

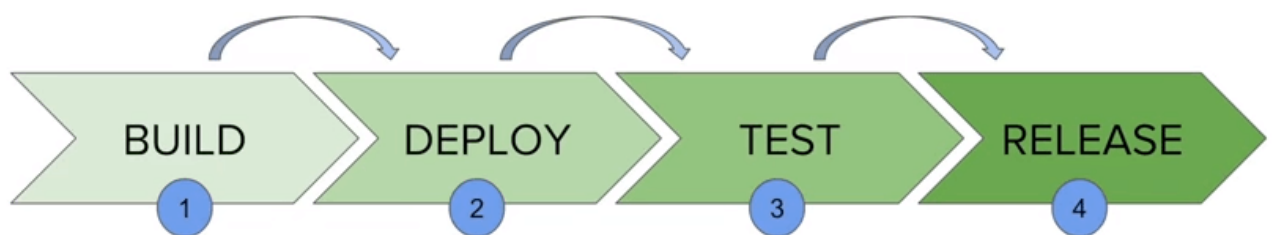
What is Jenkins Pipeline?

In Jenkins, a pipeline is a group of events or jobs which are interlinked with one another in a sequence.

In simple words, Jenkins Pipeline is a combination of plugins that support the integration and implementation of **continuous delivery pipelines** using Jenkins. A pipeline has an extensible automation server for creating simple or complex delivery pipelines "as code," via pipeline DSL (Domain-specific Language).

What is Continuous Delivery Pipelines? How it Works?

In a Jenkins pipeline, every job or event has some sort of dependency on at least one or more events.



The picture above represents a continuous delivery pipeline in Jenkins. It contains a group of states called build, deploy, test and release. These events are interlinked with each other. Every state has its events, which work in a sequence called a continuous delivery pipeline.

A continuous delivery pipeline is an automated expression to display your process for getting software for version control. Thus, every change made in your software goes through a number of complex processes on its way to being released. It also involves developing the software in a reliable and repeatable manner, and progression of the built software through multiple stages of testing and deployment.

Why Use Jenkin's Pipeline?

Jenkins is an open continuous integration server which has the ability to support the automation of software development processes. You can create multiple automation jobs with the help of use cases, and run them as a Jenkins pipeline.

Here are the reasons why you use should use Jenkins pipeline:

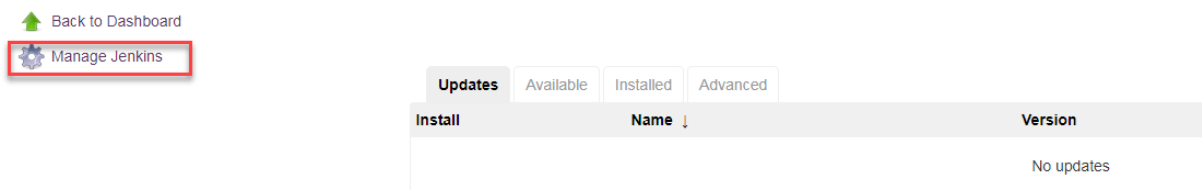
- Jenkins pipeline is implemented as a code which allows multiple users to edit and execute the pipeline process.
- Pipelines are robust. So if your server undergoes an unforeseen restart, the pipeline will be automatically resumed.
- You can pause the pipeline process and make it wait to resume until there is an input from the user.
- Jenkins Pipelines support big projects. You can run multiple jobs, and even use pipelines in a loop.

Install Build Pipeline Plugin in Jenkins

With the **build pipeline** plugin, you can create a pipeline view of incoming and outgoing jobs, and create triggers which require manual intervention.

Here is how you can install the **build pipeline** plugin in your Jenkins:

Step 1) The settings for the plugin can be found under **Manage Jenkins > Manage Plugins**.



If you have already installed the plugin, it is shown under the installed tab.

Updates Available Installed Advanced			
nabled		Name ↓	Version Previously installed
<input checked="" type="checkbox"/>	Ant Plugin	Adds Apache Ant support to Jenkins	1.8
<input checked="" type="checkbox"/>	Apache HttpComponents Client 4.x API Plugin	Bundles Apache HttpComponents Client 4.x and allows it to be used by Jenkins plugins.	4.5.5-3.0
<input checked="" type="checkbox"/>	Authentication Tokens API Plugin	This plugin provides an API for converting credentials into authentication tokens in Jenkins.	1.3
<input checked="" type="checkbox"/>	bouncycastle API Plugin	This plugin provides an stable API to Bouncy Castle related tasks.	2.16.3
<input checked="" type="checkbox"/>	Branch API Plugin	This plugin provides an API for multiple branch based projects.	2.0.20
1 <input checked="" type="checkbox"/>	Build Pipeline Plugin	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	1.5.8

Step 2) If you do not have the plugin previously installed, it shows up under the **Available** tab.

Once you have successfully installed the **build pipeline** plugin in your Jenkins, follow these steps to create your Jenkins pipeline:

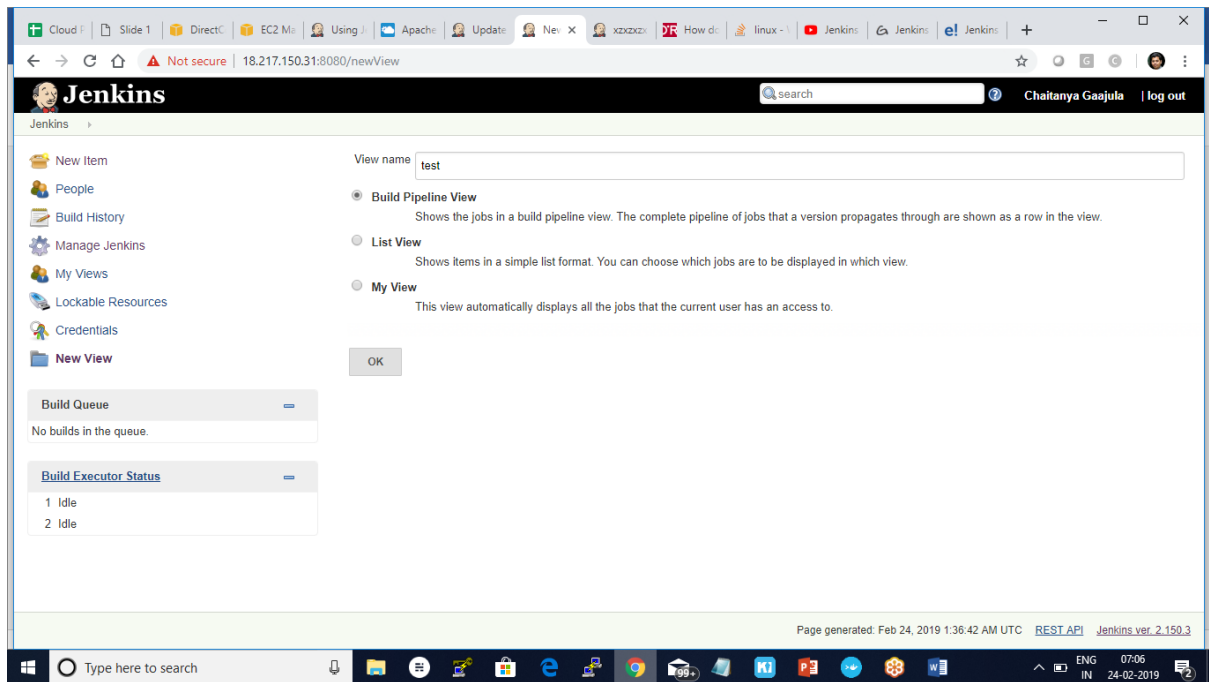
How to Create Jenkins Pipeline

Once you are logged in to your Jenkins dashboard:

Step 1) Click on the "+" button on the left-hand side of your Jenkins dashboard to create a pipeline.

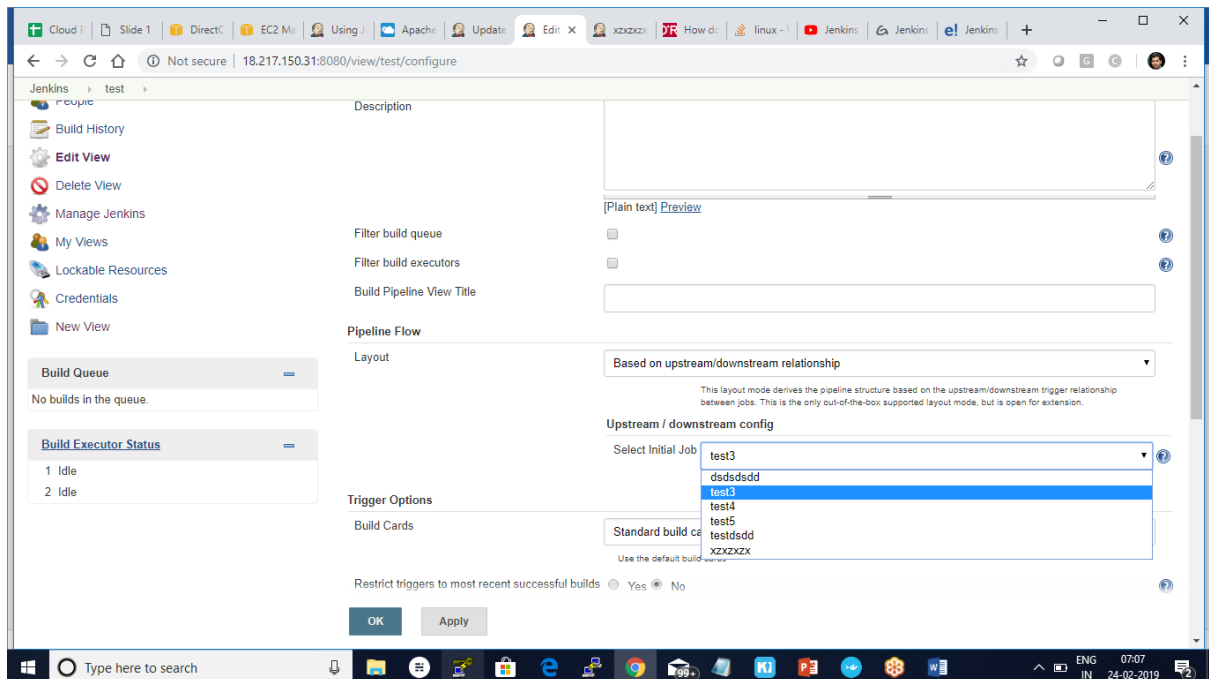
Step 2)

1. You will be asked to give a name to the pipeline view. We shall call it "**Test Pipeline**" for the duration of this demo.
2. Select **Build a pipeline view** under **options**
3. Click **ok**



Step 4) In the next page, you will be asked for some more details to configure your Jenkins pipeline. Just accept the default settings, and make sure you choose the first job under the settings.

Click on **Apply** and then **OK**.

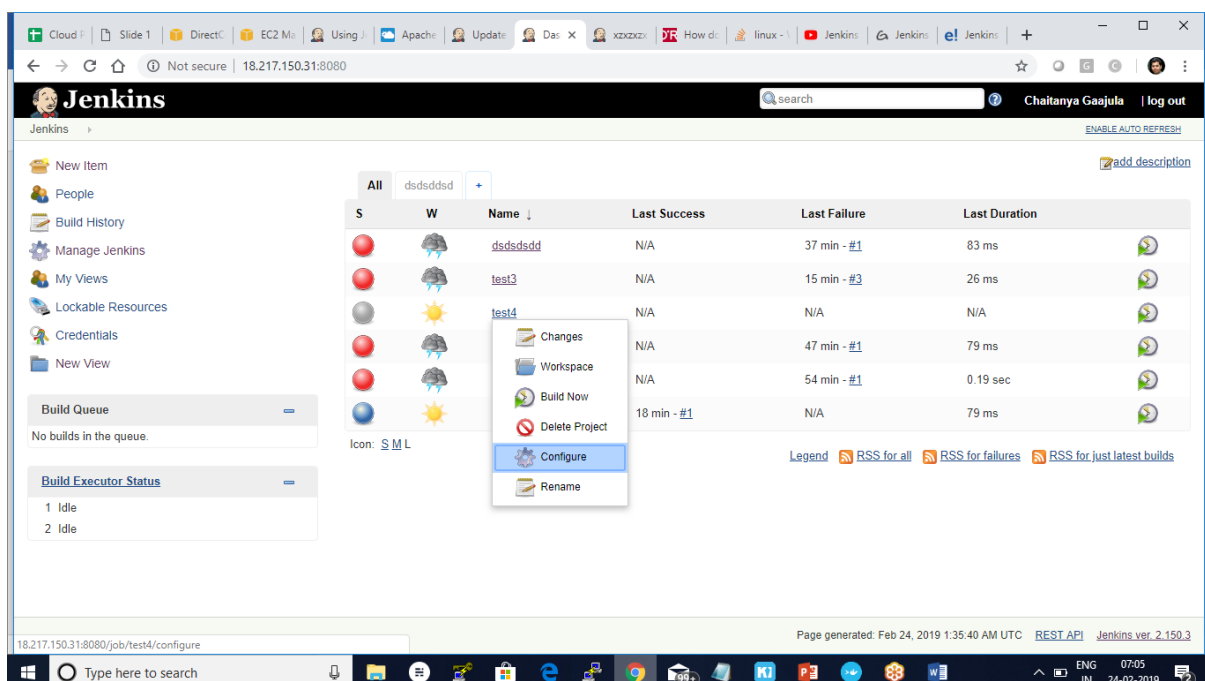


This will show you the sample pipeline view of your item, as given below:

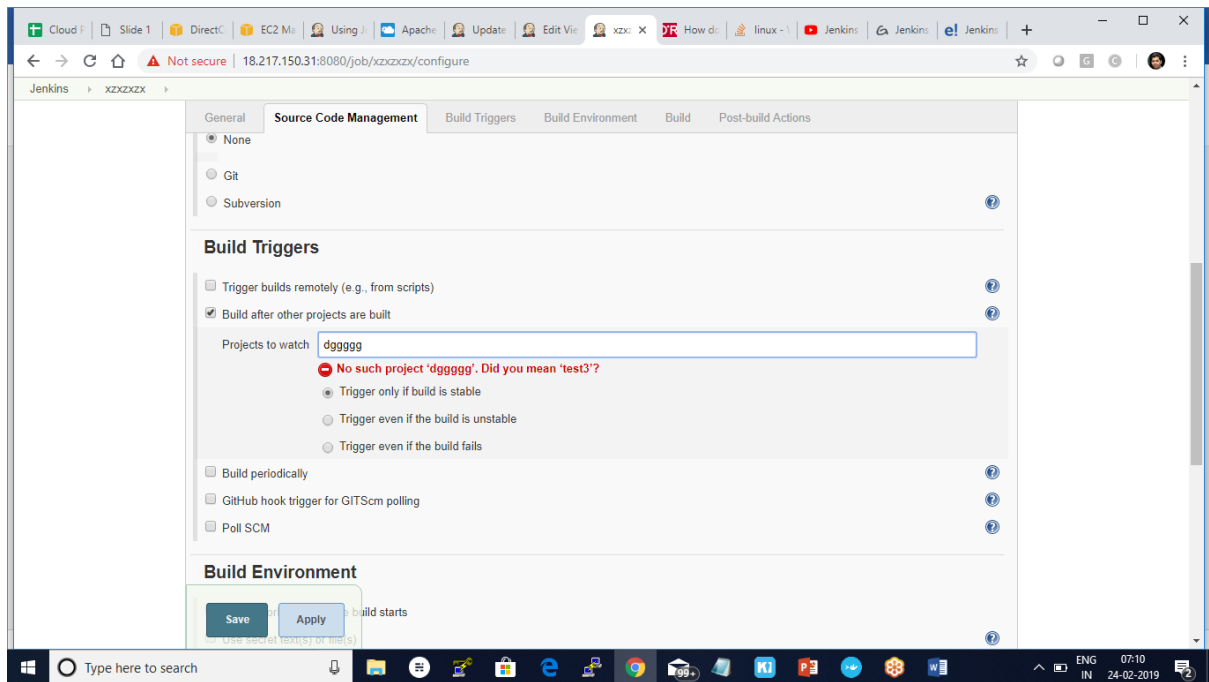


Running a pipeline build

Step 1) For running a pipeline build, you need to chain your jobs first. For this, go to your first job and click on configure.



Step 2) Now, under **Build Triggers**, check the **Build after other projects are built** option.



Thus, a chain for all your jobs has been created.

Step 3) Install the **Build Pipeline view** plugin if you don't have it installed already.

Step 4) Go to your Jenkins dashboard and create a view by clicking on the "+" button. Select the **Build Pipeline View** option and click **OK**.

Step 5) Under **Pipeline view configuration**, locate **Pipeline Flow**.

Under **Pipeline flow**, select the initial job to run.

When the Jenkins pipeline is running, you can check its status with the help of Red and Green status symbols. Red means the pipeline has failed, while green indicates success.

In this example, we see that the button is green. Hence, the pipeline is successful.

Running Jenkins pipeline

Click on **Run** to run the Jenkins pipeline. It will look something like this:

See below for a sample pipeline view.

		build	test: integration-&-quality	test: functional	test: load-&-security	approval	deploy: prod
Average stage times: (Average full run time: ~5s)		836ms	20min 43s	9ms	7ms	89ms	5ms
#17	Sep 22 15:05 No Changes Retry Download	538ms	10s	10ms	8ms	72ms (paused for 7s)	4ms
#16	Sep 22 15:04 No Changes Retry Download	479ms	6s	9ms	9ms	74ms (paused for 6s)	5ms
#15	Sep 22 15:03 No Changes Retry Download	922ms	6s	10ms	9ms failed		
#14	Sep 22 15:03 No Changes Retry Download	1s	8s	12ms	9ms	80ms (paused for 6s)	5ms
#13	Sep 22 15:02 No Changes Download	942ms	9s	13ms failed			
#12	Sep 22 15:02 No Changes Retry Download	1s	6s	13ms	11ms	111ms (paused for 6s) aborted	

Best Practices using Jenkins Pipeline:

- Use the genuine Jenkins Pipeline
- Develop your pipeline as code
- Any non-setup work in your pipeline should occur within a stage block.
- Any material work in a pipeline must be performed within a node block.
- Don't use input within a node block.
- Never set environment variables with env global variable
- Wrap your inputs in a timeout