**Working with Build Pipelines**

**Fork Repo: https://github.com/chaitanyagaajula/game-of-life**

We are going to create 3 free style projects and connect them with the help of Build Pipeline Plugin and deploy tomcat server.

**Install Plugins** :

Click on Manage Jenkins->Manage Plugins

Install the below plugins without restart:

1.JUnit Plugin to be installed

2.Build Pipeline plugin to be installed

3.Deploy to container

4.Report info

5.Maven Integration Plugin

6.Pipeline Maven Integration Plugin

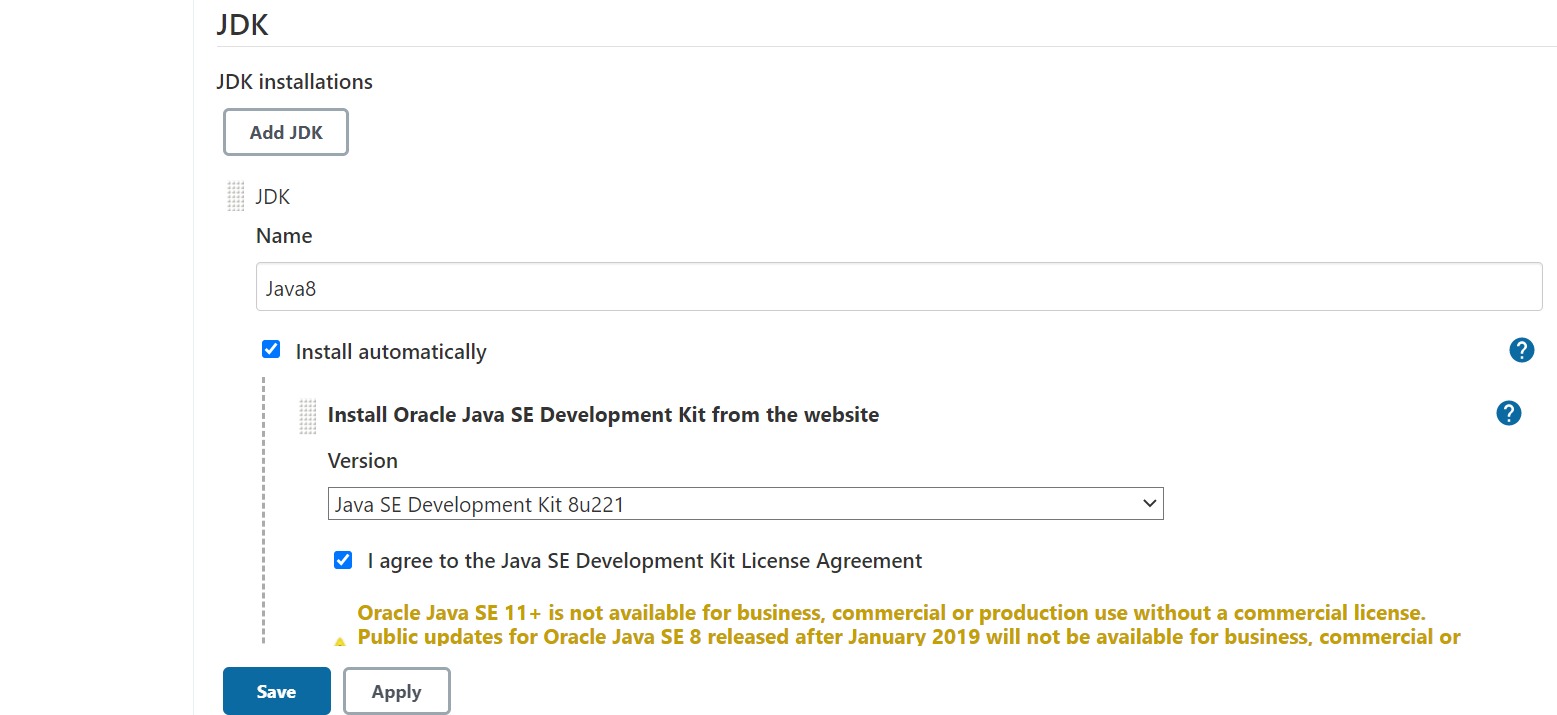
**Modify the below under Global Tool Configuration:**

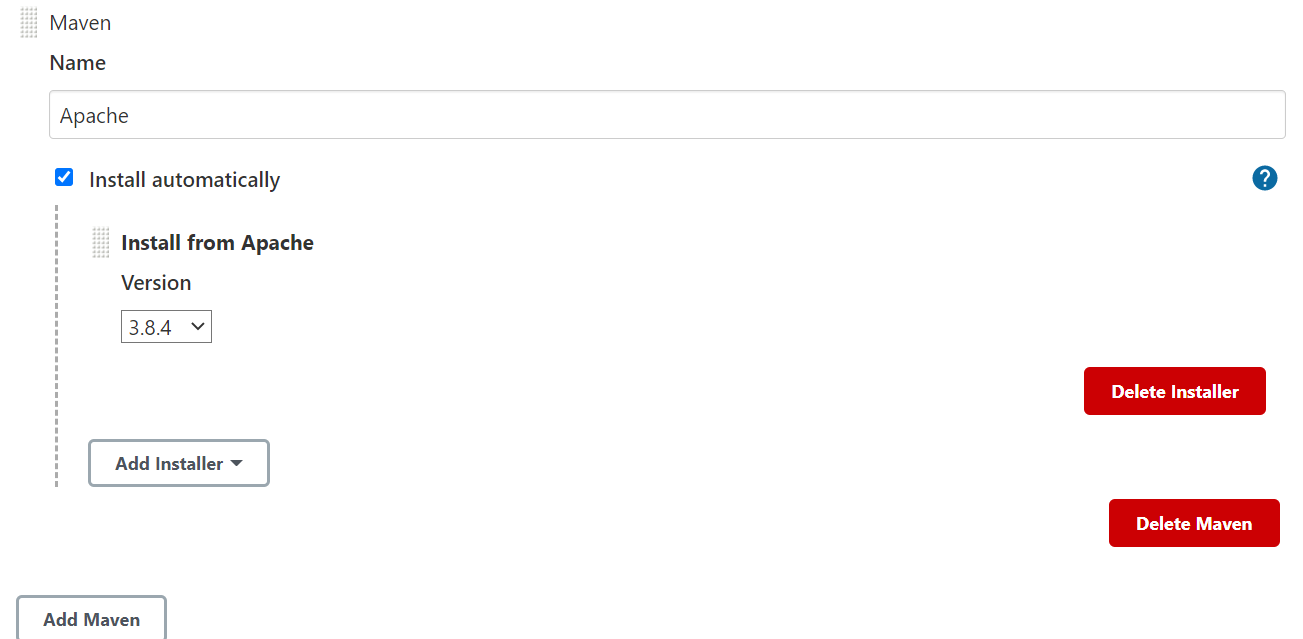
**Under JDK Installations:**

**Name:** Java8

Choose Install Automatically and

And Choose Oracle Credentials





**Install Maven:**

[root@ip-172-31-92-57 ~]# yum install maven

**Tomcat installation & Configuration:**

On the Jenkins Master server console execute the below steps:

wget https://archive.apache.org/dist/tomcat/tomcat-8/v8.5.35/bin/apache-tomcat-8.5.35.tar.gz

tar -xvzf apache-tomcat-8.5.35.tar.gz

**Rename to tomcat:**

**mv apache-tomcat-8.5.35 tomcat**

cd tomcat

yum install nano

nano /root/tomcat/conf/server.xml

Goto

<Connector port="8080" protocol="HTTP/1.1"

Modify 8080 to 9080

Save and exit:

**Modify tomcat users file:**

cd /root/tomcat/conf

nano tomcat-users.xml

In the end of the file go and add the below lines

<role rolename="manager-gui"/>

<role rolename="admin"/>

<user username="deployer" password="deployer" roles="manager-gui,manager-script,admin"/>

</tomcat-users>

**Start the tomcat server:**

Access the below location:

[root@ip-172-31-23-23 ~] # cd /root/tomcat/bin

Give execute permissions for both startup and catalina

[root@ip-172-31-23-23 bin]# chmod +x startup.sh

[root@ip-172-31-23-23 bin]# chmod +x catalina.sh

**Start tomcat :**

[root@ip-172-31-23-23 bin] /root/tomcat/bin/startup.sh

**Also have a copy of the repo contents on Jenkins Server:**

[root@ip-172-31-92-57 ~]# git clone https://github.com/chaitanyagaajula/game-of-life.git

Cloning into 'game-of-life'...

remote: Enumerating objects: 1780, done.

remote: Counting objects: 100% (1780/1780), done.

remote: Compressing objects: 100% (587/587), done.

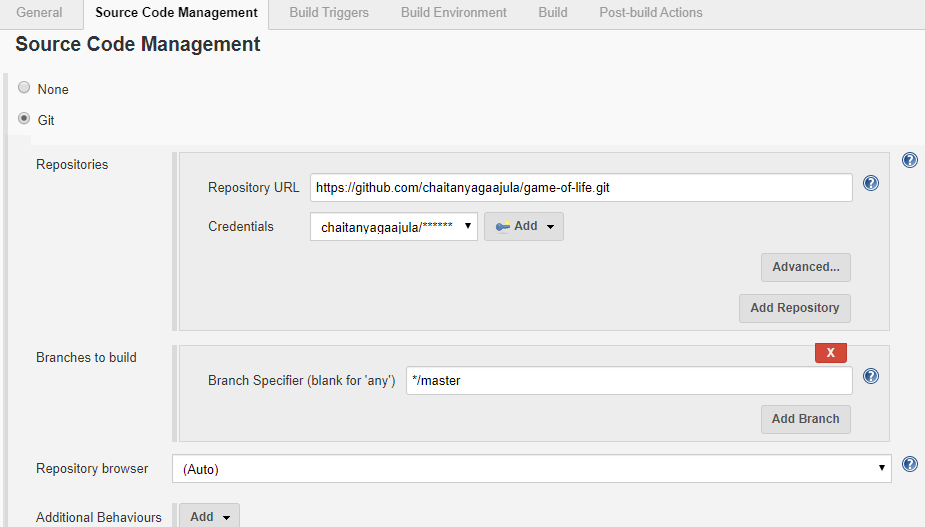
remote: Total 1780 (delta 1127), reused 1766 (delta 1120), pack-reused 0

Receiving objects: 100% (1780/1780), 14.99 MiB | 0 bytes/s, done.

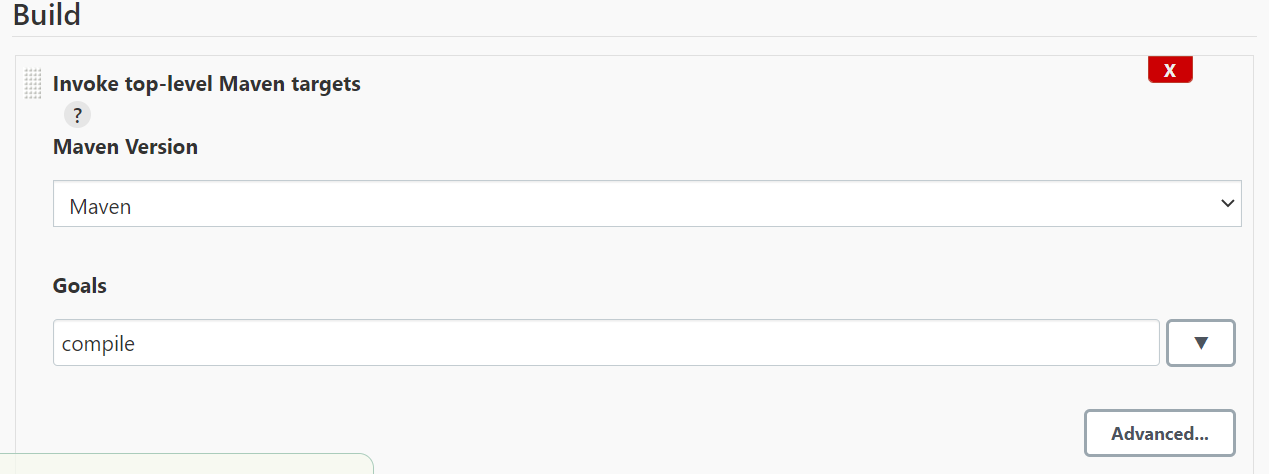
Resolving deltas: 100% (1127/1127), done.

**Creating Projects / Jobs**

1.Create a free style project named Gof\_Build .In SCM define below



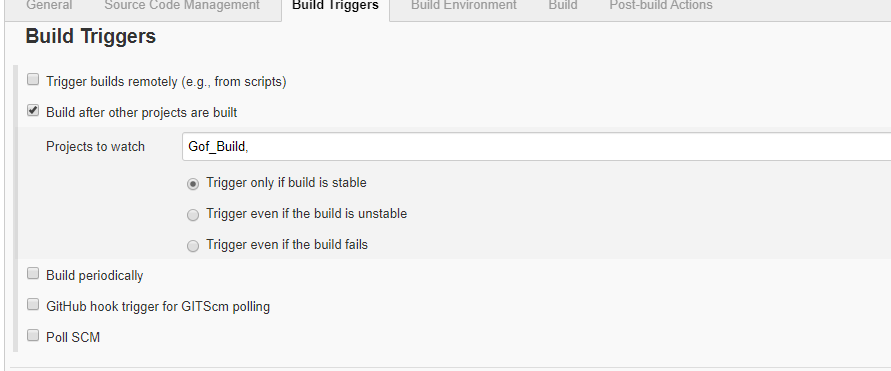
2 .In the build step define goals as compile as below



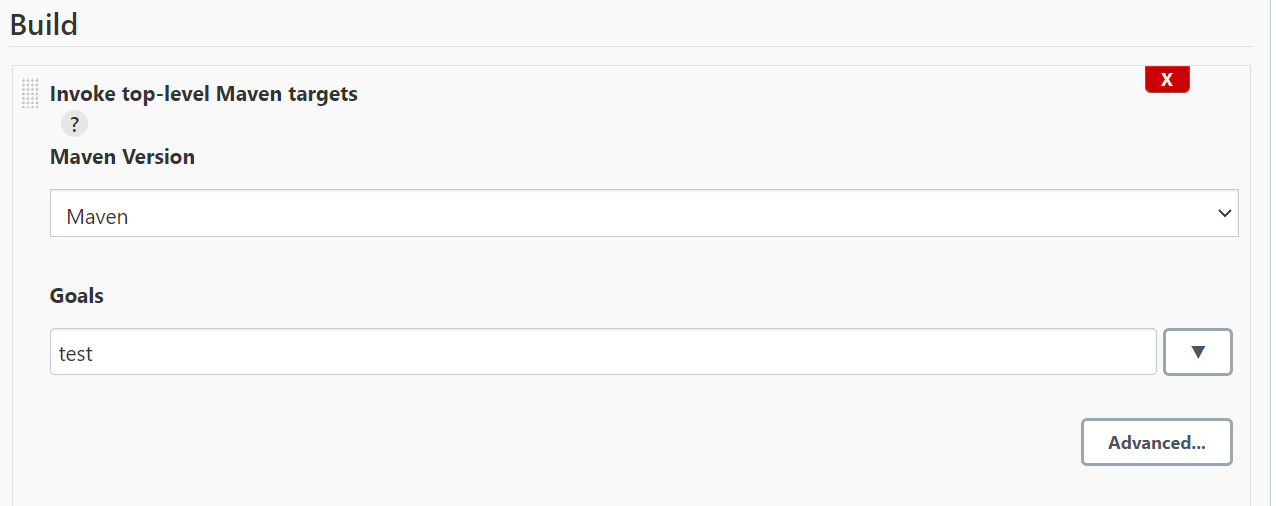
Save it.

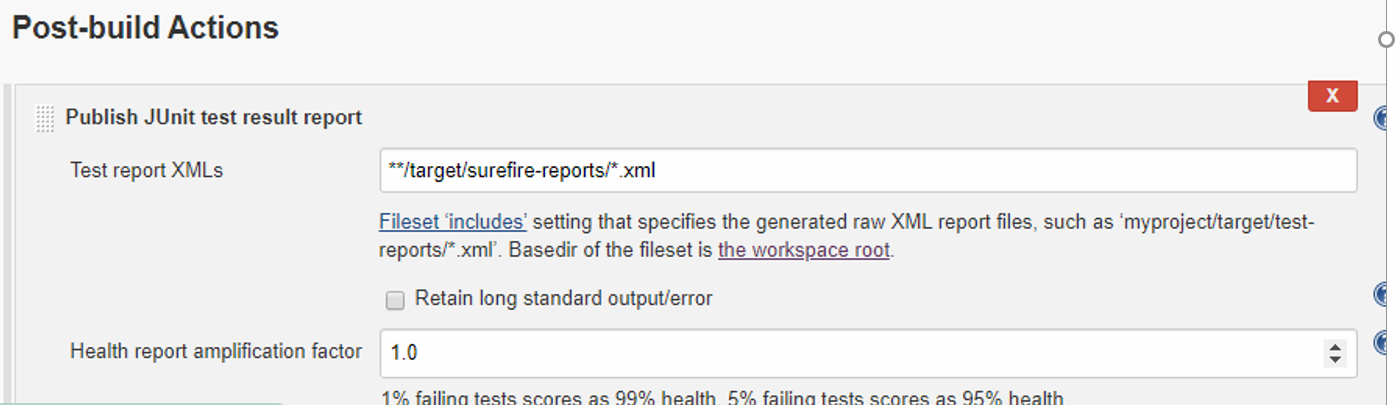
1. .Define a second free style project Gof\_test and include the same SCM github url.

In the Build Triggers tab , include Gof\_Build to be executed before the Gof\_test as shown below



4. In the Build step include the Goals as test.And in the Post Build actions include the path for the test reports.





5. Save the Job and Execute first job (Gof\_build)

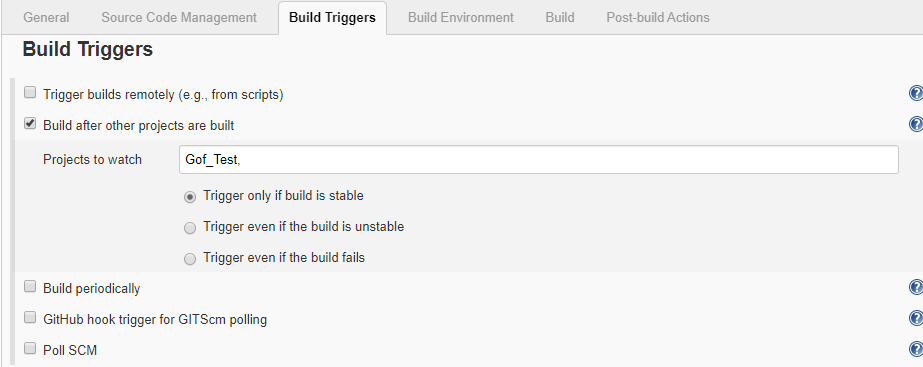
We can see the test results as below after the Job is executed.



1. Define the third job Gof\_Deploy

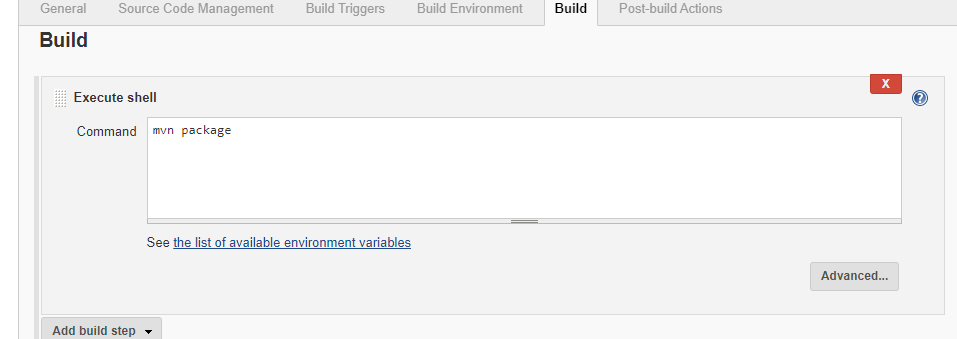
Provide the SCM Git Hub URL same as of the others.

In the Build triggers tab, define to execute Gof\_test to be executed before the Gof\_Deploy as below:

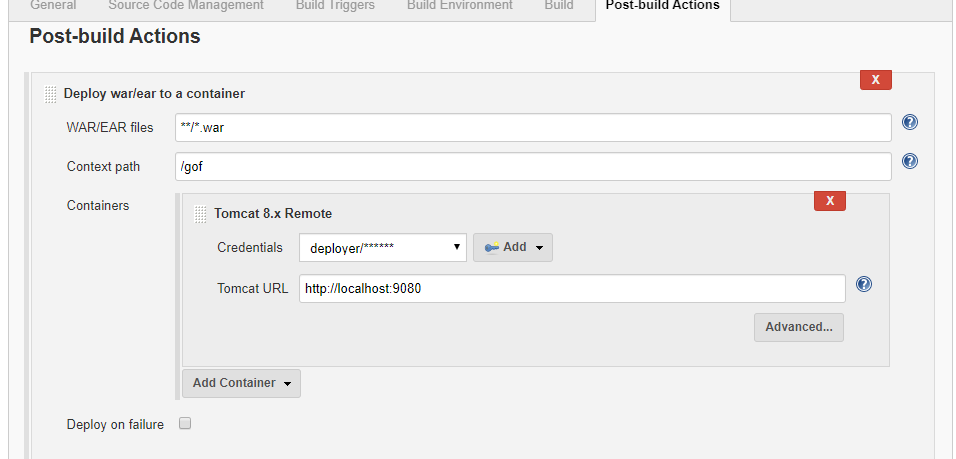


1. In the Build tab select Execute Shell and give the below command to package the code as a war file as below:

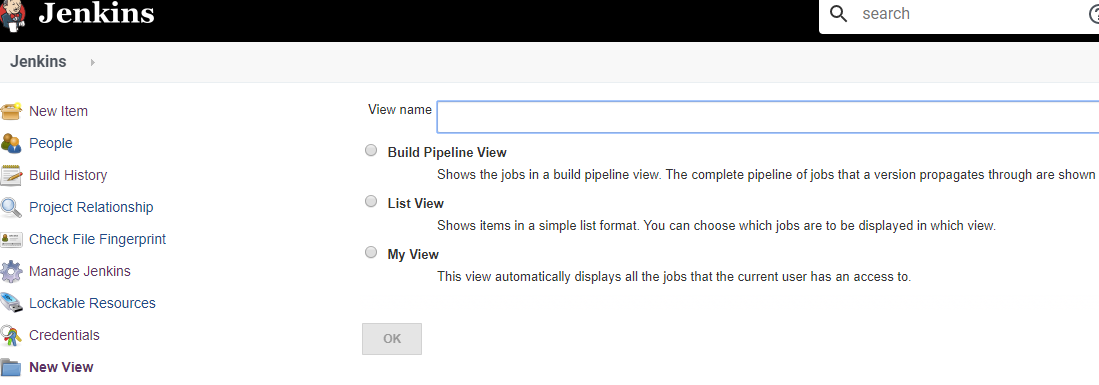
**Note:** You can also choose top level maven targets and choose goal as package

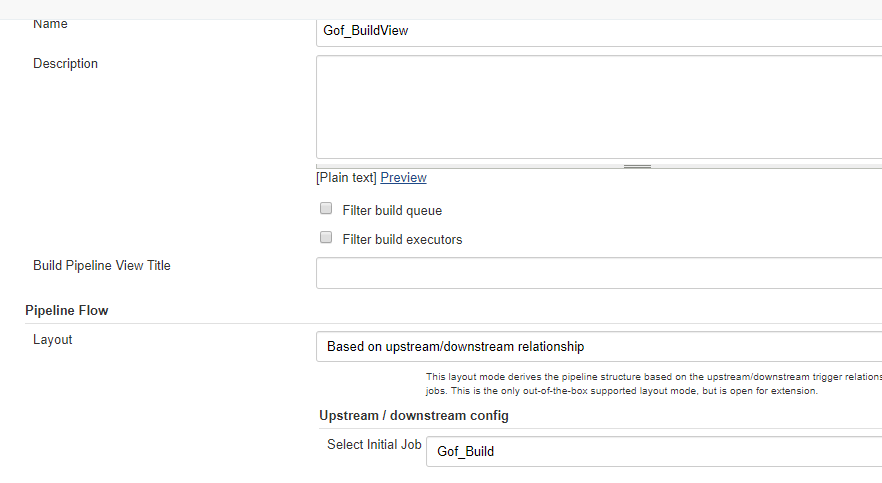


8.In the Post-build Actions, deploy the packaged war file to Tomcat container as below -



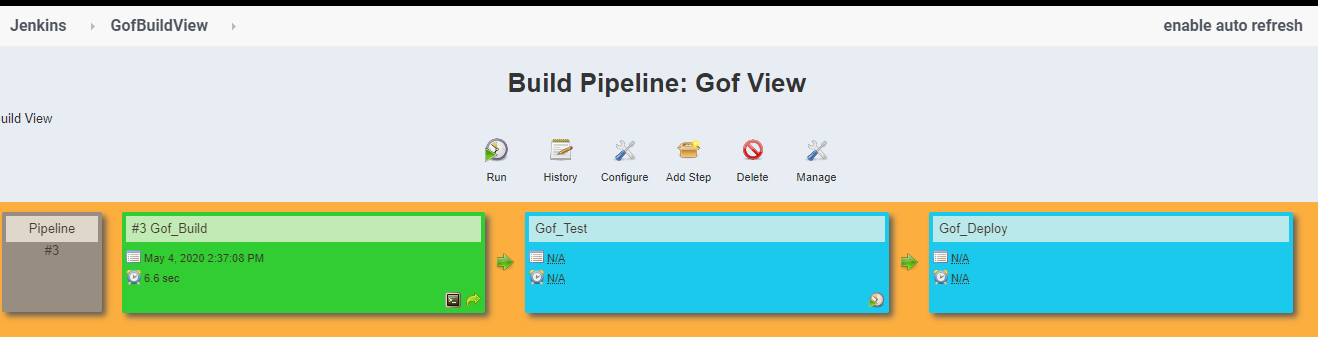
9. Select the Build Pipeline View and give the details as below.





Click Apply , ok

Click on GofBuildView, you will see the output as below



Now, Click on Run, the jobs gets executed, one after the another.

Once the war file is deployed successfully to tomcat.

You should be accessing the below URL - Publicip:9080/gof/

**Note:**

**If Deploy fails, restart tomcat script**

