

Lab Documentation

How to use Jenkins to Automate Docker build and push to Docker hub.

Note: download and run Jenkins using Docker image: "jenkins/jenkins:lts"

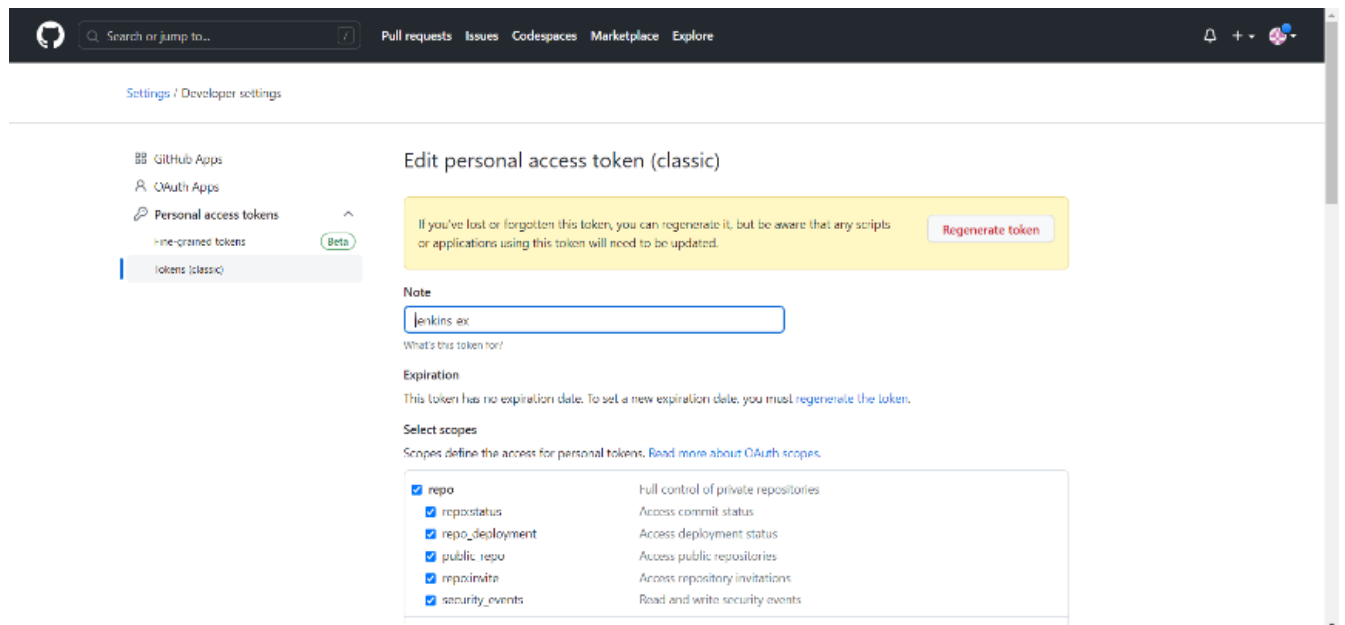
Step 1: clone github repository and push it to my github account as private repository

```
git clone https://github.com/Ahmad-Aladdin/jenkins_nodejs_example
cd .\jenkins_nodejs_example\ (the project folder)
git init
git add .
git remote add origin https://github.com/AlShaymaaHamdan/jenkins_nodeapp.git
git commit -m "first commit"
git push -u origin master
```

Step 2: Create Personal access token in github

Open <https://github.com/> and sign in to your account.

Go to "Settings > Developer Settings > Generate new Token"



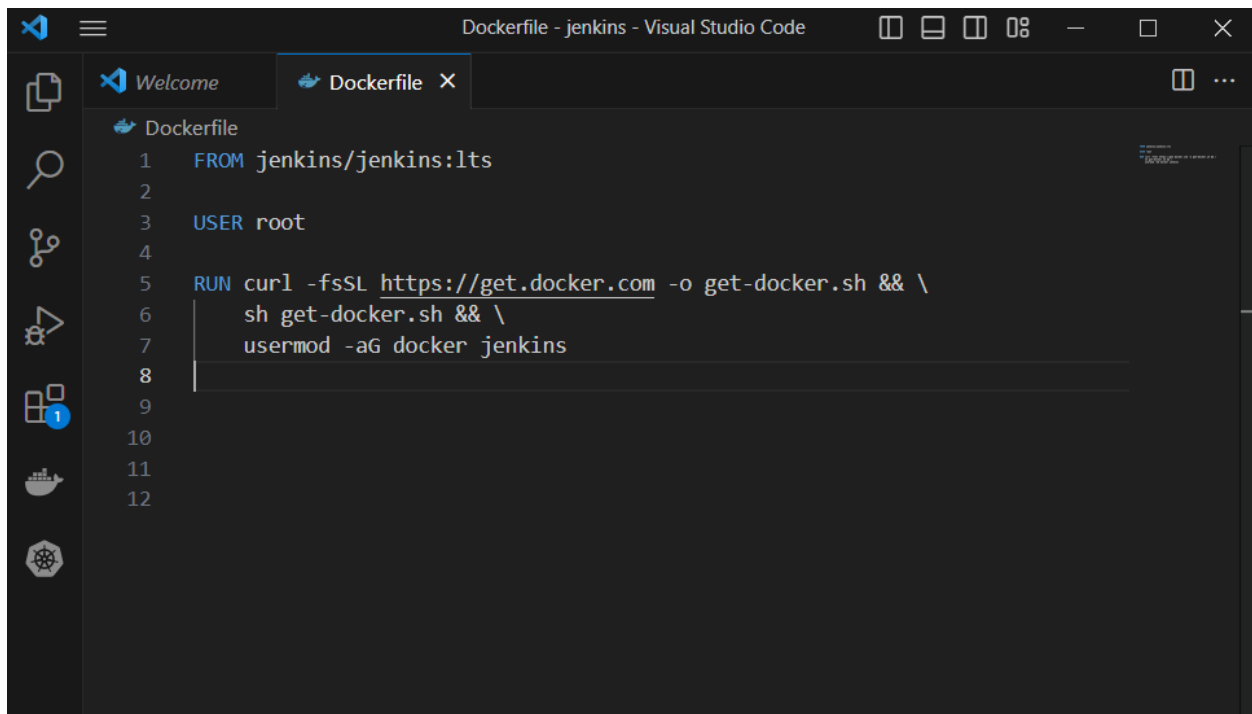
Step 3: Pull Jenkins image

After starting “Docker Desktop”, run this command to pull public Jenkins image from “DockerHub”:

```
docker pull jenkins/Jenkins:lts
```

Step 4: Build Jenkins image from Dockerfile to run docker in Jenkins, so we can use docker commands in Jenkins project

- Open Dockerfile to view commands



- Use Dockerfile to build image named Jenkins:lts, using command:

```
docker build -t jenkins:lts .
```

- Run container from jenkins:lts image that I built before.
- Map the /var/jenkins_home directory inside the container to the local directory /Desktop/Training/jenkins/jenkins-home
- Use volume mount to let jenkins run the docker client

Using command:

```
docker run -d -p 8080:8080 -v /Desktop/Training/jenkins/jenkins-home:/var/jenkins-home -v /var/run/docker.sock:/var/run/docker.sock jenkins:lts
```

- Start Jenkins:

Go to “localhost:8080”, jenkins will open and ask for admin password

Get it by running this cmd:

```
docker exec -it ab51 cat /var/jenkins_home/secrets/initialAdminPassword
```

Note: ab51 is container ID (Jenkins container)

Setup “Jenkins” and create admin user

Step 5: Create a freestyle job on Jenkins

- Click “+ New Item” and choose “Freestyle Project”

Jenkins Search (CTRL+K) ? [Notifications] [User] AA log out

Dashboard > All >

Enter an item name

» This field cannot be empty, please enter a valid name

- 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.
- 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.
- Multi-configuration project**
Suitable for projects that need a large number of different configurations, such as testing on multiple environments, platform-specific builds, etc.
- Folder**
Creates a container that stores nested items in it. Useful for grouping things together. Unlike view, which is just a filter, a folder creates a separate namespace, so you can have multiple things of the same name as long as they are in different folders.
- Multibranch Pipeline**
Creates a set of Pipeline projects according to detected branches in one SCM repository.
- Organization Folder**
Creates a set of multibranch project subfolders by scanning for repositories.

If you want to create a new item from other existing, you can use this option:

Copy from

OK

REST API Jenkins 2.387.2

- Add your “Github” private repository and your “Github” credentials.

Dashboard > nodeapp1 > Configuration

Configure

- General
- Source Code Management
- Build Triggers
- Build Environment
- Build Steps
- Post-build Actions

☐ None

☒ Git ?

Repositories ?

Repository URL ?
https://github.com/AlshaymaaHamdan/jenkins_nodeapp.git

Credentials ?
AlshaymaaHamdan/***** (github)

Add

Advanced

Add Repository

Branches to build ?

Branch Specifier (blank for 'any') ?

Save Apply

- To create “Github” credential click on Add and choose username and password (separated) then fill your username and password (your “Github access token” you created before)

Dashboard > nodeapp1 > Configuration

Configure

- General
- Source Code Management
- Build Triggers
- Build Environment
- Build Steps
- Post-build Actions

Kind
Username with password

Scope ?
Global (Jenkins, nodes, items, all child items, etc)

Username ?
AlshaymaaHamdan

☐ Treat username as secret ?

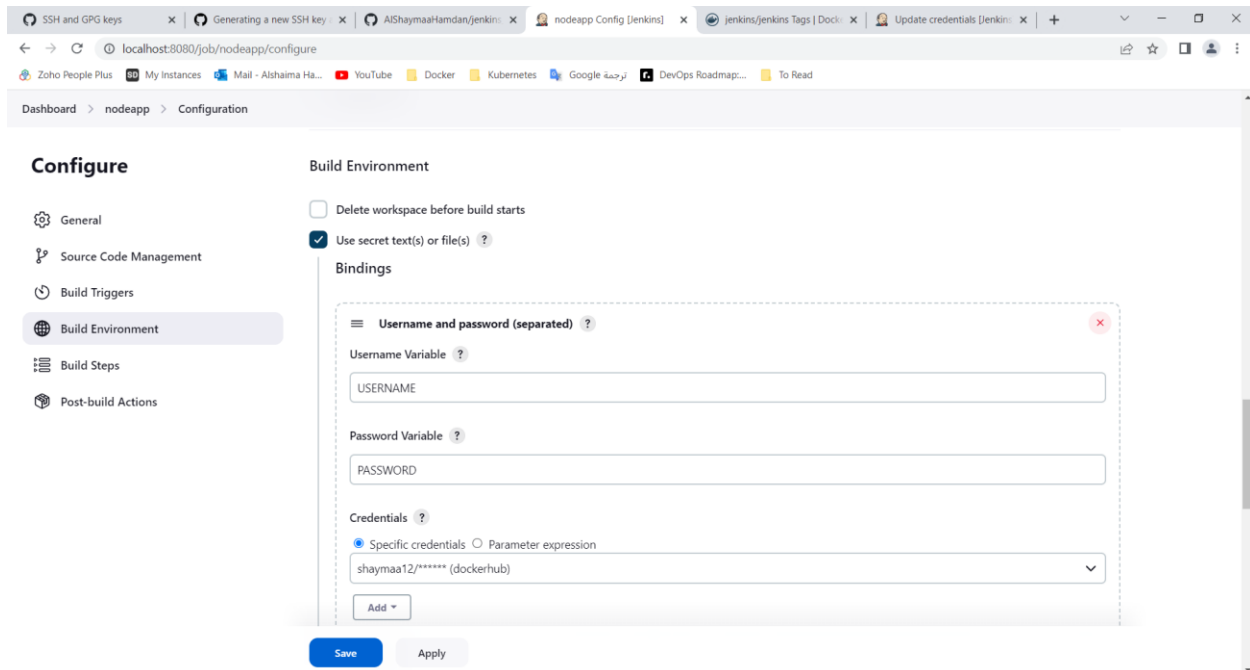
Password ?

ID ?
github1

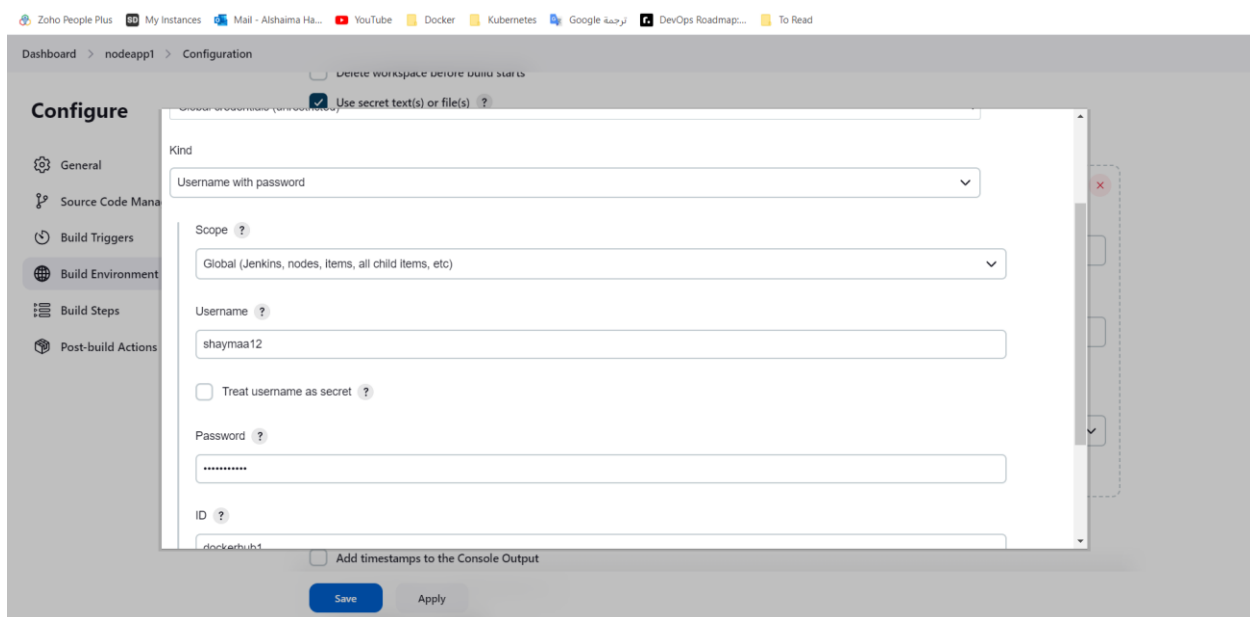
Branch Specifier (blank for 'any') ?

Save Apply

- Then, add your “DockerHub” credentials. Choose “Use secret text(s) or file(s)”

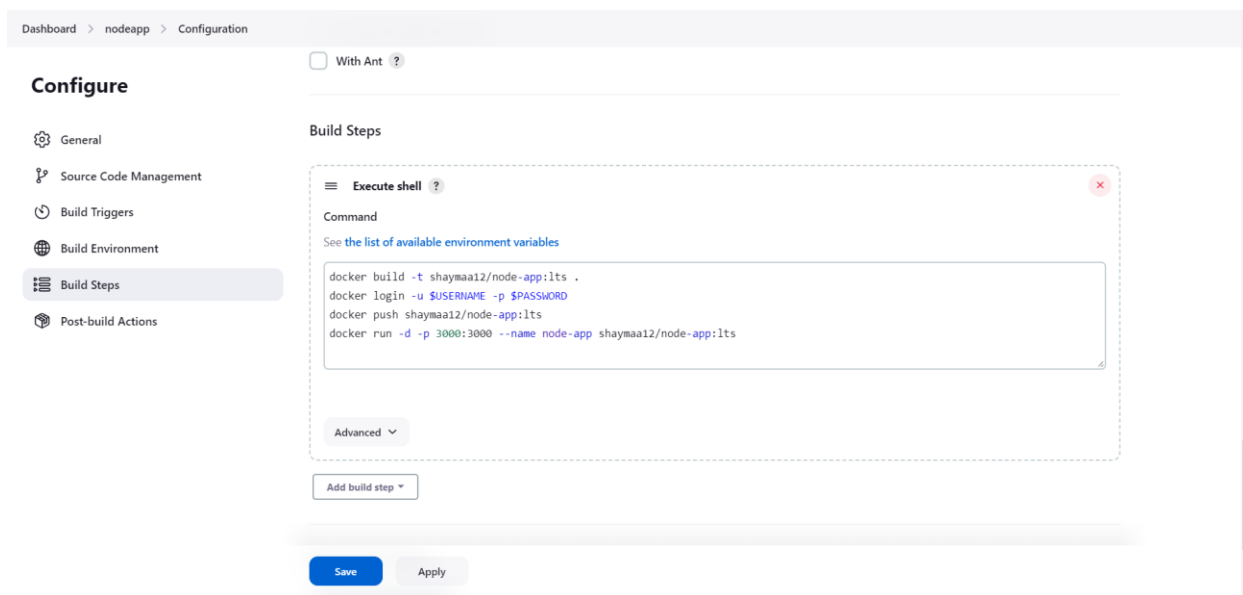


- To create “DockerHub” credentials you should choose “Username and password (separated)”



- Choose “Execute shell” to add build steps

docker build -t shaymaa12/node-app:1ts .	//use Dockerfile to build image named node-app and tag it image with your “DockerHub” account name
docker login -u \$USERNAME -p \$PASSWORD	//login to “DockerHub” with the “Secret” you created before, in order to push it there
docker push shaymaa12/node-app:1ts	//push your image to your “DockerHub” account
docker run -d -p 3000:3000 --name node-app shaymaa12/node-app:1ts	//create a container from your image and expose it to port 3000



- Save your project

Step 6: Build “nodeapp” job

- Click on “Build Now”

The screenshot shows the Jenkins web interface for a project named 'nodeapp'. The top navigation bar includes the Jenkins logo, a search bar, and user controls. The left sidebar contains a list of actions: Status, Changes, Workspace, Build Now, Configure, Delete Project, and Rename. The main content area is titled 'Project nodeapp' and includes a 'Build History' section with a 'trend' dropdown and a search bar. The build history shows three builds: #5 (successful), #4 (failed), and #3 (failed). The 'Permalinks' section provides links to various build details.

Jenkins Search (CTRL+K) [?] [bell] [lock] [1] [user] AA [log out]

Dashboard > nodeapp >

Project nodeapp

Build History trend

Filter builds...

- #5 May 10, 2023, 10:37 AM
- #4 May 10, 2023, 10:36 AM
- #3 May 10, 2023, 10:33 AM

Permalinks

- Last build (#5), 19 hr ago
- Last stable build (#5), 19 hr ago
- Last successful build (#5), 19 hr ago
- Last failed build (#4), 19 hr ago
- Last unsuccessful build (#4), 19 hr ago
- Last completed build (#5), 19 hr ago

Add description

Disable Project

- Choose the build number then Click on “Console Output” to view a detailed log of the build.

Jenkins Search (CTRL+K) AA log out

Dashboard > nodeapp > #5 > Console Output

Status
Changes
Console Output
View as plain text
Edit Build Information
Delete build '#5'
Git Build Data
Previous Build

Console Output

```

Started by user AA
Running as SYSTEM
Building in workspace /var/jenkins_home/workspace/nodeapp
The recommended git tool is: NONE
using credential github
> git rev-parse --resolve-git-dir /var/jenkins_home/workspace/nodeapp/.git # timeout=10
Fetching changes from the remote Git repository
> git config remote.origin.url https://github.com/AlshaymaHamdan/jenkins_nodeapp.git # timeout=10
Fetching upstream changes from https://github.com/AlshaymaHamdan/jenkins_nodeapp.git
> git --version # timeout=10
> git --version # 'git version 2.38.2'
using GIT_CREDSS to set credential(s) github
> git fetch --tags --force --progress -- https://github.com/AlshaymaHamdan/jenkins_nodeapp.git +refs/heads/*:refs/remotes/origin/* # timeout=10
> git rev-parse refs/remotes/origin/master^{commit} # timeout=10
Checking out Revision 12b382db412e366d73868207bd81db9fd579539 (refs/remotes/origin/master)
> git config core.sparsecheckout # timeout=10
> git checkout -f 12b382db412e366d73868207bd81db9fd579539 # timeout=10
Commit message: "-"
> git rev-list --no-walk 12b382db412e366d73868207bd81db9fd579539 # timeout=10
[nodeapp] $ /bin/sh -xe /tmp/jenkins9009215744406780623.sh
+ docker build -t node-app .
#1 [internal] load build definition from dockerfile
#1 transferring dockerfile: 31B 0.0s done
#1 DONE 0.0s

#2 [internal] load .dockerignore
#2 transferring context: 2B 0.0s done
#2 DONE 0.0s

#3 [internal] load metadata for docker.io/library/node:12
#3 DONE 0.7s

#4 [1/4] FROM docker.io/library/node:12@sha256:01627efeb110b3054a4a1405541ca095c8fca1cb6f2be9479c767a2711879e
#4 DONE 0.0s

#5 [internal] load build context
#5 transferring context: 103B 0.0s done
#5 DONE 0.1s

#6 [2/4] COPY nodeapp /nodeapp
#6 CACHED

#7 [3/4] WORKDIR /nodeapp
#7 CACHED

#8 [4/4] RUN npm install
#8 CACHED

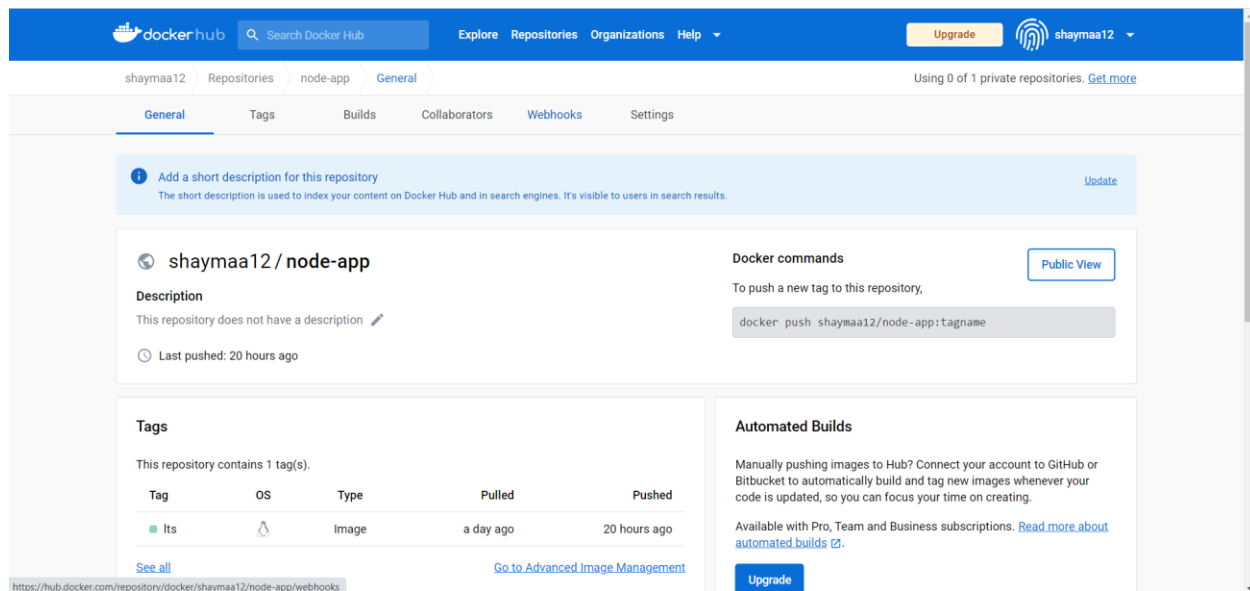
#9 exporting to image
#9 exporting layers done
#9 writing image sha256:4c3d1523953ff963d8536f1cb0e8a106db34ad13a3944591d884aa501509 0.0s done
#9 naming to docker.io/library/node-app done
#9 DONE 0.1s
+ docker login -u shayma12 -p ***
WARNING! Using --password via the CLI is insecure. Use --password-stdin.
WARNING! Your password will be stored unencrypted in /root/.docker/config.json.
Configure a credential helper to remove this warning. See
https://docs.docker.com/engine/reference/commandline/login/#credentials-store

Login Succeeded
+ docker tag node-app shayma12/node-app:its
+ docker push shayma12/node-app:its
The push refers to repository [docker.io/shayma12/node-app]
7ca657950973: Preparing
5f70bf18a086: Preparing
701f6097abbc: Preparing
586c0b41daa7: Preparing
0bf4290f3c17: Preparing
6d75cd81c26c: Preparing
95904c181913: Preparing
df60bf404785: Preparing
f35deb8d09fc: Preparing
f6c2459e2059: Preparing
f8322f3a55c: Preparing
2f4dc9775f33: Preparing
6d75cd81c26c: Layer already exists
95904c181913: Layer already exists
f35deb8d09fc: Layer already exists
f6c2459e2059: Layer already exists
df60bf404785: Layer already exists
f8322f3a55c: Layer already exists
2f4dc9775f33: Layer already exists
its: digest: sha256:1804c1a14f374402e84deef6ab4156a0932fccc3aca15c716f09a079cccd0a30 size: 2839
+ docker run -d -p 3000:3000 --name node-app shayma12/node-app:its
84ae09ba38a-91a0813c9748c9957331eb34048054996295207beca47c8e1e4
Finished: SUCCESS

```

REST API Jenkins 2.387.2

Step 7: Check your “DockerHub” account, a new repository (image) should be added



Step 8: Check if “nodeapp” application is running

- Go to browser and write “localhost:3000”

