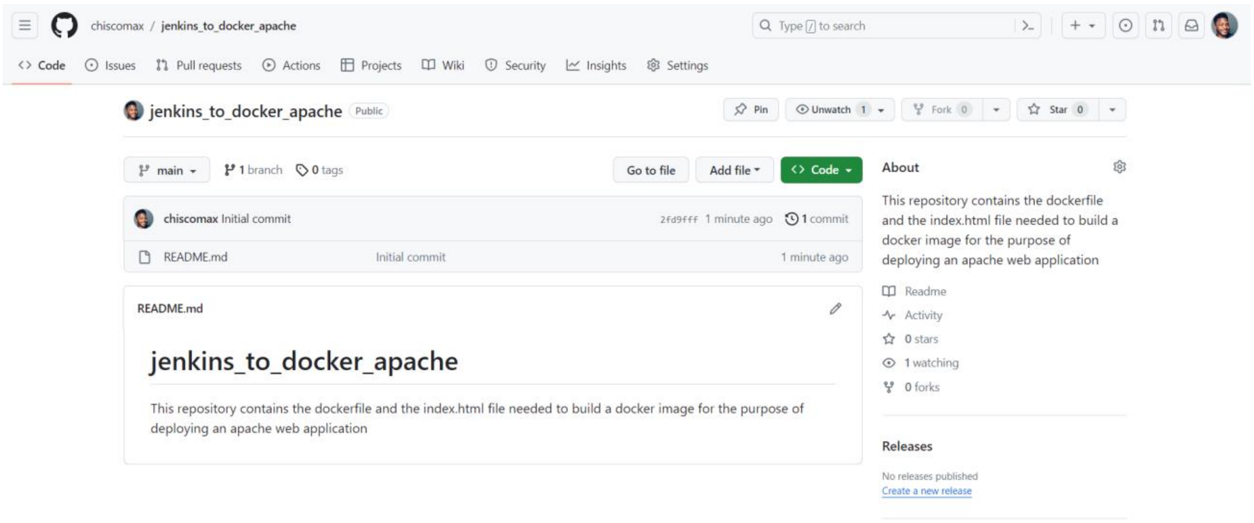
Nnadiekwe, Chiderah David

__Project__- Build a Jenkins pipeline using a plugin to automatically build and publish apache image to Dockerhub.

For this project, I'll be sourcing the code from github. Hence, I'll need a repository containing the dockerfile and the index.html file.

Step 1: Create a github repo containing the index.html file and the dockerfile



chiscomax Initial commit 2fd9fff 1 minute ago 1 commit

README.md Initial commit 1 minute ago

README.md

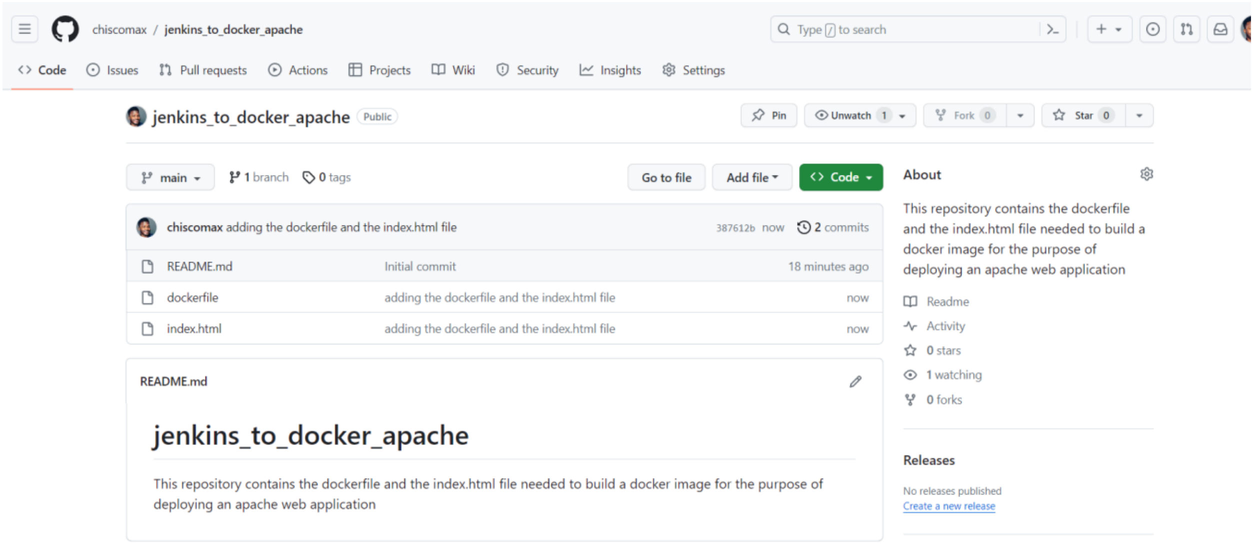
jenkins_to_docker_apache

This repository contains the dockerfile and the index.html file needed to build a docker image for the purpose of deploying an apache web application

Readme
Activity
0 stars
1 watching
0 forks

Releases
No releases published
Create a new release

Next, I added the index file and the dockerfile;



chiscomax adding the dockerfile and the index.html file 387612b now 2 commits

README.md	Initial commit	18 minutes ago
dockerfile	adding the dockerfile and the index.html file	now
index.html	adding the dockerfile and the index.html file	now

README.md

jenkins_to_docker_apache

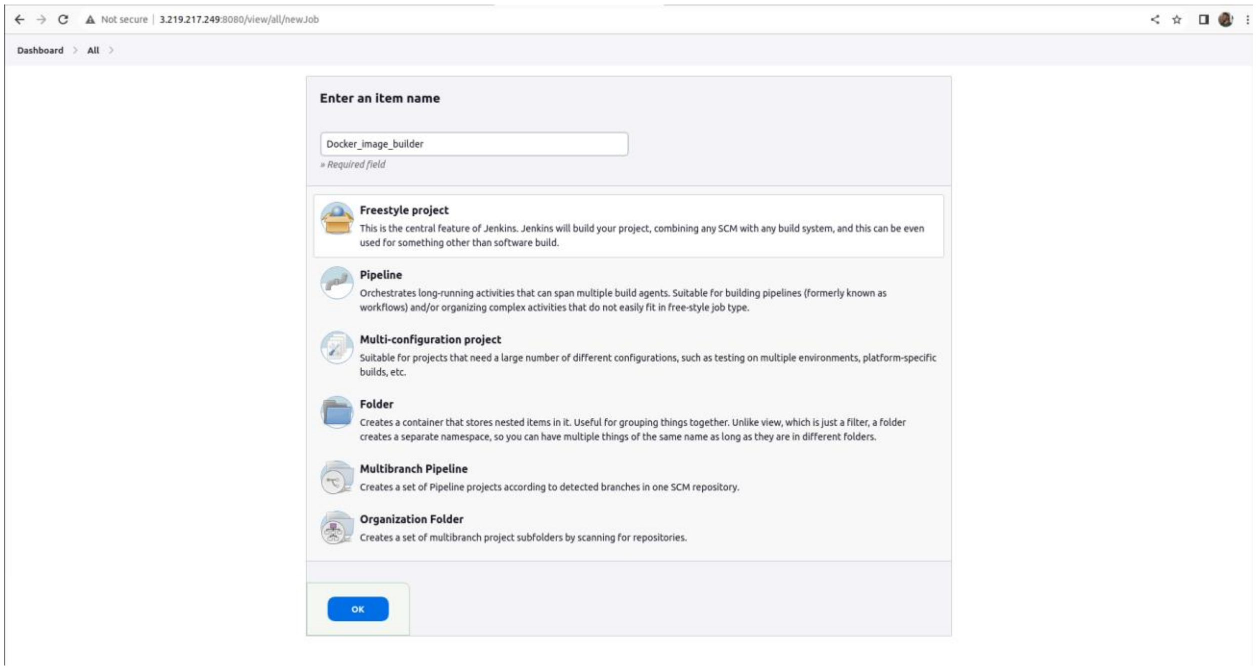
This repository contains the dockerfile and the index.html file needed to build a docker image for the purpose of deploying an apache web application

Readme
Activity
0 stars
1 watching
0 forks

Releases
No releases published
Create a new release

Step 2: Setup git plugin on Jenkins

On jenkins, click on create a new item.



Not secure | 3.219.217.249:8080/view/all/newJob

Dashboard > All >

Enter an item name

Docker_image_builder

» 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.

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.

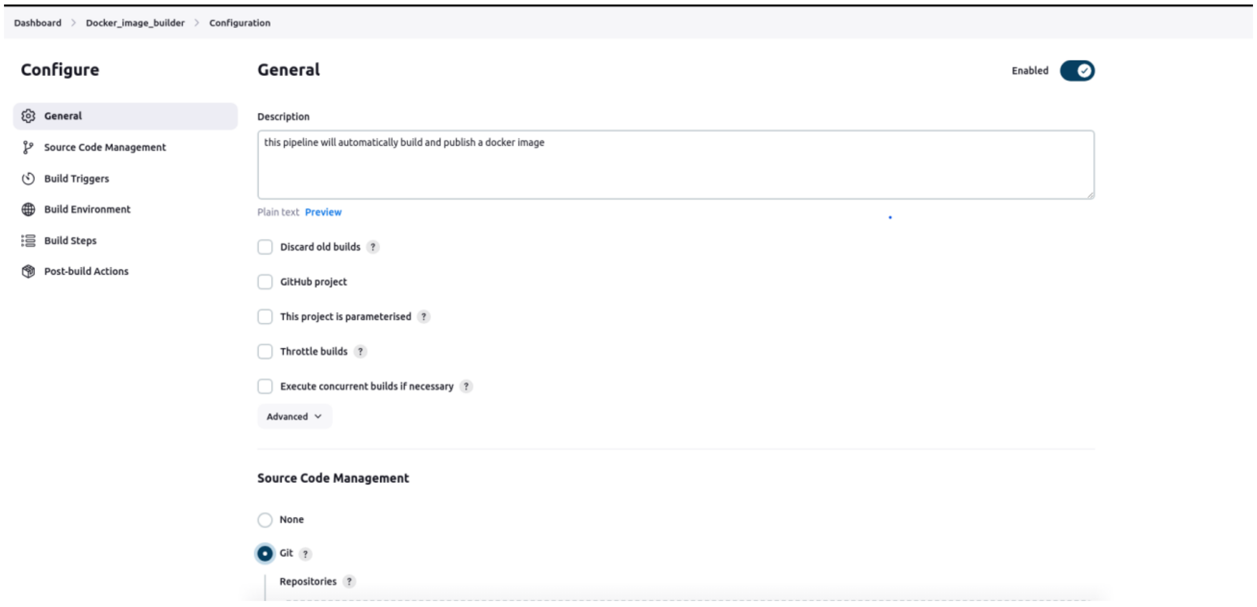
Multi-configuration project
Suitable for projects that need a large number of different configurations, such as testing on multiple environments, platform-specific builds, etc.

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.

OK



Dashboard > Docker_image_builder > Configuration

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

General — Enabled

Description

this pipeline will automatically build and publish a docker image

Plain text Preview

- [ ] Discard old builds
- [ ] GitHub project
- [ ] This project is parameterised
- [ ] Throttle builds
- [ ] Execute concurrent builds if necessary

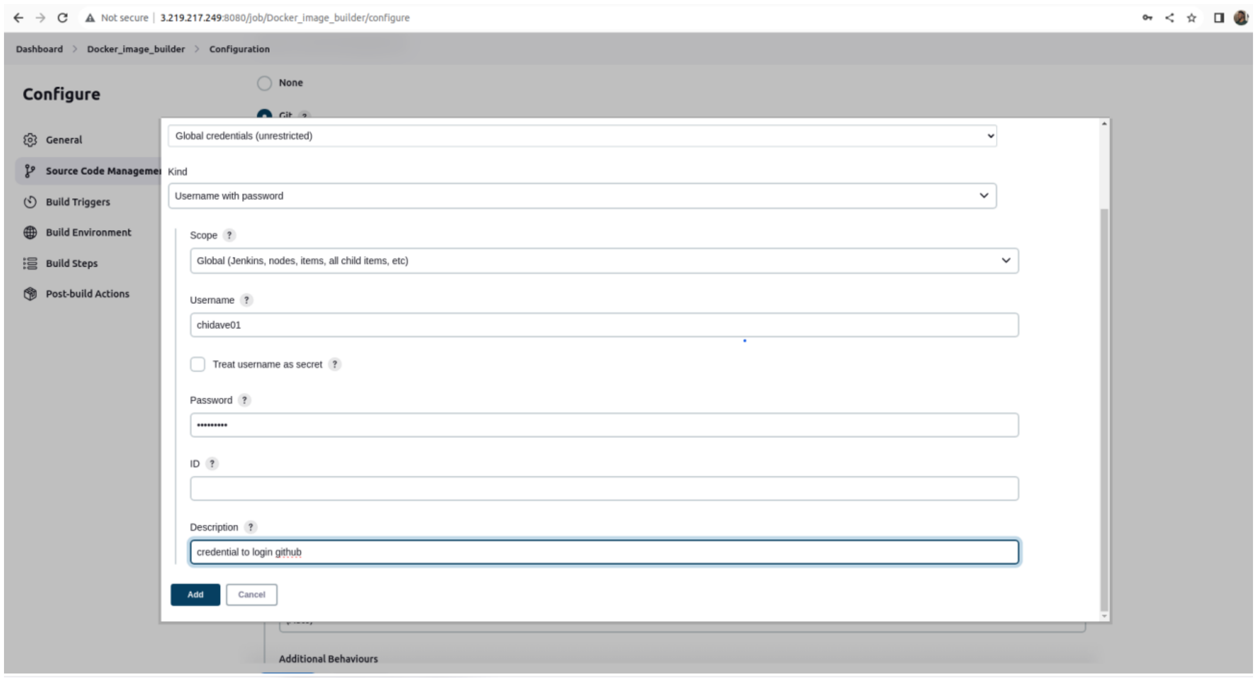
Advanced

Source Code Management

None [✓ Git]

Repositories

Set up credential to access the github repo


Not secure | 3.219.217.249:8080/job/Docker_image_builder/configure

Dashboard > Docker_image_builder > Configuration

Configure

Global credentials (unrestricted)

Kind: Username with password

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

Username: chidave01

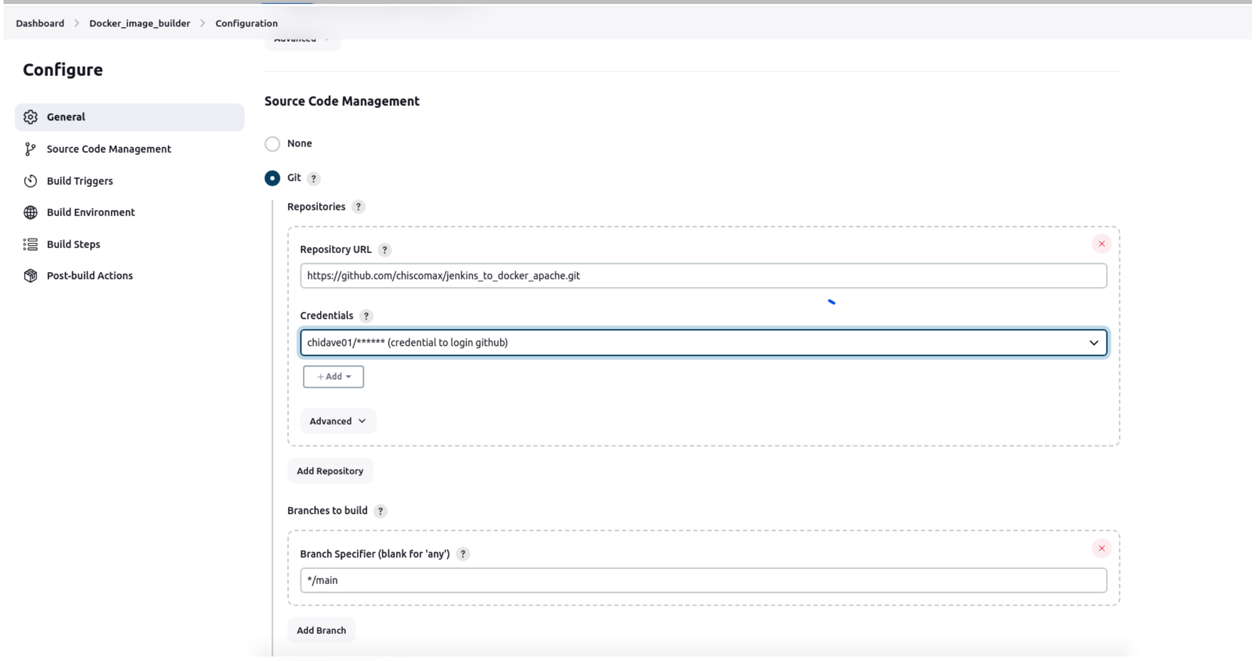
[ ] Treat username as secret

Password: •••••••••

ID:

Description: credential to login github

Add | Cancel


Dashboard > Docker_image_builder > Configuration

Configure

Source Code Management

( ) None

(●) Git

Repositories

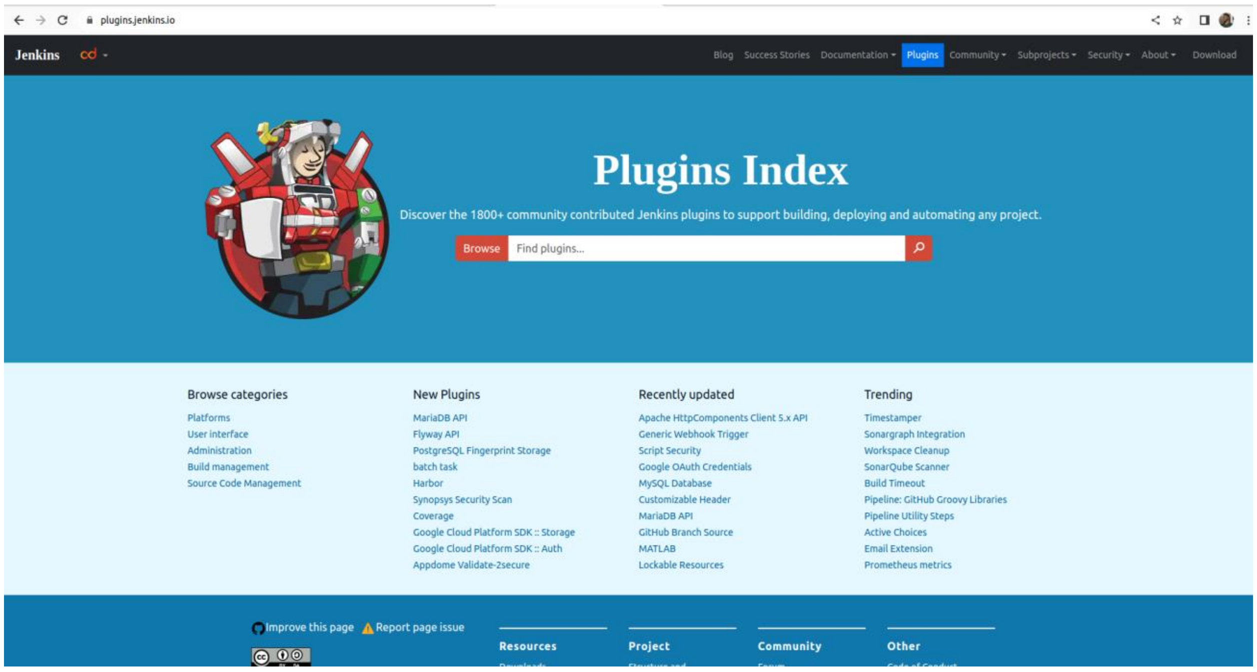
Repository URL: https://github.com/chiscomax/jenkins_to_docker_apache.git

Credentials: chidave01/****** (credential to login github)

Branches to build

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

Next, you need to install the docker plugin on the jenkins server using the plugins link; https://plugins.jenkins.io/


plugins.jenkins.io

Jenkins | Blog | Success Stories | Documentation | Plugins | Community | Subprojects | Security | About | Download

Plugins Index

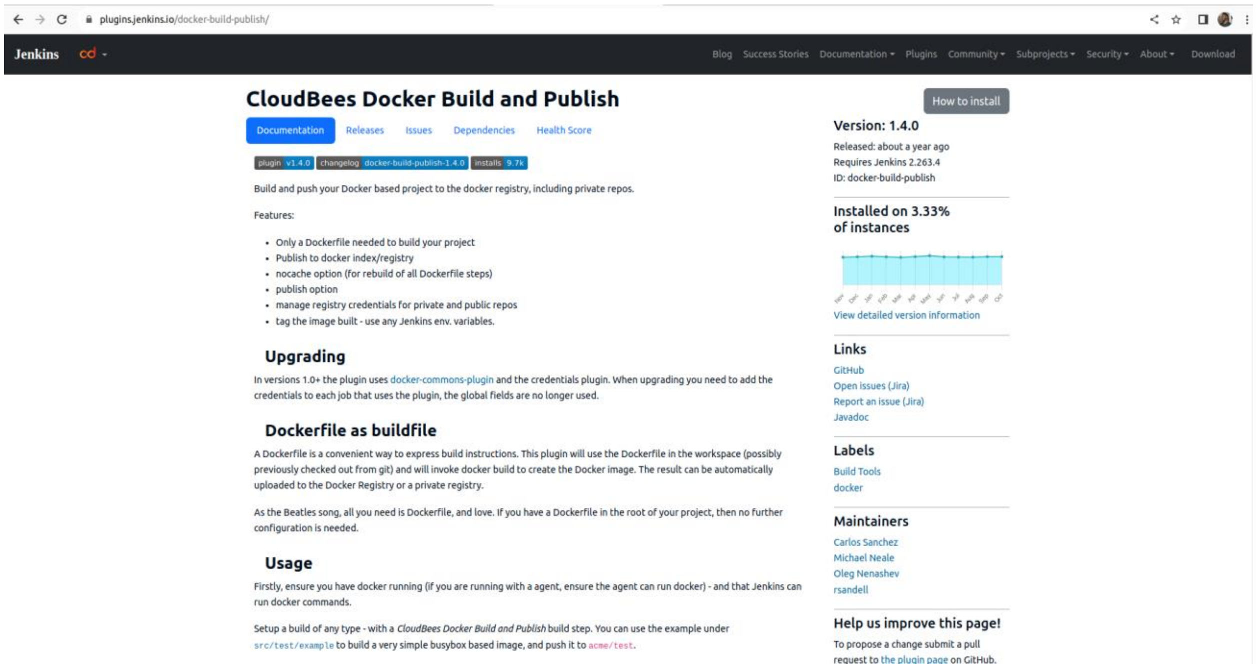
Discover the 1800+ community contributed Jenkins plugins to support building, deploying and automating any project.

Browse | Find plugins...

Browse categories	New Plugins	Recently updated	Trending
Platforms	MariaDB API	Apache HttpComponents Client 5.x API	Timestamper
User interface	Flyway API	Generic Webhook Trigger	Sonargraph Integration
Administration	PostgreSQL Fingerprint Storage	Script Security	Workspace Cleanup
Build management	batch task	Google OAuth Credentials	SonarQube Scanner
Source Code Management	Harbor	MySQL Database	Build Timeout
	Synopsys Security Scan	Customizable Header	Pipeline: GitHub Groovy Libraries
	Coverage	MariaDB API	Pipeline Utility Steps
	Google Cloud Platform SDK :: Storage	GitHub Branch Source	Active Choices
	Google Cloud Platform SDK :: Auth	MATLAB	Email Extension
	Appdome Validate-2secure	Lockable Resources	Prometheus metrics

Improve this page | Report page issue

Resources | Project | Community | Other


plugins.jenkins.io/docker-build-publish/

Jenkins | Blog | Success Stories | Documentation | Plugins | Community | Subprojects | Security | About | Download

CloudBees Docker Build and Publish

Documentation | Releases | Issues | Dependencies | Health Score

plugin v1.4.0 | changelog docker-build-publish-1.4.0 | installs 9.7k

Build and push your Docker based project to the docker registry, including private repos.

Features:

- Only a Dockerfile needed to build your project
- Publish to docker index/registry
- nocache option (for rebuild of all Dockerfile steps)
- publish option
- manage registry credentials for private and public repos
- tag the image built - use any Jenkins env. variables.

Upgrading

In versions 1.0+ the plugin uses docker-commons-plugin and the credentials plugin. When upgrading you need to add the credentials to each job that uses the plugin, the global fields are no longer used.

Dockerfile as buildfile

A Dockerfile is a convenient way to express build instructions. This plugin will use the Dockerfile in the workspace (possibly previously checked out from git) and will invoke docker build to create the Docker image. The result can be automatically uploaded to the Docker Registry or a private registry.

As the Beatles song, all you need is Dockerfile, and love. If you have a Dockerfile in the root of your project, then no further configuration is needed.

Usage

Firstly, ensure you have docker running (if you are running with a agent, ensure the agent can run docker) - and that Jenkins can run docker commands.

Setup a build of any type - with a *CloudBees Docker Build and Publish* build step. You can use the example under `src/test/example` to build a very simple busybox based image, and push it to `acme/test`.

How to install

Version: 1.4.0

Released: about a year ago
Requires Jenkins 2.263.4
ID: docker-build-publish

Installed on 3.33% of instances

Nov Dec Jan Feb Mar Apr May Jun Jul Aug Sep Oct

View detailed version information

Links

GitHub
Open issues (Jira)
Report an issue (Jira)
Javadoc

Labels

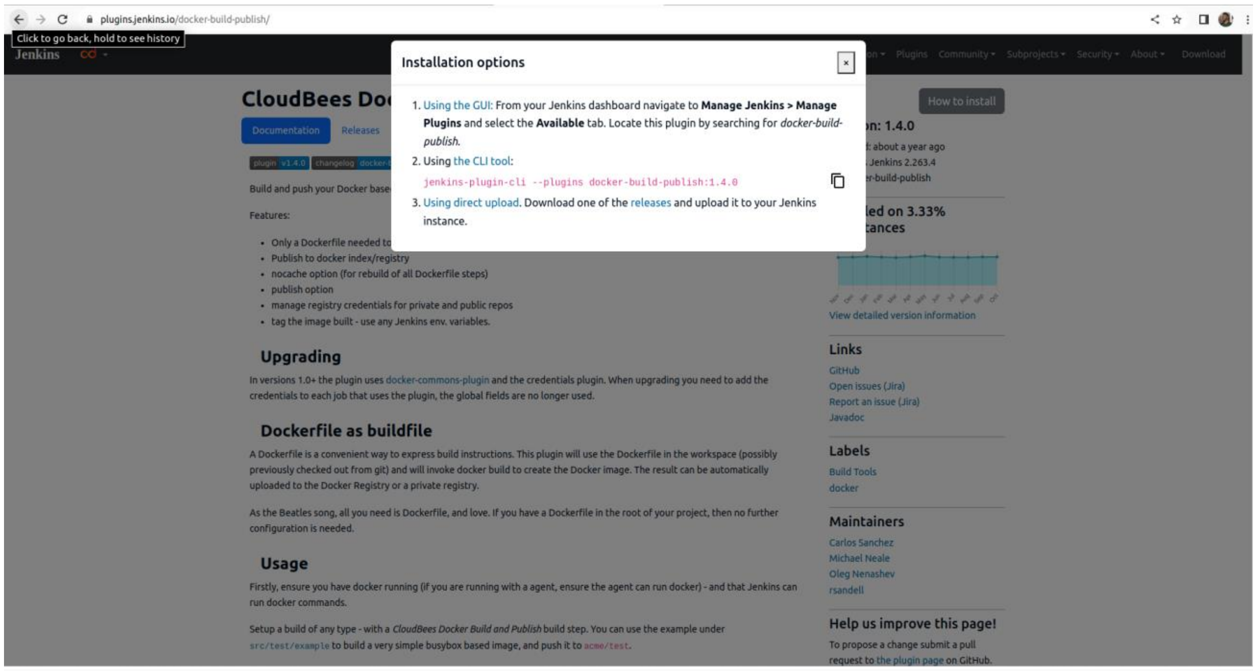
Build Tools
docker

Maintainers

Carlos Sanchez
Michael Neale
Oleg Nenashev
rsandell

Help us improve this page!

To propose a change submit a pull request to the plugin page on GitHub.



Installation options

1. Using the GUI: From your Jenkins dashboard navigate to **Manage Jenkins > Manage Plugins** and select the **Available** tab. Locate this plugin by searching for *docker-build-publish*.
2. Using the CLI tool:

```
jenkins-plugin-cli --plugins docker-build-publish:1.4.0
```

3. Using direct upload. Download one of the releases and upload it to your Jenkins instance.

CloudBees Doc

Build and push your Docker base

Features:

- Only a Dockerfile needed to
- Publish to docker index/registry
- nocache option (for rebuild of all Dockerfile steps)
- publish option
- manage registry credentials for private and public repos
- tag the image built - use any Jenkins env. variables.

Upgrading

In versions 1.0+ the plugin uses docker-commons-plugin and the credentials plugin. When upgrading you need to add the credentials to each job that uses the plugin, the global fields are no longer used.

Dockerfile as buildfile

A Dockerfile is a convenient way to express build instructions. This plugin will use the Dockerfile in the workspace (possibly previously checked out from git) and will invoke docker build to create the Docker image. The result can be automatically uploaded to the Docker Registry or a private registry.

As the Beatles song, all you need is Dockerfile, and love. If you have a Dockerfile in the root of your project, then no further configuration is needed.

Usage

Firstly, ensure you have docker running (if you are running with a agent, ensure the agent can run docker) - and that Jenkins can run docker commands.

Setup a build of any type - with a *CloudBees Docker Build and Publish* build step. You can use the example under `src/test/example` to build a very simple busybox based image, and push it to `acme/test`.

on: 1.4.0
: about a year ago
. Jenkins 2.263.4
er-build-publish

led on 3.33%
tances

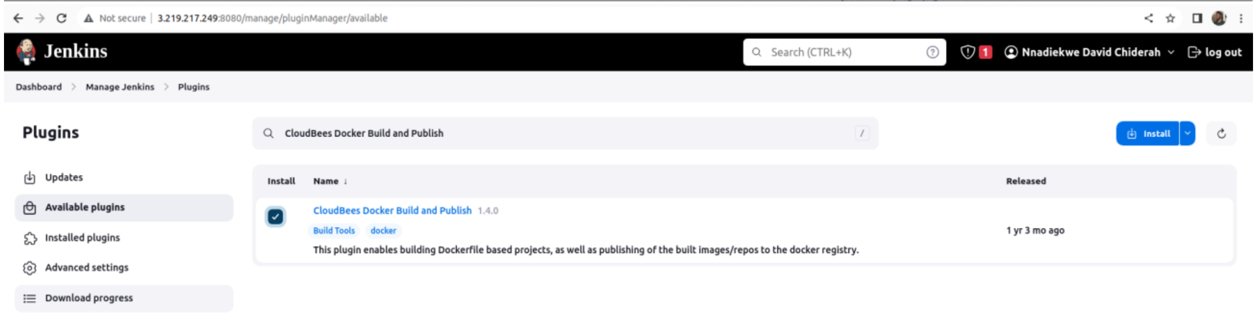
View detailed version information

Links
GitHub
Open issues (Jira)
Report an issue (Jira)
Javadoc

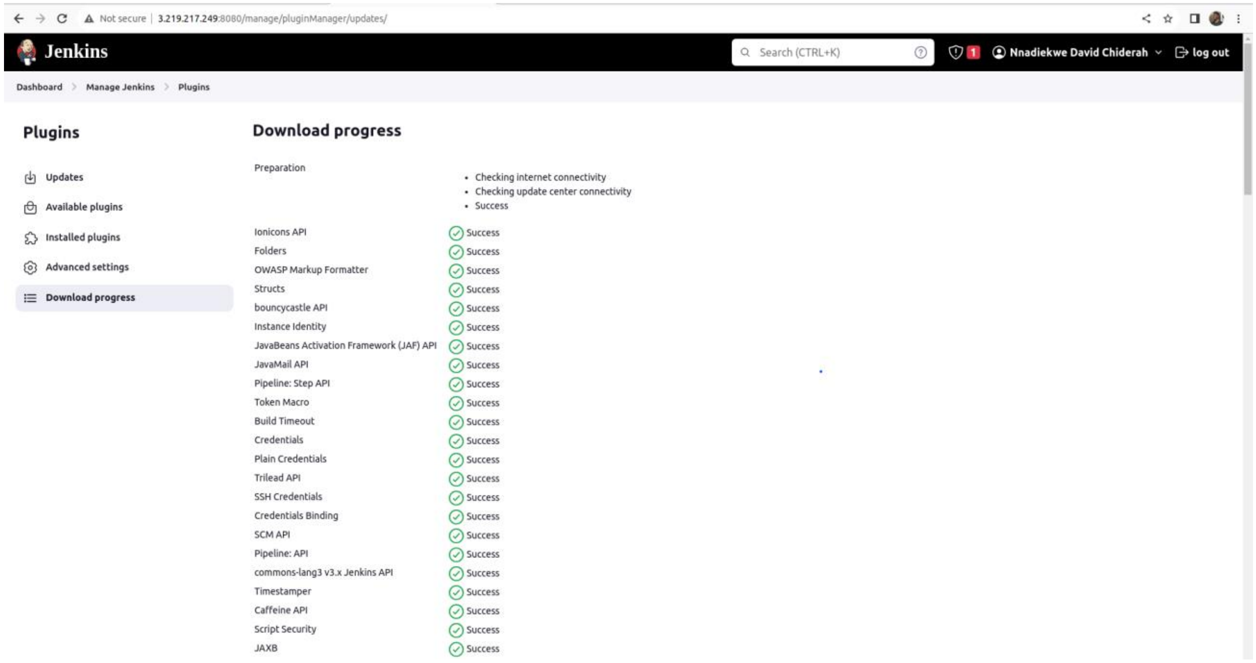
Labels
Build Tools
docker

Maintainers
Carlos Sanchez
Michael Neale
Oleg Nenashev
rsandell

Help us improve this page!
To propose a change submit a pull request to the plugin page on GitHub.



Install	Name ↓	Released
✓	CloudBees Docker Build and Publish 1.4.0 Build Tools docker This plugin enables building Dockerfile based projects, as well as publishing of the built images/repos to the docker registry.	1 yr 3 mo ago

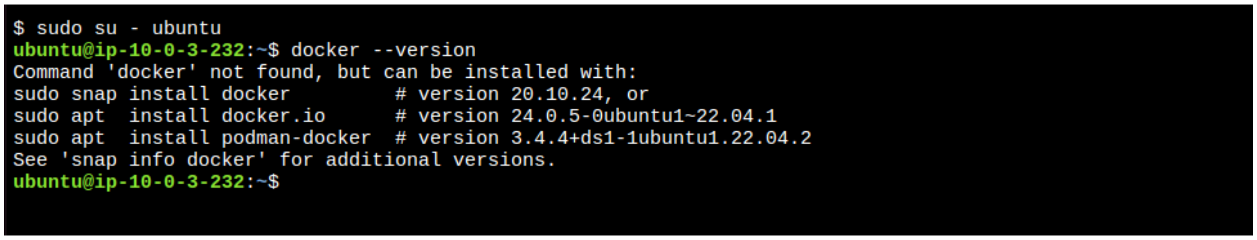


Download progress

Preparation
- Checking internet connectivity
- Checking update center connectivity
- Success

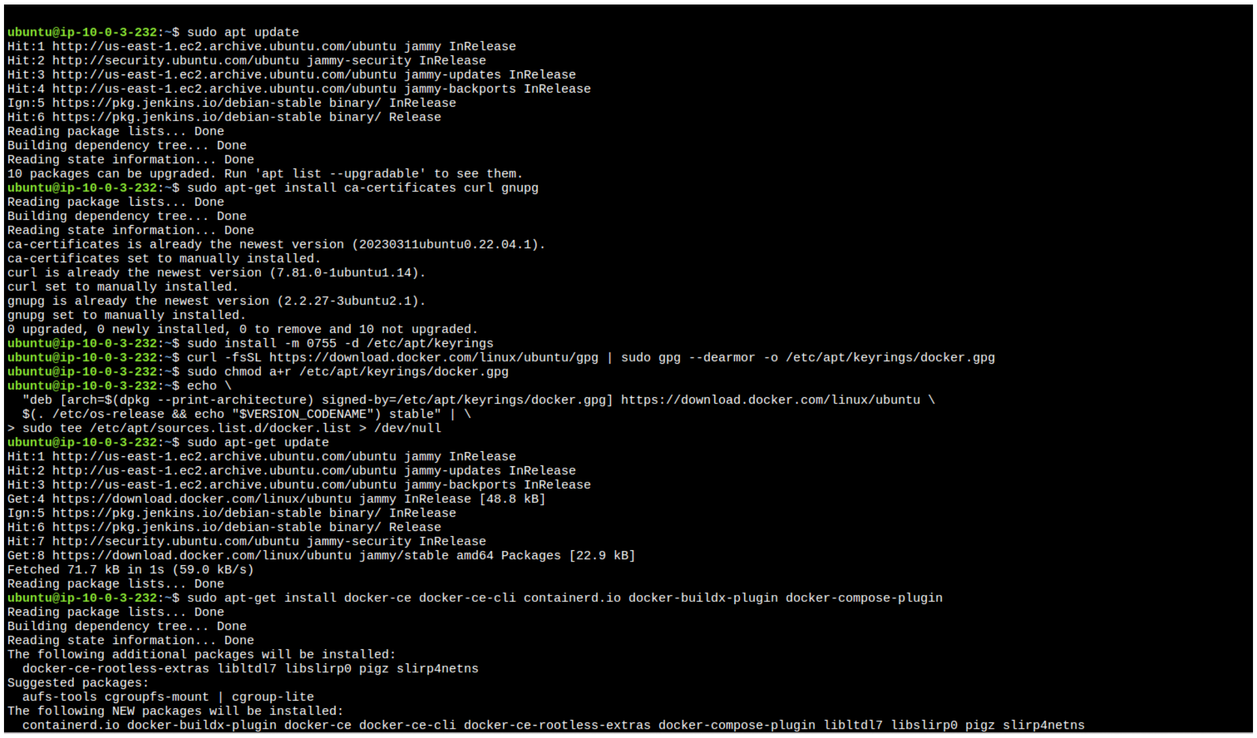
Ionicons API	✓ Success
Folders	✓ Success
OWASP Markup Formatter	✓ Success
Structs	✓ Success
bouncycastle API	✓ Success
Instance Identity	✓ Success
JavaBeans Activation Framework (JAF) API	✓ Success
JavaMail API	✓ Success
Pipeline: Step API	✓ Success
Token Macro	✓ Success
Build Timeout	✓ Success
Credentials	✓ Success
Plain Credentials	✓ Success
Trilead API	✓ Success
SSH Credentials	✓ Success
Credentials Binding	✓ Success
SCM API	✓ Success
Pipeline: API	✓ Success
commons-lang3 v3.x Jenkins API	✓ Success
Timestamper	✓ Success
Caffeine API	✓ Success
Script Security	✓ Success
JAXB	✓ Success

In order to use the plugin, we need to have docker installed in the jenkins server;

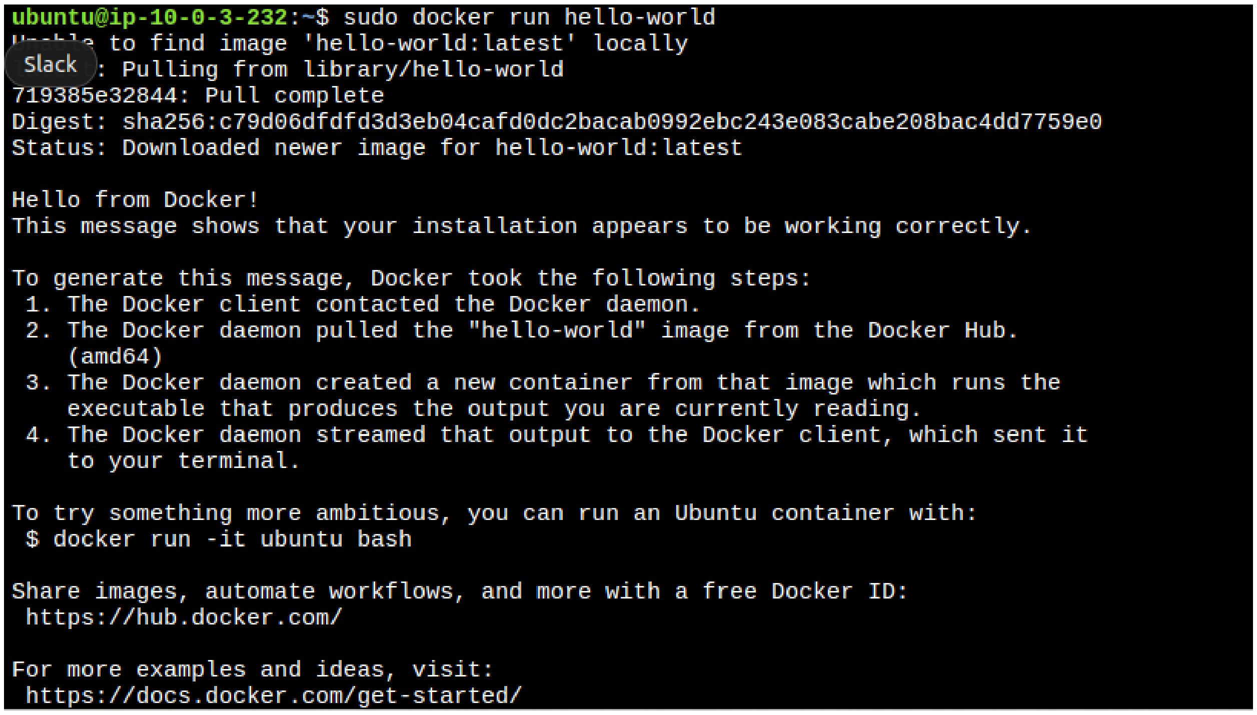


```
$ sudo su - ubuntu
ubuntu@ip-10-0-3-232:~$ docker --version
Command 'docker' not found, but can be installed with:
sudo snap install docker         # version 20.10.24, or
sudo apt  install docker.io      # version 24.0.5-0ubuntu1~22.04.1
sudo apt  install podman-docker  # version 3.4.4+ds1-1ubuntu1.22.04.2
See 'snap info docker' for additional versions.
ubuntu@ip-10-0-3-232:~$
```

As seen above, docker is not yet installed in the server. Hence , we need to install it. We can reference the documentation to install docker on ubuntu server here; https://docs.docker.com/engine/install/ubuntu/

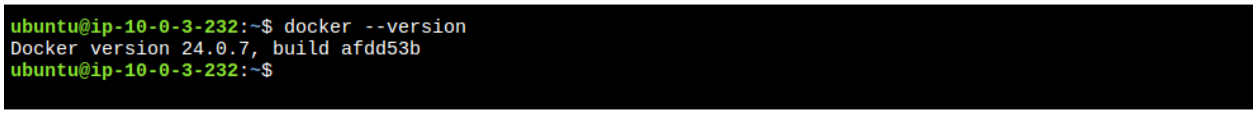


```
ubuntu@ip-10-0-3-232:~$ sudo apt update
Hit:1 http://us-east-1.ec2.archive.ubuntu.com/ubuntu jammy InRelease
Hit:2 http://security.ubuntu.com/ubuntu jammy-security InRelease
Hit:3 http://us-east-1.ec2.archive.ubuntu.com/ubuntu jammy-updates InRelease
Hit:4 http://us-east-1.ec2.archive.ubuntu.com/ubuntu jammy-backports InRelease
Ign:5 https://pkg.jenkins.io/debian-stable binary/ InRelease
Hit:6 https://pkg.jenkins.io/debian-stable binary/ Release
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
10 packages can be upgraded. Run 'apt list --upgradable' to see them.
ubuntu@ip-10-0-3-232:~$ sudo apt-get install ca-certificates curl gnupg
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
ca-certificates is already the newest version (20230311ubuntu0.22.04.1).
ca-certificates set to manually installed.
curl is already the newest version (7.81.0-1ubuntu1.14).
curl set to manually installed.
gnupg is already the newest version (2.2.27-3ubuntu2.1).
gnupg set to manually installed.
0 upgraded, 0 newly installed, 0 to remove and 10 not upgraded.
ubuntu@ip-10-0-3-232:~$ sudo install -m 0755 -d /etc/apt/keyrings
ubuntu@ip-10-0-3-232:~$ curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo gpg --dearmor -o /etc/apt/keyrings/docker.gpg
ubuntu@ip-10-0-3-232:~$ sudo chmod a+r /etc/apt/keyrings/docker.gpg
ubuntu@ip-10-0-3-232:~$ echo \
  "deb [arch=$(dpkg --print-architecture) signed-by=/etc/apt/keyrings/docker.gpg] https://download.docker.com/linux/ubuntu \
  $(. /etc/os-release && echo "$VERSION_CODENAME") stable" | \
> sudo tee /etc/apt/sources.list.d/docker.list > /dev/null
ubuntu@ip-10-0-3-232:~$ sudo apt-get update
Hit:1 http://us-east-1.ec2.archive.ubuntu.com/ubuntu jammy InRelease
Hit:2 http://us-east-1.ec2.archive.ubuntu.com/ubuntu jammy-updates InRelease
Hit:3 http://us-east-1.ec2.archive.ubuntu.com/ubuntu jammy-backports InRelease
Get:4 https://download.docker.com/linux/ubuntu jammy InRelease [48.8 kB]
Ign:5 https://pkg.jenkins.io/debian-stable binary/ InRelease
Hit:6 https://pkg.jenkins.io/debian-stable binary/ Release
Hit:7 http://security.ubuntu.com/ubuntu jammy-security InRelease
Get:8 https://download.docker.com/linux/ubuntu jammy/stable amd64 Packages [22.9 kB]
Fetched 71.7 kB in 1s (59.0 kB/s)
Reading package lists... Done
ubuntu@ip-10-0-3-232:~$ sudo apt-get install docker-ce docker-ce-cli containerd.io docker-buildx-plugin docker-compose-plugin
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following additional packages will be installed:
  docker-ce-rootless-extras libltdl7 libslirp0 pigz slirp4netns
Suggested packages:
  aufs-tools cgroupfs-mount | cgroup-lite
The following NEW packages will be installed:
  containerd.io docker-buildx-plugin docker-ce docker-ce-cli docker-ce-rootless-extras docker-compose-plugin libltdl7 libslirp0 pigz slirp4netns
```



```
ubuntu@ip-10-0-3-232:~$ sudo docker run hello-world
Unable to find image 'hello-world:latest' locally
latest: Pulling from library/hello-world
719385e32844: Pull complete
Digest: sha256:c79d06dfdfd3d3eb04cafd0dc2bacab0992ebc243e083cabe208bac4dd7759e0
Status: Downloaded newer image for hello-world:latest

Hello from Docker!
This message shows that your installation appears to be working correctly.

To generate this message, Docker took the following steps:
 1. The Docker client contacted the Docker daemon.
 2. The Docker daemon pulled the "hello-world" image from the Docker Hub.
    (amd64)
 3. The Docker daemon created a new container from that image which runs the
    executable that produces the output you are currently reading.
 4. The Docker daemon streamed that output to the Docker client, which sent it
    to your terminal.

To try something more ambitious, you can run an Ubuntu container with:
 $ docker run -it ubuntu bash

Share images, automate workflows, and more with a free Docker ID:
 https://hub.docker.com/

For more examples and ideas, visit:
 https://docs.docker.com/get-started/
```

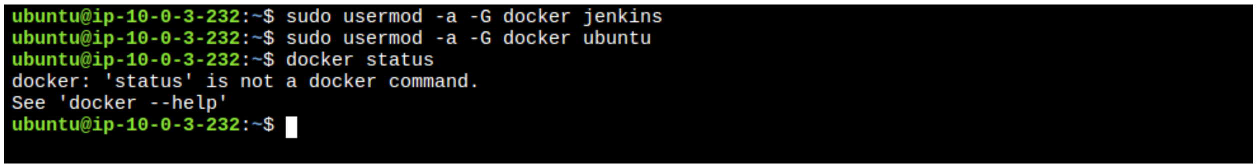
Verifying if docker has been successfully installed;



```
ubuntu@ip-10-0-3-232:~$ docker --version
Docker version 24.0.7, build afdd53b
ubuntu@ip-10-0-3-232:~$
```

Next, we need to add jenkins to the security group to be able to run docker commands:

“sudo usermod -a -G docker jenkins”

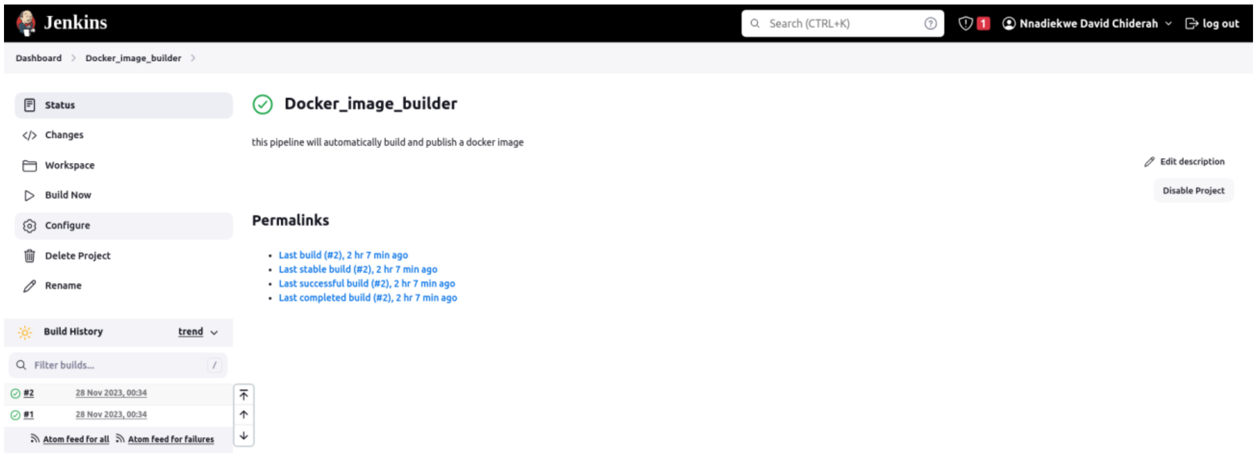


```
ubuntu@ip-10-0-3-232:~$ sudo usermod -a -G docker jenkins
ubuntu@ip-10-0-3-232:~$ sudo usermod -a -G docker ubuntu
ubuntu@ip-10-0-3-232:~$ docker status
docker: 'status' is not a docker command.
See 'docker --help'
ubuntu@ip-10-0-3-232:~$ 
```

Note: Always remember to give permission to jenkins

Step 3: Modify the project

Next, we need to modify the project using the “configure” option to add the build process.



Jenkins | Search (CTRL+K) | Nnadiekwe David Chiderah | log out

Dashboard > Docker_image_builder >

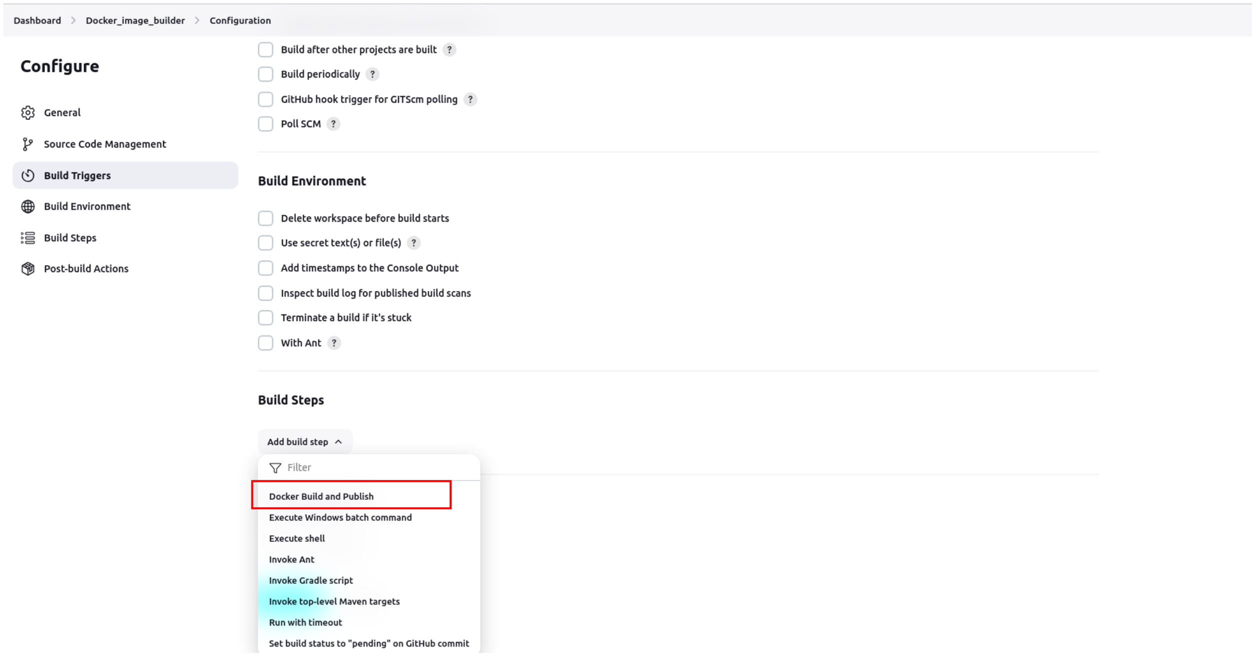
Docker_image_builder

this pipeline will automatically build and publish a docker image

Permalinks

- Last build (#2), 2 hr 7 min ago
- Last stable build (#2), 2 hr 7 min ago
- Last successful build (#2), 2 hr 7 min ago
- Last completed build (#2), 2 hr 7 min ago

#2	28 Nov 2023, 00:34
#1	28 Nov 2023, 00:34



Dashboard > Docker_image_builder > Configuration

- [ ] Build after other projects are built
- [ ] Build periodically
- [ ] GitHub hook trigger for GITScm polling
- [ ] Poll SCM

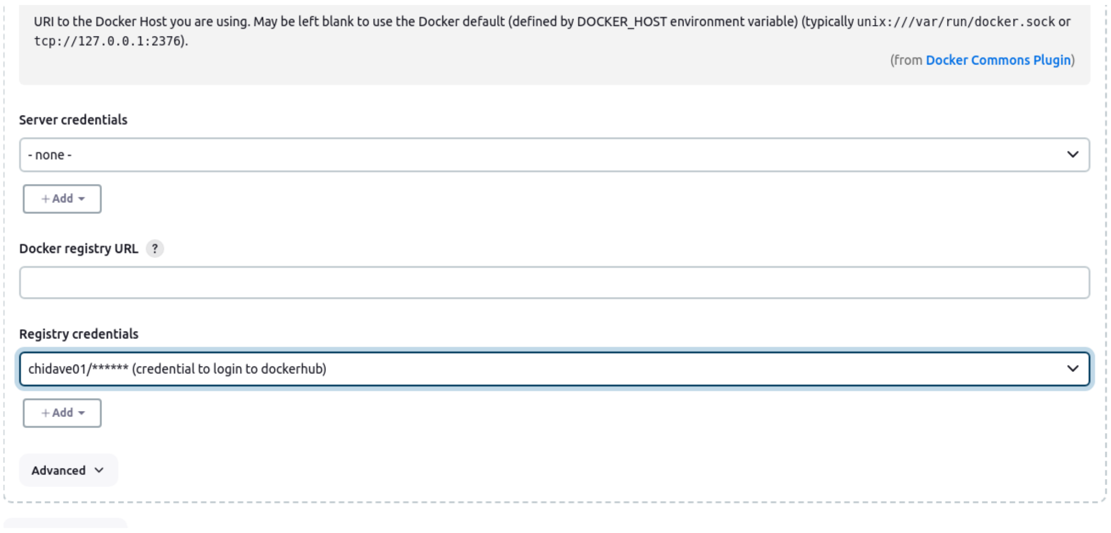
Build Environment

- [ ] Delete workspace before build starts
- [ ] Use secret text(s) or file(s)
- [ ] Add timestamps to the Console Output
- [ ] Inspect build log for published build scans
- [ ] Terminate a build if it's stuck
- [ ] With Ant

Build Steps

Add build step
Filter
Docker Build and Publish
Execute Windows batch command
Execute shell
Invoke Ant
Invoke Gradle script
Invoke top-level Maven targets
Run with timeout
Set build status to "pending" on GitHub commit

Reember to add the dockerhub credentials:


URI to the Docker Host you are using. May be left blank to use the Docker default (defined by DOCKER_HOST environment variable) (typically `unix:///var/run/docker.sock` or `tcp://127.0.0.1:2376`).

(from Docker Commons Plugin)

Server credentials

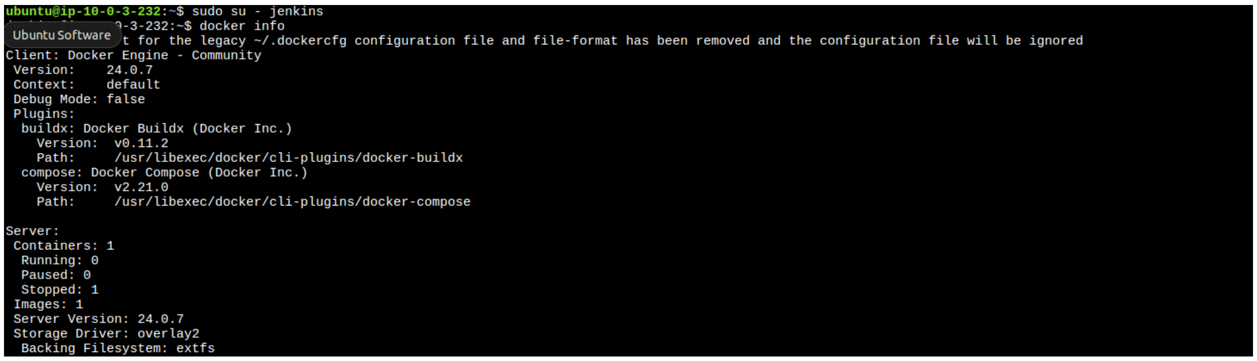
- none -

Docker registry URL ?

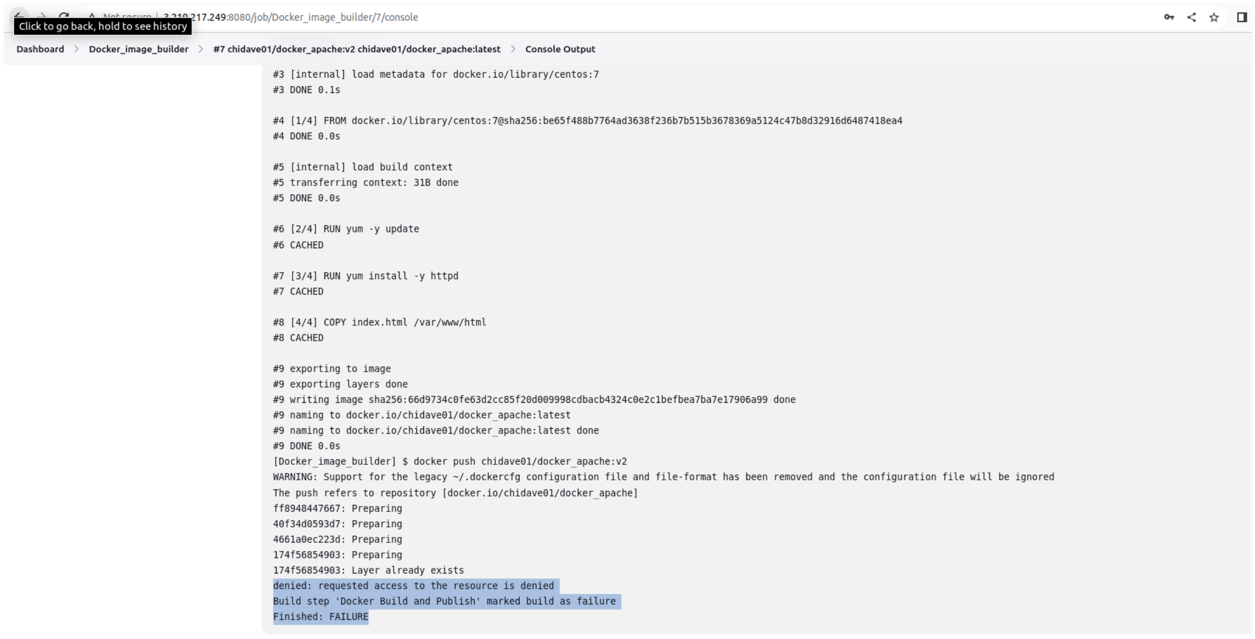
Registry credentials

chidave01/****** (credential to login to dockerhub)

Initially when I tried to run the build, it failed even when the permission has been given to jenkins. I had to use the command “sudo systemctl restart jenkins”. And to verify that the permission has been given to jenkins, we can login the server as jenkins “sudo su – jenkins” and run the command “docker info” to see if we have access to it.

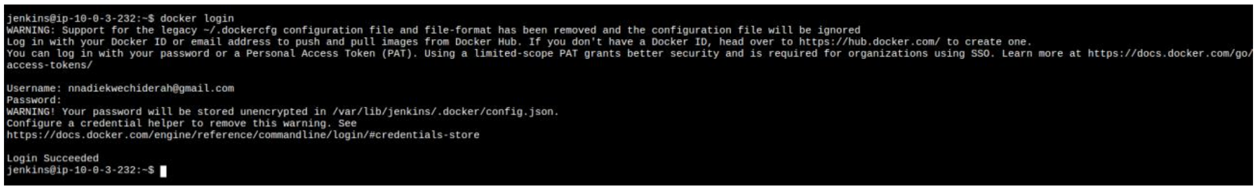

```
ubuntu@ip-10-0-3-232:~$ sudo su - jenkins
                 0-3-232:~$ docker info
Ubuntu Software t for the legacy ~/.dockercfg configuration file and file-format has been removed and the configuration file will be ignored
Client: Docker Engine - Community
 Version:    24.0.7
 Context:    default
 Debug Mode: false
 Plugins:
  buildx: Docker Buildx (Docker Inc.)
    Version:  v0.11.2
    Path:     /usr/libexec/docker/cli-plugins/docker-buildx
  compose: Docker Compose (Docker Inc.)
    Version:  v2.21.0
    Path:     /usr/libexec/docker/cli-plugins/docker-compose

Server:
 Containers: 1
  Running: 0
  Paused: 0
  Stopped: 1
 Images: 1
 Server Version: 24.0.7
 Storage Driver: overlay2
  Backing Filesystem: extfs
```

Even with the permission, the build status still failed.


Not secure | 3.219.217.249:8080/job/Docker_image_builder/7/console
Click to go back, hold to see history
Dashboard > Docker_image_builder > #7 chidave01/docker_apache:v2 chidave01/docker_apache:latest > Console Output

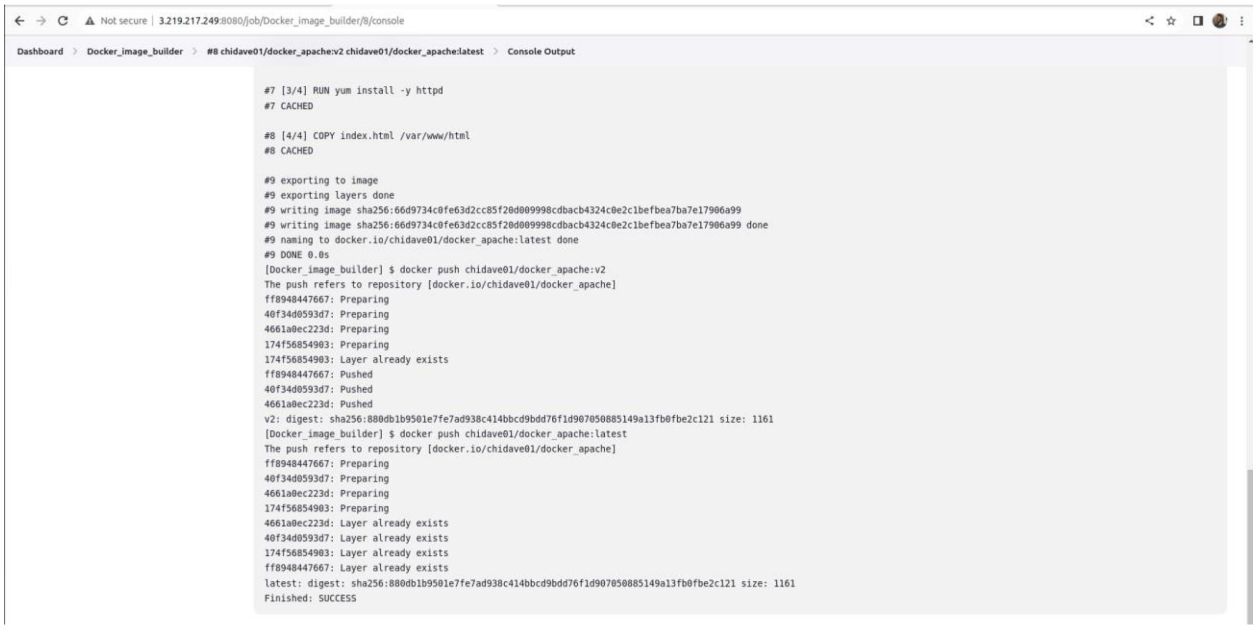
```
#3 [internal] load metadata for docker.io/library/centos:7
#3 DONE 0.1s

#4 [1/4] FROM docker.io/library/centos:7@sha256:be65f488b7764ad3638f236b7b515b3678369a5124c47b8d32916d6487418ea4
#4 DONE 0.0s

#5 [internal] load build context
#5 transferring context: 31B done
#5 DONE 0.0s

#6 [2/4] RUN yum -y update
#6 CACHED

#7 [3/4] RUN yum install -y httpd
#7 CACHED

#8 [4/4] COPY index.html /var/www/html
#8 CACHED

#9 exporting to image
#9 exporting layers done
#9 writing image sha256:66d9734c0fe63d2cc85f20d009998cdbacb4324c0e2c1befbea7ba7e17906a99 done
#9 naming to docker.io/chidave01/docker_apache:latest
#9 naming to docker.io/chidave01/docker_apache:latest done
#9 DONE 0.0s
[Docker_image_builder] $ docker push chidave01/docker_apache:v2
WARNING: Support for the legacy ~/.dockercfg configuration file and file-format has been removed and the configuration file will be ignored
The push refers to repository [docker.io/chidave01/docker_apache]
ff8948447667: Preparing
40f34d0593d7: Preparing
4661a0ec223d: Preparing
174f56854903: Preparing
174f56854903: Layer already exists
denied: requested access to the resource is denied
Build step 'Docker Build and Publish' marked build as failure
Finished: FAILURE
```

To solve this issue, I logged in manually into docker in the server as jenkins user using the command “docker login”



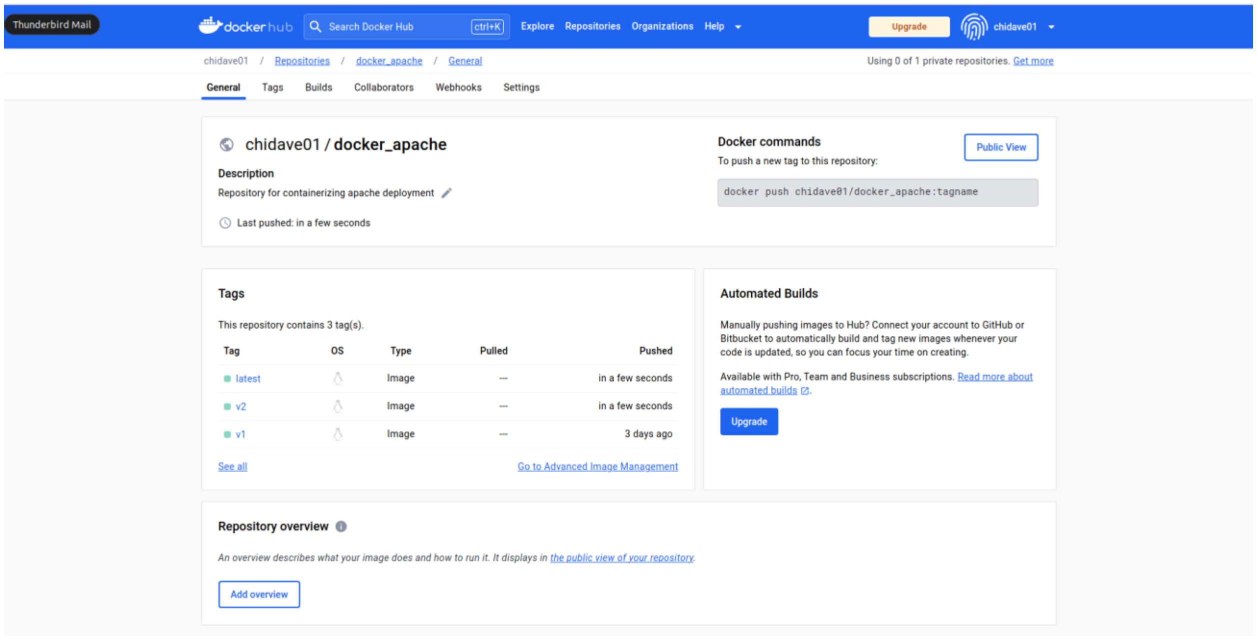
```
jenkins@ip-10-0-3-232:~$ docker login
WARNING: Support for the legacy ~/.dockercfg configuration file and file-format has been removed and the configuration file will be ignored
Log in with your Docker ID or email address to push and pull images from Docker Hub. If you don't have a Docker ID, head over to https://hub.docker.com/ to create one.
You can log in with your password or a Personal Access Token (PAT). Using a limited-scope PAT grants better security and is required for organizations using SSO. Learn more at https://docs.docker.com/go/
access-tokens/

Username: nnadiekwechiderah@gmail.com
Password:
WARNING! Your password will be stored unencrypted in /var/lib/jenkins/.docker/config.json.
Configure a credential helper to remove this warning. See
https://docs.docker.com/engine/reference/commandline/login/#credentials-store

Login Succeeded
jenkins@ip-10-0-3-232:~$
```

Now, I’ll retry the build process in Jenkins


Not secure | 3.219.217.249:8080/job/Docker_image_builder/8/console
Dashboard > Docker_image_builder > #8 chidave01/docker_apache:v2 chidave01/docker_apache:latest > Console Output

```
#7 [3/4] RUN yum install -y httpd
#7 CACHED

#8 [4/4] COPY index.html /var/www/html
#8 CACHED

#9 exporting to image
#9 exporting layers done
#9 writing image sha256:66d9734c0fe63d2cc85f20d009998cdbacb4324c0e2c1befbea7ba7e17906a99
#9 writing image sha256:66d9734c0fe63d2cc85f20d009998cdbacb4324c0e2c1befbea7ba7e17906a99 done
#9 naming to docker.io/chidave01/docker_apache:latest done
#9 DONE 0.0s
[Docker_image_builder] $ docker push chidave01/docker_apache:v2
The push refers to repository [docker.io/chidave01/docker_apache]
ff8948447667: Preparing
40f34d0593d7: Preparing
4661a0ec223d: Preparing
174f56854903: Preparing
174f56854903: Layer already exists
ff8948447667: Pushed
40f34d0593d7: Pushed
4661a0ec223d: Pushed
v2: digest: sha256:880db1b9501e7fe7ad938c414bbcd9bdd76f1d907050885149a13fb0fbe2c121 size: 1161
[Docker_image_builder] $ docker push chidave01/docker_apache:latest
The push refers to repository [docker.io/chidave01/docker_apache]
ff8948447667: Preparing
40f34d0593d7: Preparing
4661a0ec223d: Preparing
174f56854903: Preparing
4661a0ec223d: Layer already exists
40f34d0593d7: Layer already exists
174f56854903: Layer already exists
ff8948447667: Layer already exists
latest: digest: sha256:880db1b9501e7fe7ad938c414bbcd9bdd76f1d907050885149a13fb0fbe2c121 size: 1161
Finished: SUCCESS
```

As seen above, the build and push process is now successful.



Thunderbird Mail
docker hub
Search Docker Hub ctrl+K
Explore Repositories Organizations Help
Upgrade
chidave01
chidave01 / Repositories / docker_apache / General
Using 0 of 1 private repositories. Get more
General Tags Builds Collaborators Webhooks Settings
Public View
See all
Go to Advanced Image Management
Upgrade
Add overview

chidave01 / **docker_apache**

Description

Repository for containerizing apache deployment

Last pushed: in a few seconds

Docker commands

To push a new tag to this repository:

```
docker push chidave01/docker_apache:tagname
```

Tags

This repository contains 3 tag(s).

Tag	OS	Type	Pulled	Pushed
latest		Image	---	in a few seconds
v2		Image	---	in a few seconds
v1		Image	---	3 days ago

Automated Builds

Manually pushing images to Hub? Connect your account to GitHub or Bitbucket to automatically build and tag new images whenever your code is updated, so you can focus your time on creating.

Available with Pro, Team and Business subscriptions. Read more about automated builds.

Repository overview

An overview describes what your image does and how to run it. It displays in the public view of your repository.

To verify the success of the process, we can now login to dockerhub to verify.