Video to create, build and push spring boot project (from github) to create docker image using Jenkins ci/cd pipeline.

<https://youtu.be/PKcGy9oPVXg?si=xNxRFfyfYWRA_D_2>

video to integrate intellij with github and commit your project code to ur account’s github repo.

<https://www.youtube.com/watch?v=HM6CawhRvVQ>

Topic: Devops

Problem statement:

ci/cd pipeline to integrate, build and deploy spring boot application from github repo using maven and build docker image and push to docker hub.

1. Install Jenkins (for windows using windows msi installer and linux/mac using brew command

(refer/google Jenkins installation doc/steps for it)

1. Create github account and public git hub repo – devops-integration
2. Create spring boot proj using spring initializer and open in intellij id and create basic

Rest service to get name

1. Run and check its working.Then stop the application
2. Add Dockerfile (stored in same folder) to root directory i.e. main project directory. It contains instruction :- To load openjdk base image and build appln to create jar and copy jar from target folder to docker by jar name and then run that jar using java-jar command (as below) :-

From openjdk:8  
EXPOSE 8877  
COPY target/devops-integration.jar devops-integration.jar  
ENTRYPOINT ["java", "-jar", "devops-integration.jar"]

(Caution: do nor add Dockerfile to src folder, keep it in main project folder as it will

pickup Dockerfile from there during - docker build image command (mentioned in Jenkins pipeline script under stage:build docker image (ref pt no 31: docker build -t devops-integration -f Dockerfile . command.) )

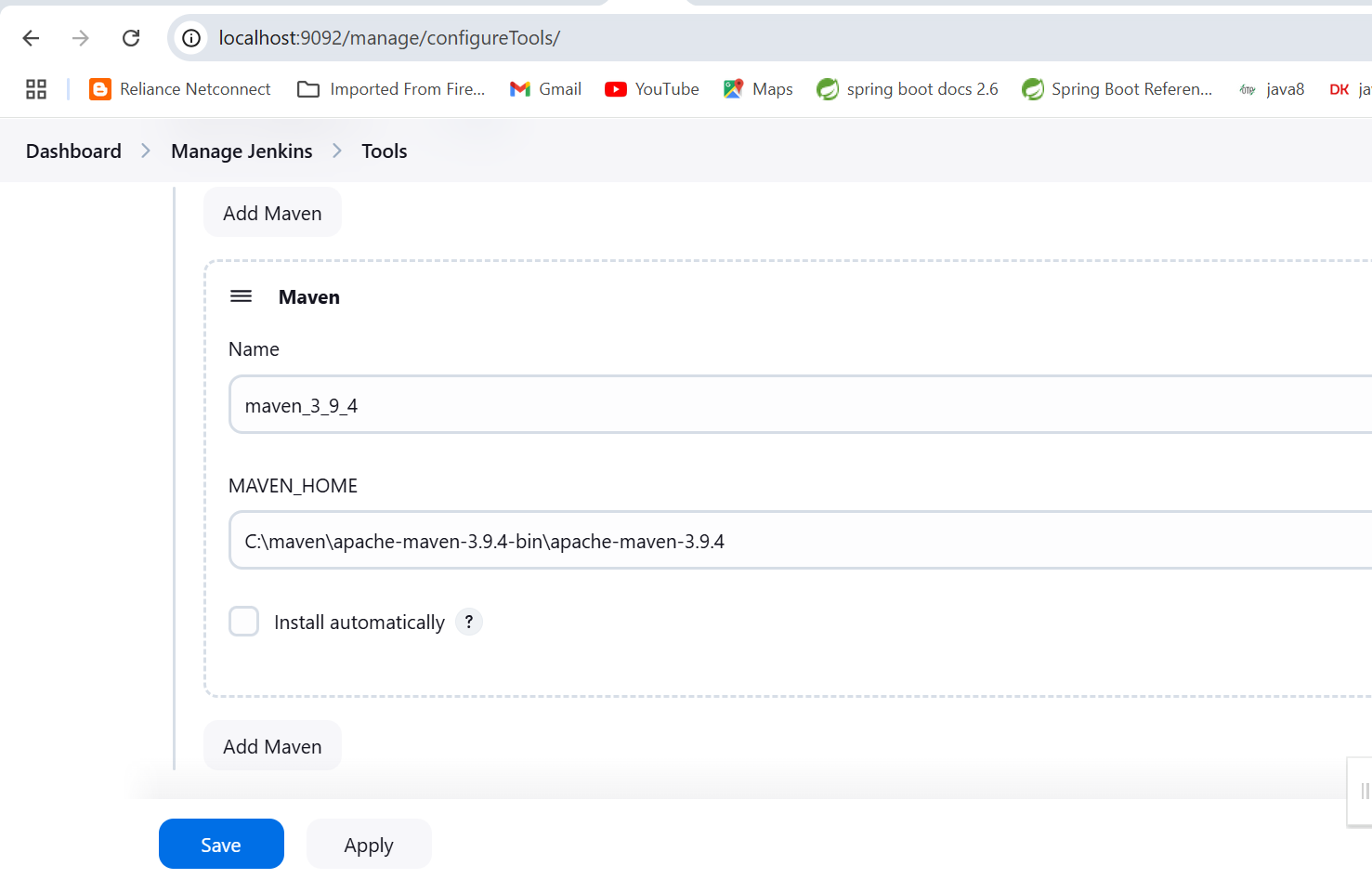
1. In pom.xml file add the tag <final-name> after <plugins> tag as below. It’s the final name with which jar will be created in target folder of your project when you do maven install phase from intellij.



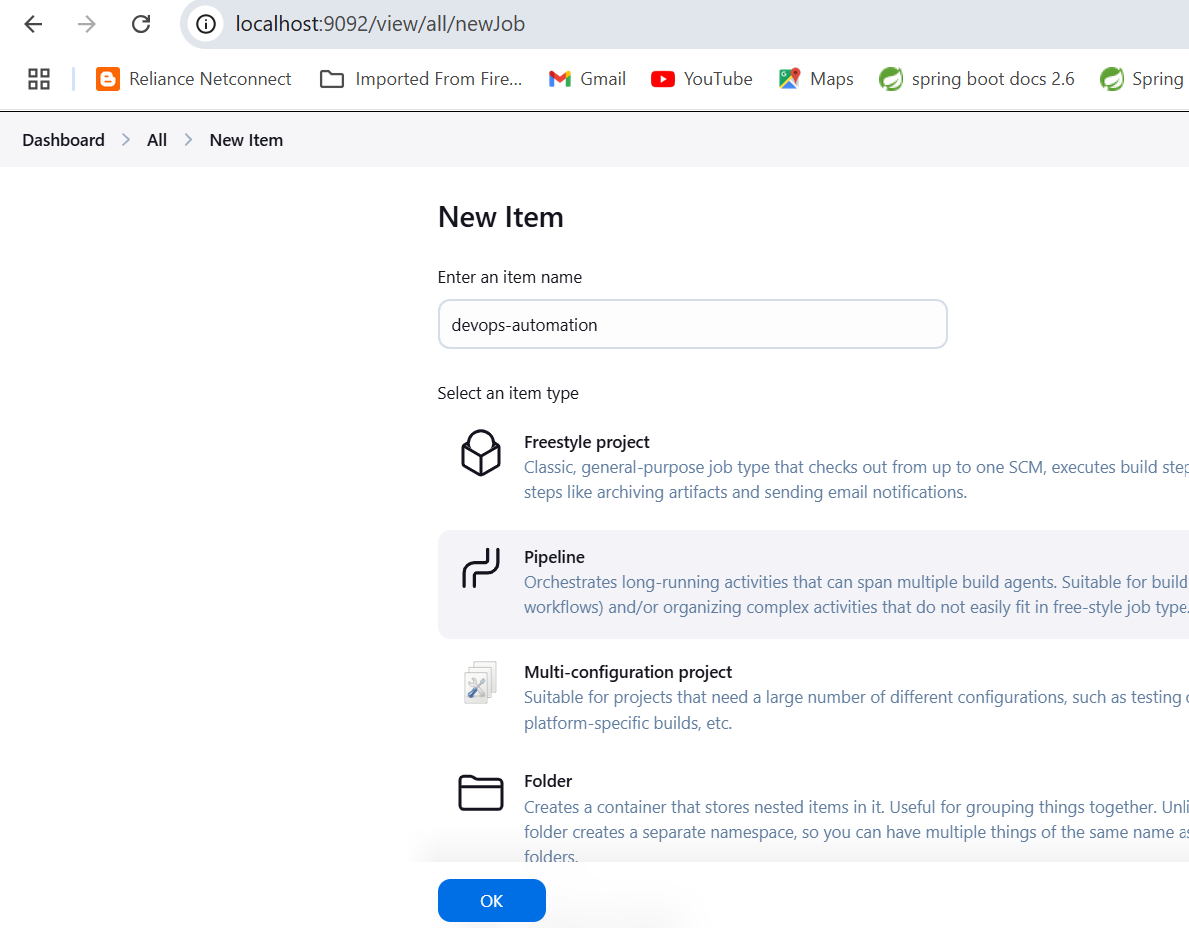
1. Integrate intellij using github and push your spring boot project directory to github repo devops-integration
2. From intellij-righ side run maven->install phase , after build successful message in console you will see devops-integration.jar gets created in target folder.
3. Start Jenkins and login to Jenkins
4. Start Docker daemon (Docker desktop)
5. From dashboard-> goto manage Jenkins-> tools

And add maven details in maven section to integrate maven installed on your local machine

With Jenkins maven as below and apply and save :-



1. Go to dashboard -> new item ->give item name as devops-automation and select item type as Pipeline and click on ok button as below:-



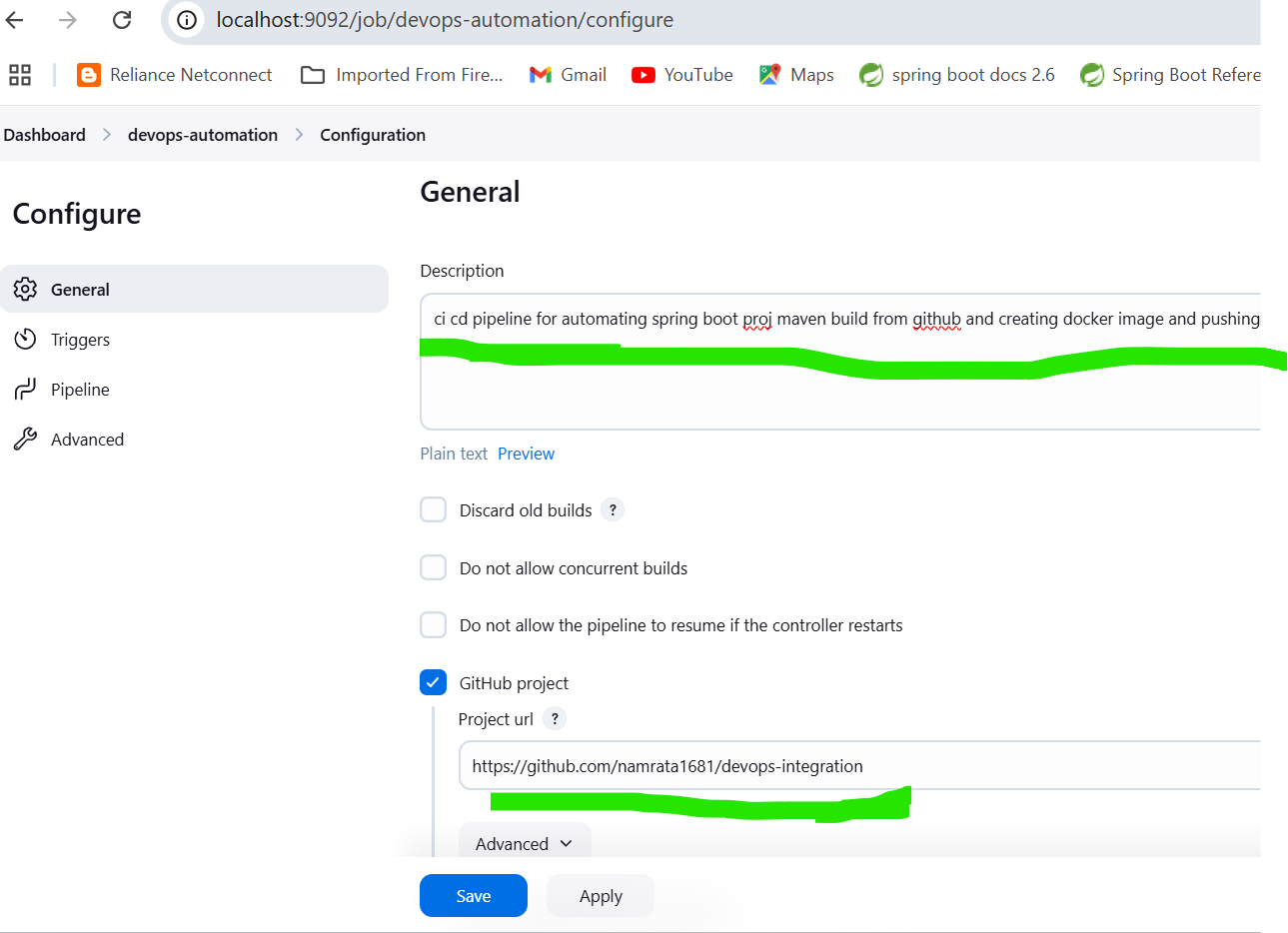
1. Give description of pipeline as ‘its ci cd pipeline to maven build spring boot github project and create docker image and push it to docker hub’

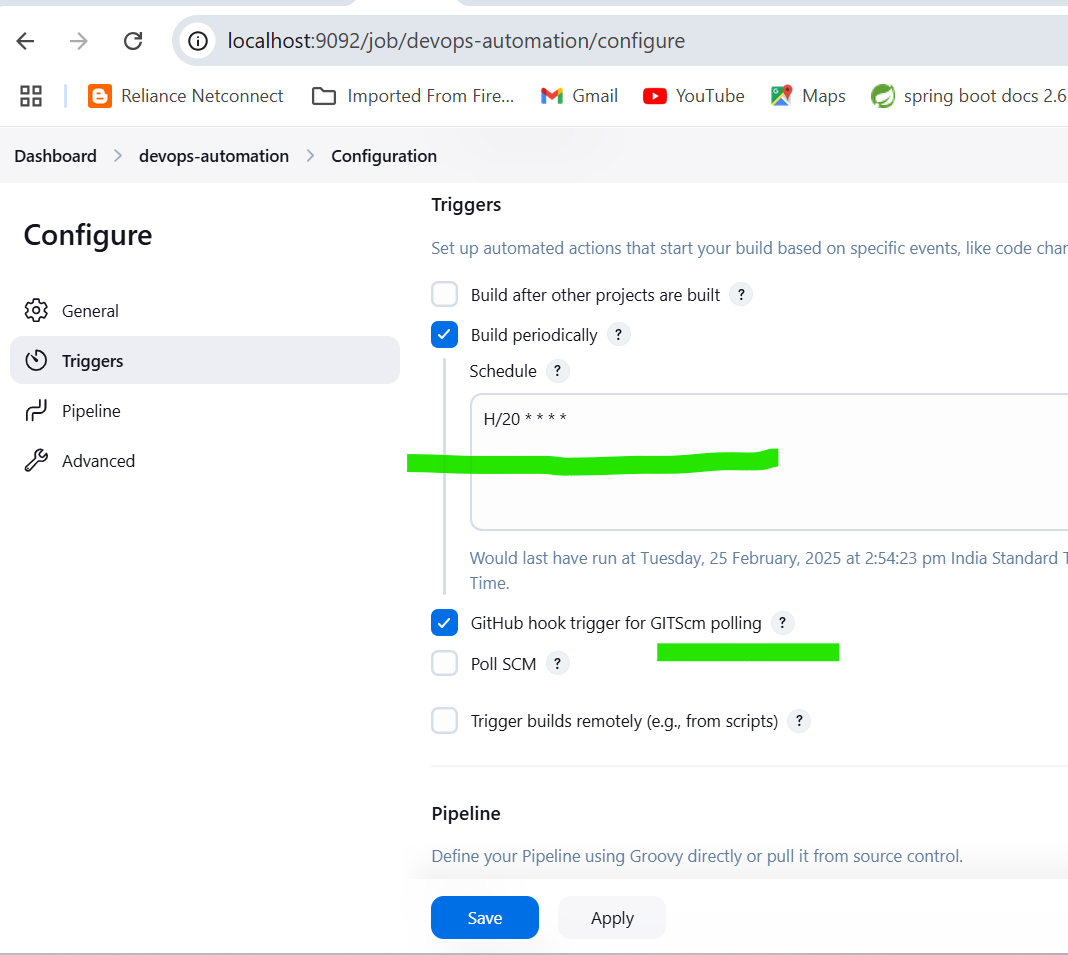
Select Git and give github repository url as below and then select triggers for periodically polling after every 20 minutes and Github hook trigger for Gitscm polling as below:-

Then apply and save it.

How to apply scheduler to Jenkins build periodically.

<https://www.lenar.io/jenkins-schedule-build-periodically/>





1. Under Pipeline section paste Jenkins pipeline script as below. (see Jenkins\_pipeline\_script groovy file store in devops\_cicd folder)

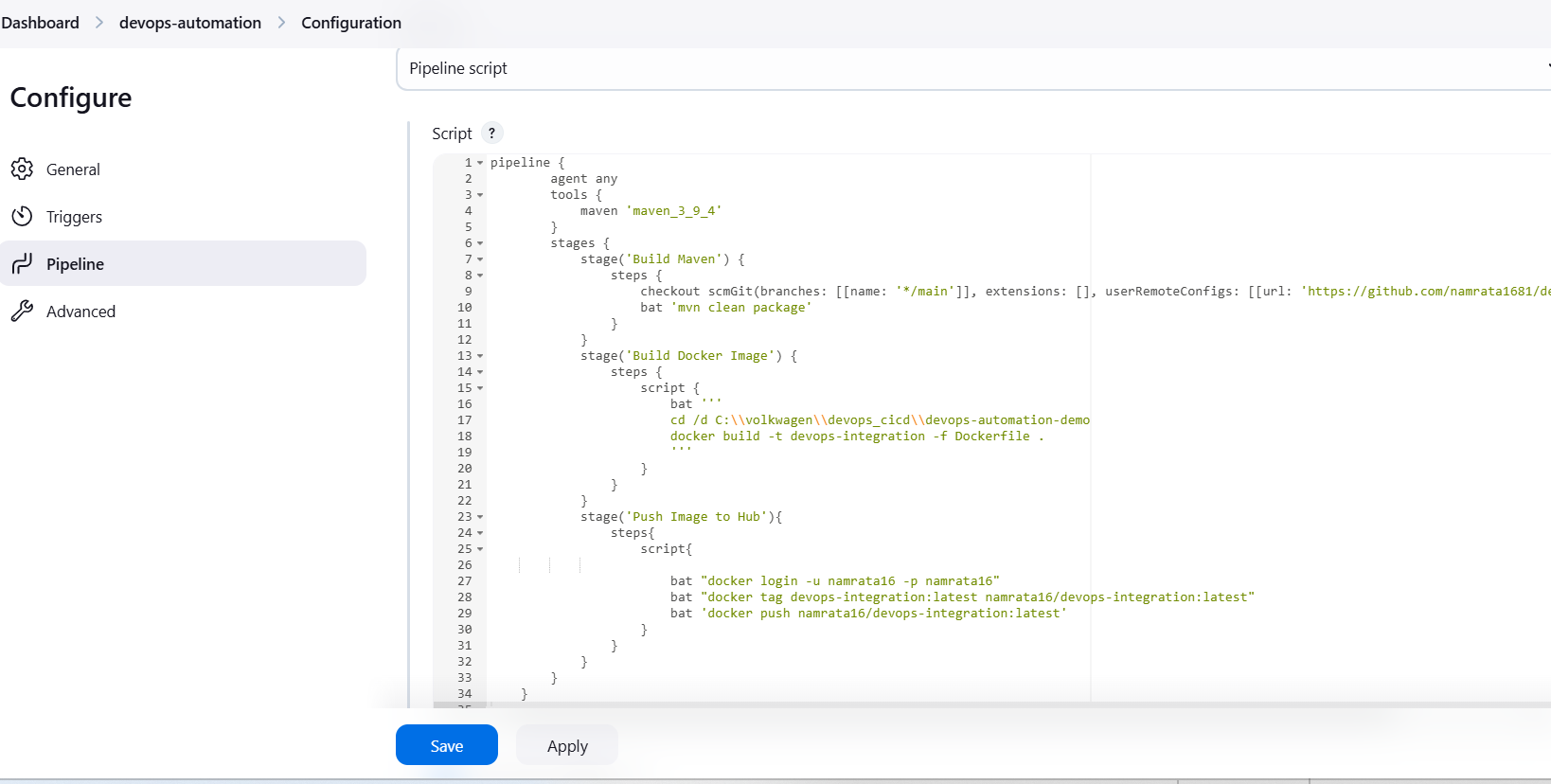
Make below changes to pipeline script before using it!!

* Change under tools-> maven : maven version to maven installed on your machine on line no 17.
* Change under --> build maven stage --> branch name --> \*/main or \*/master
* Change branch name on line no 22 to either main or master depending on branch name of your github repo
* Change githun repo url name to your github repo url in pt no 22:userRemoteConfigs
* On line no 30 under stage build docker image, give the location of your project folder on hard disk (where jar file is created in its target folder), don’t copy path till target just till project folder on ur harddisk
* Under stage ‘push image to hub’, line no 39 hard code your docker hub id and password
* Under stage ‘push image to hub’ line no 40 tag your docker jar image name with correct dockerhub id /image name:version
* Under stage ‘push image to hub’ line no 41 push the tagged image with dockerhub id and version to docker hub

1. pipeline {
2. agent any
3. tools {
4. maven 'maven\_3\_9\_4'
5. }
6. stages {
7. stage('Build Maven') {
8. steps {
9. checkout scmGit(branches: [[name: '\*/main']], extensions: [], userRemoteConfigs: [[url: 'https://github.com/namrata1681/devops-integration']])
10. bat 'mvn clean package'
11. }
12. }
13. stage('Build Docker Image') {
14. steps {
15. script {
16. bat '''
17. cd /d C:\\volkwagen\\devops\_cicd\\devops-automation-demo
18. docker build -t devops-integration -f Dockerfile .
19. '''
20. }
21. }
22. }
23. stage('Push Image to Hub'){
24. steps{
25. script{

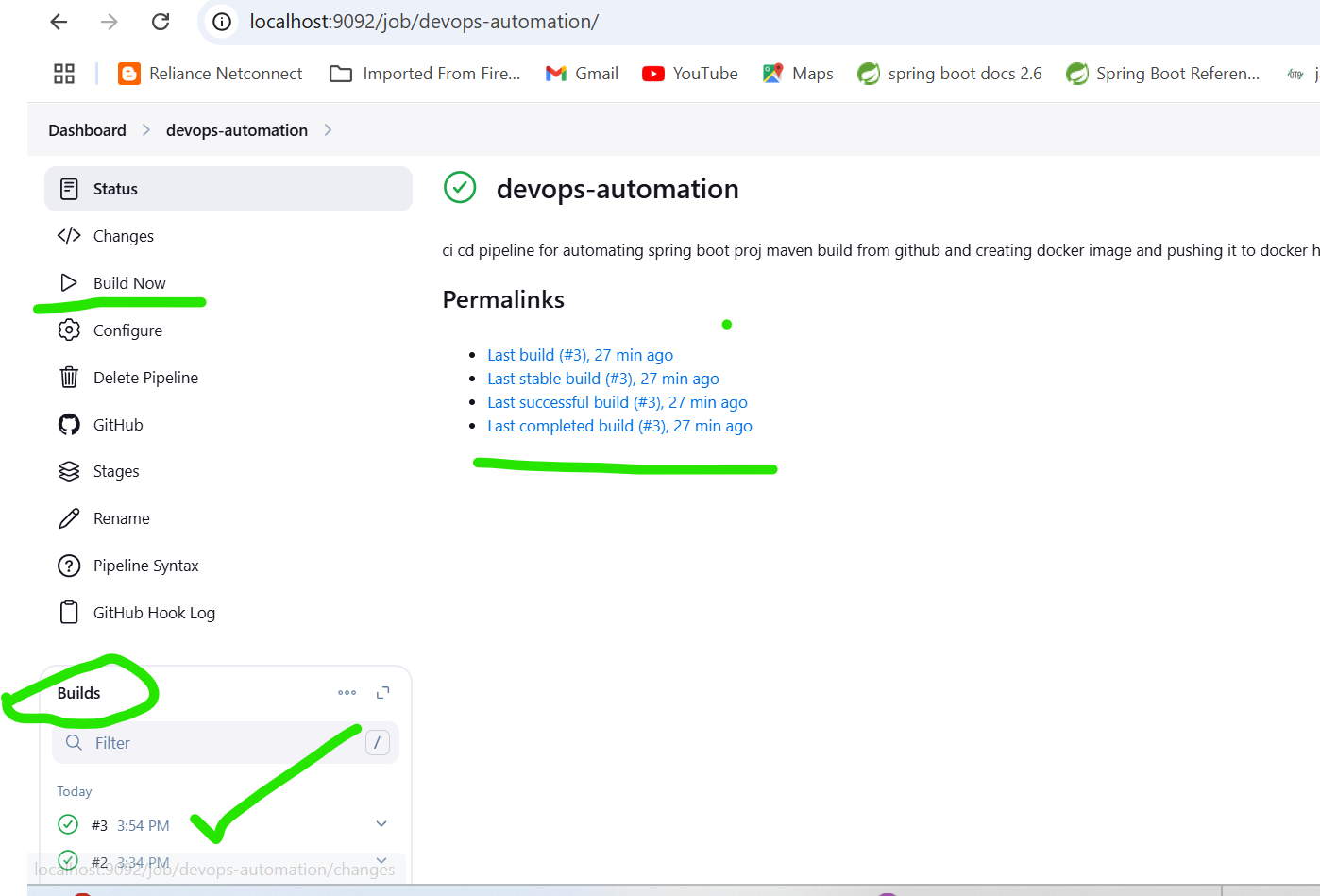
1. bat "docker login -u namrata16 -p stranger@16"
2. bat "docker tag devops-integration:latest namrata16/devops-integration:latest"
3. bat 'docker push namrata16/devops-integration:latest'
4. }
5. }
6. }
7. }
8. }

Note: namrata16 and stranger@16 is my docker hub id and password, you should enter your docker hub id and password there.



1. Click on Build now to build the project.(option is on left side)

Note: Docker desktop/docker daemon must be started before you click on build now. So that image gets build on to docker desktop.



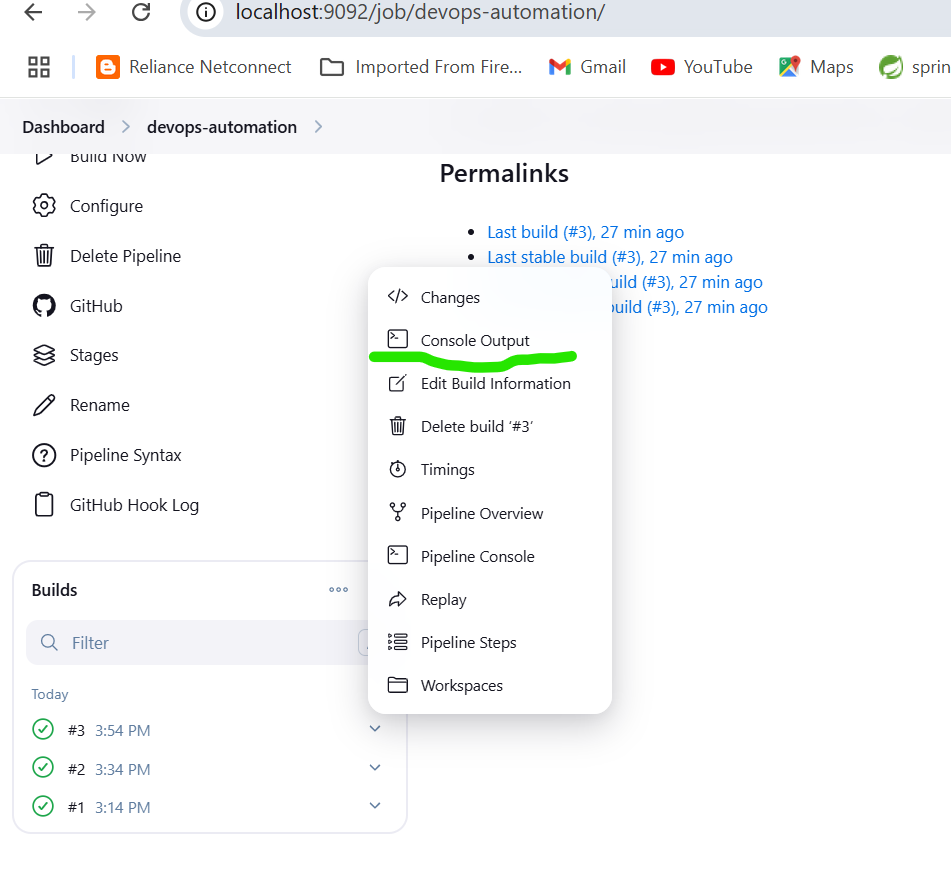
48)

You will see that project code gets pulled from github repo and image in created on docker desktop (under docker images) and image is also pushed to docker github id.

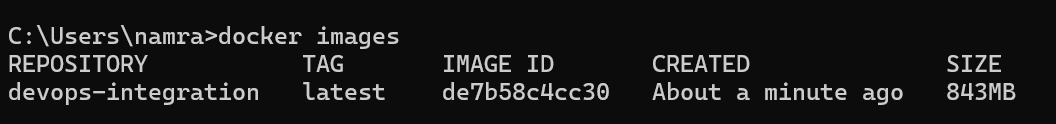
As you have also scheduled git scm pull every 20 min of evry hour project code will be pulled from git repo and maven built every 20 min and image created on docker desktop and image pushed to docker hub.

49)

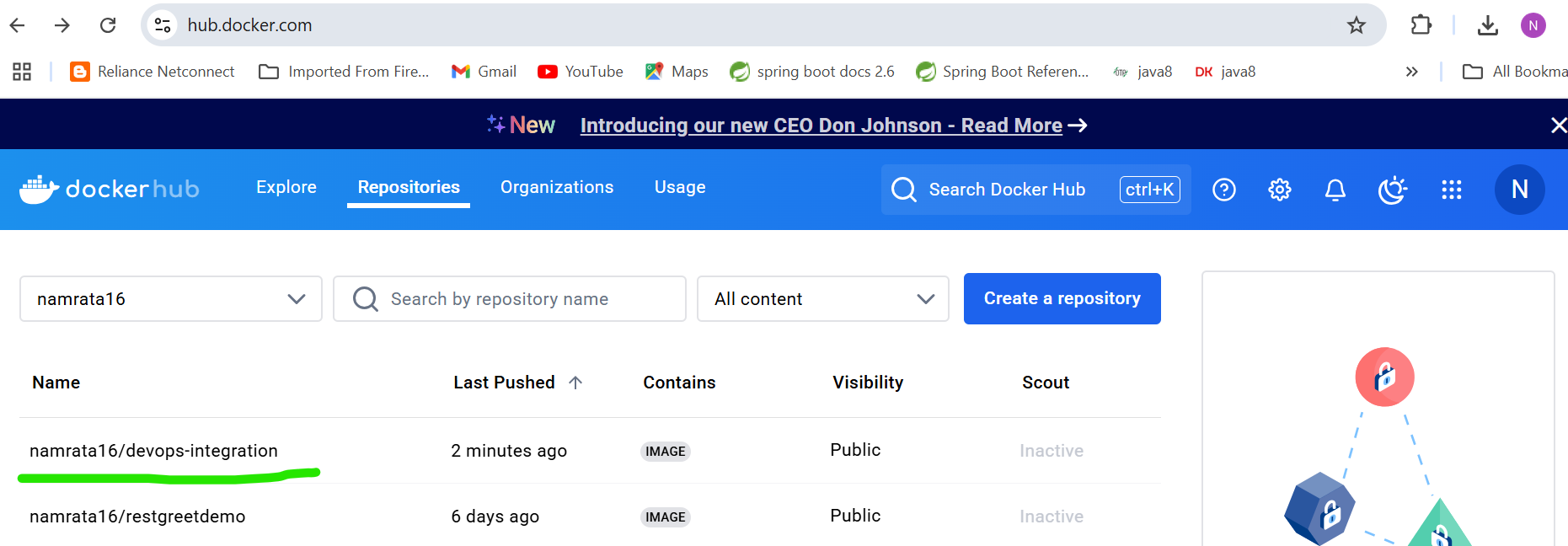
You can check last or earlier build console outputs in case build fails for any error in configuration. Correct it and apply and save and build now again.

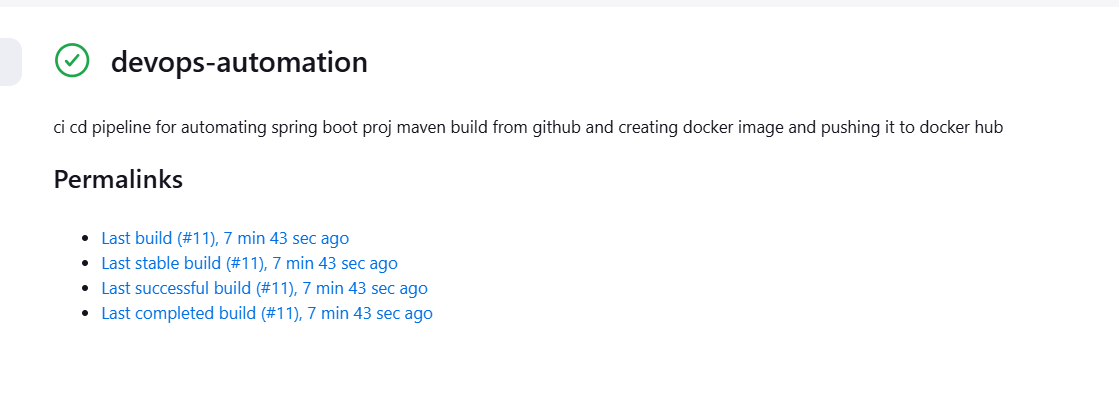


1. if you open command prompt and type docker images command you will see project docker image created on local docker desktop.



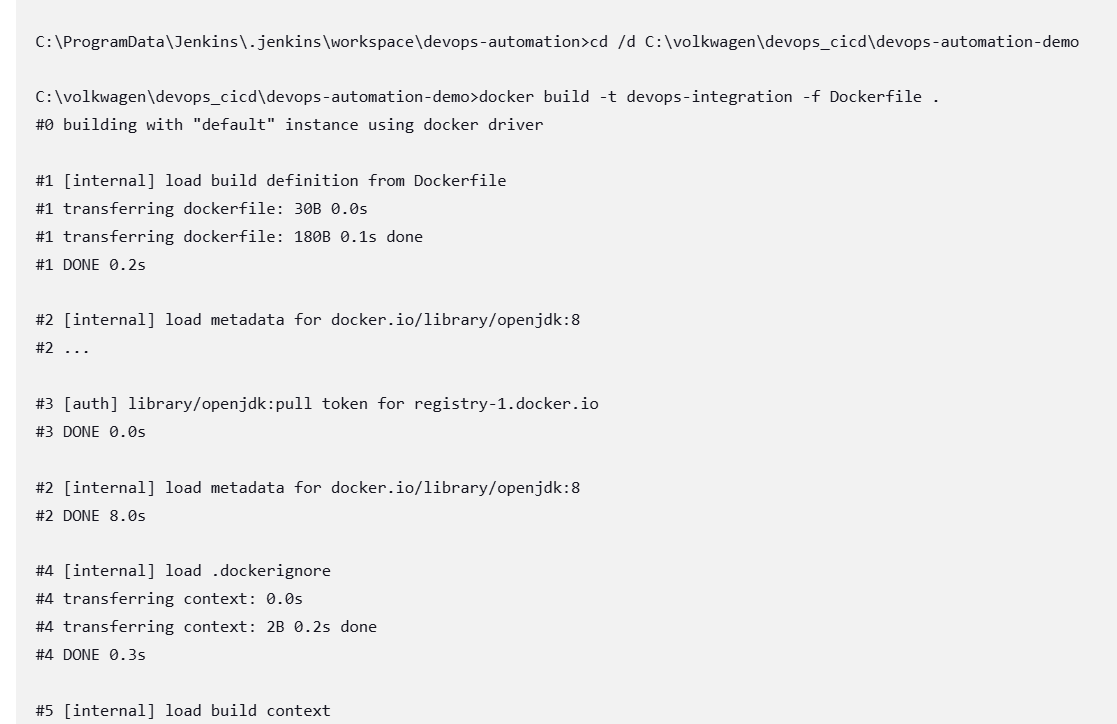
1. Open docker hub and login to docker hub id and check if image is pushed to docker hub id as below



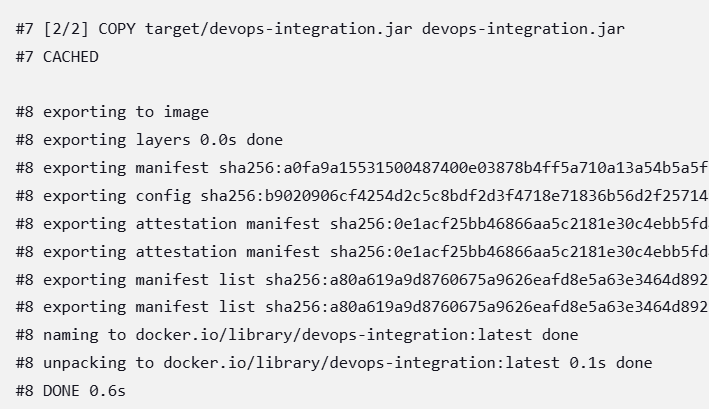
1. For latest build in Jenkins 🡪 if you see the status it should be successful will all pipeline stages completed. Only then status will be successful. 
2. Check the builds console output. You can download that entire log. I have saved that console ouput in same folder by name

#jenkinscicd\_build10\_output\_after\_image\_is\_succesfully\_pushed\_to\_doeckerhub\_repo

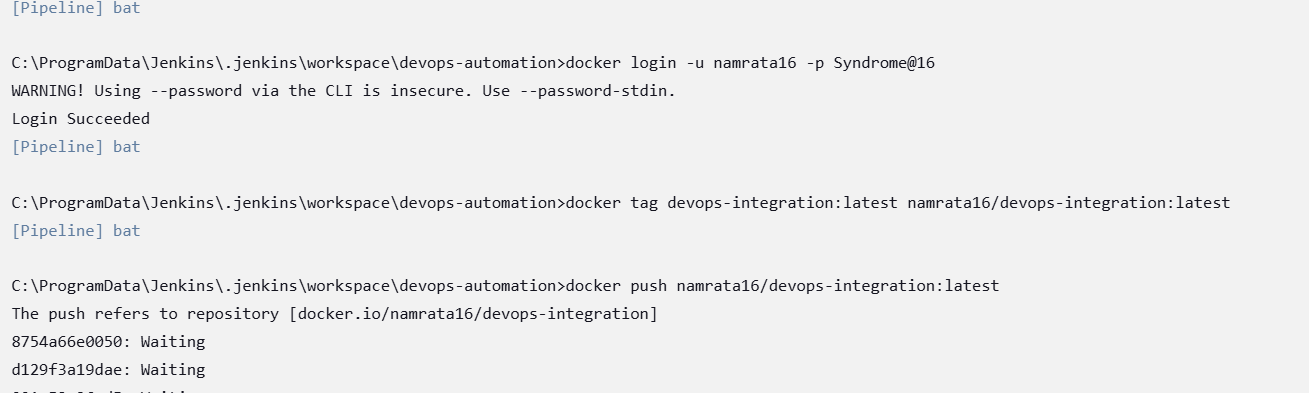
If you open console you will clearly see output of every pipeline stage as below:



Docker file – copy jar file from target folder to docker images



Pushing docker image to docker hub stage



Final pushing of image to docker hub is successful.

All stages of pipeline completed and succeeded so final build success message is seen in console ouput.

Note: pushing image to docker hub takes some waiting time.

