Practice for Lesson 7: Jenkins Integration with Docker

Practices for Lesson 7

Overview

In these practices, you will learn how to setup the environment for Docker container job creation on Jenkins instance. Further, create the Pipeline Job for Docker on Jenkins instance using a sample example and then deploying the Docker Container with the Jenkins Pipeline using the GitHub repository

Practice 7-3: Deploying Docker Container with Jenkins Pipelines

Overview

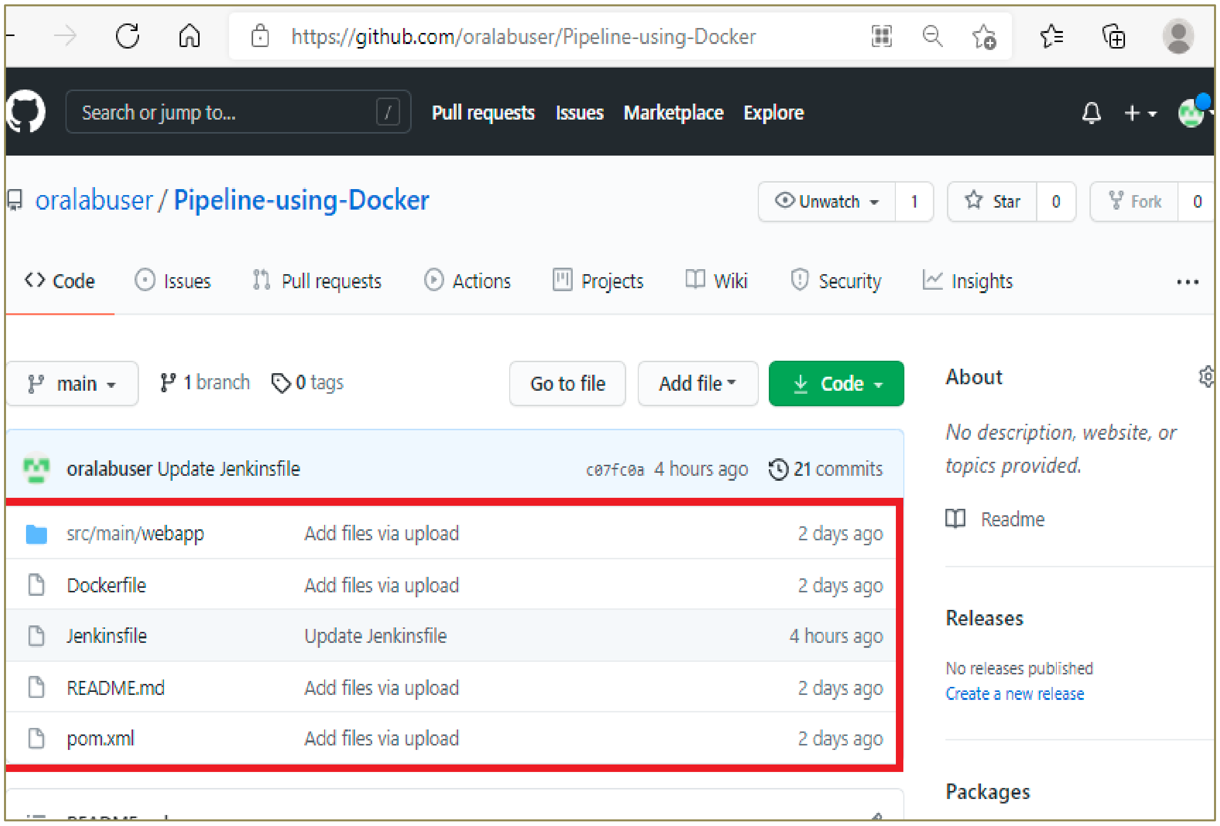
In this practice, you will learn how to deploy the Docker Container with the Jenkins Pipeline using the GitHub repository.

Assumptions

You should have completed the Practice of Lesson 7-2.

Tasks

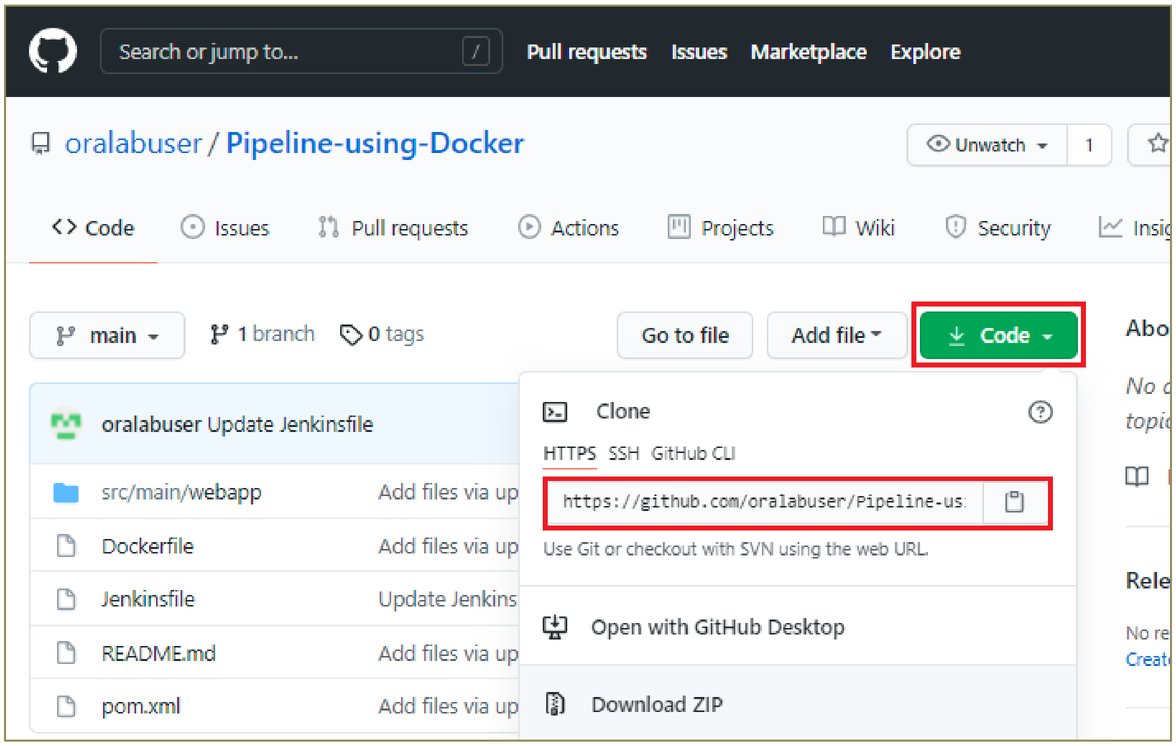
1. Create a Docker Pipeline Job on Jenkins instance using the GitHub repository.
   a. Open the GitHub repository were the link is provided below.
      Link: [oralabuser/Pipeline-using-Docker (github.com)](https://github.com/oralabuser/Pipeline-using-Docker)



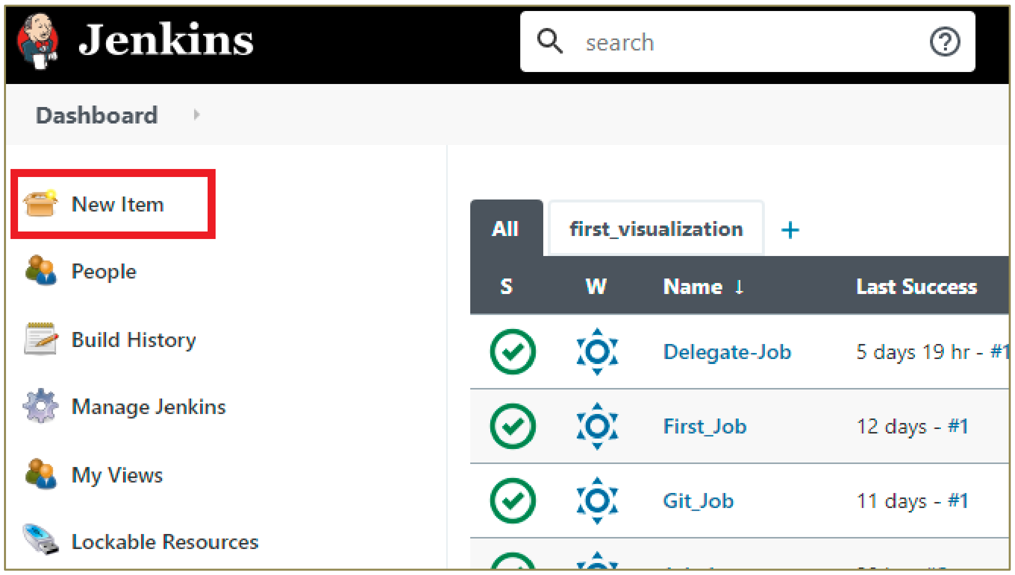
oralabuser Update Jenkinsfile c07fc0a 4 hours ago 21 commits

src/main/webapp	Add files via upload	2 days ago
Dockerfile	Add files via upload	2 days ago
Jenkinsfile	Update Jenkinsfile	4 hours ago
README.md	Add files via upload	2 days ago
pom.xml	Add files via upload	2 days ago

   b. Navigate to **Code** and copy the **HTTPS** link for the GitHub repository as given below.

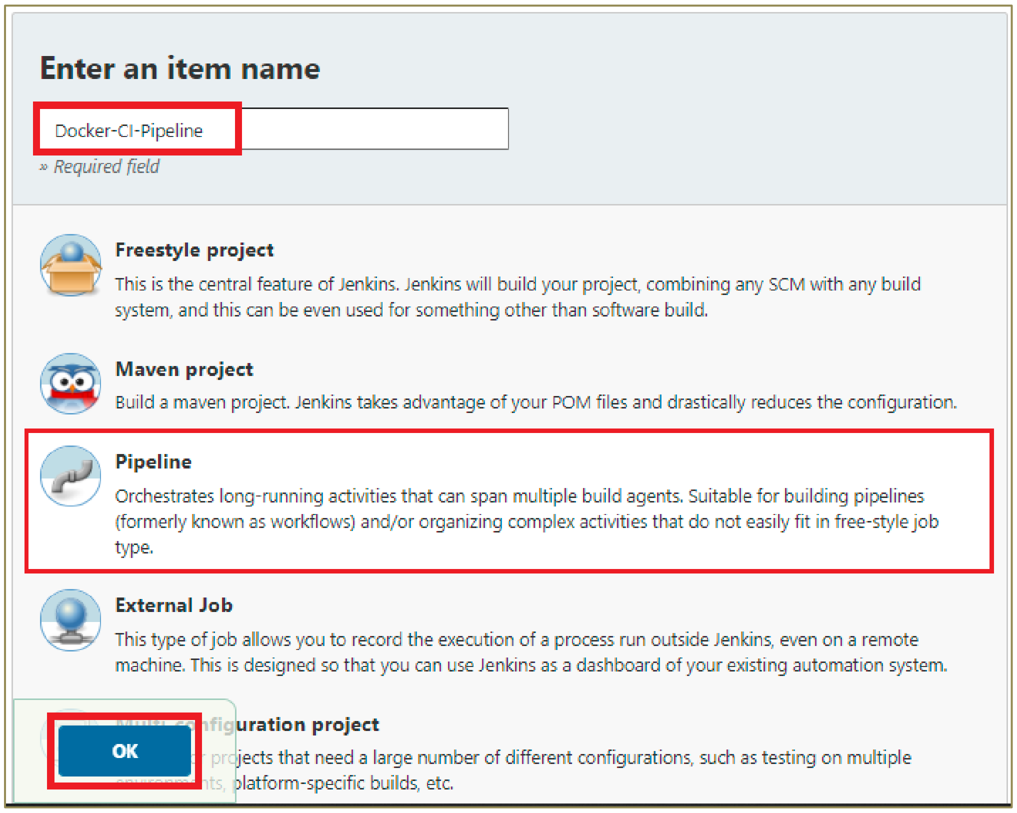


c. Open the Jenkins Dashboard, navigate to main menu and select **New Item** to create a Docker CI Pipeline Job as shown below.



S	W	Name ↓	Last Success
✓		Delegate-Job	5 days 19 hr - #1
✓		First_Job	12 days - #1
✓		Git_Job	11 days - #1

d. Provide the name for Job, select **Pipeline** and click **OK.**



Enter an item name

Docker-CI-Pipeline

» Required field

Freestyle project
This is the central feature of Jenkins. Jenkins will build your project, combining any SCM with any build system, and this can be even used for something other than software build.

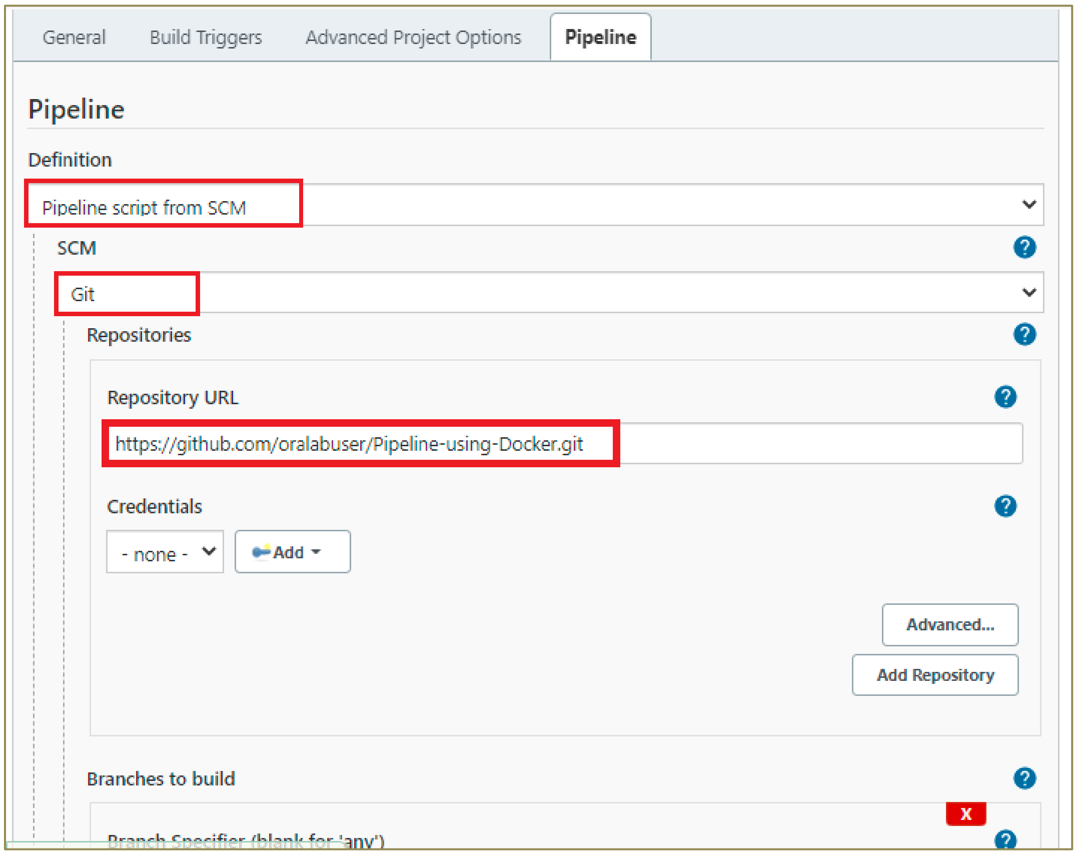
Maven project
Build a maven project. Jenkins takes advantage of your POM files and drastically reduces the configuration.

Pipeline
Orchestrates long-running activities that can span multiple build agents. Suitable for building pipelines (formerly known as workflows) and/or organizing complex activities that do not easily fit in free-style job type.

External Job
This type of job allows you to record the execution of a process run outside Jenkins, even on a remote machine. This is designed so that you can use Jenkins as a dashboard of your existing automation system.

Multi-configuration project
Suitable for projects that need a large number of different configurations, such as testing on multiple environments, platform-specific builds, etc.

e. Navigate to **Pipeline** and select **Pipeline Script from SCM** under Definition. Under **SCM** select **Git** and paste the **Git** Repository **URL** of the **GitHub** as shown below.



Pipeline

Definition: Pipeline script from SCM

SCM: Git

Repositories

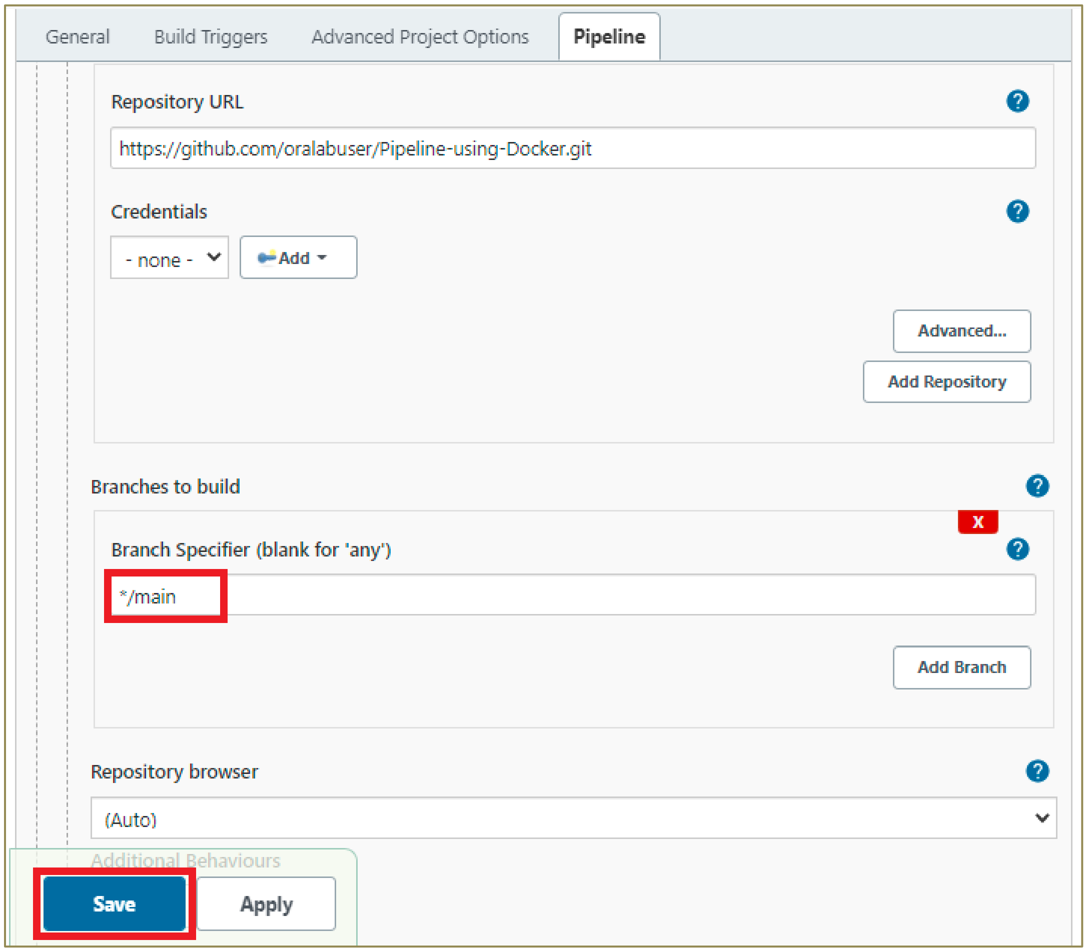
Repository URL: https://github.com/oralabuser/Pipeline-using-Docker.git

Credentials: - none -

Branches to build

Branch Specifier (blank for 'any')

f. Change the **Branch Specifier** has **main** as shown below. Click **Apply** and **Save.**



Repository URL: https://github.com/oralabuser/Pipeline-using-Docker.git

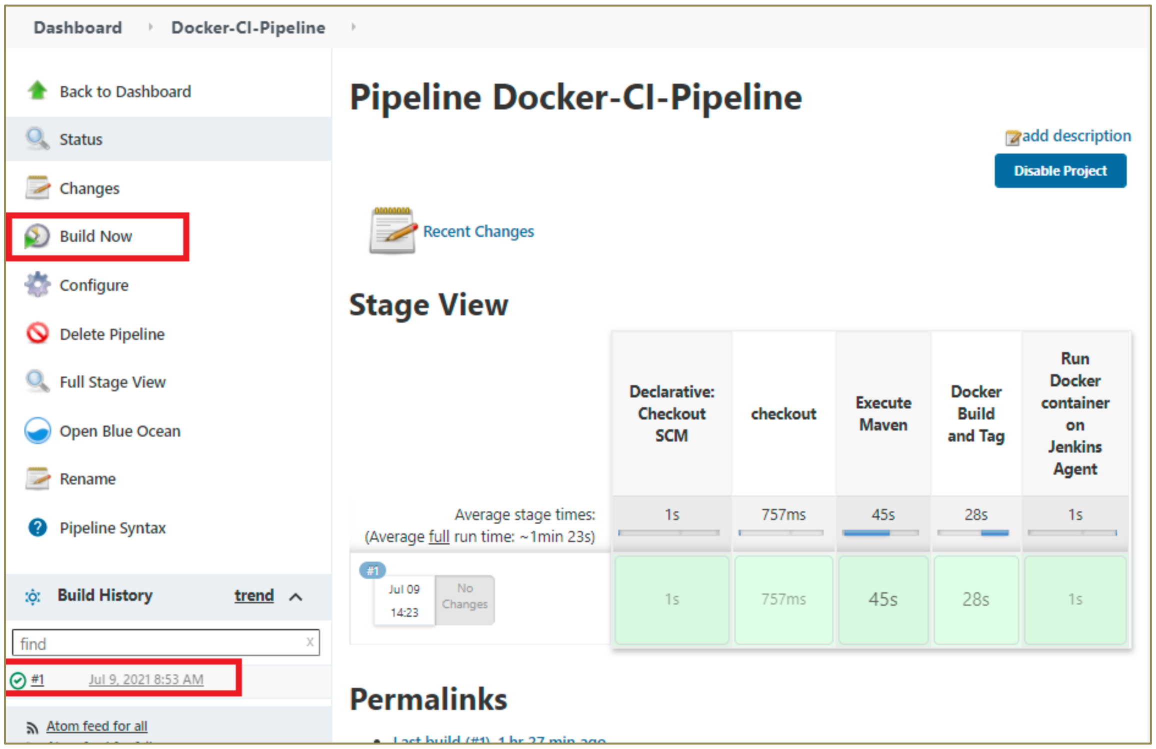
Credentials: - none -

Branches to build

Branch Specifier (blank for 'any'): */main

Repository browser: (Auto)

g. Job is created successfully, click **Build Now** to execute the pipeline. Click on build created under **Build History** as shown below.

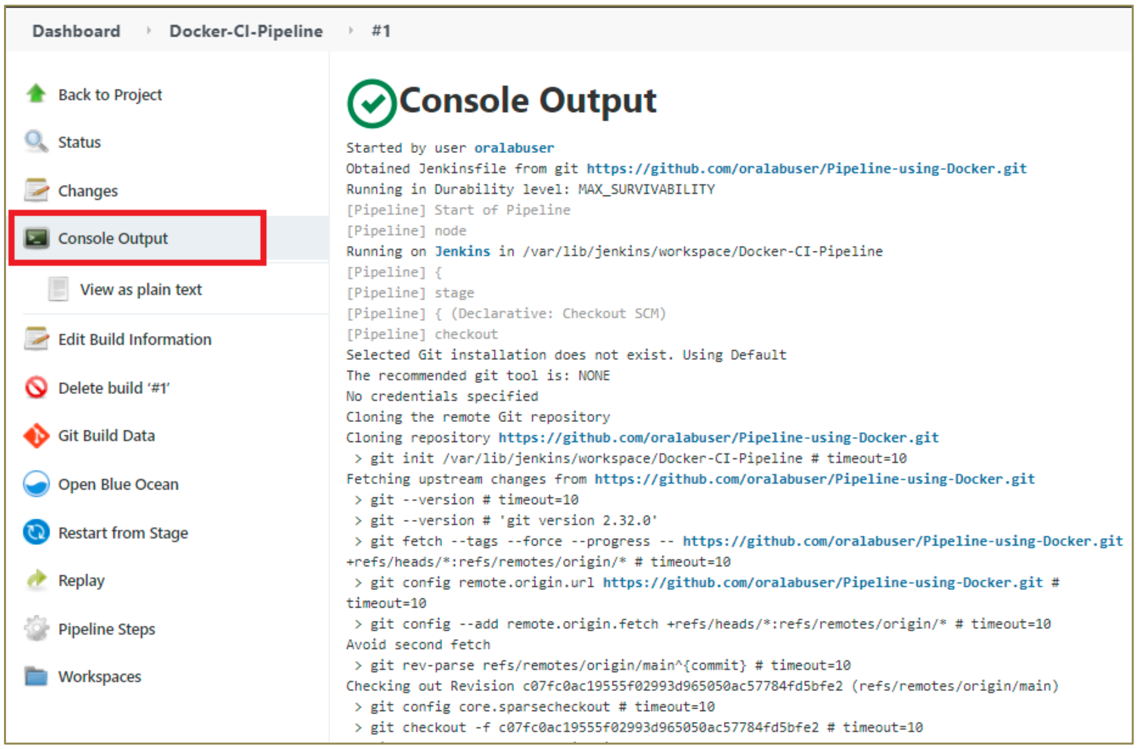


Pipeline Docker-CI-Pipeline

Stage View

	Declarative: Checkout SCM	checkout	Execute Maven	Docker Build and Tag	Run Docker container on Jenkins Agent
Average stage times: (Average full run time: ~1min 23s)	1s	757ms	45s	28s	1s
#1 Jul 09 14:23 No Changes	1s	757ms	45s	28s	1s

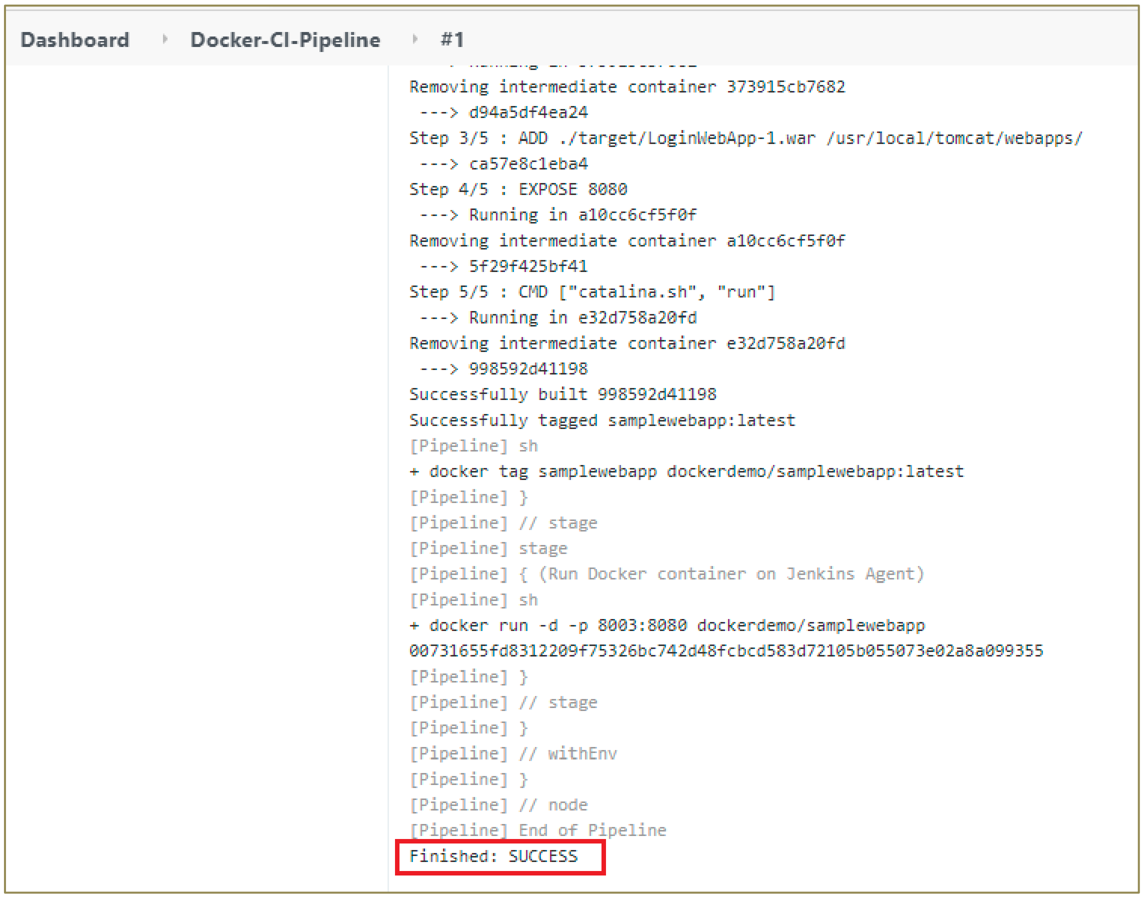
h. In the **Build** page, click **Console Output** to view the output of the job. The pipeline stages of the script is executed successfully.



Console Output

```
Started by user oralabuser
Obtained Jenkinsfile from git https://github.com/oralabuser/Pipeline-using-Docker.git
Running in Durability level: MAX_SURVIVABILITY
[Pipeline] Start of Pipeline
[Pipeline] node
Running on Jenkins in /var/lib/jenkins/workspace/Docker-CI-Pipeline
[Pipeline] {
[Pipeline] stage
[Pipeline] { (Declarative: Checkout SCM)
[Pipeline] checkout
Selected Git installation does not exist. Using Default
The recommended git tool is: NONE
No credentials specified
Cloning the remote Git repository
Cloning repository https://github.com/oralabuser/Pipeline-using-Docker.git
 > git init /var/lib/jenkins/workspace/Docker-CI-Pipeline # timeout=10
Fetching upstream changes from https://github.com/oralabuser/Pipeline-using-Docker.git
 > git --version # timeout=10
 > git --version # 'git version 2.32.0'
 > git fetch --tags --force --progress -- https://github.com/oralabuser/Pipeline-using-Docker.git
+refs/heads/*:refs/remotes/origin/* # timeout=10
 > git config remote.origin.url https://github.com/oralabuser/Pipeline-using-Docker.git #
timeout=10
 > git config --add remote.origin.fetch +refs/heads/*:refs/remotes/origin/* # timeout=10
Avoid second fetch
 > git rev-parse refs/remotes/origin/main^{commit} # timeout=10
Checking out Revision c07fc0ac19555f02993d965050ac57784fd5bfe2 (refs/remotes/origin/main)
 > git config core.sparsecheckout # timeout=10
 > git checkout -f c07fc0ac19555f02993d965050ac57784fd5bfe2 # timeout=10
```

i. On successful execution of all the pipeline stages finally the **SUCCESS** message is displayed as shown below.



```
Removing intermediate container 373915cb7682
 ---> d94a5df4ea24
Step 3/5 : ADD ./target/LoginWebApp-1.war /usr/local/tomcat/webapps/
 ---> ca57e8c1eba4
Step 4/5 : EXPOSE 8080
 ---> Running in a10cc6cf5f0f
Removing intermediate container a10cc6cf5f0f
 ---> 5f29f425bf41
Step 5/5 : CMD ["catalina.sh", "run"]
 ---> Running in e32d758a20fd
Removing intermediate container e32d758a20fd
 ---> 998592d41198
Successfully built 998592d41198
Successfully tagged samplewebapp:latest
[Pipeline] sh
+ docker tag samplewebapp dockerdemo/samplewebapp:latest
[Pipeline] }
[Pipeline] // stage
[Pipeline] stage
[Pipeline] { (Run Docker container on Jenkins Agent)
[Pipeline] sh
+ docker run -d -p 8003:8080 dockerdemo/samplewebapp
00731655fd8312209f75326bc742d48fcbcd583d72105b055073e02a8a099355
[Pipeline] }
[Pipeline] // stage
[Pipeline] }
[Pipeline] // withEnv
[Pipeline] }
[Pipeline] // node
[Pipeline] End of Pipeline
Finished: SUCCESS
```

2. Close the terminal, Logout from the AWS Management console and Jenkins Dashboard.