**ASI Insurance**

**Devops Capstone Project 1**

**Description**

ASI Insurance is facing challenges in improving the SLA to its customers due to its organizational growth and existing monolithic application architecture. It requires transformation of the existing architecture to a microservice application architecture, while also implementing DevOps pipeline and automations.

The successful completion of the project will enable ASI Insurance to improve its overall application deployment process, enhance system scalability, and deliver better products and services to its customers.

**Task (Activities)**

1. Create the Dockerfile, Jenkinsfile, Ansible playbook, and the source file of the static website

2. Upload all the created files to GitHub

3. Go to the terminal and install NodeJS 16

4. Open the browser and access the Jenkins application

5. Create Jenkins pipeline to perform CI/CD for a Docker container

6. Create Docker Hub Credentials and other necessary pre-requisites before running build

7. Set up Docker remote host on AWS and configure deploy stage in pipeline

8. Execute Jenkins Build

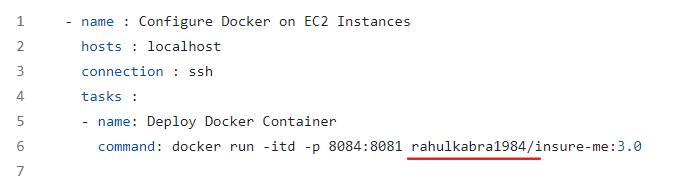
9. Access deployed application on Docker container

**Step 1 : Create the Dockerfile, Jenkinsfile, Ansible playbook**

1. Update the Jenkinsfile with the following code and commit changes

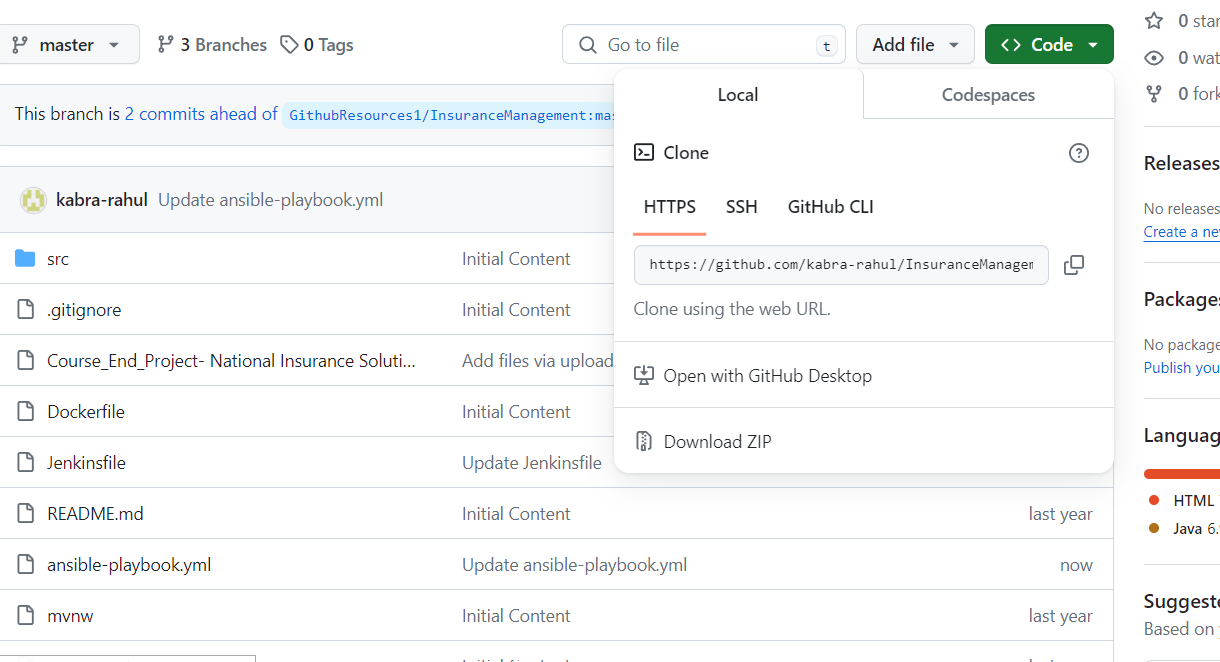


1. Update the ansible-playbook with with following code and commit the changes



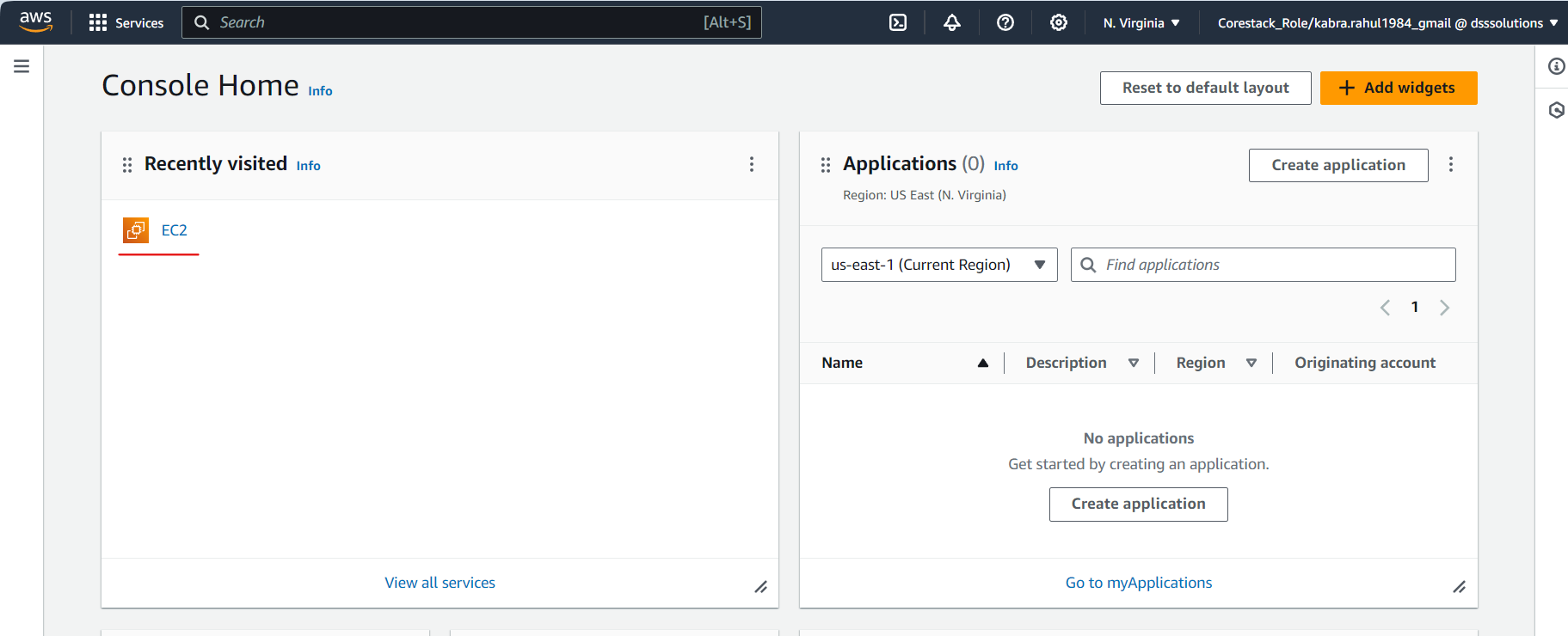
1. Copy the project link

**https://github.com/kabra-rahul/InsuranceManagement.git**

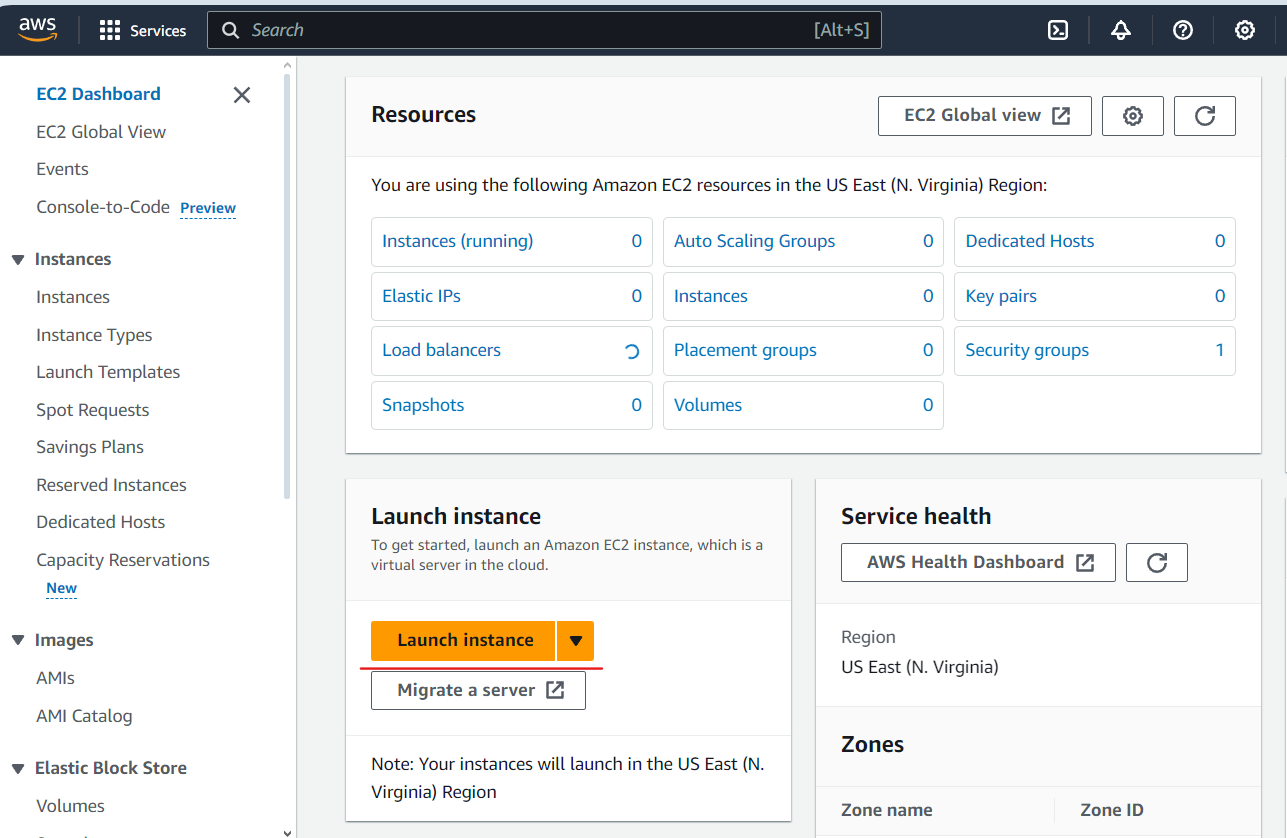


**Step 2 : Create an EC2 Instance**

* 1. Open AWS account using and click on EC2.



* 1. Click on “Launch Instance”



1.3 Fill in the below details and Click on Launch Instance

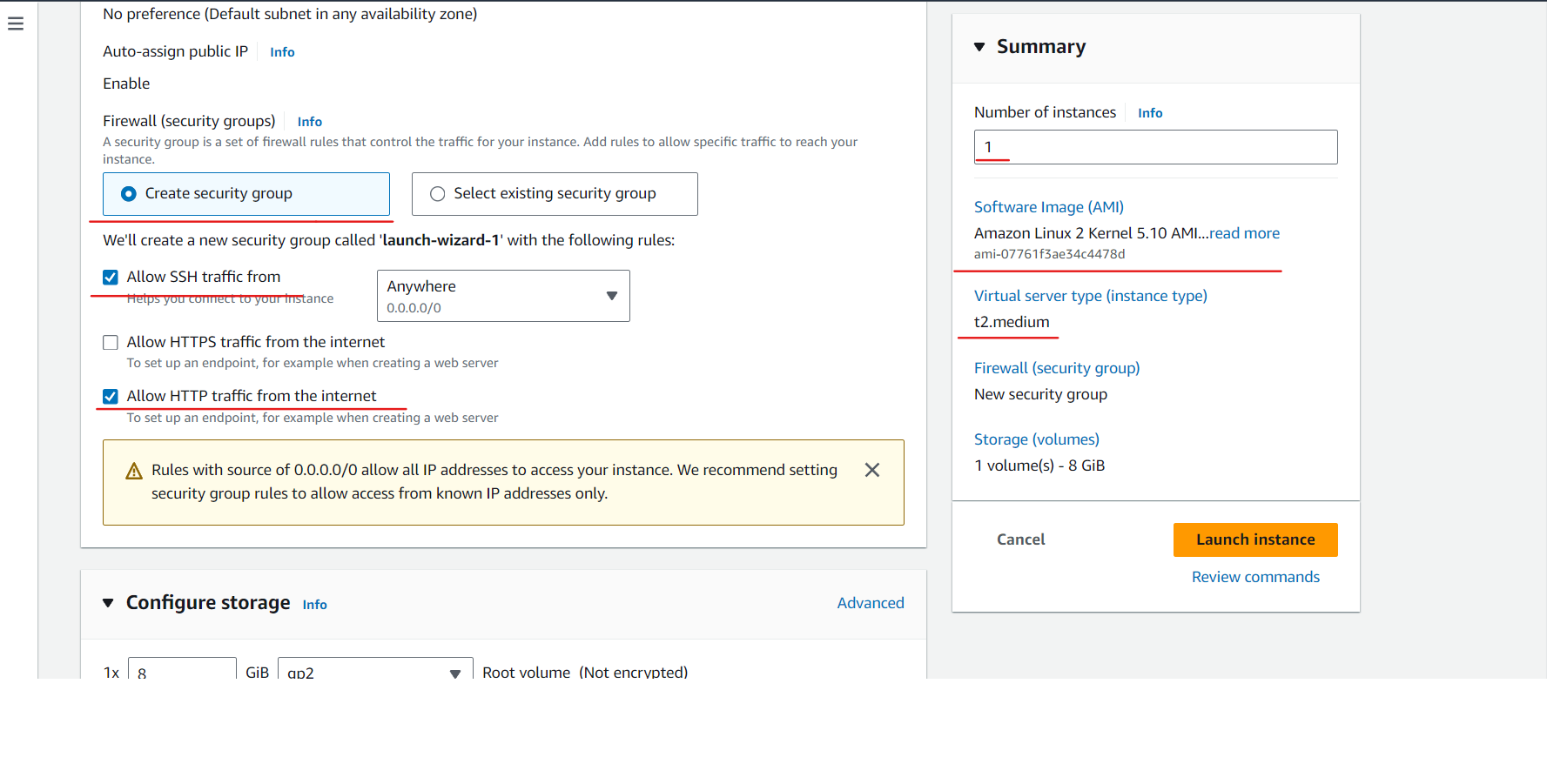
**Name – devops-server**

**Number of Instance -1**

**AMI - Amazon Linux 2 Kernel 5.10 AMI 2.0.20240223.0 x86\_64 HVM gp2Instance**

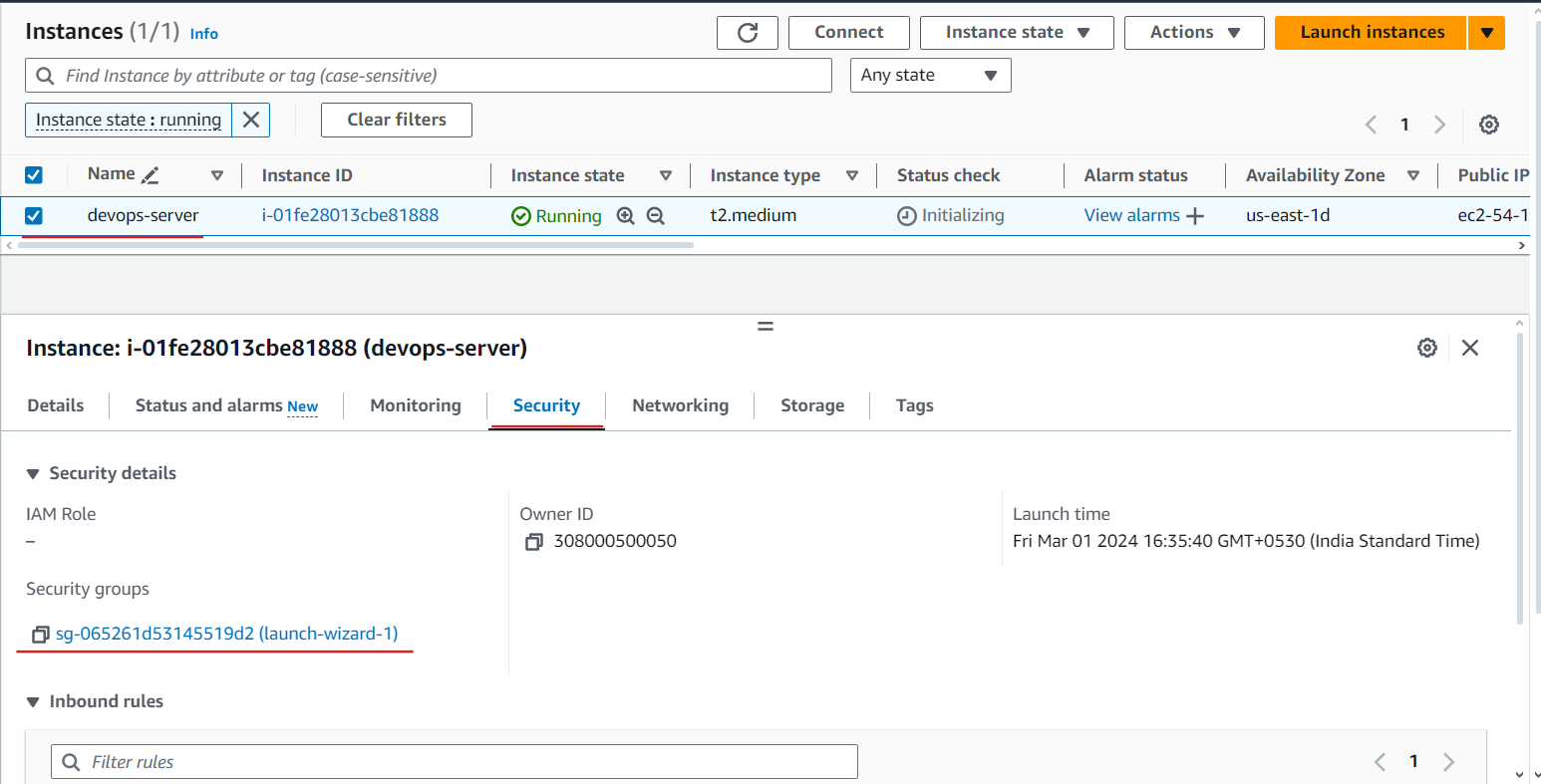
**Type – t2.medium**

**Select Create Security Group –> Select Allow SSH Traffic and HTTP traffic from**

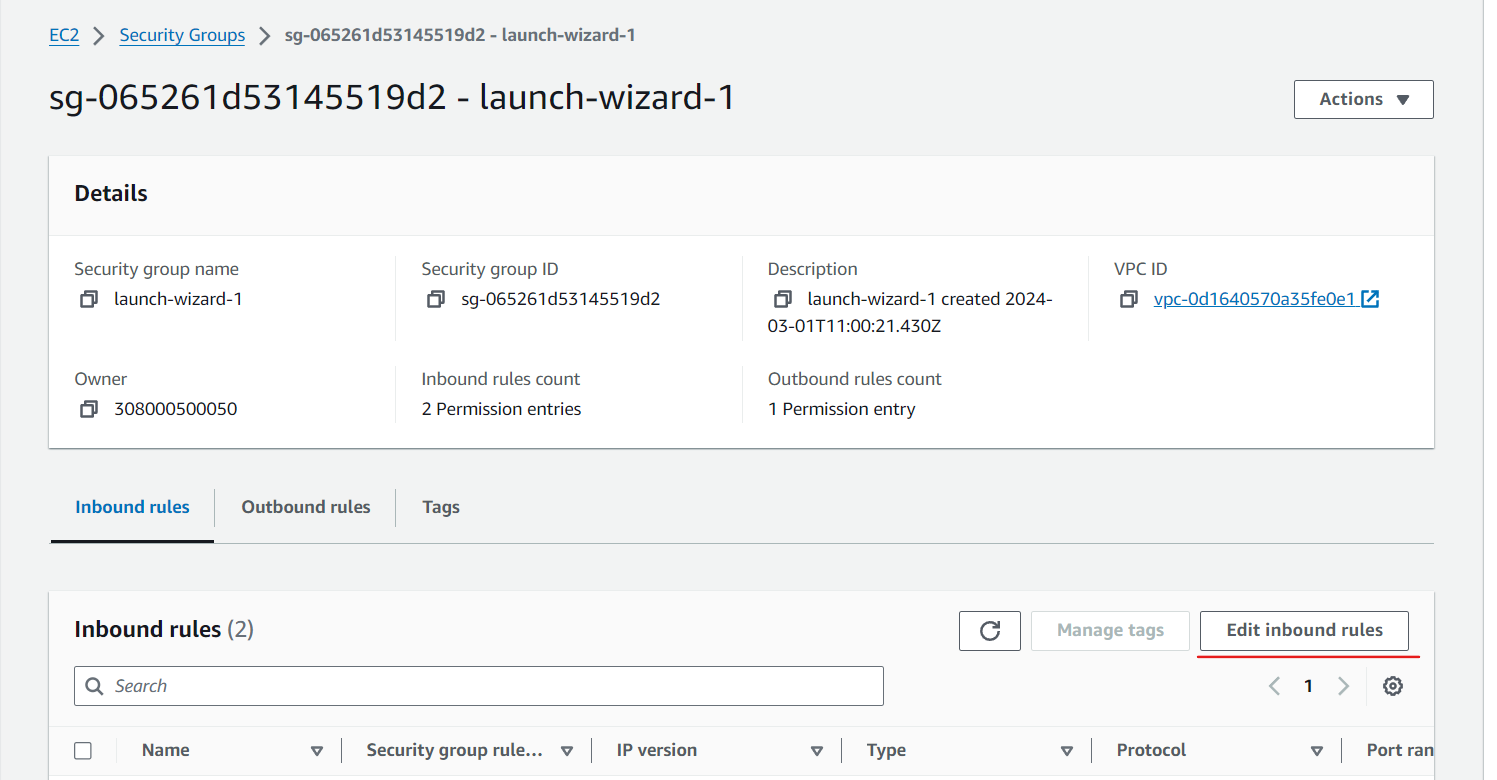


* 1. Select the devops-server instance and select the Security tab in instance details and

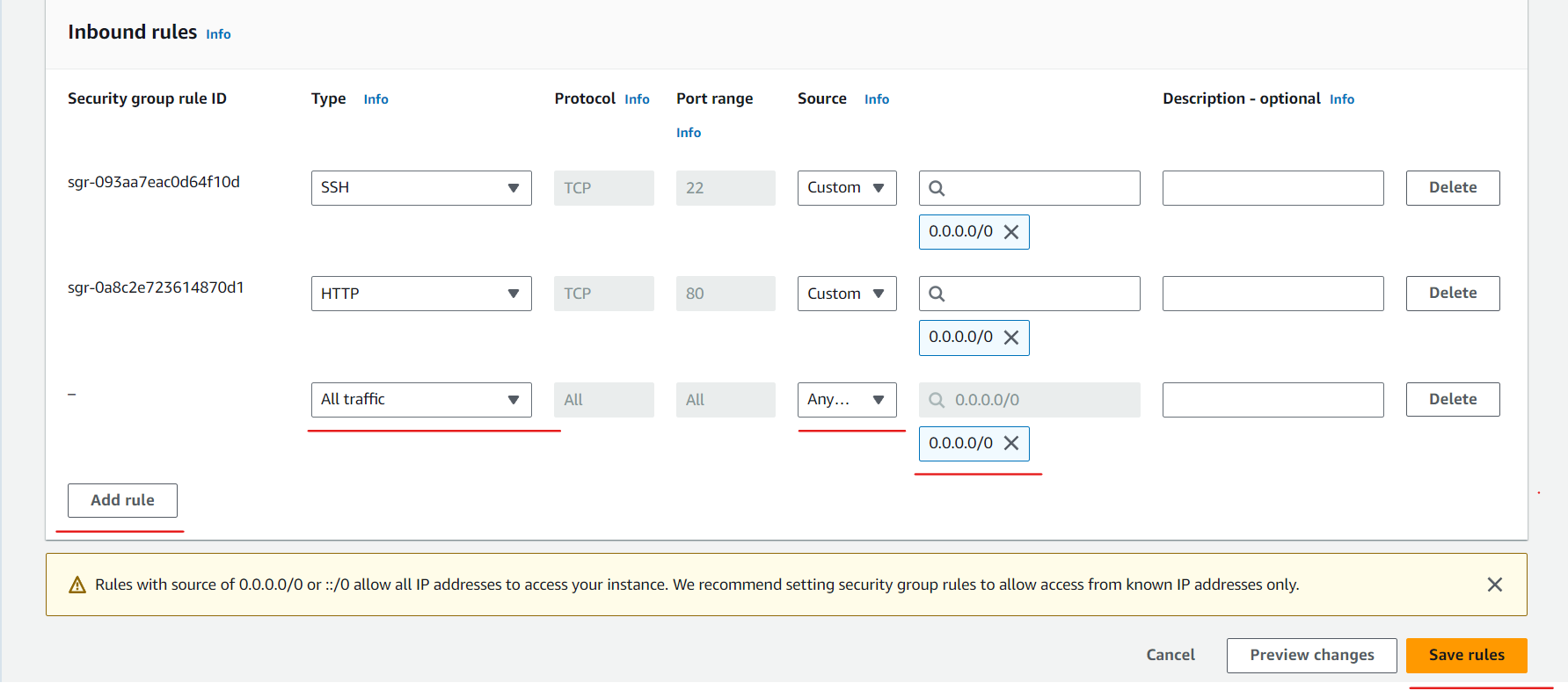
Click on the security group

****

* 1. Click on **Edit inbound rules**

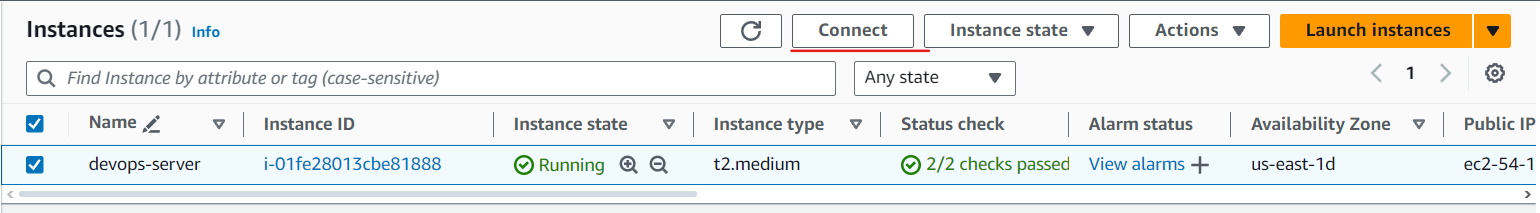
****

* 1. Now click on **Add rule and** Add the below details and click on **Save rule**

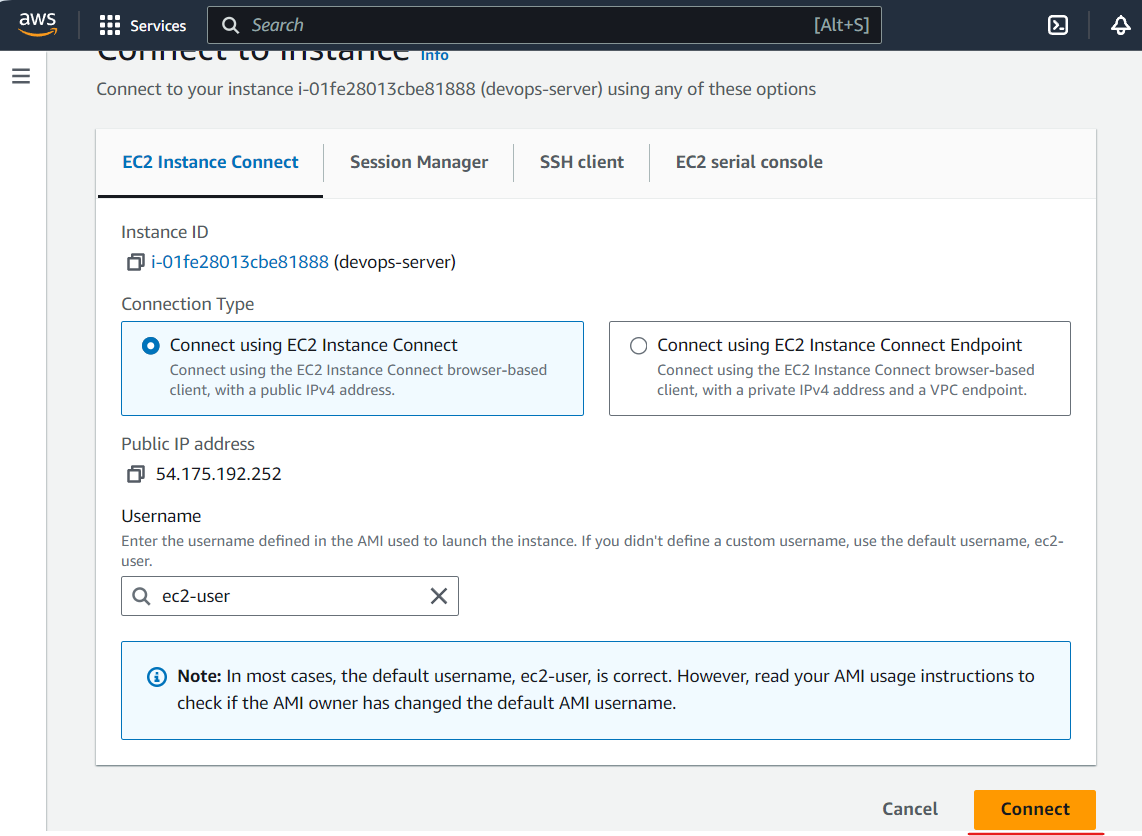
****

**Step 3 : Connecting to EC2 Machine and Installing package**

3.1 Go to EC2 instance home and select master and click on **Connect**



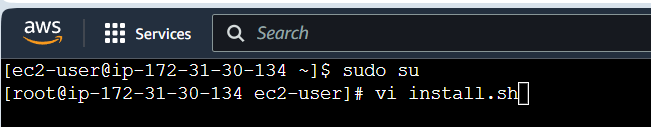
3.2 Click on **Connect**



3.3 Run the below commands in the terminal and create a file install.sh

**sudo su**

**vi install.sh**

****

3.4 Type the below command in the install.sh

**sudo yum update –y**

**sudo wget -O /etc/yum.repos.d/jenkins.repo \https://pkg.jenkins.io/redhat-stable/jenkins.repo**

**sudo amazon-linux-extras install java-openjdk11 -y**

**sudo rpm --import https://pkg.jenkins.io/redhat-stable/jenkins.io-2023.key**

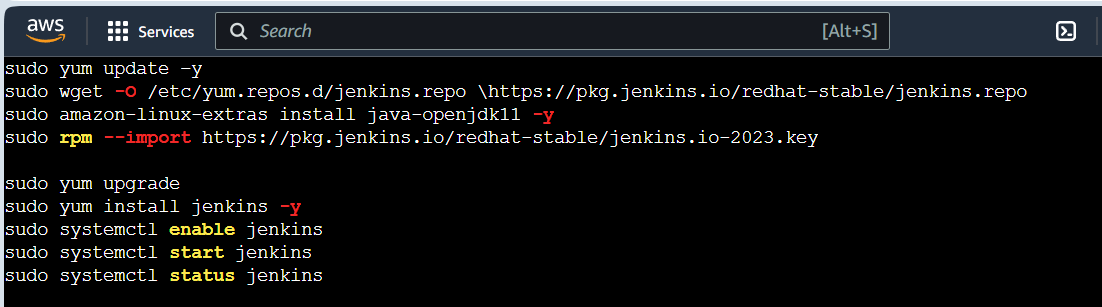
**sudo yum upgrade**

**sudo yum install jenkins -y**

**sudo systemctl enable jenkins**

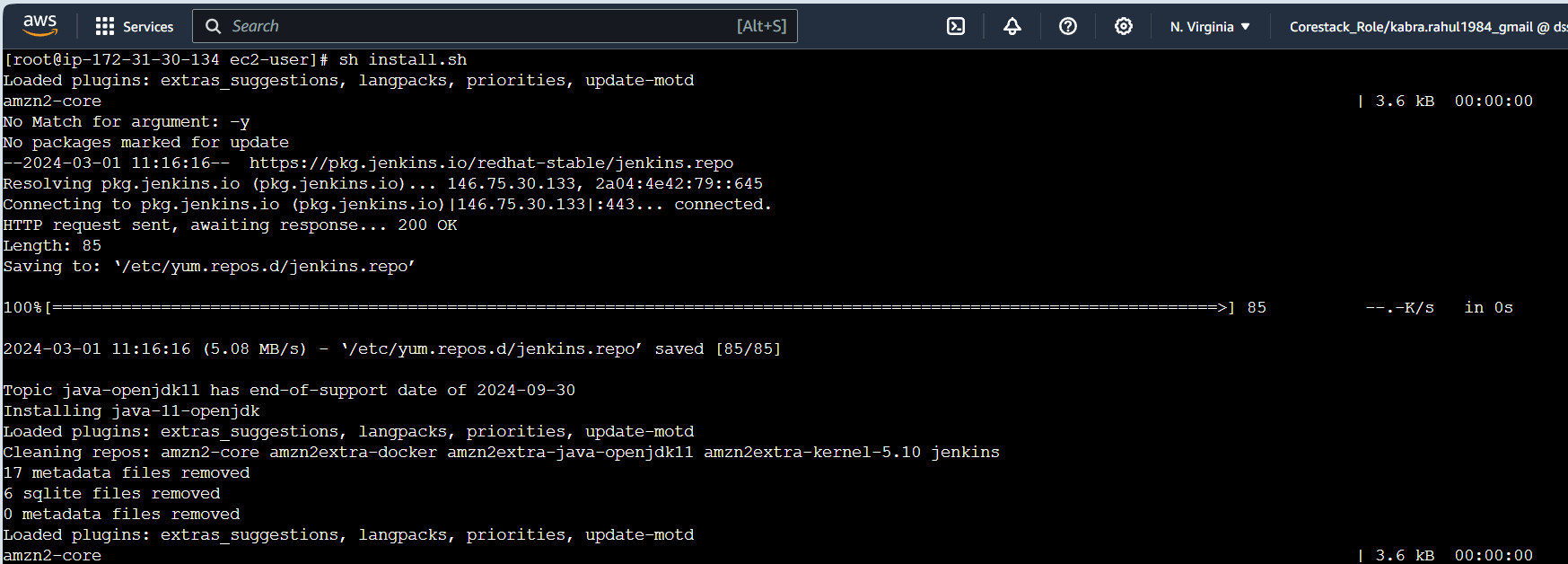
**sudo systemctl start jenkins**

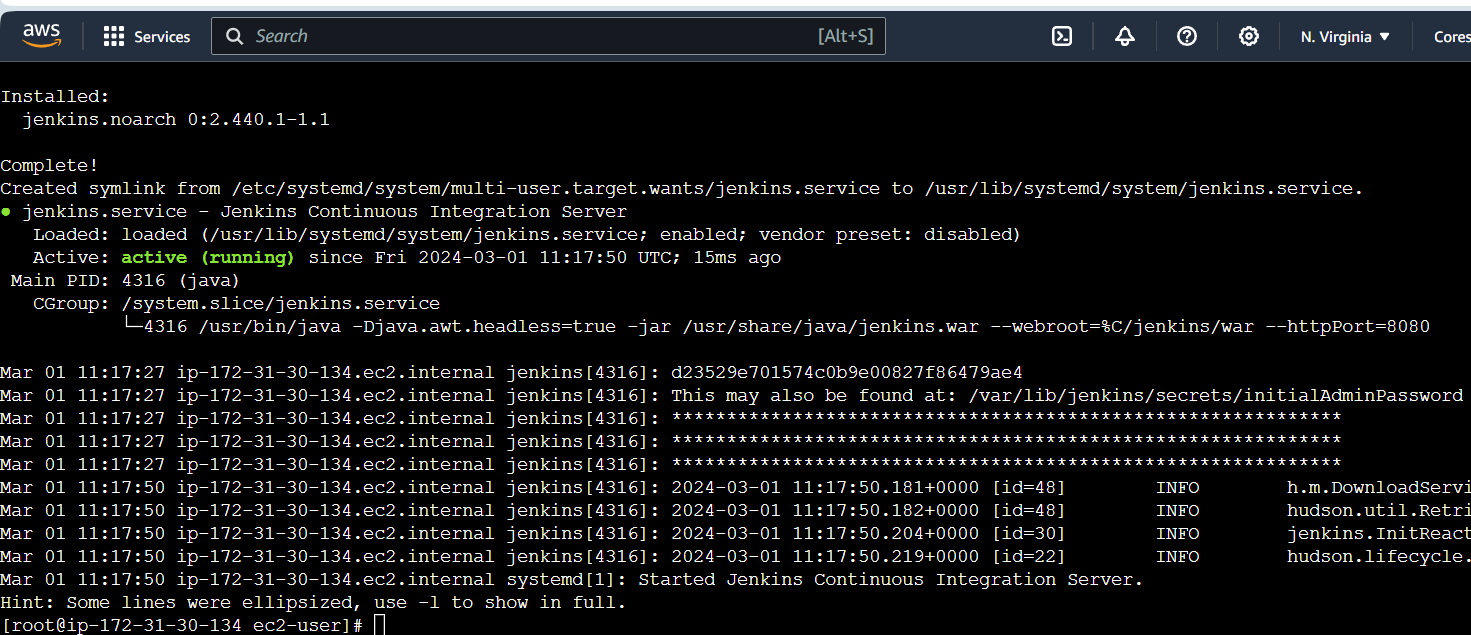
**sudo systemctl status jenkins**



3.5 Run below command to install

**sh install.sh**

****

****

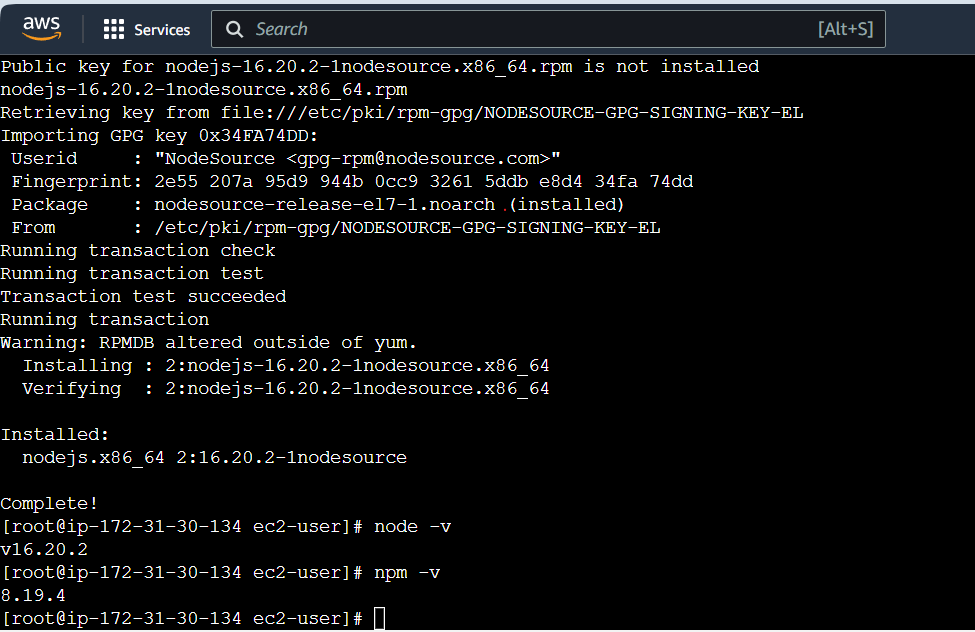
3.6 Run the below command to install Nodejs16 and verify it

**curl -fsSL https://rpm.nodesource.com/setup\_16.x | sudo bash -**

**sudo yum install -y nodejs**

**node -v**

**npm -v**

****

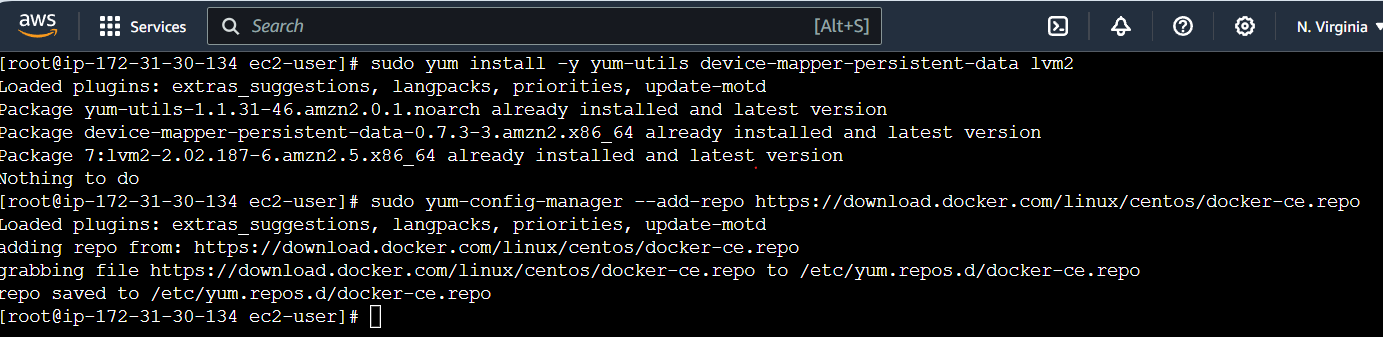
3.7 Run the below command to install Docker and verify it

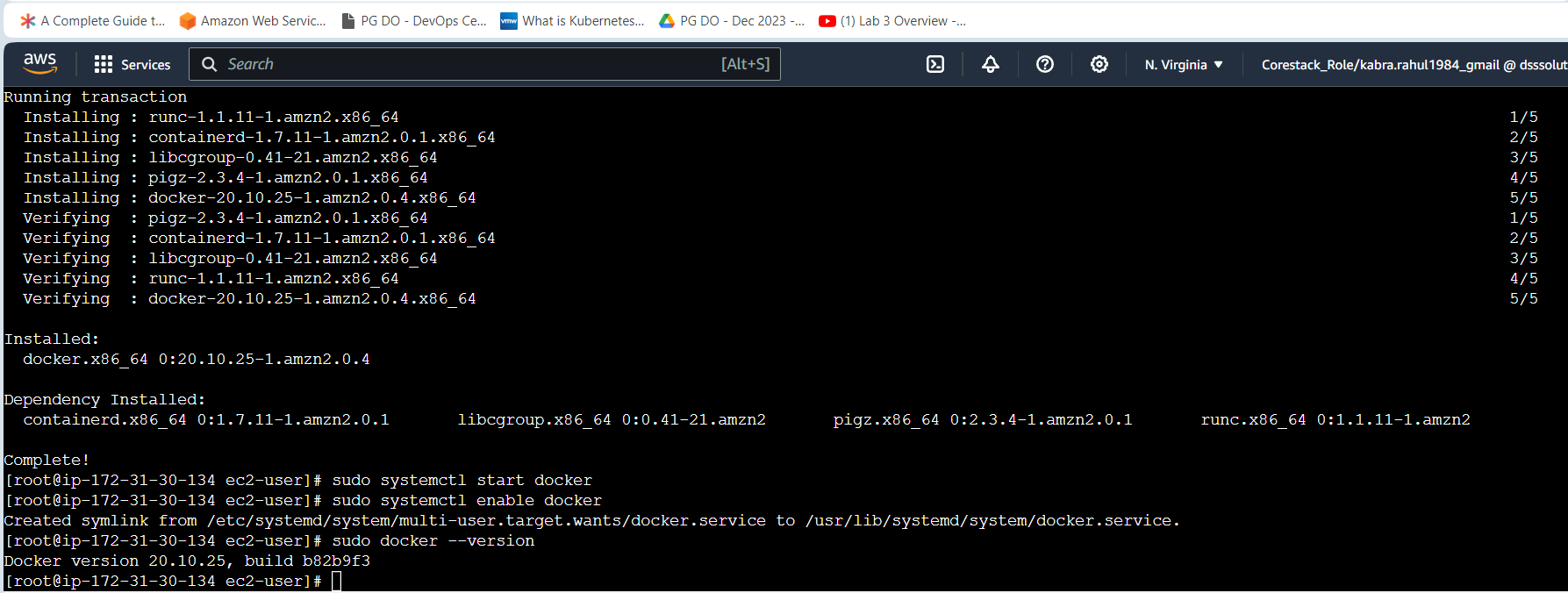
**sudo yum install -y docker**

**sudo systemctl start docker**

**sudo systemctl enable docker**

**sudo docker --version**

****

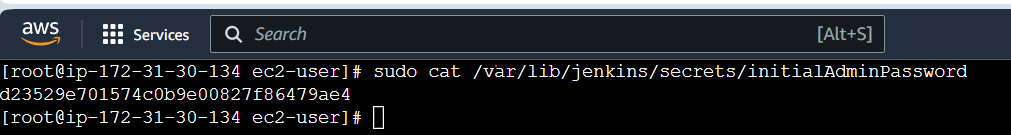
****

**Step 4 : Accessing Jenkins**

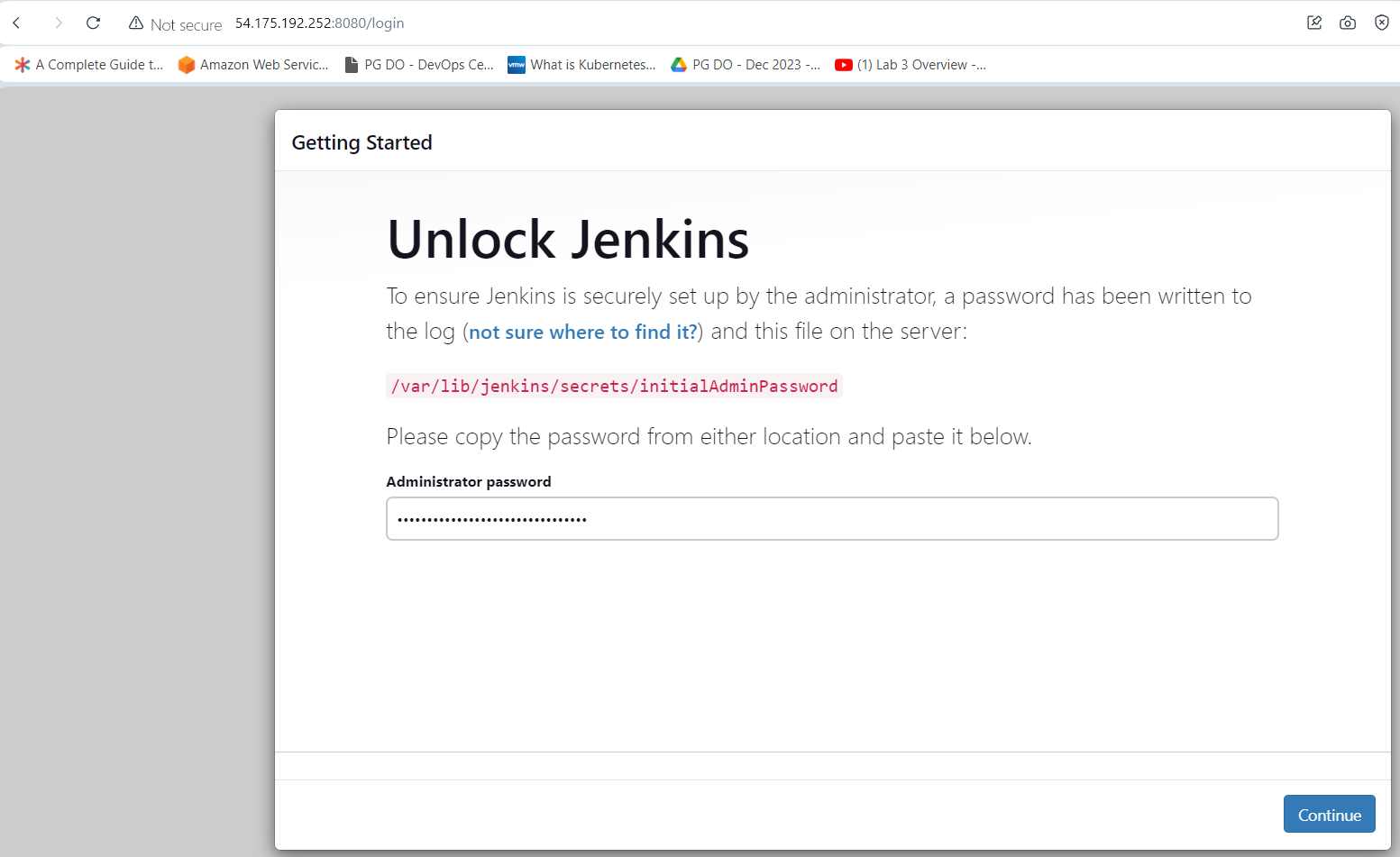
* 1. Open  [**http://54.175.192.252:8080/**](http://3.88.187.77:8080/) in the browser



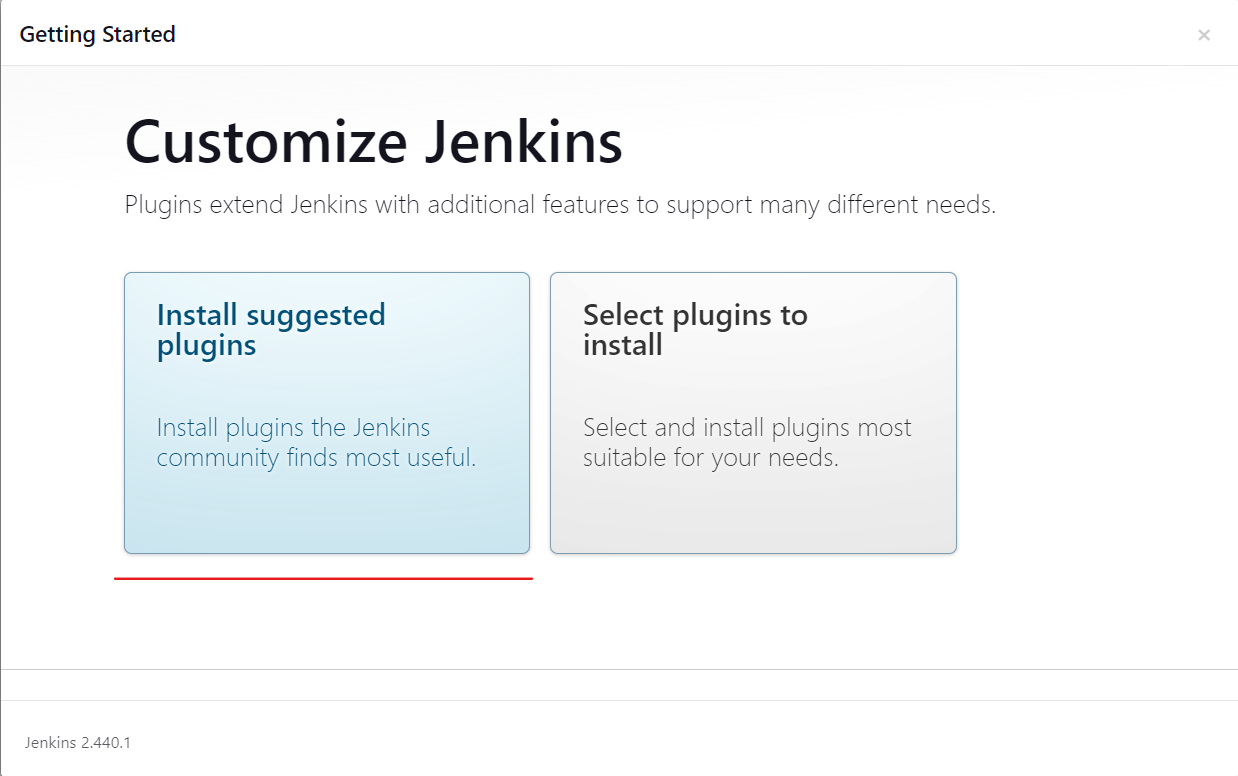
* 1. In your terminal run the following command: ***sudo cat /var/lib/jenkins/secrets/initialAdminPassword***

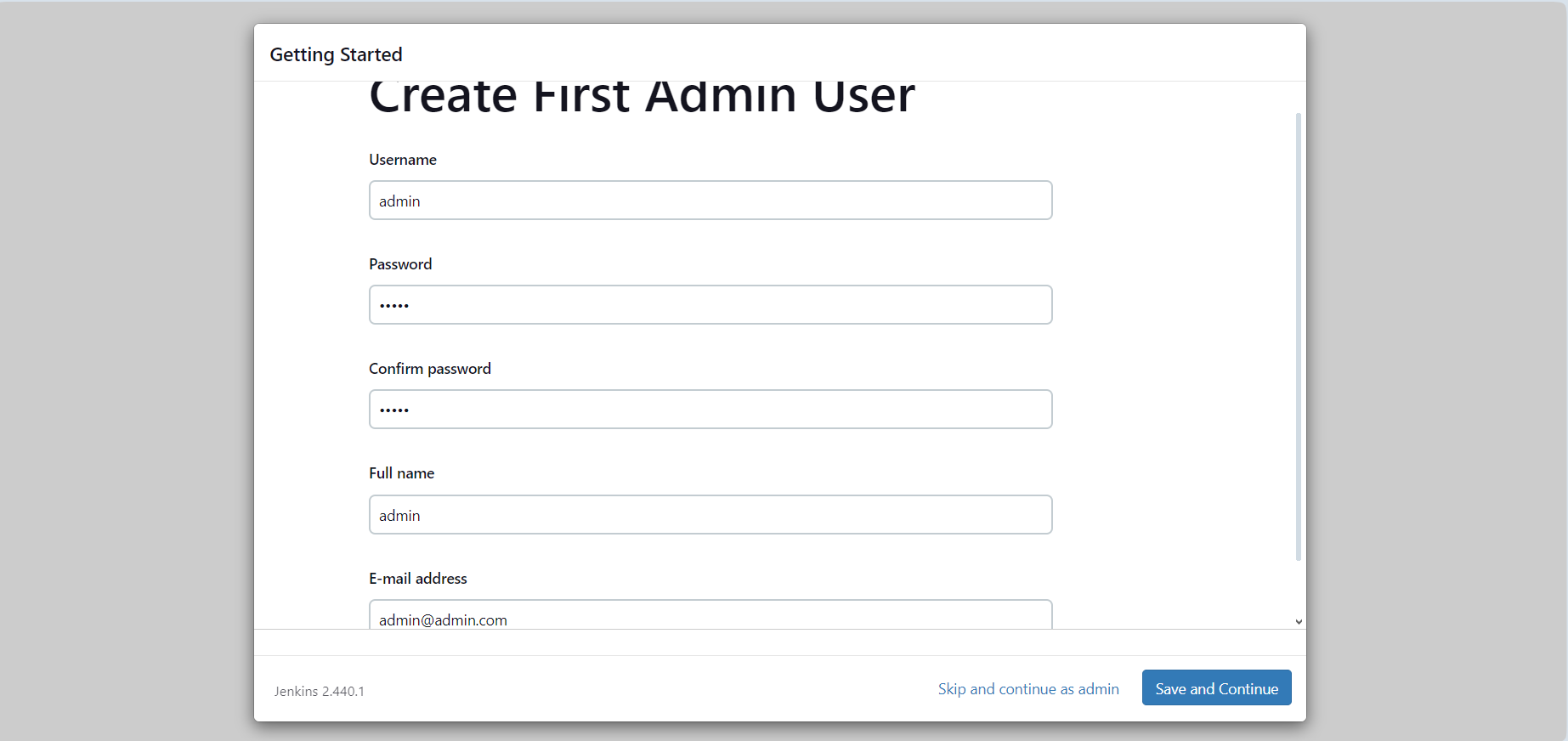


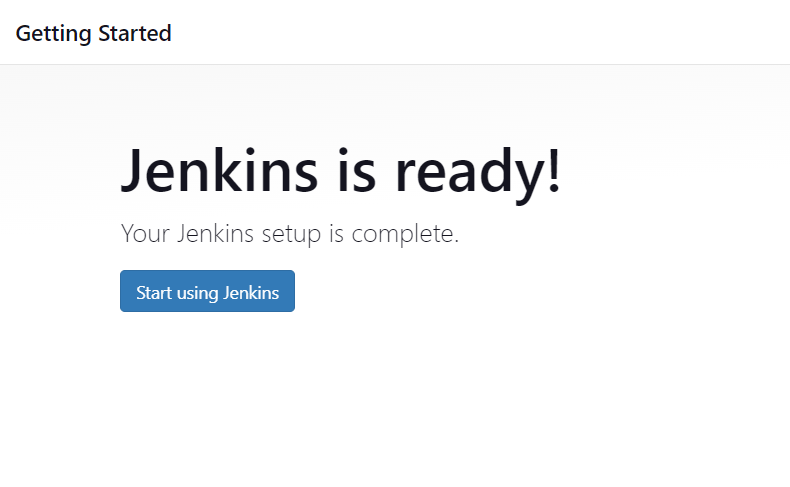
* 1. Copy this password and paste it your Jenkins page in the browser and click continue



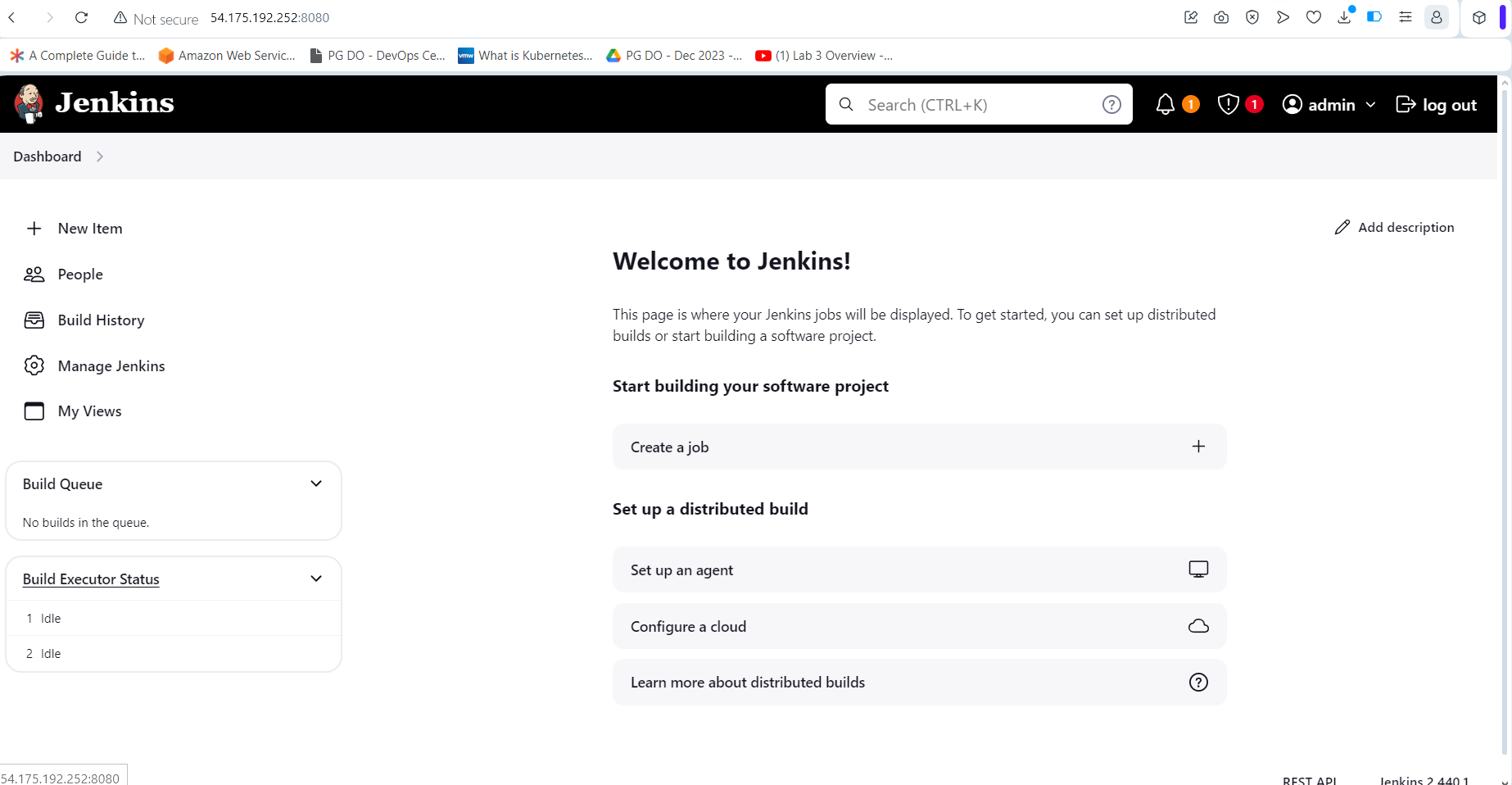
* 1. Now, click on I**nstall the suggested plugins**



* 1. You can create an admin user by filling below details
  2. In the Instance configuration page, click on the **Start using Jenkins** button.

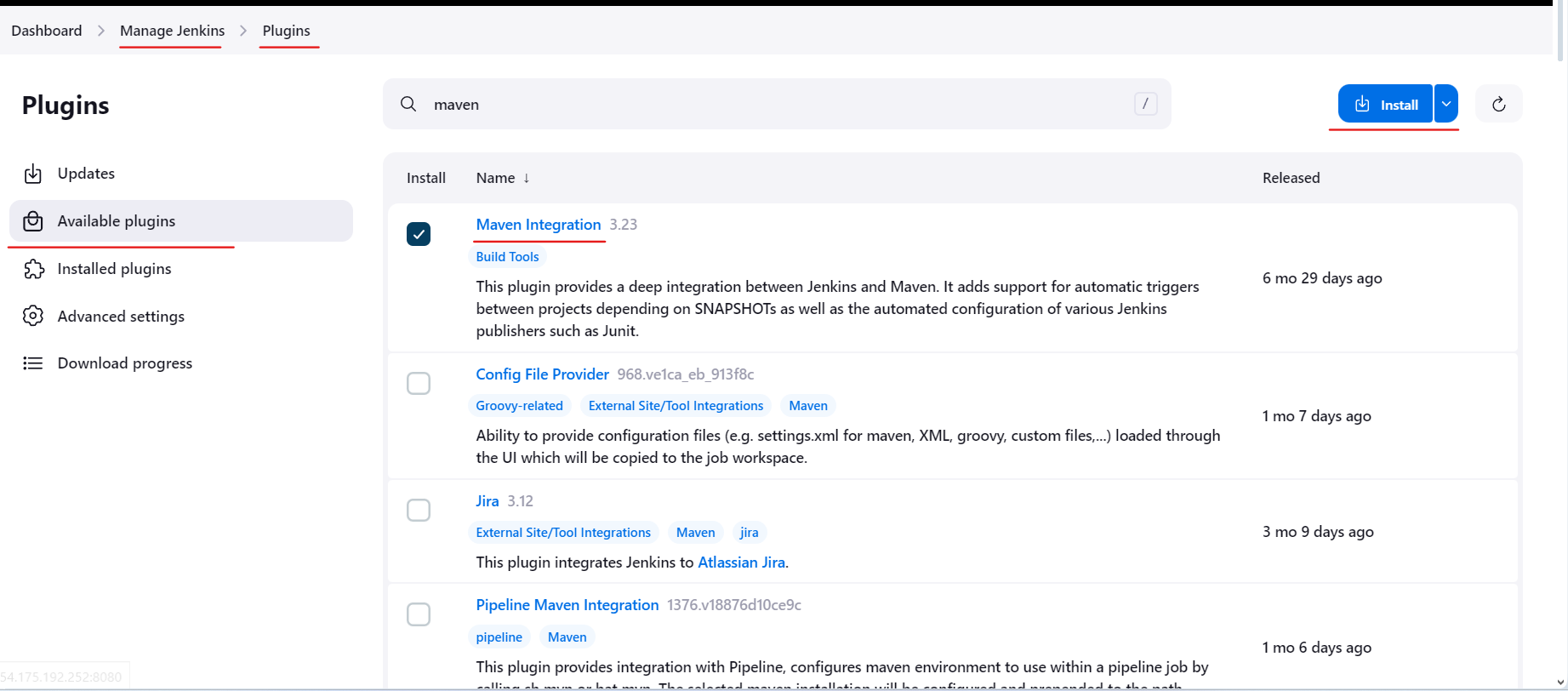


* 1. Now, you can work with Jenkins as shown in the screenshot below



**Step 5 : Installing Plugins and configuring Jenkins**

1. Go to Manage Jenkins -> Plugins -> Available Plugins -> Maven and then click install



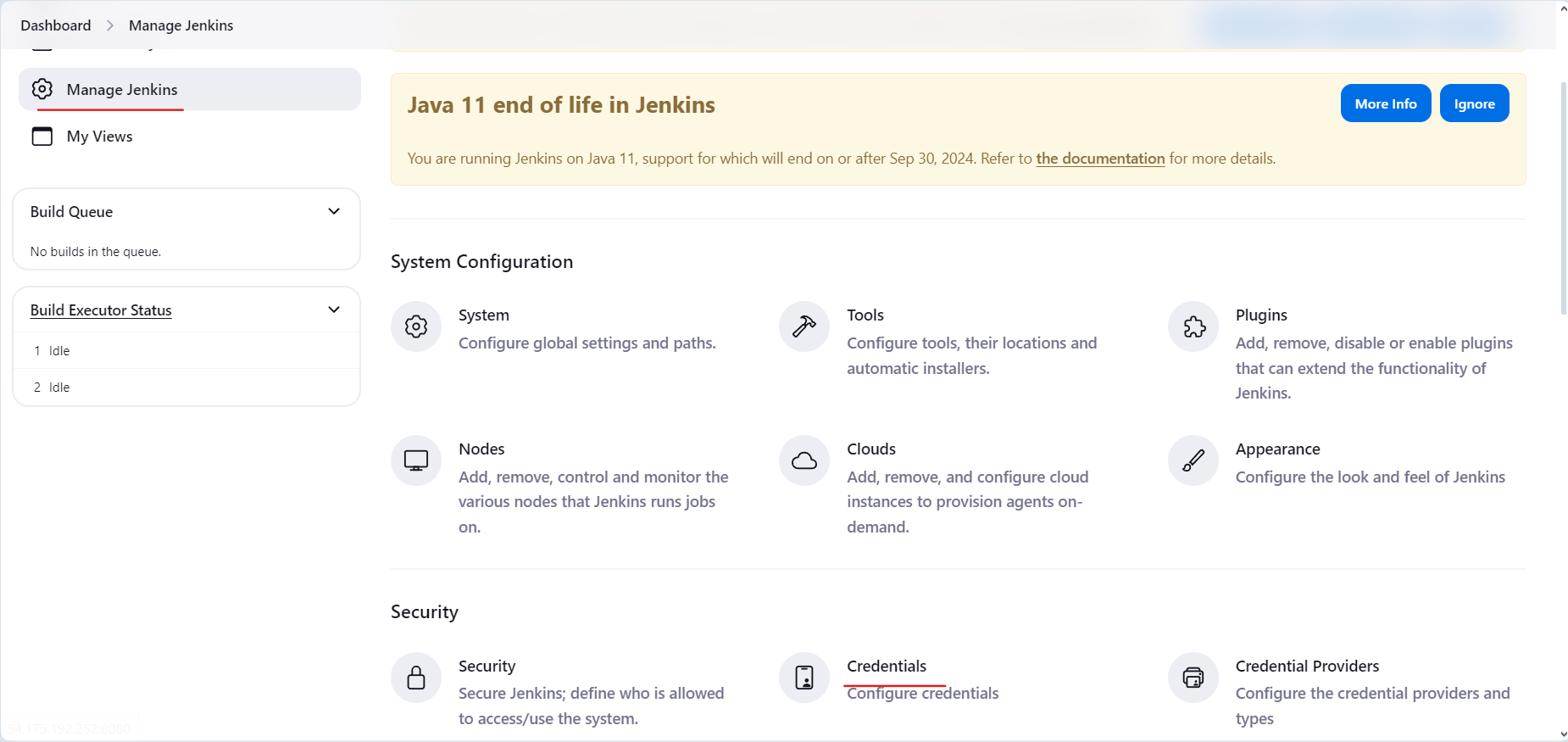
1. Similarly install the following plugins

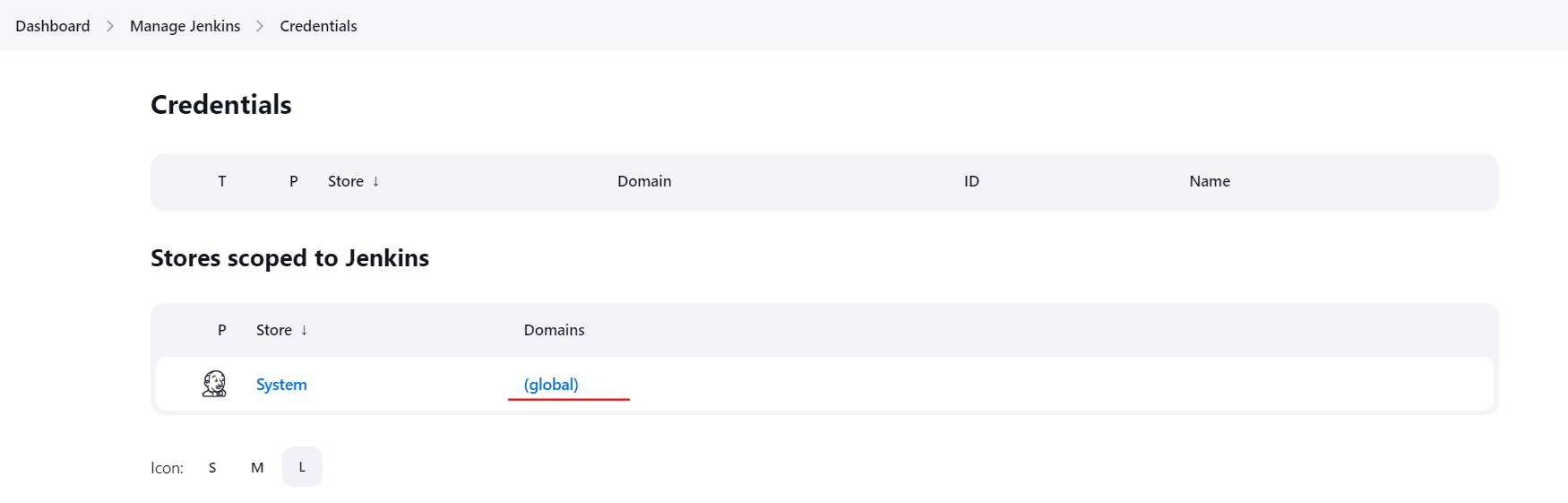
**Ansible**

**Docker Pipeline / Docker**

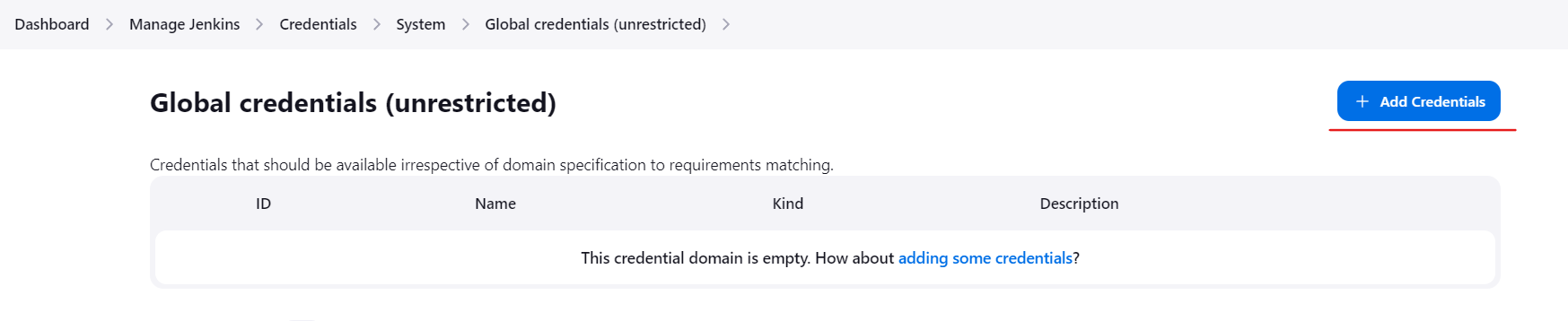
**HTML Publisher**

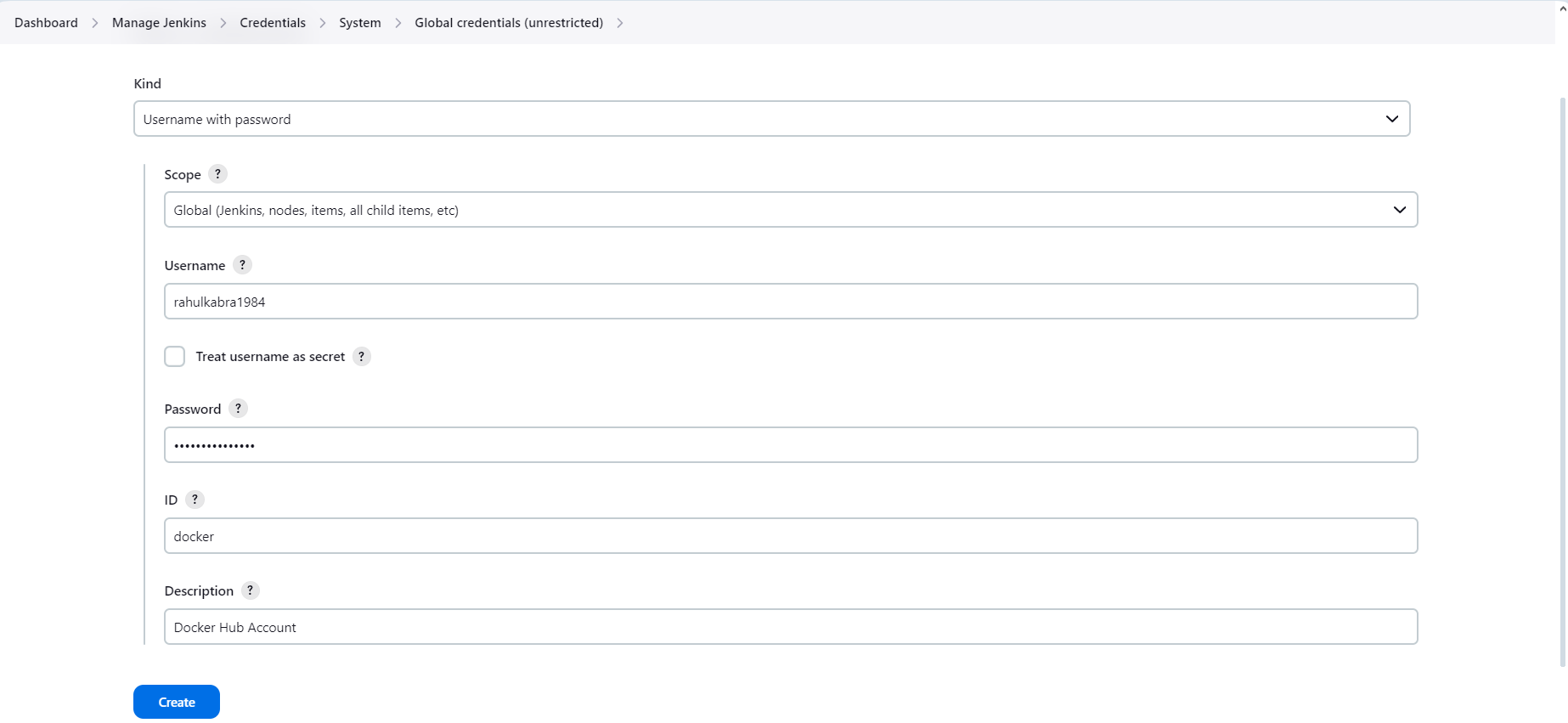
1. Now navigate to Manage Jenkins 🡪 Credentials and then select global domain.





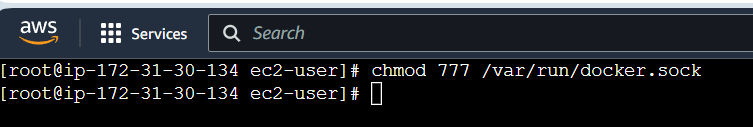
1. Click on new Credential by clicking on Add Credentials to create a new Docker hub credentials as per below details: (credential id should match the one you mentioned in jenkinsfile





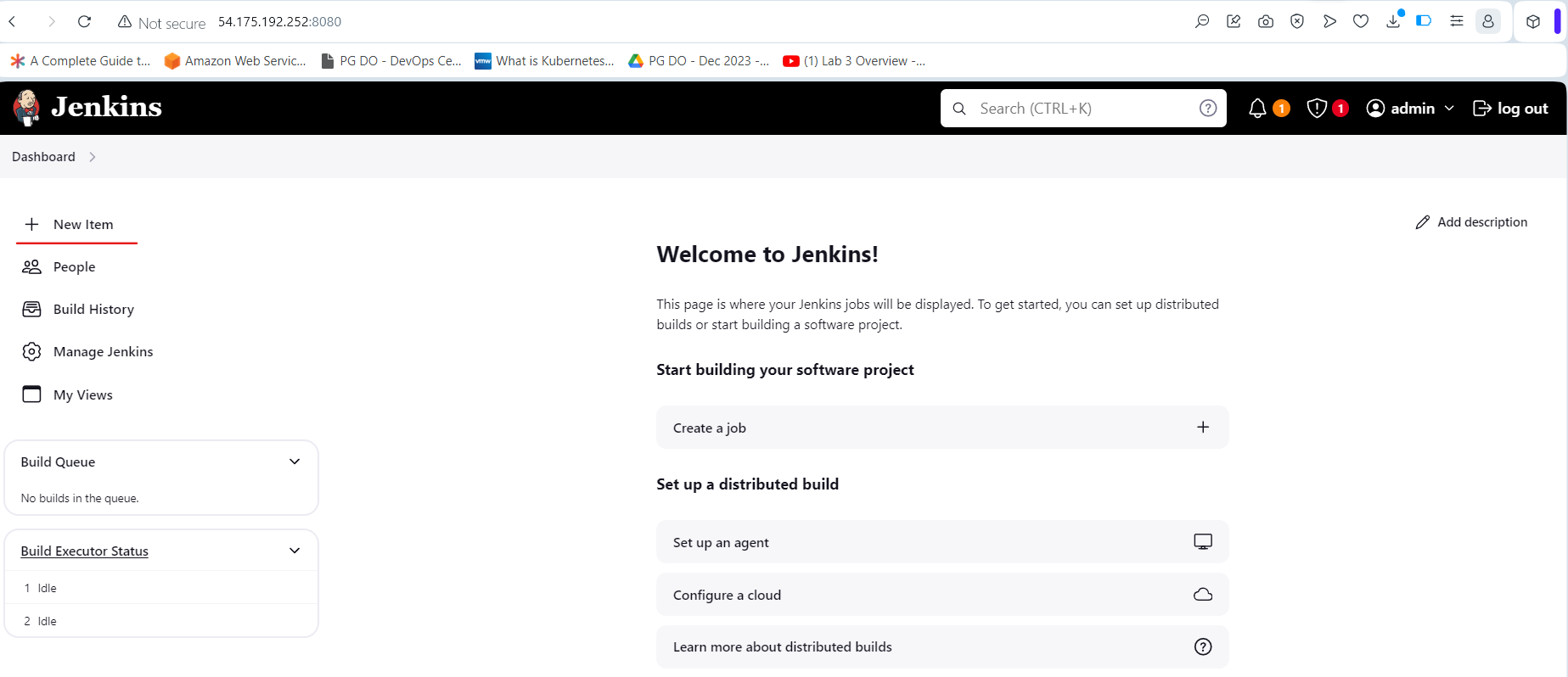
1. Next provide full access to Docker Sock file using below command:

**chmod 777 /var/run/docker.sock**

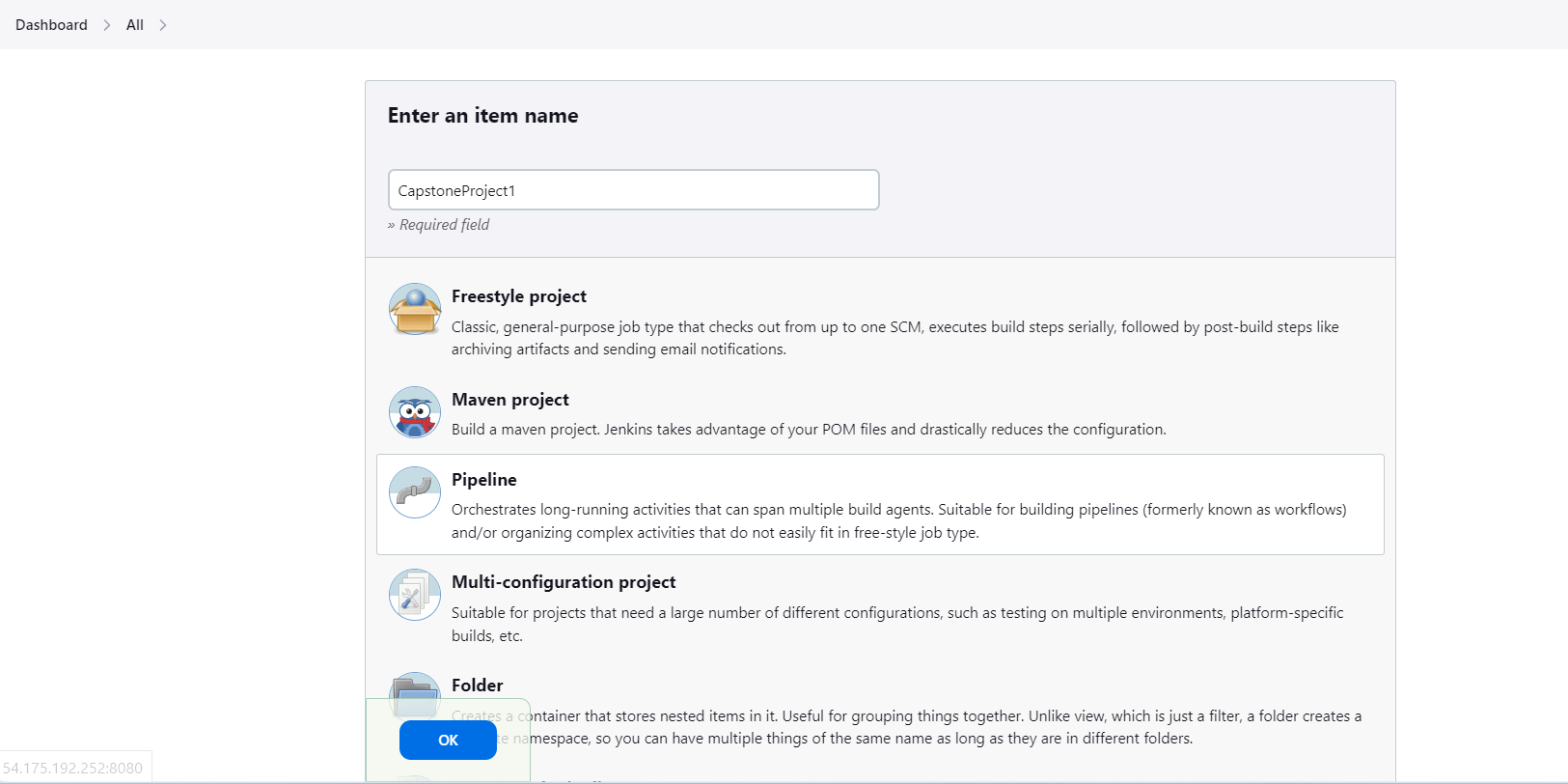


**Step 6 : Create Jenkins Pipeline**

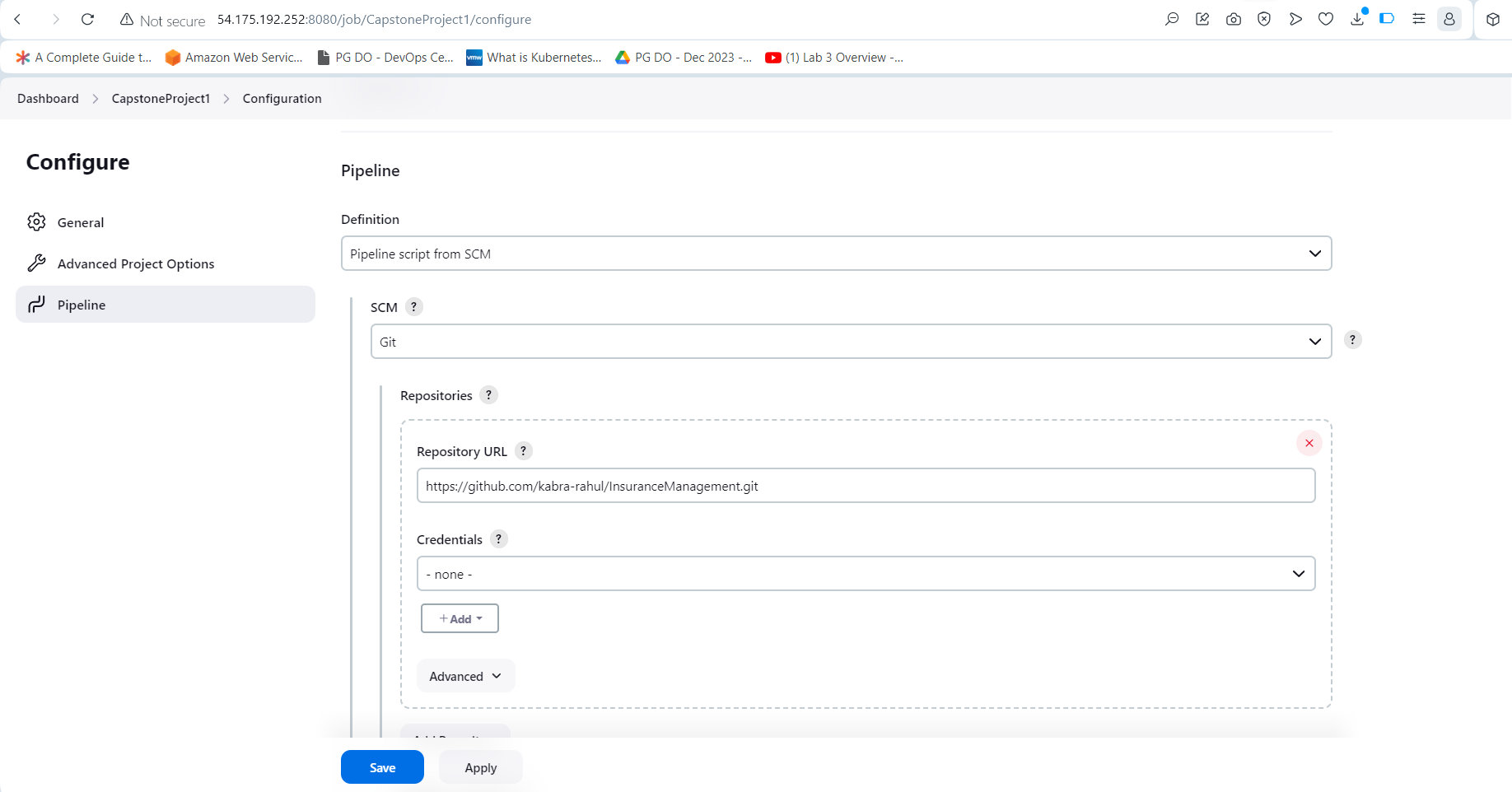
1. Access Jenkins application and click on New Item to create a new Jenkins job:

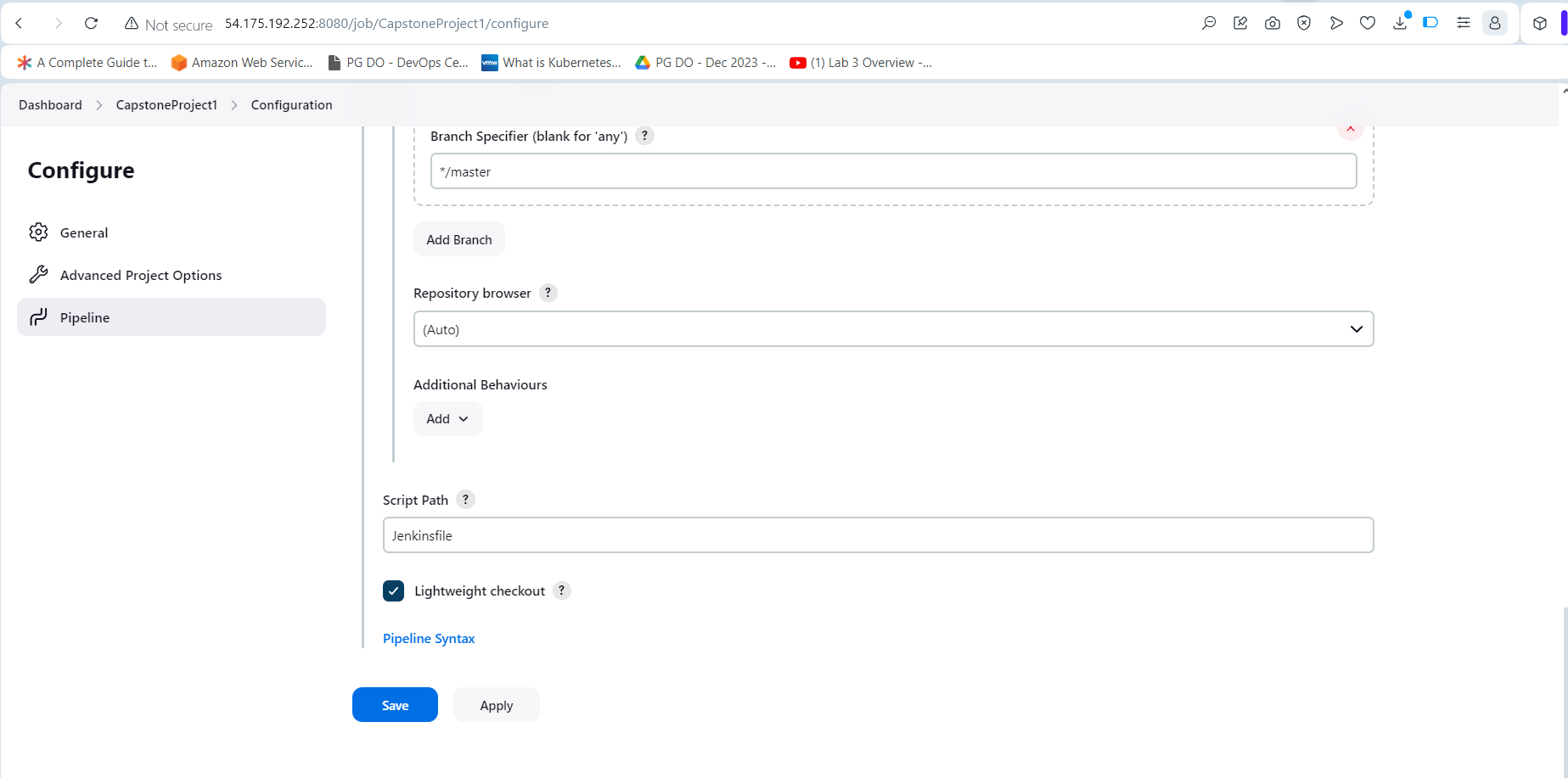


1. Select desired Jenkins pipeline job type and fill in job name as per project requirement:



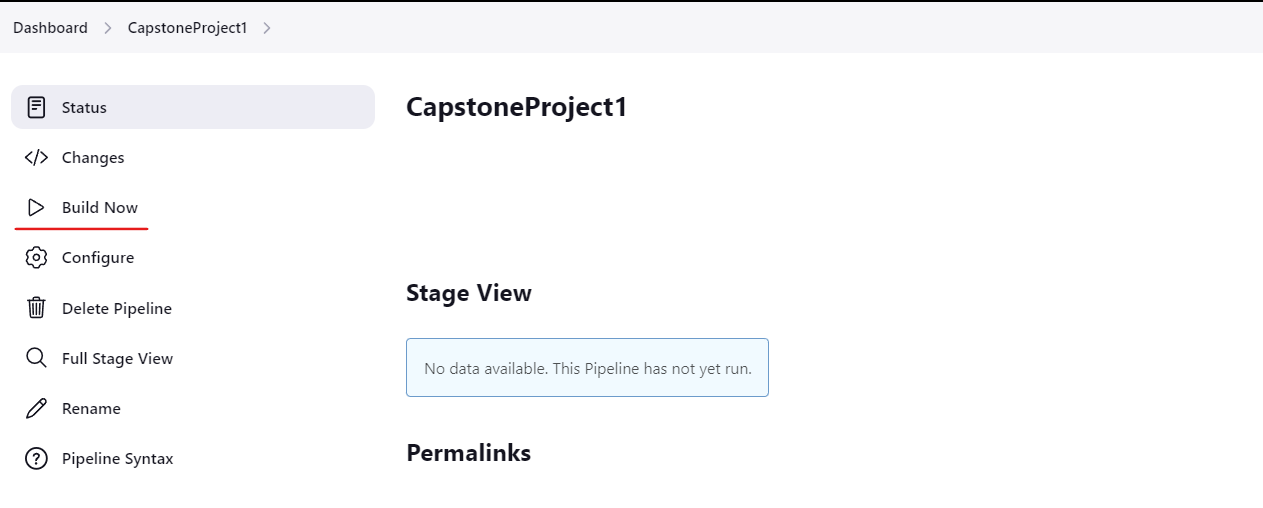
1. Once Clicked on Ok you will be navigated to Jenkins job configuration page where we can provide Jenkins job details such as parameters, Jenkins pipeline configuration etc.





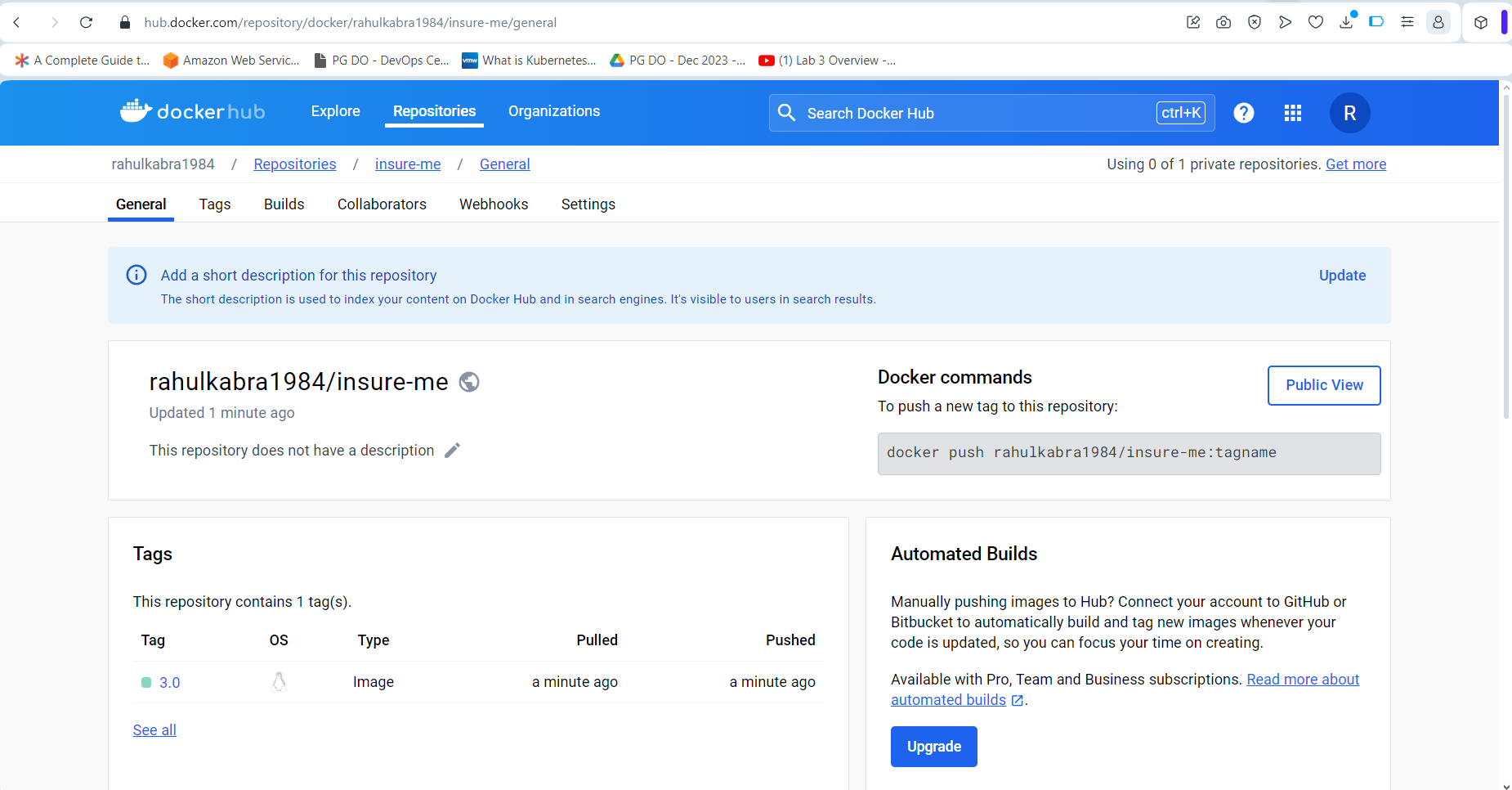
**Step 7 : Execute Jenkins Build and Validate it**

7.1 Navigate to Jenkins job created in above step and click on Build now to start running build for Jenkins job created.





7.2 Once Jenkins build is completed validate if Docker image really gets uploaded to Docker hub or not.



7.3 Now verify if the wesite is opening on the address **http://54.175.192.252:8081**.

