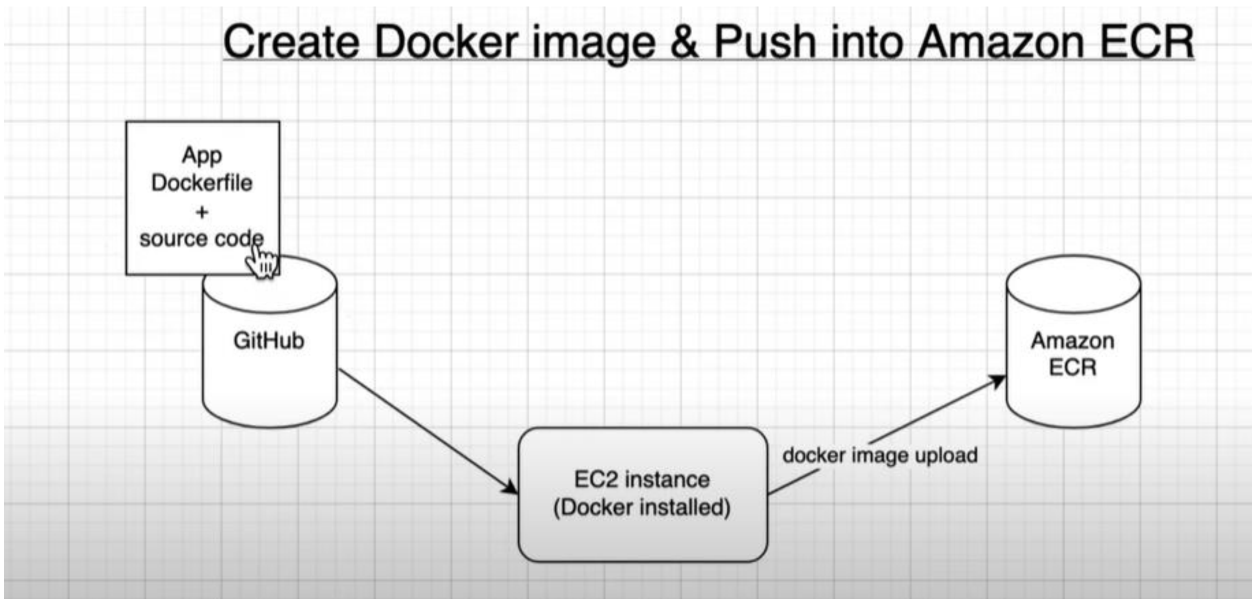
Experienced with using Docker file to create Images, Push created images to Amazon ECR Repository, Create IAM Role and Assign it to Jenkins CI Server for the purpose connecting Jenkins CI server with AWS ECR to allow successful login process, communication and pushing of the images.


Create Docker image & Push into Amazon ECR
App Dockerfile + source code
GitHub
EC2 instance (Docker installed)
docker image upload
Amazon ECR

Experienced in building images using docker file and pushing them to the Elastic Container Registry

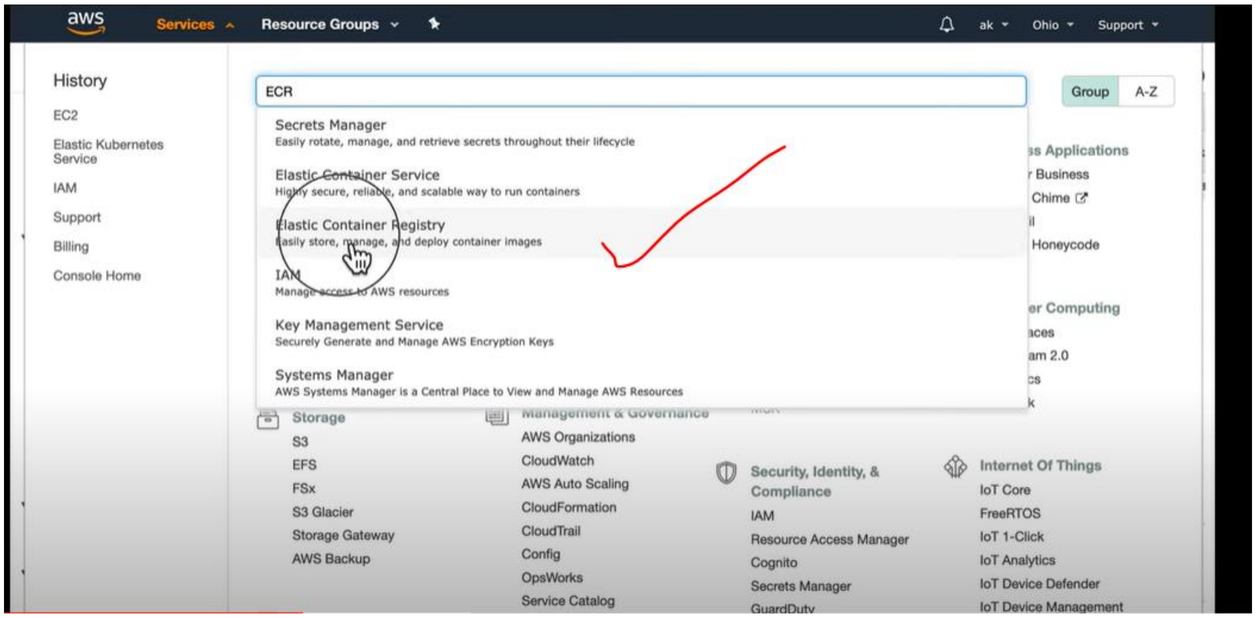
Why Elastic Container Registry

This can auto scale automatically, making this repository attractive to companies that compared to using docker hub which does not scale automatically.

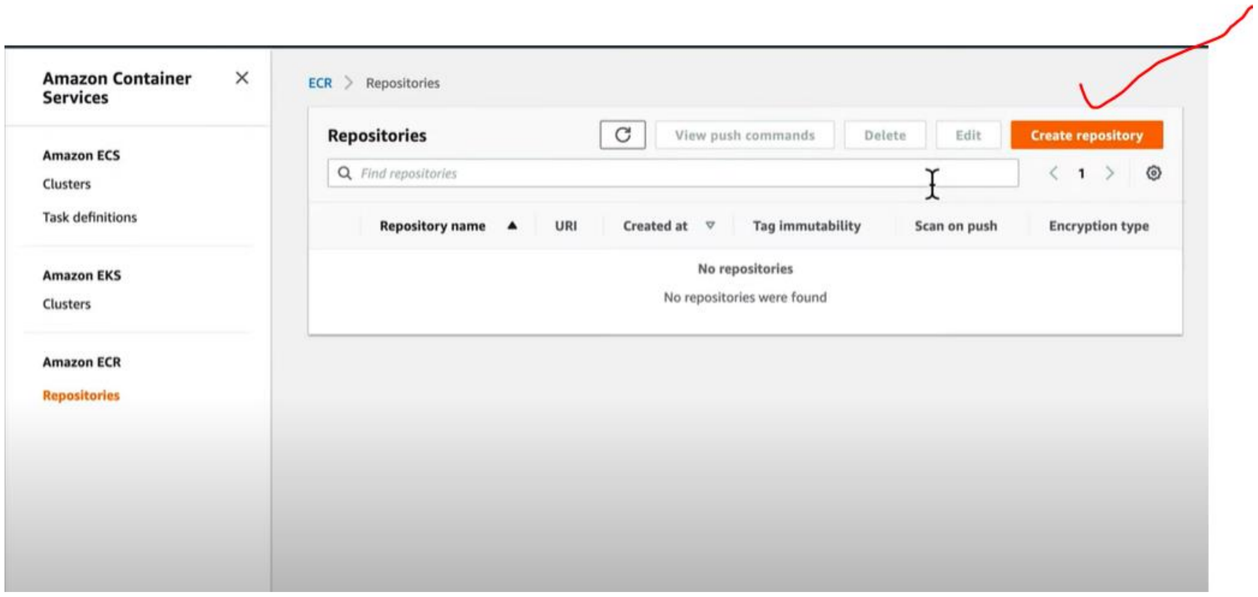
The aim of this project is to build an image and push it to AWS ECR.

Steps

1. Login to your AWS console and create an Elastic Container Registry to store the image that we are going to create.
2. Go to my github and fork the git repo with the link below:
   Link
   https://github.com/joshking1/Creating_Docker_Image_Pushing_to_AWS_ECR.git
3. The git repository contains the docker file which you are going to use to create the image. This is going to give you experience with using docker file to build images, to deploy in Kubernetes as microservices.
4. Go and search for AWS ECR and click on it



5. Click on create repository



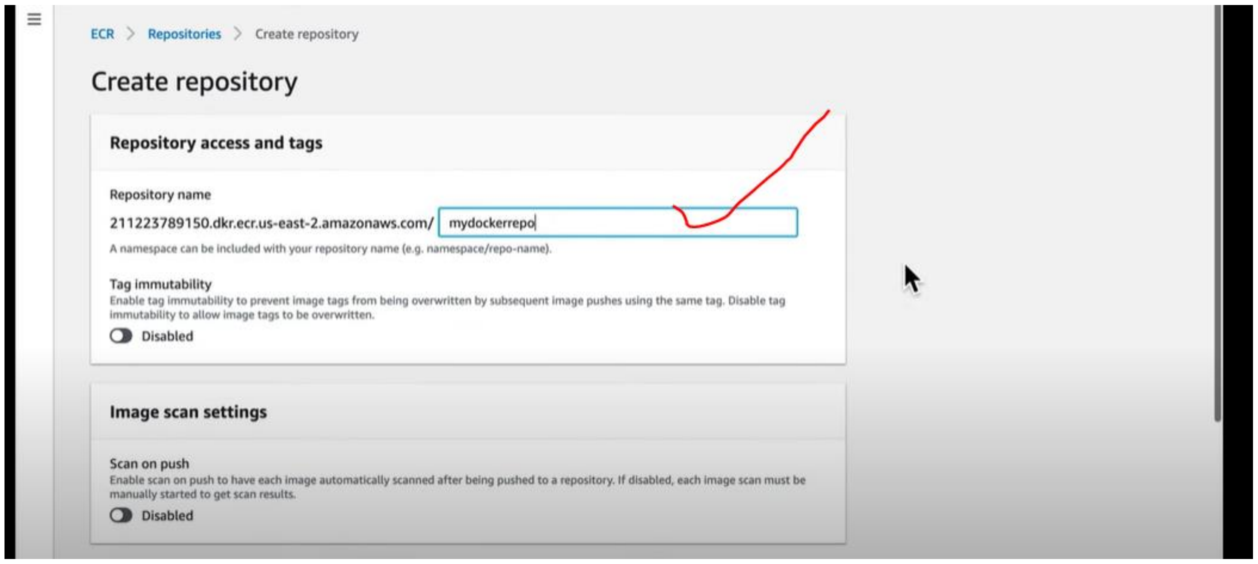
No repositories

No repositories were found

6. This will open a page to allow you to input the information for your ECR

7. Give your repository the name of your choice

   For me, I called it **mydockerrepo**


ECR > Repositories > Create repository
Create repository
Repository access and tags
Image scan settings

Repository name

211223789150.dkr.ecr.us-east-2.amazonaws.com/ mydockerrepo

A namespace can be included with your repository name (e.g. namespace/repo-name).

Tag immutability

Enable tag immutability to prevent image tags from being overwritten by subsequent image pushes using the same tag. Disable tag immutability to allow image tags to be overwritten.

Disabled

Scan on push

Enable scan on push to have each image automatically scanned after being pushed to a repository. If disabled, each image scan must be manually started to get scan results.

Disabled

8. After putting the name of the repo, go and click **Create Repository**

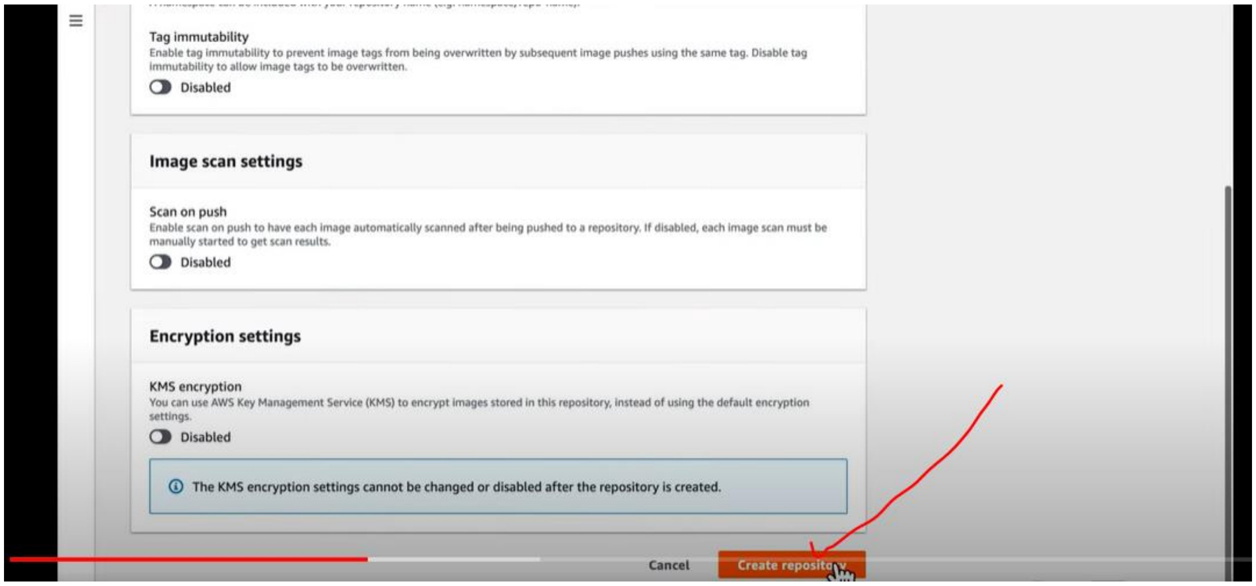
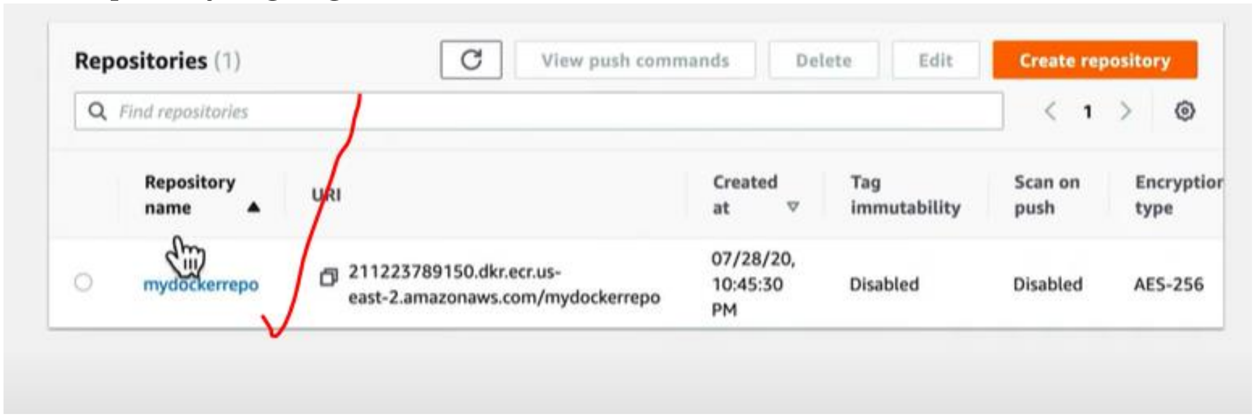

Tag immutability
Enable tag immutability to prevent image tags from being overwritten by subsequent image pushes using the same tag. Disable tag immutability to allow image tags to be overwritten.
Disabled

Image scan settings
Scan on push
Enable scan on push to have each image automatically scanned after being pushed to a repository. If disabled, each image scan must be manually started to get scan results.
Disabled

Encryption settings
KMS encryption
You can use AWS Key Management Service (KMS) to encrypt images stored in this repository, instead of using the default encryption settings.
Disabled

The KMS encryption settings cannot be changed or disabled after the repository is created.

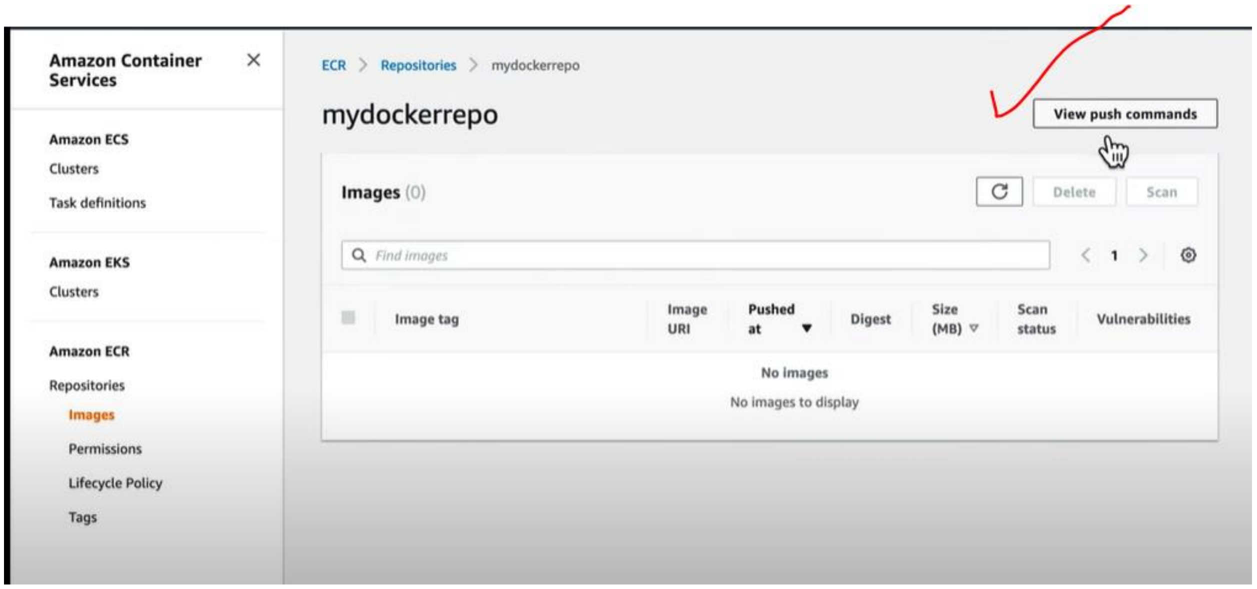
9. Your repository is going to be created



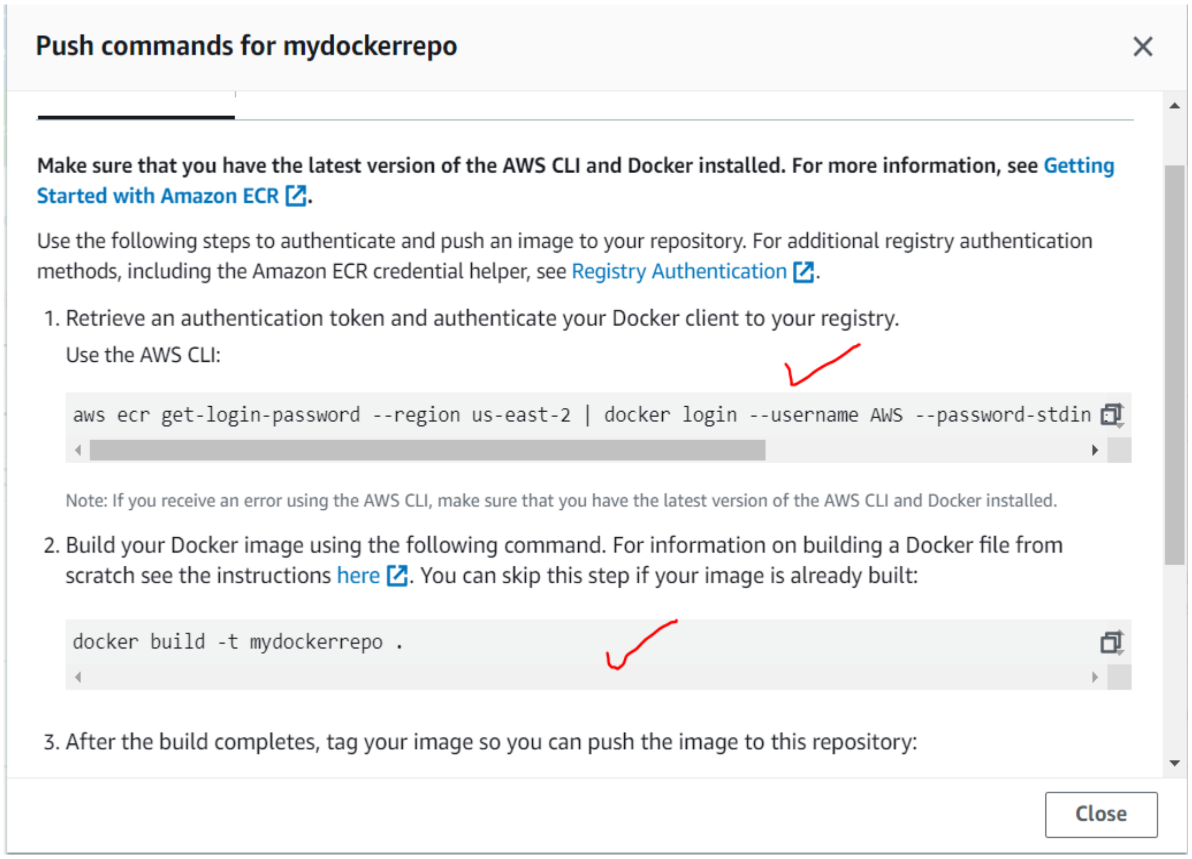
Repository name	URI	Created at	Tag immutability	Scan on push	Encryption type
mydockerrepo	211223789150.dkr.ecr.us-east-2.amazonaws.com/mydockerrepo	07/28/20, 10:45:30 PM	Disabled	Disabled	AES-256

10. Click on the repository name and should be empty because there is no image yet

11. Click on the repository again and this time go to the top right-hand side, and you should see **VIEW PUSH COMMANDS**



12. View Push Command



Push commands for mydockerrepo

Make sure that you have the latest version of the AWS CLI and Docker installed. For more information, see Getting Started with Amazon ECR.

Use the following steps to authenticate and push an image to your repository. For additional registry authentication methods, including the Amazon ECR credential helper, see Registry Authentication.

1. Retrieve an authentication token and authenticate your Docker client to your registry.
   Use the AWS CLI:

```
aws ecr get-login-password --region us-east-2 | docker login --username AWS --password-stdin
```

   Note: If you receive an error using the AWS CLI, make sure that you have the latest version of the AWS CLI and Docker installed.

2. Build your Docker image using the following command. For information on building a Docker file from scratch see the instructions here. You can skip this step if your image is already built:

```
docker build -t mydockerrepo .
```

3. After the build completes, tag your image so you can push the image to this repository:

13. Create an IAM role for the Jenkins ec2 to have full access to the ECR.

IAM role name – Amazonelasticcontainerregistryfullaccess

Go back to instance dashboard and assign the created IAM role to Jenkins's instance.

This is very important. Do not skip this step because you will not be able to connect to ECR from Jenkins terminal.

IAM Role vs. Access Key ID and Secret Access key (Do some research on the below bulleted questions)

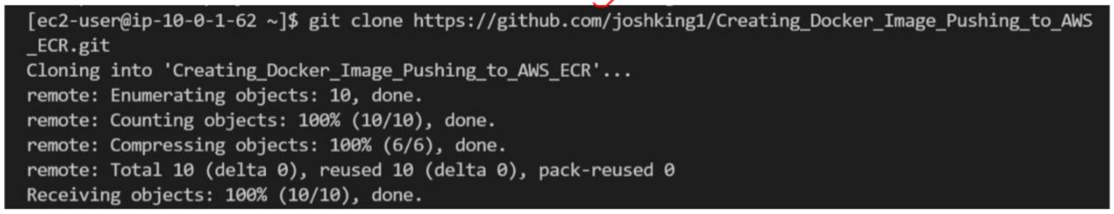
- When do you use IAM role?
- When do you use Access Key ID and Secret Access key?
- When do you use the key name or key pair?

14. After creating the repository, ssh to your Jenkins CI server.

On the Jenkin's terminal, execute the following

# git clone https://github.com/joshking1/Creating_Docker_Image_Pushing_to_AWS_ECR.git

Git Clone ✓



```
[ec2-user@ip-10-0-1-62 ~]$ git clone https://github.com/joshking1/Creating_Docker_Image_Pushing_to_AWS
_ECR.git
Cloning into 'Creating_Docker_Image_Pushing_to_AWS_ECR'...
remote: Enumerating objects: 10, done.
remote: Counting objects: 100% (10/10), done.
remote: Compressing objects: 100% (6/6), done.
remote: Total 10 (delta 0), reused 10 (delta 0), pack-reused 0
Receiving objects: 100% (10/10), done.
```

15. ls

Also

# cd to the folder

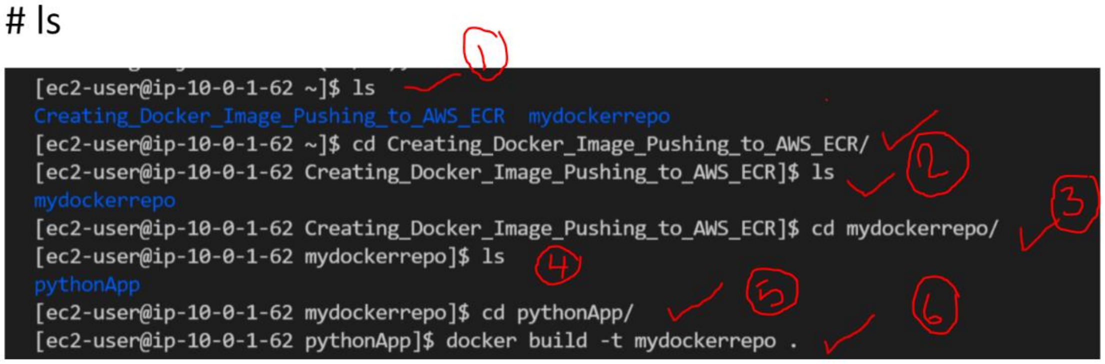
# ls - second time

# cd mydockerrepo/

Build the image by exceuting the following command

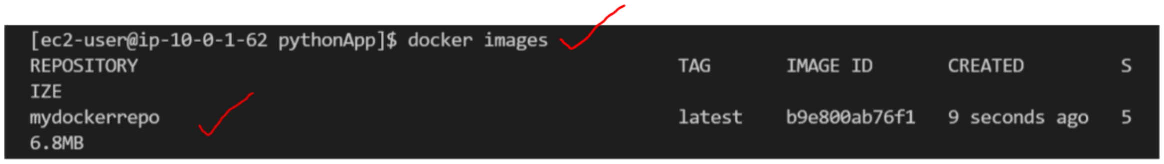
# docker build -t mydockerrepo .

ls



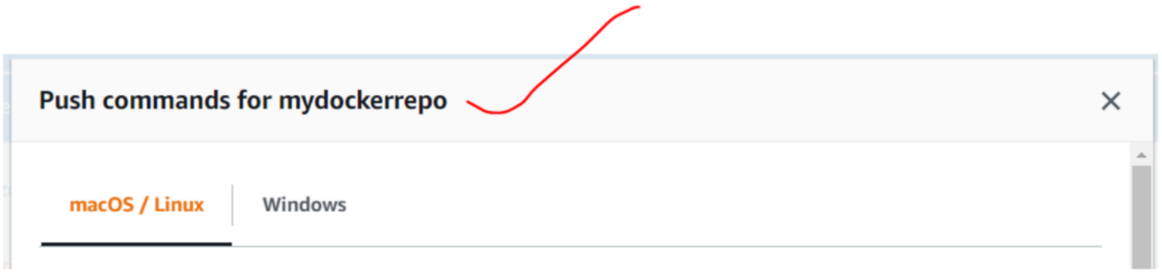
```
[ec2-user@ip-10-0-1-62 ~]$ ls
Creating_Docker_Image_Pushing_to_AWS_ECR  mydockerrepo
[ec2-user@ip-10-0-1-62 ~]$ cd Creating_Docker_Image_Pushing_to_AWS_ECR/
[ec2-user@ip-10-0-1-62 Creating_Docker_Image_Pushing_to_AWS_ECR]$ ls
mydockerrepo
[ec2-user@ip-10-0-1-62 Creating_Docker_Image_Pushing_to_AWS_ECR]$ cd mydockerrepo/
[ec2-user@ip-10-0-1-62 mydockerrepo]$ ls
pythonApp
[ec2-user@ip-10-0-1-62 mydockerrepo]$ cd pythonApp/
[ec2-user@ip-10-0-1-62 pythonApp]$ docker build -t mydockerrepo .
```

# docker build -t mydockerrepo .



```
[ec2-user@ip-10-0-1-62 pythonApp]$ docker images
REPOSITORY                                        TAG       IMAGE ID       CREATED          S
IZE
mydockerrepo                                      latest    b9e800ab76f1   9 seconds ago    5
6.8MB
```

Go to



Push commands for mydockerrepo

Login Command to AWS ECR

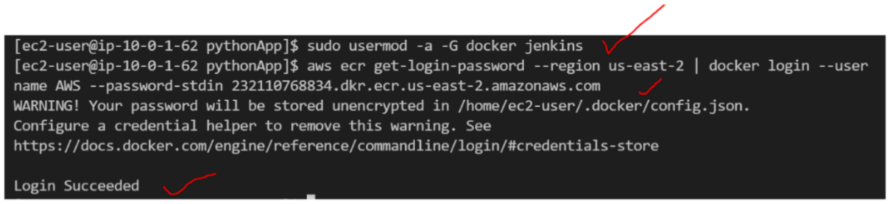
Login to AWS ECR from Jenkins CI server using the push commands

Push commands

Follow all the push commands until you push your image to the Amazon ECR.

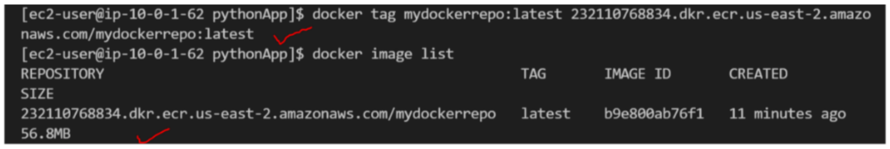
- # aws ecr get-login-password --region us-east-2 | docker login --username AWS --password-stdin 325864094195.dkr.ecr.us-east-2.amazonaws.com
- # docker build -t mydockerrepo .
- # docker tag mydockerrepo:latest 325864094195.dkr.ecr.us-east-2.amazonaws.com/mydockerrepo:latest

- # docker push 325864094195.dkr.ecr.us-east-2.amazonaws.com/mydockerrepo:latest



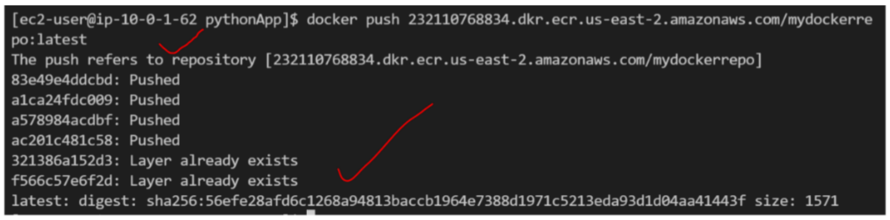
```
[ec2-user@ip-10-0-1-62 pythonApp]$ sudo usermod -a -G docker jenkins
[ec2-user@ip-10-0-1-62 pythonApp]$ aws ecr get-login-password --region us-east-2 | docker login --user
name AWS --password-stdin 232110768834.dkr.ecr.us-east-2.amazonaws.com
WARNING! Your password will be stored unencrypted in /home/ec2-user/.docker/config.json.
Configure a credential helper to remove this warning. See
https://docs.docker.com/engine/reference/commandline/login/#credentials-store

Login Succeeded
```

Tag the image build



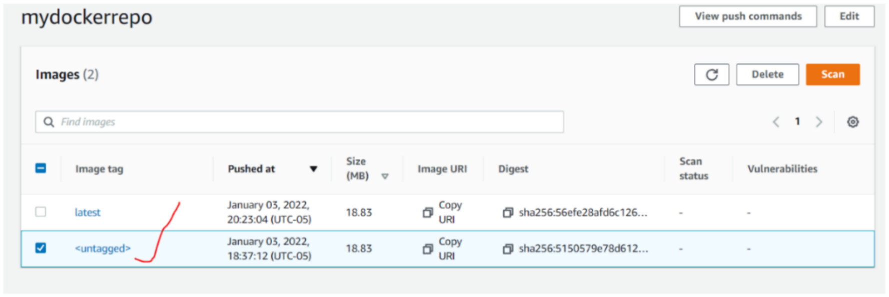
```
[ec2-user@ip-10-0-1-62 pythonApp]$ docker tag mydockerrepo:latest 232110768834.dkr.ecr.us-east-2.amazo
naws.com/mydockerrepo:latest
[ec2-user@ip-10-0-1-62 pythonApp]$ docker image list
REPOSITORY                                                TAG       IMAGE ID       CREATED
SIZE
232110768834.dkr.ecr.us-east-2.amazonaws.com/mydockerrepo   latest    b9e800ab76f1   11 minutes ago
56.8MB
```

Docker push



```
[ec2-user@ip-10-0-1-62 pythonApp]$ docker push 232110768834.dkr.ecr.us-east-2.amazonaws.com/mydockerre
po:latest
The push refers to repository [232110768834.dkr.ecr.us-east-2.amazonaws.com/mydockerrepo]
83e49e4ddcbd: Pushed
a1ca24fdc009: Pushed
a578984acdbf: Pushed
ac201c481c58: Pushed
321386a152d3: Layer already exists
f566c57e6f2d: Layer already exists
latest: digest: sha256:56efe28afd6c1268a94813baccb1964e7388d1971c5213eda93d1d04aa41443f size: 1571
```

Go to AWS and check your ECR



mydockerrepo

Images (2)

	Image tag	Pushed at	Size (MB)	Image URI	Digest	Scan status	Vulnerabilities
☐	latest	January 03, 2022, 20:23:04 (UTC-05)	18.83	Copy URI	sha256:56efe28afd6c126...	-	-
☑	<untagged>	January 03, 2022, 18:37:12 (UTC-05)	18.83	Copy URI	sha256:5150579e78d612...	-	-