

Name: Jeevanantham Balakrishnan

CICD Implementation using Jenkins, Kubernetes, ArgoCD, SonarQube, Maven

Introduction:

In this project we are going to build and deploy to the Kubernetes cluster using

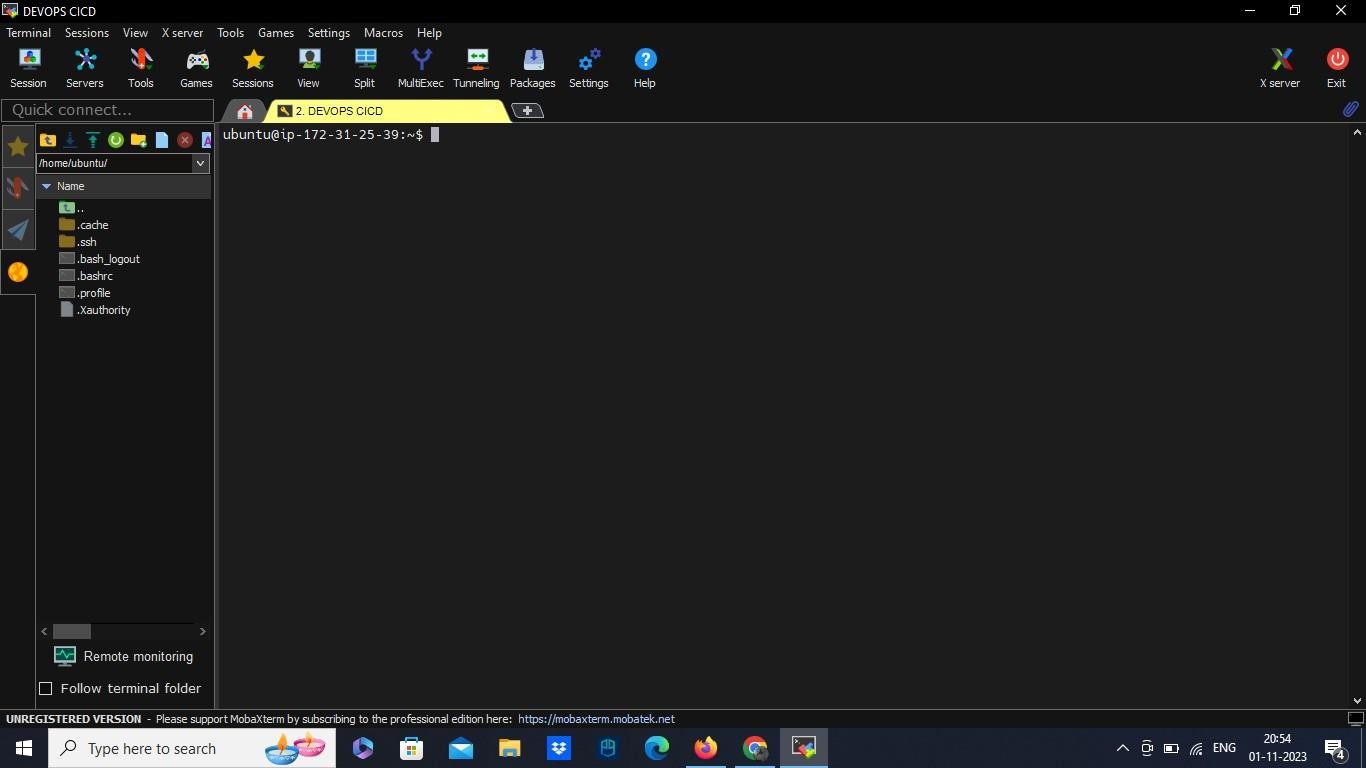
Jenkins, Kubernetes, ArgoCD. This process allows for automated and streamlined software development and deployment, ensuring that your applications are built, tested, and

deployed consistently and efficiently.

Steps Involved:

Step 1: AWS

* Launch an instance. Select the ami as Ubuntu.
* Select the instance type as large, to handle heavy work.
* Here Iam using Mobaxterm, to connect to my instance. Ensure that port is 22.



* Copy the public ip of your aws instance and launch it in your terminal using private key you have. Finally login as Ubuntu.

Step 2: Installation/Login

Here we need to install Jenkins for our server.

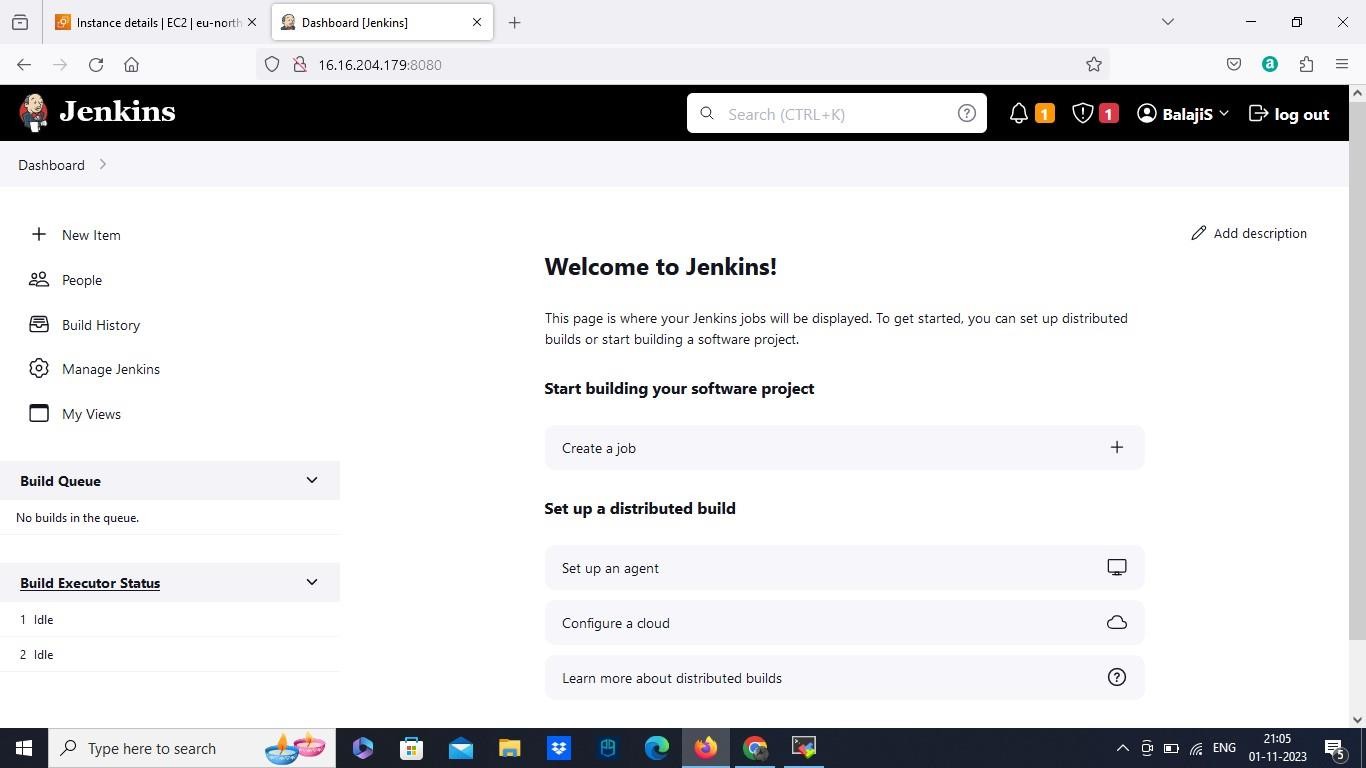
* Jenkins
  + sudo apt update
  + sudo apt install openjdk-11-jre
  + java -version
  + curl -fsSL https://pkg.jenkins.io/debian/jenkins.io-2023.key | sudo tee \

/usr/share/keyrings/jenkins-keyring.asc > /dev/null

echo deb [signed-by=/usr/share/keyrings/jenkins-keyring.asc] \ https://pkg.jenkins.io/debian binary/ | sudo tee \

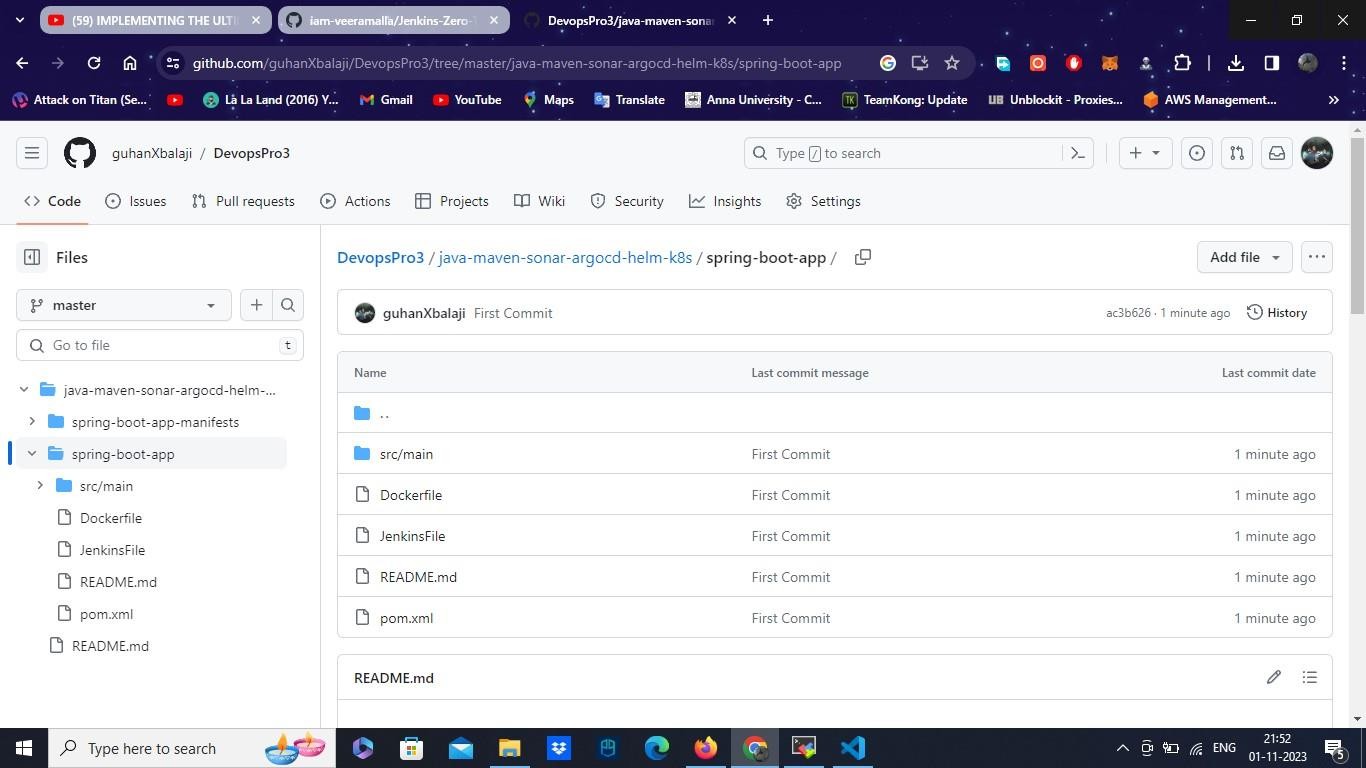
/etc/apt/sources.list.d/jenkins.list > /dev/null

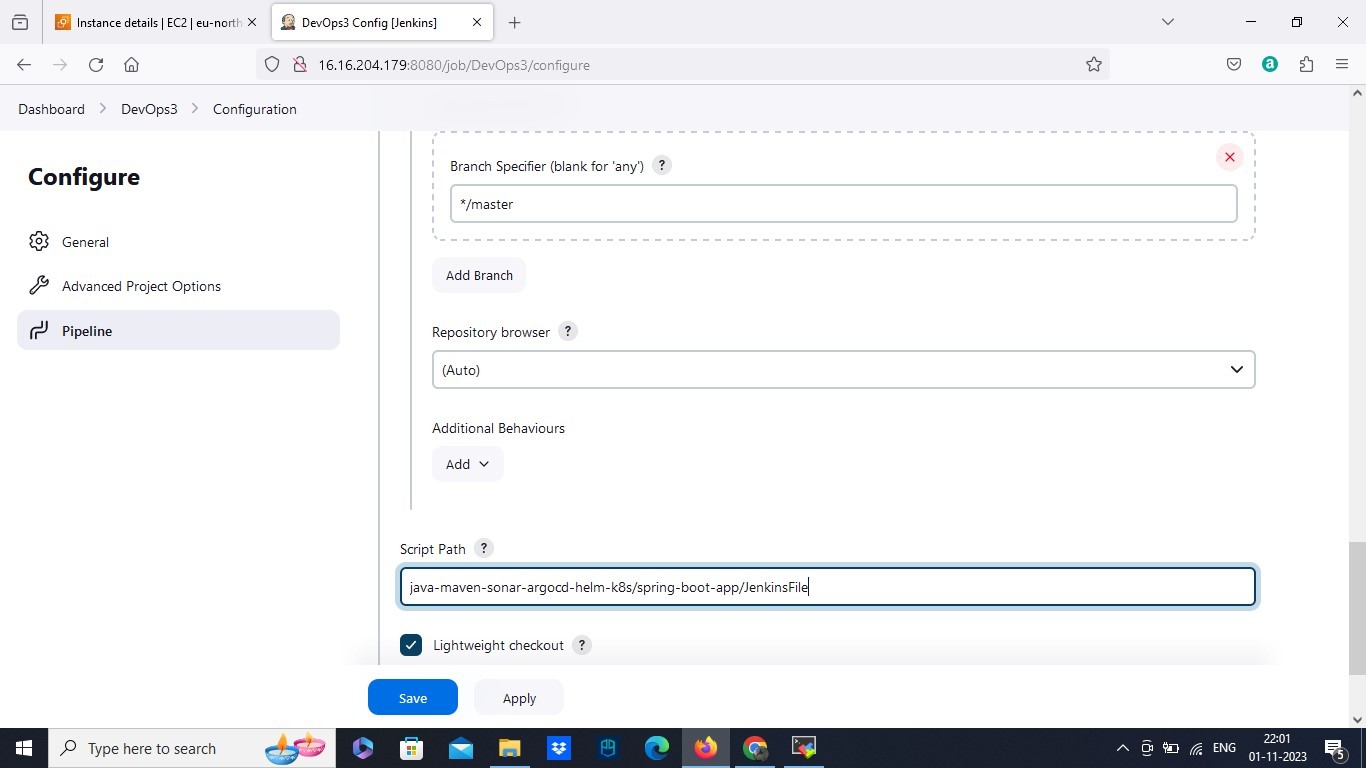
* + sudo apt-get update
  + sudo apt-get install Jenkins
  + For the login password - sudo cat /var/lib/jenkins/secrets/initialAdminPassword
* Copy the public IP and hit in the browser http:// :8080
* Now Login to the Jenkins with the user name and password. Also install necessary plugins.



Step 3: Create the pipeline:

* Enter a name to the pipeline
* Here we are going to use pipeline script from SCM. Where we are going to add predefined Jenkins file contains predefined script.
* Select scm as Git. Provide the git repository URL where our Application present.
* Select branch as master/main.
* Add the path to the jenkinsfile in our git repository. And save it.





Step 4: Plugins

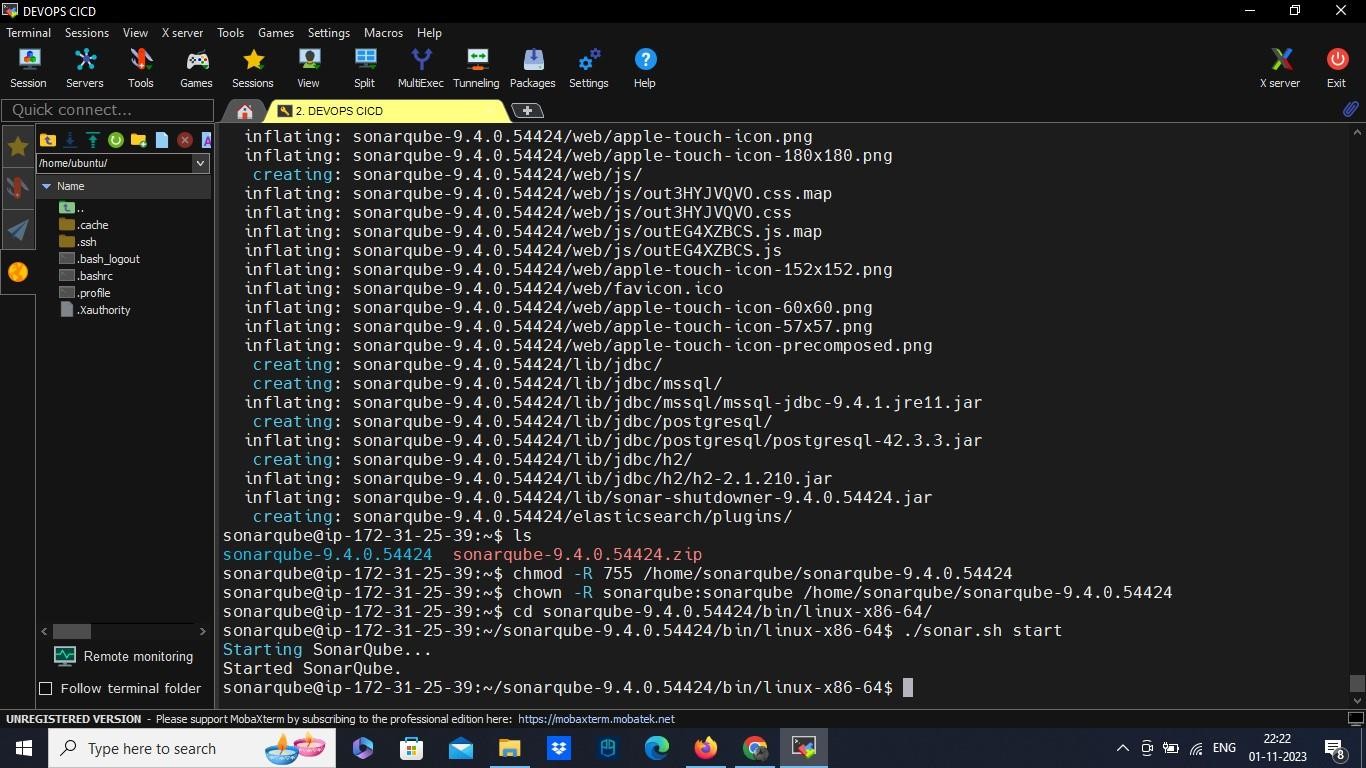
* Now go to manage plugins in your Jenkins.
* Locate available plugins, search docker pipeline plugin and install it.
* Install another plugin called sonar qube scanner in your instance. And restart the Jenkins.

Step 5: Install Sonarqube and login

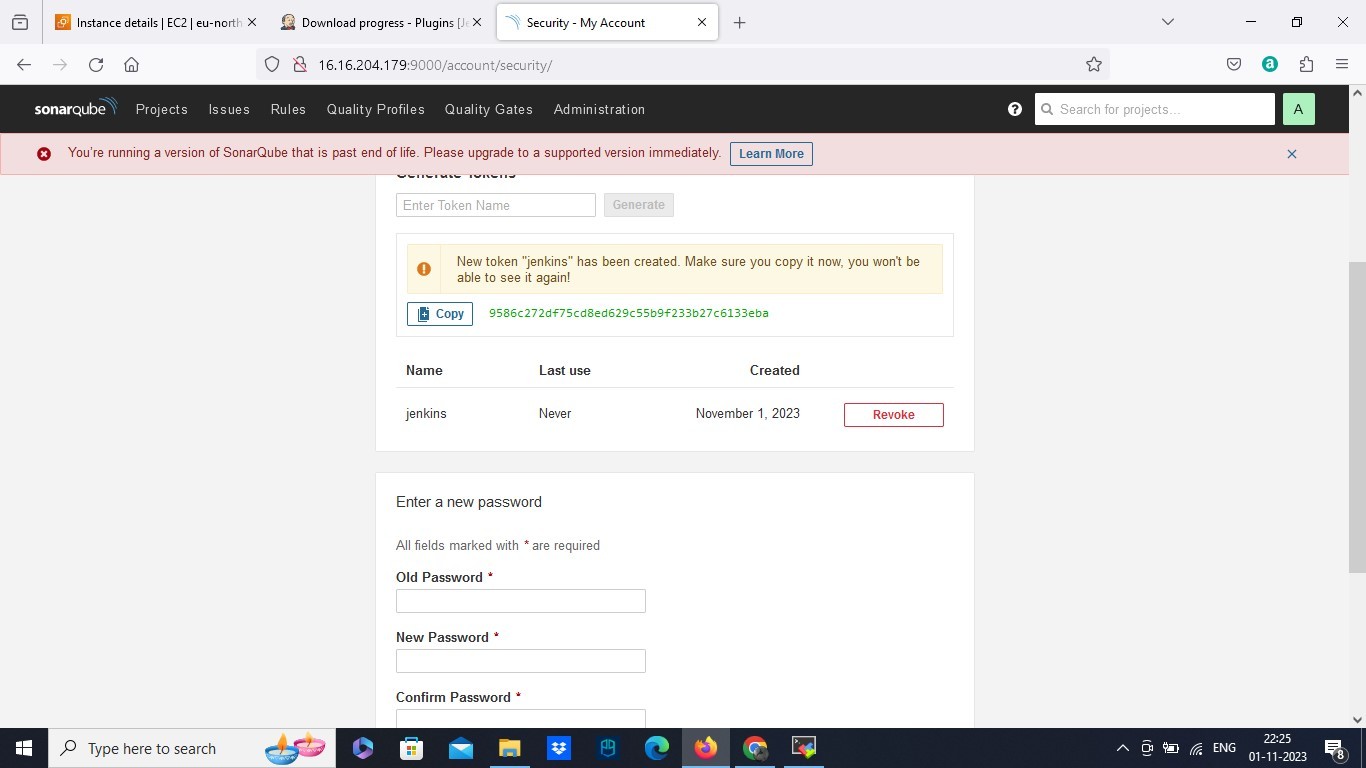
* Now go to the terminal of our instance and install sonarqube

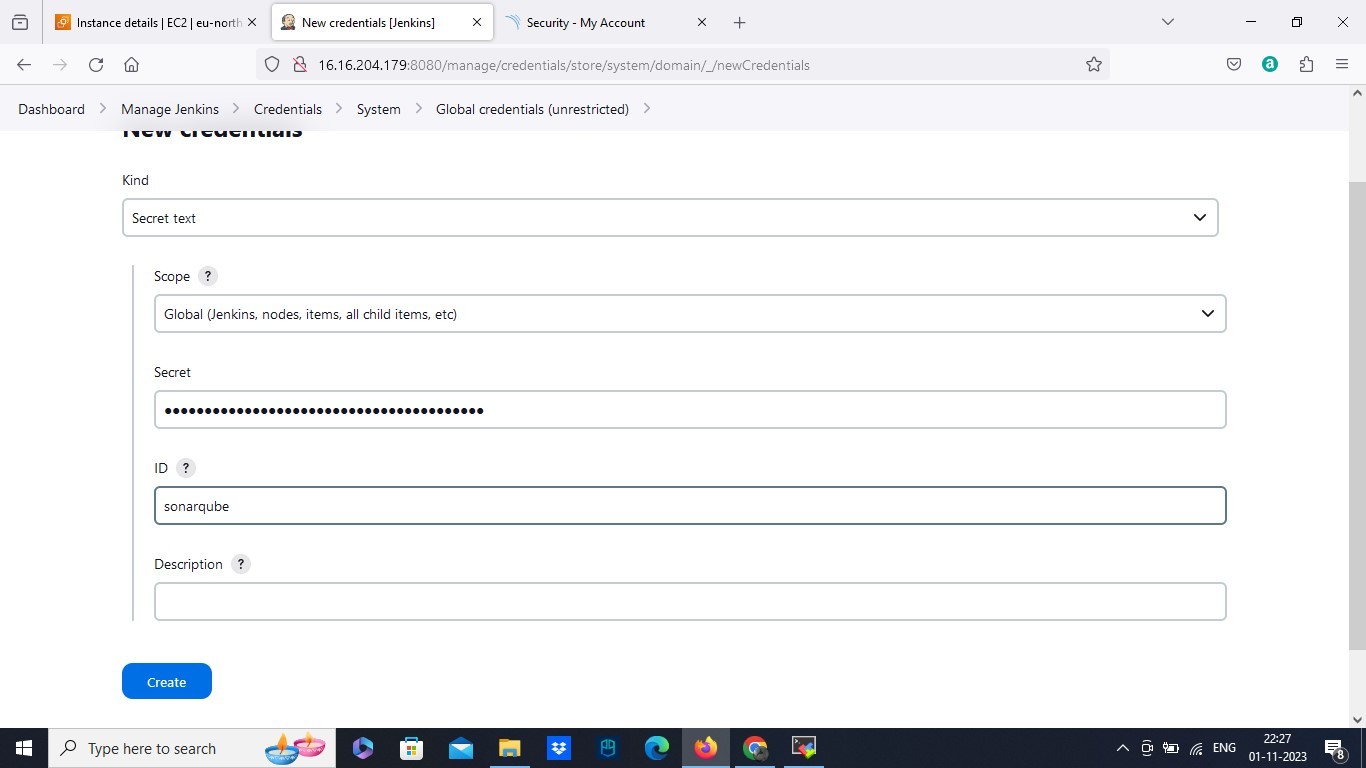
1. apt install unzip
2. adduser sonarqube
3. wget https://binaries.sonarsource.com/Distribution/sonarqube/sonarqube- 9.4.0.54424.zip
4. unzip \*
5. chmod -R 755 /home/sonarqube/sonarqube-9.4.0.54424
6. chown -R sonarqube:sonarqube /home/sonarqube/sonarqube-9.4.0.54424 7. cd sonarqube-9.4.0.54424/bin/linux-x86-64/

8. ./sonar.sh start



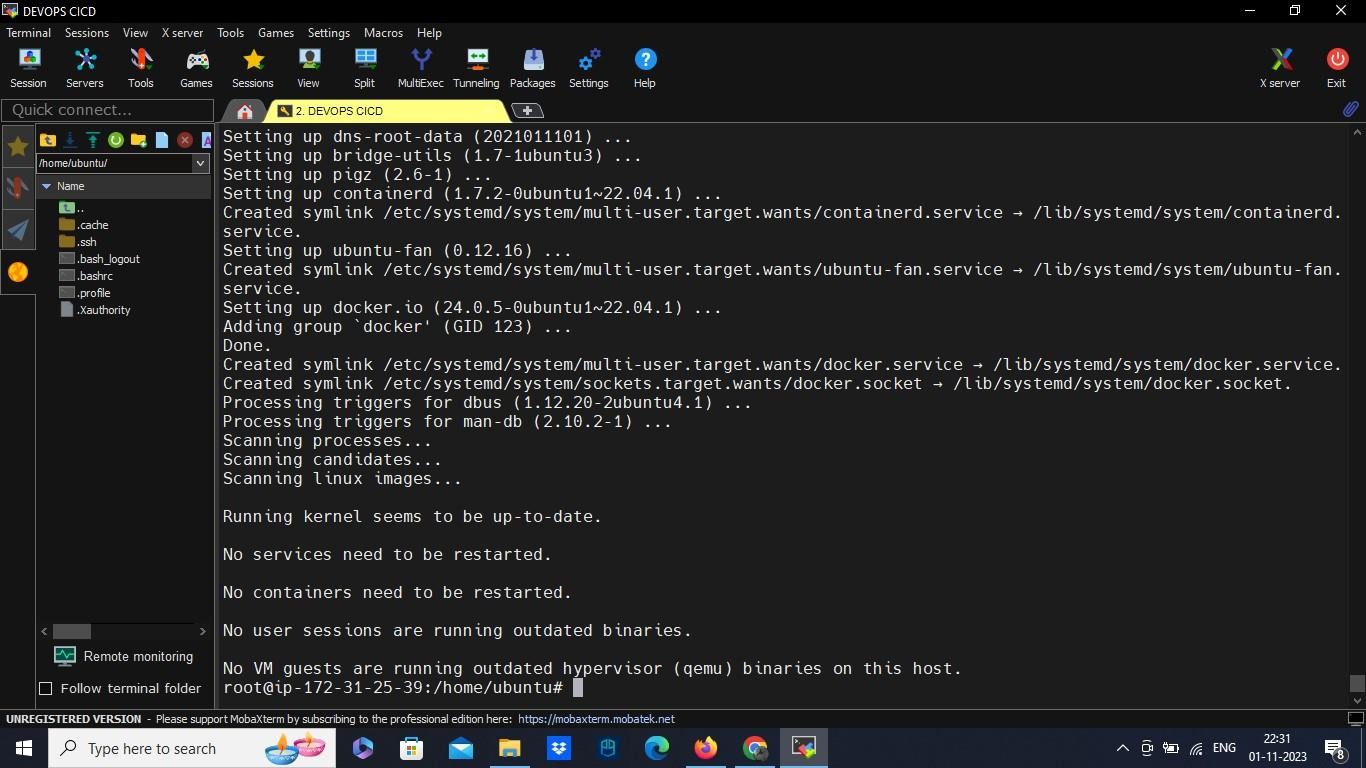
* Copy the public ip of our instance and hit the browser http:// :9000.
* Go to administration/ security. Where we can generate a sonarqube token for our Jenkins.
* Go back to Jenkins, Locate credentials and add new credentials as secret text.
* Paste the token from sonarqube in secret field and provide the name.





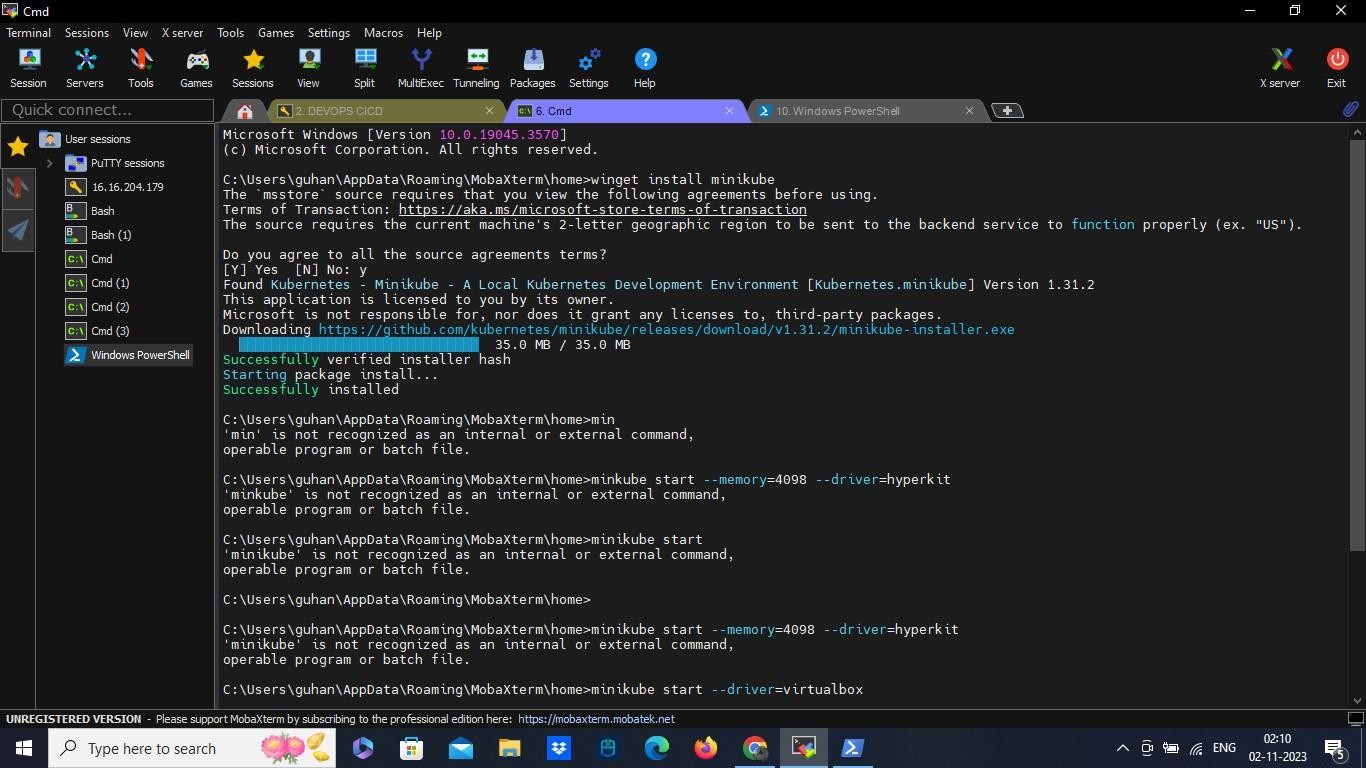
Step 6: Docker

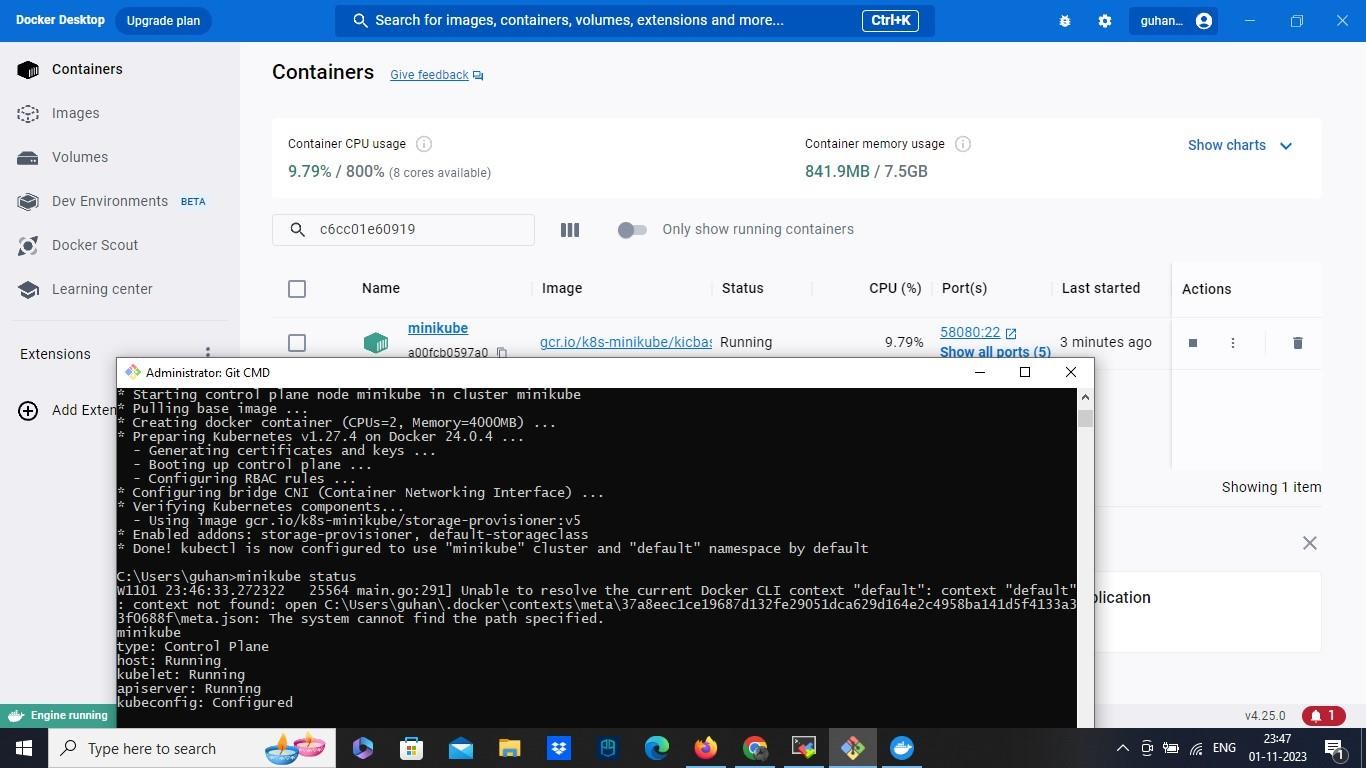
* Install Docker in our Instance and granting access which contains Jenkins.
  + sudo apt update
  + sudo apt install docker.io
  + sudo su -
  + usermod -aG docker jenkins
  + usermod -aG docker ubuntu
  + systemctl restart docker



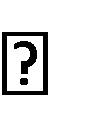
Step 7: Install Minikube

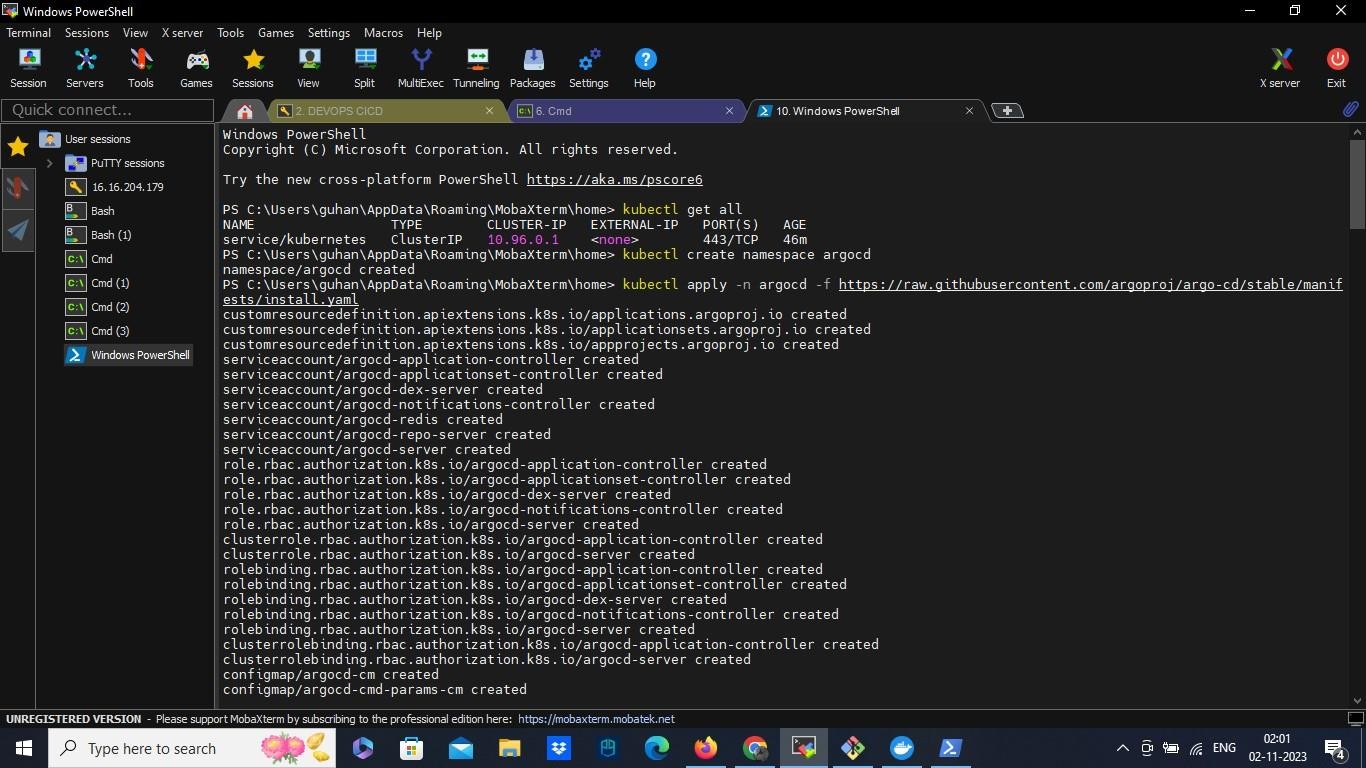
* Now we have to install minikube in our computer.
* Iam Using Mobaxterm cmd to install the minikube in my computer.
  + winget install minikube
* Open Docker desktop and run docker engine.
* Use other cmd prompt starting minikube cluster ( Iam using Git Cmd here)
  + minikube start





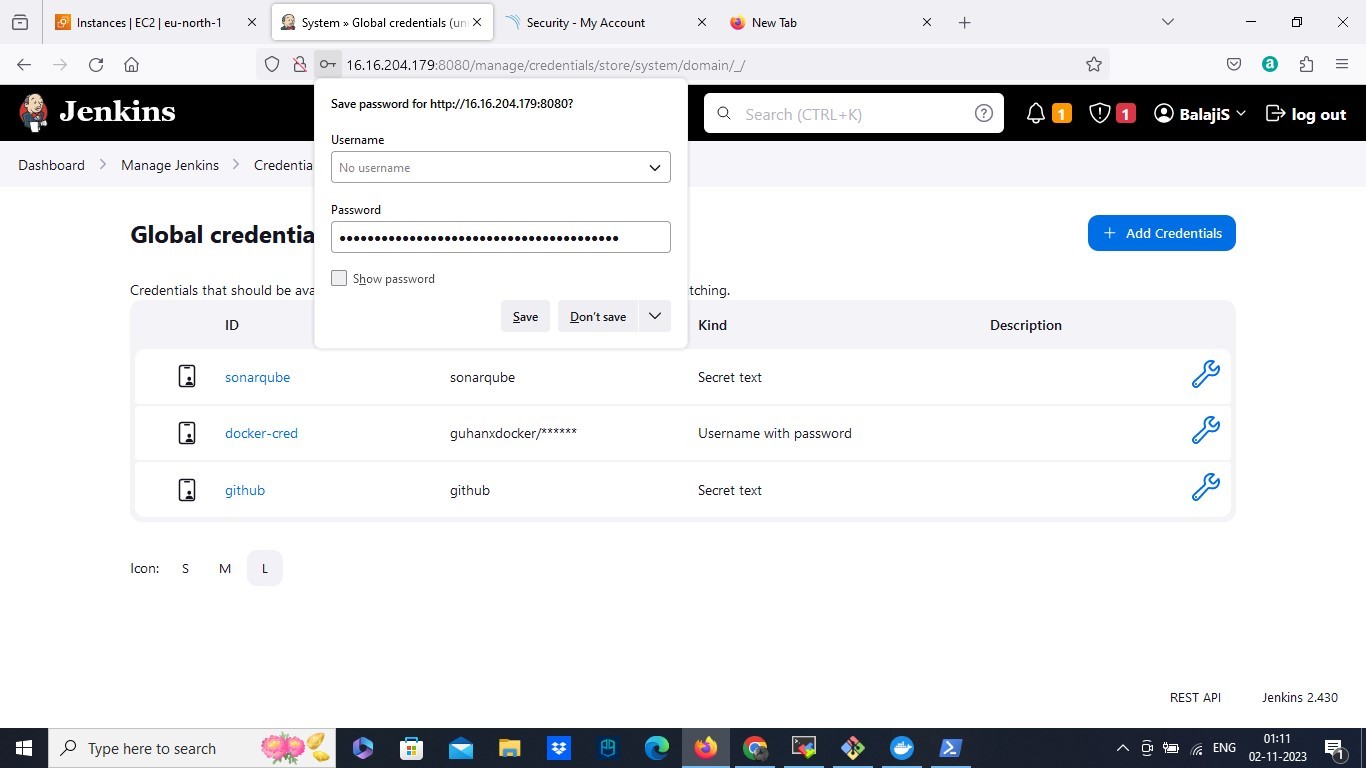
Step 8: Argo CD

* Install Argo cd by using powershell in your computer ( here iam using mobaxterm powershell). Also create namespace as argocd. 
  + kubectl create namespace argocd
  + kubectl apply -n argocd -f [https://raw.githubusercontent.com/argoproj/argo-](https://raw.githubusercontent.com/argoproj/argo-cd/stable/manifests/install.yaml) [cd/stable/manifests/install.yaml](https://raw.githubusercontent.com/argoproj/argo-cd/stable/manifests/install.yaml)



Step 9: Credentials

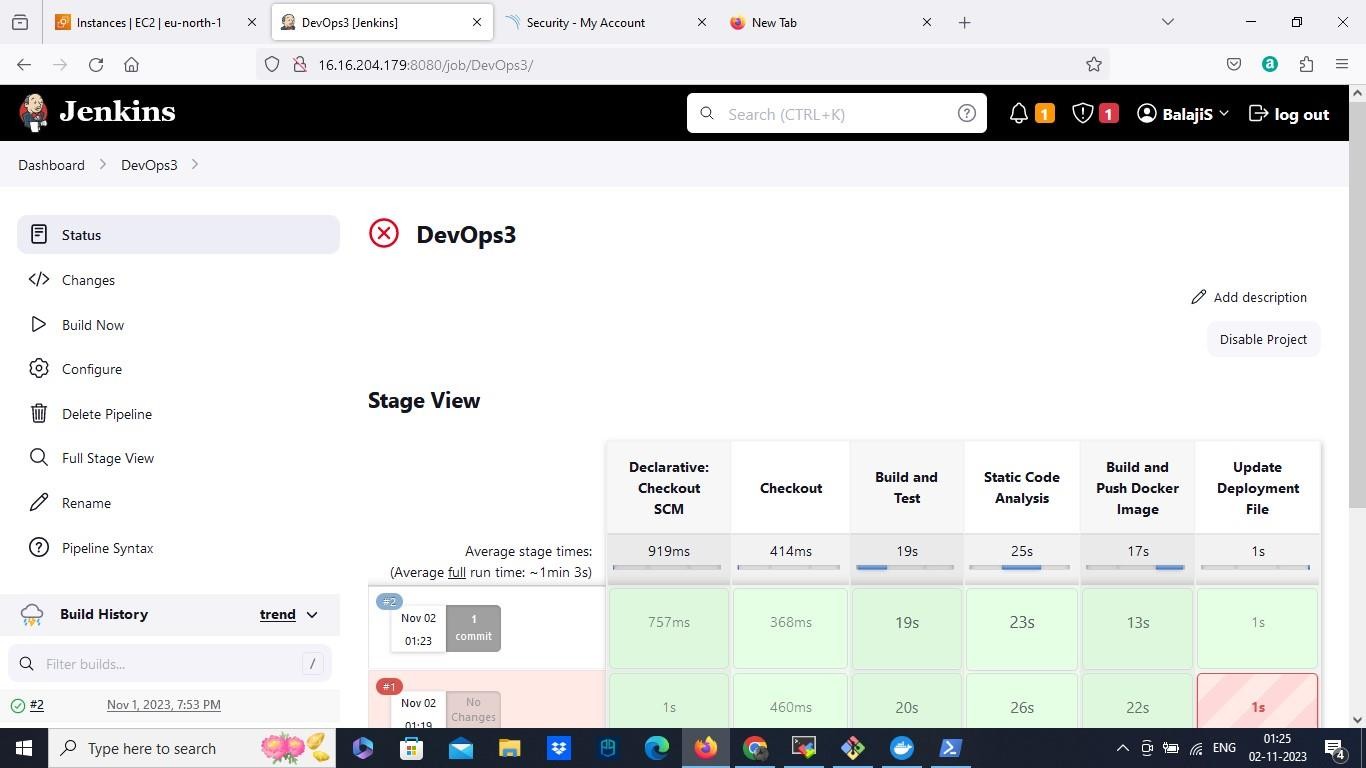
* We have to add credentials for tools to get accessed from the Jenkins file
* Go to manage credentials in Jenkins, add credentials for docker hub.
* Enter the username and password of docker hub. And add the name in the id, the name you provided in Jenkins file.
* Next one for Git Hub credentials. For that choose secret text.

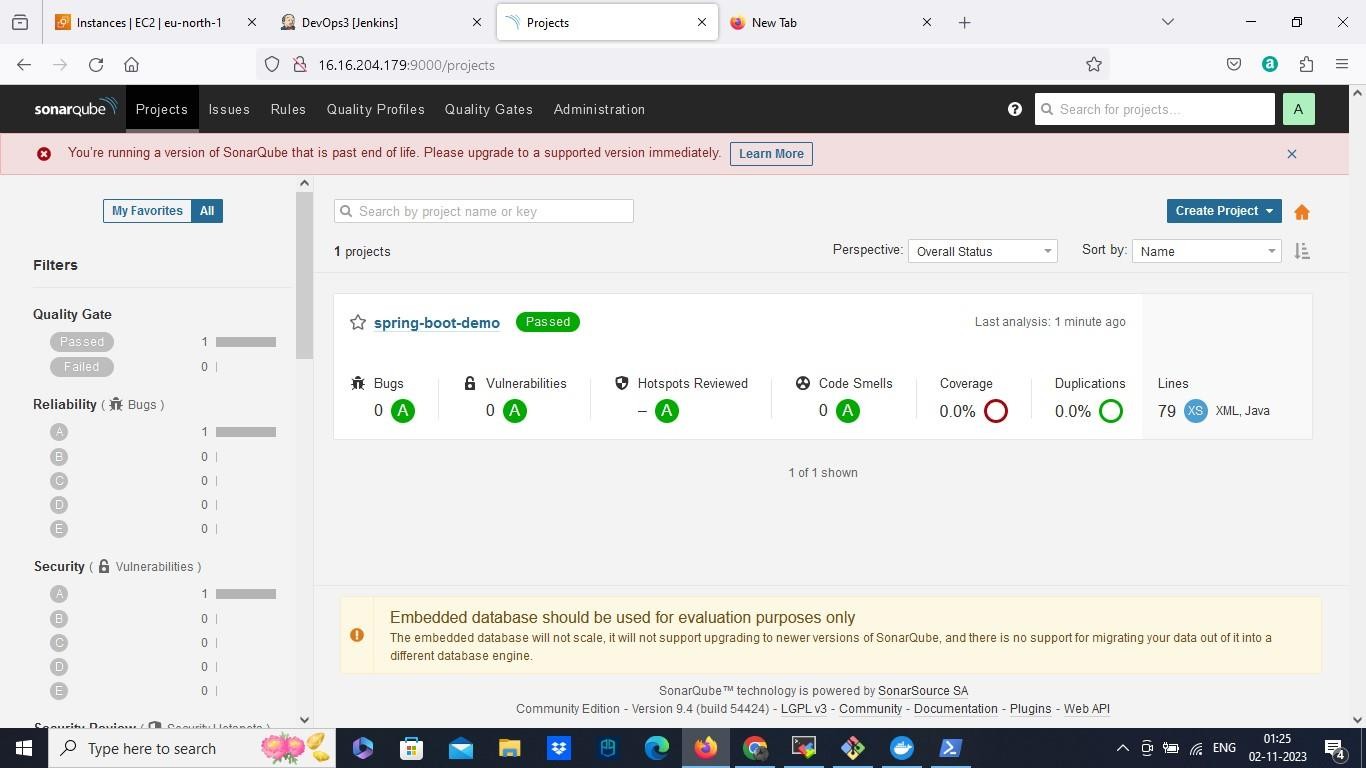


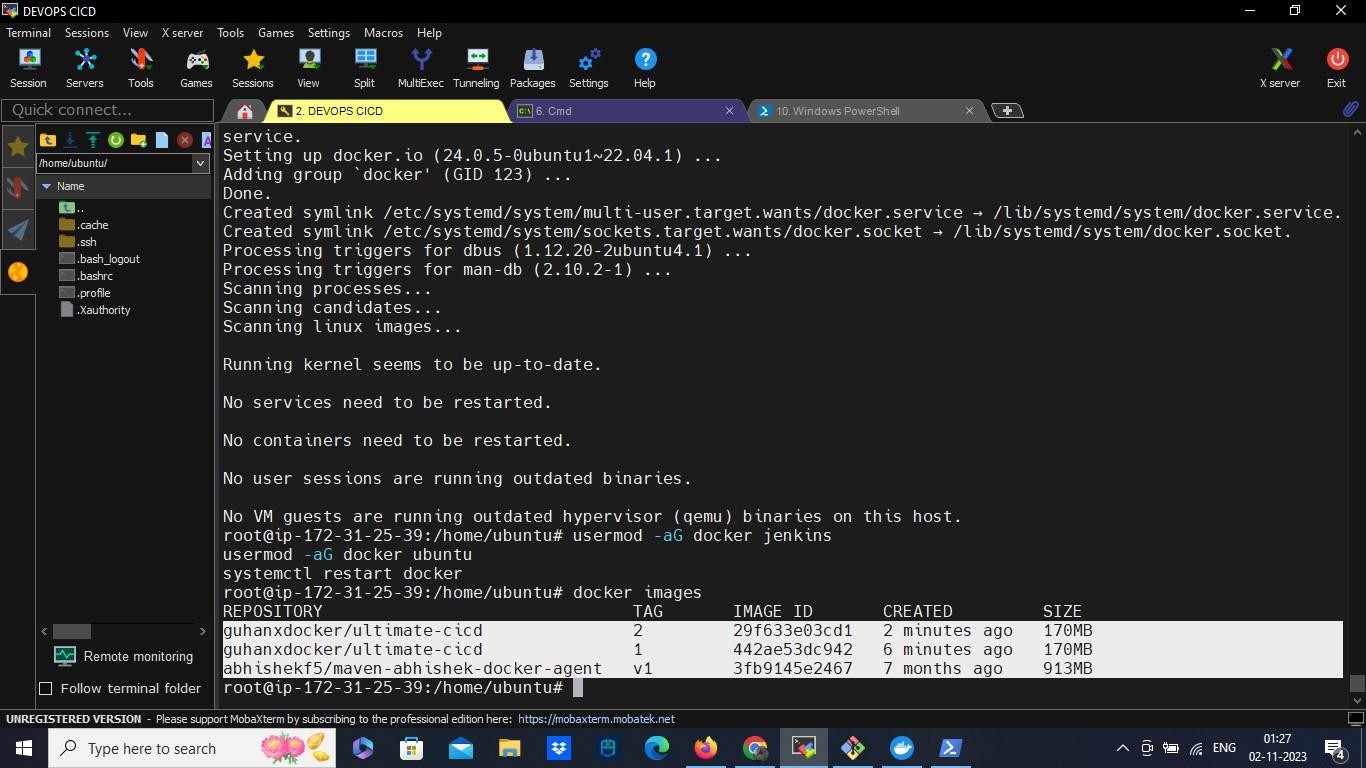
* Add the git hub token you generated in the secret field and give the id name, the name you provided in script.
* Save it and restart it.

Step 10: Build

* Before start the build, we need to do a minor change.
* Go to our application git repository, locate the Jenkins file and add our sonarqube URL in that file.
* Commit and save it.
* Now go back to Jenkins and build now. And will get built.

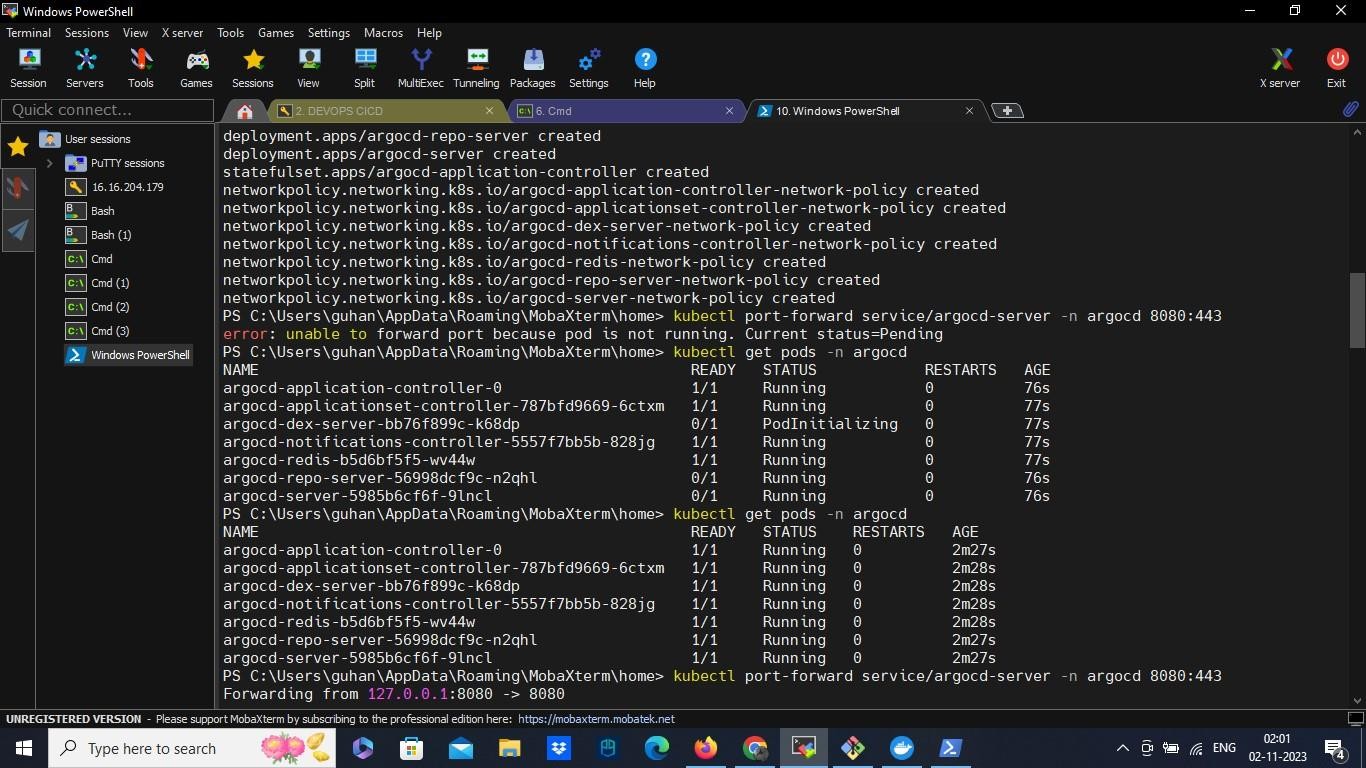


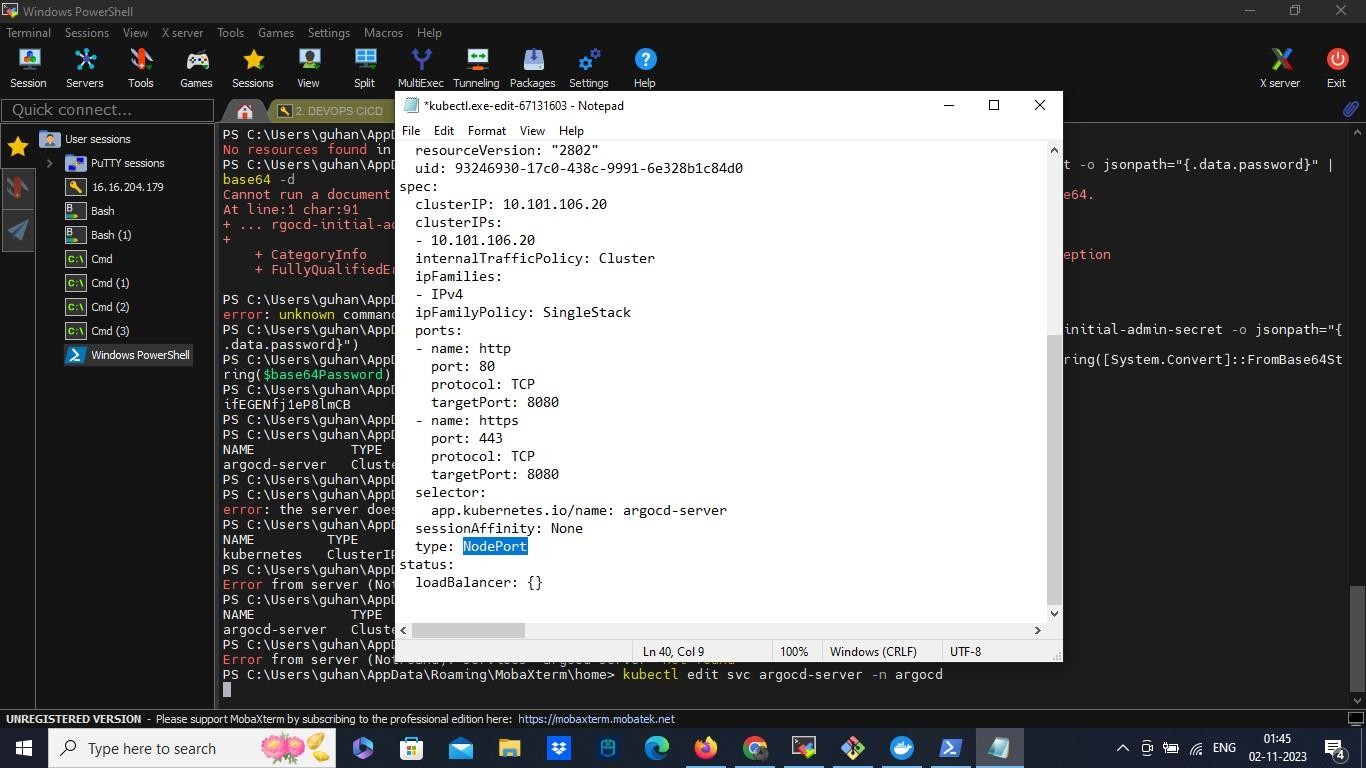




Step 11: ArgoCD

* First we need to change the argocd to run in our browser.
* To do that use kubectl get svc and edit the argocd server with kubectl edit svc name.
* It get opened with notepad, change the type from cluster ip to node port. Save it.
* Now use minikube service name and minikube service list, You can get URL for argo CD now. And also check the pods using kubectl get pods.





* Now copy the URL and paste in the browser .
* To login the Argocd use username as admin and to know the password use

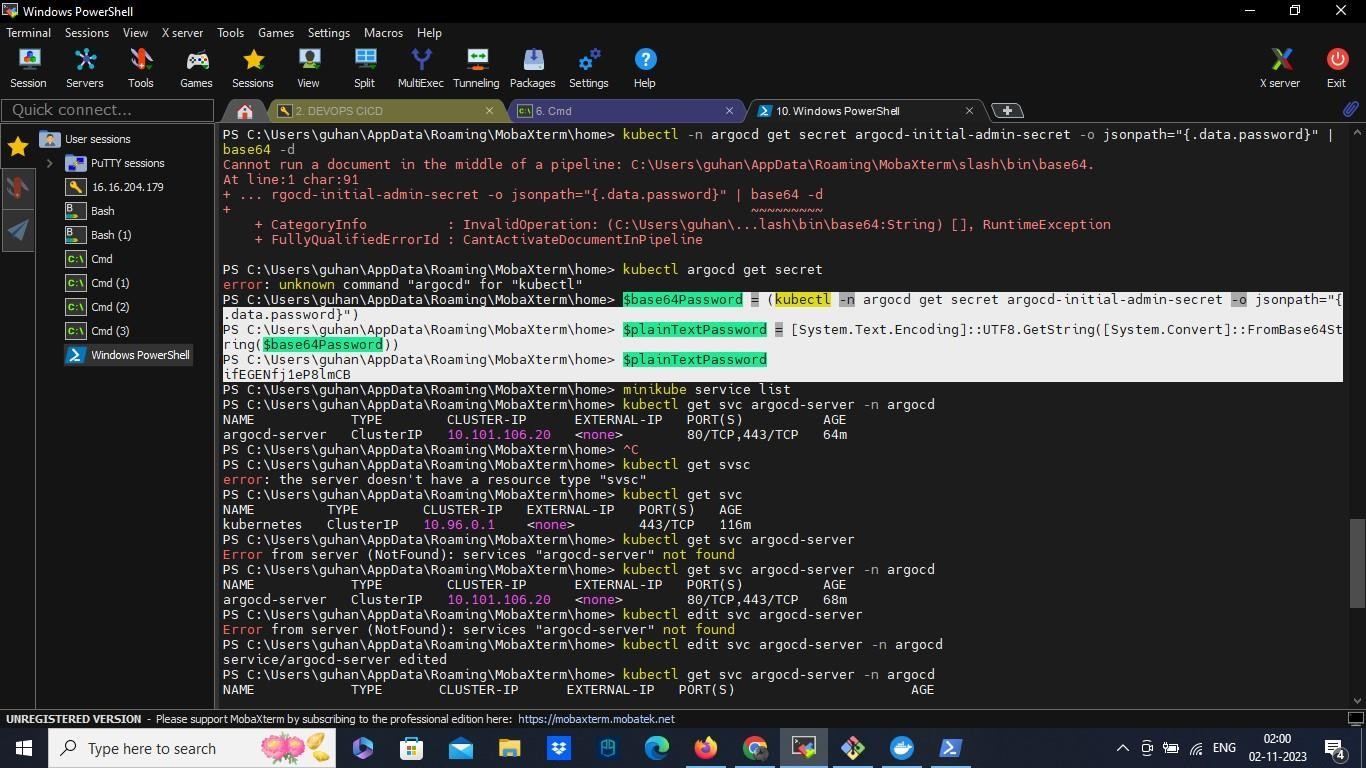
$base64Password = (kubectl -n argocd get secret argocd-initial-admin-secret -o jsonpath="{.data.password}")

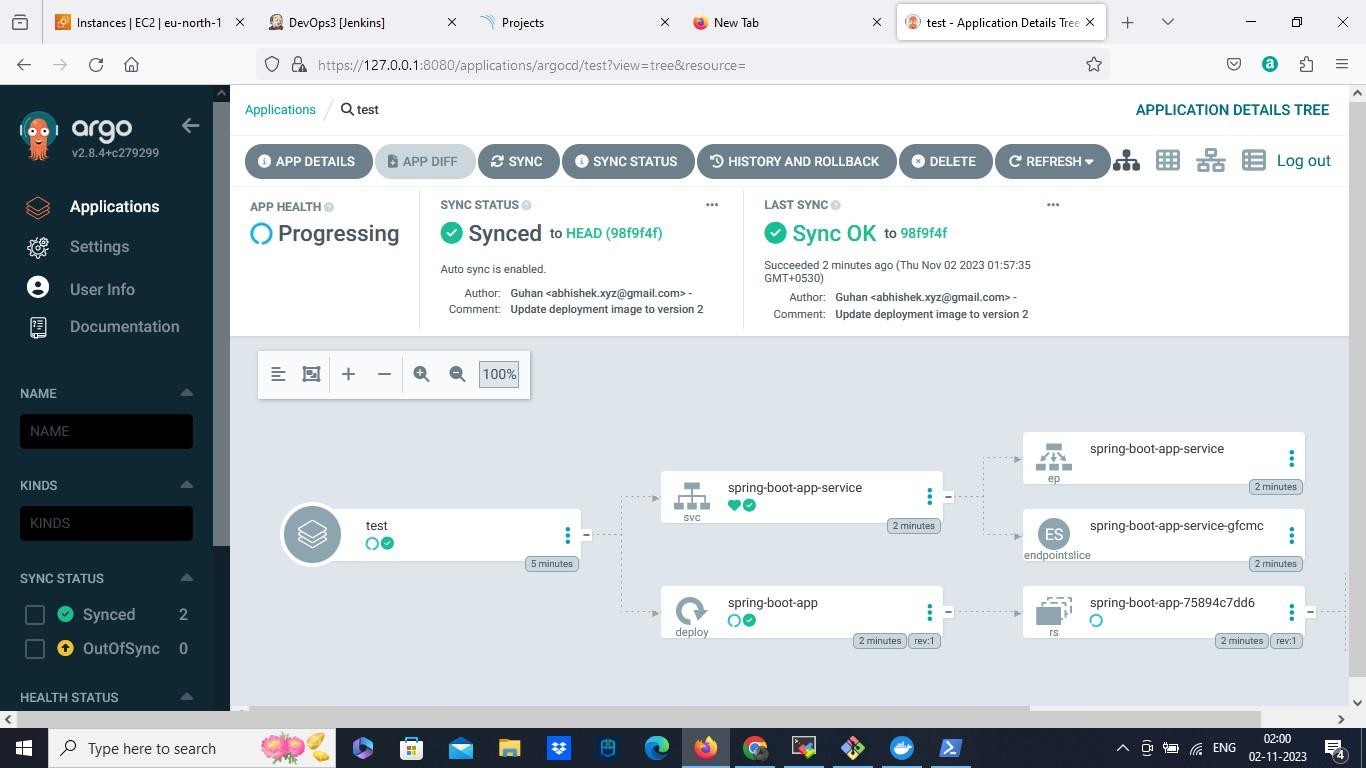
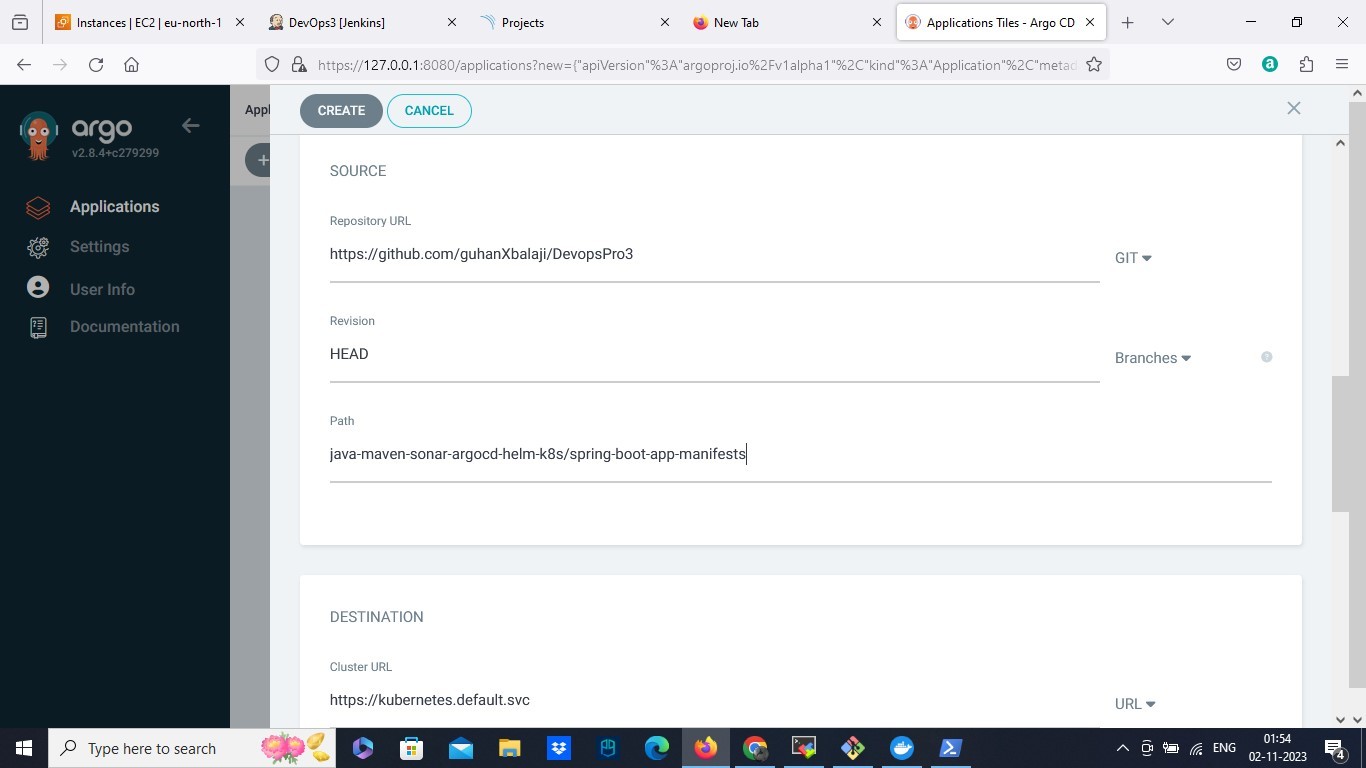
$plainTextPassword =

[System.Text.Encoding]::UTF8.GetString([System.Convert]::FromBase64String($base64Pa ssword))

$plainTextPassword

* Copy the password from the powershell and login to the Argocd



* After login Argocd. Select create application.
* Provide the application name, add default as project name. make sync policy automatic.
* Add the application git hub repo URL, add the path of yaml file from the repository.
* Select the cluster url and add the namespace as default. And it will get created. You can also check with kubectl get pods.

In summary, our implementation of CI/CD with Jenkins, Maven, SonarQube, Docker, Kubernetes, and ArgoCD has had a profound impact on our project. This is how the CICD implementation works. It has made our development and deployment processes more

efficient, elevated the quality of our applications, and significantly improved their reliability.