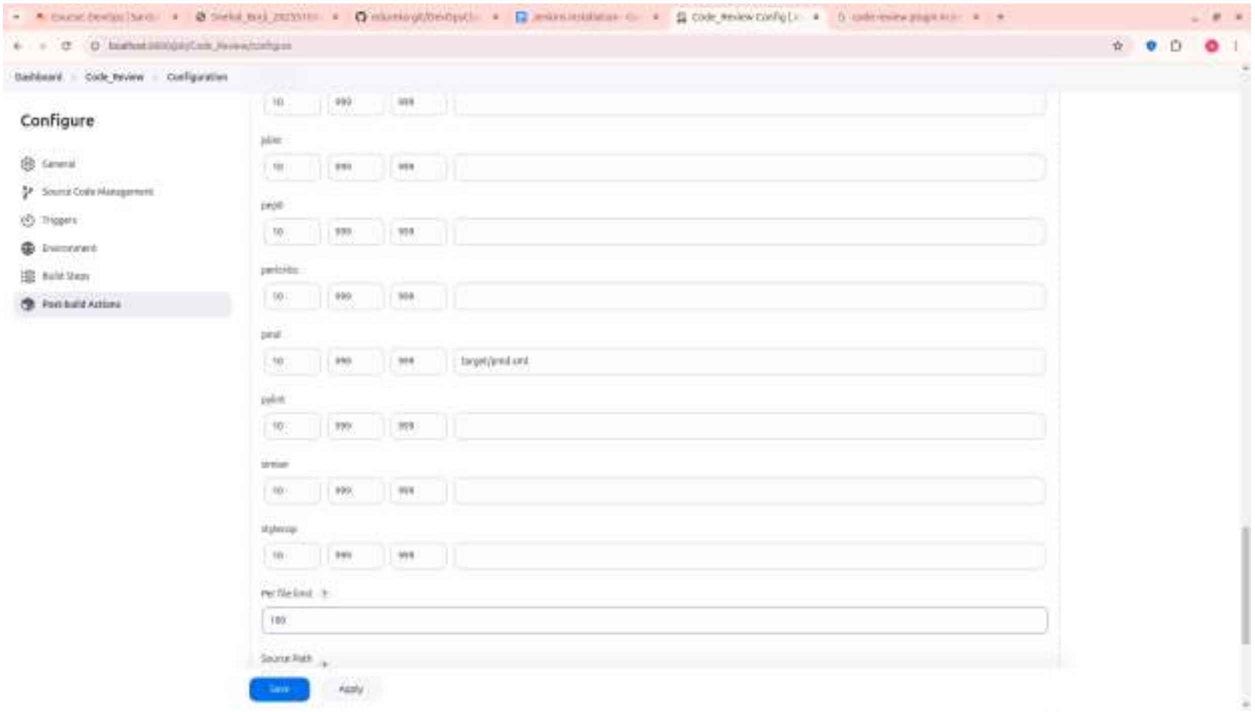
Configure settings and variables that define the context in which your build runs, like credentials, paths, and global parameters.

Delete workspace before build starts
Use secret text(s) or file(s)
Add timestamps to the Console Output
Inspect build log for published build scans
Terminate a build if it's stuck
With Ant

Build Steps

Automate your build process with ordered tasks like code compilation, testing, and deployment.

In -> Post-build Actions -> Report Violations
Now search pms and set -> target/pmd.xml
Click save

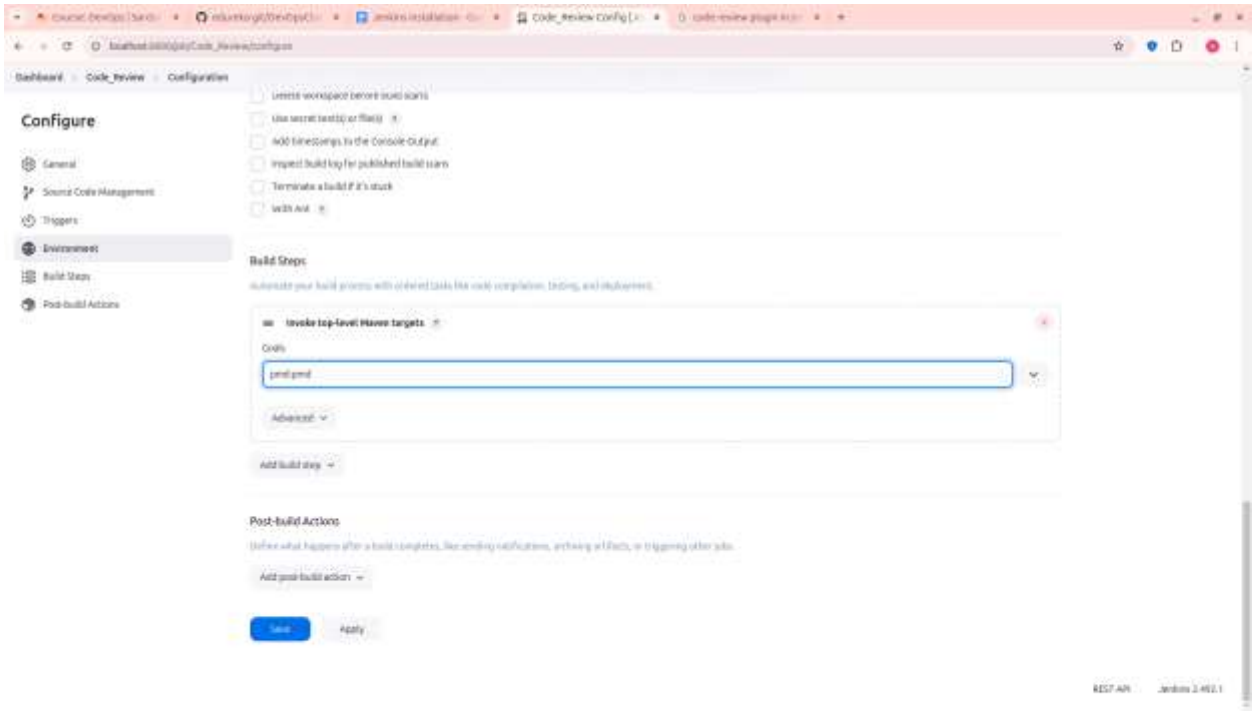


	10	999	999	
jslint	10	999	999	
pep8	10	999	999	
perlcritic	10	999	999	
pmd	10	999	999	target/pmd.xml
pylint	10	999	999	
simian	10	999	999	
stylecop	10	999	999	

Per file limit
100

Source Path

Click Save



Dashboard > Code_Review > Configuration

Configure

- Delete workspace before build starts
- Use secret text(s) or file(s)
- Add timestamps to the Console Output
- Inspect build log for published build scans
- Terminate a build if it's stuck
- With Ant

Build Steps

Automate your build process with ordered tasks like code compilation, testing, and deployment.

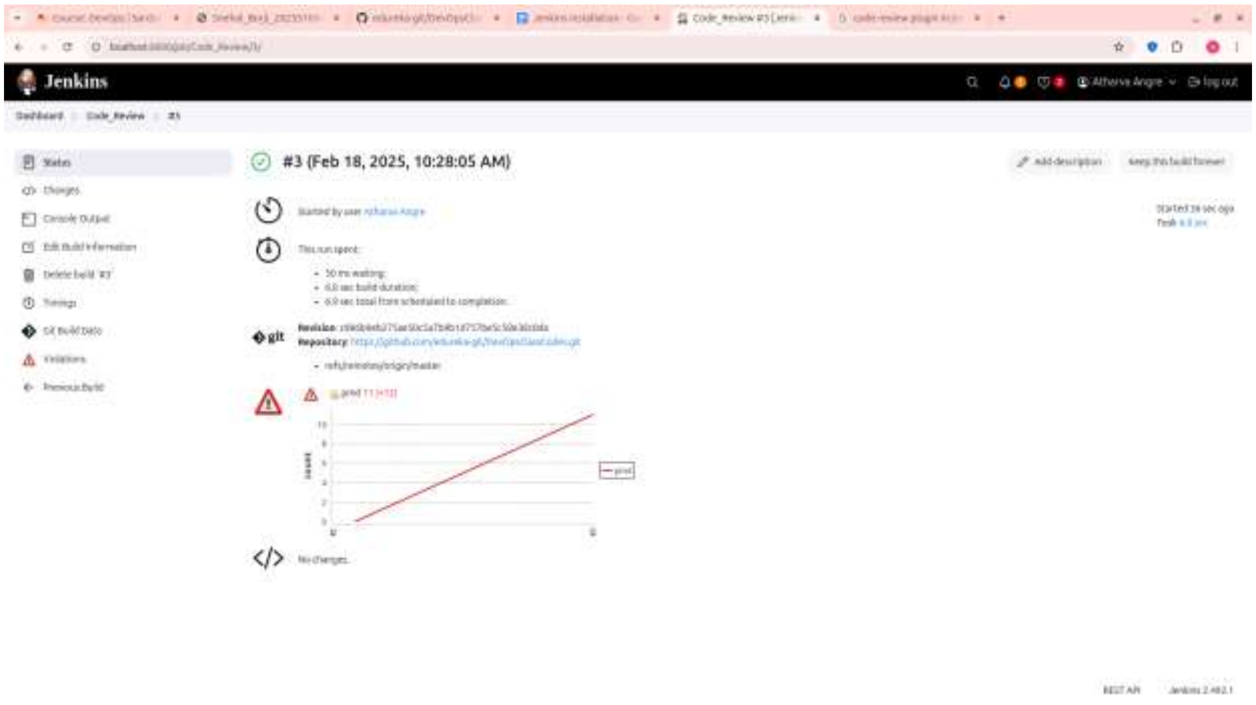
Invoke top-level Maven targets

Goals: pmd:pmd

Post-build Actions

Define what happens after a build completes, like sending notifications, archiving artifacts, or triggering other jobs.

Now Click-> Build Now



Jenkins

Dashboard > Code_Review > #3

#3 (Feb 18, 2025, 10:28:05 AM)

Started by user Atharva Angre

Started 16 sec ago
Took 6.0 sec

This run spent:

- 30 ms waiting;
- 6.0 sec build duration;
- 6.0 sec total from scheduled to completion.

Revision: c9b6b8eb275ae50c5a7b4b1d757be5c50e36ddda
Repository: https://github.com/edureka-git/DevOpsClassCodes.git

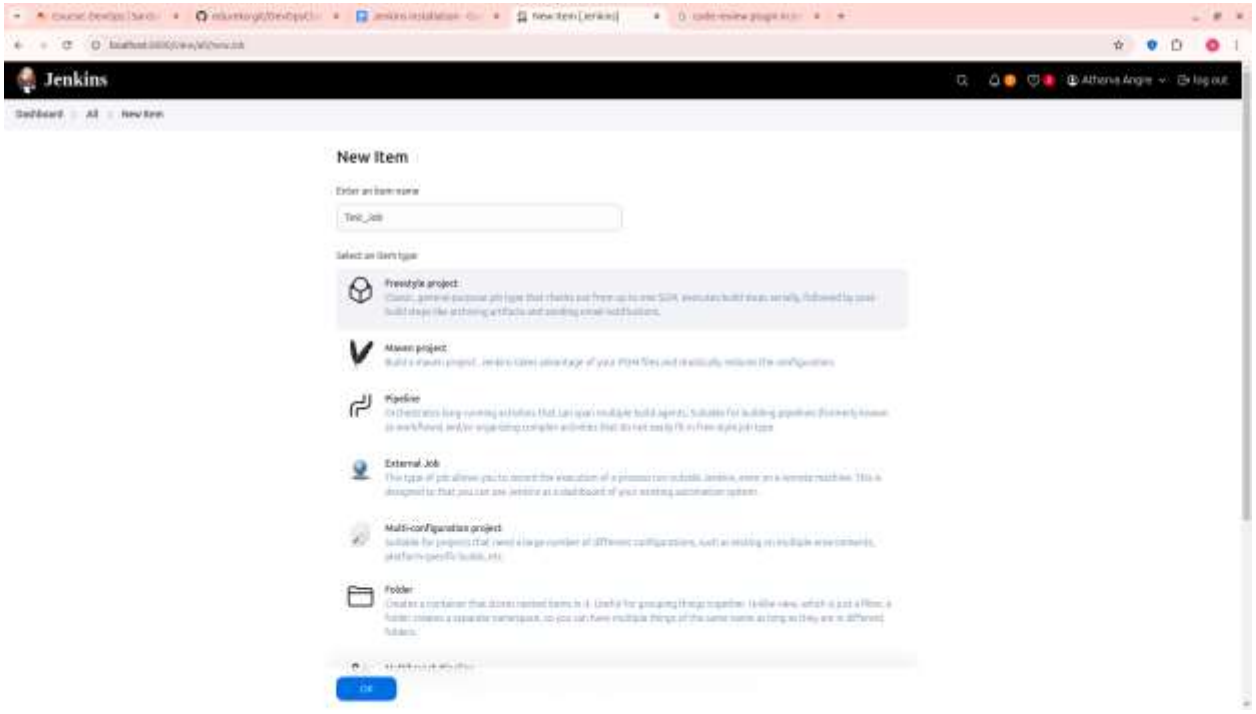
- refs/remotes/origin/master

pmd 11 (+11)

No changes.

Test :

1. Created a new job as a **Test_Job** and added a github repository on it .



localhost:8080/view/all/newJob

Jenkins

Dashboard > All > New Item

New Item

Enter an item name

Test_Job

Select an item type

Freestyle project
Classic, general-purpose job type that checks out from up to one SCM, executes build steps serially, followed by post-build steps like archiving artifacts and sending email notifications.

Maven project
Build a maven project. Jenkins takes advantage of your POM files and drastically reduces the configuration.

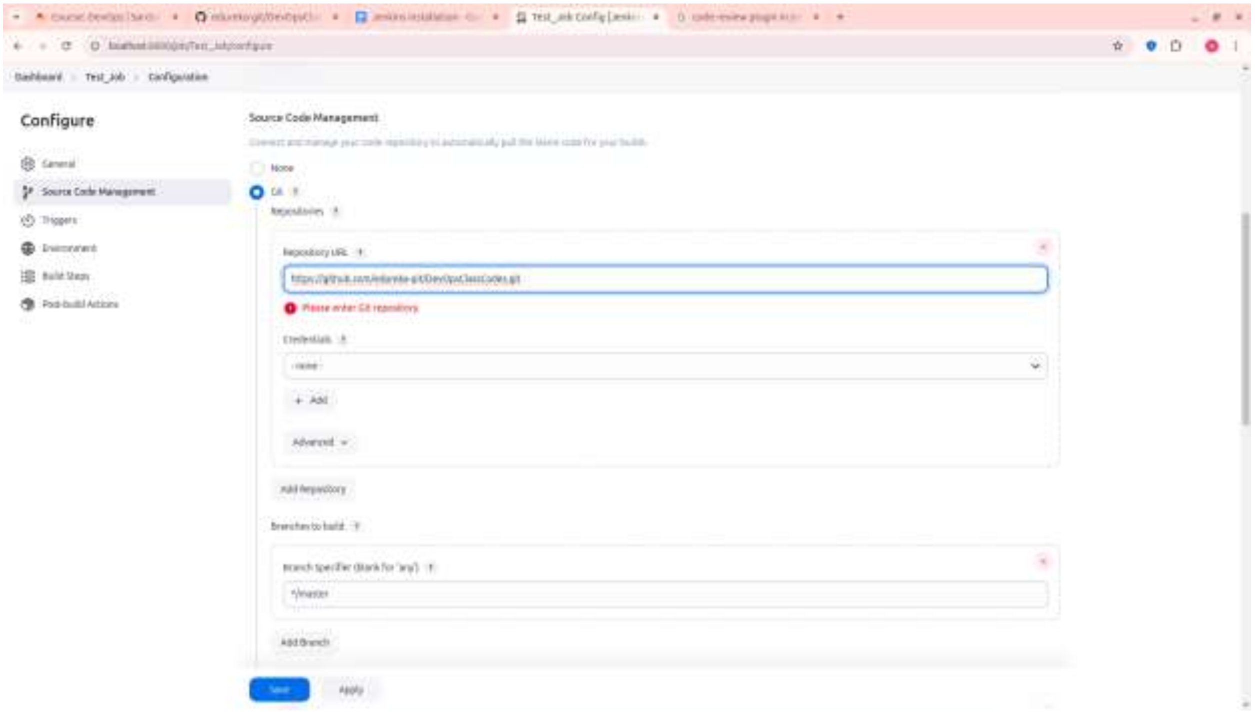
Pipeline
Orchestrates long-running activities that can span multiple build agents. Suitable for building pipelines (formerly known as workflows) and/or organizing complex activities that do not easily fit in free-style job type.

External Job
This type of job allows you to record the execution of a process run outside Jenkins, even on a remote machine. This is designed so that you can use Jenkins as a dashboard of your existing automation system.

Multi-configuration project
Suitable for projects that need a large number of different configurations, such as testing on multiple environments, platform-specific builds, etc.

Folder
Creates a container that stores nested items in it. Useful for grouping things together. Unlike view, which is just a filter, a folder creates a separate namespace, so you can have multiple things of the same name as long as they are in different folders.

OK



localhost:8080/job/Test_Job/configure

Dashboard > Test_Job > Configuration

Configure

Source Code Management

Connect and manage your code repository to automatically pull the latest code for your builds.

None [✓ Git]

Repositories

Repository URL

https://github.com/erkantp-git/DevOpsClassCodes.git

Please enter Git repository.

Credentials

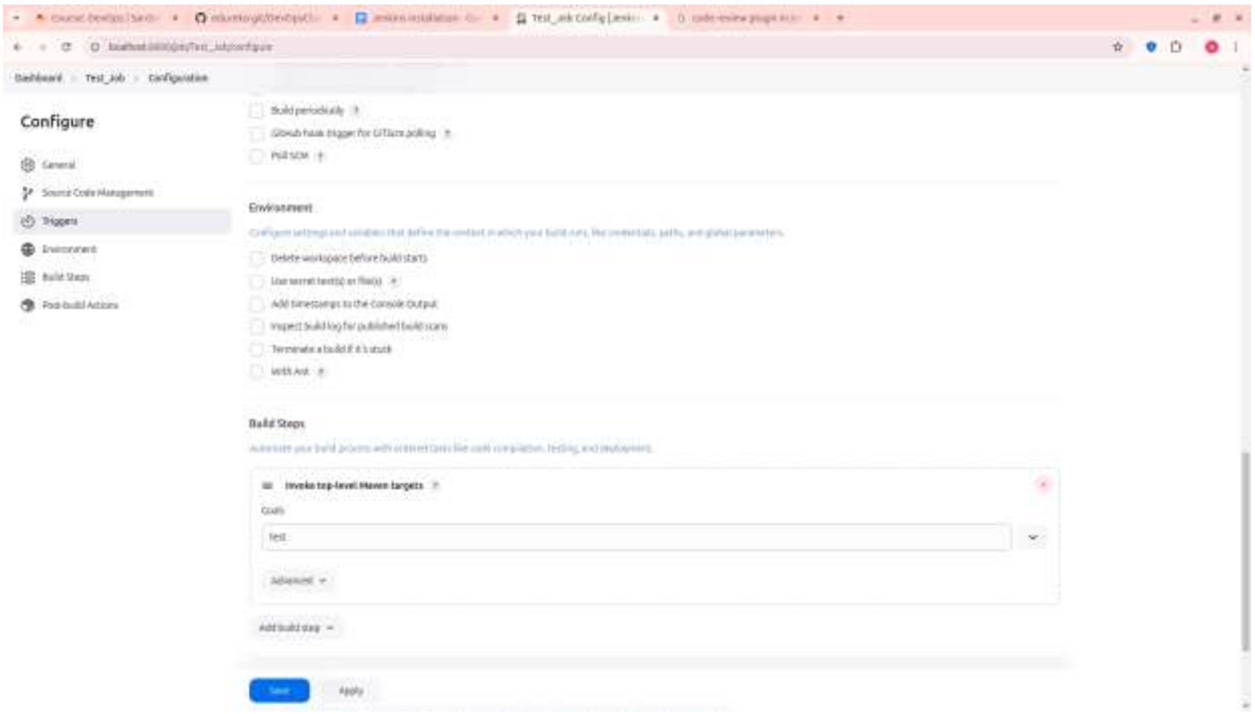
- none -

Branches to build

Branch Specifier (blank for 'any')

*/master

In Build Step select Top-level Maven target in that inside Goals type test



Dashboard > Test_Job > Configuration

Configure

- General
- Source Code Management
- Triggers
- Environment
- Build Steps
- Post-build Actions

- [ ] Build periodically
- [ ] GitHub hook trigger for GITScm polling
- [ ] Poll SCM

Environment

Configure settings and variables that define the context in which your build runs, like credentials, paths, and global parameters.

- [ ] Delete workspace before build starts
- [ ] Use secret text(s) or file(s)
- [ ] Add timestamps to the Console Output
- [ ] Inspect build log for published build scans
- [ ] Terminate a build if it's stuck
- [ ] With Ant

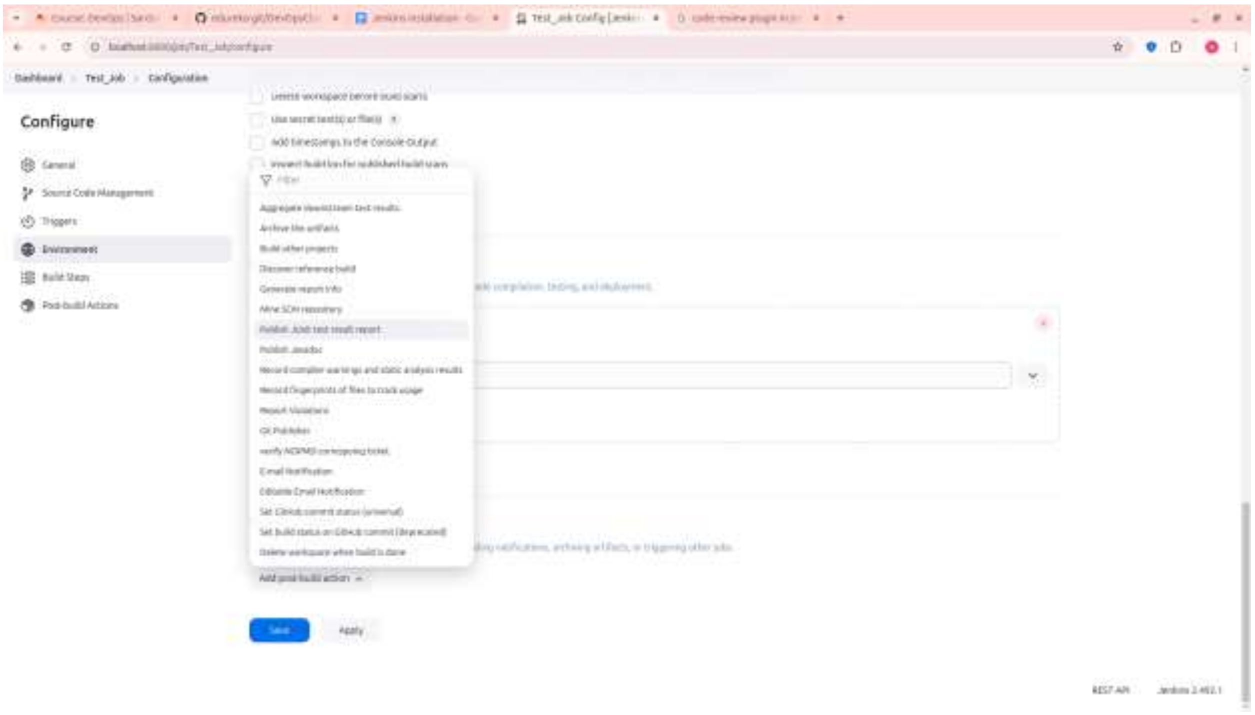
Build Steps

Automate your build process with ordered tasks like code compilation, testing, and deployment.

Invoke top-level Maven targets

Goals: test

In the post build action selected publish JUnit test result report and in it provided path for xml file in which test case report will be generate .



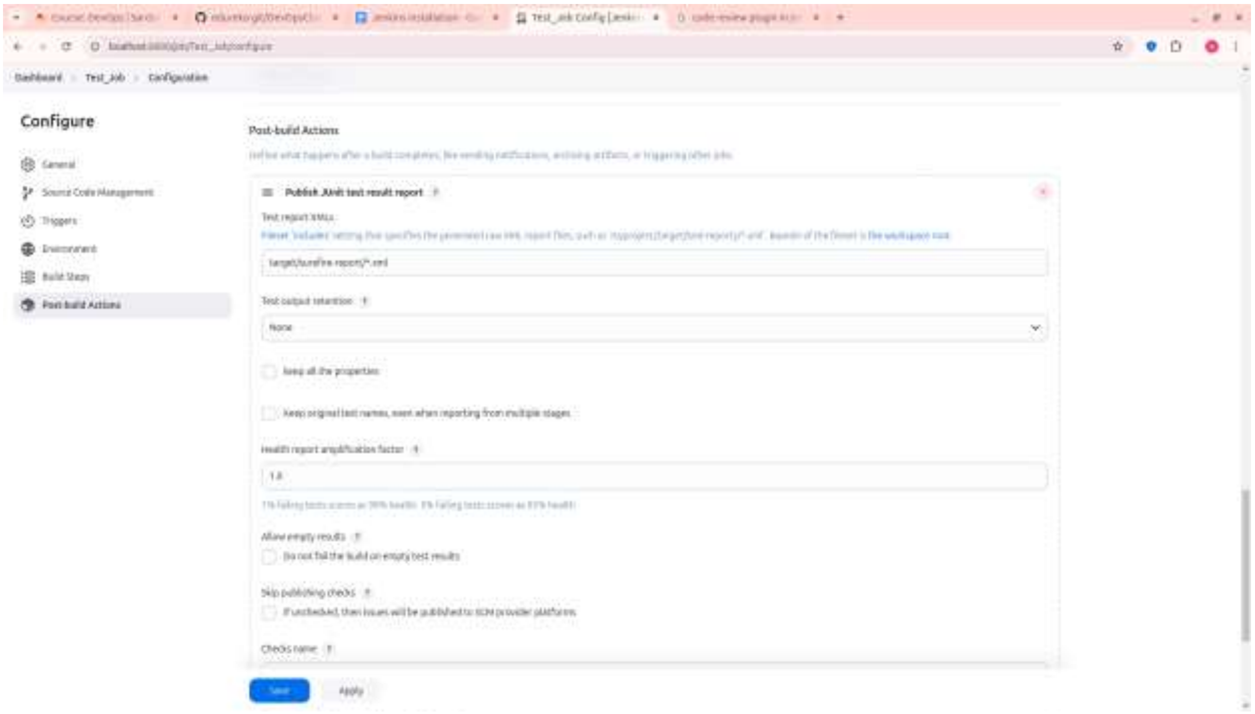
Dashboard > Test_Job > Configuration

Configure

- General
- Source Code Management
- Triggers
- Environment
- Build Steps
- Post-build Actions

- Aggregate downstream test results
- Archive the artifacts
- Build other projects
- Discover reference build
- Generate report info
- Mine SCM repository
- Publish JUnit test result report
- Publish Javadoc
- Record compiler warnings and static analysis results
- Record fingerprints of files to track usage
- Report Violations
- Git Publisher
- verify NCSPMS corresponding ticket
- E-mail Notification
- Editable Email Notification
- Set GitHub commit status (universal)
- Set build status on GitHub commit [deprecated]
- Delete workspace when build is done

Add post-build action



Configure

Post-build Actions

Publish JUnit test result report

Test report XMLs: target/surefire-reports/*.xml

Test output retention: None

Keep all the properties

Keep original test names, even when reporting from multiple stages

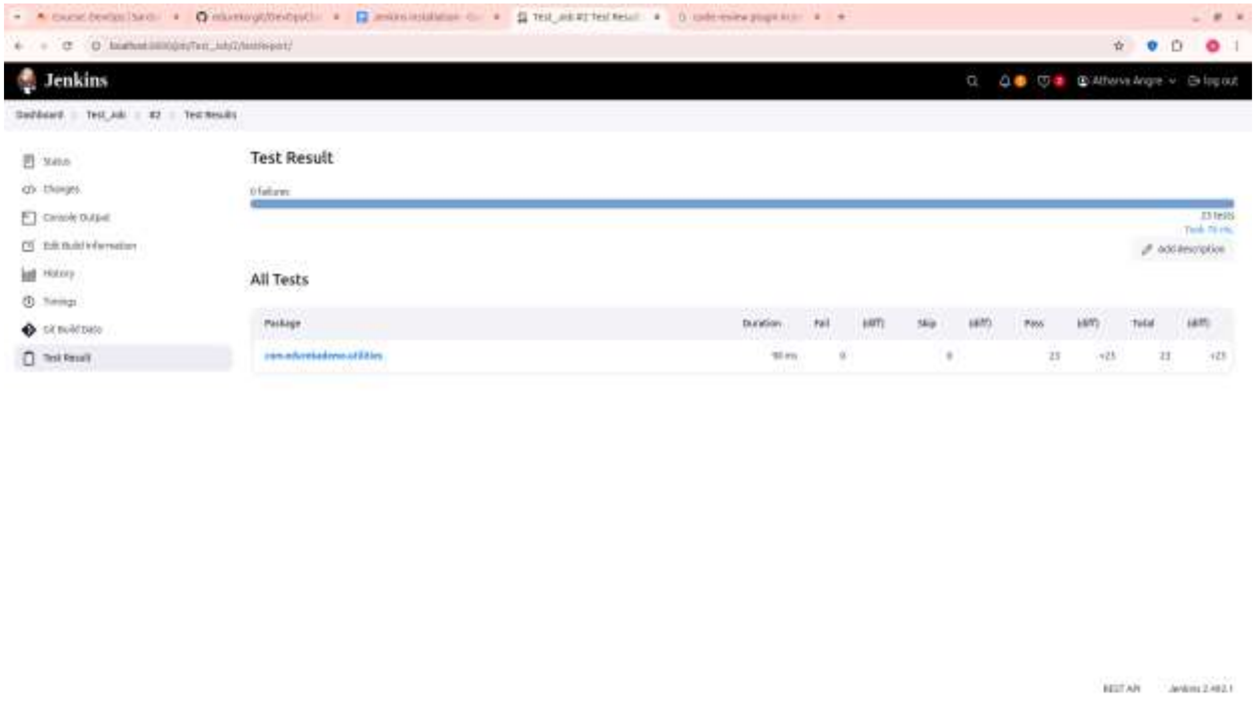
Health report amplification factor: 1.0

Allow empty results: Do not fail the build on empty test results

Skip publishing checks: If unchecked, then issues will be published to SCM provider platforms

Checks name

Click Save and Build



Test Result

0 failures

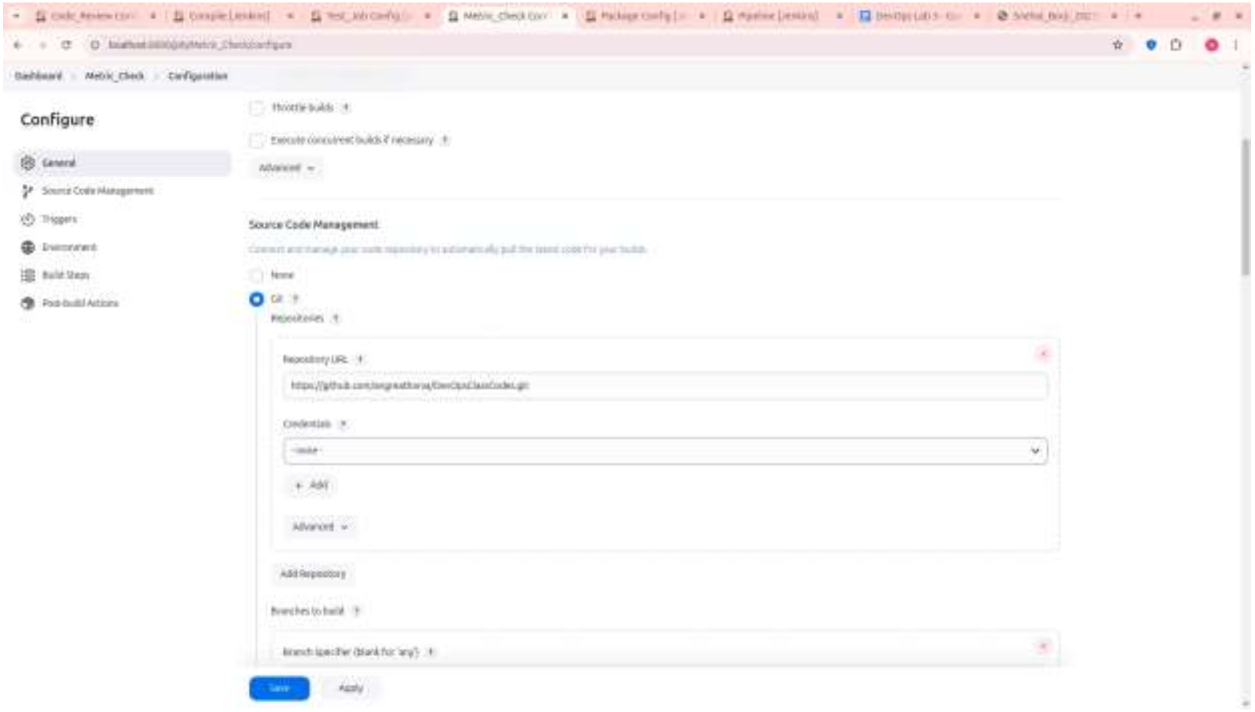
23 tests

All Tests

Package	Duration	Fail	(diff)	Skip	(diff)	Pass	(diff)	Total	(diff)
com.edurekademo.utilities	90 ms	0		0		23	+23	23	+23

Metric check

1. Created a new job as **Metric_Check** and connected a github repository on it .



Source Code Management

None

Git [✓]

Repositories

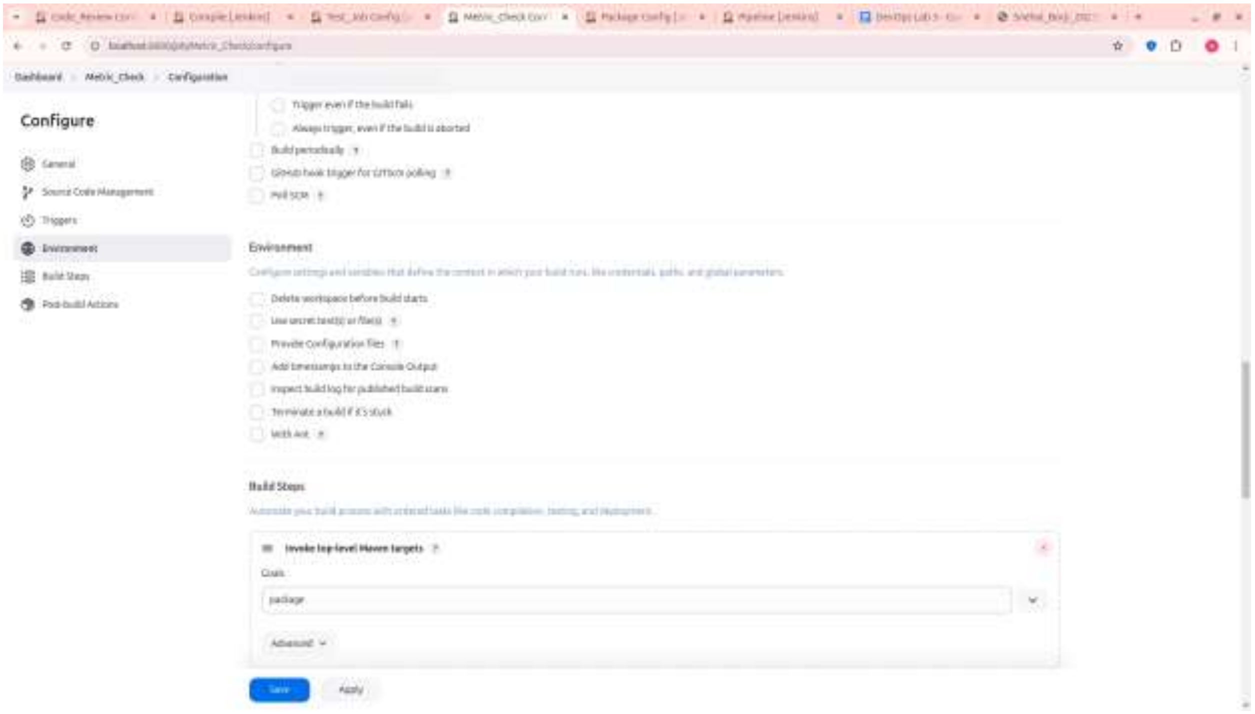
Repository URL: https://github.com/angreatharva/DevOpsClassCodes.git

Credentials: - none -

Branches to build

Branch Specifier (blank for 'any')

2. In build steps, selected a Invoke top-level Maven targets and in goals selected a package plugin.

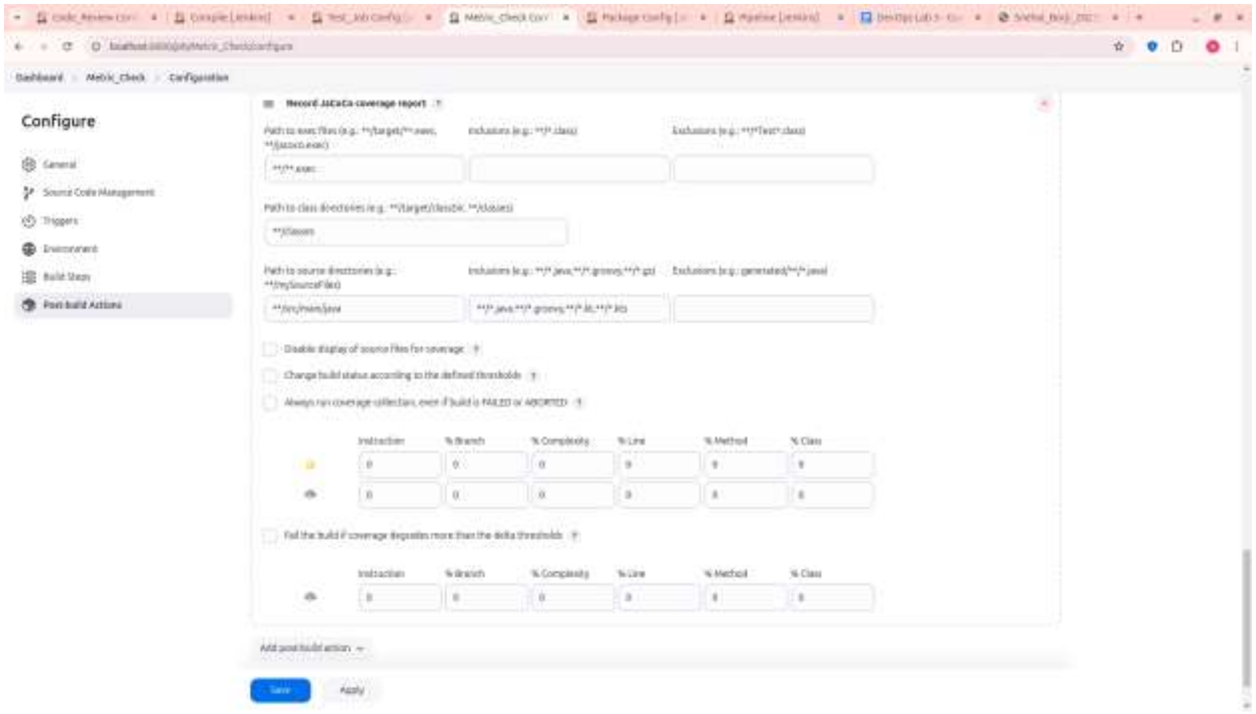


Build Steps

Invoke top-level Maven targets

Goals: package

3. In post-build actions selected the Record Jacoco coverage report .



Configure

Record JaCoCo coverage report

Path to exec files (e.g.: **/target/**.exec, **/jacoco.exec): **/**.exec

Inclusions (e.g.: **/*.class):

Exclusions (e.g.: **/*Test*.class):

Path to class directories (e.g.: **/target/classDir, **/classes): **/classes

Path to source directories (e.g.: **/mySourceFiles): **/src/main/java

Inclusions (e.g.: **/*.java,**/*.groovy,**/*.gs): **/*.java,**/*.groovy,**/*.kt,**/*.kts

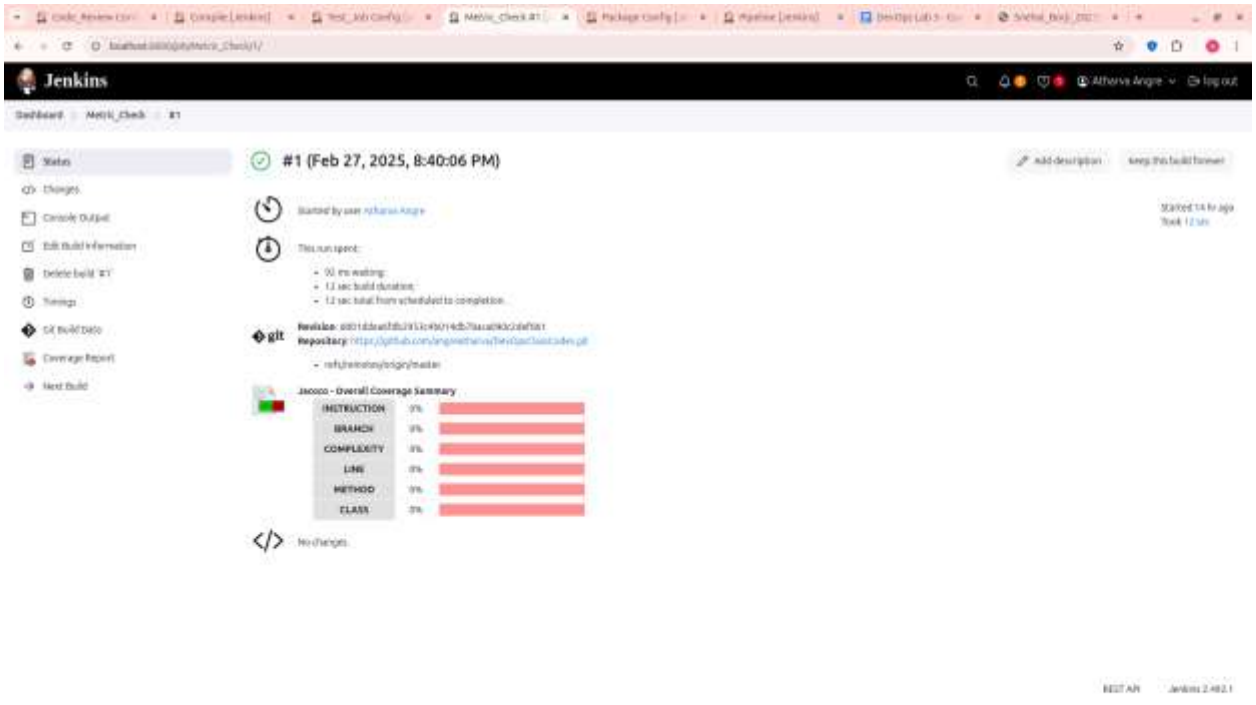
Exclusions (e.g.: generated/**/*.java):

- [ ] Disable display of source files for coverage
- [ ] Change build status according to the defined thresholds
- [ ] Always run coverage collection, even if build is FAILED or ABORTED

	Instruction	% Branch	% Complexity	% Line	% Method	% Class
	0	0	0	0	0	0
	0	0	0	0	0	0

- [ ] Fail the build if coverage degrades more than the delta thresholds

	Instruction	% Branch	% Complexity	% Line	% Method	% Class
	0	0	0	0	0	0



#1 (Feb 27, 2025, 8:40:06 PM)

Started by user Atharva Angre

This run spent:

- 92 ms waiting;
- 12 sec build duration;
- 12 sec total from scheduled to completion.

Revision: [illegible]
Repository: [illegible]

- refs/remotes/origin/master

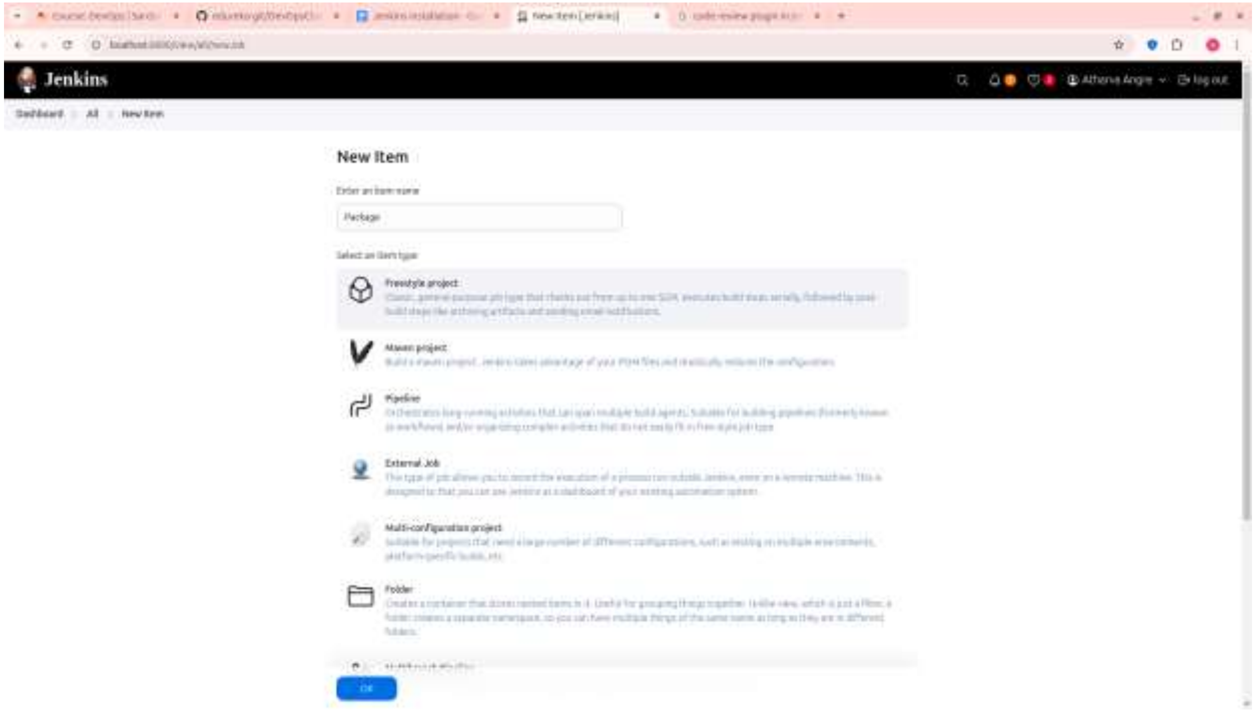
Jacoco - Overall Coverage Summary

INSTRUCTION	0%	🟥
BRANCH	0%	🟥
COMPLEXITY	0%	🟥
LINE	0%	🟥
METHOD	0%	🟥
CLASS	0%	🟥

No changes.

Package

1. Created a new job as a Package and connected github repository on


Dashboard > All > New Item

New Item

Enter an item name

Package

Select an item type

Freestyle project

Maven project

Pipeline

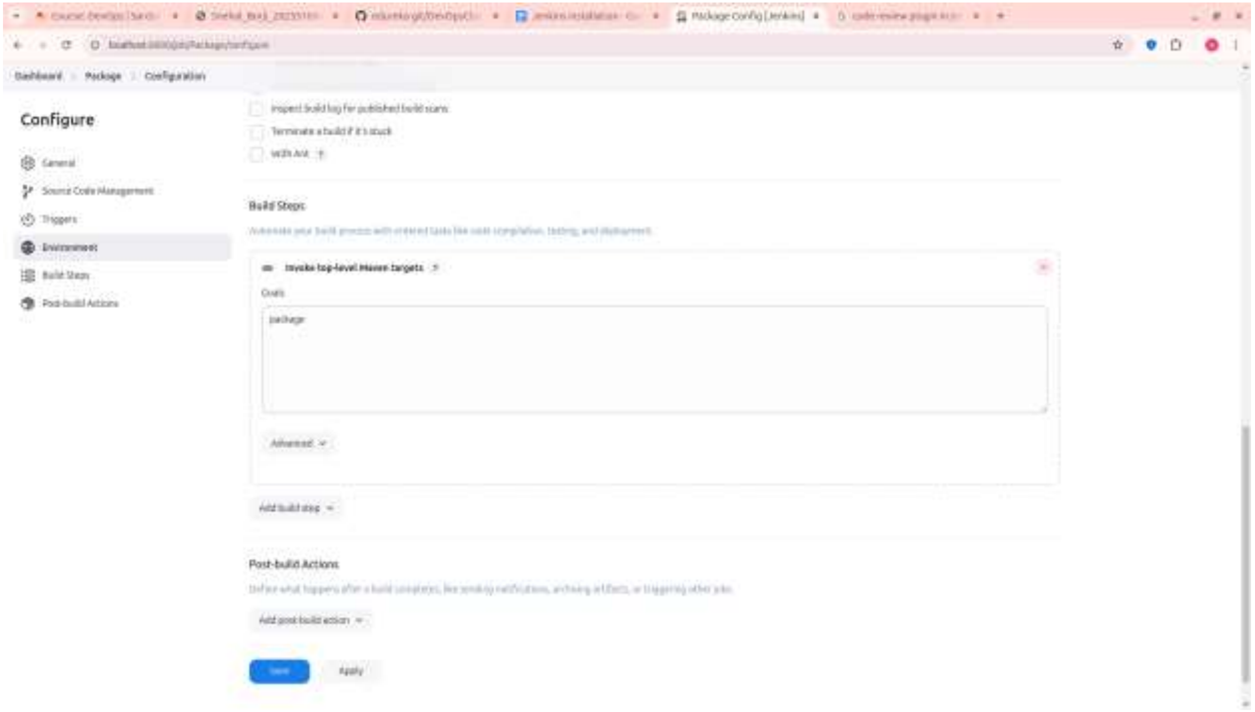
External Job

Multi-configuration project

Folder

OK

2. In build steps, selected a Invoke top-level Maven targets and in goals selected a package plugin


Dashboard > Package > Configuration

Configure

- General
- Source Code Management
- Triggers
- Environment
- Build Steps
- Post-build Actions

Inspect build log for published build scans

Terminate a build if it's stuck

With Ant

Build Steps

Invoke top-level Maven targets

Goals

package

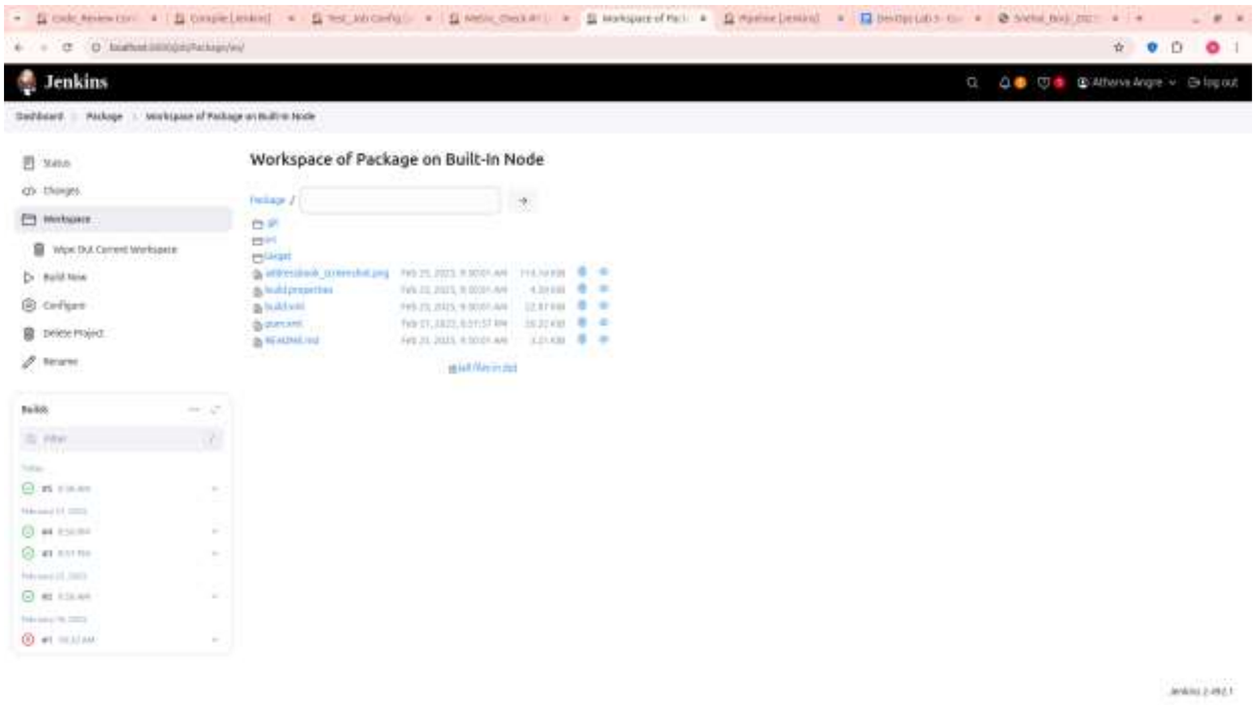
Advanced

Add build step

Post-build Actions

Add post-build action

Save Apply



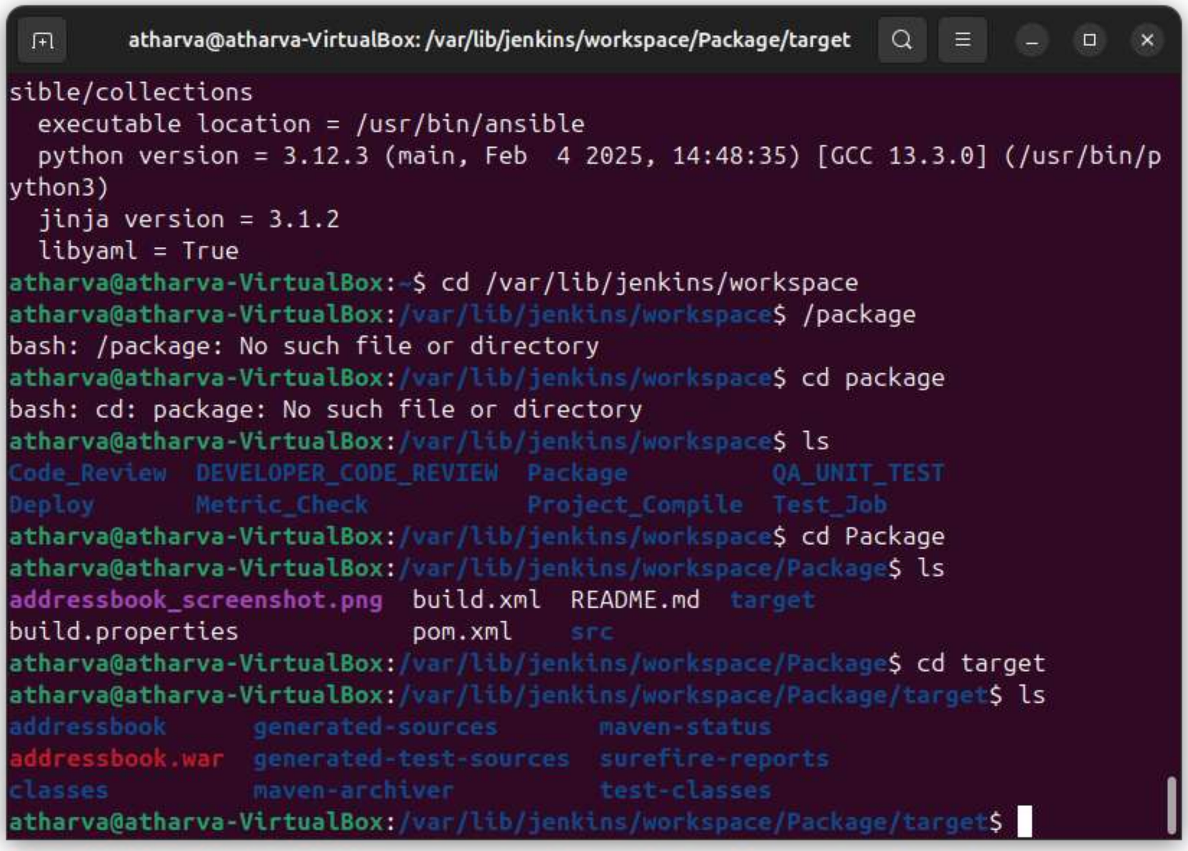
Dashboard > Package > Workspace of Package on Built-In Node

Workspace of Package on Built-In Node

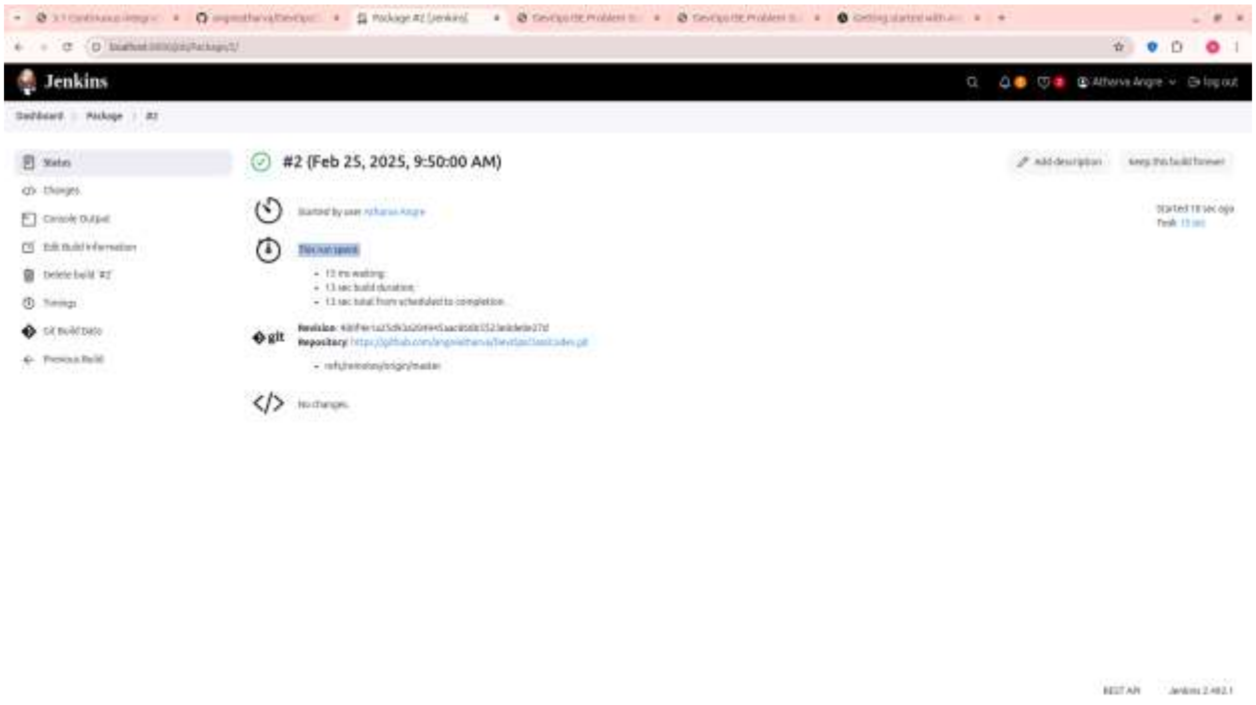
Package /

```
src
target
addressbook_screenshot.png
build.properties
build.xml
pom.xml
README.md
```

(all files in zip)



```
atharva@atharva-VirtualBox: /var/lib/jenkins/workspace/Package/target
sible/collections
  executable location = /usr/bin/ansible
  python version = 3.12.3 (main, Feb  4 2025, 14:48:35) [GCC 13.3.0] (/usr/bin/p
ython3)
  jinja version = 3.1.2
  libyaml = True
atharva@atharva-VirtualBox:~$ cd /var/lib/jenkins/workspace
atharva@atharva-VirtualBox:/var/lib/jenkins/workspace$ /package
bash: /package: No such file or directory
atharva@atharva-VirtualBox:/var/lib/jenkins/workspace$ cd package
bash: cd: package: No such file or directory
atharva@atharva-VirtualBox:/var/lib/jenkins/workspace$ ls
Code_Review  DEVELOPER_CODE_REVIEW  Package         QA_UNIT_TEST
Deploy       Metric_Check           Project_Compile  Test_Job
atharva@atharva-VirtualBox:/var/lib/jenkins/workspace$ cd Package
atharva@atharva-VirtualBox:/var/lib/jenkins/workspace/Package$ ls
addressbook_screenshot.png  build.xml  README.md  target
build.properties            pom.xml    src
atharva@atharva-VirtualBox:/var/lib/jenkins/workspace/Package$ cd target
atharva@atharva-VirtualBox:/var/lib/jenkins/workspace/Package/target$ ls
addressbook      generated-sources       maven-status
addressbook.war  generated-test-sources  surefire-reports
classes          maven-archiver          test-classes
atharva@atharva-VirtualBox:/var/lib/jenkins/workspace/Package/target$
```



Dashboard > Package > #2

#2 (Feb 25, 2025, 9:50:00 AM)

Started by user Atharva Angre

This run spent:

- 13 ms waiting;
- 13 sec build duration;
- 13 sec total from scheduled to completion.

Revision: [illegible]
Repository: [illegible]

- refs/remotes/origin/master

No changes.

Started 18 sec ago
Took 13 sec

Observation : In this practical, we learned that Jenkins jobs streamline software development by automating tasks like compiling, testing, and packaging code. They enhance efficiency, maintain consistency across teams, and can be customized to meet specific requirements, ultimately optimizing the development process.

PRACTICAL 3.2

Aim: Create a CI/CD pipeline in Jenkins

Objectives:
● To understand Creation of CICD Pipeline
● The objective of Jenkins is to automate the building, testing, and deployment of software Projects.

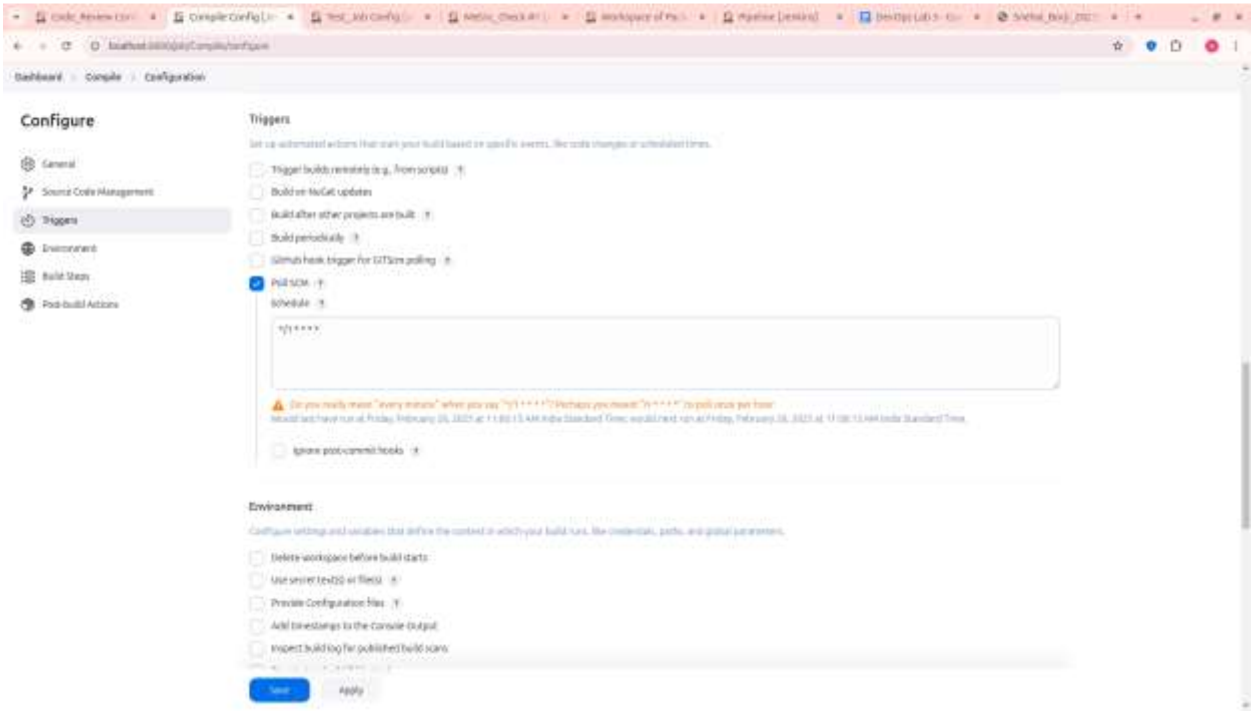
Tools Used: Virtual box manager, Ubuntu , Jenkins

Concepts: CI/CD pipeline: It's a workflow that automates the steps from writing code to deploying it. Continuous Integration (CI) checks code changes frequently, integrating them into the main codebase. Continuous Deployment (CD) automates the deployment process, making software delivery faster and more reliable.

Problem statement :
1) Create a freestyle project with the name QA_UNIT_TEST in Jenkins that is driven from the job DEVELOPER_CODE_REVIEW and performs unit testing. Take a screenshot of the console output showing successful build of unit testing
2) Create a freestyle project with the name QA_ METRICS _CHECKin Jenkins to check the test cases. Make sure the Cobertura Plugin is installed in Jenkins. Take a screenshot of the metrics from the dashboard of the project.
3) Create a freestyle project with the name QA_ PACKAGE in Jenkins to create an executable jar/war file. Take a screenshot of the target folder created in the workspace.
4) Create a pipeline named SAMPLE_COMPILE_VIEWwith Build PipelineView option, select DEVELOPER_CODE_REVIEW project under layout section,andrun the pipeline to check the console output Take a screenshot of the pipeline dashboard showing the status of the projects

Solution :

1.Opened a compile job , selected build triggers in which I selected poll scm and entered the following cron expression .



Dashboard > Compile > Configuration

Configure

Triggers

Set up automated actions that start your build based on specific events, like code changes or scheduled times.

- [ ] Trigger builds remotely (e.g., from scripts)
- [ ] Build on NuGet updates
- [ ] Build after other projects are built
- [ ] Build periodically
- [ ] GitHub hook trigger for GITScm polling
- [x] Poll SCM

Schedule

```
*/1 * * * *
```

Do you really mean "every minute" when you say "*/1 * * * *"? Perhaps you meant "H * * * *" to poll once per hour
Would last have run at Friday, February 28, 2025 at 11:08:13 AM India Standard Time; would next run at Friday, February 28, 2025 at 11:08:13 AM India Standard Time.

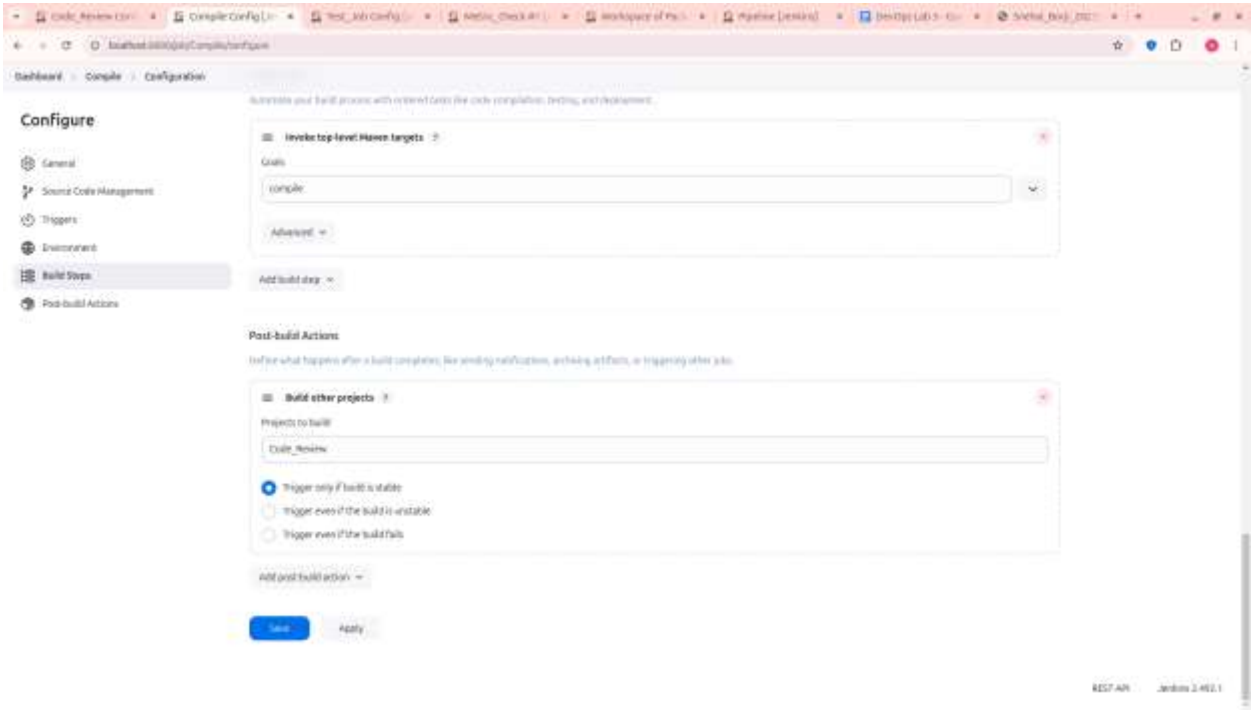
- [ ] Ignore post-commit hooks

Environment

Configure settings and variables that define the context in which your build runs, like credentials, paths, and global parameters.

- [ ] Delete workspace before build starts
- [ ] Use secret text(s) or file(s)
- [ ] Provide Configuration files
- [ ] Add timestamps to the Console Output
- [ ] Inspect build log for published build scans

2. In the post-build action, I selected "Build other project," in which I added the "Code_Review" job.



Dashboard > Compile > Configuration

Configure

Automate your build process with ordered tasks like code compilation, testing, and deployment.

Invoke top-level Maven targets

Goals: compile

Post-build Actions

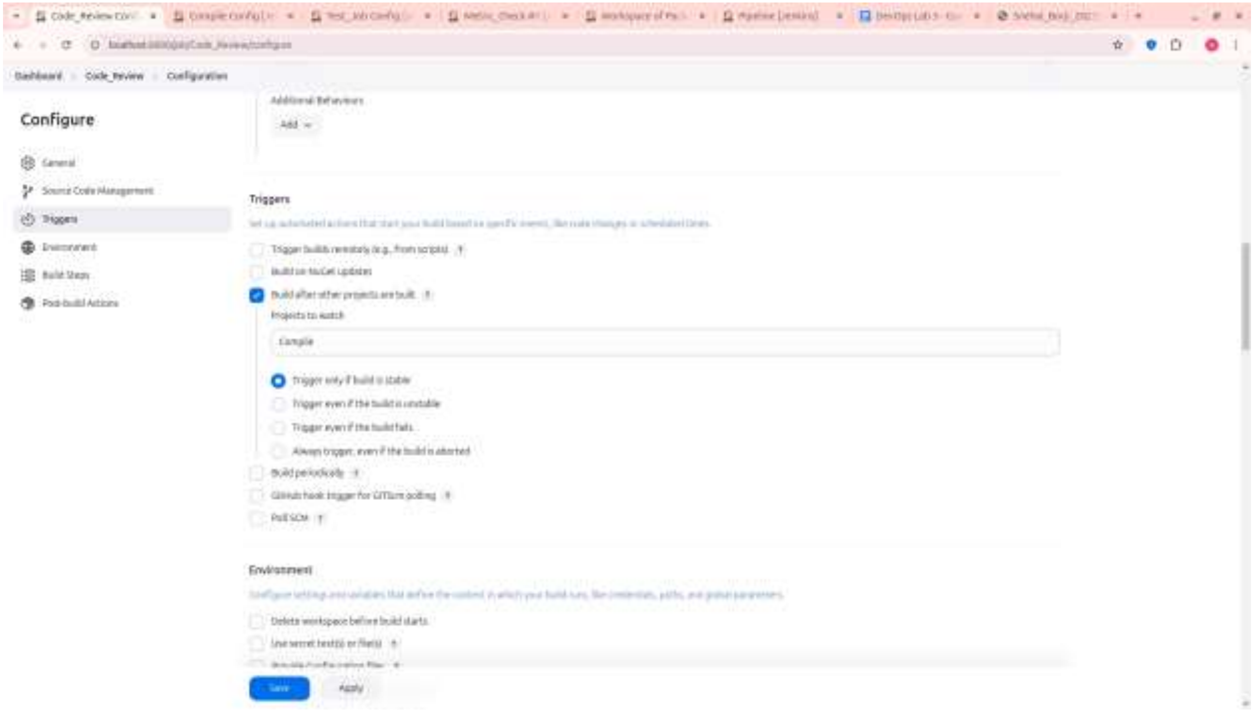
Define what happens after a build completes, like sending notifications, archiving artifacts, or triggering other jobs.

Build other projects

Projects to build: Code_Review

[✓ Trigger only if build is stable]
Trigger even if the build is unstable
Trigger even if the build fails

3.Opened a Code_Review job , selected built after other projects are built in which I selected a compile job.



Dashboard > Code_Review > Configuration

Configure

- General
- Source Code Management
- Triggers
- Environment
- Build Steps
- Post-build Actions

Additional Behaviours

Add

Triggers

Set up automated actions that start your build based on specific events, like code changes or scheduled times.

- [ ] Trigger builds remotely (e.g., from scripts)
- [ ] Build on NuGet updates
- [x] Build after other projects are built

Projects to watch: Compile

[✓ Trigger only if build is stable] Trigger even if the build is unstable / Trigger even if the build fails / Always trigger, even if the build is aborted

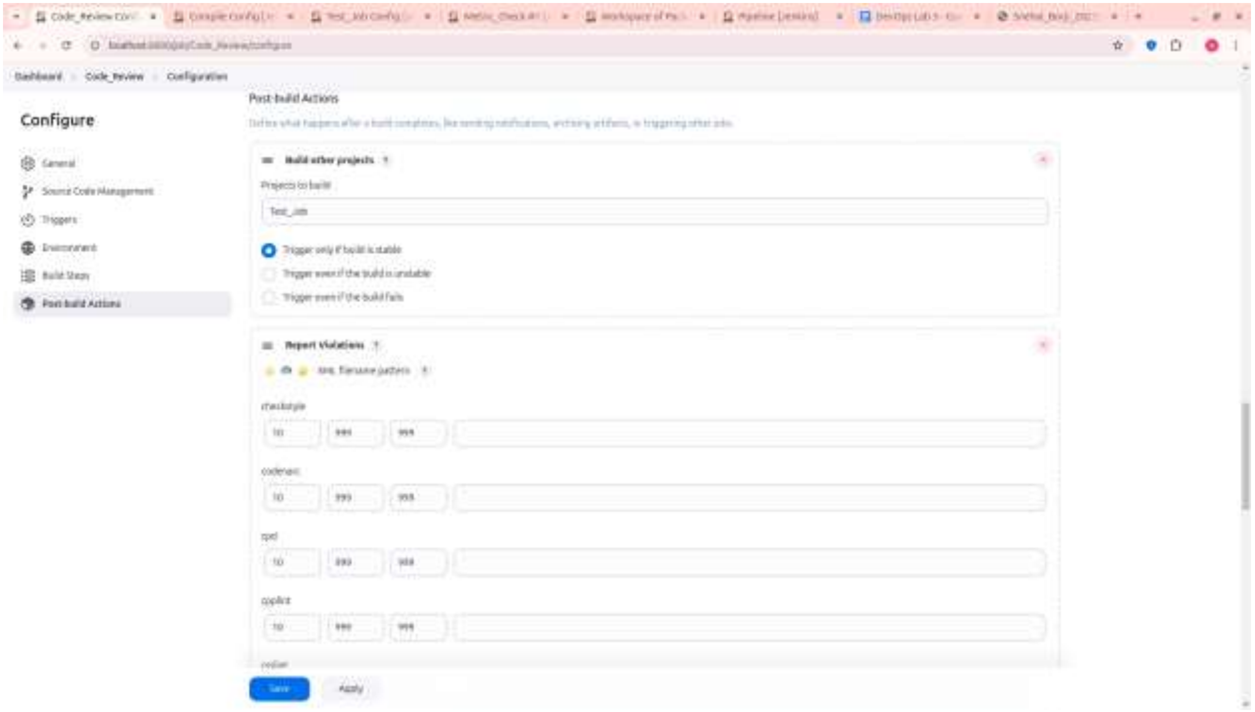
- [ ] Build periodically
- [ ] GitHub hook trigger for GITScm polling
- [ ] Poll SCM

Environment

Configure settings and variables that define the context in which your build runs, like credentials, paths, and global parameters.

- [ ] Delete workspace before build starts
- [ ] Use secret text(s) or file(s)

4. In the post-build action, I selected "Build other project," in which I added the "Test_Job" job.



Dashboard > Code_Review > Configuration

Configure

- General
- Source Code Management
- Triggers
- Environment
- Build Steps
- Post-build Actions

Post-build Actions

Define what happens after a build completes, like sending notifications, archiving artifacts, or triggering other jobs.

Build other projects

Projects to build: Test_Job

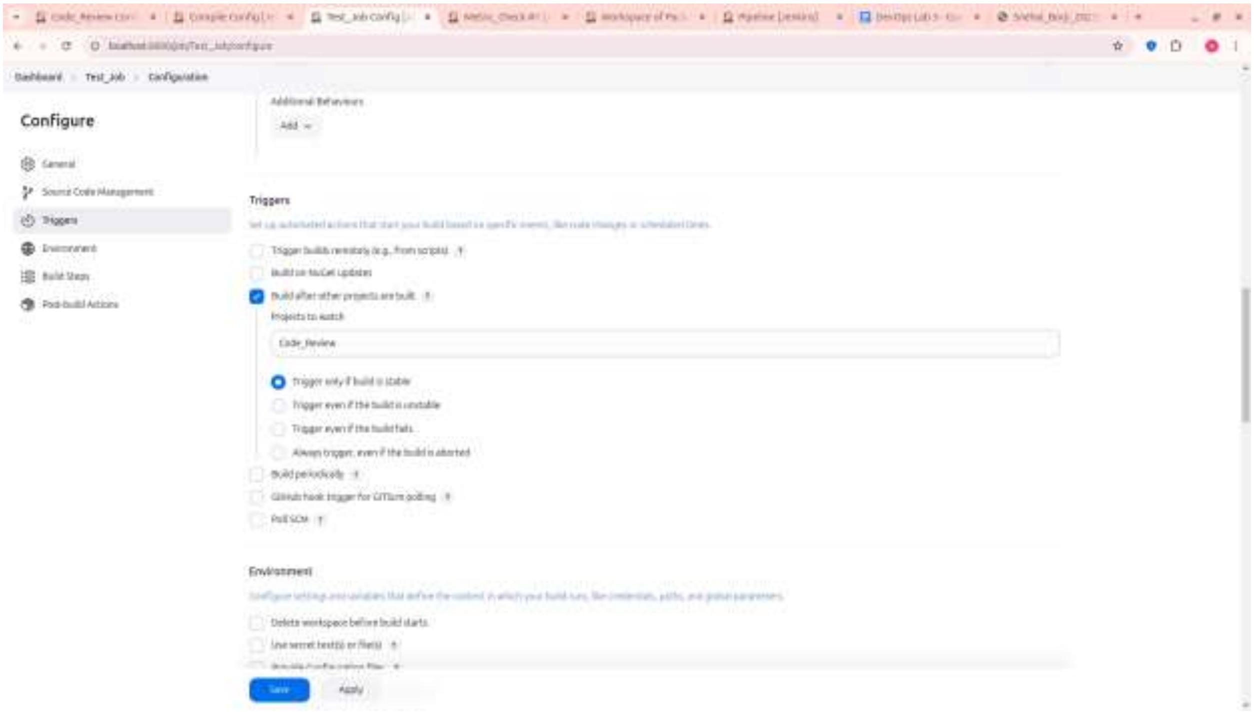
[✓ Trigger only if build is stable] Trigger even if the build is unstable / Trigger even if the build fails

Report Violations

XML filename pattern

checkstyle	10	999	999
codenarc	10	999	999
cpd	10	999	999
cpplint	10	999	999

5.Opened a test job , selected built after other projects are built in which I selected a Code_Review job.



Dashboard > Test_Job > Configuration

Configure

- General
- Source Code Management
- Triggers
- Environment
- Build Steps
- Post-build Actions

Additional Behaviours

Add

Triggers

Set up automated actions that start your build based on specific events, like code changes or scheduled times.

- [ ] Trigger builds remotely (e.g., from scripts)
- [ ] Build on NuGet updates
- [x] Build after other projects are built

Projects to watch: Code_Review

- [✓ Trigger only if build is stable]
- Trigger even if the build is unstable
- Trigger even if the build fails
- Always trigger, even if the build is aborted

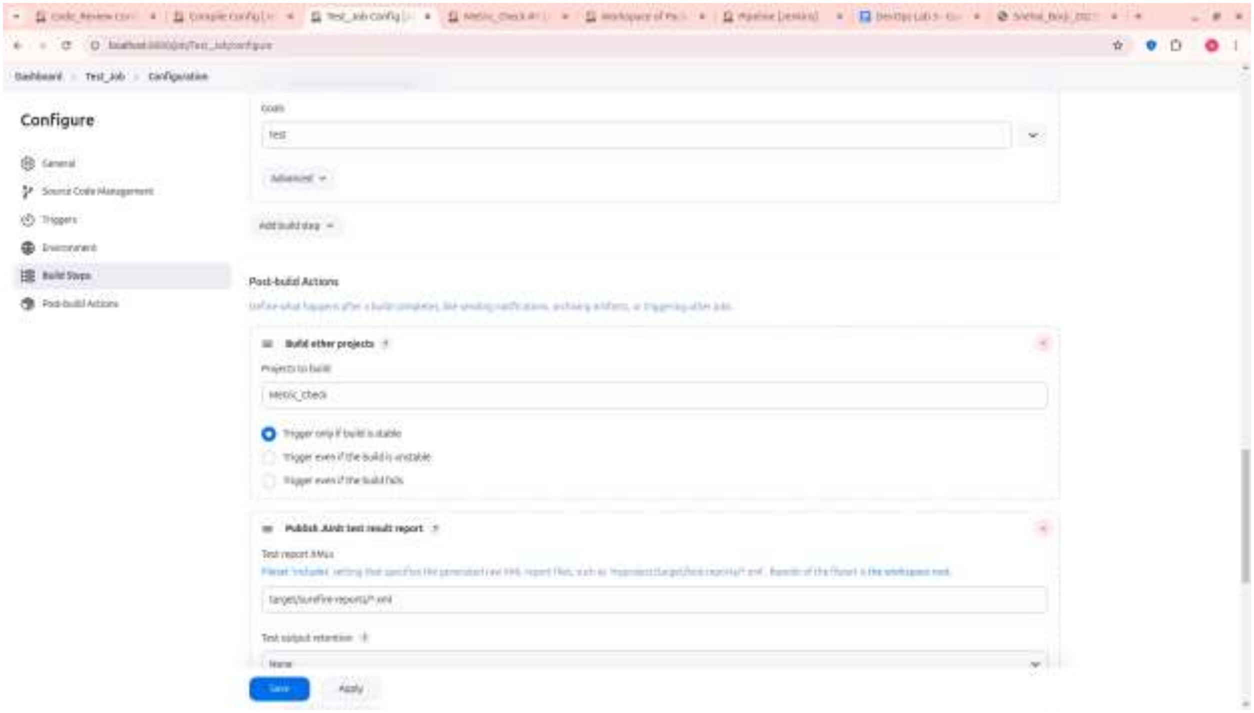
- [ ] Build periodically
- [ ] GitHub hook trigger for GITScm polling
- [ ] Poll SCM

Environment

Configure settings and variables that define the context in which your build runs, like credentials, paths, and global parameters.

- [ ] Delete workspace before build starts
- [ ] Use secret text(s) or file(s)

6. In the post-build action, I selected "Build other project," in which I added the "Metric_Check" job.



Dashboard > Test_Job > Configuration

Configure

- General
- Source Code Management
- Triggers
- Environment
- Build Steps
- Post-build Actions

Goals: test

Advanced

Add build step

Post-build Actions

Define what happens after a build completes, like sending notifications, archiving artifacts, or triggering other jobs.

Build other projects

Projects to build: Metric_check

- [✓ Trigger only if build is stable]
- Trigger even if the build is unstable
- Trigger even if the build fails

Publish JUnit test result report

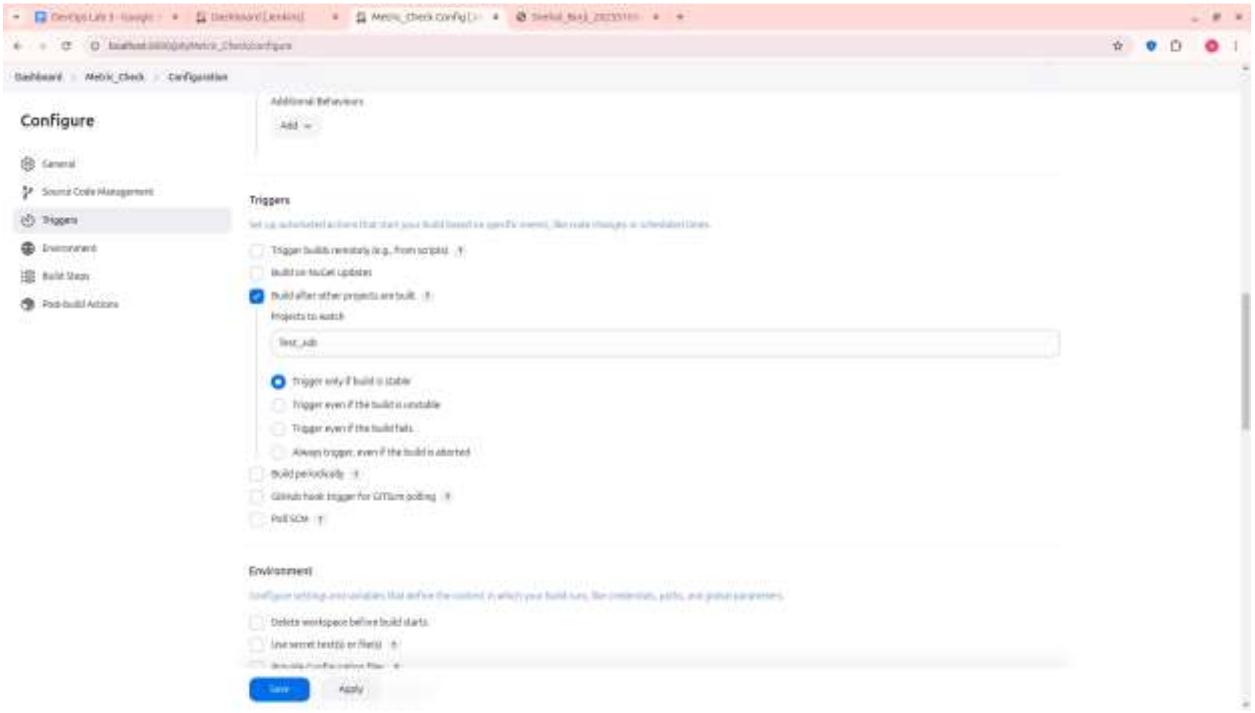
Test report XMLs

Fileset 'includes' setting that specifies the generated raw XML report files, such as 'myproject/target/test-reports/*.xml'. Basedir of the fileset is the workspace root.

target/surefire-reports/*.xml

Test output retention: None

7.Opened a Metric_Check job , selected built after other projects are built in which I selected a test_Job job.



Dashboard > Metric_Check > Configuration

Configure

- General
- Source Code Management
- Triggers
- Environment
- Build Steps
- Post-build Actions

Additional Behaviours

Add

Triggers

Set up automated actions that start your build based on specific events, like code changes or scheduled times.

- [ ] Trigger builds remotely (e.g., from scripts)
- [ ] Build on NuGet updates
- [x] Build after other projects are built

Projects to watch

Test_Job

- (●) Trigger only if build is stable
- ( ) Trigger even if the build is unstable
- ( ) Trigger even if the build fails
- ( ) Always trigger, even if the build is aborted

- [ ] Build periodically
- [ ] GitHub hook trigger for GITScm polling
- [ ] Poll SCM

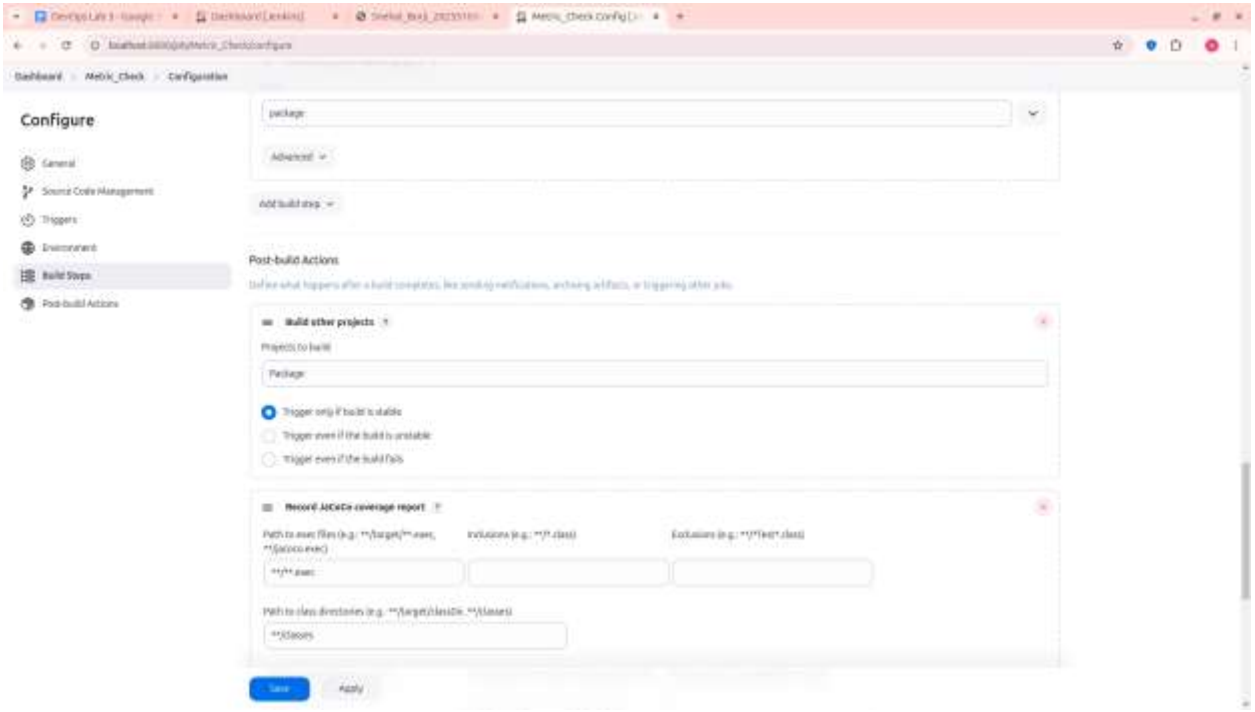
Environment

Configure settings and variables that define the context in which your build runs, like credentials, paths, and global parameters.

- [ ] Delete workspace before build starts
- [ ] Use secret text(s) or file(s)

Save Apply

8.In the post-build action, I selected "Build other project," in which I added the "Package" job.



Dashboard > Metric_Check > Configuration

Configure

- General
- Source Code Management
- Triggers
- Environment
- Build Steps
- Post-build Actions

package

Advanced

Add build step

Post-build Actions

Define what happens after a build completes, like sending notifications, archiving artifacts, or triggering other jobs.

Build other projects

Projects to build

Package

- (●) Trigger only if build is stable
- ( ) Trigger even if the build is unstable
- ( ) Trigger even if the build fails

Record JaCoCo coverage report

Path to exec files (e.g.: **/target/**.exec, **/jacoco.exec): **/**.exec

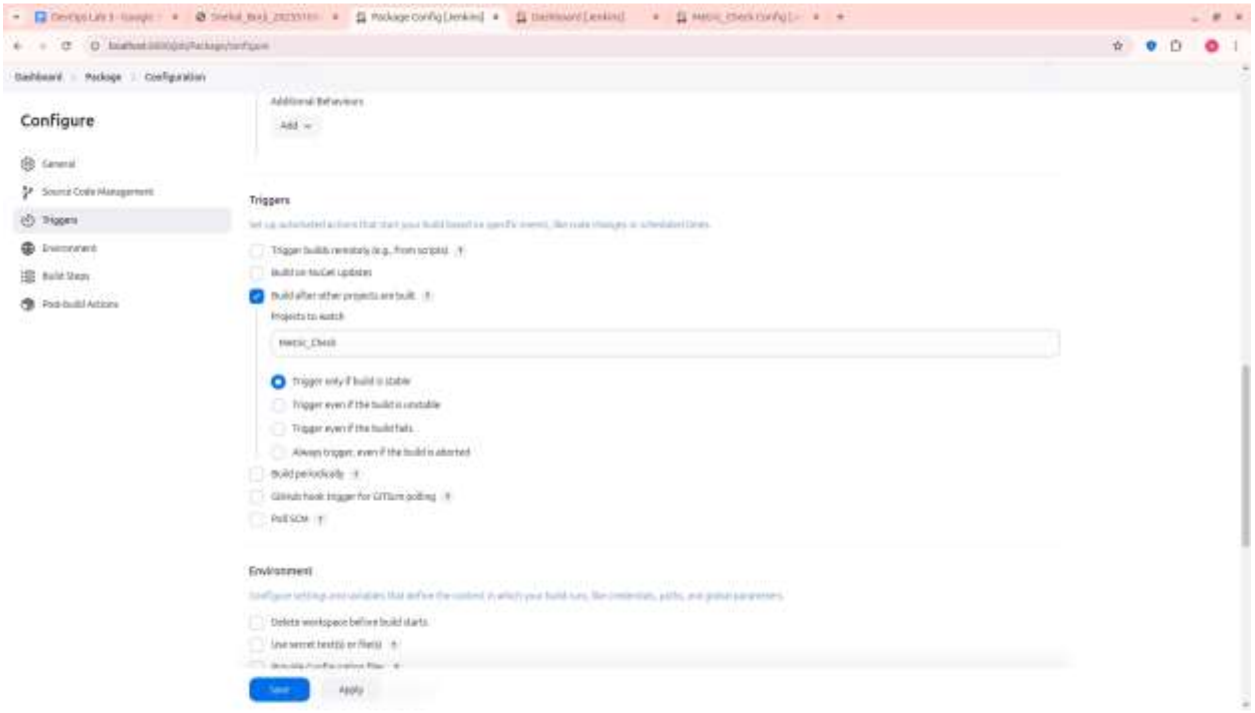
Inclusions (e.g.: **/*.class):

Exclusions (e.g.: **/*Test*.class):

Path to class directories (e.g.: **/target/classDir, **/classes): **/classes

Save Apply

9. Opened a package job , selected built after other projects are built in which I selected a Metric_Check job.



Configure

Triggers

- [ ] Trigger builds remotely (e.g., from scripts)
- [ ] Build on NuGet updates
- [✓] Build after other projects are built

Projects to watch: Metric_Check

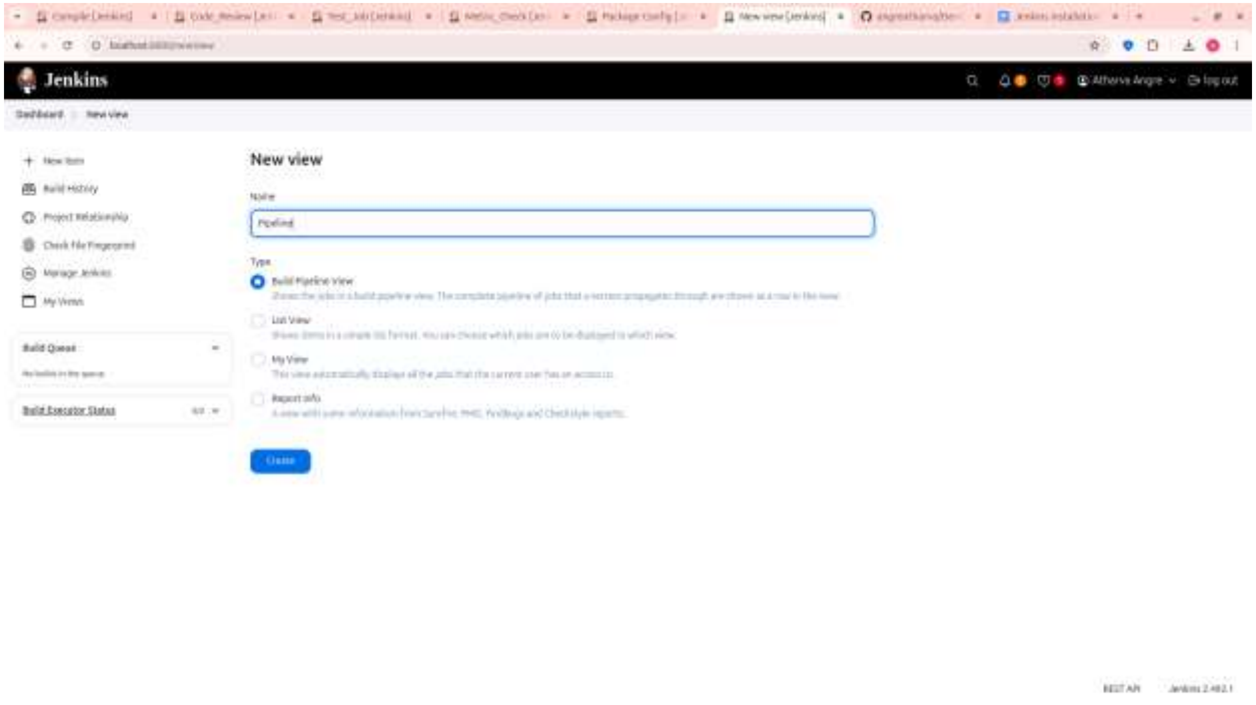
[✓ Trigger only if build is stable]
Trigger even if the build is unstable
Trigger even if the build fails
Always trigger, even if the build is aborted

- [ ] Build periodically
- [ ] GitHub hook trigger for GITScm polling
- [ ] Poll SCM

Environment

- [ ] Delete workspace before build starts
- [ ] Use secret text(s) or file(s)

10. In dashboard , I selected new view with name pipeline and type of Build Pipeline view



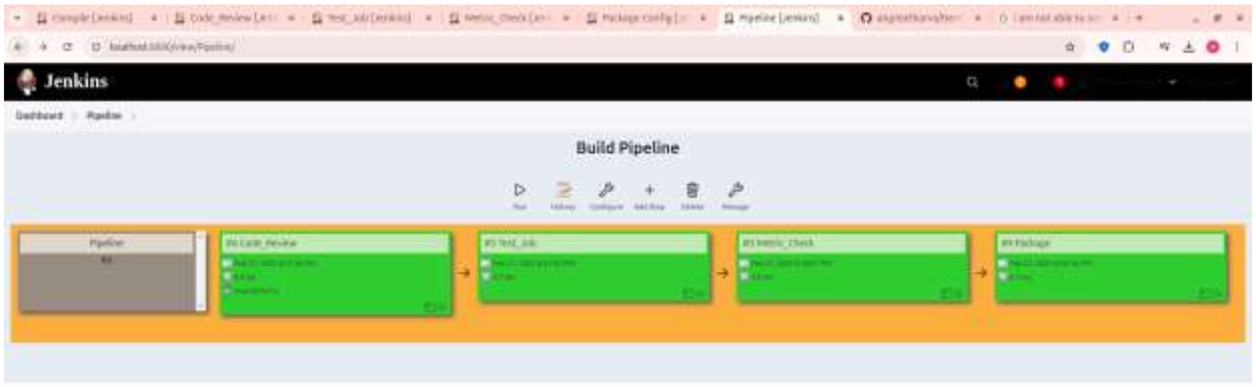
New view

Name: Pipeline

Type

[✓ Build Pipeline View]
List View
My View
Report Info

11. Output shows the CICD pipeline is successfully build



5) The pipelines can also be extended to running web tests and load tests. Explain How you would do the same using Jenkins?
Ans :
Jenkins automates web and load testing to ensure web applications function correctly and handle high traffic. Web testing simulates user interactions like clicks and form submissions using tools like Selenium, while load testing evaluates performance under heavy traffic using JMeter or Taurus. Necessary plugins are installed, and Jenkins jobs are configured to execute test scripts and generate reports. Tests can be scheduled or triggered after code changes, and results are analyzed within Jenkins to identify issues. This automation helps maintain reliability, efficiency, and scalability before deployment.

Observation : In this practical, we observed that the Jenkins pipeline serves as a structured roadmap guiding software from code to deployment. It breaks the process into stages like building, testing, and deployment, with each step acting as a checkpoint to ensure seamless progress. By automating these tasks, Jenkins simplifies workflow management and enables efficient tracking of changes.