Lab 12. Build a Docker Image with DevCS & Push it to OCIR

Objectives:

- Download and use sample source code for the Building and Deployment.
- Upload the source to Oracle DevCS and build container image using DevCS builder instance. Push this image to OCIR from DevCS

Pre-Requisite

- A web browser – Google chrome or Microsoft Edge
- An **Oracle Cloud Account** with an active Oracle Developer Cloud Service subscription. It comes with IDCS. Therefore, you should have Admin/Tenancy level access.
- Your Oracle Cloud account credentials
- Git

Sequence 1. Download a sample source code in your local VM, *server*.

1. Create a directory and pull sample app by cloning a Github repo for testing

   # mkdir ~/DockerFlash

   # cd ~/DockerFlash

   # git clone https://github.com/dockersamples/node-bulletin-board

2. Change to the app directory

   # cd node-bulletin-board/bulletin-board-app

   # mv * ~/DockerFlash

   # cd ~/DockerFlash

3. Check the contents of Dockerfile in the bulletin board application.
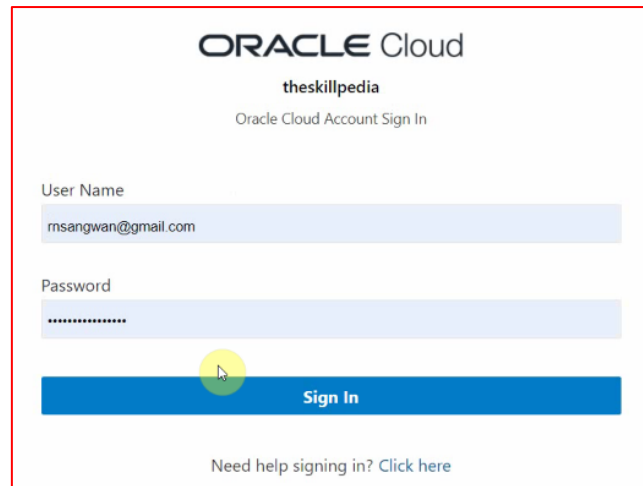
   # ls

   # cat Dockerfile

```
FROM node:current-slim

WORKDIR /usr/src/app
COPY package.json .
RUN npm install

EXPOSE 8080
CMD [ "npm", "start" ]

COPY . .
```

Sequence 2. Login to Oracle DevCS and setup your Build Job

1. Login to Oracle Cloud using IDCS (not regular OCI account)
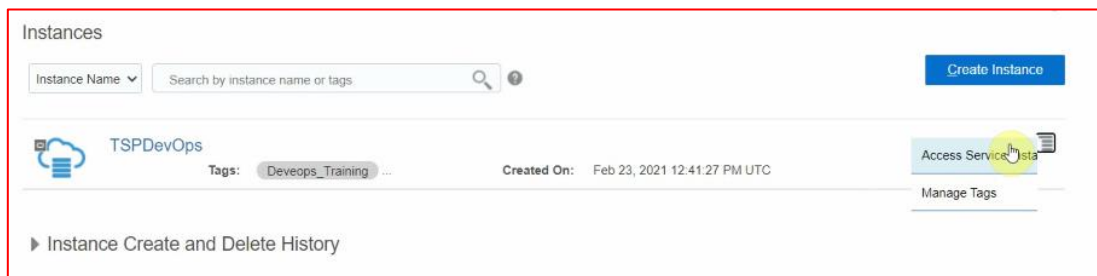
2. Click on "Access Service Instance" from menu given on right side as given I screen shot.

3. Click on the Project you have created earlier. If you have not created a project, create a new project.
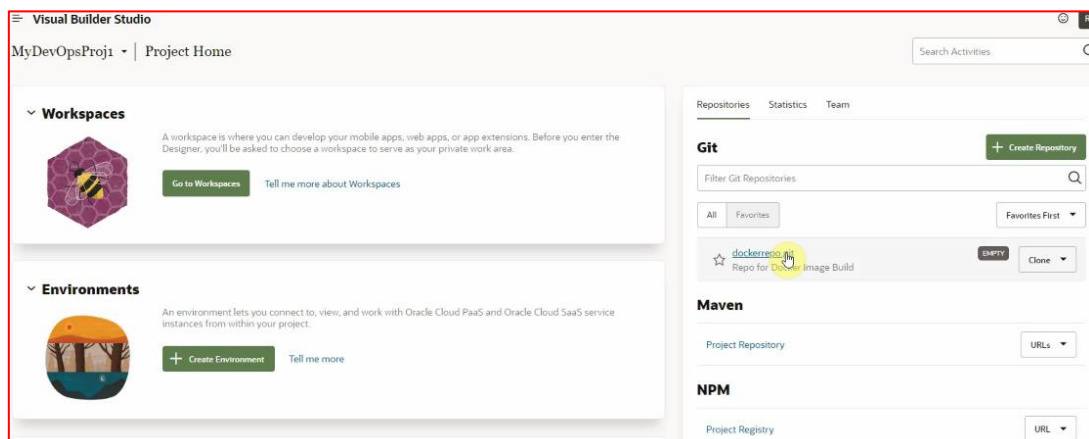
4. Create a repository by clicking on the link give on the right side. If you already have a repo, open it by clicking on its name.

5. Copy the '***git clone ..***" command from the Basic Setup information box.

6. Paste this command in the terminal window of your Local linux VM.

```
[root@server ~]# git clone https://rnsangwan%40gmail.com@tspdevops-theskillped
Cloning into 'dockerrepo'...
Password for 'https://rnsangwan@gmail.com@tspdevops-theskillpedia.developer.od
warning: remote HEAD refers to nonexistent ref, unable to checkout.

[root@server ~]#
```

7. Switch to your DevCS repo directory you cloned and copy the content of the App source directory

```
[root@server ~]# cd dockerrepo/
[root@server dockerrepo]# ll
total 0
[root@server dockerrepo]# cp -r ../DockerFlash/* ./
[root@server dockerrepo]# ll
total 36
-rw-r--r-- 1 root root 1239 Feb 23 10:39 app.js
drwxr-xr-x 2 root root   53 Feb 23 10:39 backend
-rw-r--r-- 1 root root  127 Feb 23 10:39 Dockerfile
drwxr-xr-x 3 root root   23 Feb 23 10:39 fonts
-rw-r--r-- 1 root root 1826 Feb 23 10:39 index.html
```

8. Add the App source files to git staging area and commit the changes to local repository.
   # git add .
   # git commit -m "Initial Commit"

```
[root@server dockerrepo]# git add .
[root@server dockerrepo]# git commit -m "Initial Commit"
[master (root-commit) 105377d] Initial Commit
 Committer: root <root@server.example.com>
Your name and email address were configured automatically based
on your username and hostname. Please check that they are accurate.
You can suppress this message by setting them explicitly:

    git config --global user.name "Your Name"
    git config --global user.email you@example.com

After doing this, you may fix the identity used for this commit with:

    git commit --amend --reset-author

13 files changed, 307 insertions(+)
create mode 100644 Dockerfile
```

9. Push the local repo to DevCS repository

    # git push -u origin master

```
[root@server dockerrepo]# git push -u origin master
Password for 'https://rnsangwan@gmail.com@tspdevops-theskillpedia.d
Counting objects: 18, done.
Compressing objects: 100% (16/16), done.
Writing objects: 100% (18/18), 27.69 KiB | 0 bytes/s, done.
Total 18 (delta 0), reused 0 (delta 0)
remote: [Push Options] Do you want to create a merge request? Use g
remote: Updating references: 100% (1/1)
```
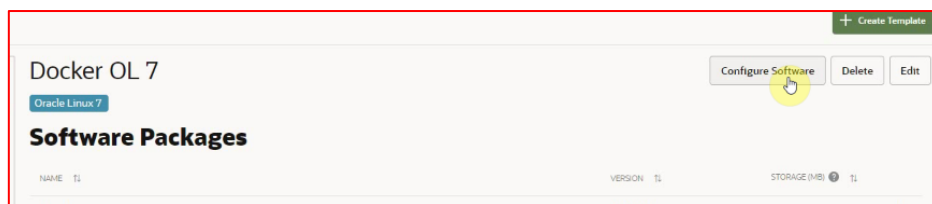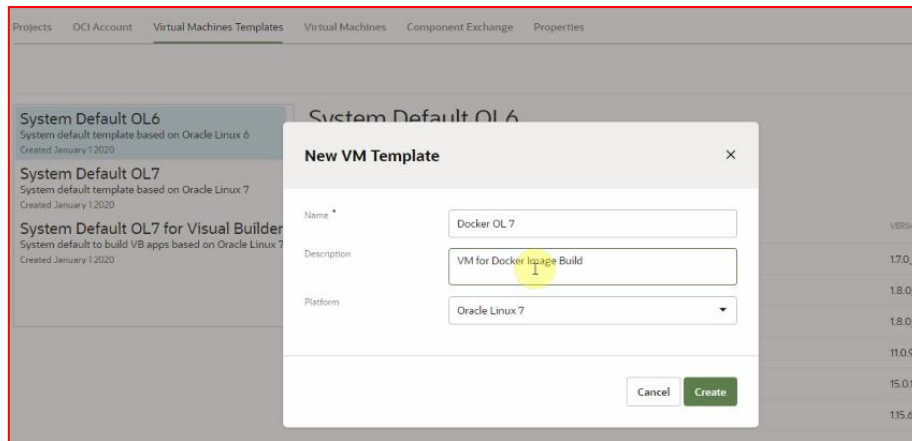
10. Verify the source files are copied
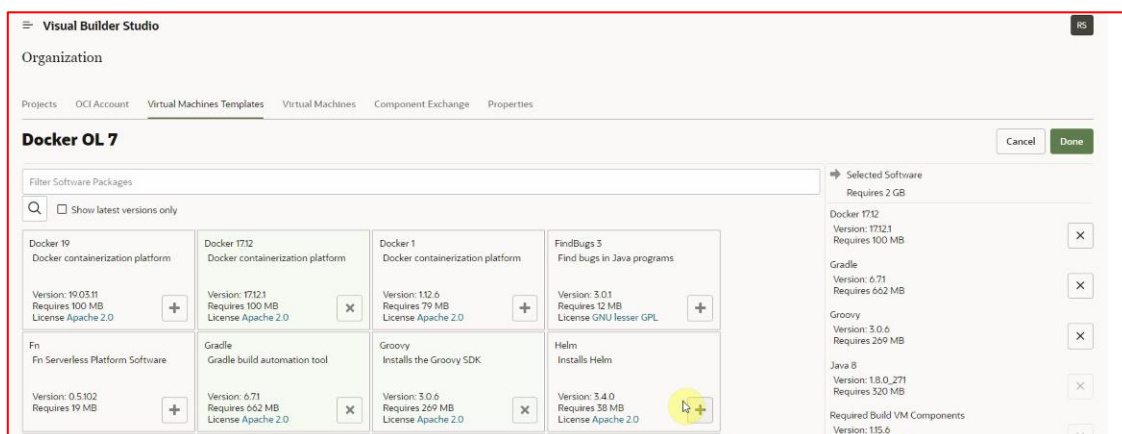
MyDevOpsProj1 ▾ | Git

| dockerrepo.git | ▾ | ⑂ master | ▾ | ⛶ |

Repo for Docker Image Build ✎

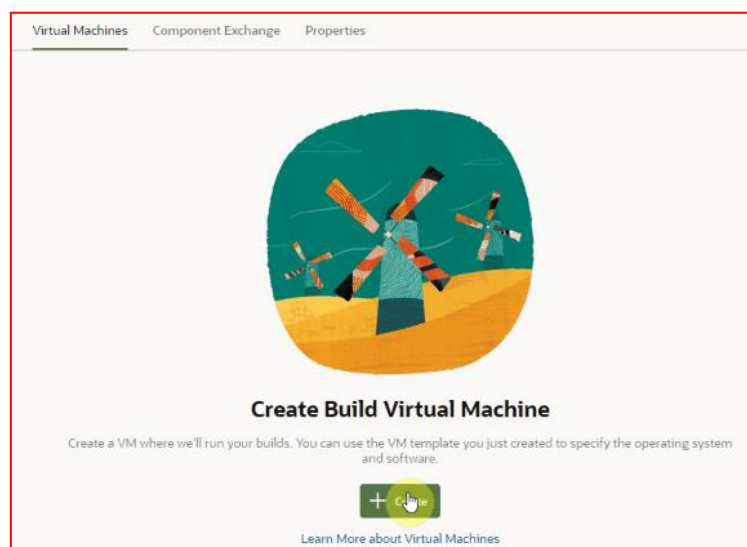| 📁 | backend | Initial Commit \| root |
📁	fonts / geomanist	Initial Commit | root
⬚	app.js	Initial Commit | root
⬚	Dockerfile	Initial Commit | root
⬚	index.html	Initial Commit | root

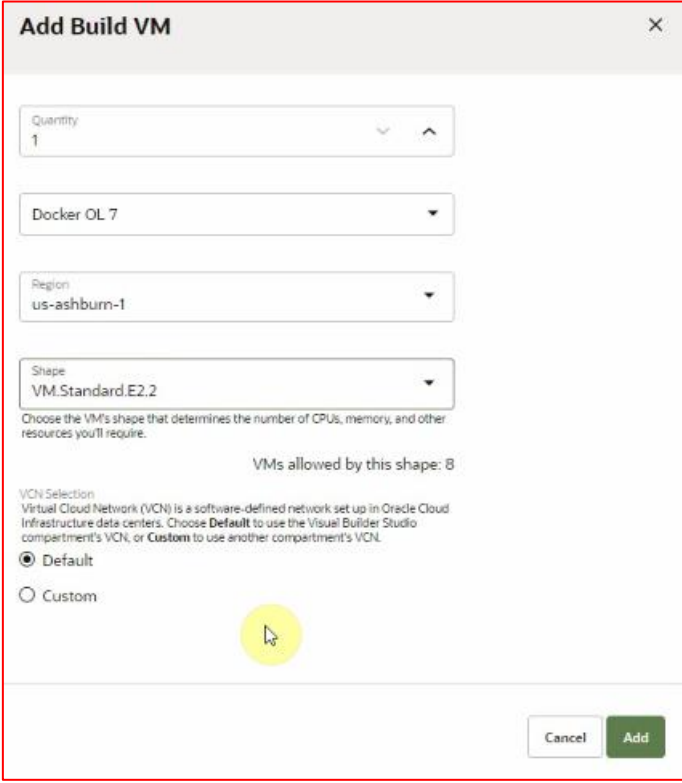11. Create a Virtual Machine template to be used by your build job

12. Select Docker and other relevant packages for the template.

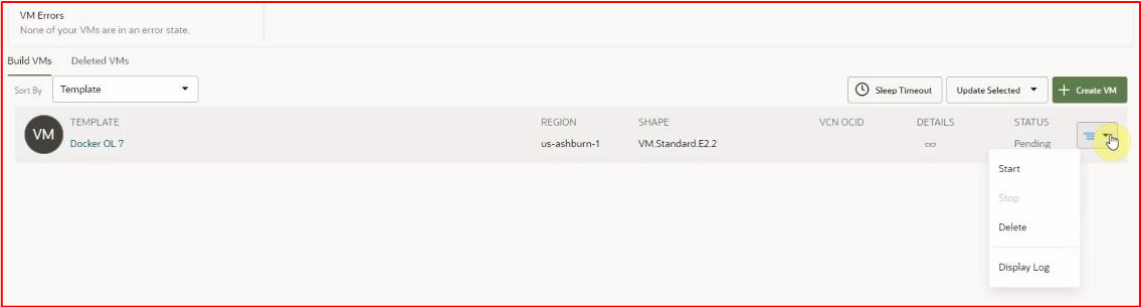13. Click on Virtual Machines Tab and Click on Create Build Virtual Machine

14. Select the Template created above and Shape for the VM

15. Once the VM is created, start it. It will take approximately 20 minutes to come up.

16. Now click on Build option from the left side bar and select Create Job
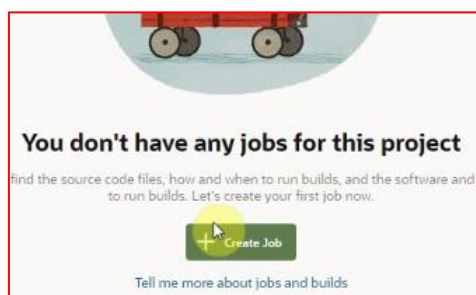
17. Specify the Build Job Details. In the Template box select the VM you have created from the template.

18. The Job will appear in the list.

19. Now add the git repository to this job under job configuration.

20. Select the Repo you have created.

Job Configuration

Git   Parameters   Before Build   Steps   After Build

Configure Git

::: Git ⑦

Repository *
dockerrepo.git   ▼

> **Advanced Repository Options**

☑ Automatically perform build on SCM commit

Auto Branch

◉ Include   ○ Exclude

Branches
master

Branch name, * expression, or /regex/, one per line

Exceptions

Branch name, * expression, or /regex/, one per line

Local Checkout Directory

> **Advanced Git Settings**

Job Configuration

Git   Parameters   Before Build   Steps   After Build

Configure Git

::: Git ⑦

Repository *
dockerrepo.git   ▼

> **Advanced Repository Options**

☑ Automatically perform build on SCM commit

Auto Branch

◉ Include   ○ Exclude

Branches
master

Branch name, * expression, or /regex/, one per line

Exceptions

Branch name, * expression, or /regex/, one per line

Local Checkout Directory

> **Advanced Git Settings**

21. Under advanced git Settings, Select Docker Files as "**/*"

22. Add a Job step by select "Steps" tab and then select "Add step" drop down given at the extreme right. Select Docker - > Docker Login step

23. To login using docker, you will need auth token and tenancy namespace along with your user id. For this visit "User settings" page in another tab of your browser and click on Auth Token option given on the left menu.

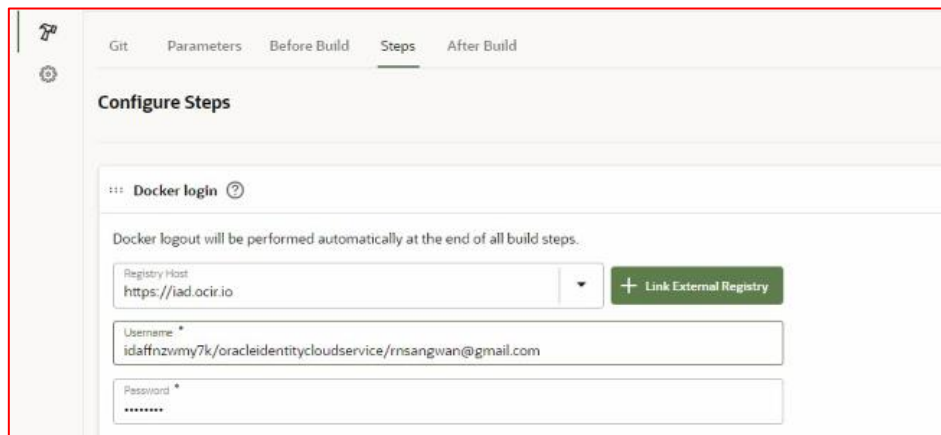24. Give a name to your token and click on Generate

25. It will display the token. Copy it to a notepad file. You will need it in many steps. Fill in the Registry host, user name and password as given in the screen shot. Use your auth token as password.
    User Name Format : ***<Object Storage Namespace>/<Full Username>***
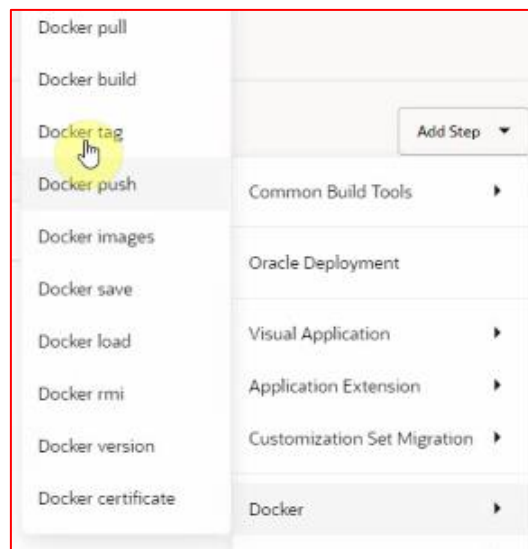
Password *: <Auth Token>*

26. Similarly add another step. Add Step - > Docker -> Docker Build
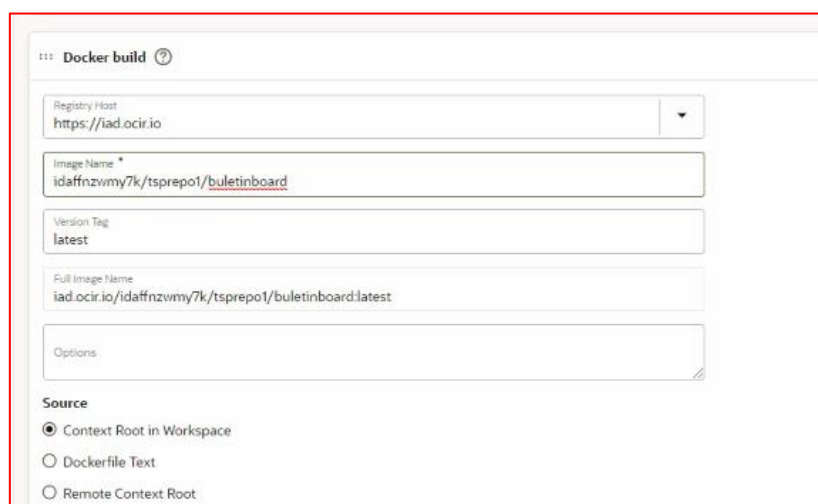
27. Fill in the Docker Build steps as given in the screen shot.
    Registry Host : ***https://<Region Code>.ocir.io***
    Image Name Format : <Object Storage NS>/<OCIR Repo>/<Image Name>

28. Finally add a Docker Push step to push your image to OCIR.

29. Build your image by clicking on "Build Now" Button as given in following screen shot.
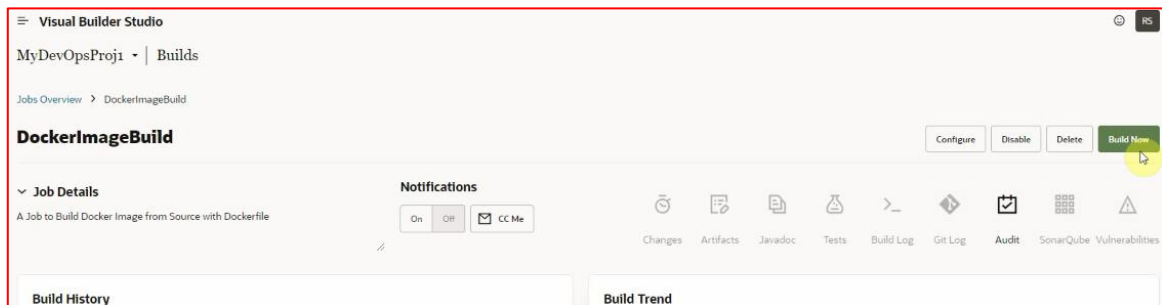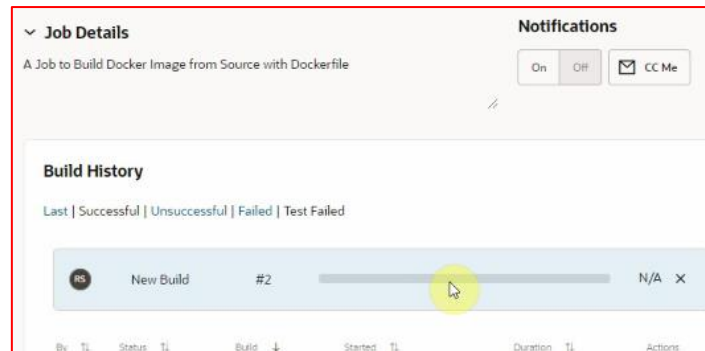
30. Congratulations. Your Build is successful and pushed to OCIR
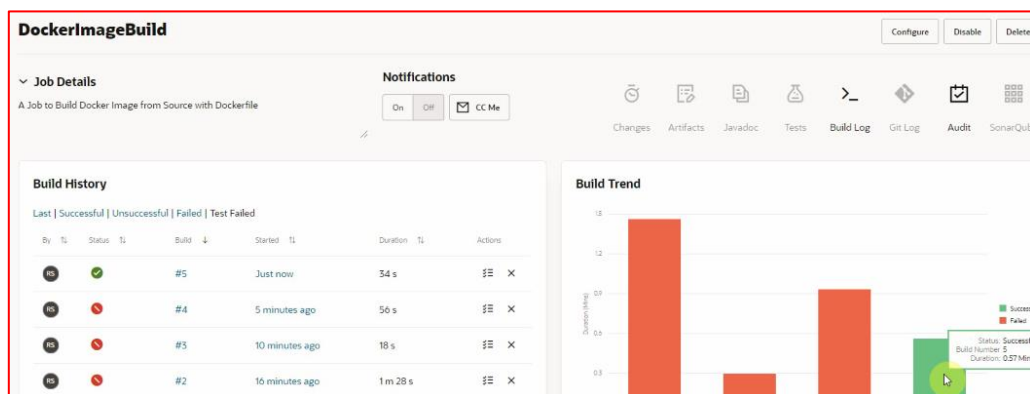
31. From your OCI web console, visit the OCIR page

32. Verify the images in the OCIR

33. Add Developer and EPEL repository to your existing oracle linux repository
    # vi /etc/yum.repos.d/oracle-linux-ol7.repo
    [ol7_developer]
    name=Oracle Linux $releasever Development Packages ($basearch)

```
baseurl=https://yum.oracle.com/repo/OracleLinux/OL7/developer/$basearch/
gpgkey=file:///etc/pki/rpm-gpg/RPM-GPG-KEY-oracle
gpgcheck=1
enabled=1

[ol7_developer_EPEL]
name=Oracle Linux $releasever Development Packages ($basearch)
baseurl=https://yum.oracle.com/repo/OracleLinux/OL7/developer_EPEL/$basearch/
gpgkey=file:///etc/pki/rpm-gpg/RPM-GPG-KEY-oracle
gpgcheck=1
enabled=1
```

```
[ol7_developer]
name=Oracle Linux $releasever Development Packages ($basearch)
baseurl=https://yum.oracle.com/repo/OracleLinux/OL7/developer/$basearch/
gpgkey=file:///etc/pki/rpm-gpg/RPM-GPG-KEY-oracle
gpgcheck=1
enabled=1

[ol7_developer_EPEL]
name=Oracle Linux $releasever Development Packages ($basearch)
baseurl=https://yum.oracle.com/repo/OracleLinux/OL7/developer_EPEL/$basearch/
gpgkey=file:///etc/pki/rpm-gpg/RPM-GPG-KEY-oracle
gpgcheck=1
enabled=1
```

34. Install Docker on your local VM

    # yum install docker -y

```
[root@server ~]# yum install docker
Loaded plugins: langpacks, ulninfo
Resolving Dependencies
--> Running transaction check
---> Package docker-engine.x86_64 0:18.09.1.ol-1.6
--> Processing Dependency: container-selinux >= 2:
--> Processing Dependency: containerd for package:
--> Processing Dependency: runc for package: docke
--> Processing Dependency: docker-cli for package:
--> Running transaction check
```

35. Start Docker daemon and enable it to run at restart.

    # systemctl enable --now docker

```
[root@server ~]# systemctl enable --now docker
Created symlink from /etc/systemd/system/multi-u:
```

36. Login to OCIR
    Username : <Object Storage NS>/<Full User Name>
    Password : <Auth Token>

```
[root@server ~]# docker login iad.ocir.io
Username: idaffnzwmy7k/oracleidentitycloudservice/rnsangwan@gmail.com
Password:
WARNING! Your password will be stored unencrypted in /root/.docker/con
Configure a credential helper to remove this warning. See
https://docs.docker.com/engine/reference/commandline/login/#credential:

Login Succeeded
```

37. Pull the Docker Image from OCIR

```
[root@server ~]# docker pull iad.ocir.io/idaffnzwmy7k/tsprepo1/buletinboard:latest
Trying to pull repository iad.ocir.io/idaffnzwmy7k/tsprepo1/buletinboard ...
latest: Pulling from iad.ocir.io/idaffnzwmy7k/tsprepo1/buletinboard
cae7303ade7f: Pull complete
8dd505804d99: Pull complete
612e79453ef6: Pull complete
512787aab62c: Pull complete
b84cff3b1e06: Pull complete
1da74df5cade: Pull complete
5b2191a42322: Pull complete
9ebf4d6f86af: Pull complete
32b524fc8137: Pull complete
Digest: sha256:82f23bf60608bda1fafa4340e3bd605f4d1bf30ead2b5545d75bc2bf187113fc
Status: Downloaded newer image for iad.ocir.io/idaffnzwmy7k/tsprepo1/buletinboard:latest
[root@server ~]# docker images
REPOSITORY                                          TAG        IMAGE ID        CREATED
iad.ocir.io/idaffnzwmy7k/tsprepo1/buletinboard     latest     5c3960a36af8    14 hours ago
```
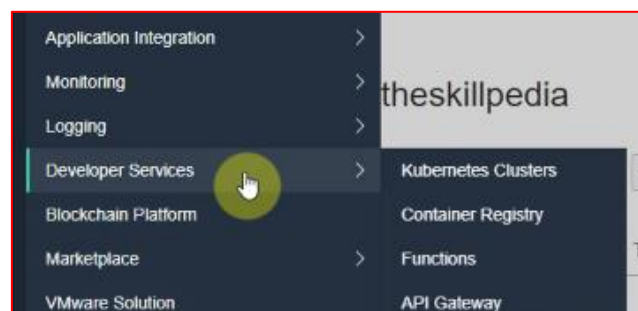
38. Verify the Docker image and Run it in detached mode.

```
[root@server ~]# docker images
REPOSITORY                                          TAG
iad.ocir.io/idaffnzwmy7k/tsprepo1/buletinboard     latest
[root@server ~]# docker run --publish 8000:8080 --detach --name bb i
12be0c6a57817b8c2cd091b54db949607db81b687130fafdc39a77aa76b019b2
[root@server ~]#
```

39. Open the Browser in your host machine and visit http://10.10.0.100:8000. Your Bulletin Board
Website running in container will open