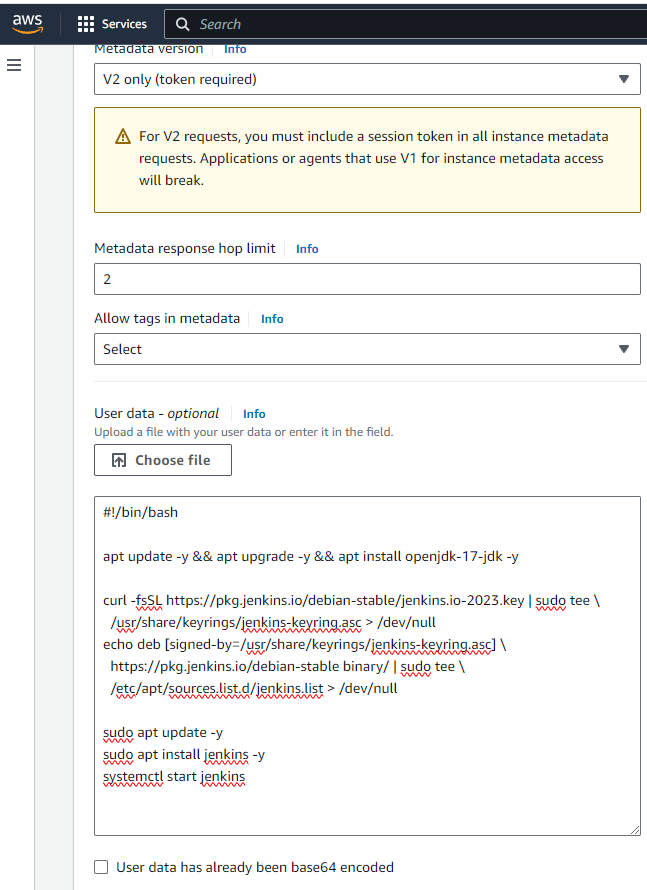
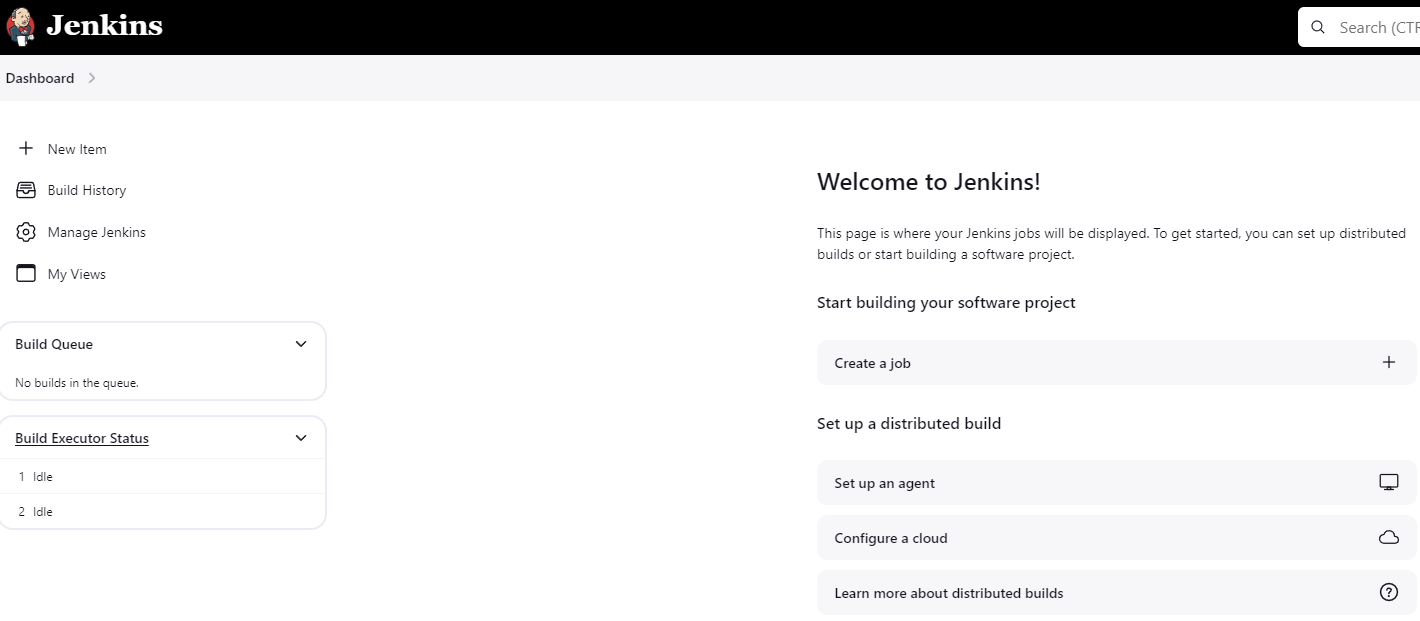
**Automate Project Build with Tomcat 10 using Jenkins**

1. To automate project build, we need to have a Jenkins Server EC2 instance.

Add the below commands in user data section of EC2 instance before launching it to install Jenkins on it



1. Once the EC2 instance is up, Post deployment steps are performed for configuring Jenkins to setup the Jenkins server. Once Jenkins is configured, we see the below dashboard screen.

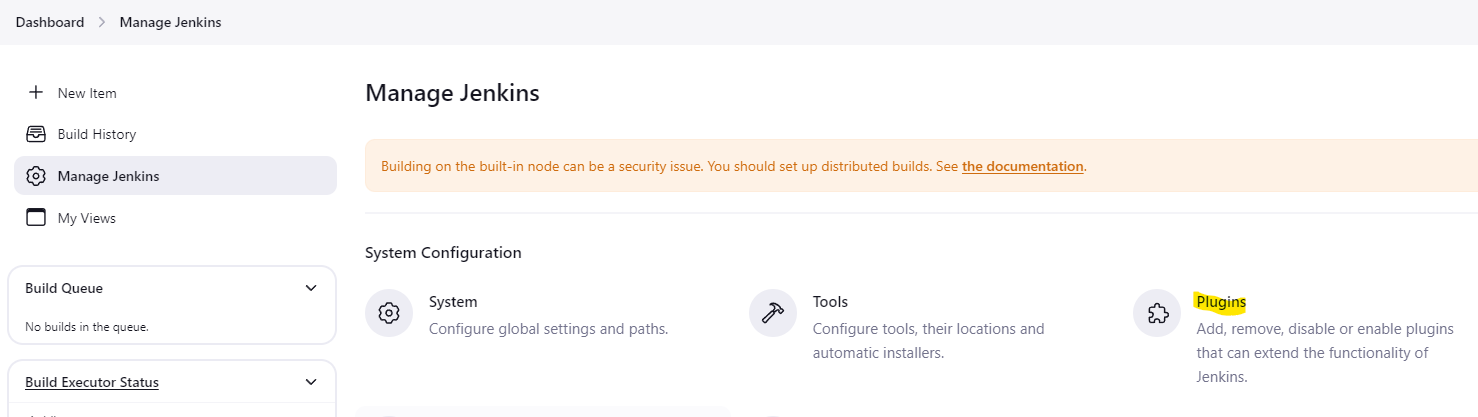


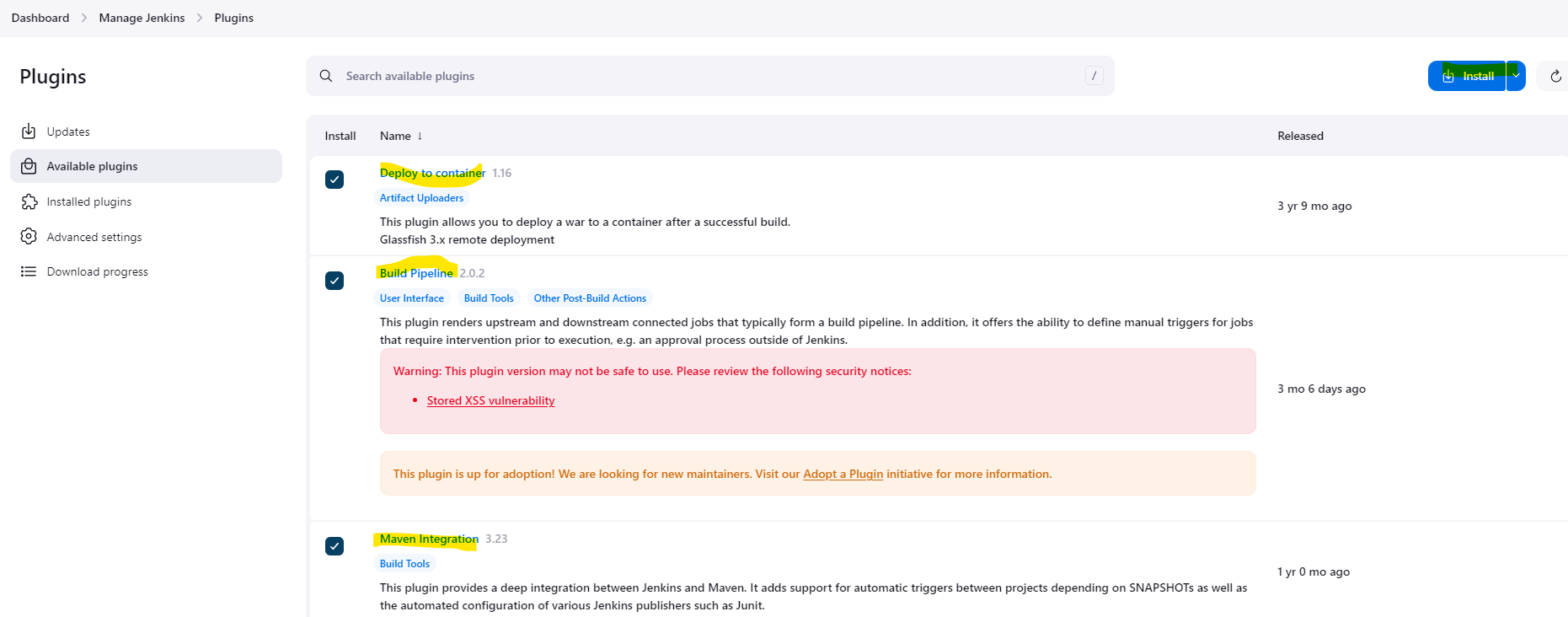
1. Once Jenkins dashboard is visible, we need to install the below mentioned 3 plugins for automating our project build by going to Manage Jenkins > Plugins > Available Plugins

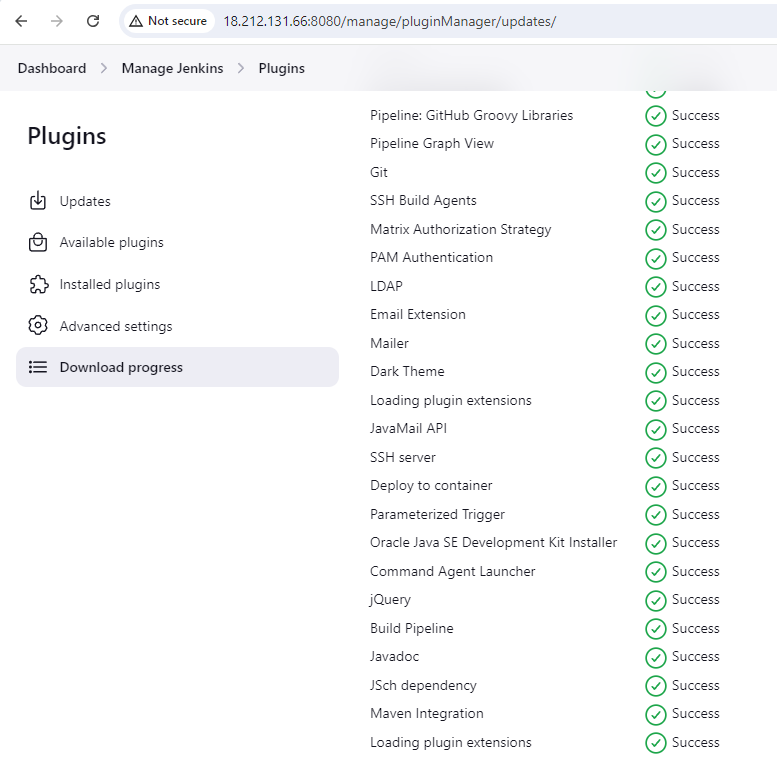
build pipeline

maven Integration

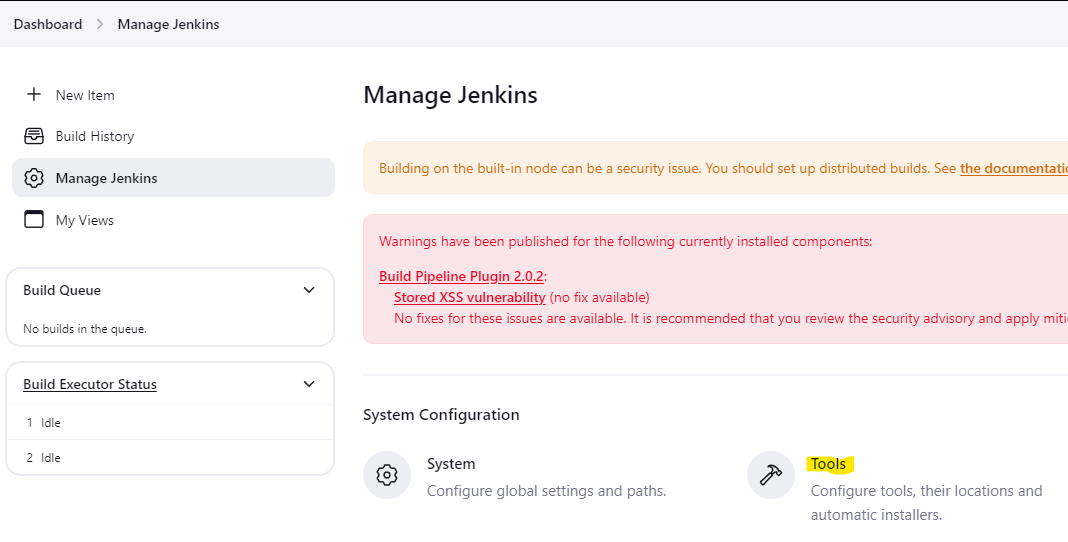
deploy to container



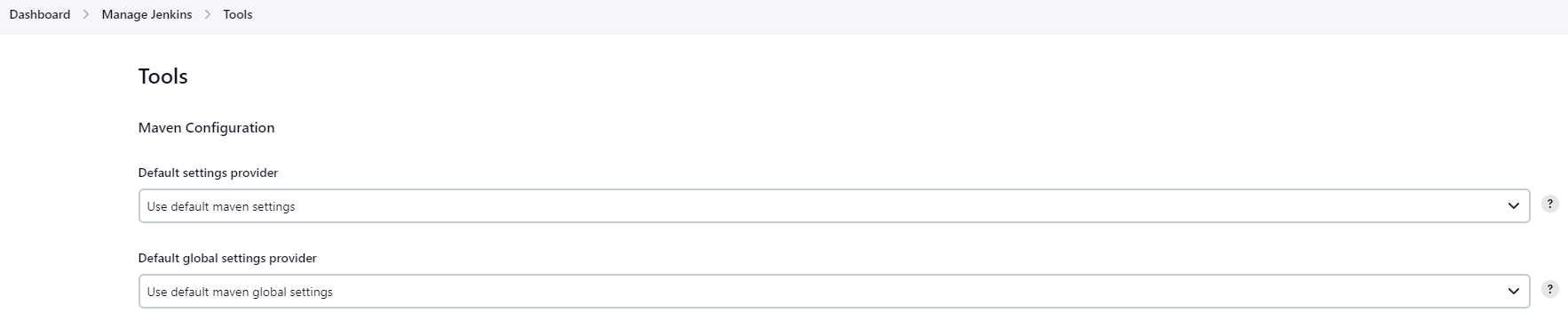




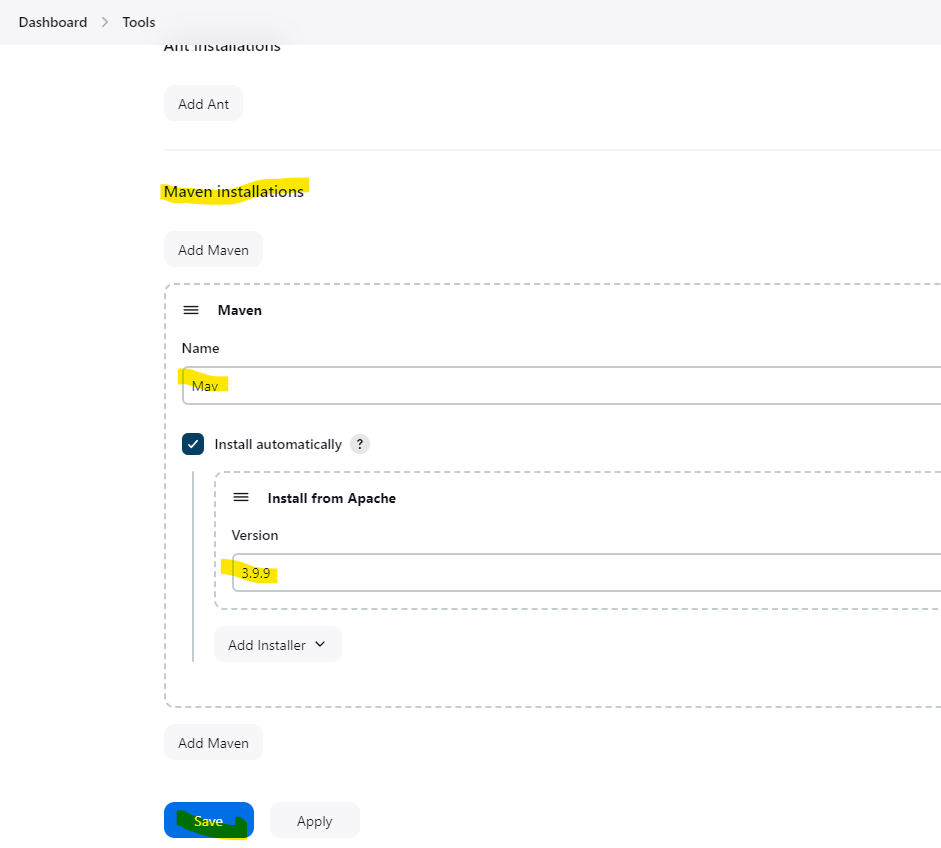
1. Once the Plugins are installed, we will go to Manage Jenkins > Tools



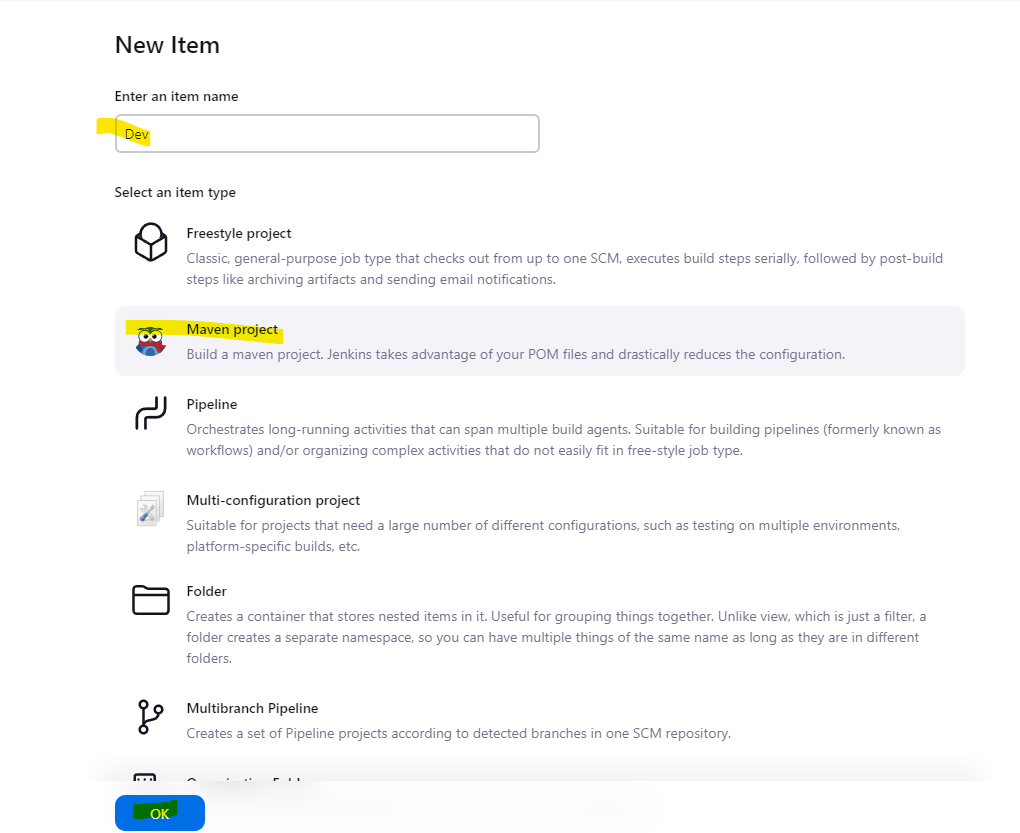
1. Under Maven Configuration, Below Settings should be there.



and also, we need to define Maven Installations, we can give it a name for e.g. MAV and select install automatically and under Install from Apache > Version, keep the default visible version. In our case 3.9.9 and click Save.



1. Now we need to create the first job, which we will give the name as “DEV”, it will be a maven project since we need to run the maven test command to check the source code and pom.xml



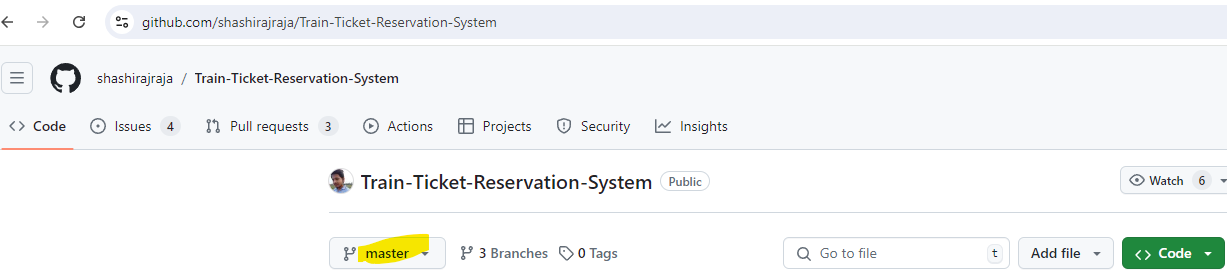
1. Now we will configure the “Dev” job as below.

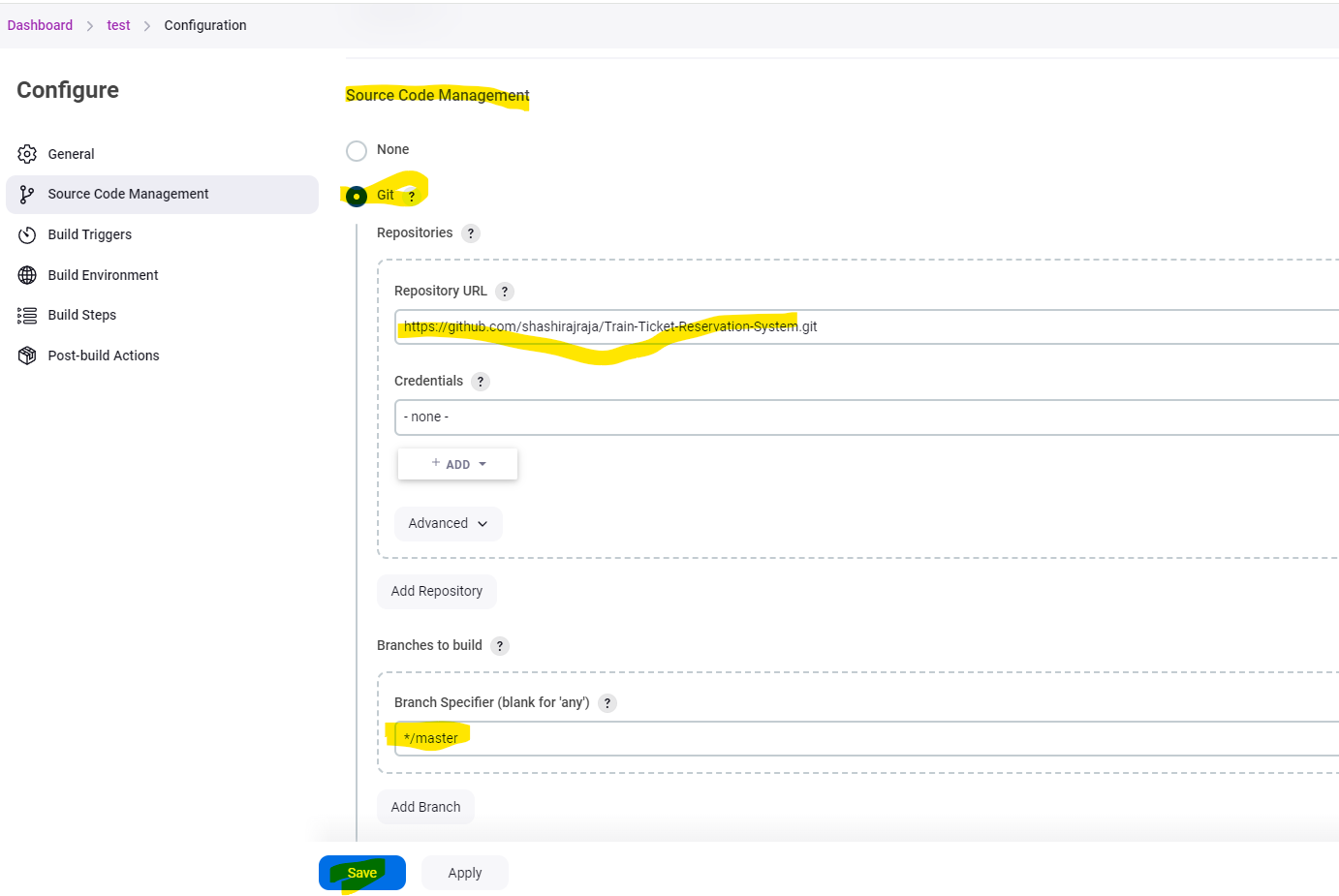
We need to select git under Source Code Management

We need to give the Repository URL under repositories

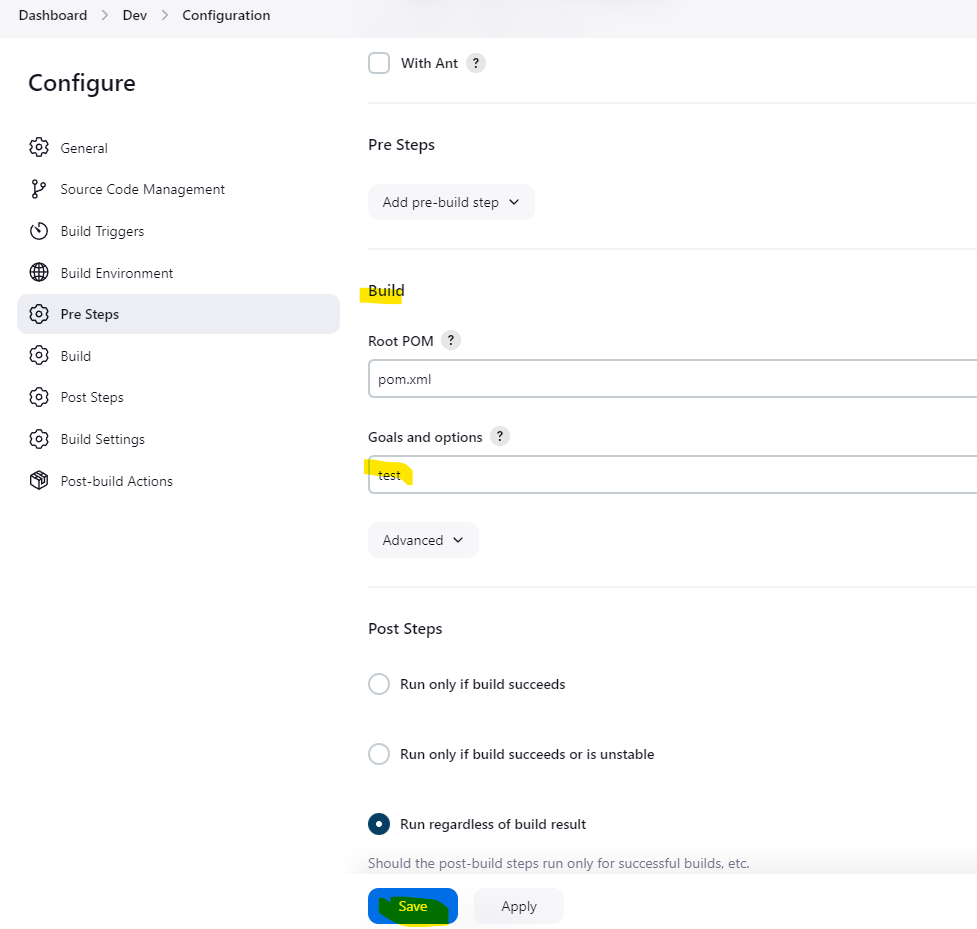
<https://github.com/shashirajraja/Train-Ticket-Reservation-System.git>

Lastly, we need to specify the branch name same as it is on the git hub link, in our case its \*/master in the git link also so we need to specify the same in the job also. Then Click Save

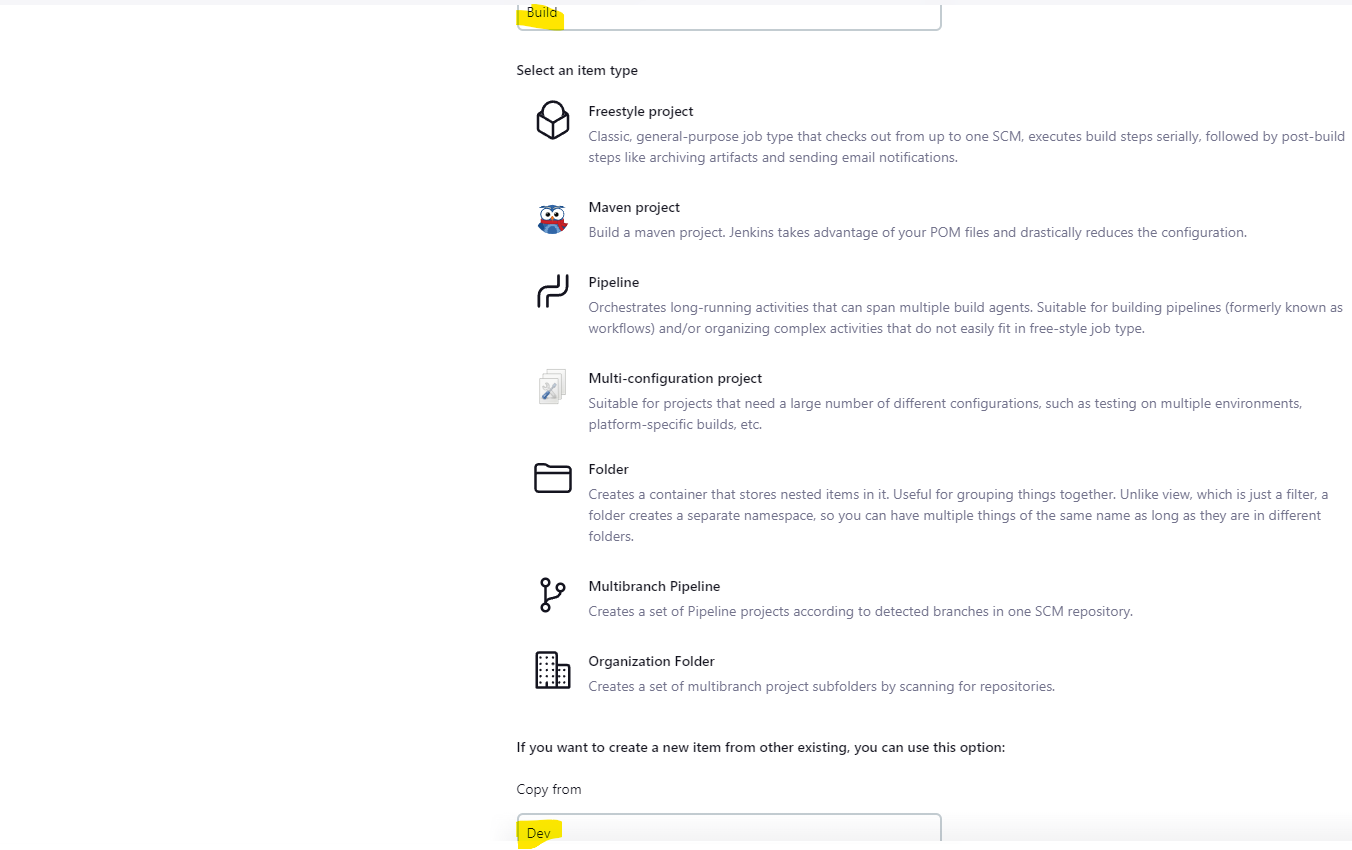


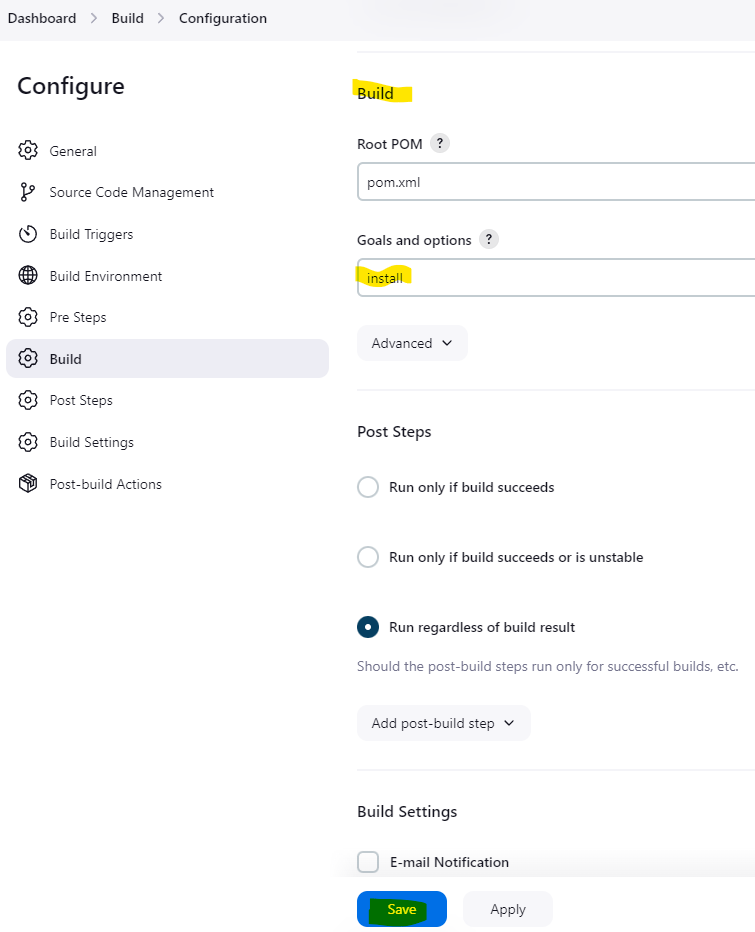


1. Under Build, we need to give Goals and options as test since we are testing the build and then click Save.

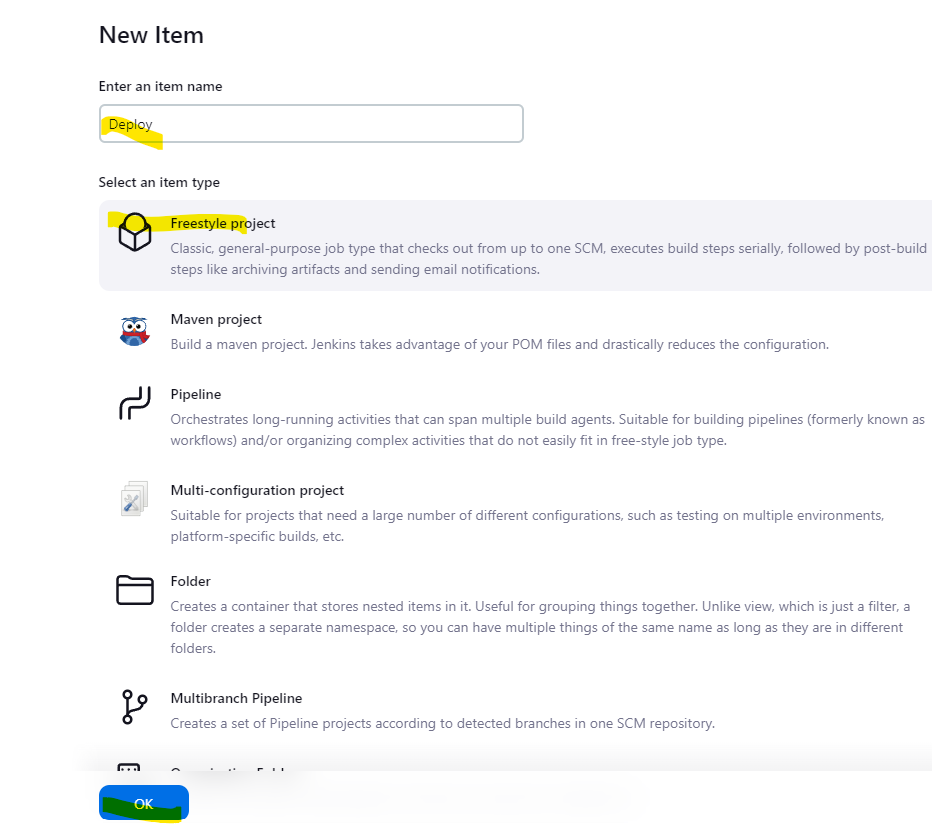


1. Now we create the second job as “Build” which we will copy the project from “Dev”. Just we need to change the Goal and Options under Build to install and click save, since we are building the code.

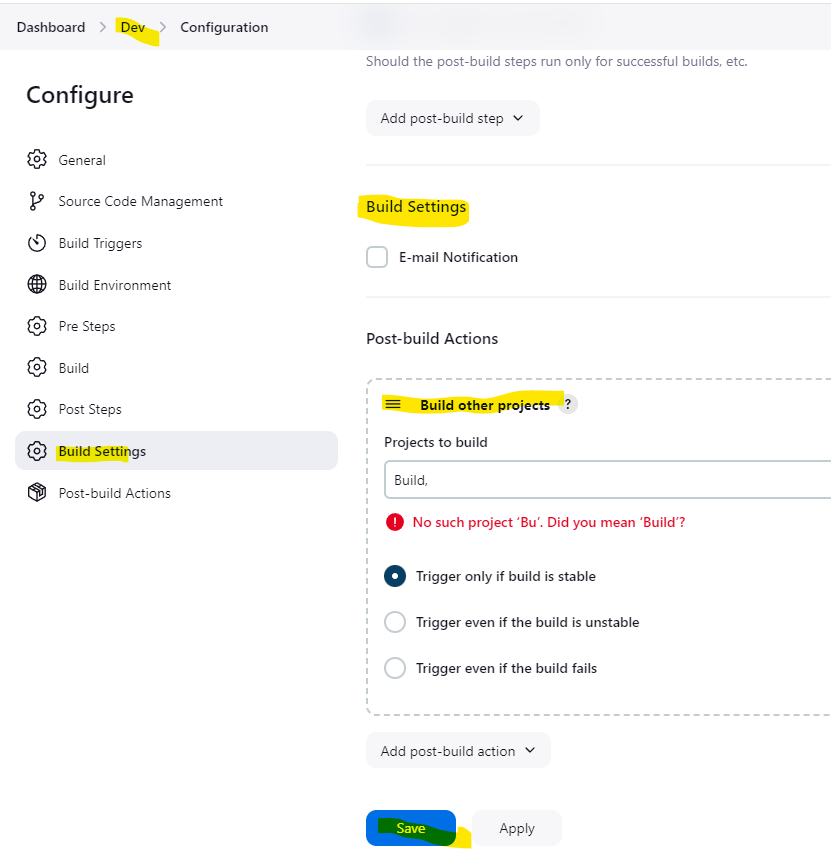


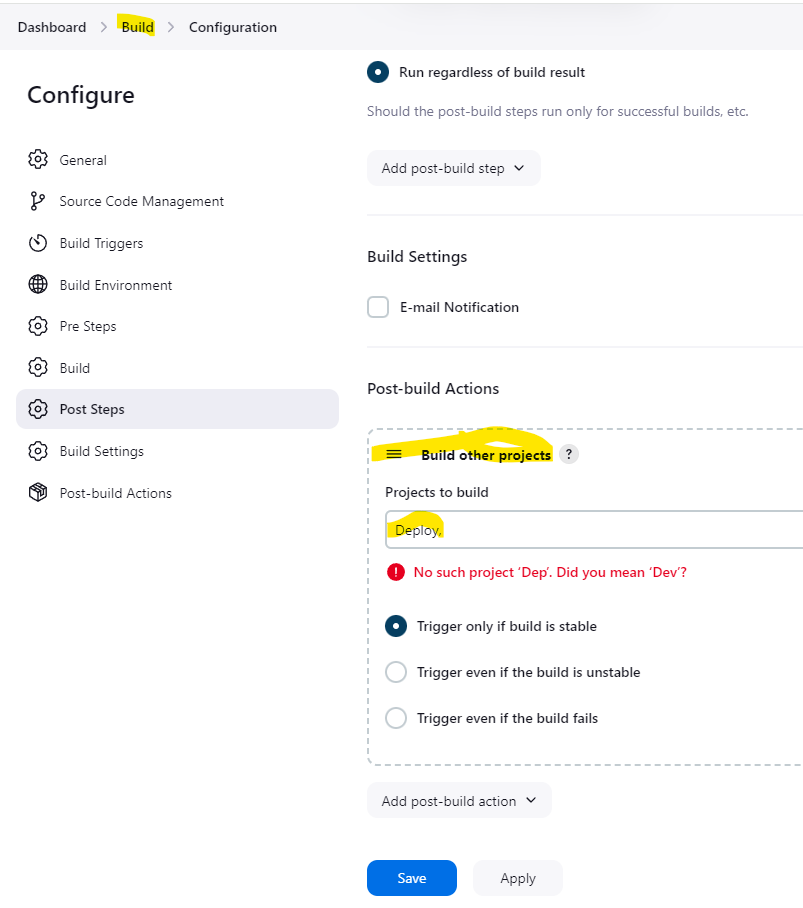


1. Now we will create the third job as “Deploy” and select it freestyle project and for now, we do not need to configure anything in that job.



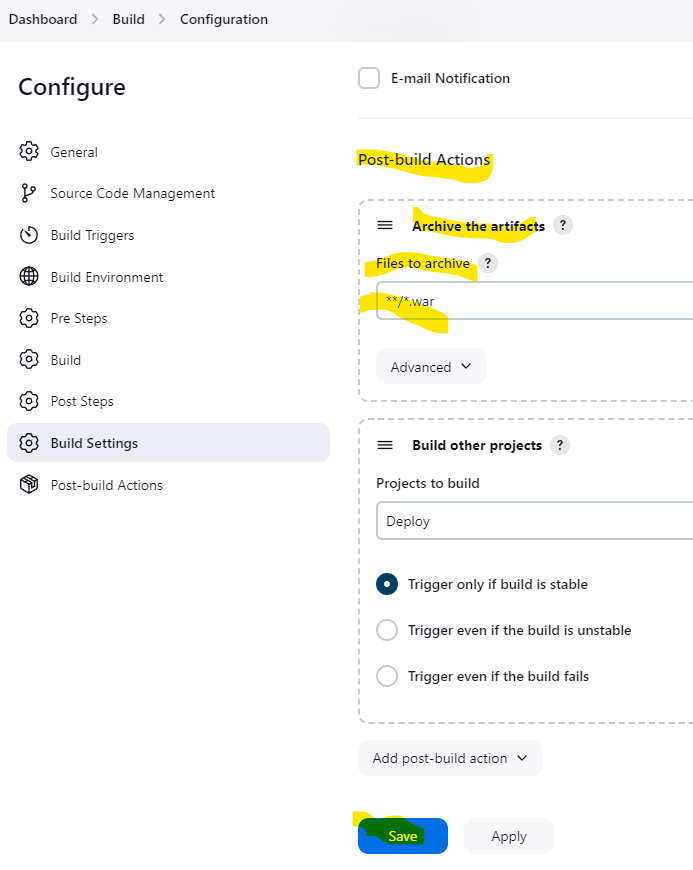
1. Now we define the dependency on each job i.e. Dev to trigger Build Project and Build to Trigger Deploy Project.



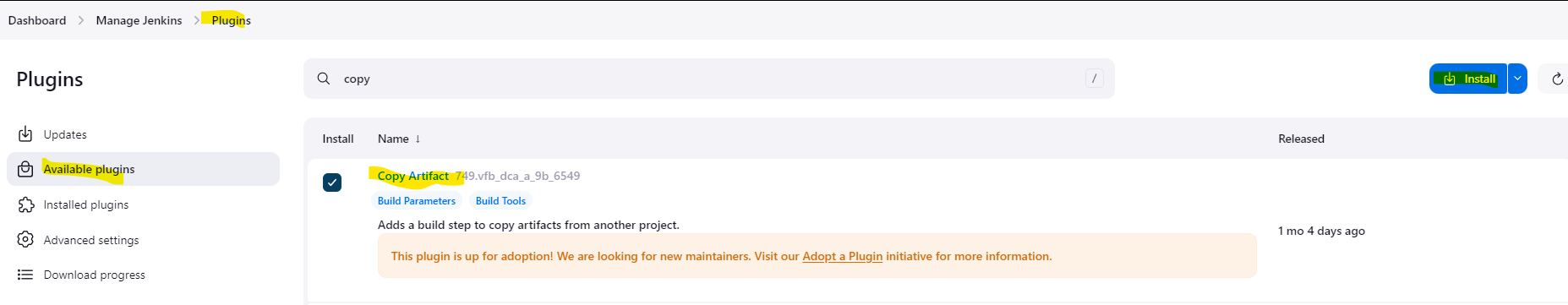


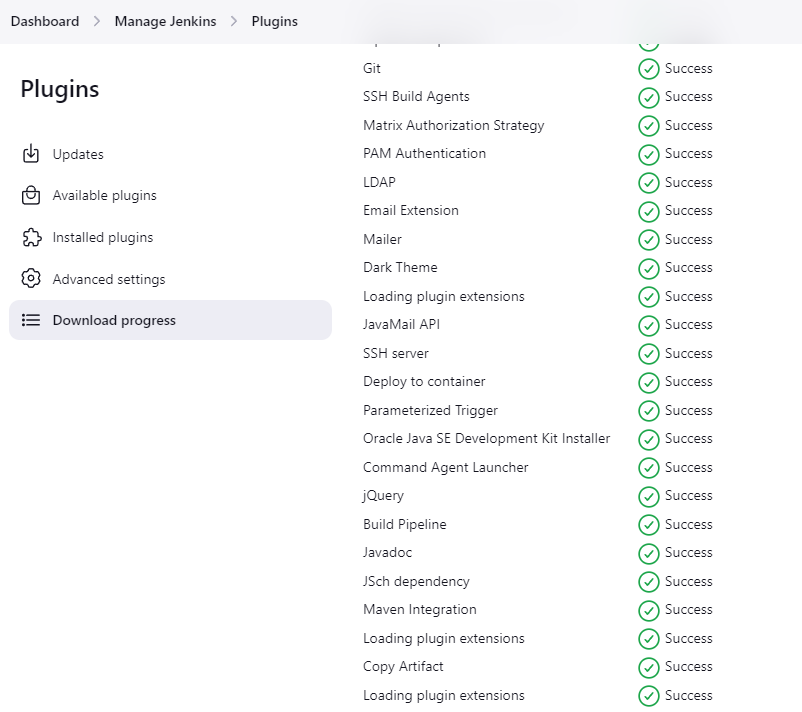
1. Now we configure the “Build” Project to archive the artifact by following the below steps.

Configure the Build Job and go to Post Build Actions and add post-build action and select Archive the artifacts and in that under Files to archive, Type \*\*/\*.war (this defines any type of war file) and click save.

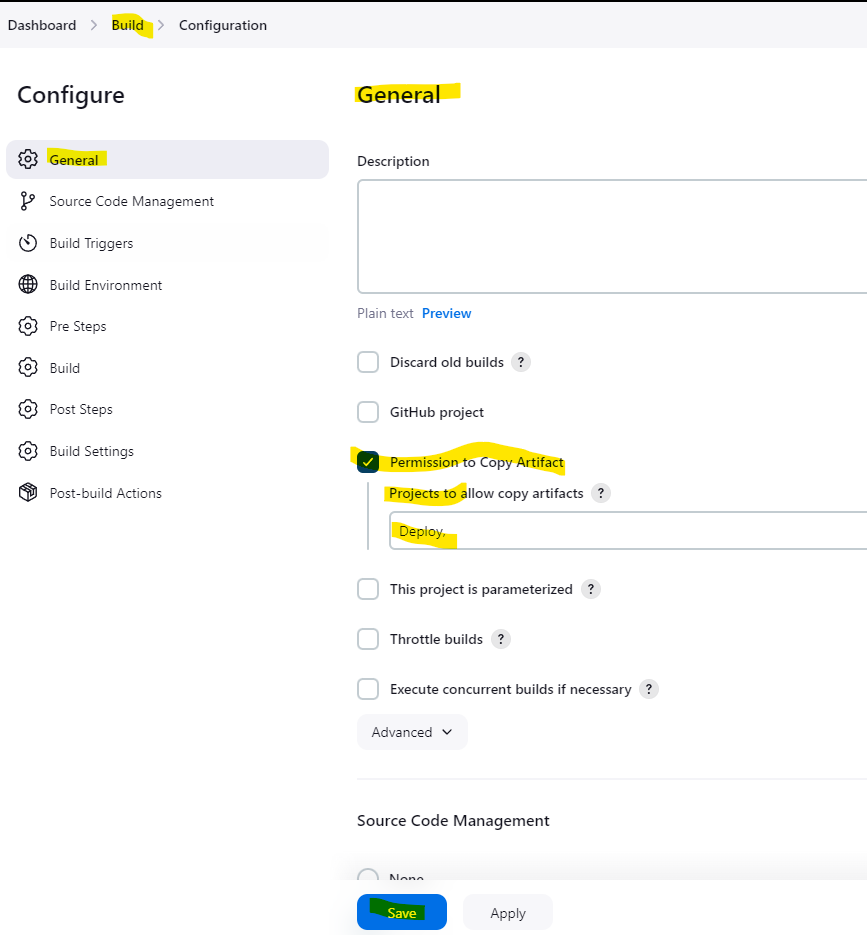


1. Now we need to install a plugin named copy artifact (So that archives files like war can be copied to artifacts)

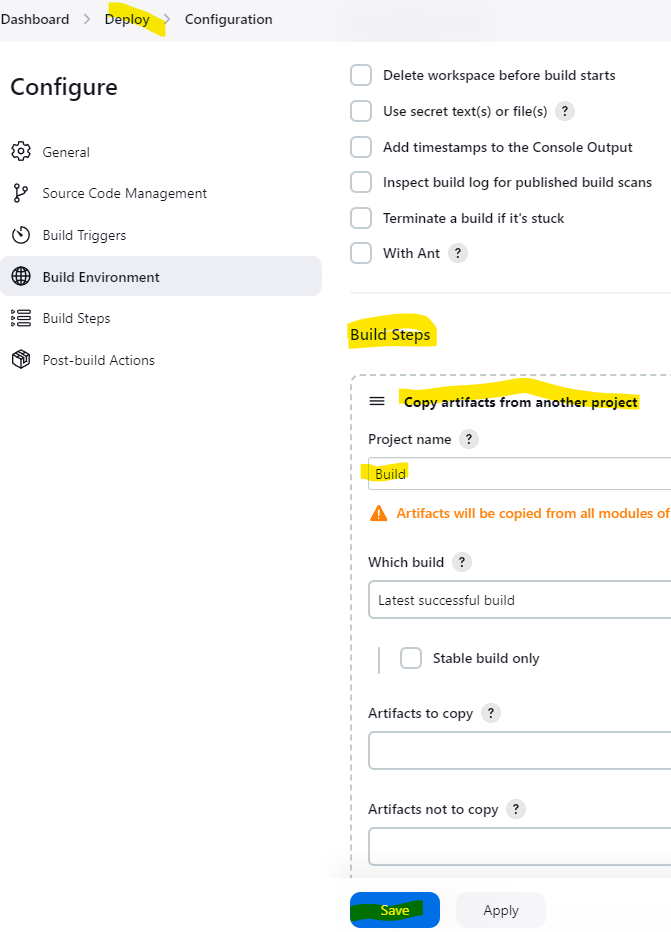




1. Now we will configure the “Build” project again and under General select the option of Permission to Copy Artifact and under Projects to allow copy artifacts we need to give “Deploy” project and then click Save.

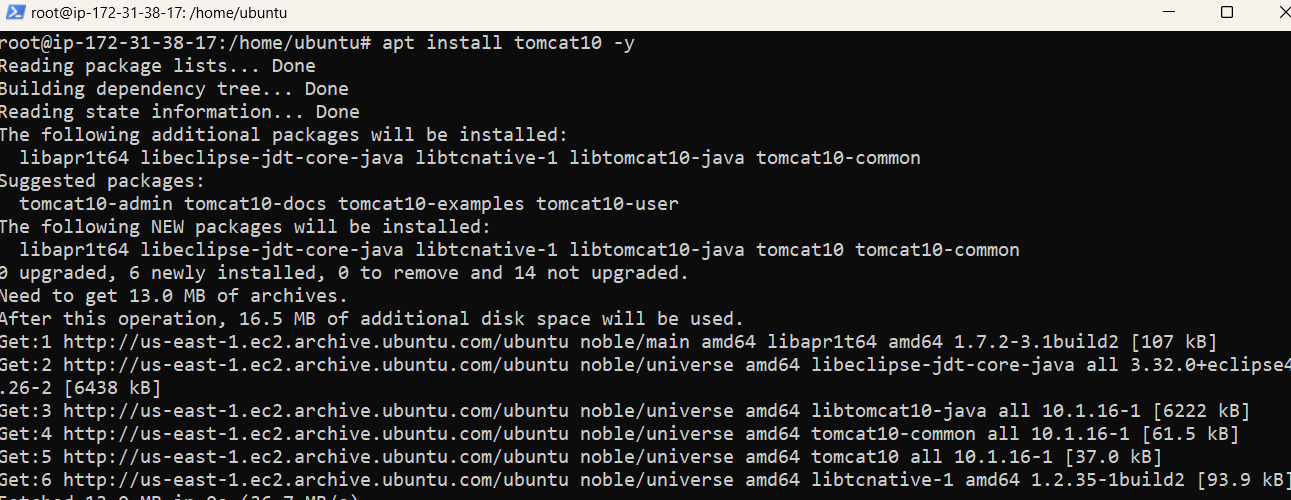


1. Now we will configure “Deploy” project and go to option Build steps > Add Build step and select Copy artifact from another project and under Project Name, we will give “Build” since we are copying artifacts from build project and click Save.



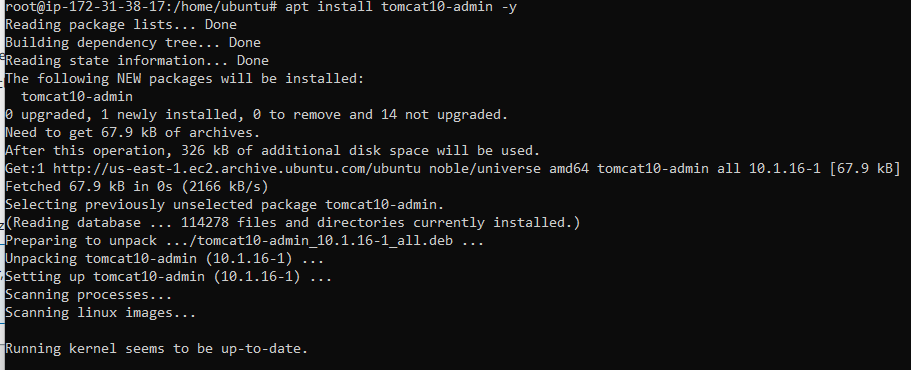
1. Now we will install Tomcat10 on Jenkins server through terminal.

apt install tomcat10 -y



Next, we will install tomcat10 admin to configure roles and users.

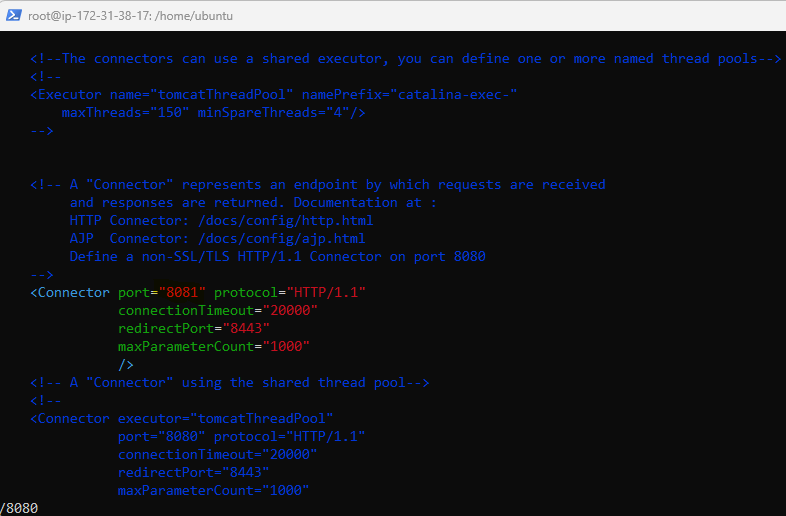
apt install tomcat10-admin -y



Since tomcat and Jenkins run on same port 8080 by default, so we need to change the port for tomcat10 from 8080 to 8081.

vi /etc/tomcat10/server.xml

Search for the line <Connector port=”8080” and change it to 8081

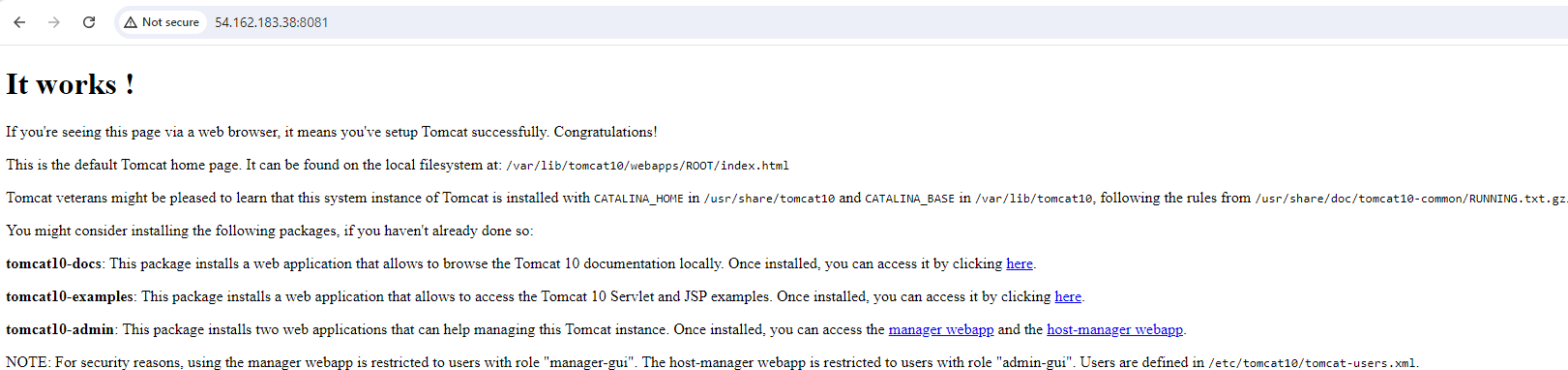


Now we will restart tomcat10 to make the changes

systemctl restart tomcat10



Verify tomcat10 is running on the new port 8081. So, it works



Now we will provide the ownership to tomcat user from root.

Before ownership change



We will run the ownership change command

chown -R tomcat:tomcat /var/lib/tomcat10

After Ownership change



Now we will assign the tomcat10 roles by going to the location /etc/tomcat10

vi tomcat-users.xml and append the below lines in the file.



<role rolename="manager-gui"/>

<role rolename="manager-script"/>

<role rolename="manager-jmx"/>

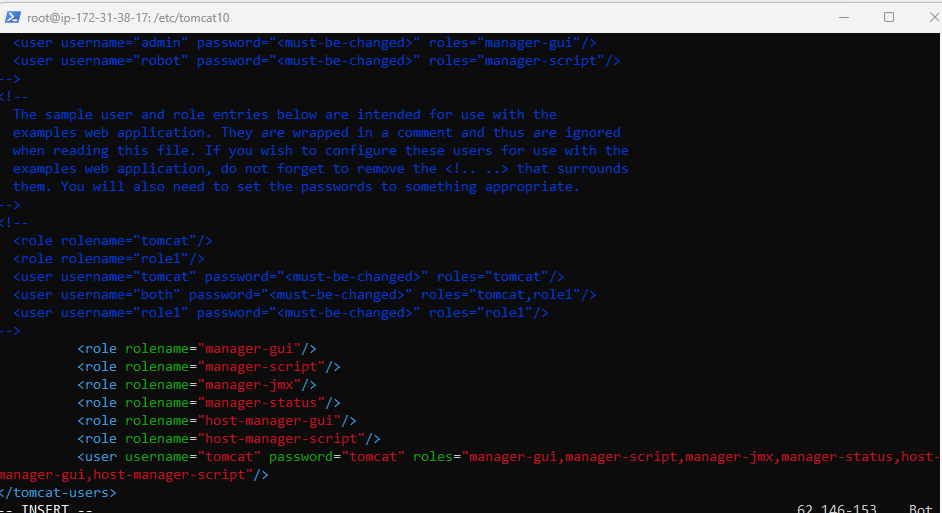
<role rolename="manager-status"/>

<role rolename="host-manager-gui"/>

<role rolename="host-manager-script"/>

<user username="tomcat" password="tomcat" roles="manager-gui, manager-script, manager-jmx, manager-status, host-manager-gui, host-manager-script"/>

(In the above lines, we also make sure that username should be tomcat only since tomcat user has been given ownership of tomcat10 folder)

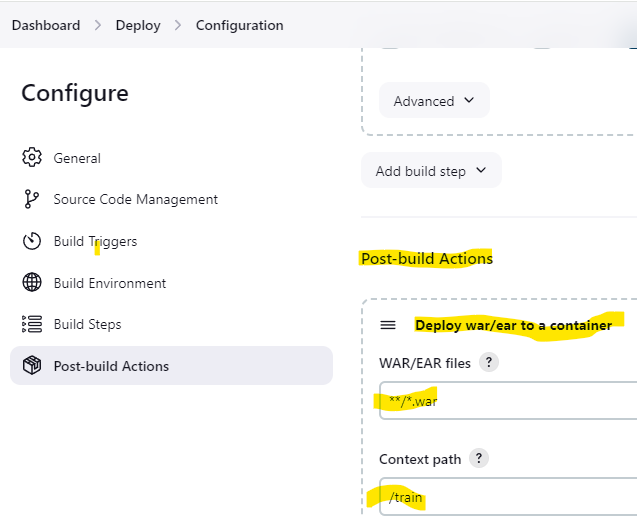


Again, we will restart tomcat10 service.

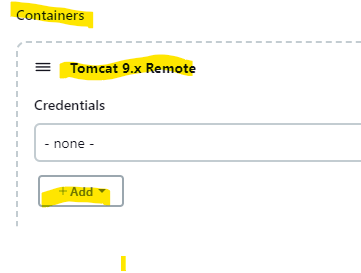
systemctl restart tomcat10



1. Next, we will configure again “Deploy” Project then go to Post Build Actions> Add post-build actions then select Deploy war/ear to a container and in WAR/EAR files we will give \*\*/\*.war and in the context path we will give /train in our example.



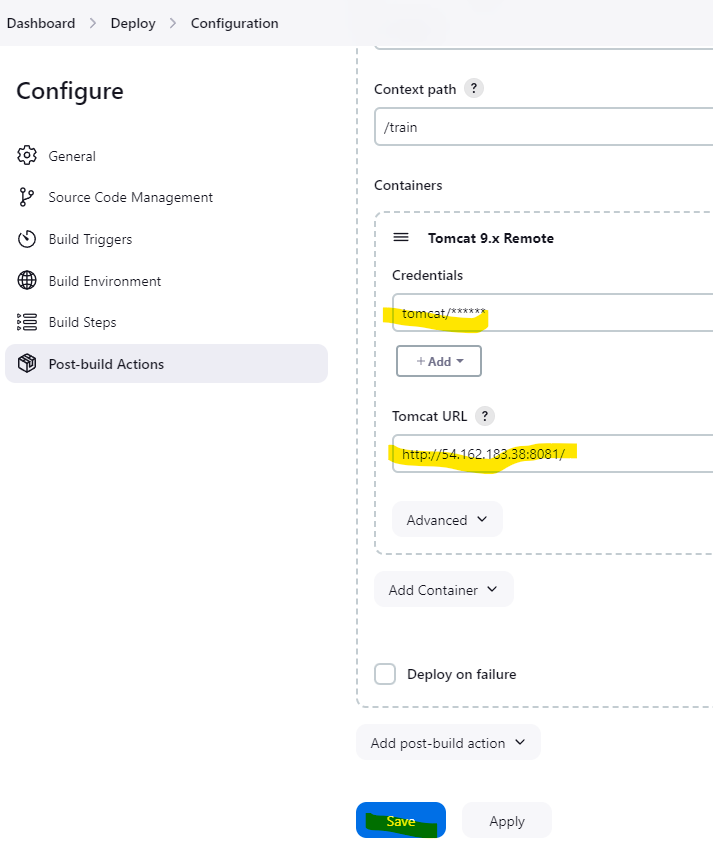
After this, under Containers > Add Container then select Tomcat 9.x Remote. Under this we will click Add the credentials.



Under Credentials we will give username – tomcat and password – tomcat (same credentials which we gave while adding roles in tomcat-users.xml file) and click Add.

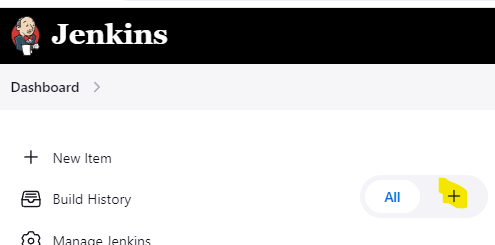


Once the credentials are added, then we will select those credentials under Tomcat 9.x Remote and also add the Tomcat URL with port 8081 and click Save.

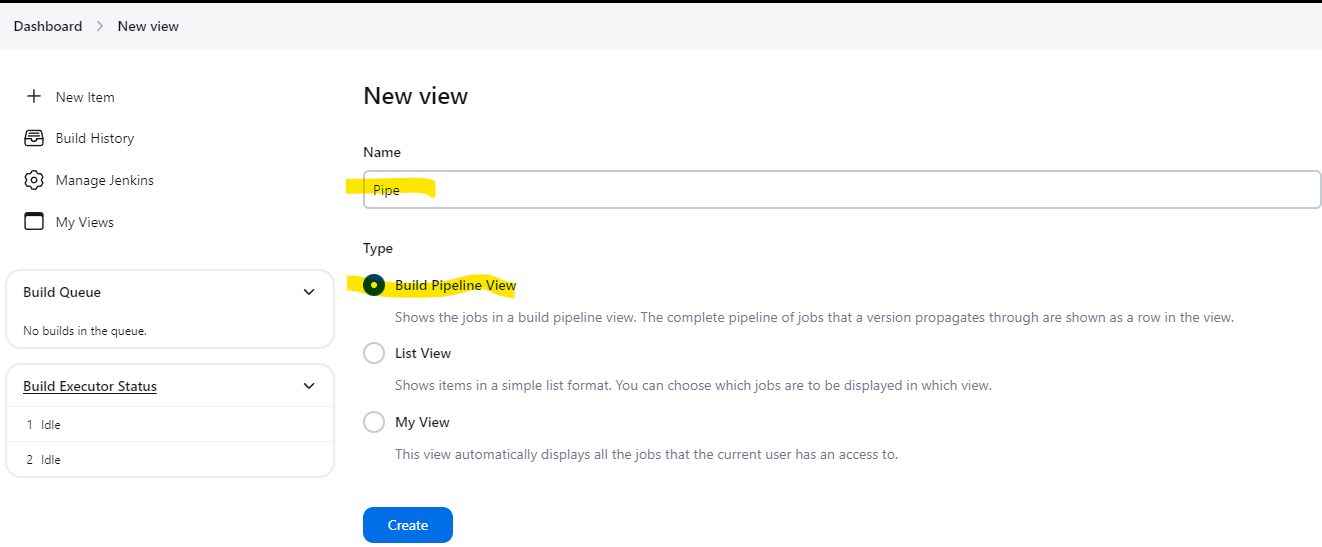


1. Now we will build the pipeline to check whether it works according to the flow.

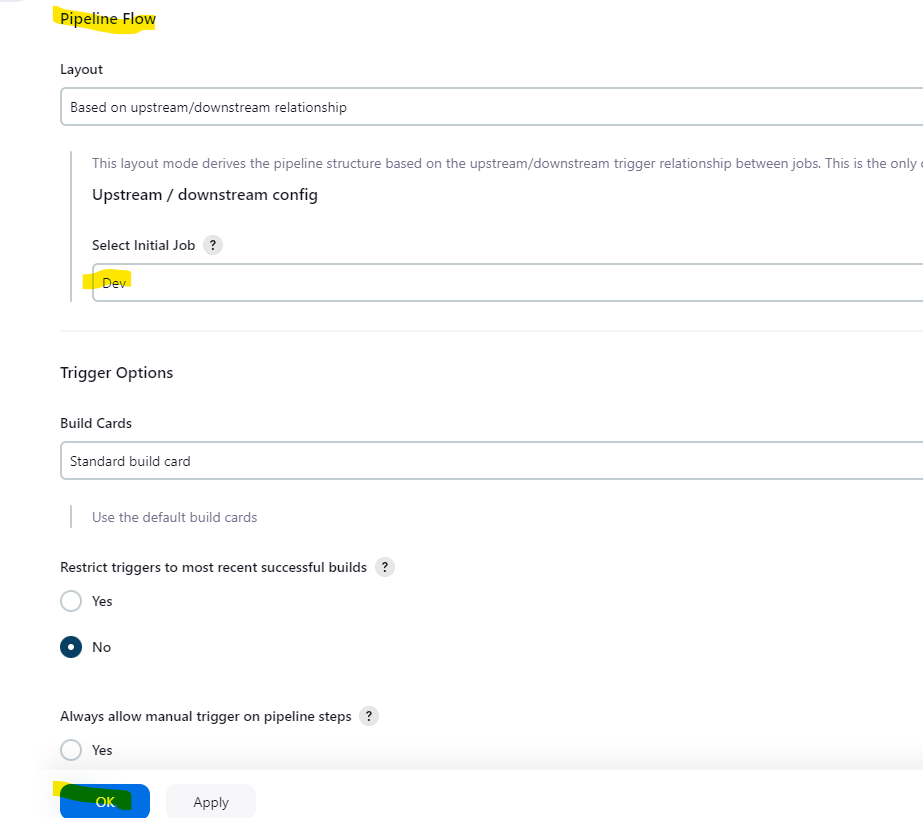
Now click on + sign on dashboard



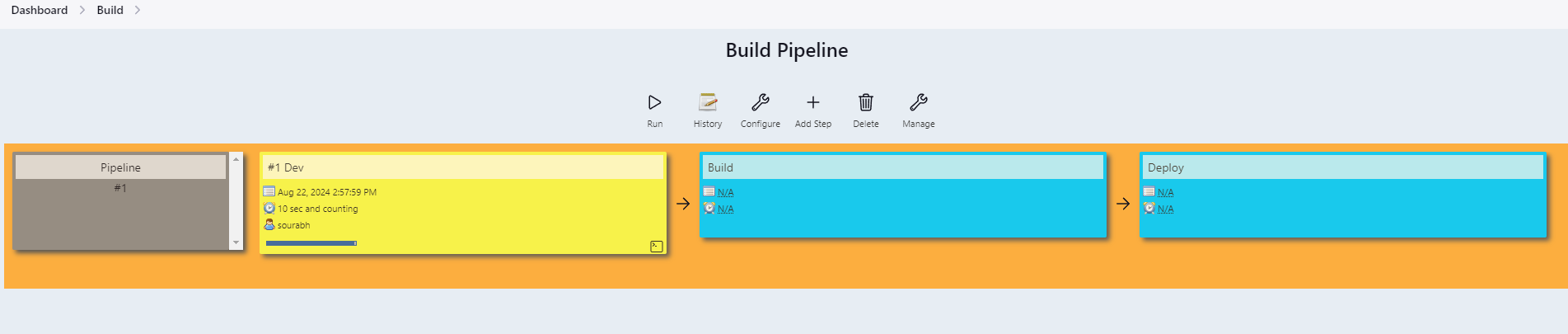
Give Pipeline view a name and select Build Pipeline View

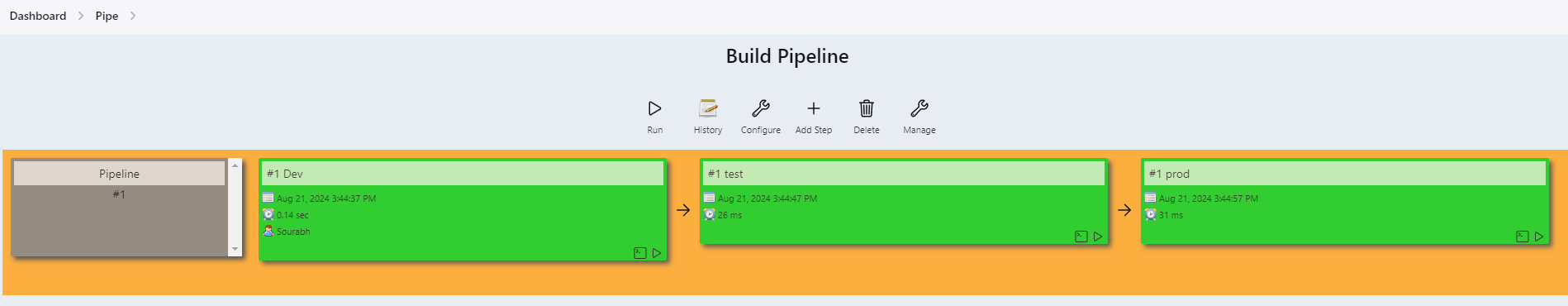


Select Initial Job as Dev under Pipeline Flow and Click OK.

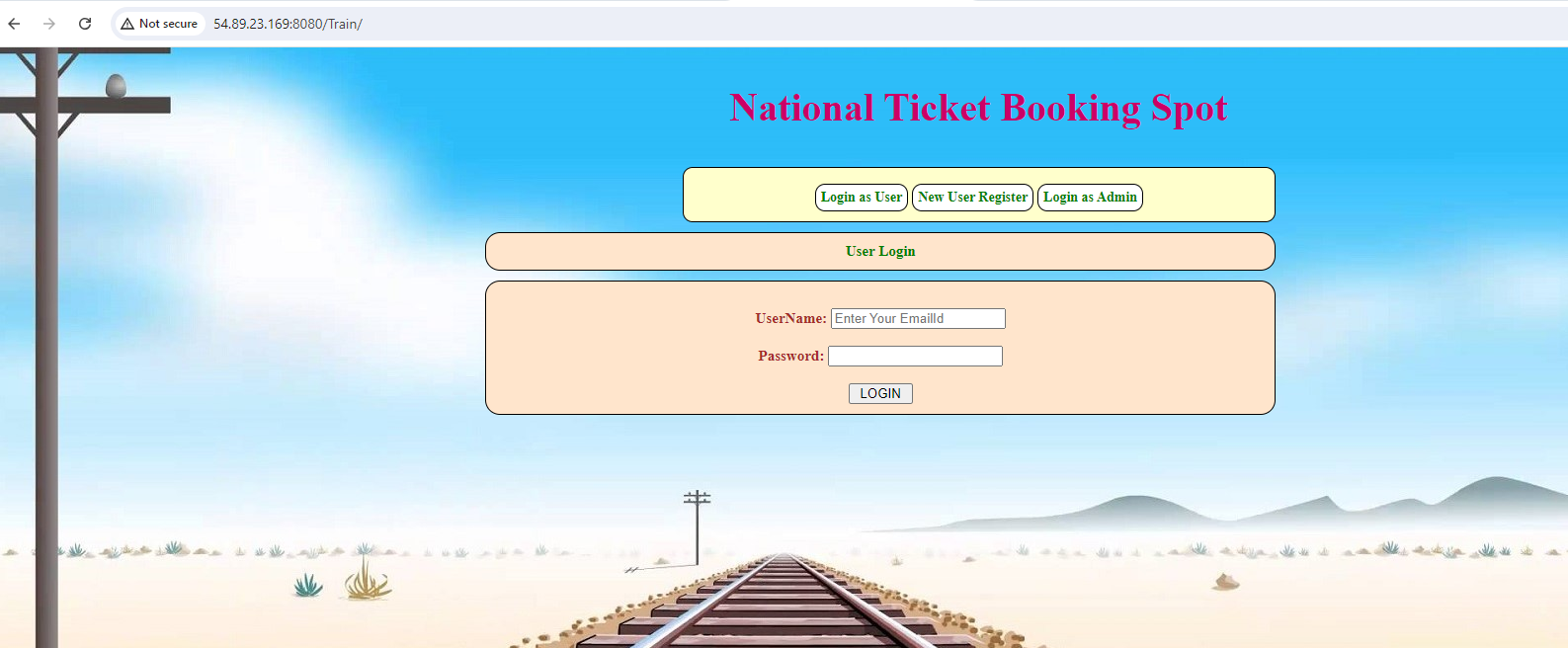


Now will run the Build Pipeline and see the output below.

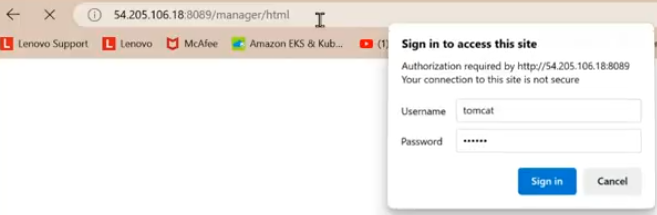




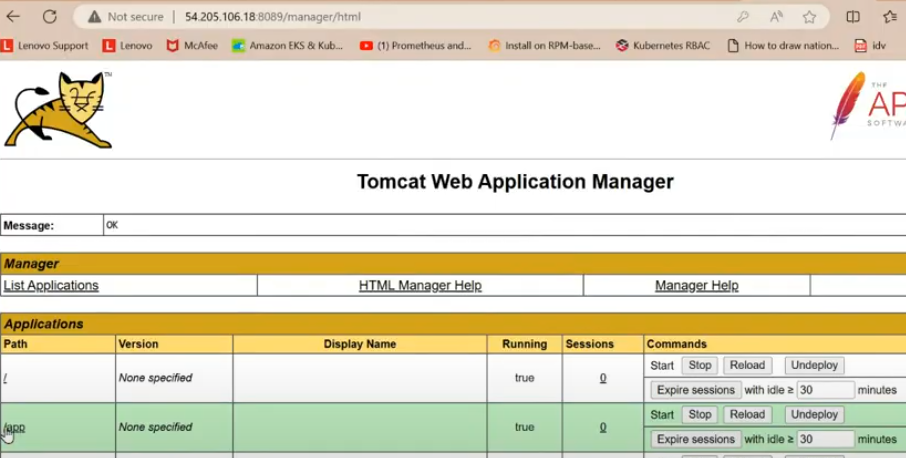
1. Our job ran successfully in the pipeline.
2. The output will come out as below, Note Screenshot details may differ but it works.

****

1. We will access the manager also in tomcat10 and see that our project is visible there. In the below screenshot, we need to enter the tomcat role credentials i.e. username – tomcat and password – tomcat



1. Below screenshot is different from the actual one. But the result will be same. /App is there in screenshot instead of /train.



1. Automate Project build is completed using Jenkins.