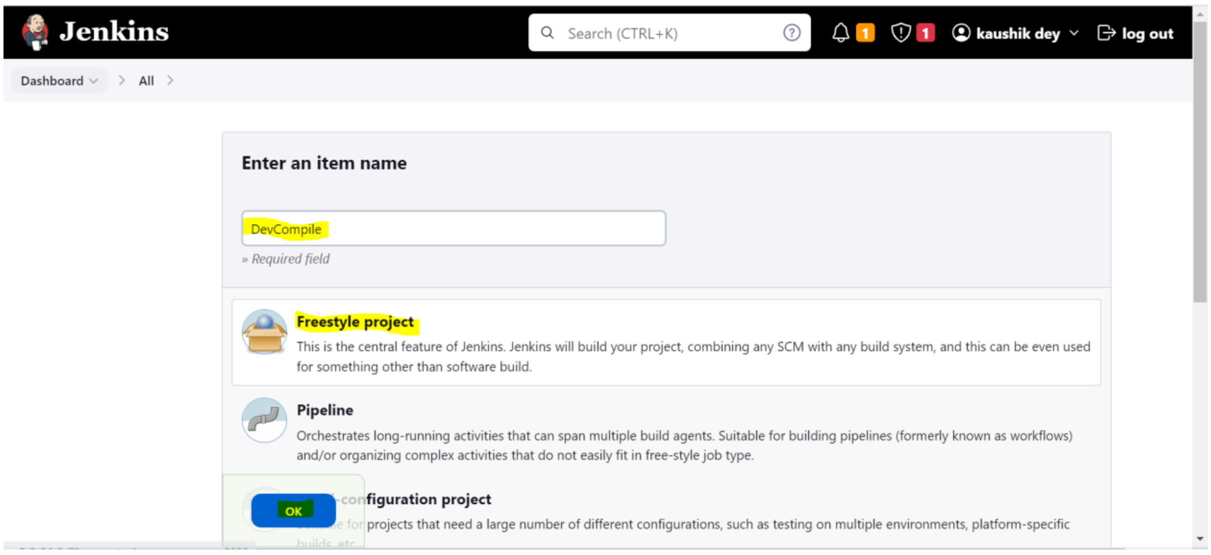
CONTINUOUS INTEGRATION WITH JENKINS

Jenkins build pipeline

Kaushik Dey

Step1 : Create Pipeline view using DevCompile and QAUnitTest.



Enter an item name

DevCompile

» Required field

Freestyle project
This is the central feature of Jenkins. Jenkins will build your project, combining any SCM with any build system, and this can be even used for something other than software build.

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.

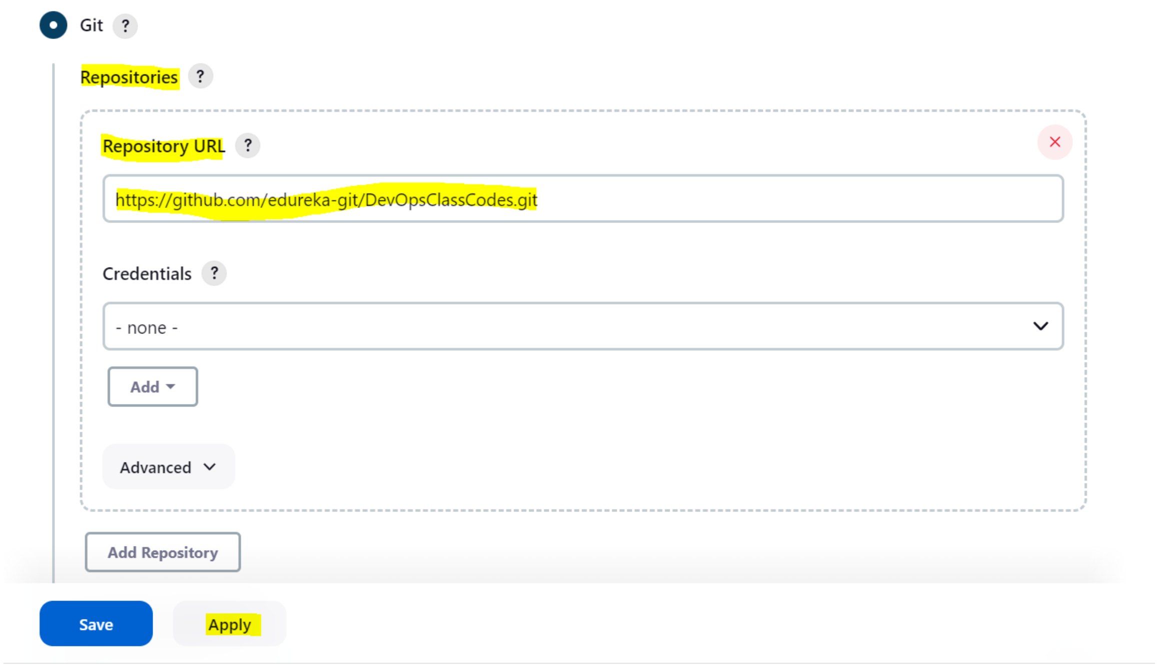
configuration project
projects that need a large number of different configurations, such as testing on multiple environments, platform-specific builds, etc.

Step 2 : After creating the free style project we must add our git repository , in my case using this following

git repo : https://github.com/edureka-git/DevOpsClassCodes.git (public repo)

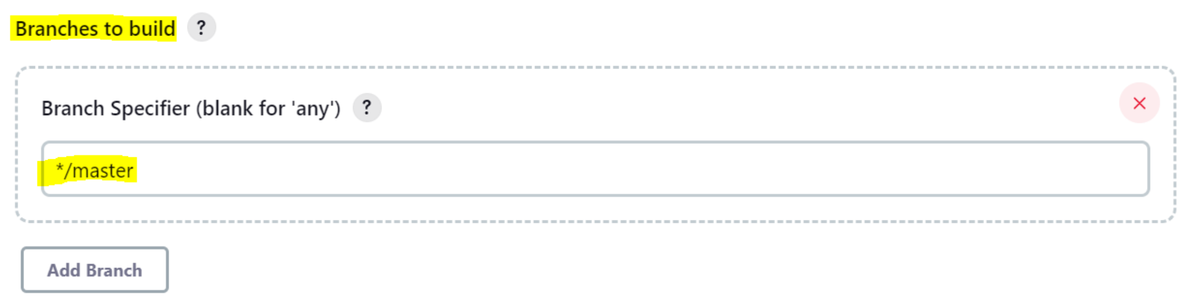
credentials None.

After that apply the changes.



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

Credentials: - none -



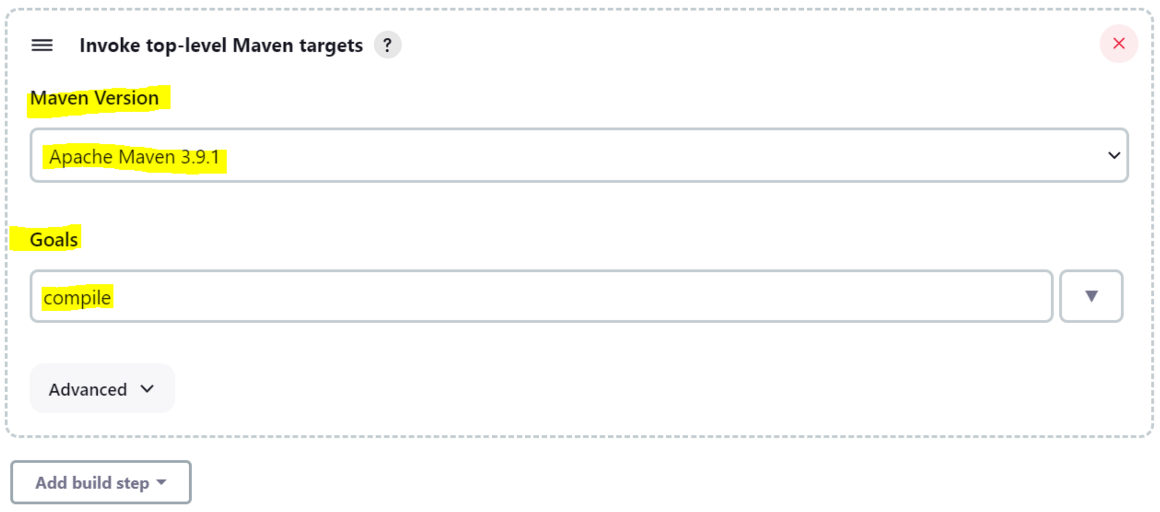
Branches to build ?

Branch Specifier (blank for 'any') ?

*/master

In later stages we can also add our branches.

Step 3 : now we can concentrate on build steps , in this case please choose **Invoke top-level Maven targets.** The screenshot is given below.



Invoke top-level Maven targets ?

Maven Version

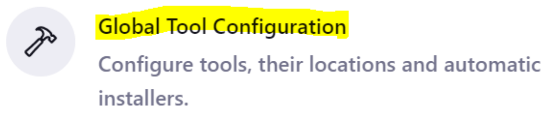
Apache Maven 3.9.1

Goals

compile

Note : In first time MVN is not installed in the Jenkins workspace , first we have to install it via global configuration, the screenshot is given below.

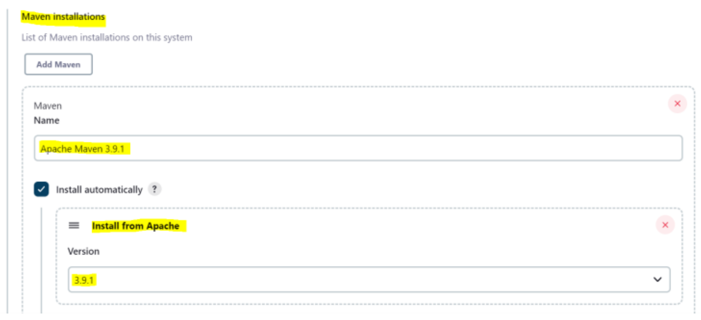
First navigate to Global Tool Configuration from Managed Plugins



Global Tool Configuration

Configure tools, their locations and automatic installers.

And then



Maven installations

List of Maven installations on this system

Maven Name

Apache Maven 3.9.1

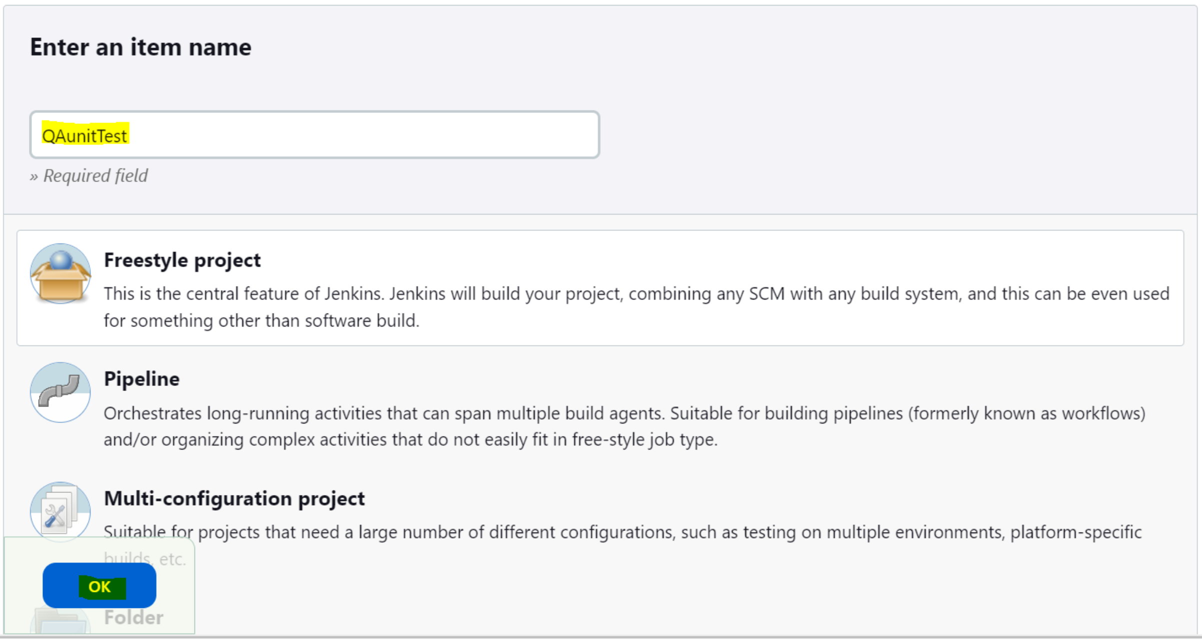
Install automatically ?

Install from Apache

Version

3.9.1

Step 4 : Create QAunitTest project and provide the same SCM path and execute it after DevCompile Action by setting build.



Enter an item name

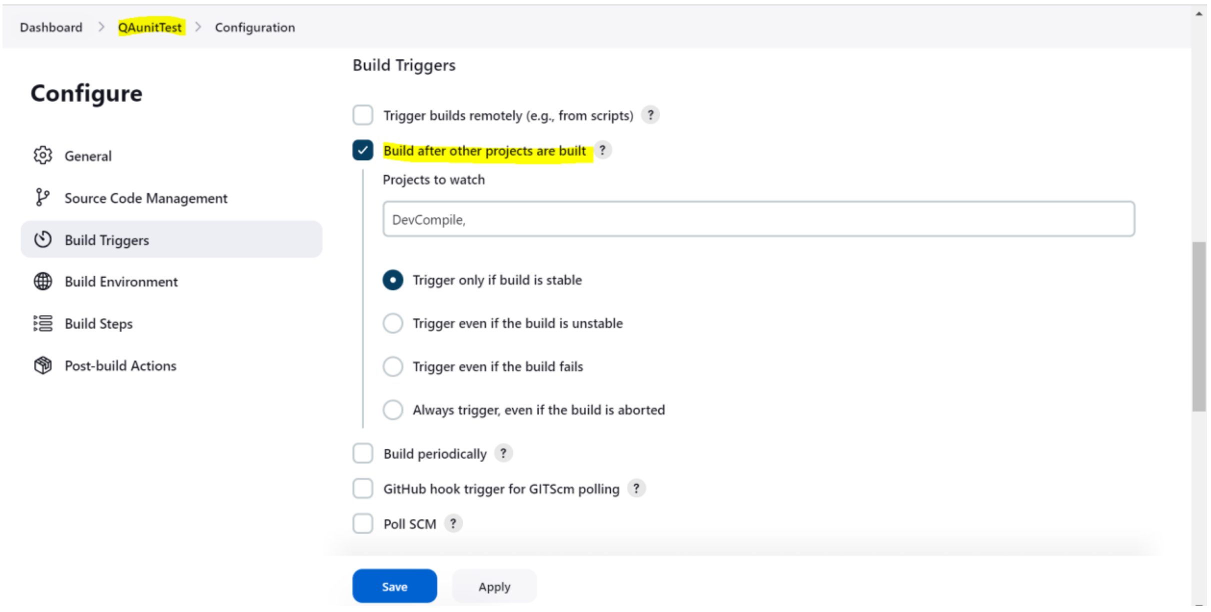
QAunitTest

» *Required field*

Freestyle project
This is the central feature of Jenkins. Jenkins will build your project, combining any SCM with any build system, and this can be even used for something other than software build.

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.

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



Build Triggers

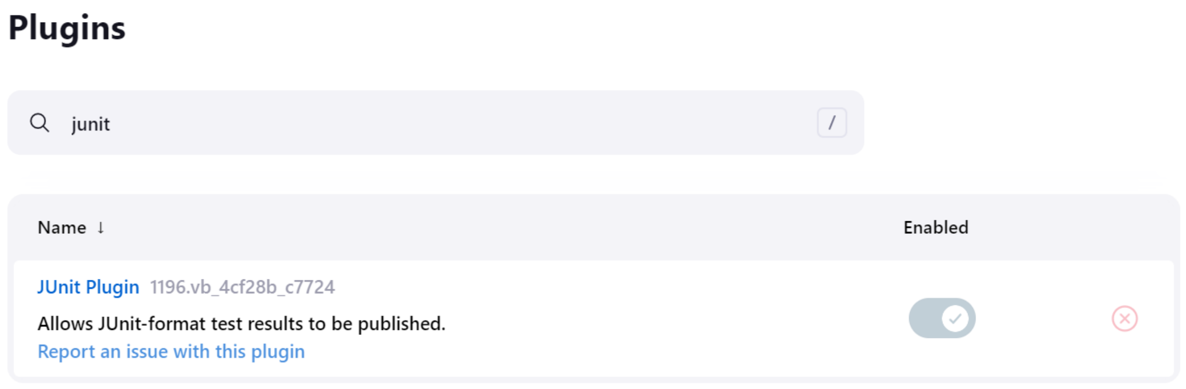
- [ ] Trigger builds remotely (e.g., from scripts)
- [✓] Build after other projects are built

Projects to watch: DevCompile,

- ● 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

Step 5 : Check Junit plugins are present or not, the screenshot is given below.

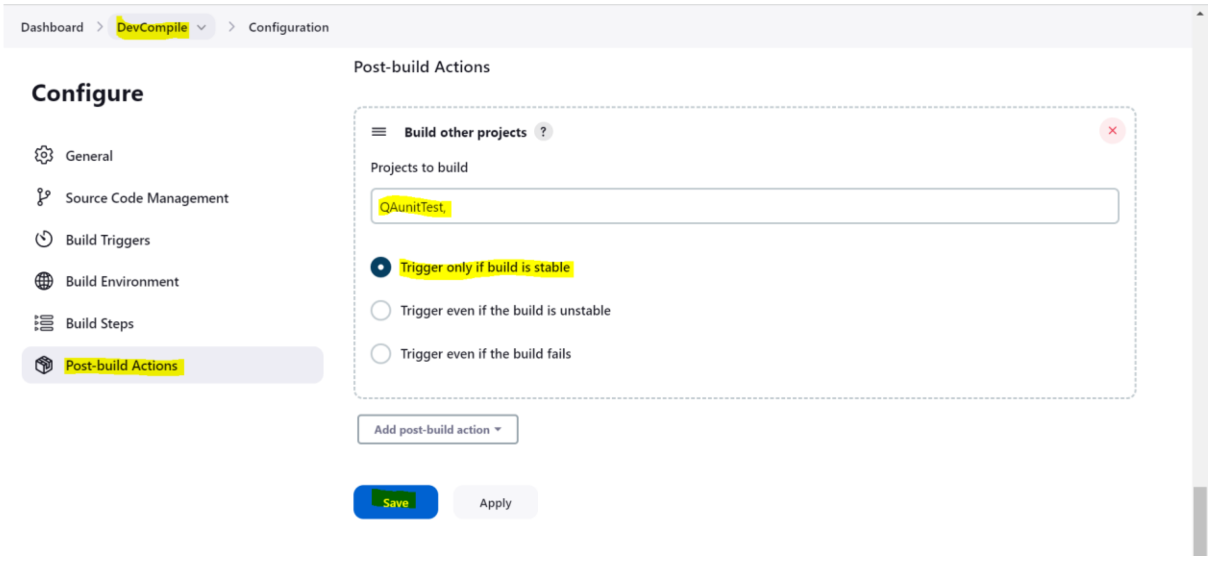


Plugins

junit

Name ↓	Enabled
JUnit Plugin 1196.vb_4cf28b_c7724 Allows JUnit-format test results to be published. Report an issue with this plugin	✓

Step 6 : Configure DevCompile project and set post-build action as QAUnitTest.


Dashboard > DevCompile > Configuration

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

Post-build Actions

Build other projects

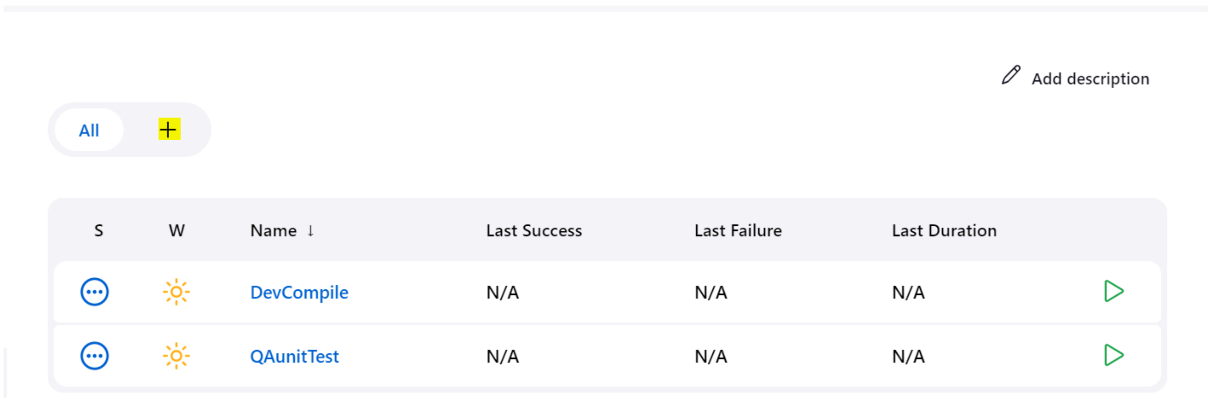
Projects to build: QAunitTest,

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

Step 7 :

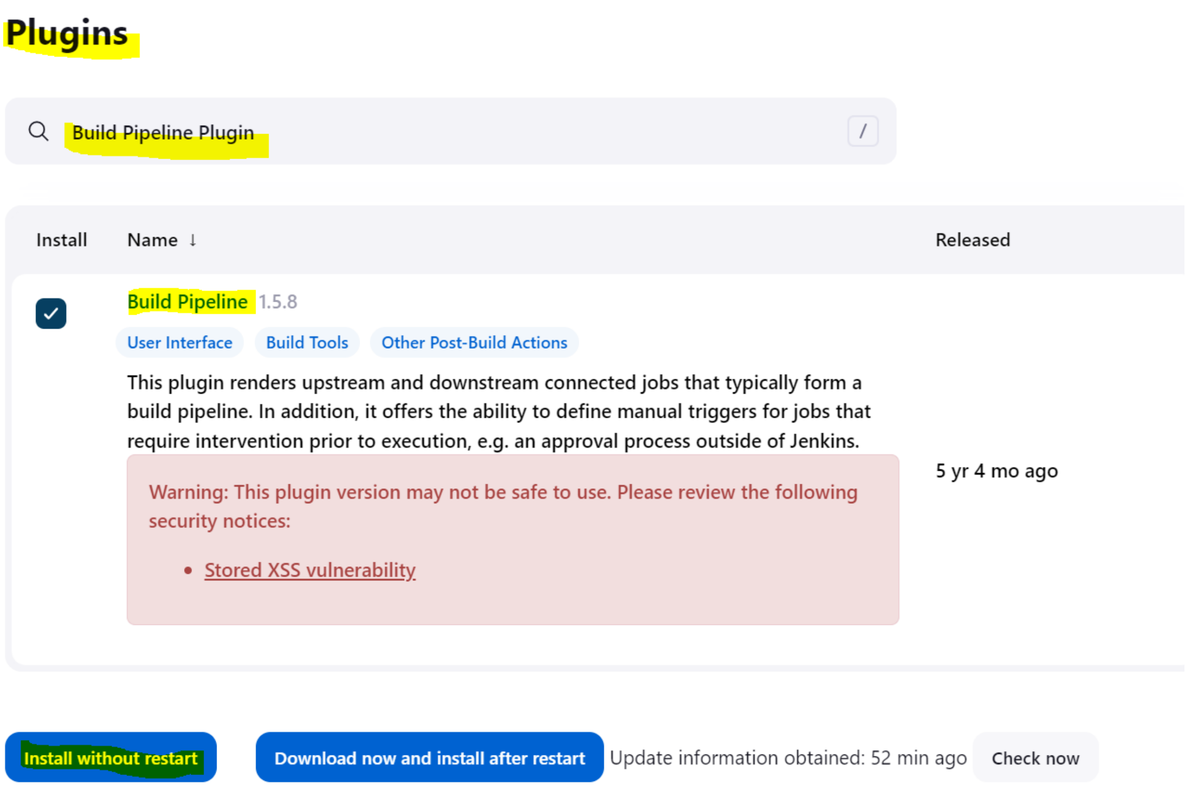
- ✓ Go to Jenkins Dashboard and click on the "+" button. That button is for adding a view.
- ✓ You will be redirected to the following screen.
- ✓ Give the View name and Select the build Pipeline View radio button, and press OK.

Screenshot 1 :



S	W	Name ↓	Last Success	Last Failure	Last Duration
		DevCompile	N/A	N/A	N/A
		QAunitTest	N/A	N/A	N/A

Screenshot 2: before going to the new view we have to install build pipeline plugins, so all jobs should be showing as in pipeline interface, the screenshot is given below.



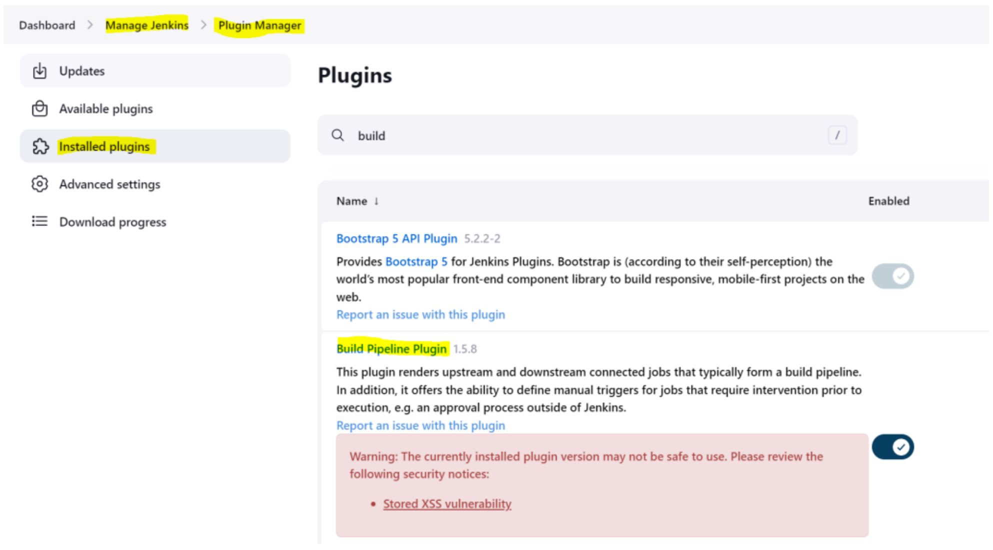
Plugins

Build Pipeline Plugin

Install	Name ↓	Released
✓	Build Pipeline 1.5.8 User Interface · Build Tools · Other Post-Build Actions This plugin renders upstream and downstream connected jobs that typically form a build pipeline. In addition, it offers the ability to define manual triggers for jobs that require intervention prior to execution, e.g. an approval process outside of Jenkins. Warning: This plugin version may not be safe to use. Please review the following security notices: • Stored XSS vulnerability	5 yr 4 mo ago

Update information obtained: 52 min ago

- ✓ Now we can check that plugins are installed or not, if it is installed it should be present in installed plugins. The screenshot is given below.

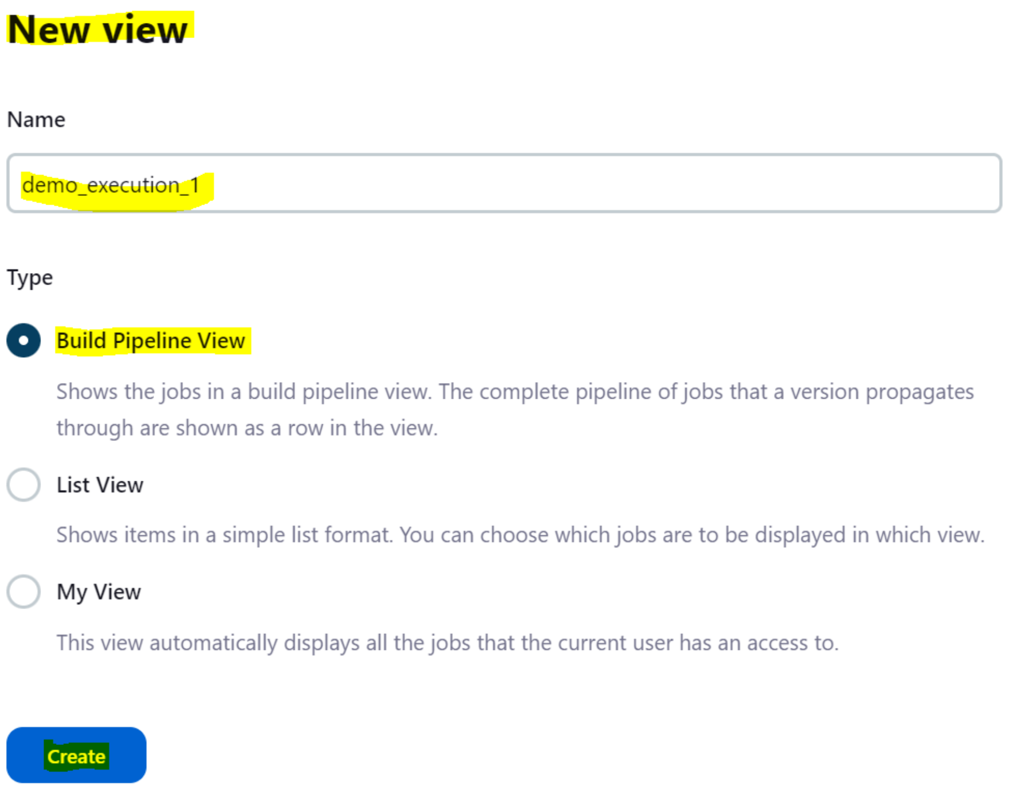


Plugins

build

Name ↓	Enabled
Bootstrap 5 API Plugin 5.2.2-2 Provides Bootstrap 5 for Jenkins Plugins. Bootstrap is (according to their self-perception) the world's most popular front-end component library to build responsive, mobile-first projects on the web. Report an issue with this plugin	✓
Build Pipeline Plugin 1.5.8 This plugin renders upstream and downstream connected jobs that typically form a build pipeline. In addition, it offers the ability to define manual triggers for jobs that require intervention prior to execution, e.g. an approval process outside of Jenkins. Report an issue with this plugin Warning: The currently installed plugin version may not be safe to use. Please review the following security notices: • Stored XSS vulnerability	✓

Screenshot 3 :



New view

Name

demo_execution_1

Type

● Build Pipeline View

Shows the jobs in a build pipeline view. The complete pipeline of jobs that a version propagates through are shown as a row in the view.

○ List View

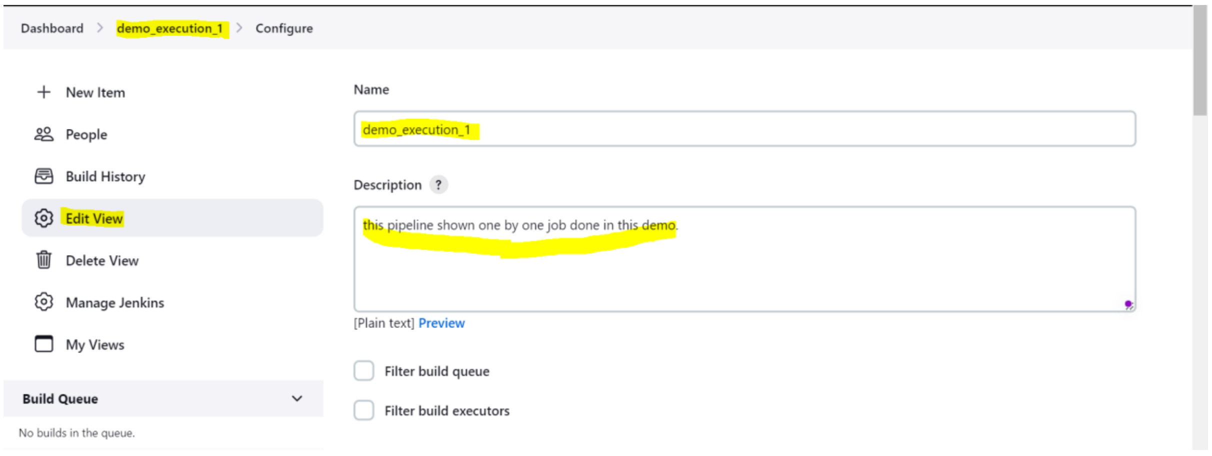
Shows items in a simple list format. You can choose which jobs are to be displayed in which view.

○ My View

This view automatically displays all the jobs that the current user has an access to.

Step 8 : Now we have to configure build pipeline view.

Screenshot 1:



Name

demo_execution_1

Description ?

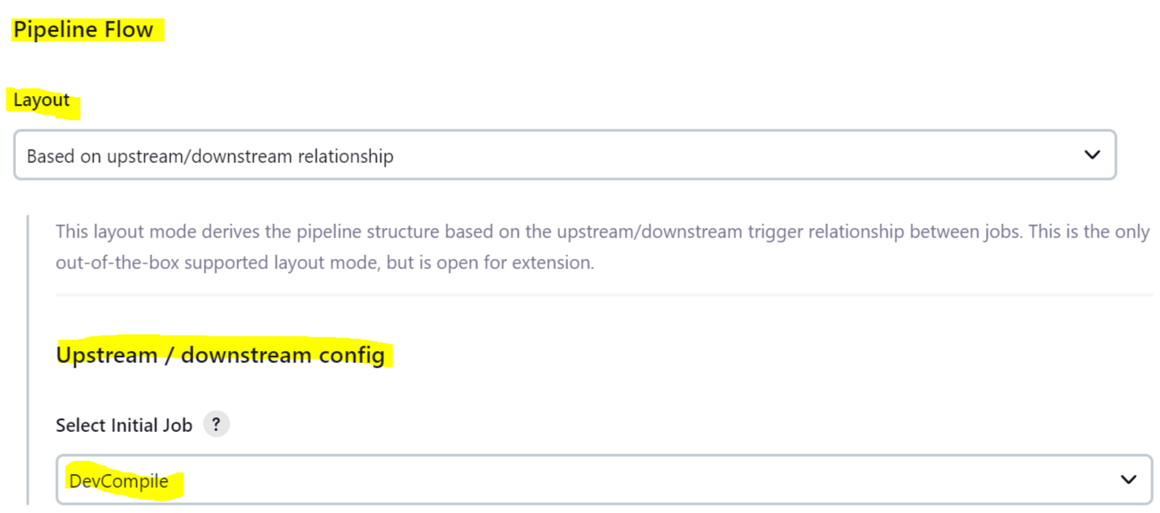
this pipeline shown one by one job done in this demo.

[Plain text] Preview

☐ Filter build queue

☐ Filter build executors

Screenshot 2:



Pipeline Flow

Layout

Based on upstream/downstream relationship

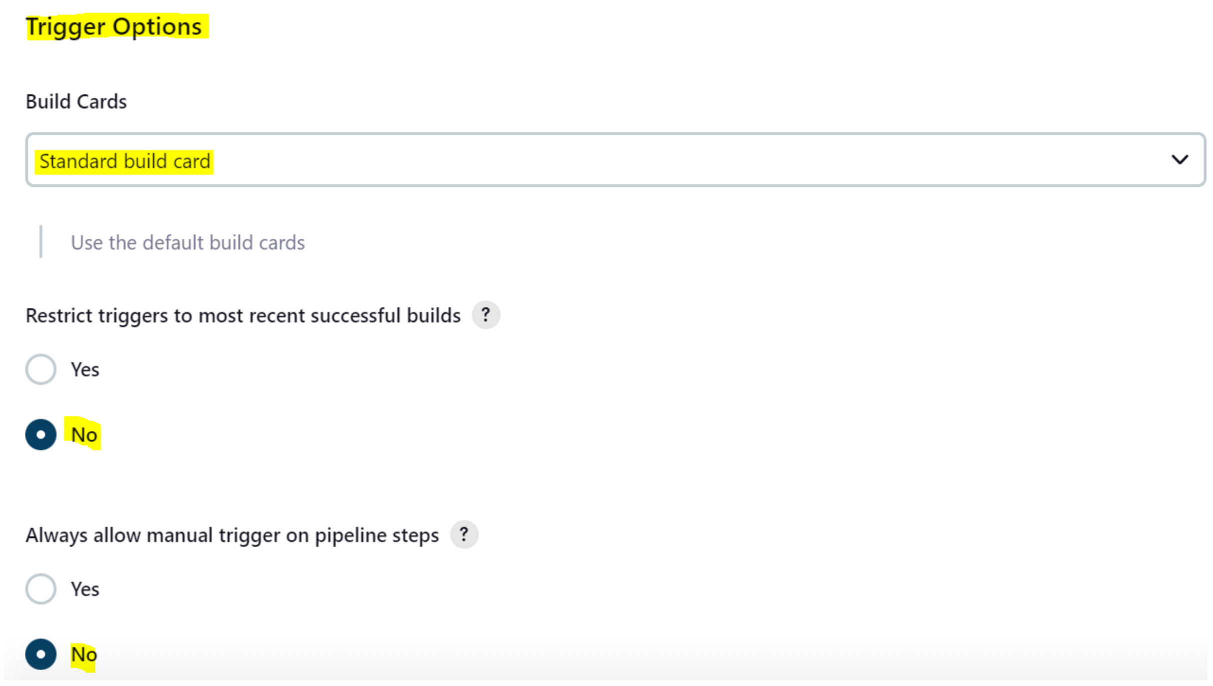
This layout mode derives the pipeline structure based on the upstream/downstream trigger relationship between jobs. This is the only out-of-the-box supported layout mode, but is open for extension.

Upstream / downstream config

Select Initial Job

DevCompile

Screenshot 3:



Trigger Options

Build Cards

Standard build card

Use the default build cards

Restrict triggers to most recent successful builds

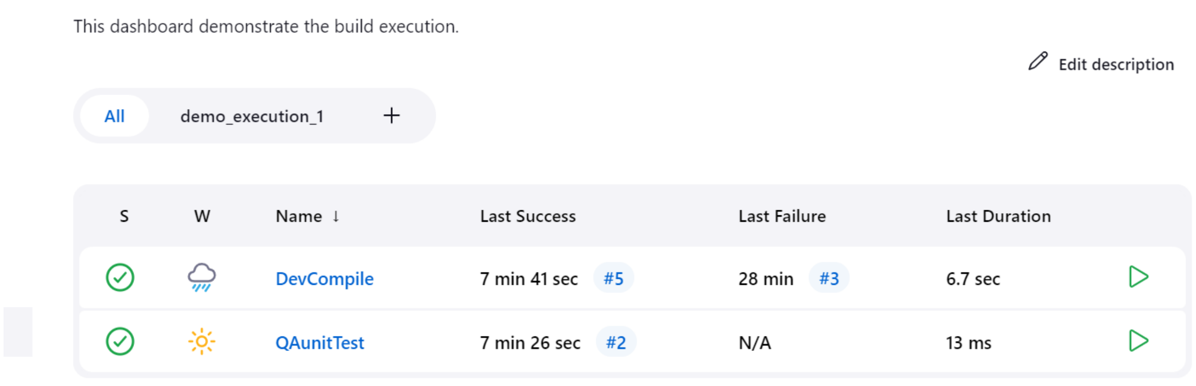
Yes [✓ No]

Always allow manual trigger on pipeline steps

Yes [✓ No]

Then click on OK button to proceeds.

Step 9 : our jobs (Upstream and downstream) done successfully. We can see the following output.

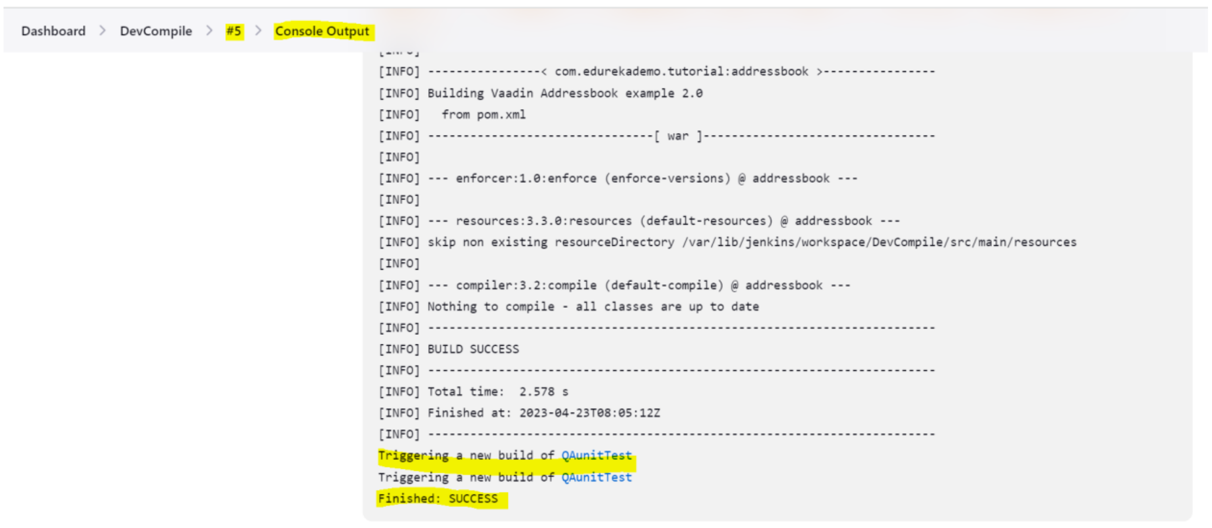


This dashboard demonstrate the build execution.

S	W	Name ↓	Last Success	Last Failure	Last Duration	
✓	🌧	DevCompile	7 min 41 sec #5	28 min #3	6.7 sec	▷
✓	☀	QAunitTest	7 min 26 sec #2	N/A	13 ms	▷

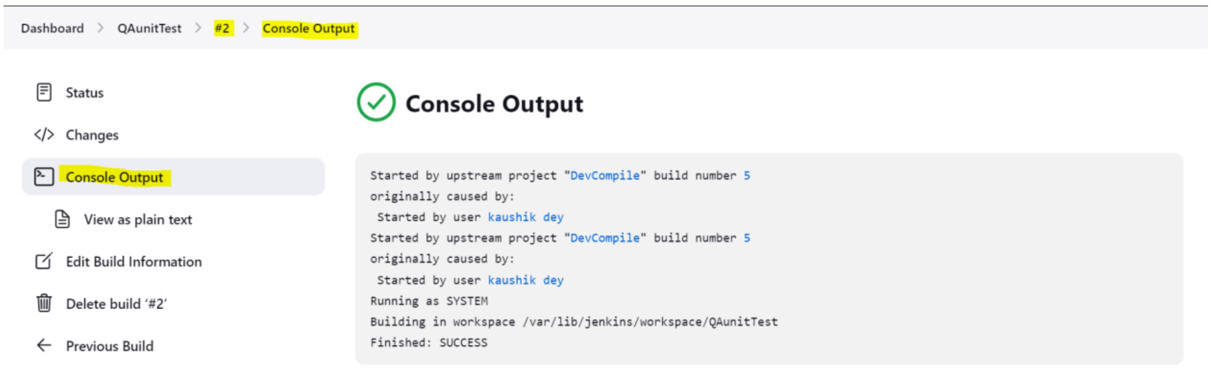
Step 10 : we can see the console output of this two jobs , the screen shot is given below.

Screenshot1 : This is my build no.5 and the console output is that. (DevCompile). Also noticed that **Triggering a new build of QAUnitTest**



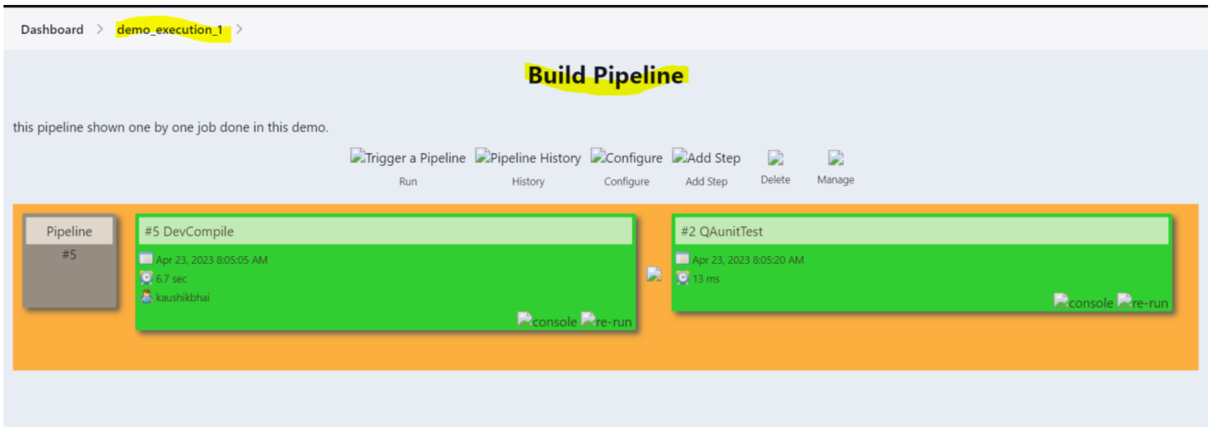
```
[INFO] ---------------< com.edurekademo.tutorial:addressbook >---------------
[INFO] Building Vaadin Addressbook example 2.0
[INFO]   from pom.xml
[INFO] --------------------------------[ war ]---------------------------------
[INFO]
[INFO] --- enforcer:1.0:enforce (enforce-versions) @ addressbook ---
[INFO]
[INFO] --- resources:3.3.0:resources (default-resources) @ addressbook ---
[INFO] skip non existing resourceDirectory /var/lib/jenkins/workspace/DevCompile/src/main/resources
[INFO]
[INFO] --- compiler:3.2:compile (default-compile) @ addressbook ---
[INFO] Nothing to compile - all classes are up to date
[INFO] ------------------------------------------------------------------------
[INFO] BUILD SUCCESS
[INFO] ------------------------------------------------------------------------
[INFO] Total time:  2.578 s
[INFO] Finished at: 2023-04-23T08:05:12Z
[INFO] ------------------------------------------------------------------------
Triggering a new build of QAunitTest
Triggering a new build of QAunitTest
Finished: SUCCESS
```

Screenshot2 : This is my build no.5 and the console output is that. (QAUnitTest)



```
Started by upstream project "DevCompile" build number 5
originally caused by:
 Started by user kaushik dey
Started by upstream project "DevCompile" build number 5
originally caused by:
 Started by user kaushik dey
Running as SYSTEM
Building in workspace /var/lib/jenkins/workspace/QAunitTest
Finished: SUCCESS
```

Step 11: Now we can see the pipeline view of this job, just click on demo_execution_1. The screenshot is given below.



Conclusion: Successfully build the pipeline with upstream and downstream the project.