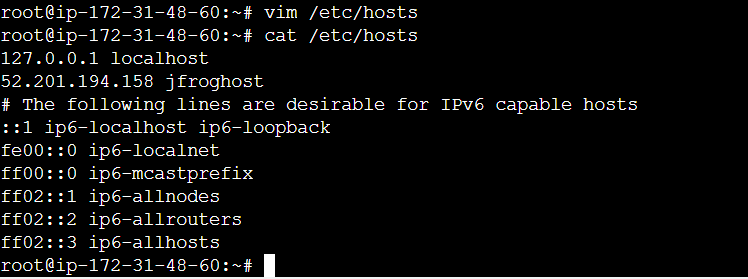
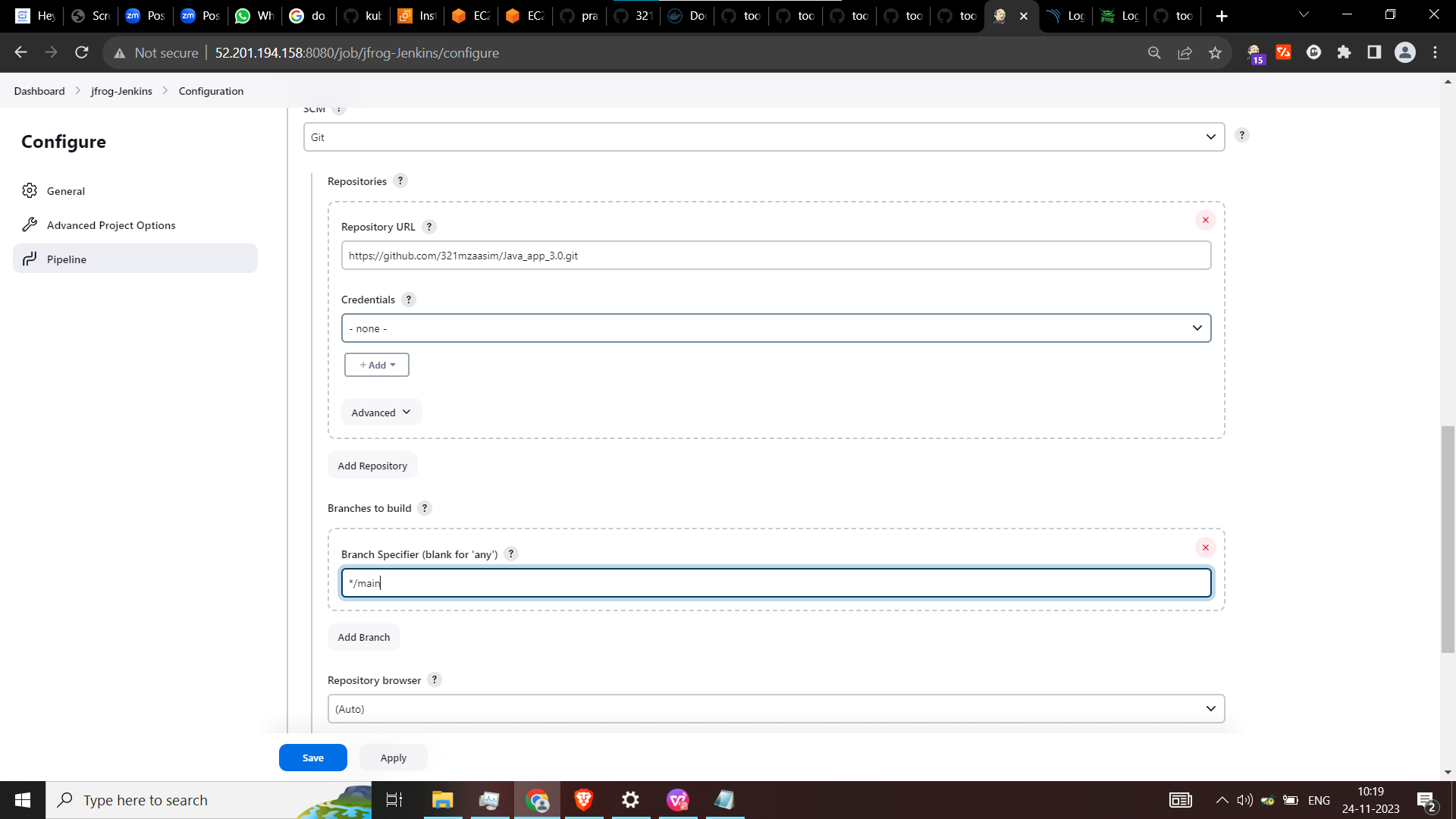
Add server ip as jfroghost in /etc/hosts

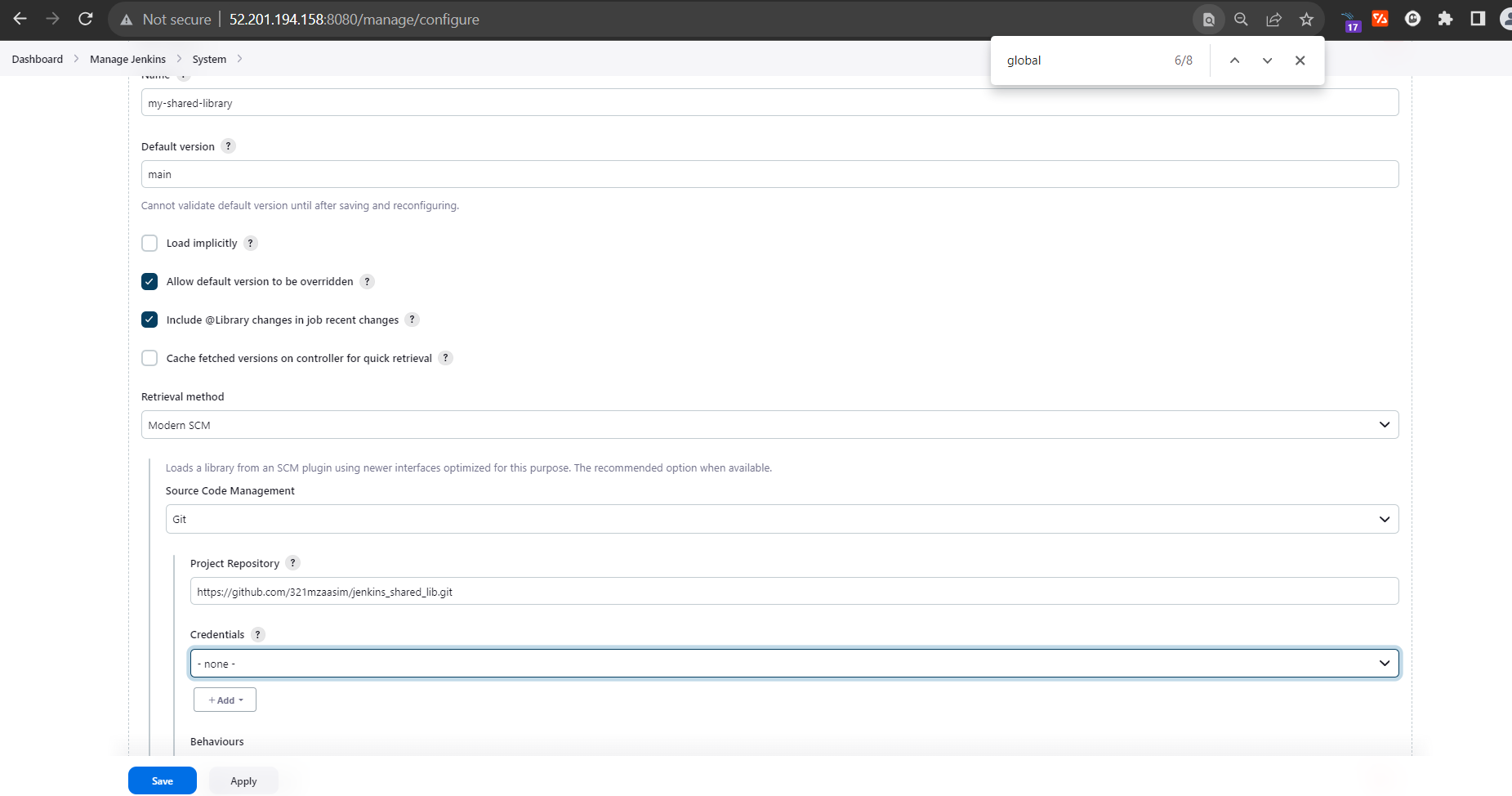


Create a pipeline using your github repository URL - Bcs we need to modify the jenkinsfile

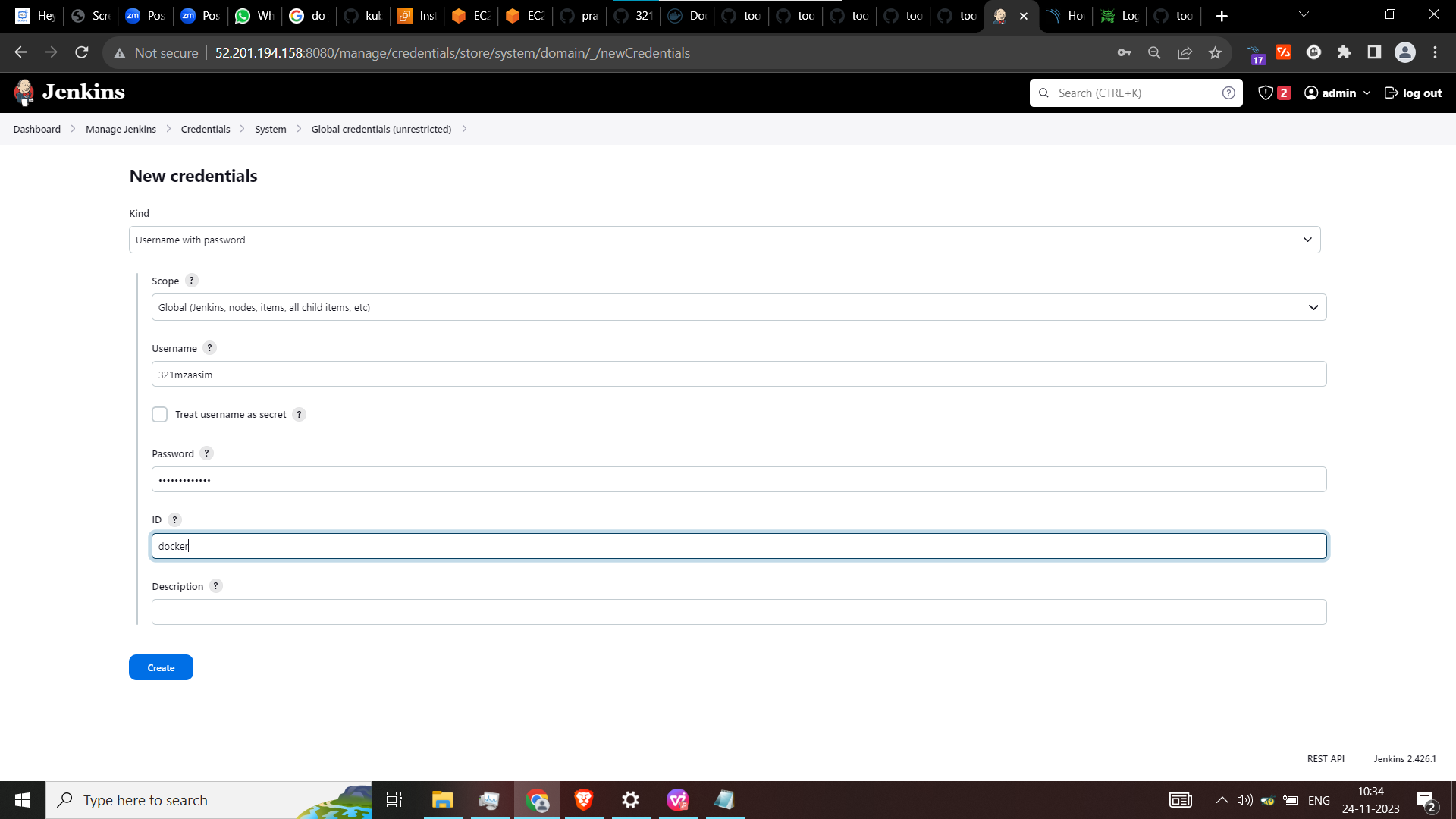


**Global Pipeline Libraries - give your github shared library - bcs we need to modify this also**

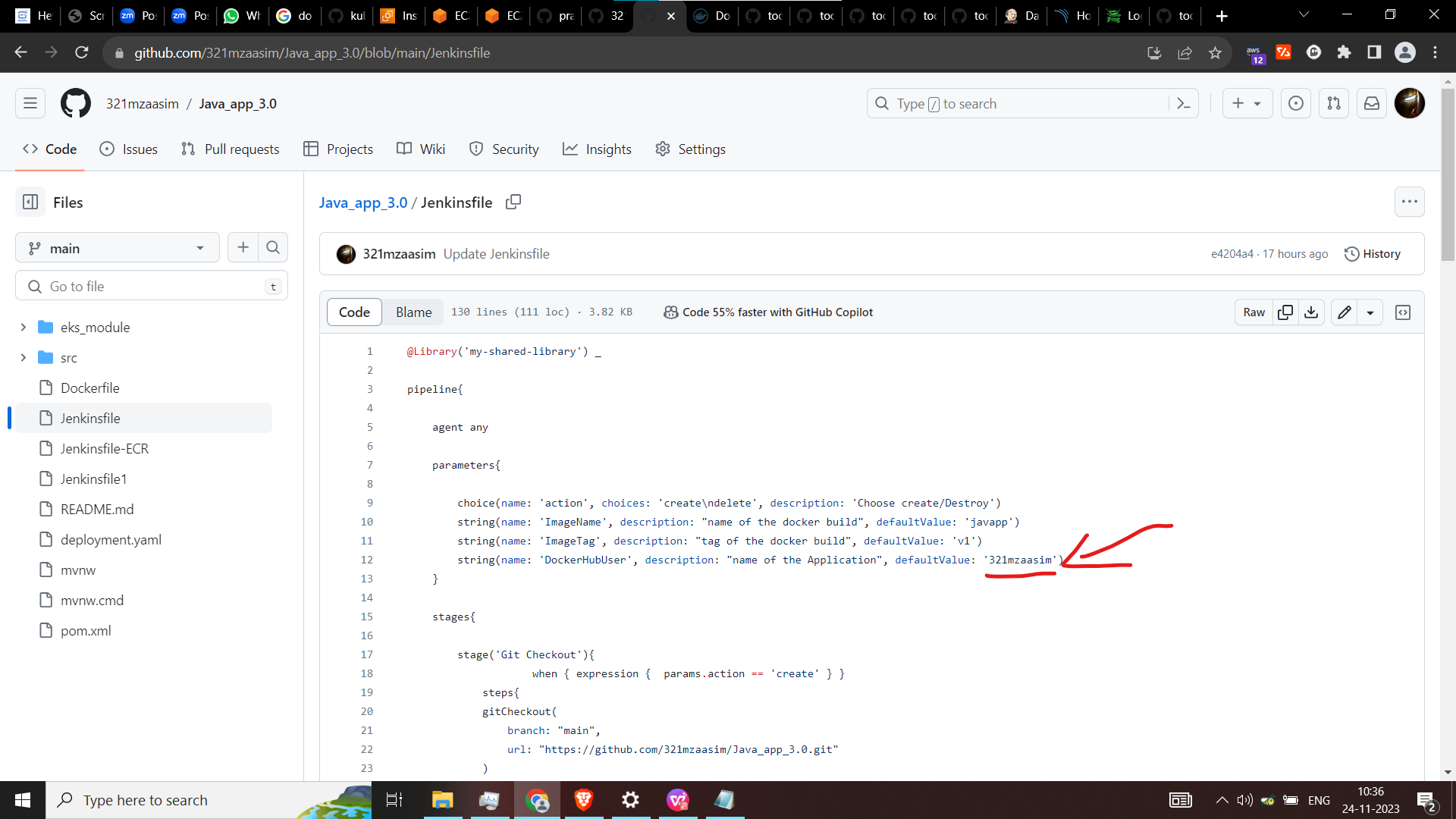
**///////// Note -Just copy the both github to your repo /////////////**



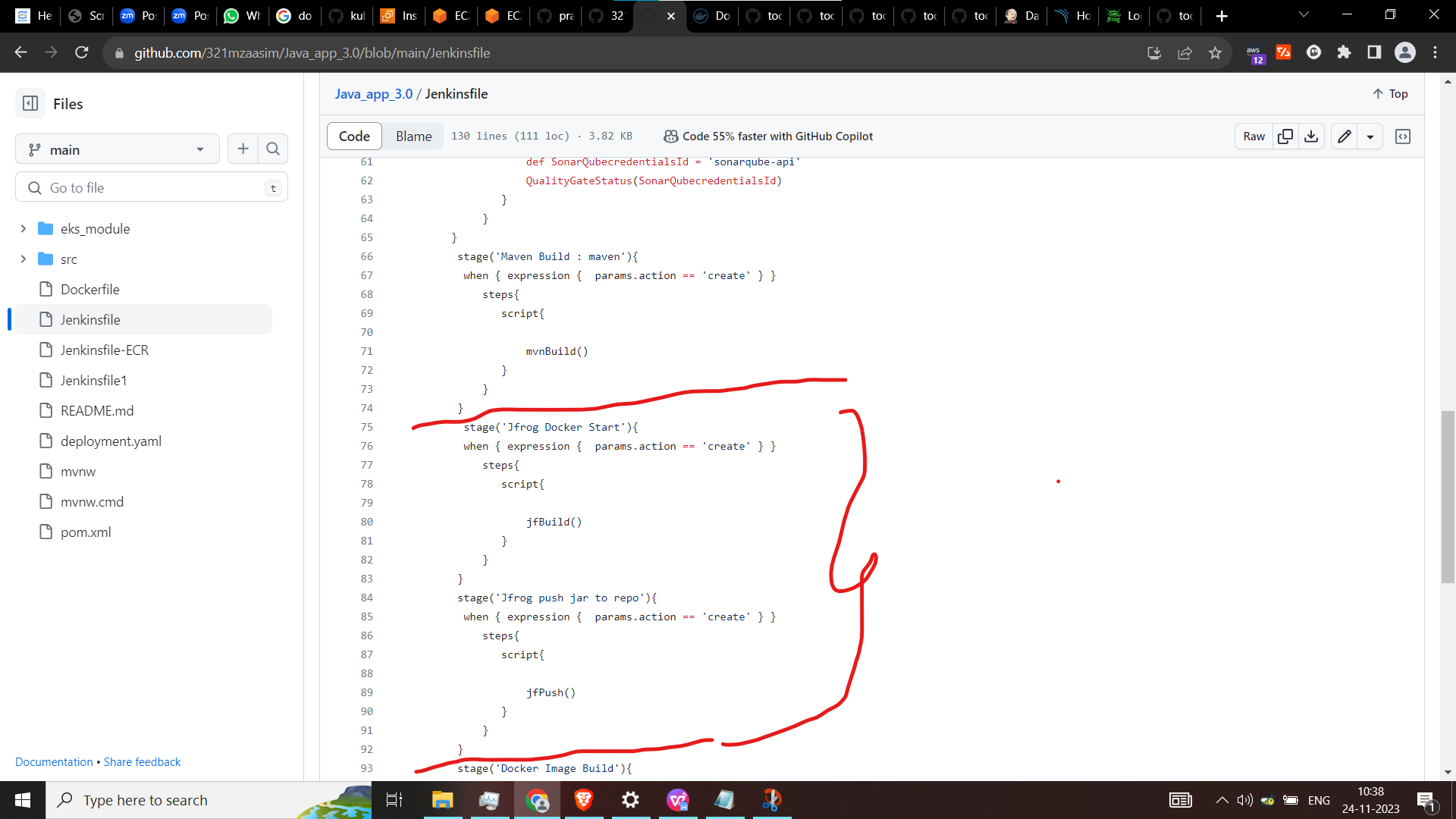
Give your docker hub credentials



Now we are going to modify the **Jenkinsfile in github**



Add your dockerhub username here what you have given for credentials



This where the extra line we have added after the maven build, as you can see I have used the same code of maven build function - just changed the **stage name and used jfBuild() and jfPush() function**

**-------------------------------------------------------------------------------**

**Code:**

stage('Jfrog Docker Start'){

when { expression { params.action == 'create' } }

steps{

script{

jfBuild()

}

}

}

stage('Jfrog push jar to repo'){

when { expression { params.action == 'create' } }

steps{

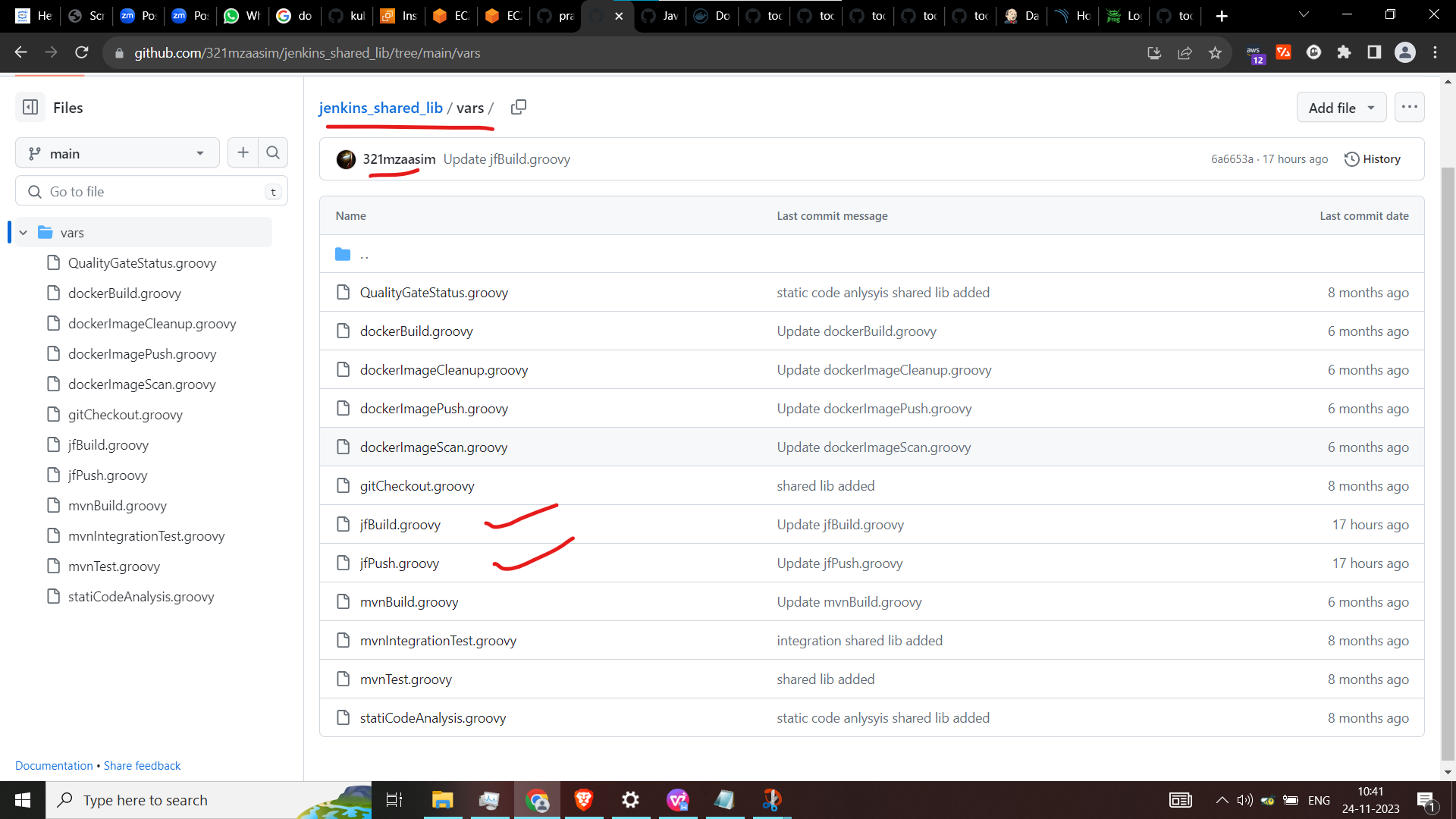
script{

jfPush()

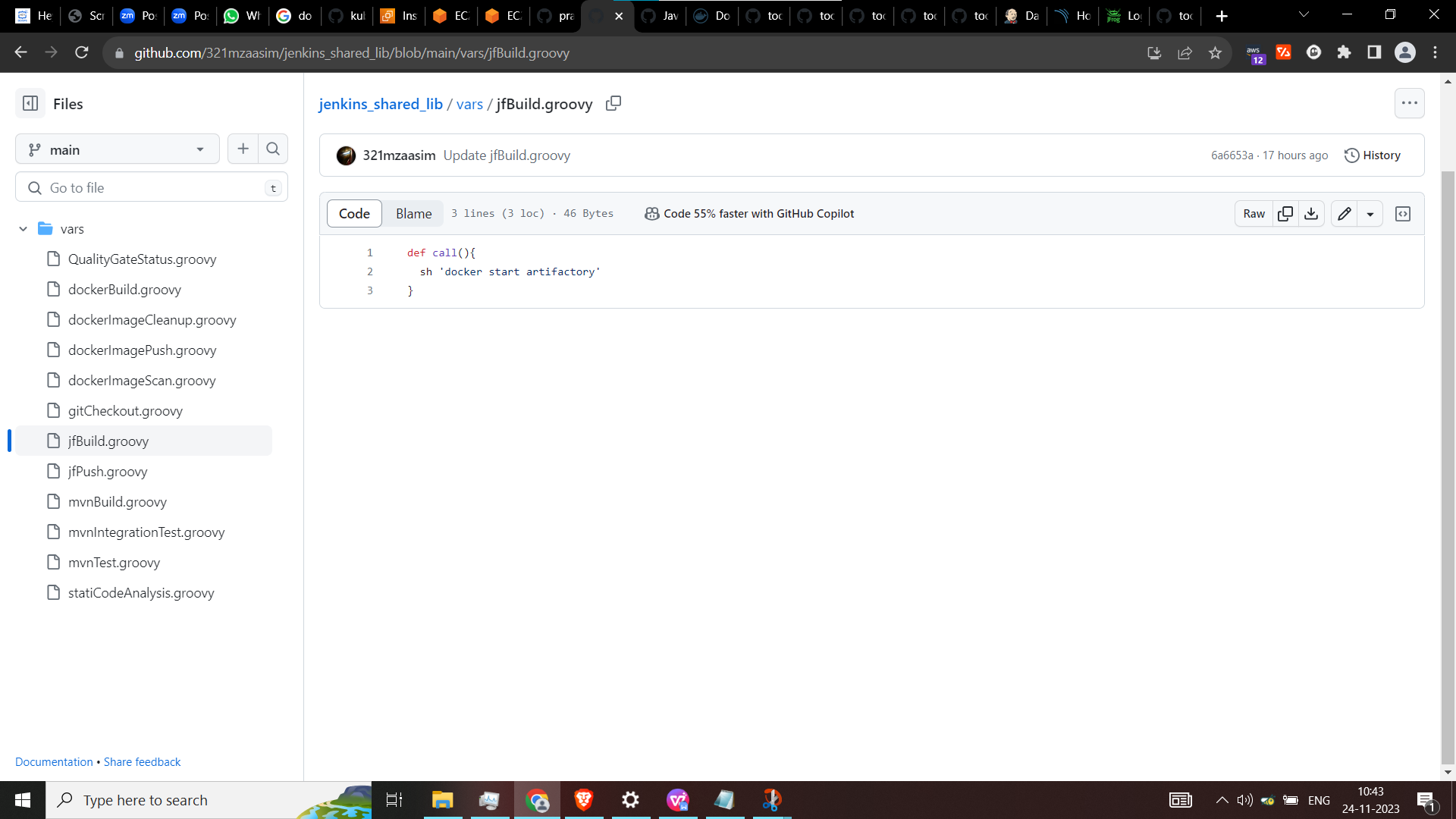
}

}

}

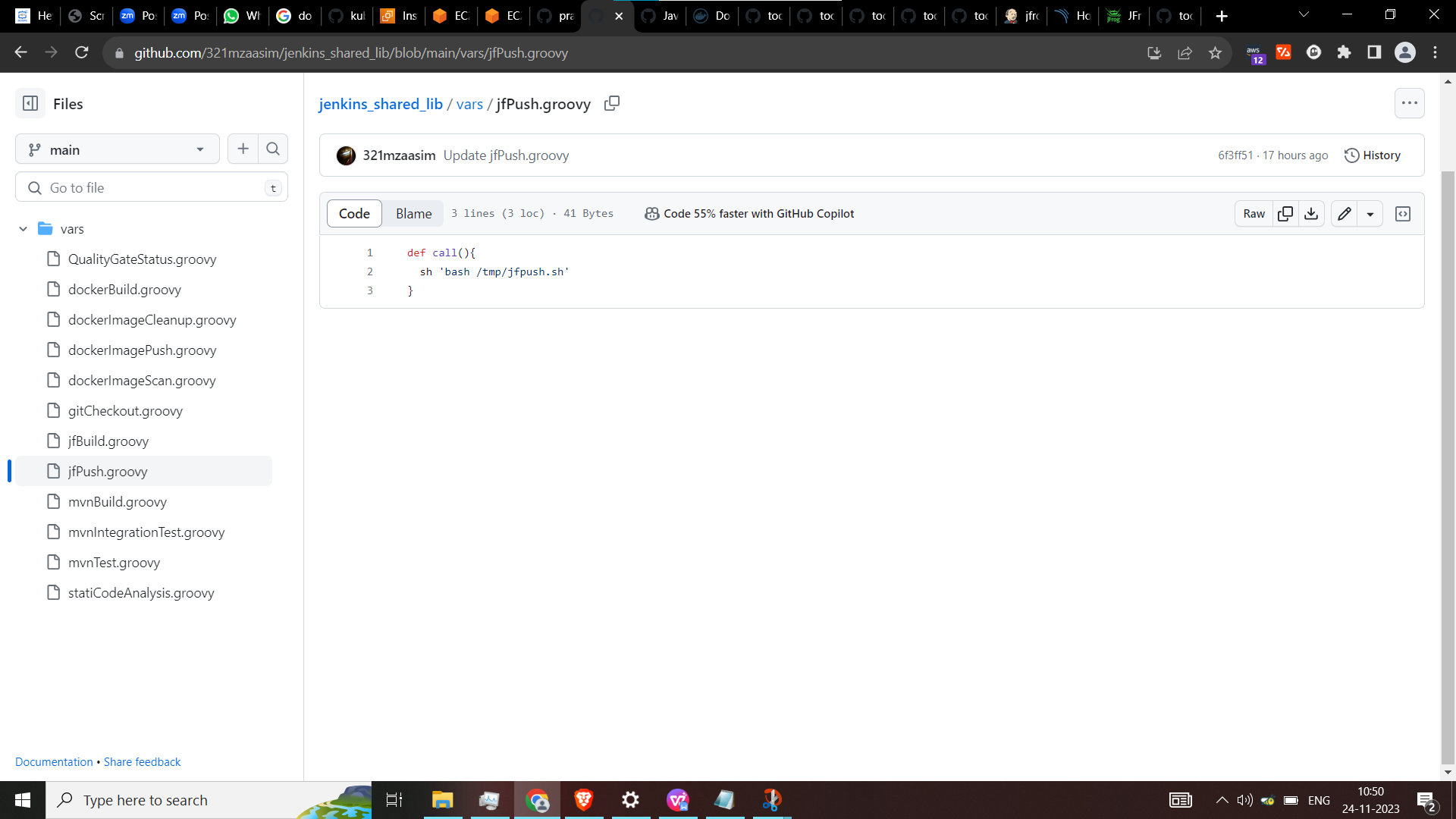


Whatever you have given the function name create same groovy script in the shared folder of yours



In jbuild script - I am just starting the docker of jfrog

docker start artifactory



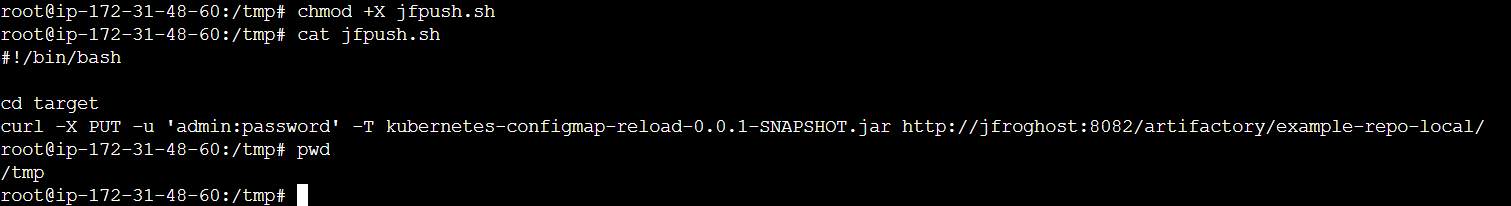
After the maven build it will create a target folder inside jar file is present.

So, move to target folder and execute the command

In jfpush script I have used bash file to run.

I can give the same command in this script itself but it has username and password.

So, I have used the script to run it.



**Code:**

#!/bin/bash

cd target

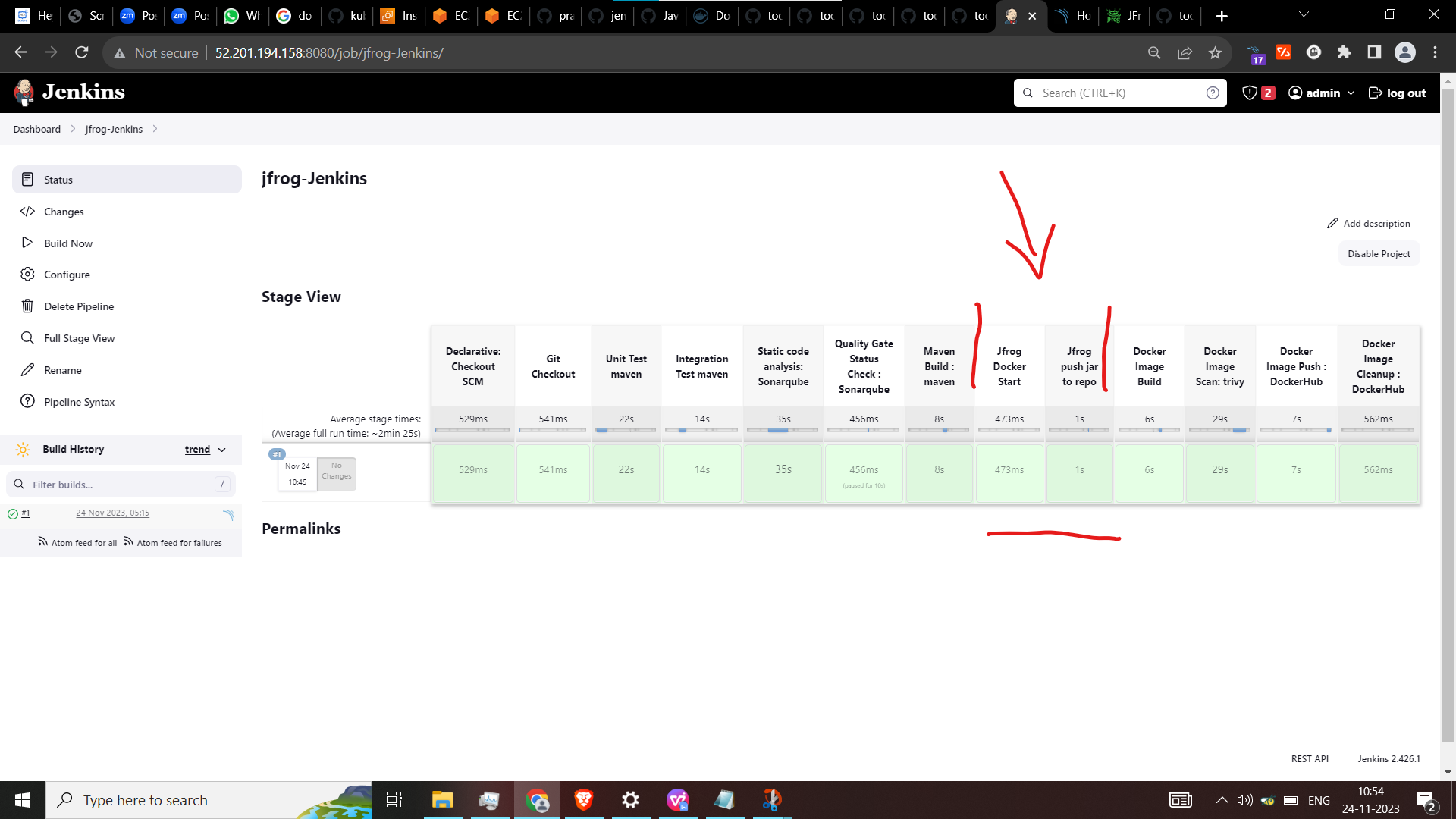
curl -X PUT -u 'admin:password' -T kubernetes-configmap-reload-0.0.1-SNAPSHOT.jar <http://jfroghost:8082/artifactory/example-repo-local/>

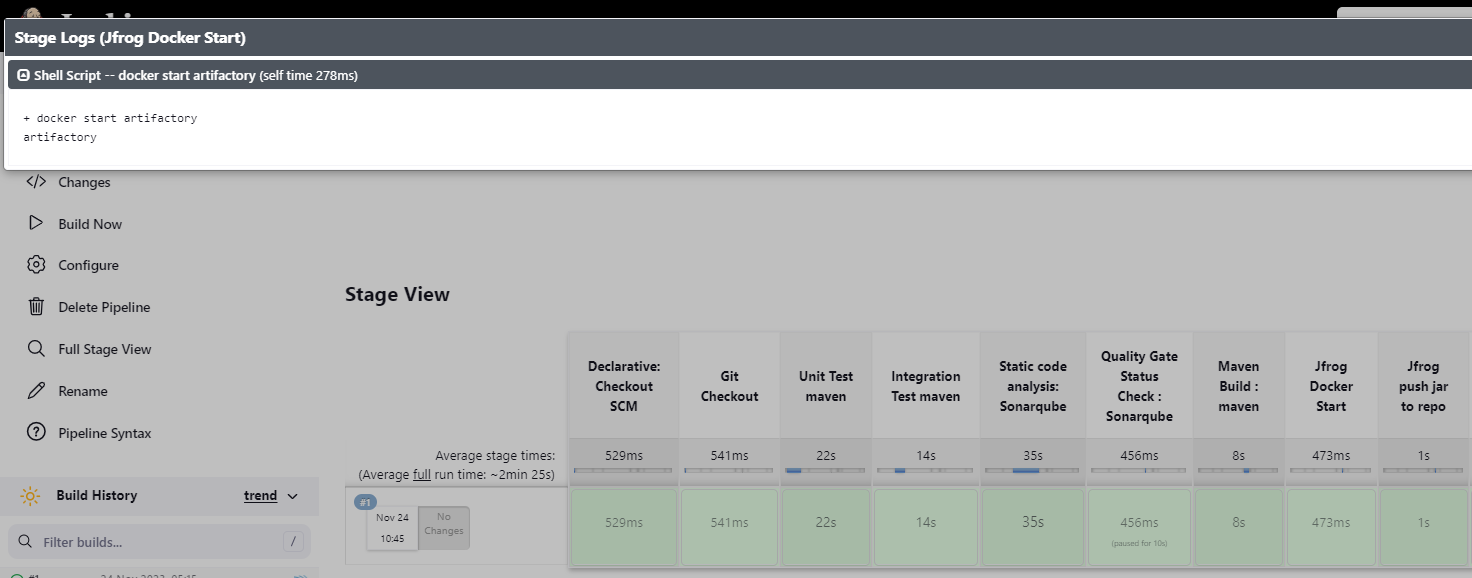
You can give host server ip directly here then giving in the /etc/hosts

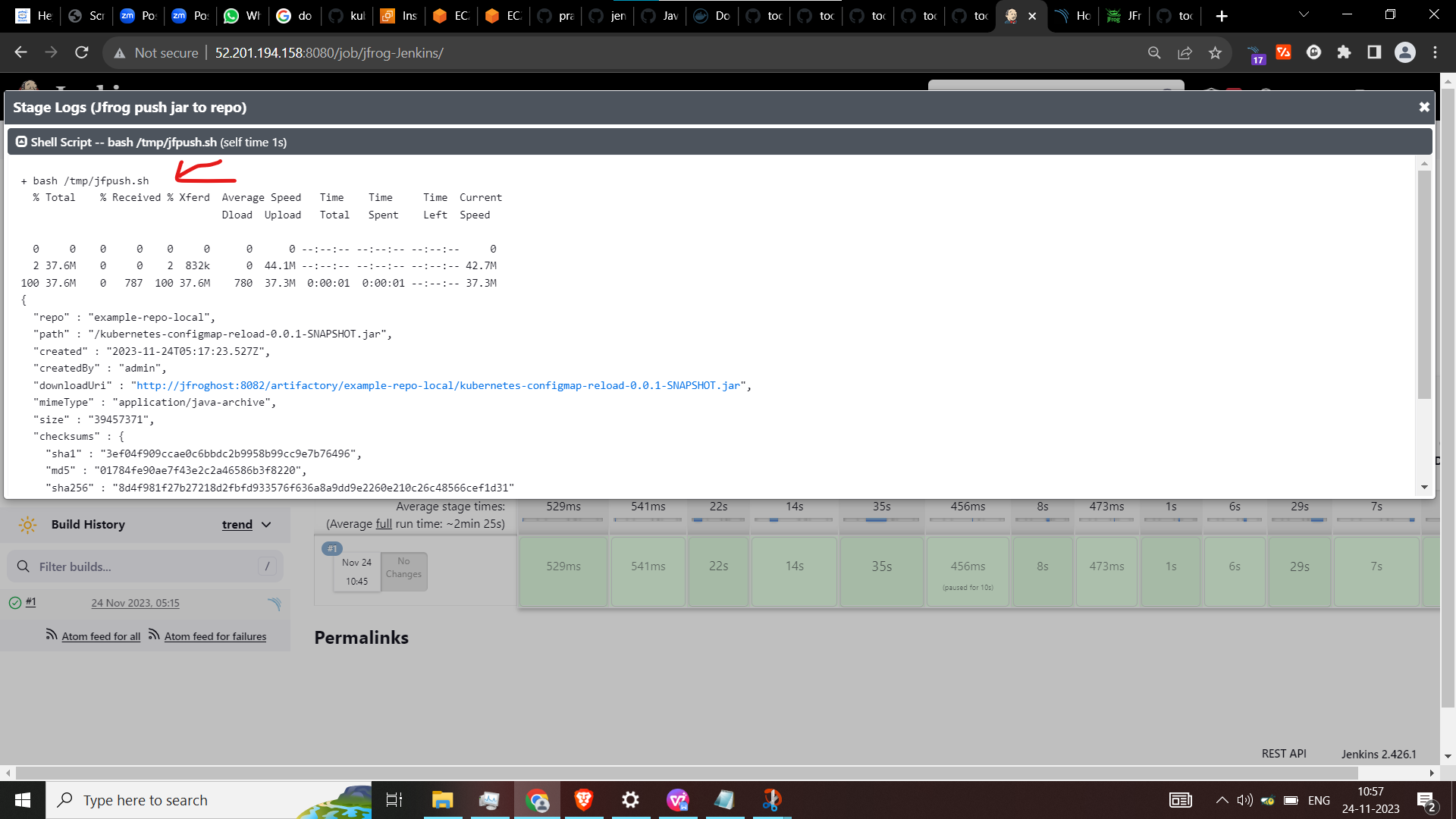
///Note - make the script file executable

chmod +x jfpush.sh

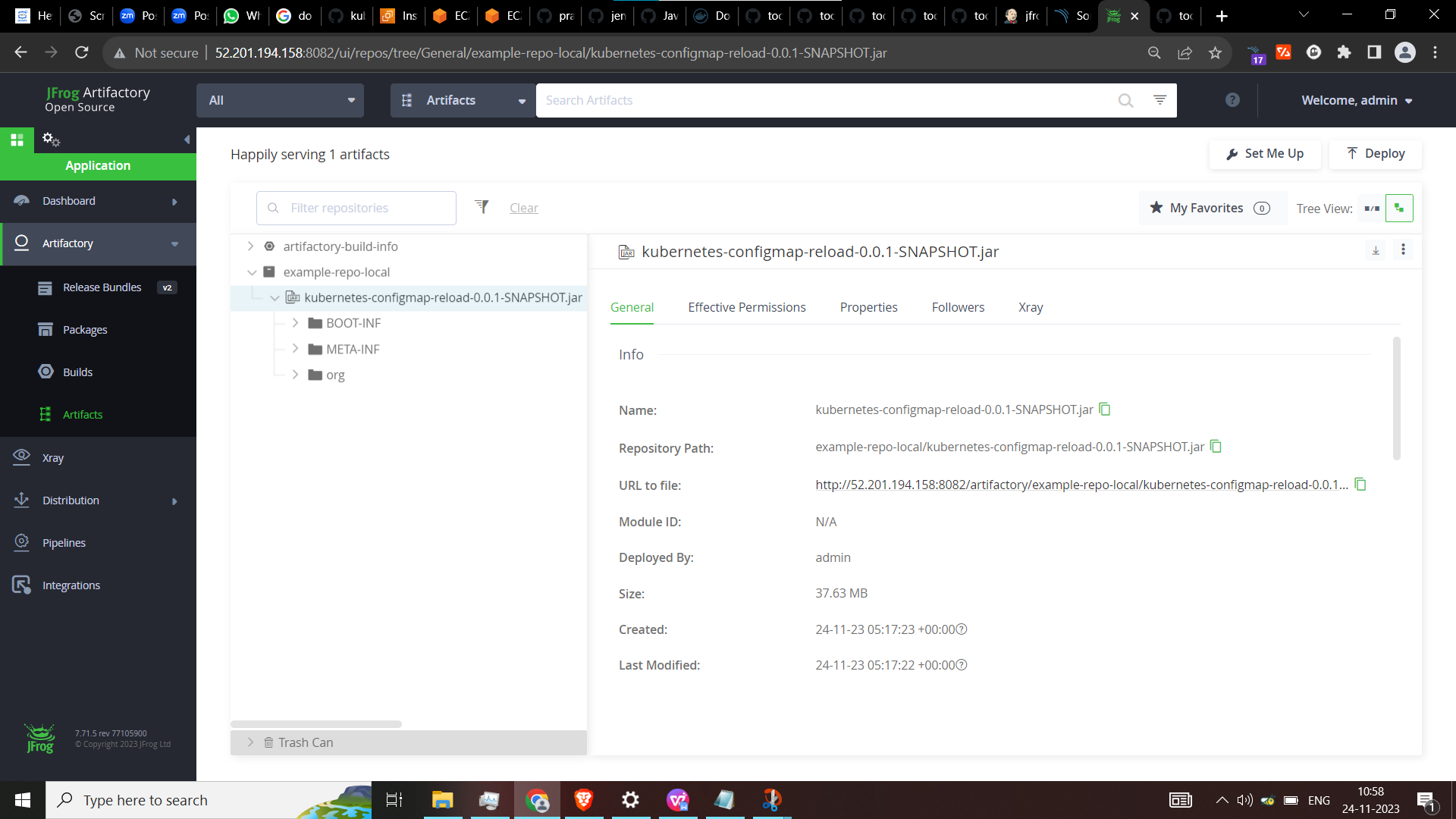
**Now execute the pipeline**







It had executed the script and have pushed the file



Sonarqube status

