**JENKINS PLUG-IN MANAGEMENT**

**Plug-in is piece of software once installed it is going to add one feature in Jenkins.**

**Plugins Names**

**1.Next Build Number**

**2.Maven**

**3.Deploy to Container Plugin**

**4.[WebSphere Deployer](https://plugins.jenkins.io/websphere-deployer" \t "_blank)**

**5.Safe Restart**

**6.JoCoCo plugin**

**7.Audit Trail**

**8.Schdule Build**

**9.Build Name and Description Setter**

**10.Publish Over SSH**

**11.Job Configuration History**

**12.Job Import Plug-in**

**13.GitHub Webhook**

**14.Maven Integration**

**15.SonarQube Scanner**

**16.Email Extension**

**17.Job Configuration history**

**18. Next Build Number**

**19.Build Name an description setter**

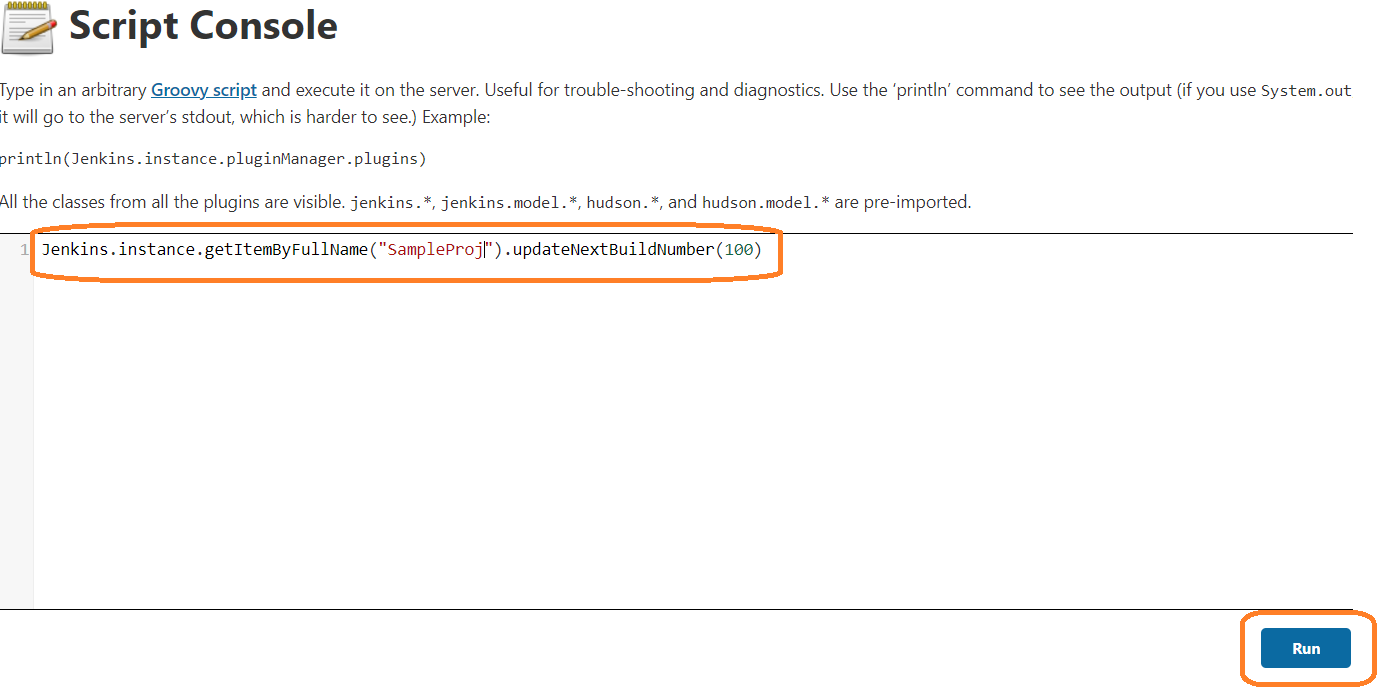
**20.Blue ocean**

**21.ThinBackup**

We can change *nextBuildNumber* two ways(**(under each job level)**

Sets the build number Jenkins will use for a job's next build

1.Manage Jenkins 🡪 Script Console



*Jenkins.instance.getItemByFullName("<****PROJECT-NAME****>").updateNextBuildNumber(<****NUMBER****>*

* Click *Run* and test

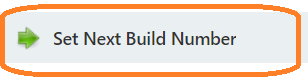
**(OR)**

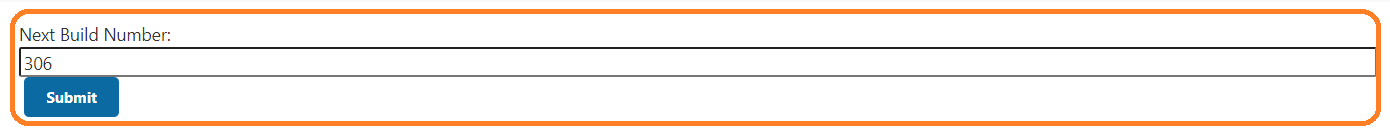
We can change using home directory

BuildNumber will be store in Jenkins home directory under **jobs**

Install “**Next Build Number”** Plugin

Set value manually and test





**Note:**

We should not suppose to give less than current build number

**2.Maven**

We can create job for only java project which are build tool “Maven”

Designed specific for java project

Having some specific feature

Installing Maven in Jenkins server

**Manage Jenkins -> Global Tool Configuration**

Under Maven section give required version details for install

**3.Deploy to Container Plugin**

Deploy application to TOMCAT server

This plugin allows you to deploy a war to a container after a successful build

*Manage Jenkins🡪 Manage Plugin 🡪 Available*

4. [**WebSphere Deployer**](https://plugins.jenkins.io/websphere-deployer)

Deploys your web artifacts automatically to IBM WebSphere Application Server and IBM WebSphere Liberty Profile

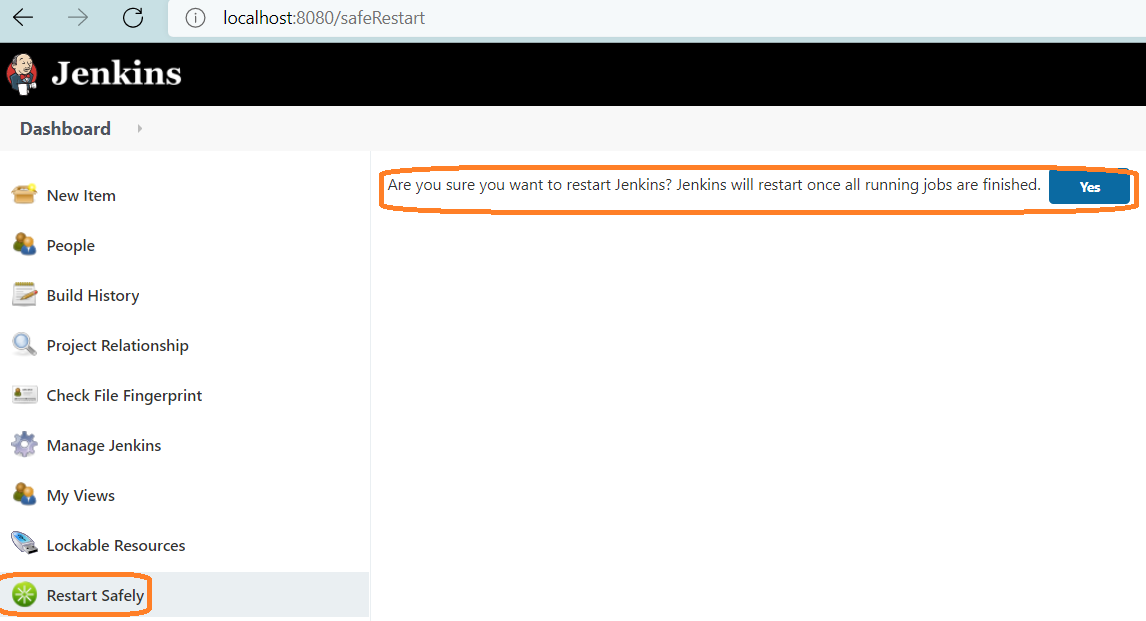
*Manage Jenkins🡪 Manage Plugin 🡪 Available*

**5.Safe Restart**

If we don’t have access to restart JENKINS server using LINUX commands in this case we can restart using URL

By using URL we can restart Jenkins

*Manage Jenkins🡪 Manage Plugin 🡪* **Safe Restart**

****

**URL:**

**http://<Public-IP> :<Port>/<context-path>**

[**http://52.66.151.79:8080/safeRestart**](http://52.66.151.79:8080/safeRestart)

****

**#saftRestart:** It will wait until running jobs should complete

<http://52.66.151.79:8080/restart>

****

**#restart:** Forcefully restart, it won’t wait running jobs should complete

Other option, we can do manually

[**http://localhost:8080/safeRestart**](http://localhost:8080/safeRestart)

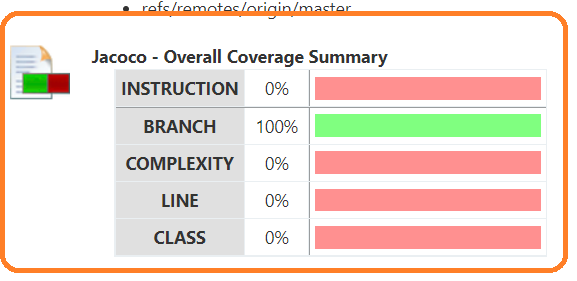
**http://localhost:8080/restart**

**6.JaCoCo plugin (Java Code Coverage)**

To Generate the Test report, we will use this plug-in **(under each project level)**

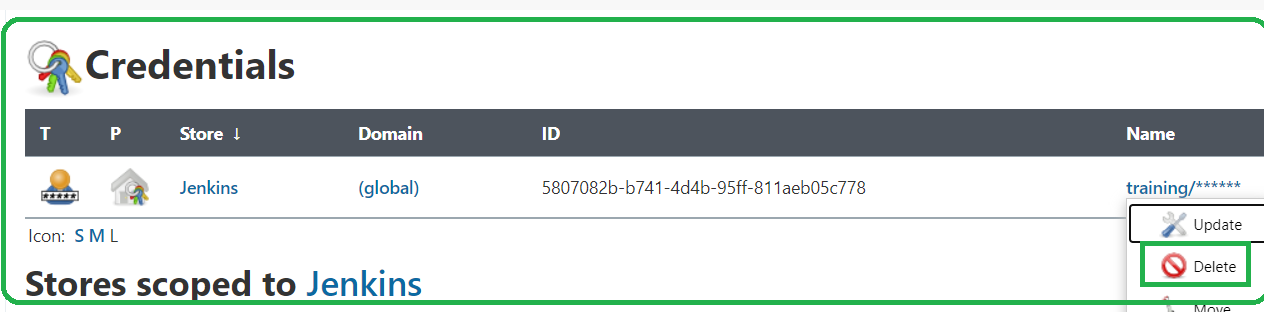
Under each Job level we can see code coverage result.

This plugin integrates ***JaCoCo code coverage report to Jenkins***



**\*\*\*\*\*Delete credentials in Jenkins**

*Manage Jenkins🡪 Manage Credentials*

**

**7.Audit Trail**

Keep a log of who performed particular Jenkins operations, such as configuring jobs.

Install Audit Trail plug-in from **manage Jenkins**

In DevOps culture many DevOps Engineers (DE) will be running jobs based on requirement to rack all information.

*Manage Jenkins🡪 Configure System(to configure global setting and path)*



We have to give location where log file should store

Example log location:

/root/.jenkins/logs/**audittrail.log**

/root/.jenkins/logs/**audittrail.log.0**

/root/.jenkins/logs/**audittrail.log.1**

Once the file size reaches it will add on more file until file count reaches

**8.Schedule Build**

If we want to run job (TODAY OR Specific time/day for temporary) we can use this plugin

**Example:** I want to run job today @1:00 PM IST, today alone

We have to give Jenkins’s server time (where the Jenkins installed) based on that we have to update schedule time and test

*Manage Jenkins🡪 Manage Plugin 🡪 Schedule Build Plugin*



**9.Build Name and Description Setter**

This plug-in sets the display name and description of a build to something other than #1, #2, #3

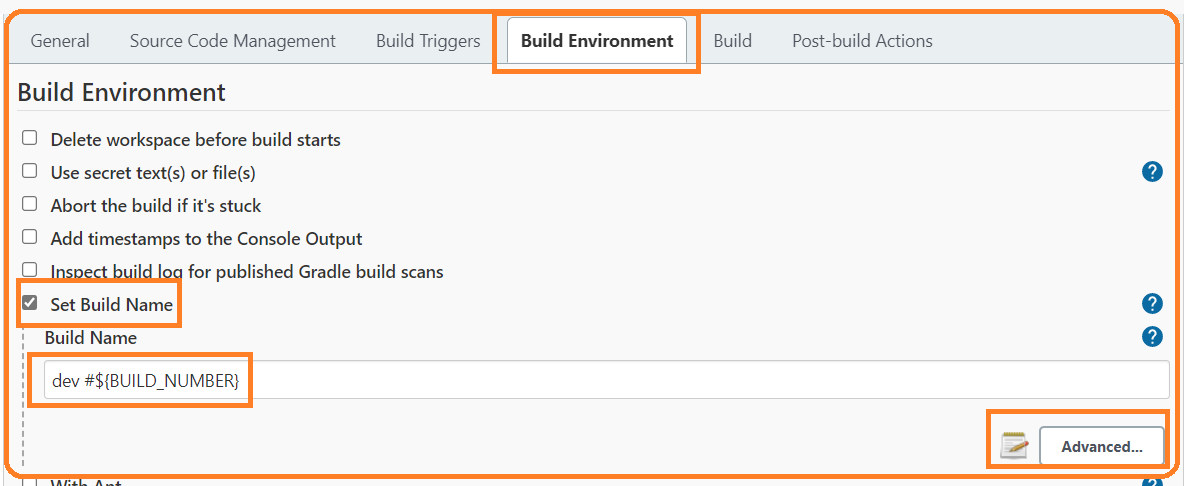
If we want to add environment name instead of build number, we will use this plugin

*Manage Jenkins🡪 Manage Plugin 🡪 Build Name and Description Setter*

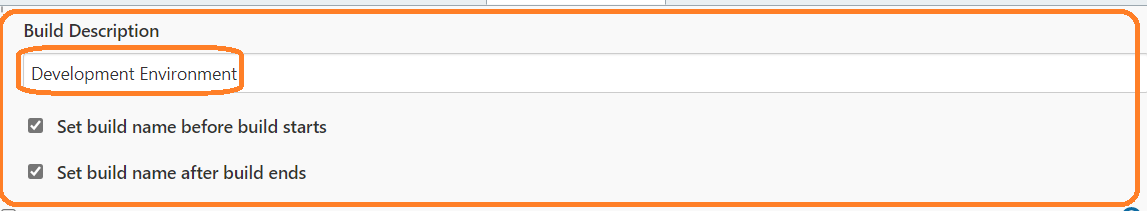
*Once Plug in installed*

*Go to particular job*

*Example: Dashbord -> JobName -> Configure -> <Under Build Trigger Tab>*

**

*Click on* **“Advanced”** *tab*

**

**Save and Test**

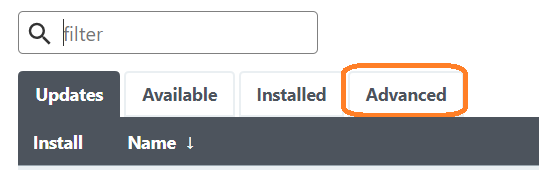
*Syntax:* <Shot/Full Environment Name> #${BUILD\_NUMBER}

Dev #${BUILD\_NUMBER}

* **\*\*\*\*\*To install External Plugins, we will use “Advanced” Tab**

Example: **Urban Code Deploy**

*Manage Jenkins🡪 Manage Plugin*



**9.Publish Over SSH**

Send build artifacts over SSH

Going to upload STATIC content

To deploy STATIC content to the server we will use the ***“Publish Over SSH”*** plugin

* **\*\*\*\*\*To get Jenkins password (if forgot)**

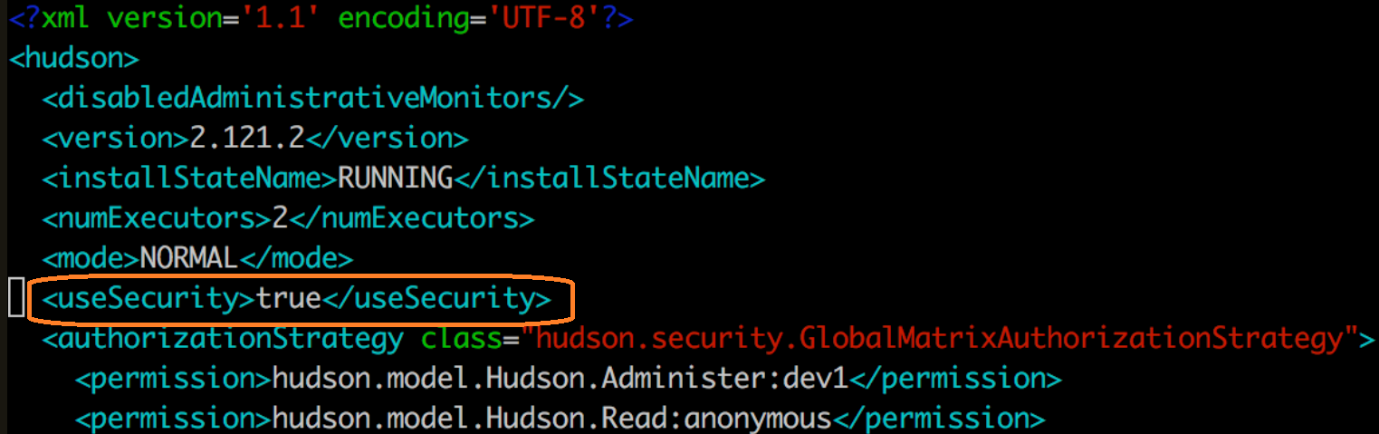
Let's say you have forgotten Jenkins admin password. How to set or retrieve the admin password?

Here below are the steps for resetting Jenkins admin password:

1. Edit the Jenkins config.xml under /var/lib/jenkins to disable securityFlag by executing below command:  
<Jenkins Home Directory>/config.xml

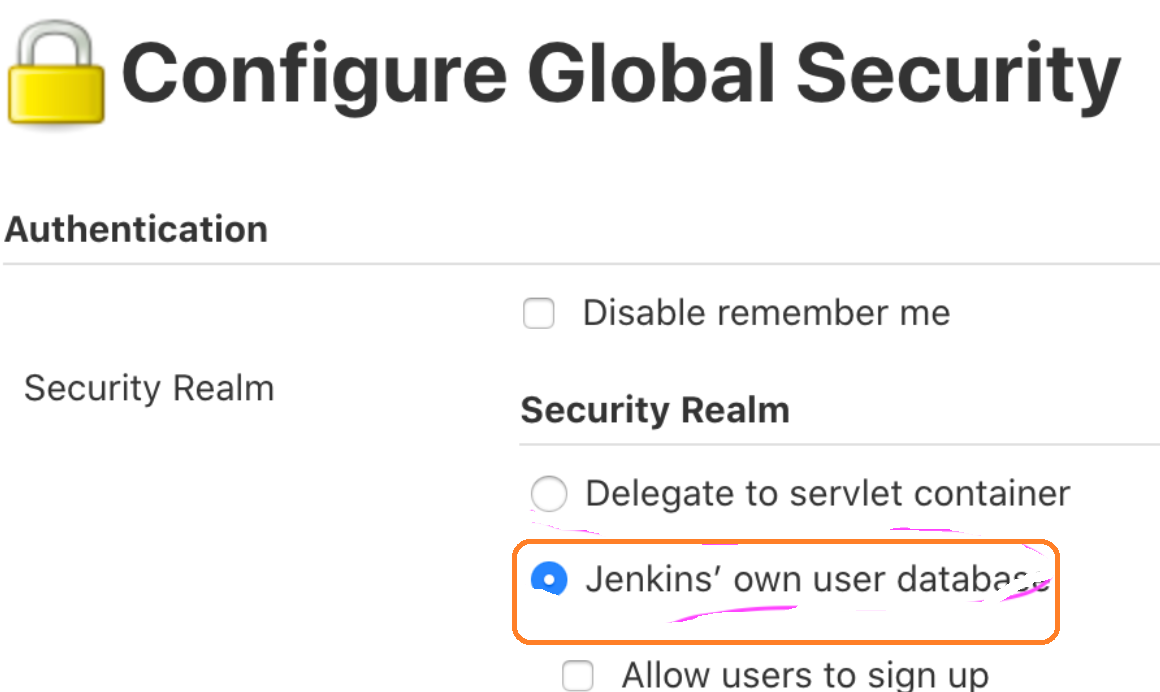
sudo vi /var/lib/jenkins/config.xml

**/root/.jenkins**/**config.xml**

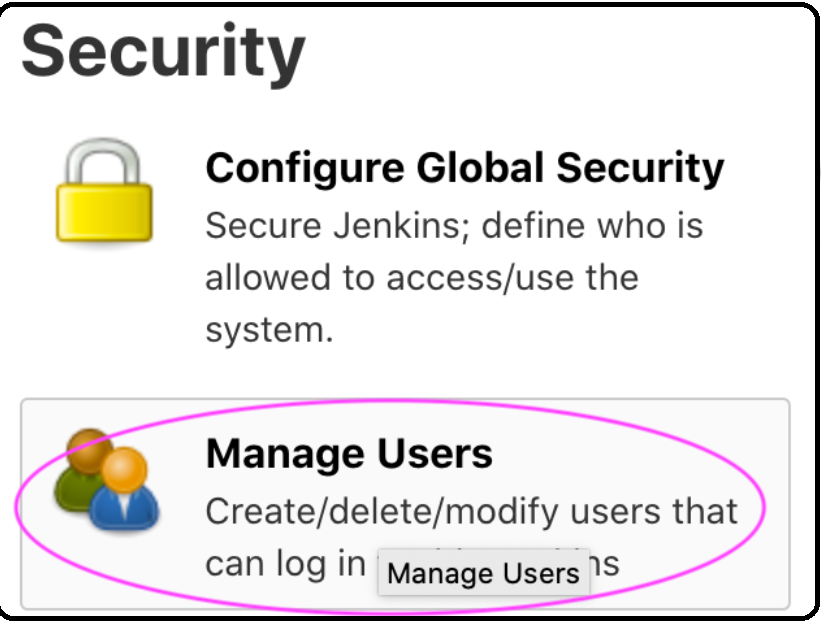


you need to disable security to Jenkins by below change  
2. Find this ***<useSecurity>true</useSecurity>*** and change it ***to <useSecurity>false</useSecurity>***  
3. Restart jenkins by below command:  
***sudo service jenkins restart***4**. Access Jenkins now, click on Manage Jenkins, click on configure Global security**   
5. click on Enable security

6. Click on Jenkins Own user database option.

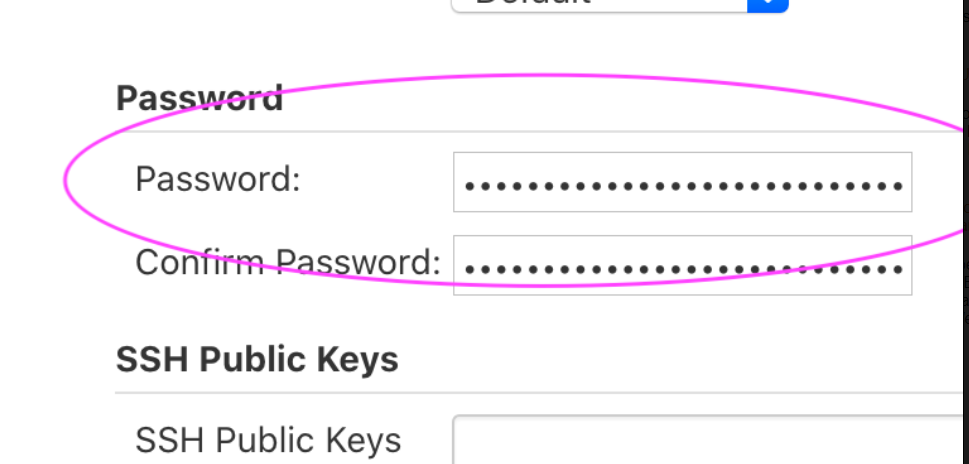


7. click on apply and save  
8. Now click on Manage users, click on admin, click on configure





  9. Enter new password and apply/save



* **\*\*\*\*\*Change Port number for Jenkins (default)**

**Go to below path**

/etc/sysconfig -> RedHat linux

/etc/default -> Ubuntu

# vi Jenkins

**JENKINS\_HOME = “root/.Jenkins/” // Change default home directory**

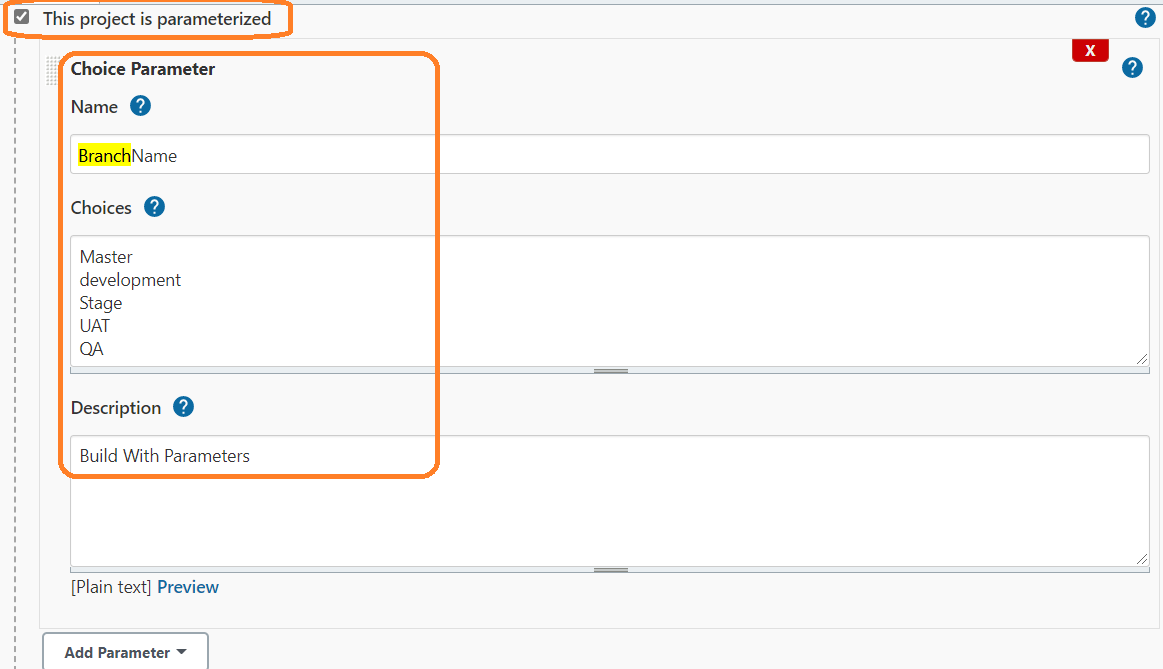
**JENKINS\_USER = “Jenkins” // Change default username**

**JENKINS\_PORT = “8080” //Change default PORT number**

**Once changes done, restart Jenkins**

**10.Build With Parameters**

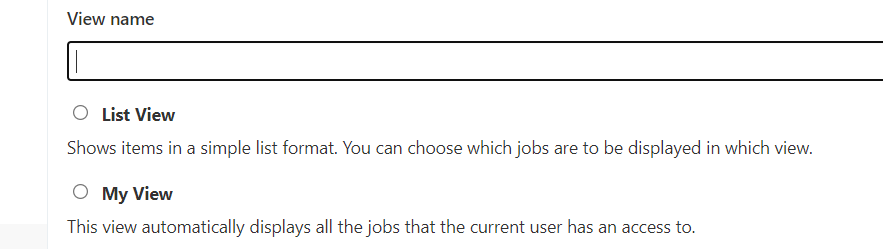
Instead of hard coding values in Jenkins configuration to get values dynamically wee need to enable check box i.e **“Build With Parameter”**



Click on “**Add Parameter”** for more parameters setting.

**Create View**

If we have multiple jobs i.e more than 200+ different purpose, instead of putting all jobs in one place we can create **VIEW** and we can copy all in that view.



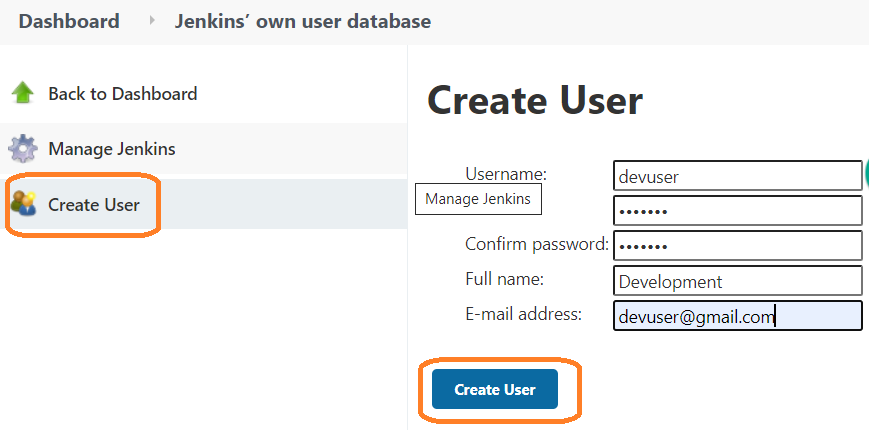
Views is very useful to Navigate the project based on requirement

User full if Jenkins have multiple projects

We can ***Edit View*** and we can ***Delete View*(if we delete view, view only delete)**

**Jenkins Security:** To create users

*Manage Jenkins🡪 Manage Users 🡪 Create User*

**

User name will be store users **user.xml**

**#cd /root/.jenkins/users**

Login with new user and verify.

By default, new user will have all access

We can restrict Admin access navigating below steps

Login with Main Admin user

*Manage Jenkins🡪 Configure Global Security*

*MavenIntegration:*

**Job Configuration history**

It is going to maintain all job configuration information

****

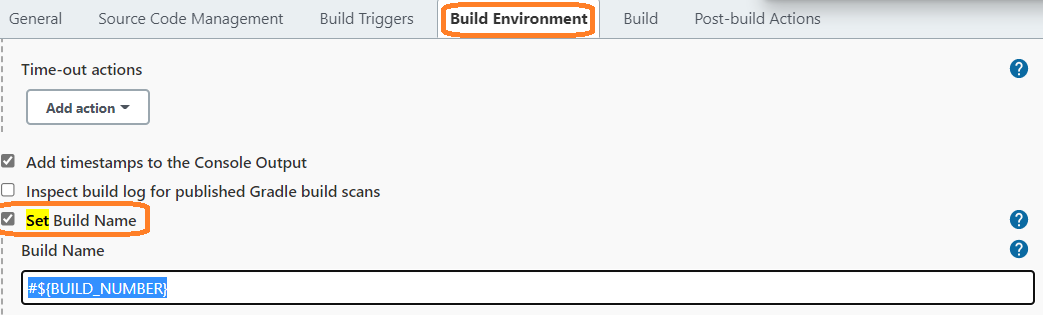
**Build Name an description setter**

This plug-in sets the display name of a build to something other than #1, #2, #3, ... so that you can use an identifier that makes more sense in your context. When you install this plug-in, your job configuration page gets additional setting that lets you specify a build name for each new build.

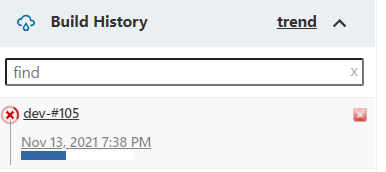
We need to install Build Name an description setter from manage Jenkins🡪 plugin

Each environment we have separate build

Once installation done, we can see set Build name plug-in under Build Environment



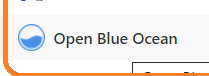
**Click on Build now button.**

****

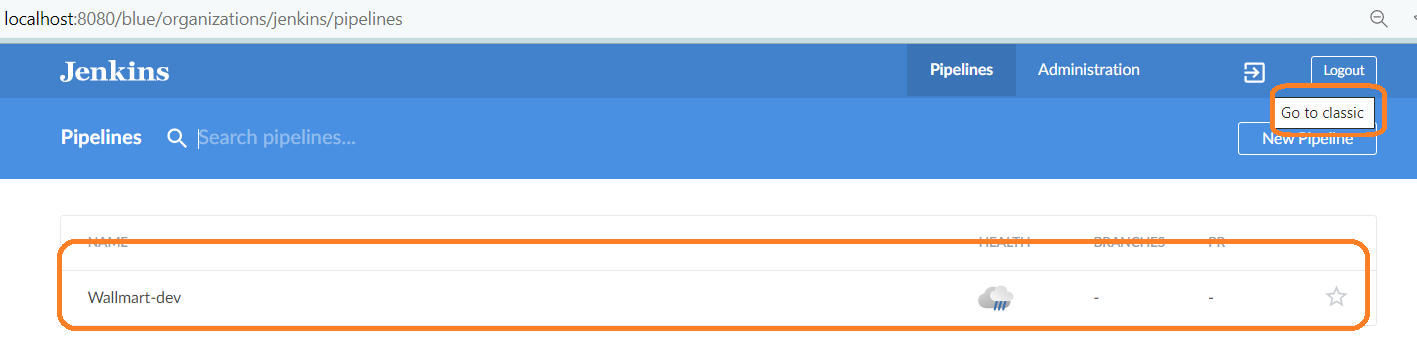
**Blue Ocean**

Installed Blue ocean plug-in from manage Jenkins

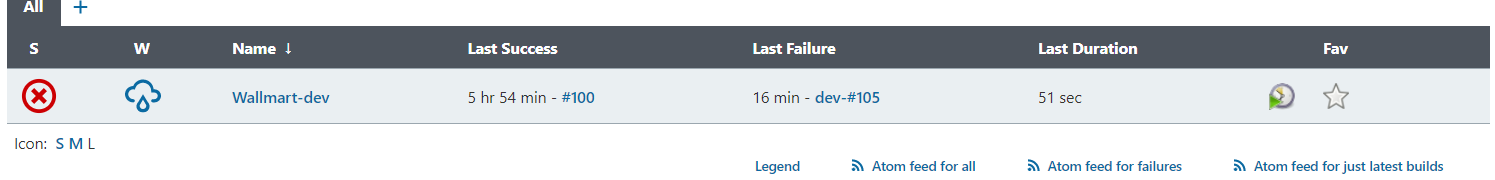
Once installation done, we can see Open Blue Ocean in Jenkins dash board



**Click on Open Blue Ocean**

****

If we click ****



**Backup Jenkins Backup and Restore**

**Backing up data is crucial to any business and Jenkins is not a exception. if you have right backup's it gives you peace of mind in case of any recovery needed to tackle with any unexpected situations like, datacenter outage,Jenkins system failure or accidental deletion of data**

**There are multiple methods of backup the jenkins configuration such as**

**-Using ThinBackup or Backup plugin**

**-Using the git repo**

**-manually backing up the Jenkins configuration or schedule a shell script using the cron**

**The easiest approach which i found is to use the Jenkins plugin for the backup.In this quick demo we are going install ThinkBackup plug-in and configure the backups and then a test restore.**

**Installing the “ThinBackup” plugin**

**Login to Jenkins UI**

**Manage Jenkins 🡪 Manage Plug-in**

**Click on Available > Select 'ThinBackup'**

**Install without restart**

**We will create a directory as storage location for the backups.**

**#mkdir /apps/jenkins\_backup**

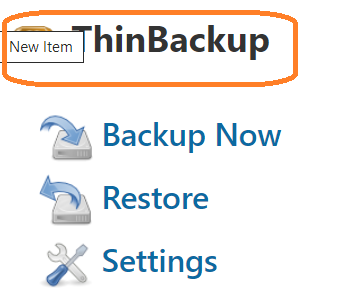
**#chmod jenkins:jenkins /apps/jenkins\_backup**

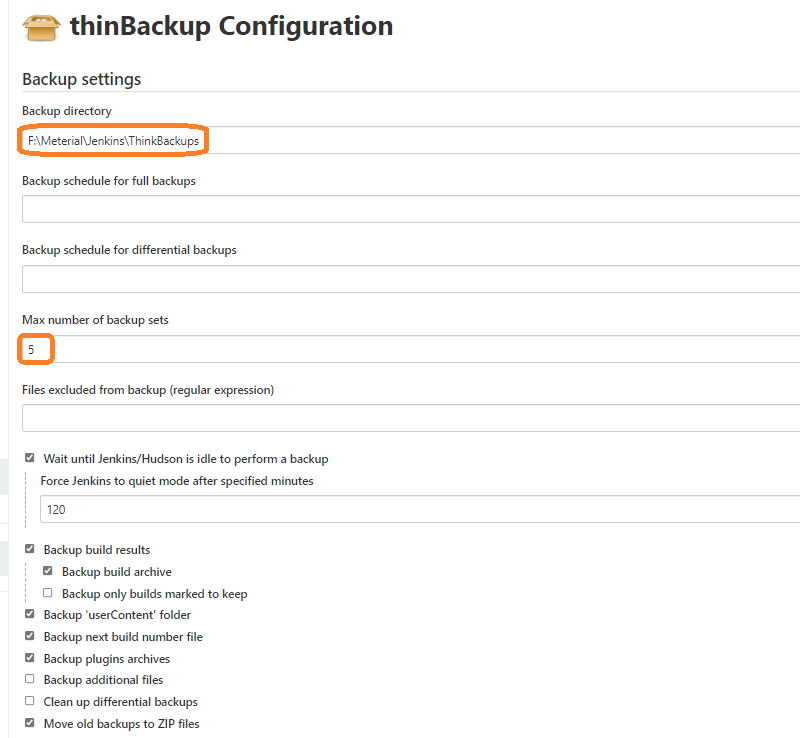
****

**Note = "/apps/jenkins\_backup" need to be under your backup/snapshot policies in case whole system crashes, so it can be restored.**

**After installing plug-in first thing we have to do “ThinBackup” configuration**

**Manage Jenkins 🡪 ThinBackup 🡪Settings**

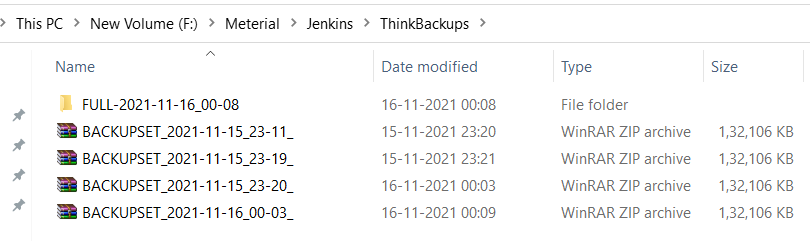
****

****

**Performing Backup:**

Taking Backup – as per the backup schedule in the setting, backup will happening automatically, incase you want to trigger the backup manually can do as following

**ManageJenkins 🡪 ThinBackup 🡪 Backup Now**

****

**Max number of backup sets:**

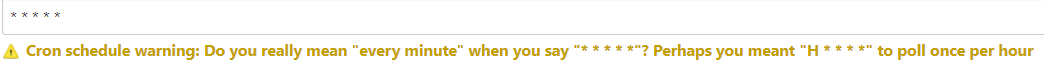
Whatever number we mentioned it will maintain that number of backups in that location

**-1** represent will take ‘n’ number of backup will take

**Backup schedule for full backup:**

Whatever files we selected those files will take full backup

We will take backup for every 1 hour in real time



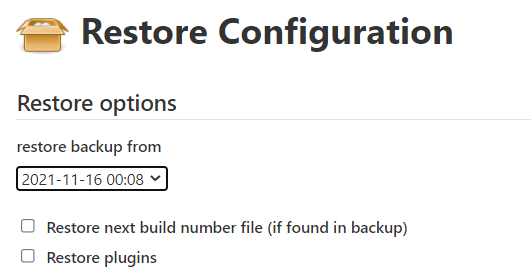
**Backup schedule for differential backups:**

Last backup whatever updates are their those things only will take backups

**Restoring Backup**

**Manage Jenkins 🡪 ThinBackup 🡪 Restore**

**Note:** When restoring the data you can select the backups available

****